AI-Powered System-Scientific Defense for High-Confidence Cyber-Physical Systems: Modeling, Analysis, and Design

DISSERTATION

Submitted in Partial Fulfillment of

the Requirements for

the Degree of

DOCTOR OF PHILOSOPHY (Electrical Engineering)

at the

NEW YORK UNIVERSITY
TANDON SCHOOL OF ENGINEERING

by

Linan Huang

May 2022

AI-Powered System-Scientific Defense for High-Confidence Cyber-Physical Systems: Modeling, Analysis, and Design

DISSERTATION

Submitted in Partial Fulfillment of

the Requirements for

the Degree of

DOCTOR OF PHILOSOPHY (Electrical Engineering)

at the

NEW YORK UNIVERSITY
TANDON SCHOOL OF ENGINEERING

by

Linan Huang

May 2022

Approved:

Ivan Selesnick

Department Chair Signature

May 9, 2022

Date

University ID: N13447341
Net ID: lh2328

Approved by the Guidance Committee:

Major: Electrical Engineering

Quanyan Zhu
Associate Professor of
Electrical and Computer Engineering

5/9/2022
Date

Zhong-Ping Jiang
Professor of
Electrical and Computer Engineering

5/8/2022
Date

Nasir Memon

Nasir Memon
Professor of
Computer Science and Engineering

5/9/22
Date

Microfilm or other copies of this dissertation are obtainable from

UMI Dissertation Publishing
ProQuest CSA
789 E. Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

Vita

Linan Huang was born in Luoyang, Henan, China. He attended Tsinghua University High School from 2006 to 2009 and from 2009 to 2012 for junior high school and high school, respectively. After receiving the B.Eng. degree in electrical engineering from the Beijing Institute of Technology (BIT) in 2016, he entered the doctoral program in electrical engineering at New York University (NYU) Tandon School of Engineering. He was the recipient of the **Ernst Weber fellowship** from the Department of Electrical and Computer Engineering (ECE) to support his studies and research and was advised by Professor Quanyan Zhu at the Laboratory for Agile and Resilient Complex Systems (LARX). During his Ph.D., he was awarded the **best student paper award** at the 2021 Conference on Decision and Game Theory for Security (GameSec 2021) and the **2022 Dante Youla award** for research excellence by the NYU ECE department.

His research interests are broadly in dynamic decision-making of multi-agent systems, mechanism design, artificial intelligence, security, and resilience for the cyber-physical systems.

Acknowledgements

As I approach the end of my Ph.D. journey, I feel genuinely thankful to spend the most challenging and fulfilling six years of my life under the guidance of my advisor, Prof. Quanyan Zhu. It is extremely rewarding and enjoyable working with him on a wide variety of projects, and I am deeply impressed by his enthusiasm, wisdom, creativity, and leadership. He is a great educator who devotes his heart and soul to his students and is truly delighted in their growths and successes. His constructive suggestions have helped me overcome my insecurity, shape my thinking, and see the beauty and truth that people may not see. He is my role model, and I still have so much to learn from him.

I would also like to thank my guidance committee members, Prof. Nasir Memon and Prof. Zhong-Ping Jiang, for their precious time reviewing my dissertation and providing insightful feedback. I am indebted to all my collaborators for their support and fruitful discussions. In particular, I would like to thank Dr. Yunfei Zhao and Dr. Carol Smidts from Ohio State University, Dr. Emily Balcetis from New York University, and many others whom I have worked with. Furthermore, I feel genuinely honored to work in a group with great minds and thank all the group members in the Laboratory for Agile and Resilient Complex Systems (LARX). Special thanks to my 21 roommates in apartment 2204 who have been family to me and all my friends who have made my life in New York an unforgettable experience.

My deepest thanks are to my parents for their unconditional support, love, and trust. They have provided me with everything in my life and been the best parents that one can get.

Linan Huang

May 2022

ABSTRACT

AI-Powered System-Scientific Defense for High-Confidence Cyber-Physical Systems: Modeling, Analysis, and Design

by

Linan Huang

Advisor: Prof. Quanyan Zhu, Ph.D.

Submitted in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy (Electrical Engineering)

May 2022

Cyber-Physical Systems (CPSs) are smart systems that include engineered interacting networks of physical, digital, and computational components. It is critical to build high-confidence CPSs that behave in a well-understood, predictable, and justifiably trusted fashion to fulfill life-critical tasks such as medical surgery, autonomous driving, and nuclear power plant control. The status quo, however, is remote from this objective due to the challenges that arise from the distinctive features of CPSs, e.g., diversity and heterogeneity, large-scale connections and

complex interdependence, human-in-the-loop, and openness and dynamic properties. Environmental (e.g., cascading failures), accidental (e.g., human errors), and deliberate threats (e.g., advanced persistent threats) have attested to the inadequacy of the off-the-shelf defense mechanisms.

This dissertation focuses on developing Defense through AI-powered SYstem-scientific methods (DAISY) for high-confidence CPSs. To this end, the dissertation starts by delineating a brief history of security technologies by designating five generations of Security Paradigms (SPs) that have evolved since the birth of the Internet. They include the first-generation SP (1G-SP) of laissez-faire security, the 2G-SP of perimeter security, the 3G-SP of reactive security, the 4G-SP of proactive security, and the 5G-SP of federated security. DAISY addresses central challenges of the current security landscape, including lack of security standards and metrics, human-targeted and human-induced attacks, strategic and intelligent attacks, imperfect security, piecemeal design of CPS, and defenders' time, space, information, and cooperation disadvantages. Positioned as the foundation of 5G-SP, DAISY enables the following six dimensions of the evolution of security solutions, i.e., from empirical to theoretical, from technical to socio-technical, from single-agent to multi-agent, from secure to resilient, from add-on to built-in, and from reactive to proactive security.

This dissertation develops system-scientific modeling and design frameworks to achieve the 5G-SP objectives. We organize the contributions in this dissertation into three parts in accordance with the three types of vulnerabilities that DAISY aims to mitigate, i.e., posture-related vulnerabilities in Part II, information-related vulnerabilities in Parts III and IV, and human-related vulnerabilities in Parts V and VI, respectively. Part II mitigates the defender's resource disadvantage to protect from known and unknown vulnerabilities on the attack surface. Part III and Part IV aim to tilt the information asymmetry (i.e., the attacker has more information about the defender than the other way around) by undermining the attacker's information advantage (i.e., counteracting adversarial deception) and establishing information advantage for the defender (i.e., designing defensive deception), respectively. Following the definition that acquired (resp. innate) human vulnerability can (resp. cannot) be mitigated through short-term security training and awareness programs, Part V and Part VI mitigate the acquired vulnerability of incentive misalignment and the innate vulnerability of bounded attention, respectively.

This dissertation bridges several research fields, including game theory, data science, socio-economic sciences, and cybersecurity, which are accustomed to their individual advances in silos. Its contributions have a multitude of impacts in both theory and practice. First, the established models and frameworks enable quantitative design, circumvent laborious and expensive trial-and-error design procedures, and empower the design of high-confidence CPSs. Second, leveraging a broad range of system science tools and AI (e.g., control and game theory, information design, and learning theory), this dissertation lays solid theoretical foundations to characterize fundamental limits and tradeoffs, discover security principles and laws, and design strategic security mechanisms. Third, we develop efficient and scalable algorithms to create implementable technologies and built-in defense for high-confidence CPSs. Finally, this dissertation contributes to a large number of critical CPS application fields, including resilient interdependent critical infrastructure networks in Chapter 3, secure nuclear power plants in Chapter 4, attack-aware manufacturing systems in Chapter 5, deception-resistant robotics in

Chapter 6, honeypot-driven security intelligence in Chapters 7 and 8, insider threat mitigation in Chapters 9 and 10, and human-machine interactions in Chapters 11 and 12.

This dissertation bridges science and technology to create provable and implementable solutions that accelerate the development of high-confidence CPSs. The proposed methodologies are universal for a broad class of problems, and the insights from one problem are transferable to another. These insights lead to a rich volume of future work and are promising in pushing the boundaries of the current research to encompass more impactful real-world applications. This dissertation is the epitome of system-thinking that integrates the concepts of feedback, tradeoff, equilibrium, and data. It canvasses perspectives that rise above the traditional realm of engineering and create several concomitant impacts in related fields, such as human factors engineering and meta-system theory.

Contents

VI Hodatology for Cognitive Security 360

11 ADVERT: An Attention Enhancement Mechanism for Phishing Prevention 361

Notations

This dissertation includes a wide breadth of models and applications. Therefore, the mathematical notation is flexible. Each notation will have a specific definition in each chapter, and the definition of each notation might vary from chapters to chapters. Nevertheless, we present some consistent styles here.

We use $\mathbb{R}$ and $\mathbb{Z}$ to represent spaces of real numbers and integers, respectively, where $\mathbb{R}^n$ represents a Euclidean space of dimension n. The Euclidean norm of a vector x is represented by $||x||_2$. Calligraphic letter, e.g., $\mathcal{A}$, defines a set, and $|\mathcal{A}|$ represents its cardinality. Define $\mathcal{B} \setminus \mathcal{A}$ as the set of elements in $\mathcal{B}$ but not in $\mathcal{A}$. Define $\Delta(\mathcal{A})$ as the set of probability distributions over $\mathcal{A}$. Let $f : \mathcal{A} \mapsto \mathcal{B}$ be a function or a mapping from set $\mathcal{A}$ to set $\mathcal{B}$. Define $\{a_i\}_{i\in\mathcal{N}} := \{a_1, \cdots, a_N\}$, $[a_i]_{i\in\mathcal{N}} := [a_1, \cdots, a_N]$, and $(a_i)_{i\in\mathcal{N}} := (a_1, \cdots, a_N)$ as a set, a vector, and a tuple of N elements, respectively. We use Pr and $\mathbb{E}$ to represent probability and expectation, respectively, where $\mathbb{E}_{a\sim A}[f(a)]$ denotes the expectation of $f(a)$ over random variable a whose probability distribution is A. The notation $A := B$ means that A is defined as B. Let $\mathbf{1}_{A=B}$ be an indicator function which equals one when $A = B$ and zero otherwise. For $\theta_i \in \Theta_i, i \in \{1, 2, \cdots, I\}$, let $\theta_{-i} := \{\theta_j\}_{j\in\mathcal{I}\setminus\{i\}}$ and $\Theta_{-i} := \prod_{j\in\mathcal{I}\setminus\{i\}} \Theta_j$, then $\theta_{-i} \in \Theta_{-i}$.

A piece of information for a group of players is called *common knowledge* if all players know it, all players know that all players know it, and so on ad infinitum.

List of Figures

List of Tables

Acronyms

AI Artificial Intelligence. 2, 19, 24, 26, 41, 432, 440, 443, 444

ALP Approximate Linear Program. 83, 85, 88, 93, 94, 97–99

APT Advanced Persistent Threat. 10, 13, 14, 17–19, 23, 29, 33, 40, 41, 52, 156–159, 172–174, 178, 183, 233, 235, 241, 270

BCE Bayesian Correlated Equilibrium. 51, 60–62

CIN Critical Infrastructure Network. 2, 5–9, 11, 14, 36, 38–40, 71, 72, 79, 82–84, 99

CPS Cyber-Physical System. 2, 3, 6–11, 13, 15, 19, 23, 25–28, 30–35, 38, 39, 45, 46, 49, 63–68, 156, 192, 430, 437, 439, 440, 442–444, 446

DAISY Defense through AI-powered SYstem-scientific methods. 19, 24, 25, 39, 45, 430, 439

DiD Defense in Depth. 23, 156, 167, 441

DMZ Demilitarized Zone. 21, 22, 237, 239, 246, 255, 441

DoS Denial-of-Service. 20, 31, 391, 392

HMI Human Machine Interface. 3, 4, 7, 9, 33

ICS Industrial Control System. 2, 3, 6–9, 14, 103, 157, 394–397, 399, 401, 417

ICT Information and Communication Technology. 4, 5, 105

IDoS Informational Denial-of-Service. 40, 43, 44, 389–399, 402, 404, 407, 408, 410, 411, 413–418, 420–422, 424–426, 438

IDS Intrusion Detection System. 13, 22–24, 158, 259, 390, 394, 398, 399

IoC Indicator of Compromise. 42, 243, 258–260, 269

IoT Internet of Things. 4, 5, 14

IPS Intrusion Prevention System. 21–24

IRS Intrusion Response System. 22

LHS Left-Hand Side. 82, 89, 208

LP Linear Program. 62, 80–83, 85, 90, 91, 96–100, 299, 336

MDP Markov Decision Process. 63–65, 69, 267, 268, 431, 435, 436

MTD Moving Target Defense. 23, 24, 31, 48, 66, 441

NE Nash Equilibrium. 29, 50, 53–55, 111, 118, 119, 121, 122, 126, 136, 142, 151, 152, 442

PBNE Perfect Bayesian Nash Equilibrium. 38, 51, 53, 58, 59, 164, 165, 167, 171, 172, 199, 200, 205, 206, 209–211, 432

PLC Programmable Logic Controller. 4, 7, 33, 55, 105, 130, 417

PRA Probabilistic Risk Assessment. 104, 105, 114–117, 124, 131, 139–141, 431

RHS Right-Hand Side. 75, 81, 90, 91, 207, 208

RL Reinforcement Learning. 16, 17, 25, 29, 39, 46, 63–66, 69, 276, 434–436, 438, 444

RTU Remote Terminal Unit. 3, 4, 7

SCADA Supervisory Control and Data Acquisition. 3, 4, 6, 7, 33, 105, 172, 173, 178, 181

SMDP Semi-Markov Decision Process. 36, 69, 107, 261, 262, 267, 270, 276–278, 436

SMG Semi-Markov Game. 36, 69, 104, 106–109, 114, 115, 118, 119, 121, 124, 125, 133, 135, 431

SOC Security Operation Center. 16, 28, 325, 328, 329, 351–359, 394, 395, 399, 400, 404, 407, 409, 417, 418

SoS System of Systems. 10, 46, 71, 431, 446

SP Security Paradigm. 19–24, 30, 31, 430, 437, 439, 442

SPNE Subgame Perfect Nash Equilibrium. 53, 56

URL Uniform Resource Locator. 12, 16, 174

Part I

Motivation and Framework

Chapter 1

Introduction to High-Confidence Cyber-Physical Systems

The National Institute of Standards and Technology (NIST) in 2017 has defined Cyber-Physical Systems (CPSs) as "*smart systems that include engineered interacting networks of physical and computational components*" [67]. In Section 1.1, we illustrate the integration of cyber and physical layers in different application domains, where Industrial Control Systems (ICSs) and Critical Infrastructure Networks (CINs) are provided as two classical and essential CPS applications. Using these two CPS applications as running examples, we identify four distinctive features of CPS in Section 1.2 and the necessity of *high-confidence* CPS in Section 1.3. After identifying threats and vulnerabilities, we introduce AI-powered system-scientific defense mechanisms in Section 1.4 to build high-confidence CPS. Finally, we summarize our contributions and the dissertation organization in Section 1.5 and 1.6, respectively.

1.1 Applications of CPSs

By integrating computation, communication, sensing, and actuation with physical objects and infrastructures, CPSs enable varying degrees of interactions with the environment and humans to fulfill *time-sensitive* and *mission-critical* tasks. The technological advances in CPSs have been applied within and across multiple “smart” application domains, including smart manufacturing, transportation, energy, and healthcare. In Sections 1.1.1 and 1.1.2, we select two typical CPS applications to illustrate the seamless integration of the varied cyber layers and physical layers, the interdependence within and across layers, and the resulting new features and functionalities.

1.1.1 Industrial Control Systems

Industrial Control System (ICS) is a collective term to describe different types of control systems and associated instrumentation for automated industrial processes. ICS plays critical roles in nearly every industrial sector and critical infrastructure such as the manufacturing, transportation, energy, and water treatment industries [204]. Among varied types, the most common one is the Supervisory Control and Data Acquisition (SCADA) system as shown in Fig. 1.1.

The control center components include data historians, Human Machine Interfaces (HMIs), workstations, and control servers (e.g., Master Terminal Units (MTUs)). These components are connected by a Local Area Network (LAN) and treated as the cyber layer. The control center is mainly responsible for centralized alarming, trend analyses, and reporting.

The field site components include low-level control devices (e.g., Remote Termi-

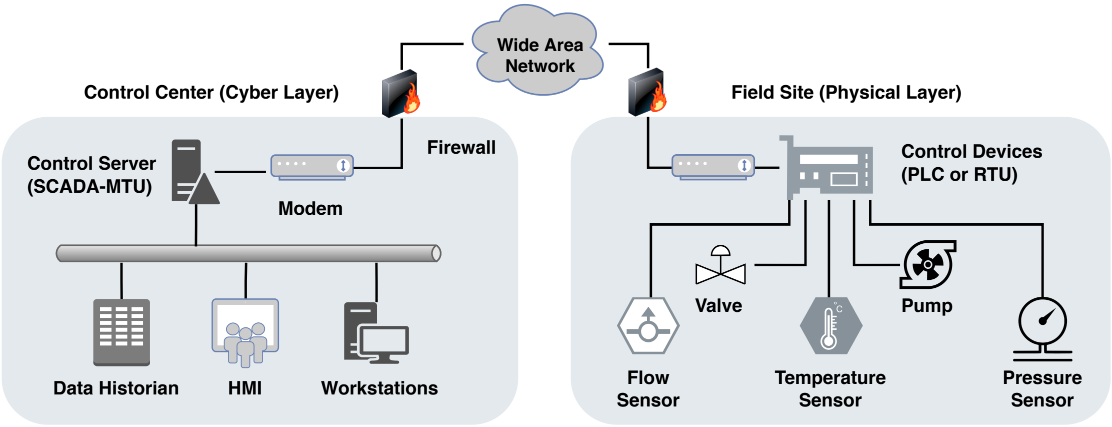


Figure 1.1: The components and structure of a typical SCADA system that consists of a control center and multiple field sites over a WAN. The control center collects real-time control signals and sensor data at different field sites, displays information via the HMIs, and generates responsive actions autonomously.

nal Units (RTUs) and Programmable Logic Controllers (PLCs)), actuators (e.g., valves and pumps), and sensors. These components are also connected by a LAN and treated as the physical layer. Depending on industrial applications, there are usually multiple field sites to perform a local control of actuators and monitor sensors at different physical locations. The components of the control center and these field sites communicate over a Wide Area Network (WAN), where firewalls are adopted at the network boundaries to monitor and filter incoming and outgoing network traffic based on pre-established security rules.

1.1.2 Critical Infrastructure Networks

Presidential Policy Directive 21 (PPD-21) identifies 16 critical infrastructure sectors, including chemical, food, water, nuclear, healthcare, energy, communication and transportation systems [161]. Driven by the recent advances in Information and Communication Technologies (ICTs), cloud computing, and the Internet of Things

(IoT), these sectors become highly interconnected and interdependent at multiple levels [164], enabling faster information exchange and a higher level of situational awareness for real-time operations. We examine in detail the cyber-physical realm of Critical Infrastructure Networks (CINs) in Fig. 1.2.

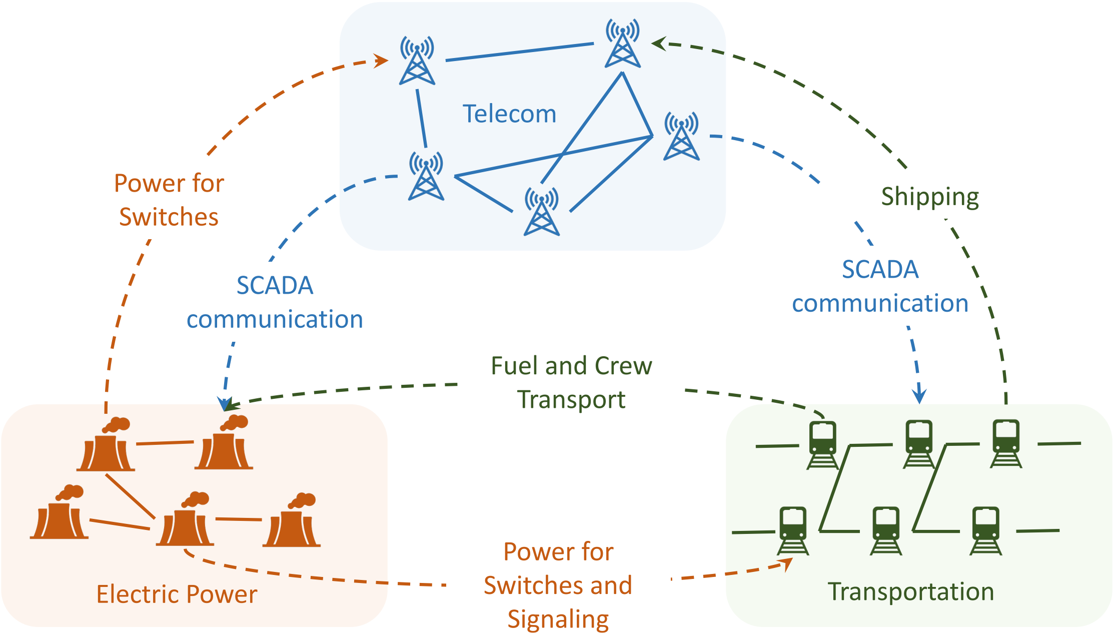


Figure 1.2: The cyber-physical realm of CINs such as the power, transportation, and communication networks. These interdependent infrastructures can be viewed as a large-scale aggregated network that forms the physical layer. The ICTs, cloud computing, and the IoT are integrated to provide surveillance, storage, computation, and communication services to these critical infrastructure sectors.

The physical layer includes different infrastructure sectors that rely on each other to enhance their overall performance and provide essential services. For example, the communication network provides control signals for subway dispatch and power generation; the energy sector provides the power to guarantee the normal operation of the communication and subway networks; the transportation network provides commute service for the workers and employees in the communication and power networks. The cyber layer (e.g., data centers to store and analyze data collected in communication and power networks) expands the capacity of infrastructure sectors.

The authors in [179] have defined *physical*, *cyber*, *geographic*, and *logical* interdependence as the four principal classes of interdependencies in CINs. A physical interdependence (e.g., shipping and power supplies) arises from a physical linkage between the inputs and outputs of two agents while a cyber interdependence (e.g., SCADA communication) is the result of information transmitted through the information infrastructure. Infrastructures are geographically interdependent if one infrastructure affects another due to spatial proximity.For example, an explosion of a power station may disable transportation services within its proximity. Other types of interdependence relationships are logical. For example, the explosion of a power station may lead to the irrational panic of crowds and result in traffic congestion in a town remote from the station.

1.2 Distinctive Features of CPSs

In Section 1.2, we identify the following four major features of CPSs and elaborate on them using the ICS and CIN examples in Section 1.1.

1.2.1 Diversity and Heterogeneity

In Section 1.1, we have touched upon the diversity of CPSs based on their functionalities and applications. Within each CPS, heterogeneous components that are manufactured, specified, or implemented by different entities are eventually composed in varied ways. Take the ICS in Fig. 1.1 as an example, there are different hardware components such as sensors (e.g., flow, temperature, and pressure monitors), actuators (e.g., valves and pumps), control devices (e.g., PLCs and RTUs), and computers of different functionalities (e.g., control servers, data historians, and workstations). There are also different collections of software products for control, communication, and monitoring [102]. The diverse CPSs and their components result in heterogeneous attributes.

1.2.2 Large-Scale and Complex Interdependence

Manufacturers have improved CPSs by adding services that rely on open networks and wireless technologies that enable large-scale interconnections and complex interdependence. In the ICS example in Fig. 1.1, one control center can control multiple field sites remotely and simultaneously. These field sites can be located far apart and assigned different non-synchronized tasks. The CIN example in Fig. 1.2 also illustrates the cyber (e.g., SCADA communication), physical (e.g., shipping and power supplies), geographical (e.g., explosions in spatial proximity), and logical interdependence (e.g., human decisions) across and within different CPS networks.

1.2.3 Human-In-The-Loop

One indispensable element of a CPS is humans, including network administrators, users, and field operators. Human-In-The-Loop (HITP) integrates human cognition with the increased level of autonomous capabilities of CPSs and has the potential to handle complex tasks in unstructured environments [64]. The ICS example in Fig. 1.1 shows that the HMI plays a critical role in generating responsive actions and managing alerts. The CIN example in Fig. 1.2 also includes human crew transports and manual inspections for failure prevention and response.

1.2.4 Openness and Dynamic Property

The dynamic property of a CPS results from the changing devices and the changing environment. On the one hand, the 'Plug-n-Play' functionality of CPSs introduces openness and enables changing devices and agents without a total network reconfiguration or the prior knowledge of the entire set of connected devices. For example, the telecommunication layer of the CIN in Fig. 1.2 needs to support the channel switch as people commute and change their locations. On the other hand, the physical layer naturally undergoes uncertainty and dynamic environmental factors. For example, despite the control systems, the normal operation of the physical plant in Fig. 1.1 inevitably fluctuates based on the variations in temperature and atmospheric pressure.

1.3 High-Confidence CPSs

In Section 1.3, we use the ICS and CIN examples in Section 1.1 to illustrate the meaning and motivation of high-confidence CPSs in Section 1.3.1, the potential threats in Section 1.3.2, and the status quo in Section 1.3.3, respectively.

1.3.1 Meaning and Motivation

A high-confidence CPS is required to behave in a *well-understood*, *predictable*, and *justifiably trusted* fashion. Moreover, it needs to protect itself from and withstand natural disasters and malicious attacks to fulfill *life-critical* tasks such as medical surgery, autonomous driving, and nuclear power plant control.

The high-confidence design of a system aims to protect it across the entire attack cycle, which can be decomposed into three stages: *ante impetus*, *per impetus*, and *post impetus*. *Ante-impetus high confidence* entails that a CPS needs to be well-prepared before natural or human attacks to reduce the probability of failures. All the cyber-physical components need to fulfill their functionalities with high accuracy and reliability, even under extreme events and malicious attacks. In the ICS example of Fig. 1.1, the essential components should have backups. If a pump fails due to aging or a compromise, there should be alerts shown on the HMI while the system automatically implements a backup pump. In the CIN example of Fig. 1.2, it is useful to introduce redundancy both within and across the layers. Once a component fails within an infrastructure sector, there should be a contingency plan that prevents a cascading failure of other components within the sector and even in other sectors.

Per-impetus high confidence requires a CPS to be responsive and resilient during failures. Once a malfunction or a compromise happens inevitably, the system needs to detect it *timely*, locate the anomaly *precisely*, and respond to it *cost-efficiently*. If the system cannot recover in the short run, then the major goal of the response is to contain the propagation of failures and reduce the damage to the entire system. Due to the interdependence, we need to guarantee that the response does not introduce a negative impact on the normal operation of other components. Besides performance, high confidence also implicitly requires the detection and response to be cost-efficient so that the defender has sufficient budgets to invest in other components and prevent their failures proactively.

Finally, *post-impetus high confidence* means that a CPS needs to be reflective after the failures. When the system has recovered to its normal operation state, we need to collect and analyze the information related to the anomaly. The information provides beneficial guidelines and feedback to update the system design to be more *ante-impetus* and *per-impetus* high-confidence. For example, after the system returns to normal, field operators or humans in the control room can analyze the incident to improve the design of the pump or the structure of the system (e.g., the pump may fail because it is located in a room of high temperature).

1.3.2 Threats to High-Confidence CPSs

The features identified in Section 1.2 enhance the performance of a CPS to provide essential services that support economic prosperity, governance, and quality of life. However, as a double-edged sword, they also create complex System of Systems (SoS), introduce vulnerabilities, and bring cross-cutting concerns to build high-confidence CPSs. The growth of intelligent and sophisticated attacks also significantly exacerbates the concerns.

The authors in [102] categorize the vulnerabilities based on three types (i.e., cyber, physical, and cyber-physical) and five causes (i.e., isolation assumption, connectivity, openness, heterogeneity, and incoordination among stakeholders). Under certain circumstances, these vulnerabilities can lead to various threats that compromise a CPS's confidentiality, integrity, availability, privacy, safety, and other composing dimensions of a high-confidence design. According to the ISO/IEC 27001:2013 standard, threats can be environmental, accidental, or deliberate [103]. In the following, we introduce cascading failures (caused by natural dusters), human errors (e.g., phishing victims and unintentional insider threats), and Advanced Persistent Threats (APTs) as the representative environmental, accidental, and deliberate threats, respectively. Note that each representative threat is not exclusively restricted to the associated category. For example, cascading failures caused by human errors are also accidental threats. Cascading failures caused by attacks and intentional insider threats are also deliberate threats. Attacks can also exploit environmental factors and human errors to save effort and maximize their impact.

Cascading Failures

In Fig. 1.2, we illustrate the cyber, physical, logical, and geographical dependence within and across heterogeneous CINs that are open to devices and inherently dynamic. These CPS features can lead to cascading failures. First, the *openness* and *dynamic* properties make the entire CPS valuable to faults and attacks. Second, the *heterogeneous attributes* make it challenging to develop a unified failure prevention and response mechanism. Third, as a result of the *large-scale* and *complex* interdependence, cyber and mechanical outages in one component can affect others and exacerbate to cause cascading failures. For example, during Hurricane Sandy, failures inside the power grids led to a large-size blackout, and then the power outage propagated negatively to the dependent infrastructures (e.g., transportation and communications), which wreaked havoc [182].

Human Errors: Phishing and Insider Threats

We classify human vulnerabilities into *acquired* and *innate* vulnerabilities [86], depending on whether they can be mitigated through short-term security training and awareness programs. Based on the classification, we regard social engineering and insider threats as examples of *innate vulnerability* (e.g., bounded attention and rationality) and *acquired vulnerability* (e.g., lack of security awareness and incentive), respectively.

As a quintessential example of social engineering, phishing attacks use emails or malicious websites to serve malware or steal credentials by masquerading as a legitimate entity. The authors in [38] have identified three human vulnerabilities that make humans the unwitting victims of phishing.

- Lack of knowledge for computer system security; e.g., the Uniform Resource Locator (URL) `www.ebay-members-security.com` does not belong to `www.ebay.com`.
- Inadequacy to identify visual deception; e.g., a phishing email can contain an image of a legitimate hyperlink, but the image itself serves as a hyperlink to a malicious site. Humans cannot identify the deception by merely looking at it.
- Lack of attention (e.g., careless users fail to notice the phishing indicators such as spelling errors and grammar mistakes) and *inattentional blindness* (e.g., users focusing on the main content fail to perceive unloaded logos in a phishing email [17]).

Insider threats in cyberspace refer to vulnerabilities and risks posed to an organization due to the misbehavior of its trusted but not trustworthy insiders, such as employees, maintenance personnel, and system administrators. In 2020, insider threats have caused around 30% of breaches [15], which result in significant operational disruption, data loss, and reputation damage. Besides the human-in-the-loop feature, the openness and the dynamic entering (or leaving) of devices (or agents) further make it difficult to deter insider threats. For unintentional insider threats, misbehavior is caused by human errors. As the weakest link in cybersecurity, humans unavoidably make mistakes due to their innate and acquired vulnerabilities.

Advanced Persistent Threats

Advanced Persistent Threats (APTs) are a class of emerging threats for CPSs with the following distinct features. Unlike opportunistic attackers who spray and pray, APTs have specific targets and sufficient knowledge of the system architecture, valuable assets, and even defense strategies. Thus, APT attackers can tailor their strategies and invalidate cryptography, firewalls, and Intrusion Detection Systems (IDSs). Unlike myopic attackers who smash and grab, APTs are stealthy and can disguise themselves as legitimate users for a long sojourn in the victim's system.

A few well-accepted APT models have divided the entire intrusion process into a sequence of phases, such as Lockheed-Martin's Cyber Kill Chain [105], MITRE's ATT&CK [31], the NSA/CSS technical cyber threat framework [37], and the ones surveyed in [140]. Fig. 1.3 illustrates an exemplary multi-stage structure of APTs. During the reconnaissance phase, a threat actor collects open-source or internal intelligence to identify valuable targets. After the attacker obtains a private key and establishes a foothold, he escalates privilege, propagates laterally in the cyber network, and eventually either accesses confidential information or inflicts physical damage. As a sophisticated class of attacks, standalone defense on a physical

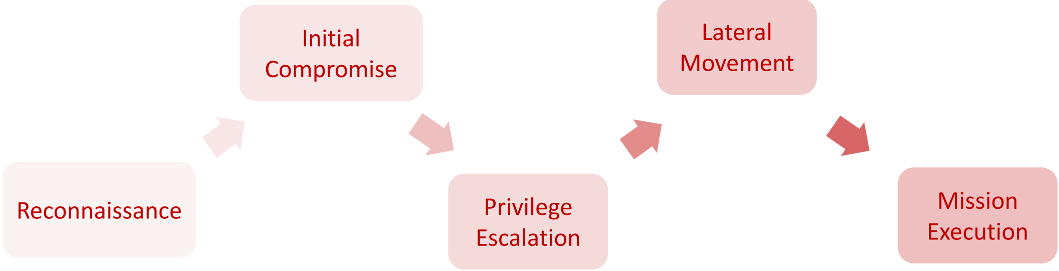


Figure 1.3: An example of the multi-stage structure of APTs. The multi-stage attack is composed of reconnaissance, initial compromise, privilege escalation, lateral movement, and mission execution. An attack originating from an early-stage cyber network can damage a physical system.

system cannot deter attacks originating from a cyber network. Solely technical defense cannot prevent APTs as they can use social engineering to compromise humans. Moreover, the static defense fails to deal with persistent lateral movement.

1.3.3 Status Quo and Related Works

In Section 1.3.3, we present the status quo of the three threat categories in Section 1.3.2 and the existing mitigation methods.

Cascading Failures and Risk Management

Cascading failure analysis and modeling has been widely conducted in power systems [70, 189] and other complex infrastructure systems [40] concerning the causes, procedures, impact, and restoration. Recently, the impact of cascading failures on IoT has drawn increasing attention [57, 222]. In [222], the authors identify seven fundamental causes of cascading failures. They are dynamic conditions, cyber attacks, physical attacks, operator error, overload, extreme weather, and natural disasters. The major types of cascading failure models can be characterized as self-organized critical-based, network-based, and simulation-based [222].

To analyze and manage the risks of interdependent CINs, various models have been proposed based on network flows [123], numerical simulations [117], dynamic coupling [180], and the ones summarized in [164]. For risk assessment of intelligent attacks, a framework for assessing the physical impact of cyber attacks on ICSs has been proposed in [81]. The proposed method has been demonstrated using a hardware-in-the-loop testbed with a boiling water power plant model. In [165], a probabilistic risk analysis framework has been proposed. The risk analysis in the framework is based on attacker demographics and entry points, attack scenarios, and attack impacts. In [121], Markov processes and semi-Markov processes have been used to model cyber attacks on CPSs and dependability measures, e.g., availability, reliability, mean time to failure, and confidentiality. In [163], a game-theoretic model has been proposed to obtain the strategies of the defender and the attacker during two phases of cyber attacks on CPSs, i.e., the penetration phase and the disruption phase. In summary, the existing methods for cybersecurity risk assessment can be divided into two classes: non-game-theoretic approaches and game-theoretic approaches. In non-game-theoretic approaches, the parameters related to the attackers (e.g., probability of a certain type of attack) are usually estimated based on expert opinions or data. The main criticism of such approaches is that they do not consider the adaptation of intelligent attackers, as pointed out in [33]. Game theory provides a natural tool to predict the behavior of intelligent attackers and game-theoretic approaches have been trying to fill the gap of parameter estimation in traditional risk analysis methods. We provide an overview of game theory for security (i.e., security games) in Section 2.1.

Human-Induced Vulnerability and Assistive Technologies

Social engineering [186] is a common attack vector that targets the acquired human vulnerabilities such as fear to express anger, lack of assertiveness to say no, and the desire to please others. Threat actors use psychological manipulation techniques to mislead people to break normal security procedures or divulge confidential information. Non-technical anti-phishing solutions include security training and education programs, while technical solutions include blacklisting, whitelisting, and feature-based detection. Visual support systems have been used for rapid cyber event triage [142] and alert investigations [52], and eye-tracking data have been incorporated to enhance attention for phishing identification [86]. The authors in [206] perform an anthropological study in a corporate Security Operation Center (SOC) to model and mitigate security analyst burnout. To handle zero-day phishing attacks, (deep) Reinforcement Learning (RL) has been used both to detect phishing emails [199], phishing websites [24], spear phishing [46], and social bots [129] in online social networks.

Insider threats can be classified into unintentional or intentional ones. For unintentional insider threats, the authors in [66, 209] have identified three contributing factors (i.e., organizational, human, and demographic) and a set of proactive mitigation strategies (e.g., awareness training, relieving time and workload pressure, and usability of security tools to help overcome user errors). For intentional insider threats, the authors in [74, 146, 213] have recognized incentives as a leading factor and incentive design as a promising mitigation strategy. Insider threat mitigation needs to rely on an integrated technical (e.g., audit and access control) and social or organizational (e.g., security policies and positive organizational culture) solution [65, 104, 188].

Biosensors such as eye trackers and electroencephalogram (EEG) devices enable an analytical understanding of human perception and cognition to enhance security and privacy [111]. In particular, researches have investigated the users' gaze behaviors and attention when reading URLs [175], phishing websites [144], and phishing emails [32, 138, 223]. These works illustrate the users' visual processing of phishing contents [138, 144, 169, 175] and the effects of visual aids [223]. The authors in [144] further establish correlations between eye movements and phishing identification to estimate the likelihood that users may fall victim to phishing attacks.

Due to the unpredictability and modeling challenges of human behaviors, RL and feedback control serve as the tools to affect human incentives and perceptions effectively and efficiently. In [23], the penalty and reward are changed adaptively through a feedback system to improve the compliance of human employees and mitigate insider threats. In [86], RL is used to develop the optimal visual aids to enhance users' attention and help them identify phishing attacks. In [98], the authors use RL to determine resilient and adaptive strategies for alert and attention management.

APTs: Prevent, Detect, Response, and Recovery

One well-known industrial solution to APT defense is the ATT&CK matrix [31]. It illustrates disclosed attack methods and possible detection and mitigation countermeasures at different phases of APTs. However, as argued in [41], it lists all possible attack methods in one matrix and lacks prioritization. A lot of false alarms arise as legitimate users may generate a majority of the activities in the ATT&CK matrix. Besides, despite a persistent update, the matrix is far from complete and can lead to missed detection.

Many papers have attempted to deal with the above two challenges, i.e., false alarms and missed detection. To prevent security specialists from overwhelming alarms, [136] has analyzed high volumes of network traffic to reveal weak signals of suspect APT activities and ranked them based on the computation of suspiciousness scores. To identify attacks that exploit zero-day vulnerabilities or other unknown attack techniques, [55] has managed to learn and maintain a white-list of normal system behaviors and report all actions that are not on the white-list. There is also a rich literature on detecting essential components of an APT attack, such as malicious Portable Document Format (PDF) files in phishing emails [157], malicious Secure Sockets Layer (SSL) certificate during command, control, and communications [62], and data leakage at the final stage of the APT campaign [197]. These works have focused on a static detection of abnormal behaviors in one specific stage but have not taken into account the correlation among multiple phases of APTs. The authors of [60] have managed to build a framework to correlate alerts across multiple phases of APTs based on machine learning techniques so that all those alerts can be attributed to a single APT scenario. The authors of [61] have constructed a correlation framework to link elementary alerts to the same APT campaign and applied the hidden Markov model to determine the most likely sequence of APT stages.

An alternative perspective from the aforementioned APT detection frameworks is to address how to respond to and mitigate potential attacks. The authors of [125] have captured the dynamic state evolution through a network-based epidemic model and provided both prevention and recovery strategies for defenders based on optimal control approaches. Since APTs are controlled by human experts and can act strategically, the defender's response should adapt to the change of APT behaviors. Thus, decision and game theory becomes a natural quantitative framework to capture constraints on defense actions, attack consequences, and attackers' incentives. The authors in [215] have proposed *FlipIt* game to model the key leakage under APTs as a private takeover between the system operator and the attacker. Many works have integrated *FlipIt* with other components for the APT defense such as the signaling game to defend cloud service [166], an additional player to model the insider threats [49], and a system of multiple nodes under limited resources [231]. *FlipIt* has described a high-level abstraction of the attacker's behavior to understand optimal timing for resource allocations.

1.4 AI-Powered System-Scientific Defense

Lessons learned from hurricanes Katrina and Sandy, as well as ongoing incidents caused by human errors and APTs, have shown the inadequacy of the off-the-shelf defense mechanisms. Given the fast-growing technologies such as Artificial Intelligence (AI) and the unsatisfying status quo in Section 1.3.3, we need to develop next-generation defense mechanisms to build high-confidence CPS against the environmental, accidental, and deliberate threats in Section 1.3.2.

In Section 1.4.1, we first provide a characterization of five generations of Security Paradigms (SPs) based on a brief history of the epoch-making technologies, attacks, and defense methods. Then, we position Defense through AI-powered SYstem-scientific methods (DAISY) as the essence of the fifth-generation SP and elaborate on the advantages of DAISY induced by its composing six features in Section 1.4.2.

1.4.1 Five Generations of Defense Mechanisms

In recent decades, we have witnessed not only a surge in the number of attacks but also their increasing sophistication and capacity. Researchers, engineers, and scientists have endeavored to persistently develop new SPs to keep up with the evolving attacks. We roughly characterize those SPs into the following five generations as shown in Fig. 1.4.

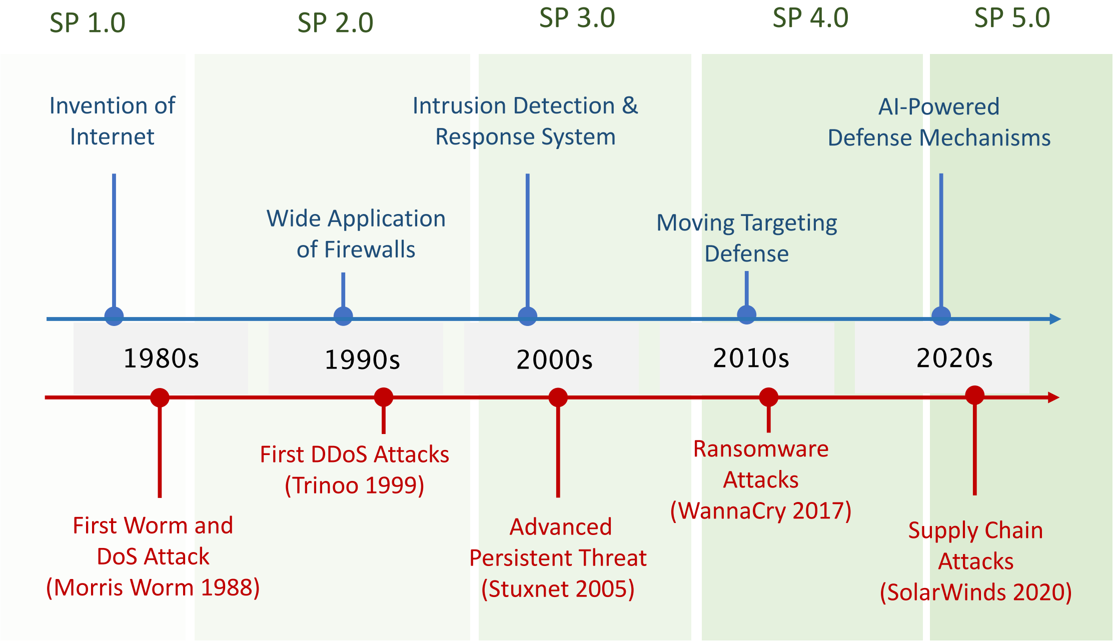

SP 1.0
SP 2.0
SP 3.0
SP 4.0
SP 5.0
Invention of Internet
Wide Application of Firewalls
Intrusion Detection & Response System
Moving Targeting Defense
AI-Powered Defense Mechanisms
1980s
1990s
2000s
2010s
2020s
First Worm and DoS Attack (Morris Worm 1988)
First DDoS Attacks (Trinoo 1999)
Advanced Persistent Threat (Stuxnet 2005)
Ransomware Attacks (WannaCry 2017)
Supply Chain Attacks (SolarWinds 2020)

Figure 1.4: Five generations of SPs are driven by new technologies, attacks, and defense methods. Representative attack categories and attack incidents are positioned below the timeline in red. Epoch-making technologies and defense methods are positioned above the timeline in blue. SPs 1.0 to 5.0 are illustrated in progressively darkening green, respectively.

1G-SP: Laissez-Faire Security

At the infant stage of the Internet (i.e., 1960s to 1980s), the focus has been on designing reliable systems to share information, whereas security has not been taken into consideration. Such laissez-faire security is natural and acceptable at that time as the size of the network is rather small, the components are fully controlled by trustworthy entities, and the users follow *Postel's law* [171].

Even for the infamous Morris worm of November 2, 1988, which was considered one of the oldest computer worms and Denial-of-Service (DoS) attacks distributed via the Internet, its creator intended to demonstrate the weaknesses of the networks rather than to cause damage. Despite the creator's non-malicious intent, the Morris worm infected around 6,000 major Uniplexed Information and Computing System (UNIX) machines and led to an estimated cost of damage of $100,000$ to $10,000,000$. This incident, together with other upcoming malware, marks the end of SP 1.0.

2G-SP: Perimeter Security

The wide application of firewalls marks the beginning of SP 2.0 which mainly focuses on intrusion prevention. Developed in the 1980s at several technology companies (e.g., *Cisco Systems* and *Digital Equipment Corporation*), the first firewalls are referred to as the 'network layer' firewalls as they monitor and filter packets based on rules concerning the source, destination, and types of the packets. As an *add-on* solution, 'application layer' firewalls emerged in the early 1990s to perform a more thorough inspection, e.g., analyze the application layer headers. Both types of firewalls set up black-lists or white-lists based on a series of configured policies and have been proven to reduce indiscriminate attacks effectively. A single firewall with at least three network interfaces or multiple firewalls can be used to create a network architecture containing a Demilitarized Zone (DMZ). A DMZ serves as an isolated network between an untrusted network (e.g., Internet) and the private network.

Although fast and transparent, rule-based firewalls can be easily deceived and evaded by targeted attacks. The first known Distributed Denial-of-Service (DDoS) attack in 1996 targeted Panix, the oldest Internet Service Provider (ISP) in New York. The attack swamped the computer systems with an SYN flood, and it took Panix approximately 36 hours to get back on track. SP 2.0 becomes insufficient as the defender starts to acknowledge that the attacker can evade Intrusion Prevention Systems (IPSs) and penetrate the system.

3G-SP: Reactive Security

The 2G-SP of perimeter security mimics the castle defense strategies in the middle ages, where firewalls at different layers serve the purpose of moats, ramparts, and walls. Concentric castles with two or more concentric curtain walls create several DMZs among the internal and external curtain walls. Despite the layers of walls to break, once an attacker breaches the perimeter and penetrates the system, the system cannot protect itself from unauthorized lateral movement and asset compromise.

The third-generation SP complements the static and network-based perimeter defense with a dynamic protection of users, assets, and resources. Its solution is a response to enterprise network trends of remote users, Bring-Your-Own-Device (BYOD), and complexity that has outstripped legacy methods to identify the perimeter. As an analogy to the above castle defense example, besides IPSs (e.g., firewalls and DMZs), 3G-SP further recruits and equips soldiers in the castle to detect and respond to Trojan horses that have entered the castle. Compared to perimeter security where defense is enforced at "choke points" to achieve acceptable security at the minimal effort, the new SP needs persistent monitoring, detection, and response. Thus, 3G-SP, referred to as the reactive security, only becomes possible when technology evolves to support the communication and real-time analysis of the large data and log files.

The growth of various Intrusion Detection System (IDS) and Intrusion Response System (IRS) in the early 2000s is a landmark of SP 3.0 and the best security practice at that time. The IDS has also gradually evolved from being rule-based to behavioral-based, and the growing success of machine learning techniques has contributed to increasing the detection rate and reducing the false alarm rate.

The 3G-SP of reactive security combines IPSs, IDSs, and IPSs to achieve the approach of Defense in Depth (DiD). However, due to the increasing sophistication and the adoption of new attack methods (e.g., social-engineering and adversarial cyber deception), IDSs and IPSs become less sufficient to protect a CPS. One example of APT attacks, Stuxnet, starts its initial infection through the Universal Serial Bus (USB) driver of the hardware provider. These USB drives are stealthily compromised by Stuxnet when the hardware provider serves other less secure clients. Thus, Stuxnet manages to compromise the air gap even though the nuclear system is carefully isolated from the Internet. Stuxnet and other APT attacks indicate the insufficiency of SP 3.0 and motivate proactive defense mechanisms.

4G-SP: Proactive Security

Broadly speaking, proactive defense refers to *acting in anticipation to counteract an attack through cyber and cognitive domains*. Compared to reactive defense methods in 3G-SP, proactive defense in 4G-SP focuses on taking initiative by *acting* rather than *reacting* to threat events. Cyber deception is a quintessential proactive defense method. It strategically changes the attacker's behaviors by preventing them from forming a true belief. The use of deception in military defense is not new, and it can date back to as early as roughly 5th century BC in Sun Tzu's Art of War [214]. However, it is not until the 2010s that we have witnessed increased popularity and advantages of Moving Target Defense (MTD) and honeypots to support the security need of large-scale CPSs.

MTD makes systems inherently dynamic to limit the exposure of vulnerabilities and the effectiveness of the attacker's reconnaissance by increasing the complexities and costs of attacks. Since its first introduction as a cyber-defense paradigm in

2009 [63], MTD has shown its success to deter attackers [45]. A honeypot emulates the real production system but has no production activities or authorized services. Thus, an interaction with a honeynet, e.g., unauthorized inbound connections to any honeypot, directly reveals malicious activities, which results in a low false alarm rate. Moreover, honeypots provide a constrained environment to interact with attackers and gather threat intelligence, as shown in Chapter 8.

5G-SP: Federated Security

In the late 2010s, we have witnessed the threats from AI-power attacks [30, 68, 225]. For example, researchers have found that tools like OpenAI's GPT-3 can help craft spear-phishing messages, which significantly lower the barrier to entry for crafting spear-phishing campaigns at a massive scale [154]. Equipped with AI and big data, attackers may launch massive advanced threats of high pertinence at low cost. These new threats motivate us to integrate intelligently and systematically the defense technologies (e.g., encryption, IDSs, IPSs, IPSs, and defensive deception technologies) in prior SP generations to develop a holistic next-generation SP of federated security. Federated security relies on AI-powered and system-scientific defense mechanisms that entail varied dimensions on which we will elaborate in Section 1.4.2.

1.4.2 Six Dimensions of DAISY to Achieve 5G-SP

Compared to the off-the-shelf defense mechanisms in 1G-SP to 4G-SP in Section 1.4.1, DAISY as the 5G-SP adopts system science to connect heterogeneous technologies with holistic and scientific understandings of the entire system. AI, on the one hand, enables the defender to understand the intents and behaviors of attacks and users. On the other hand, it provides planning and learning tools (e.g., RL in Section 2.3) to obtain knowledge from data and enable data-driven optimization of the defense.

In Section 1.4.2, we elaborate on the transition from off-the-shelf defense mechanisms to DAISY and the current security landscape in the six dimensions summarized in Table 1.1. We further illustrate how our works in Part II to Part VI achieve the six dimensions of DAISY and address the challenges in the current security landscape.

Current Security Landscape	**Off-the-Shelf Defense**	**Dimensions of DAISY**
Lack of security standards and metrics	Empirical	Theoretical
Human-targeted and human-induced attacks	Technical	Socio-Technical
Strategic attackers	Single-Agent	Multi-Agent
Imperfect security	Secure	Resilient
Piecemeal design of CPS	Add-On	Built-In
Defender's four disadvantages	Reactive	Proactive

Table 1.1: The six dimensions of the current security landscape, the off-the-shelf defense mechanisms, and the next-generation defense mechanisms.

From Empirical to Theoretical

Due to the lack of security standards and metrics in heterogeneous CPSs illustrated in Section 1.2.1, many defense practices rely on empirical rules and trial and error, which lead to the following three challenges. First, empirical rules are inaccurate, unreliable, and time-costly to collect, especially in dynamic and uncertain environments as shown in Section 1.2.4. Second, it is challenging to automate the design of rule-based defense and further transfer the design to diverse and heterogeneous CPSs. Third, the empirical defense fails to assess the risk of a CPS to achieve a tradeoff of important security metrics, e.g., Confidentiality, Integrity, and Availability (the CIA triad), which is exacerbated in large-scale CPSs of complex interdependence as shown in Section 1.2.2.

To this end, we establish quantitative models in Parts II to VI to distill principles from the case-by-case investigations, strike security-usability tradeoff, and optimize the security design based on system-scientific tools and AI. Quantification is indispensable to developing theoretical defense mechanisms and addressing their challenges. On the one hand, these quantitative models address the challenges of uncertainty and risk assessment by incorporating probability and expectation, respectively, to characterize the randomness and the average impacts of the system and agents' behaviors. On the other hand, these quantitative models enable an automated and transferable design of defense policies by incorporating feedback and learning. Automated and adaptive defense is instrumental to developing timely and reliable defenses against AI-powered attacks with dynamic and intelligent strategies.

From Technical to Socio-Technical

Humans have been closely integrated into CPSs as shown in Section 1.2.3 and are often treated as the weakest link in CPS security. Following Section 1.3.2, the increasing human-targeted attacks (e.g., social engineering) and human-induced attacks (e.g., insider threats), have emphasized the desideratum to transform from solely technical solutions to socio-technical ones. Socio-technical solutions in CPSs incorporate both observable human elements (e.g., behaviors, eye-gaze locations, and EEG signals) and unobservable human elements (e.g., rationality, attention, risk attitudes, and learning capacity). With these human elements considered, socio-technical solutions further aim to understand, characterize, and further guide the human perception and decision-making processes. For example, the defender can design negative incentives (e.g., punishments) and positive incentives (e.g., rewards and recognition) to mitigate insider threats when the insiders are self-interested and pursue benefits [146].

The socio-technical solutions developed in this dissertation consider human elements at different granularity levels. Parts II, III, IV, and V model humans *implicitly* as rational decision-makers who take action to maximize their benefits. Focusing on quantifying the impacts of human behaviors on CPS security, these *population-based* models simplify the complicated human decision process as optimization problems or game-theoretical problems. In Part VI, we *explicitly* model human attention dynamics and real-time decision-making processes when users or defenders encounter phishing emails or a large number of alerts. Incorporating empirical laws (e.g., Yerkes–Dodson law), biographical data (e.g., eye-tracking data and surveys), and essential human factors (e.g., levels of expertise, stress, and efficiency), these *agent-based* models make the unobservable human elements measurable. On the one hand, these agent-based models enable us to understand human perception of security and lead to the foundation of a *theory of security mind*. On the other hand, they lead to the design of assistive technologies (e.g., ADVERT in Chapter 11 and RADAMS in Chapter 12) to compensate for the 'unpatchable' human attentional vulnerabilities.

From Single-Agent to Multi-Agent

Compared to natural disasters, man-made and AI-powered threats can strategically and intelligently take actions to counteract defense methods and affect the outcomes. Therefore, although we can predict floods, hurricanes, and even earthquakes with increased accuracy and timeliness, it is still challenging to predict, detect, and deter cyber attacks. To predict natural disasters, we need to understand the physical laws that lead to the outcomes. To predict cyber attacks, we need to understand the *laws of intelligent agents*, e.g., the adversaries' preference, knowledge, capacity, and rationality when launching the attacks. To discern why these cyber attacks happen to some CPS components at this time in this way, we should not treat CPS defense as a single-agent decision problem of the defender but a multi-agent problem consisting of the following interacting agents.

Attacker is one player in a security game. There are various types of attackers such as script kiddies, cyber punks, insider threats, and cyber terrorists. Their different motivations, capacities, and attack goals can affect their behaviors and payoffs. A defender is another player in a security game. The defenders can represent security experts in an SOC of a corporate, operators in the control room and the field, a third-party security provider (e.g., FireEye), a research institution, and a governmental department (e.g., the Department of Homeland Security (DHS)). Legitimate users are sometimes overlooked in a security game. As humans are the weakest link, it is important to increase users' security awareness and compliance to prevent social engineering and reduce unintentional insider threats. Security games can include many other players, including cyber insurance agents [137].

The works in this dissertation provide different characterizations of the multi-agent interactions based on the security applications. In Chapters 8, 11, and 12, we establish models with unknown parameters to characterize the attackers' response to honeypot strategies, the users' response to phishing emails under visual aids, and the defenders' response to a large volume of alerts, respectively. Then, we adopt learning methods such as RL to estimate the unknown parameters from the data. In Chapters 9 and 10, we develop *principle-agent* models and design incentive-*compatible* mechanisms so that self-interested and adversarial insiders follow the defender's security policies. In Chapters 3, 4, 5, and 6, we incorporate game theory to model the interactions of these agents explicitly. The celebrated concept of Nash Equilibrium (NE) provides a reliable prediction of the agents' behaviors and the interaction outcome because no players can benefit from unilateral deviations from the equilibrium.

From Secure to Resilient

Lessons from APT incidents have highlighted that perfect security is usually impossible or cost-prohibitive. Moreover, pursuing absolute security *locally* and *temporarily* can result in unexpected insecurity to the entire system in the long run, as shown in the following two examples.

- When the defender isolates a computer network from the external network, the air gap blocks not only attacks but also a real-time update of the virus database and vulnerability patches. Then, once an attack bridges the air gap, it can remain in the isolated system without being detected.

- When the company's security team sets up complicated password rules and requires a frequent password change, then the employees end up writing down their passwords and putting them next to their computers, making the entire corporate network vulnerable to insider threats and social engineering.

Focusing on both intrusion prevention and response, resilience plays an increasingly significant role to complement the imperfect security in CPS applications. As shown in Fig. 1.5, we decompose a resilient SP into the following four stages [101], referred to as the P2R2 stages. The design goal is to minimize the delays (i.e., D_1, D_2, and D_3) and the performance degradation (i.e., M and G).

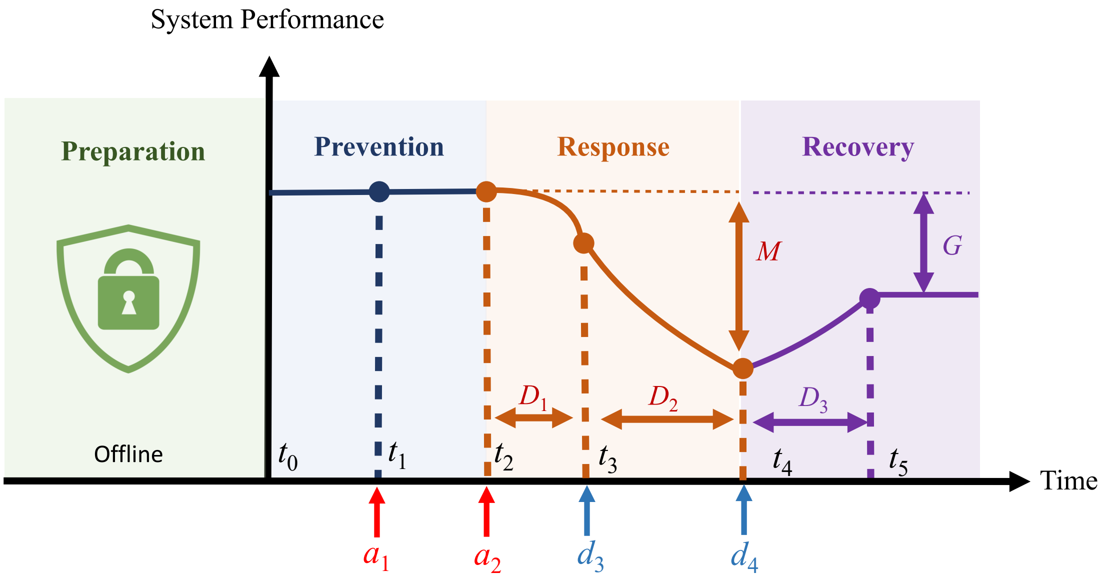


Figure 1.5: The four stages of a resilient SP: Preparation, Prevention, Response, and Recovery (P2R2). The x-axis represents the operation timeline (both offline and online). The y-axis represents the system performance. The CPSs starts to operate at t_0 while natural disasters or attacks occur at t_1, t_2, and t_5. The system performance decreases when an attack a_2 successfully penetrates the system at t_2. After the detection d_3 at t_3, defense d_4 takes place at t_4, and the system partially restores to its best-effort post-attack performance at t_5.

Preparation is the first stage in designing a high-confidence CPS and is often done offline and ahead of the real-time operations. The goal of preparation is to identify valuable assets and vulnerabilities to reduce the attack surface, assess the security risk, and design appropriate security policies, including awareness and training [2], proper configurations of detection systems [235], and deployment of deception technologies [6, 168]. Good preparation can help facilitate effective prevention and fast response to unanticipated scenarios in later stages.

The second stage is prevention. At this stage, we implement the designed security policies to protect the CPS in real-time. Due to the meticulous preparation and the policy design in the preparation stage, some attacks can be readily deterred, detected, and thwarted. For example, consolidating MTDs [106] into the communication protocols would make it harder for the attacker to map out the traffic patterns and consequently thwart the DoS attacks. However, due to the natural inferior position of the defender against attackers, there is still a probability for an attacker to become successful, especially for a highly resourceful and stealthy one.

The response stage is critical to defending against attacks when the defense fails to thwart them at the prevention stage. At this stage, we acquire the information based on the footprint of the attacker and reconfigure the CPS to minimize the further risk of the attack [73, 176]. In Fig. 1.5, after attack a_2 successfully penetrates the system at t_2, it takes the defender a delay of D_1, D_2, and D_3 to detect, respond, and contain the attack, respectively. The detection D_1 and response delay D_2 lead to the worst-case performance degradation of M.

The fourth stage of a resilient SP is recovery, where the goal is to reduce the spill-over impact of an attack and restore the system performance as much as possible. The response to attacks in real-time is often at the sacrifice of the performance of the CPS. There is a need to maintain the system's operation and gradually restore its functionality to normal while reacting to the attacks. In Fig. 1.5, as defense d_4 counteracts the adversarial impact of the attack a_2, the system performance gradually restores to the best-effort post-attack performance at t_5 with a performance gap G.

The major challenges for resilience are to withstand uncertainties in dynamics, pursue long-term benefits, and reduce delay. First, the response needs to be adaptive and robust due to the dynamic environment (e.g., uncertainty in the physical plant operation) and the changing devices and agents enabled by the 'Plug-n-Play' functionality of CPSs. The human behaviors in a CPS also change the status of the CPS and lead to additional uncertainty. Second, the defender needs to properly allocate the limited resources to the P2R2 stages illustrated in Fig. 1.5. Due to the budget limit, increasing investment in the preparation and prevention stages to increase security may result in insufficient resources in the response and recovery stages and consequently impair resilience. To strike a balance between security and resilience, the defender needs to take non-myopic actions to maximize long-term benefit. Third, reducing the delay in detection, response, and containment is challenging due to the complexity and interdependence of CPSs, the operators' knowledge limitations, and their increased stress during the incident. As reported in [131], United States companies in 2018 have taken an average of 197 and 69 days, respectively, to detect and contain a data breach.

To address the first two challenges, we adopt Markov transition models (to characterize the dynamic and stochastic state transitions) and optimize cumulative utility functions, respectively, as illustrated in Chapters 3, 4, 5, 6, 8, 11, and 12. Based on the CPS applications in these chapters, the state could represent the status of digital and physical components (e.g., Chapters 3, 4, and 5), the physical locations of robots (e.g., Chapter 6), the location and the status of attacks (e.g., Chapter 8), and the status of human attention (e.g., Chapters 11, and 12). Besides Markov transition models, Chapter 7 adopts stochastic time-expanded networks to model the random arrivals of attacks and services. To resolve the third challenge in delay-sensitive CPS applications, e.g., nuclear power plants, we further incorporate semi-Markov models in Chapter 4 to obtain a time-sensitive attack response with a real-time risk assessment.

From Add-On to Built-In

As the CPS components are designed in a piecemeal rather than a holistic fashion, add-on security leaves parts of a system vulnerable [126] and also brings challenges to identify valuable assets and vulnerabilities accurately, especially under large-scale interconnections, complex interdependence, and heterogeneous attributes identified in Section 1.2. Moreover, add-on security expands the attack surface and allows adversaries to damage the physical part of a CPS by compromising the cyber layer. For example, Stuxnet [122], one example of APTs, has infected over $200,000$ computers all over the world to compromise the targeted PLCs in the air-gapped SCADA system and ruined almost one-fifth of Iran's nuclear centrifuges (over $1,000$ centrifuges). Without the proper identification of assets and vulnerabilities, it takes security experts more than 5 years to unveil this stealthy attack.

In this dissertation , we endeavor to make security a built-in feature of CPSs by holistically consider the six-layer hierarchical structure of CPSs [100] in Fig. 1.6. The physical layer consists of a physical plant embedded with actuators and sensors. The control system receives commands and observations and sends commands to actuators to achieve desired system performance. The communication layer provides wired or wireless data communications that enable advanced monitoring and intelligent control. The network layer allocates network resources for routing and provides interconnections between system units. The supervisory layer serves as the executive brain of the entire system, provides HMIs, and coordinates and

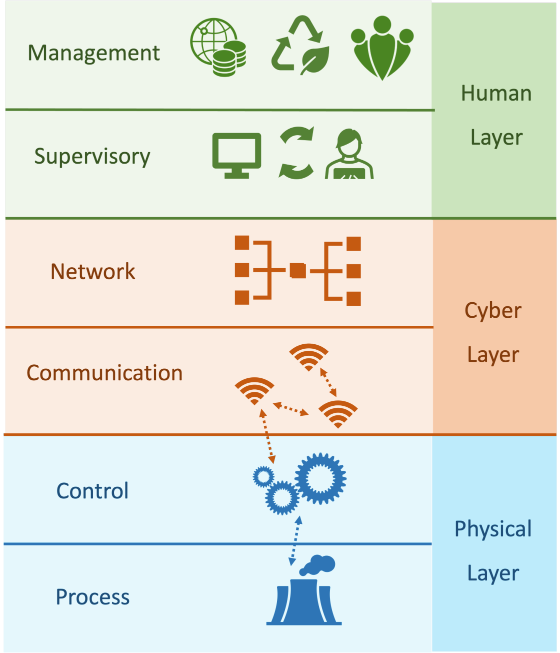


Figure 1.6: The six-layer hierarchical structure of a CPS: the physical layer of control and process, the cyber layer of network and communication, and the human layer of supervisory and management.

manages lower layers through centralized command and control. The management layer resides at the highest echelon. It deals with social and economic issues, such as market regulation, pricing, incentive, and environmental affairs.

In Chapter 3, we develop a holistic model of the interdependence among different infrastructure sectors to strike the balance of prevention and response under budget and resource limits. In Chapters 4, 5, and 6, we develop holistic models of multi-stage multi-phase attacks in cyber and physical layers to enable proactive defense at these attack stages and make the attacks less dominant when they reach the final stage. In Chapter 7, we consider time and spatial locations holistically to discover latent attack paths. In Parts V and VI, we further integrate human elements of incentives and attention with the cyber and physical layers for mitigating acquired and innate human vulnerabilities and improving organizational cyber hygiene.

From Reactive to Proactive

Reactive defense methods in Section 1.4.1 naturally suffer from the defender's disadvantages of *space*, *time*, *information*, and *cooperation*. First, compared to attackers who only need to compromise one component to sabotage the entire CPS, a defender has to protect all CPS components. Second, attackers can launch attacks at any time and only need to succeed once, while a defender has to protect the system during the entire operation time. Third, persistent attackers can collect information about the system in their reconnaissance stages in Fig. 1.3 to find the weakest link. Constrained by budget and defense technology, the defender usually cannot collect and analyze all the user data to identify attackers and learn the threat intelligence. Fourth, a successful defense requires coordination of multiple parties with varied goals and conflicting incentives. A typical example is insider threats, where compromises are caused by the intentional and unintentional misbehavior of insiders, such as employees, maintenance personnel, and system administrators. On the contrary, an attacker, either an individual or a state-sponsored group, has determined targets and can independently launch attacks of his own volition. Capable of tilting these disadvantages, proactive defense mechanisms have drawn increasing attention.

This dissertation develops proactive defense methods to mitigate the defender's four disadvantages. The methods in Chapters 3, 5, and 6 make it less likely for the attacks to succeed at any time and location. The defensive deception methods in Chapters 7 and 8 delay the adversarial lateral movement and yield threat intelligence, respectively. The incentive mechanisms in Chapters 9 and 10 facilitate the cooperation between the defender and insiders of different incentives.

1.5 Contributions

The contributions of this dissertation span multiple dimensions, which are summarized as follows.

1.5.1 Models and Frameworks

The proposed models and frameworks enable a quantitative design, avoid lengthy and expensive trial-and-error design procedures, and drastically increase the confidence level. In Chapter 3, we formulate a zero-sum dynamic game model to design protection mechanisms for large-scale interdependent CINs against cyber and physical attacks. In Chapter 7, we model the adversarial lateral movement in the enterprise network as a time-expanded network, where the additional temporal links connect the isolated spatial service links across a long time to reveal persistent attack paths explicitly.

Semi-Markov Decision Processs (SMDPs) are adopted explicitly or implicitly in Chapters 8, 11, and 12 to model the stochastic state transition and sojourn duration during the interactions of defenders, attackers, and users. Chapter 4 further considers a finite-horizon Semi-Markov Game (SMG) between the defender (i.e., plant operator) and the attacker to obtain the time-sensitive attack response strategy and the real-time risk assessment in nuclear power plants.

Dynamic Bayesian games are applied in Chapters 5 and 6 to model the attack-defender interaction and robot interactions, respectively, under deception. We further develop information design models in Chapters 9 and 10 to quantify insiders' incentives and determine the optimal incentive control mechanisms.

1.5.2 Theoretical Advances

Our works provide security guidance and insights based on a solid theoretical foundation. We briefly list some of them here. In Chapter 6, we derive a set of extended Riccati equations with cognitive coupling under the linear-quadratic setting and extrinsic belief dynamics. Moreover, we propose metrics, such as deceivability, reachability, and the price of deception, to evaluate the strategy design and the system performance under deception. In Chapter 7, the analysis of the long-term vulnerability under two heuristic honeypot policies illustrates that without proper mitigation strategies, vulnerability never decreases over stages and the target node is doom to be compromised given sufficient stages of adversarial lateral movement. Moreover, even under the improved honeypot strategies, a *vulnerability residue* exists; i.e., long-term vulnerability cannot be reduced to 0 and perfect security does not exist.

In Chapter 9, we create a theoretical underpinning for understanding trust, compliance, and satisfaction, which leads to scoring mechanisms of how compliant and persuadable an employee is. In Chapter 10, we develop a *separation principle* that decouples the reward design from the holistic design and an *equivalence principle* that turns the joint design of information and trust into the single unconstrained trust design. In Chapter 12, the integrated modeling and theoretical analysis lead to the Product Principle of Attention (PPoA), fundamental limits, and the tradeoff among crucial human and economic factors.

1.5.3 Computationally Efficient Algorithms

Algorithms enable us to design implementable technologies. Chapter 3 develops a scalable algorithm to approximate the optimal strategies for large-scale networks, which reduces the growth of computation complexity from exponential to polynomial. Analytical and Monte Carlo simulation-based algorithms in Chapter 4 enable the derivation of the following three risk metrics: the probability of the first arrival time at the undesirable states; the probability of arriving at the undesirable states before or at a specified time; and the probability distribution of system states at any time.

Chapter 5 and Chapter 6 propose offline and moving-horizon algorithms, respectively, to compute the Perfect Bayesian Nash Equilibrium (PBNE). In Chapter 7, to counter the curse of multiple attack paths, we propose an iterative algorithm and approximate the long-term vulnerability with the union bound for computationally efficient deployment of cognitive honeypots. In Chapter 9, we leverage the feedback of insiders' compliance status, the policy separability principle, and the set convexity to develop efficient incentive learning algorithms that are provably convergent in finite steps. Chapters 11 and 12 adopt adaptive algorithms to learn the optimal visual aid and attention management strategies, respectively.

1.5.4 Applications

Our works contribute to a large number of critical CPS application fields, including resilient interdependent CINs in Chapter 3, secure nuclear power plants in Chapter 4, deception-resistant robotics in Chapter 6, network security in Chapter 5, honeypot-driven security in Part IV, insider threat mitigation in Part V, and human-machine interaction in Part VI.

1.6 Outline and Organization

The rest of this dissertation is organized as follows. In Section 1.6, we categorize our works based on the following three types of vulnerabilities that DAISY aims to mitigate, i.e., posture-related vulnerabilities in Section 1.6.1, information-related vulnerabilities in Section 1.6.2, and human-related vulnerabilities in Section 1.6.3, respectively. We summarize the hierarchical structure of the dissertation concerning Chapters 3 to 12 in Table 1.2. Chapter 2 presents preliminaries on game theory for security, information design, and RL. Chapter 13 concludes the dissertation, proposes future directions, and provides broader insights.

1.6.1 Mitigation of Posture-Related Vulnerabilities

The class of posture-related vulnerabilities arises from the *criteria asymmetry* between an attacker and a defender; i.e., an attacker only needs to compromise one component at a single time to sabotage CPSs, while the criteria of security require a defender to protect the entire attacker surface during continuous operation time. Due to the disadvantage in security posture, the defender with limited resources cannot afford to prepare for all possible attacks. Part II aims to mitigate posture-related vulnerabilities in two CPS applications of large-scale interdependent CINs in Chapter 3 and nuclear power plant in Chapter 4, respectively.

Chapter 3 is based on series of works that focus on natural disasters [82, 84] and attackers [83, 85], respectively. The proposed defense policies strike a balance between prevention of and response to cascading failures. Building on the factored

Vulnerability of posture	Pt.2: mitigate the defender's time and space disadvantage	Ch.3: protecting large-scale interdependent CINs with expanded attack surface [82, 83, 84, 85]
		Ch.4: time-sensitive defense strategies for timely attack response [233, 234]
Vulnerability of information	Pt.3: deception countermeasures	Ch.5: zero-trust defense against APTs [87, 89, 90, 91, 232]
		Ch.6: robot deception defense [97]
	Pt.4: defensive deception design	Ch.7: cognitive honeypots to reduce vulnerability with high stealthiness, low roaming cost, and little interference [92]
		Ch.8: risk-averse, cost-effective, and time-efficient honeypot policies to gather threat intelligence [88]
Vulnerability of human	Pt.5: acquired vulnerability	Ch.9: improving insider compliance by zero-trust audit and strategic recommendations [99]
		Ch.10: a joint design of information, reward, and trust to elicit desirable user behaviors [96]
	Pt.6: innate vulnerability	Ch.11: attention enhancement to improve users' phishing recognition [86]
		Ch.12: alert and attention management to combat IDoS attacks [94, 98]

Table 1.2: Hierarchical structure of the dissertation to mitigate posture-related, information-related, and human-related vulnerabilities.

graph that exploits the interdependence structure of CINs, we further propose a computationally tractable approximation to protect large-scale networks with expanded attack surfaces. Chapter 4 is based on two works [233, 234], where the proposed time-sensitive defense strategy enables a timely response to undermine the attacker's time advantage.

1.6.2 Mitigation of Information-Related Vulnerabilities

The class of information-related vulnerabilities results from the *information asymmetry* between a defender and an attacker, especially when that attacker is deceptive and stealthy. The attacker has more information about the defender than the defender has about the attacker. The defender cannot make a meticulous plan to protect his assets if he cannot map out the attack paths. Part III and Part IV aim to tilt the information asymmetry by undermining the attacker's information advantage (i.e., counteracting adversarial deception) and establishing information advantage for the defender (i.e., designing defensive deception), respectively.

In Chapter 5 and Chapter 6, we develop dynamic Bayesian games to counteract deception from APT attackers and intelligent robots, respectively. Both the advanced attackers and AI-powered robots are modeled as rational decision-makers who keep private information to deceive the defender or other robots. The private information is modeled as a random variable and Bayesian learning is adopted to counteract deception. The deception is persistent as each decision-maker's private type remains unknown to others during the entire interaction process. Chapter 5 consists of the following works [87, 89, 90, 91, 232], and Chapter 6 is based on [97].

Besides compensating for information disadvantage, the defender can proactively create uncertainties and increase the attack cost by designing defensive deception. Part IV focuses on honeypots, one of the widely applied defensive deception technologies. Since advanced attackers, such as APTs, can identify the honeypots located at fixed machines that are segregated from the production system, we develop a cognitive honeypot strategy that reconfigures idle production nodes as honeypots at different stages based on the probability of service links and successful compromise in Chapter 7. Besides the main objective of reducing the target node's long-term vulnerability, we also consider the level of stealthiness, the probability of interference, and the cost of roaming as three tradeoffs. Chapters 7 and 8 are based on [92] and [88], respectively.

In Chapter 8, we develop risk-averse, cost-effective, and time-efficient honeypot engagement policies that lure the attacker into the target honeypot in the shortest time. The engagement with attackers can reveal a large range of Indicators of Compromise (IoCs) at a lower rate of false alarms and missed detection. However, it also introduces the risks of attackers identifying the honeypot setting, penetrating the production system, and a high implementation cost of persistent synthetic traffic generations. The developed engagement policies strike a balance between learning threat intelligence and reducing these risks.

1.6.3 Mitigation of Human-Related Vulnerabilities

The class of human-related vulnerabilities is the result of human misbehavior and cognition limitation. The vulnerabilities of all human groups in the cyber system can expose the system to cyber threats. Human users can unintentionally fall victim to phishing attacks (as shown in Chapter 11), self-interested insiders can intentionally break security rules for their convenience (as shown in Part V), and human operators and network administrators in charge of real-time monitoring and inspections of alerts and system status can suffer from alert fatigue (as shown in Chapter 12). We characterize exemplary cyber-physical attacks in Fig. 1.7 based on how much they exploit human vulnerabilities. A larger bubble size means a more frequent or a higher exploitation level of human vulnerabilities. Following the categorization of human vulnerability in Section 1.3.2, Part V and Part VI aim to mitigate acquired vulnerability of incentive misalignment and innate vulnerability

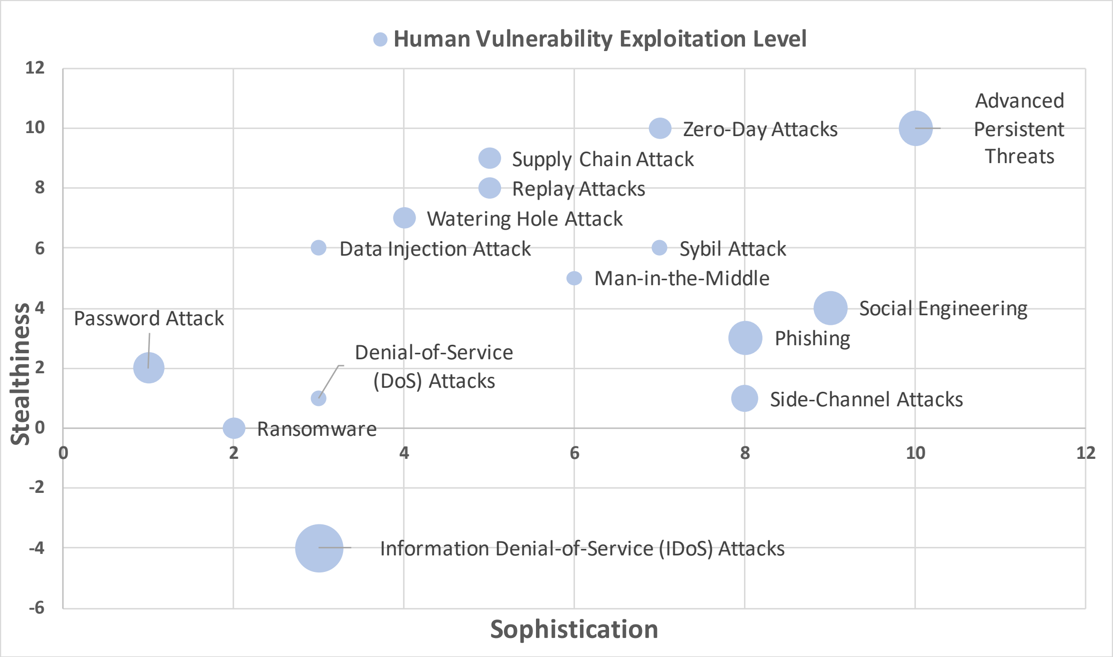


Figure 1.7: The threat landscape under different levels of human vulnerability exploitation. The size of the bubble increases as the attack is more likely to exploit human vulnerabilities. The x-axis has an increased sophistication in the attackers' Tactics, Techniques, and Procedures (TTPs). The y-axis has an increased stealthiness or a delay of detection. The Informational Denial-of-Service (IDoS) attack introduced in Chapter 12 receives a negative score of stealthiness as the attack aims to intentionally draw human operators' attention and increase their cognitive loads.

of bounded attention, respectively.

In Chapter 9, we develop ZETAR, a zero-trust audit and recommendation framework, to provide a quantitative approach to model incentives of the insiders and design customized and strategic recommendation policies to improve their compliance. In Chapter 10, we further complement the information design (e.g., the recommendation mechanism in Chapter 9) with the reward and trust designs and propose the duplicity game as a unified design framework. To achieve the joint design of information, reward structures, and trusts, the duplicity game consists of an information generator, an incentive modulator, and a trust manipulator, respectively. Chapters 9 and 10 are based on [99] and [96], respectively.

Chapter 11 focuses on *reactive attentional attacks* where the attackers attempt to evade the attention of defenders and users. Using phishing as a prototypical scenario, we develop a socio-technical solution called ADVERT to guide the users' attention to the right contents of the email and consequently improve their accuracy in phishing recognition. Chapter 12 focuses on *proactive attentional attacks* where attackers generate a large volume of feints to overload human operators and hide real attacks among feints. Based on the system-scientific human attention and alert response model, we have developed a Resilient and Adaptive Data-driven alert and Attention Management Strategy (RADAMS) to assist human operators in combating this new class of advanced attacks called the Informational Denial-of-Service (IDoS) attacks.

Chapter 11 is based on [86], and Chapter 12 is based on [94, 98]. Both chapters aim to develop human-assistive technologies as corrective compensation for attention-related vulnerability. Existing works fall into two regions of either *big model* via sophisticated modeling or *big data* via data analysis and learning. My works in Part VI bridge the gap between the two regions and pioneer a new research direction that uses system-scientific tools to distill *deep intelligence* (represented by *incisive laws and principles*) from data of human behaviors and biometrics. I have coined the terminology '*ho-da-tology*' for this new *human-centric*, *data-driven*, and *system-scientific* approach in Part VI.

Chapter 2

Modeling, Design, and Learning Theories for High-Confidence CPSs

In this chapter, we introduce the essential modeling, design, and learning tools adopted by DAISY to defend high-confidence CPSs. Game theory in Section 2.1 provides a formal paradigm to model the strategic interactions among rational players, predict the interaction outcomes, and design their equilibrium strategies. Since each agent is self-interested, the game equilibrium may not be satisfactory from the perspective of the defender or the entire system. Therefore, it is important for the system designer, e.g., the organizational defense team, to design the equilibrium and induce desirable behaviors. Such equilibrium design can be achieved by controlling payoff and allocation rules as shown in mechanism design [156] or by revising information available to other agents as shown in information design [18]. We focus on information design in Section 2.2. Finally, the payoff and information structures of attackers and users are usually not known exactly, especially in CPSs that contain a large number of complex components. Thus, it is often challenging or costly to characterize the exact attack model and the system model, which is referred to as the *curse of modeling*. In Section 2.3, we introduce feedback and Reinforcement Learning (RL) to address the challenges of incomplete information in game modeling. Many extensions will be presented in later chapters to enrich these baseline frameworks under different CPS applications, and we briefly introduce some of them in Section 2.4.

2.1 Security Games

Security games can be used to model the strategic interactions among cross-domain players in SoS. Fig. 2.1 presents the overall architecture of essential components in security games applied in uncertain and dynamic cyber-physical applications. In Section 2.1.1, we elaborate on these components, including players, actions, uncertainty, utilities, information, dynamics, and objectives. We formally define these components in Section 2.1.2, where we present three progressive classes of security game models based on different information structures and system dynamics.

2.1.1 Components of Security Games

We dissect security games into the following seven components and elaborate on each one in the context of cybersecurity. The goal is to provide a multi-dimension explanation of how these components characterize the strategic interaction between agents.

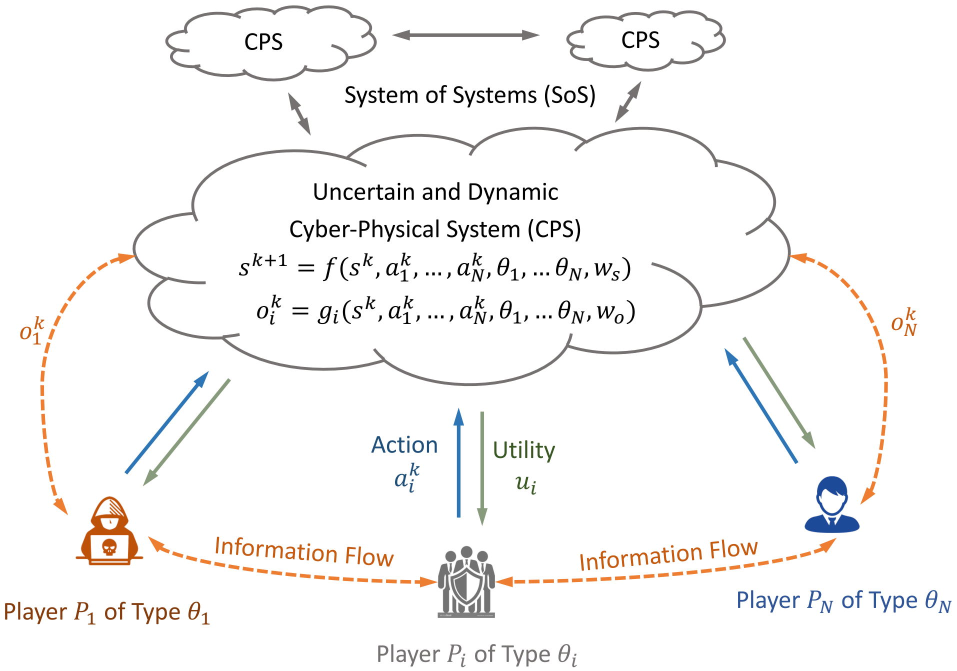

CPS
CPS
System of Systems (SoS)
Uncertain and Dynamic
Cyber-Physical System (CPS)
$s^{k+1} = f(s^k, a_1^k, \ldots, a_N^k, \theta_1, \ldots \theta_N, w_s)$
$o_i^k = g_i(s^k, a_1^k, \ldots, a_N^k, \theta_1, \ldots \theta_N, w_o)$
o_1^k
o_N^k
Action
a_i^k
Utility
u_i
Information Flow
Information Flow
Player P_1 of Type θ_1
Player P_N of Type θ_N
Player P_i of Type θ_i

Figure 2.1: The overall architecture of essential components in security games. The blue, green, and orange arrows represent the flows of actions, utilities, and information. The state transition dynamics and player P_i's observation are represented by functions f and g_i, respectively. On the one hand, type $\theta_i, i \in \mathcal{I}$, represents player P_i's internal uncertainty. On the other hand, w_s and w_o represent the external uncertainty in the control and observation processes, respectively.

Players

Following Section 1.4.2, a security game usually involves attackers, defenders, and users as players. We denote the number of players as N and index player $i \in \mathcal{I} := \{1, 2, \cdots, N\}$ as P_i.

Actions and Policies

Player $P_i, i \in \mathcal{I}$, has a set of actions, denoted as $\mathcal{A}_i$, where each action $a_i \in \mathcal{A}_i$ captures the behaviors of player P_i toward his/her attack or defense goal. We provide some examples of attackers', users', and defenders' actions as follows.

- Attacker's actions include adversarial reconnaissance, initial access, privilege escalation, defense evasion, credential access, lateral movement, command and control, and exfiltration, as shown in Fig. 1.3.
- User's actions in the security game context are usually confined to the ones that lead to insider threats, such as whether the user follows the security rules. Actions related to their normal assignments can be considered to characterize the security-usability tradeoff if the security measures negatively affect the normal operation.
- A defender can take the following actions toward attackers.
  - Prevention: data backup, sandbox, encryption, access control, and network segmentation.
  - Detection: audit, SSL/TLS inspection, antivirus, and exploit protection.
  - Response: disabling features, patching software, and restricting file and directory permissions.
  - Proactive defense: penetration tests, MTD, and honeypots.
- A defender can take the following actions toward users.
  - Reduce human-induced attack: password policies, multi-factor authentication, and behavior prevention on endpoint.
  - Increase security awareness: security training.
  - Increase compliance: penalty and reward.

We do not limit the agents' policies to be pure, i.e., instead of deciding which action to take, a player can decide the probability to take these actions. Intentionally introducing randomness can enlarge P_i's policy space from $\mathcal{A}_i$ to $\Delta\mathcal{A}_i$ to capture a more general case of interaction. When a player applies such *mixed strategies* in a CPS, the player can first roll a die according to the probability specified by the policies and then choose the realization as the action to implement.

Uncertainty

External uncertainty results from nature and unconsidered factors. As the system models and the attack models become increasingly complicated, we cannot consider all contributing factors. Thus, the unconsidered factors will result in randomness to the outcome of actions (represented by w_s in Fig. 2.1) and the observation of the current system state (represented by w_o in Fig. 2.1).

Internal uncertainty results from strategic and intelligent players when each player has different motivations, preferences, knowledge, capacities, and rationality. A common approach is to introduce a random variable θ_i as the 'type' of player P_i [75]. The support Θ_i and the prior distribution b_i^0 of the random variable θ_i are assumed to be *common knowledge*. Take insider threats as an example, player P_i is a user yet its type can be malicious or legitimate. From the statistics, the proportion of malicious users is public information. Thus, the prior distribution is commonly known by other players such as the defenders and the attackers. A player P_i's types can affect the transition and the observation processes, represented by the f and g_i functions, respectively, in Fig. 2.1.

Utilities

A player's utility is usually a function of all players' actions and types, the current system state, and the external uncertainty. The utility can be multi-dimensional and capture the following factors.

- Reward of threat intelligence, such as attack tools, TTP, and attack goals.

- Loss of money, information, and reputation.

- Cost of attack/defense actions, human resources, and insurance premium.

Information and Rationality

Due to the interaction of the multiple players, information structure in game theory can be complicated. One essential concept is *common knowledge* which is a special kind of knowledge for a group of agents. There is *common knowledge* of p in a group of agents G when all the agents in G know p they all know that they know p, they all know that they all know that they know p, and so on ad infinitum. Information in a security game specifies what a player knows, what a player knows that other players do not know, and what a player knows that other players are uncertain of.

- Information about other players' actions or policies: If the defender knows that his action or policy will be known by the attacker, and he knows that attackers will best respond to that action or policy, he can choose the optimal action accordingly. In this two-player game, the solution concept is Stackelberg equilibrium, and the defender as the leader usually has the *first-move advantage* over the attacker who is the follower. If all agents have to take actions or make policies without knowing others', then the solution concept is NE.

- Information about statistics of internal uncertainty: If the player knows the prior distribution of other players' types, then we can use Bayesian games to model it and obtain Bayesian Nash equilibrium.

- Information about external uncertainty: Players may obtain information or signals from the environment or other players to estimate the external uncertainty. If the signal is from other players, then deception can be introduced. Signaling games and information design games are usually used to model this scenario with the solution concept of Perfect Bayesian Nash Equilibrium (PBNE) and Bayesian Correlated Equilibrium (BCE), respectively.

Rationality specifies how players obtain their utilities, perceive the risk resulting from uncertainty, and react to signals and information. The benchmark security game models assume that all players have perfect rationality. Such assumption has been relaxed in game models with bounded rationality, such as level-k thinking, non-Bayesian update, and cumulative prospect theory for human factors.

Multi-agent interaction brings unique features and paradoxes, including the *curse of resources* (i.e., more resources reduces the player's utility), the *curse of information* (i.e., more information reduces the player's utility), and the curse of *rationality* (i.e., irrational actions brings the player a higher utility). The Braess's paradox [203], the winner's curse [212], and the impacts of bounded computational abilities [25] are three examples of these three curses.

Dynamic and Timing

Dynamic security games usually model the current system status (e.g., the user's privilege level and the location of the attack in the attack graph) as a vector denoted by $s^k \in \mathcal{S}$ in Fig. 2.1. The superscript k represents the time index that can be either discrete or continuous. The dynamic state transition function f can be stochastic. The state can be either fully observable, fully unobservable, or partially observable with uncertainty, as captured by function g_i in Fig. 2.1.

Objective Functions

We can categorize players' objective functions concerning the following three differences.

- Time differences
  - Players aim to maximize one-shot reward; e.g., an attacker aims to directly compromise a computer and implement ransomware.
  - Players aim to maximize long-term reward; e.g., a defender aims to protect valuable assets in the long run.
- Space differences
  - Players only consider local rewards or partial rewards of a complicated process. For example, a defender applies segmentation and only aims to protect its critical assets from attacks.
  - Players consider the global rewards or the complete rewards of the entire process. For example, APT attackers design their attack tools for the entire hacking stage.
- Uncertainty differences
  - Players aim to optimize the expected loss, which is applied if a player cares about the average performance under attacks.

– Players aim to mitigate the worst case, which is applied if a player wants to estimate the worst-case loss.

2.1.2 Classes of Security Games

In Section 2.1.2, we introduce three progressive classes of security game models to illustrate the basic game components in Section 2.1.1. First, we introduce static games with complete information as the baseline game-theoretic models and define the solution concept of NE. The baseline model is mainly used to model one-shot attacks. It assumes that the defender has a well understanding of the target system and the attacker's goals. Second, we extend the baseline model to dynamic models with complete information to capture multi-stage and multi-phase attacks. We introduce stochastic state transition models over discrete stages of finite or infinite horizons. The basic game elements are revised accordingly (e.g., the action set becomes state-dependent) and the solution concept of NE is extended to be Subgame Perfect Nash Equilibrium (SPNE), which requires *sequential rationality* across the stages. Third, we incorporate incomplete information into the dynamic game models to design responses when the attack is deceptive and stealthy. As the information, knowledge, and beliefs of the players change over time, we consolidate the *belief consistency* condition into SPNE and arrive at the solution concept of PBNE.

Static Models with Complete Information

The most basic games are static games with complete information. Among two-person complete-information static games, the most elementary type of zero-sum and nonzero-sum games are matrix and bimatrix games, respectively, as shown in

[14]. Following Section 2.1.1, we define a set of utility functions $u_i : \prod_{i=1}^{N} \mathcal{A}_i \mapsto \mathbb{R}$ to quantify the revenue of player $P_i, i \in \mathcal{I}$, upon the joint actions taken by all N players. For example, if an attacker chooses to compromise the main computer (i.e., $a_1 = a_1^{cm} \in \mathcal{A}_1$) and the defender chooses to use the backup computer (i.e., $a_2 = a_2^{ub} \in \mathcal{A}_2$), then the attacker spends costs to launch the attack yet does not achieve the attack goal. Thus, the attacker's utility $u_1(a_1^{cm}, a_2^{ub})$ is given by a negative value that quantifies the attack cost. The defender's utility $u_2(a_1^{cm}, a_2^{ub})$ is a positive value that quantifies the operational gain under the backup computer. Each player's action $a_i \in \mathcal{A}_i$ is unknown to other players before being implemented.

Due to the coupling effect of all players' actions on the interaction outcome measured by these utility functions, each player P_i needs to determine his/her action $a_i \in \mathcal{A}_i$ strategically. Due to the *common knowledge* assumption, a player may predict other players' actions based on the utility functions. Thus, each player P_i adopts a mixed strategy $\sigma_i \in \Delta\mathcal{A}_i$ to make his/her action less predictable. Let $\sigma_i(a_i) \in [0, 1]$ be the probability of player P_i taking action $a_i \in \mathcal{A}_i$ and $\sum_{a_i \in \mathcal{A}_i} \sigma_i(a_i) = 1$. Define shorthand notations $\sigma_{1:N} \in \Delta\mathcal{A}_{1:N}$ and $a_{1:N} \in \mathcal{A}_{1:N}$ as the tuple of N players' policies $(\sigma_1 \in \Delta\mathcal{A}_1, \sigma_2 \in \Delta\mathcal{A}_2, \cdots, \sigma_N \in \Delta\mathcal{A}_N)$ and actions $(a_1 \in \mathcal{A}_1, a_2 \in \mathcal{A}_2, \cdots, a_N \in \mathcal{A}_N)$, respectively. Then, his expected utility v_i is defined as

$$v_i(\sigma_{1:N}) := \sum_{a_{1:N} \in \mathcal{A}_{1:N}} \prod_{j \in \mathcal{I}} \sigma_j(a_j) u_i(a_{1:N}). \tag{2.1}$$

Each player P_i aims to choose the policy $\sigma_i^* \in \Delta\mathcal{A}_i$ that maximizes v_i, which leads to the NE defined in Definition 1. Let the shorthand notations a_{-i} and σ_{-i} denote the actions and strategies of players other than P_i, respectively.

Definition 1 (**Nash Equilibrium (NE)**). *The set of N players' policies $\sigma_{1:N}^* \in$*

$\Delta\mathcal{A}_{1:N}$ *comprises a mixed-strategy* NE *if*

$$v_i(\sigma_i^*, \sigma_{-i}^*) \geq v_i(\sigma_i, \sigma_{-i}^*), \forall \sigma_i \in \Delta\mathcal{A}_i, \forall i \in \mathcal{I}. \tag{2.2}$$

Based on the definition, no single player can benefit from deviating a NE when other players follow it. Thus, NE provides a reliable prediction of the interaction outcome and optimizes the design of response systems. By incorporating a random generator in the response system, we can automate the implementation of the defense action as a realization of the corresponding mixed strategy.

Dynamic Models with Complete Information

To incorporate system dynamics, we extend static game models to the following discrete-stage Markov game model (also known as the stochastic games [193]). The system status changes at discrete stages indexed by $k \in \mathcal{K} := \{1, 2, \cdots, K\}$, where the final stage K can be infinity. We define $\mathcal{S}$ as the finite set of states, and state $s^k \in \mathcal{S}$ represents the system status, e.g., whether a valve has failed, at stage $k \in \mathcal{K}$. Each player P_i's action set $\mathcal{A}_i^s$ at state $s \in \mathcal{S}$ is a subset of his/her pre-defined action set $\mathcal{A}_i$. For example, if the state $s \in \mathcal{S}$ indicates that a valve has failed, then the action to control the valve is not feasible, i.e., it belongs to $\mathcal{A}_i$ but not $\mathcal{A}_i^s$. Analogously, the utility function of P_i also depends on the state $s \in \mathcal{S}$ and all players' actions $a_i \in \mathcal{A}_i^s, i \in \mathcal{I}$, at state s, i.e., $u_i : \mathcal{S} \times \prod_{i=1}^N \mathcal{A}_i^s \mapsto \mathbb{R}$.

The state transition is stochastic and can depend on all players' actions. For example, when the attackers take no actions and the operators have set the improper (resp. proper) control parameter fors the PLC, then there is a high (resp. low) chance that the system will transit from the state of normal operation to the state of the core meltdown. We define a transition kernel $f : \mathcal{S} \times \mathcal{S} \times \prod_{i=1}^{N} \mathcal{A}_i^s$. Then, $f(s^{k+1}|s^k, a_{1:N}^k)$ represents the probability of the state transition from $s^k \in \mathcal{S}$ to $s^{k+1} \in \mathcal{S}$ under action tuple $a_i^k \in \mathcal{A}_i^{s^k}, i \in \mathcal{I}$.

Since the action at the current state can affect the transition kernel f and the state in the future, we build upon the expected utility in (2.1) to form the Cumulative Expected Utility (CEU) from any initial stage $k_0 \in \mathcal{K}$ in (2.3) as the new objective function for the dynamic games. Each player P_i's policy $\sigma_i^k : \mathcal{S} \mapsto \Delta\mathcal{A}_i^s$ is a mapping from the current state space to the distribution of the current action space. When K is infinite, we limit each player P_i to a stationary policy $\sigma_i^k = \sigma_i, \forall k \in \mathcal{K}$. Analogously, the shorthand notations $s^{k_0:K}$ and $\sigma_{1:N}^{k_0:K}$ represent the tuple $(s^{k_0}, \cdots, s^K)$ of states and the tuple $(\sigma_{1:N}^{k_0}, \cdots, \sigma_{1:N}^K)$ of N players' policies, respectively, from stage $k_0 \in \mathcal{K}$ to K. The discounted factor $\gamma^k \in (0, 1)$ penalizes the expected utility obtained in the future.

$$v_i(s^{k_0}, \sigma_{1:N}^{k_0:K}) := \mathbb{E}_{s^{k_0:K}} \left[\sum_{k=k_0}^{K} \gamma^k \cdot \sum_{a_{1:N}^k \in \mathcal{A}_{1:N}^{s^k}} \prod_{j \in \mathcal{I}} \sigma_j^k(a_j^k|s^k) u_i(s^k, a_{1:N}^k) \right], \forall i \in \mathcal{I}. \quad (2.3)$$

As each player P_i aims to maximize his CEU $v_i(s^{k_0}, \sigma_{1:N}^{k_0:K})$ at any initial stage $k_0 \in \mathcal{K}$, we define the solution concept of SPNE in Definition 2.

Definition 2 (**Subgame Perfect Nash Equilibrium** (**SPNE**))**.** *The set of N player's policies $\sigma_{1:N}^{*,k_0:K}$ comprises a SPNE if for all $k_1 \in \{k_0, \cdots, K\}$, we have*

$$v_i(s^{k_1}, \sigma_i^{*,k_1:K}, \sigma_{-i}^{*,k_1:K}) \geq v_i(s^{k_1}, \sigma_i^{k_1:K}, \sigma_{-i}^{*,k_1:K}), \forall \sigma_i^{k_1:K}, i \in \mathcal{I}. \quad (2.4)$$

Based on the definition, no players benefit from deviating the SPNE at any future stage $k_1 \in \{k_0, \cdots, K\}$ when other players follow the equilibrium. Using dynamic programming, we can write out the expectation of the future states $\mathbb{E}_{s^{k_0:K}}$ explicitly in (2.5) to compute the equilibrium policy σ_i^{*,k_0} at each stage $k_0 \in \mathcal{K}$.

$$v_i(s^{k_0}, \sigma_{1:N}^{*,k_0:K}) = \sum_{a_{1:N}^{k_0} \in \mathcal{A}_{1:N}^{s^{k_0}}} \Bigg[\prod_{j \in \mathcal{I}} \sigma_j^{*,k_0}(a_j^{k_0}|s^{k_0}) u_i(s^{k_0}, a_{1:N}^{k_0}) + \gamma^k \sum_{s^{k_0+1} \in \mathcal{S}} f(s^{k_0+1}|s^{k_0}, a_{1:N}^{k_0}) v_i(s^{k_0+1}, \sigma_{1:N}^{k_0+1:K}) \Bigg], \forall i \in \mathcal{I}. \qquad (2.5)$$

We can solve the system of N equations in (2.5) by backward induction and mathematical programming [14] if K is finite and infinite, respectively.

Dynamic Models with Incomplete Information

Following Section 2.1.1, we introduce dynamic Bayesian games to incorporate incomplete information and players' internal uncertainty in dynamic game models. We classify each player into different types based on their capacities, knowledge, and identities. Each player P_i's type $\theta_i \in \Theta_i$ is unknown to other players. The joint type $\theta_{1:N} := \{\theta_1, \cdots, \theta_N\}$ is assumed to be a random vector with distribution b. Then, each player P_i observing his type θ_i knows with probability $b_i^0(\theta_{-i}|\theta_i) := b^0(\theta_{-i}, \theta_i)$ that other players' types are θ_{-i}. Each player's utility $u_i : \mathcal{S} \times \prod_{i=1}^N \mathcal{A}_i^s \times \prod_{i=1}^N \Theta_i \mapsto \mathbb{R}$ depends on the state s^k, the joint action $a_{1:N}^k$, and the joint type $\theta_{1:N}$. For example, it costs less for professionals than amateurish defenders to take the same defense action and achieve the same protection level. Furthermore, the cost also depends on the attacker's sophistication level.

Each player gets access to the entire history of state (denoted as $s^{1:k}$) until the current stage $k \in \mathcal{K}$. Based on the history and his own type $\theta_i \in \Theta$, player P_i at stage k takes a *behavioral strategy* $\sigma_i^k : \prod_{k'=1}^k \mathcal{S} \times \Theta_i \mapsto \Delta \mathcal{A}_i^{s^k}$. Each player P_i can form a time-varying belief of other players' types based on the observed state history. Define b_i^k as P_i's belief of other players' types at stage $k \in \mathcal{K}$. Then, we have the following Bayesian belief update:

$$b_i^{k+1}(\theta_{-i}|s^{1:k+1}, \theta_i) = \frac{\Pr(s^{k+1}|\theta_{-i}, s^k, \theta_i) b_i^k(\theta_{-i}|s^{1:k}, \theta_i)}{\sum_{\bar{\theta}_{-i}} \Pr(s^{k+1}|\bar{\theta}_{-i}, s^k, \theta_i) b_i^k(\bar{\theta}_{-i}|s^{1:k}, \theta_i)}, \forall i \in \mathcal{N}, \tag{2.6}$$

where the transition probability $\Pr(s^{k+1}|\theta_{-i}, s^k, \theta_i)$ can be computed based on the transition kernel f and $\sigma_{1:N}^k$ (i.e., the mixed strategies of N players at stage k). Each player P_i' CEU from stage k_0 to K is defined as

$$v_i(s^{k_0:K}, \sigma_{1:N}^{k_0:K}) = \sum_{k=k_0}^{K} \mathbb{E}_{\{a_j^k \sim \sigma_j^k(\cdot|s^{1:k}, \theta_j)\}_{j \in \mathcal{N}}, \theta_{-i} \sim b_i^k(\cdot|s^{1:k}, \theta_i)} \left[u_i(s^k, a_{1:N}^k, \theta_{1:N}) \right]. \tag{2.7}$$

Each player aims to maximize his CEU while satisfying a *belief consistency* constraint that makes the posterior beliefs consistent with the player policies and the incomplete observations. The associated solution concept is called the PBNE in Definition 3. We illustrate players' private types and PBNE in Fig. 2.2 with two players and binary types.

Definition 3 (**Perfect Bayesian Nash Equilibrium** (**PBNE**))**.** *The set of N player's policies $\sigma_{1:N}^{*,k_0:K}$ constitutes a PBNE if for all $k_1 \in \{k_0, \cdots, K\}$, the following two conditions hold.*

1. *Sequential rationality:*

$$v_i(s^{k_1:K}, \sigma_i^{*,k_1:K}, \sigma_{-i}^{*,k_1:K}) \geq v_i(s^{k_1:K}, \sigma_i^{k_1:K}, \sigma_{-i}^{*,k_1:K}), \forall \sigma_i^{k_1:K}, i \in \mathcal{I}. \tag{2.8}$$

2. *Belief consistency in* (2.6).

Analogously, we can use dynamic programming to represent (2.7) iteratively

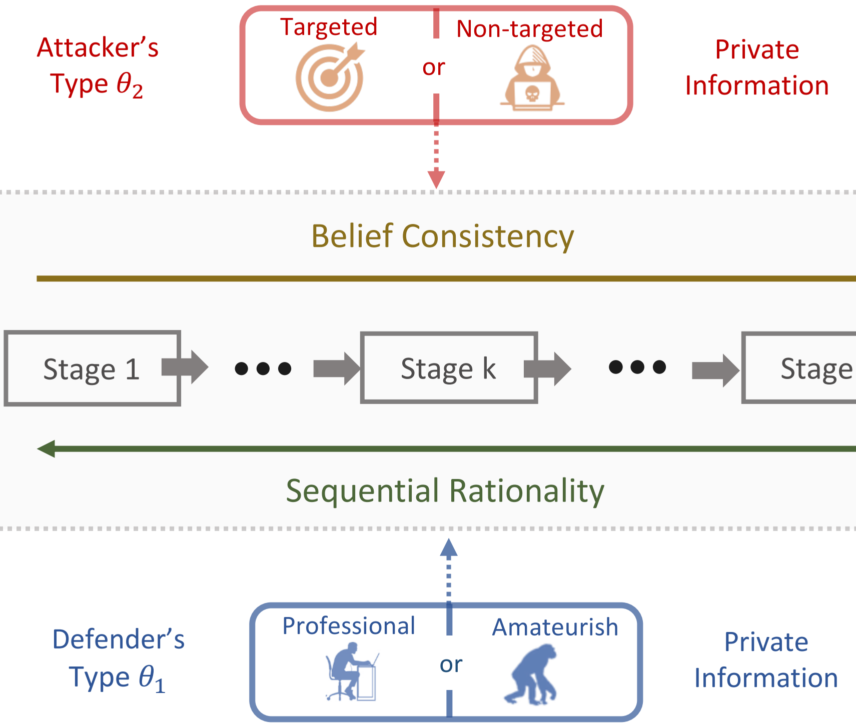


Figure 2.2: Illustration of players' private types and the solution concept of PBNE.

and then form mathematical programs to compute the PBNE strategy. The authors in [95] provide constructive proof for the convergence of continuous-type Bayesian games under a sequence of finer discretization schemes.

2.2 Information Design

Compared to the classical mechanism design [156] of payment and allocation rules, information design provides an affordable, scalable, and complementary way to change agents' beliefs in favor of the designer. We consider the information design framework that consists of N interacting agents and one designer [18]. When $N = 1$, the framework degenerates to the Bayesian persuasion framework [109].

We model the system uncertainty as a finite random variable X with support

$\mathcal{X}$. The distribution of X, i.e., $\psi(x) := \Pr(X = x), \forall x \in \mathcal{X}$, is *common knowledge* for all agents and the information designer, yet its realization is unknown to all. Based on the system uncertainty, each agent $i \in \mathcal{I}$ has a finite random variable $\theta_i \in \Theta_i$ according to the allocation distribution π, i.e.,

$$\pi : \mathcal{X} \mapsto \Theta := \prod_{i \in \mathcal{I}} \Theta_i. \tag{2.9}$$

The allocation distribution π is *common knowledge* to all, yet the realization of each θ_i is the private information of agent $i \in \mathcal{I}$. We assume that each agent truthfully reports his type θ_i to the designer. If not, the designer can incorporate mechanism design methods to make the truth-reporting strategy incentive-compatible for each agent. For example, the defender can send each agent some feedback based on their reported types.

Each agent i chooses his action a_i from a finite action set $\mathcal{A}_i$ and receives a utility $u_i(a_1, ..., a_N, x)$ which is determined by all agents' actions and the realization $x \in \mathcal{X}$ of the system uncertainty X. On the other hand, the designer's utility $u_0(a_1, ..., a_N, x)$ usually represents the system utility, and is also determined by all agents' actions and the realization of the system uncertainty. Define the joint action set $\mathcal{A} := \prod_{i=1}^{N} \mathcal{A}_i$. Then, the agents' and the designer's utilities are the mappings $u_i, u_0 : \mathcal{A} \times \mathcal{X} \mapsto \mathbb{R}$, respectively.

Without the information designer's interference, the equilibrium concept we consider for this incomplete information game is the BCE defined as follows. Note that σ is contained implicitly in the conditional probability term $\Pr(a_{-i}, x | a_i, \theta_i)$ as shown in (2.13).

Definition 4 (**Bayesian Correlated Equilibrium** (**BCE**)**).** *Decision rule* σ :

$\Theta \mapsto \Delta\mathcal{A}$ *is a BCE if* $\forall i \in \mathcal{I}, \theta_i \in \Theta_i, a_i \in \mathcal{A}_i$,

$$\sum_{x}\sum_{a_{-i}} \Pr(a_{-i}, x|a_i, \theta_i)\Big[u_i(a_i, a_{-i}, x) - u_i(a_i', a_{-i}, x)\Big] \geq 0, \forall a_i' \in \mathcal{A}_i. \qquad (2.10)$$

From Definition 4, we know that BCE is a mixed strategy. Equation (2.10) shows that when the BCE decision rule σ is revealed to all agents, and agent i obtains his action a_i as the realization of the mixed-strategy σ, any other deviation action a_i' does not bring any benefit to him on average.

Since BCE may not be unique, we define Σ as the set of all BCEs and the designer may be able to choose a proper σ^* in his favor from all the feasible BCEs in Σ. One way to choose the policy σ^* is by designing a signal $m \in \mathcal{M}$ and revealing it to all agents to affect their behaviors. Basically, we assume the designer has the freedom to pick any signal structure χ based on the reported types of all agents, i.e., $\chi : \Theta \mapsto \mathcal{M}$. If χ is *common knowledge* to all, then the agent can update their posterior belief from $\Pr(a_{-i}, x|a_i, \theta_i)$ to $\Pr(a_{-i}, x|a_i, \theta_i, m)$ based on the signal realization $m \in \mathcal{M}$.

The authors in [18] proves a version of *revelation principle* in the information design as an analogy to the *revelation principle* in mechanism design. That is, we can limit our attention to information structures where the signal space is set equal to the action space without loss of generality[1]. Then the signals can be interpreted as the designer's action recommendations to the agents. Define the designer's

[1] The term 'without loss of generality' means that the designer cannot further improve his expected utility for any other feasible signal structures.

objective function as his expected utility under the recommended policy σ; i.e.,

$$\begin{aligned} J(\sigma) :=& \mathbb{E}_{a\sim\sigma, x\sim\psi}[u_0(a_1, ..., a_N, x)] \\ =& \sum_{x\in\mathcal{X}} \sum_{\theta\in\Theta} \sum_{a\in\mathcal{A}} [\psi(x)\pi(\theta|x)\sigma(a_1, ..., a_N|\theta)u_0(a_1, ..., a_N, x)]. \end{aligned} \tag{2.11}$$

Then, the information design problem can be formulated as the following linear program $\max_{\sigma\in\Sigma} J(\sigma)$, i.e., the designer picks the optimal BCE

$$\sigma^* = arg \max_{\sigma\in\Sigma} J(\sigma) \tag{2.12}$$

from the BCE set Σ to maximize the social utility $J^* = J(\sigma^*)$.

To simplify (2.10), we use Bayesian rule on the conditional probability, i.e.,

$$\begin{aligned} \Pr(a_{-i}, x|a_i, \theta_i) &= \sum_{\theta_{-i}} \Pr(a_{-i}, \theta_{-i}, x|a_i, \theta_i) \\ &= \sum_{\theta_{-i}} \frac{\Pr(a_{-i}, \theta_{-i}, x, a_i, \theta_i)}{\sum_{a_{-i}, \theta_{-i}, x} \Pr(a_{-i}, \theta_{-i}, x, a_i, \theta_i)} \\ &= \sum_{\theta_{-i}} \frac{\Pr(x)\Pr(\theta|x)\sigma(a_1, ..., a_N|\theta)}{\sum_{a_{-i}, \theta_{-i}, x} \Pr(a_{-i}, \theta_{-i}, x, a_i, \theta_i)}. \end{aligned} \tag{2.13}$$

The denominator $\sum_{a_{-i},\theta_{-i},x} \Pr(a_{-i}, \theta_{-i}, x, a_i, \theta_i)$ in (2.13) cancels out in (2.10), which makes all constraints linear in the decision variable σ. The information design problem can be formulated as the following Linear Program (LP):

$$\begin{aligned} & \max_{\sigma} \sum_{x} \sum_{\theta} \sum_{a} \psi(x)\pi(\theta|x)\sigma(a_1, ..., a_N|\theta)u_0(a_1, ..., a_N, x) \\ \text{s.t. } & \sum_{x} \sum_{a_{-i}} \sum_{\theta_{-i}} \psi(x)\pi(\theta|x)\sigma(a_1, ..., a_N|\theta) \Big[u_i(a_i, a_{-i}, x) \\ & - u_i(a_i', a_{-i}, x) \Big] \geq 0, \forall i \in \{1, \cdots, N\}, \theta_i \in \Theta_i, a_i, a_i' \in \mathcal{A}_i. \end{aligned} \tag{2.14}$$

2.3 Reinforcement Learning

In Section 2.3, we introduce efficient learning methods for adaptive and autonomous cyber-physical defense. Due to the adversarial deception, external noises, and the absent knowledge of the other players' behaviors and goals, defense schemes can possess three progressive levels of information restrictions, i.e., the parameter uncertainty, the payoff uncertainty, and the environmental uncertainty [93]. Different learning schemes can be adopted for varied information restrictions and application scenarios. As shown in Section 2.3.1, these learning schemes share the same feedback structure. RL is an important gathering of algorithms that epitomize the feedback architecture. In Section 2.3.2, we introduce Markov Decision Process (MDP) and the associated Q-learning as a representative RL scheme.

2.3.1 Feedback Structure

The feedback structure in the CPS context contains four stages of system operation, monitoring, decision-making, and response, as shown in Fig. 2.3. Monitoring aims to acquire information about the system as well as the footprints of the attacker. Decision-making builds on the acquired information to infer the attack behaviors and design the optimal resilience strategies. Response reconfigures the system according to the optimal strategy by adapting the system parameters and attributes to unknown threats.

The feedback loop of monitoring, decision-making, and response establishes an adaptive and dynamic system architecture for high-confidence CPSs. RL is an important gathering of algorithms that epitomize the feedback architecture to provide dynamic and sequential responses to attacks with limited or without

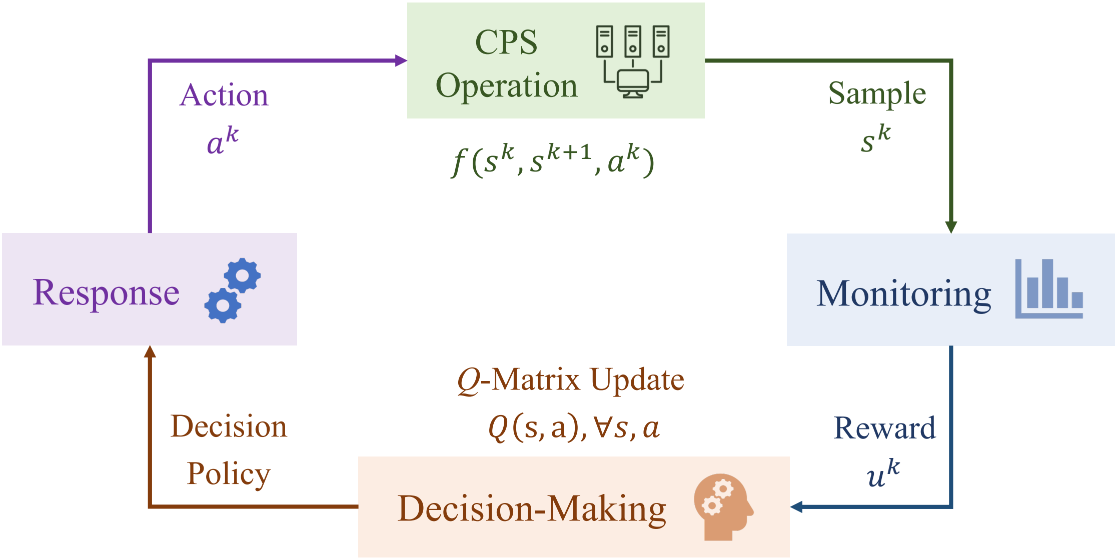


Figure 2.3: Feedback structure contains four stages of system operation, monitoring, decision-making, and response. During the online operation of a CPS, information is persistently collected through monitoring and analyzed to formulate the defense decisions. Then, actions are taken to adjust the CPS operation.

prior knowledge of the environment and the attacker. We provide a sketch of the Q-learning algorithm in Section 2.3.2 to illustrate the essence of RL.

2.3.2 MDP and Q-learning

An MDP is denoted by a 5-tuple $\langle \mathcal{S}, \mathcal{A}, u, f, \gamma \rangle$, where $\mathcal{S}$ is the state space containing all possible states of the cyber system. The action space $\mathcal{A}$ denotes the actions available for the defender to protect the cyber system, recover from the damage caused by attacks, or mitigate the effect of attacks. The reward u depends on the current and/or the next security states, and the current action, which is usually denoted by a function that maps $\mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}_+$. The transition kernel f defines the rule of how the system state evolves based on the actions taken by the defender. The discount factor $\gamma \in [0, 1]$ is a weighting factor that assigns more weight to current rewards than future rewards. The goal of an MDP problem is to find a policy $\pi : \mathcal{S} \rightarrow \mathcal{A}$ to maximize a certain form of the accumulative rewards $\sum_{k=0}^{\infty}(\gamma)^k u^k$ over time.

MDPs are generic modeling tools that can model the dynamic and feedback nature of various types of CPSs. Conventional MDP approaches require the knowledge of the transition probability and the explicit definition of the reward function to find an appropriate strategy. Since such information is usually prohibitive to obtain in practice, RL is introduced to solve the MDP problem without the knowledge of the transition kernel f and the reward function u. Instead of utilizing a prior known transition kernel and reward function, RL agent interacts with the environment, obtains sequences of states $\{s^k\}_{k\in\mathbb{Z}}$ and rewards $\{u^k\}_{k\in\mathbb{Z}}$, and learns to take the optimal sequence of actions $\{a^k\}_{k\in\mathbb{Z}}$, as shown in Fig. 2.3.

Q-learning is one of the most common value-based RL methods. It employs a value-action function $Q : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}+$, which is also referred to as the Q function. The goal is to find the optimal Q-values that satisfy the Bellman equation [19]

$$Q(s,a) = u(s,a) + \gamma \sum_{s'} f(s,s',a) \min_{a'} Q(s',a'), \quad \text{for } s \in \mathcal{S}, a \in \mathcal{A}, \tag{2.15}$$

where $f(s,s',a)$ is the probability that the state of the cyber system at the next step is s' given the current state s and the current action a. Without the knowledge of the transition probability f and the reward function u, the RL agent can update its Q-values by interacting with the environment:

$$Q^{k+1}(s^k,a^k) = Q^k(s^k,a^k) + \alpha^k \cdot \left[\gamma \min_{a'} Q^k(s^{k+1},a') + u^k - Q^k(s^k,a^k)\right], \tag{2.16}$$

where the sequences of states $\{s^k\}_{k\in\mathbb{Z}}$, rewards $\{u^k\}_{k\in\mathbb{Z}}$ are from the environment, while the sequence of actions $\{a^k\}_{k\in\mathbb{Z}}$ are chosen by the agent.

We briefly discuss the challenges and the related future directions about RL in defending high-confidence CPSs as follows. First, it is important to deal with system and performance constraints in the learning process. On the one hand, cyber systems have many system constraints that need to be taken into account explicitly. For example, certain addresses or functions that are not allowed or undesirable when configuring the MTD. On the other hand, the performance of the cyber systems can impact the performance of the physical systems that they serve. Hence, the requirement on the physical system performance naturally imposes a constraint on the performance of the cyber systems. A second challenge is to improve the learning speed. Fast learning would enable a speedier and more resilient response to the attack to restore the CPSs after an attack. To achieve it, we would need to resort to control-theoretic ideas, such as optimal control [116] and adaptive control theory [9], and leverage recent advances in RL to speed up the convergence rate or improve the finite-time learning performances. A third challenge is to deal with the nonstationarity of the CPSs. The classical RL algorithms assume that the environment is stationary and ergodic, which may not hold in many CPS applications. For example, the attack surface may grow when the system is connected with other nodes or used by new users. There is a need to develop nonstationary RL schemes to guarantee performance in a finite horizon.

2.4 Extensions and Applications

In Sections 2.1, 2.2, and 2.3, we review the basic tools for modeling, design, and learning, respectively. Many extensions will be presented in later chapters to enrich these baseline frameworks under different CPS applications. We briefly introduce some of them as follows.

In Section 2.1.1, while players optimize their objective functions, they also need to consider the constraints of time, resources, and stealthiness. For example, in Chapter 7, since the vulnerability may only exist for certain time, attacks exploiting the time window undergo time constraints. In Chapter 12, the human operators have limited attention capacities and are under cognition constraints. In Chapters 5 and 8, APT attackers and honeypots are restricted to be stealthy, respectively.

Following Section 2.1.2, other widely used security game models include differential games [56], signaling games [34] (a variant of two-stage dynamic Bayesian games), evolutionary games [76], and consolidated games that incorporate the above games as components. We select 14 works[2] and position them in Fig. 2.4 based on the sophistication levels of attacks and defense methods considered in the game model. We evaluate the attacks' sophistication levels based on their adaptiveness, stealthiness, and persistence. We evaluate the defense's sophistication levels based on the defenders' strategic level, proactiveness, and adaptiveness. We further characterize these game models based on whether they incorporate Deception Technologies (DT) and Human Factors (HF).

Section 2.2 focuses on designing information. On the one hand, we can extend the information design of one random variable into a chain of random variables, as shown in ZETAR framework in Chapter 9. On the other hand, as shown in the duplicity game framework in Chapter 10, the players (or the information receivers)

[2] These 14 works are the *Flipit* Game [215], the consolidated game [166], the signaling game with evidence [167], the signaling game for compliance [23], the time-sensitive stochastic game [233], the incomplete-information stochastic game [78], the information design game [96], the dynamic Bayesian game [91], the differential game [226], the Bayesian Stackelberg game [219], the matrix game [1], the evolutionary game [79], the hyper game [220], and the matrix game [3].

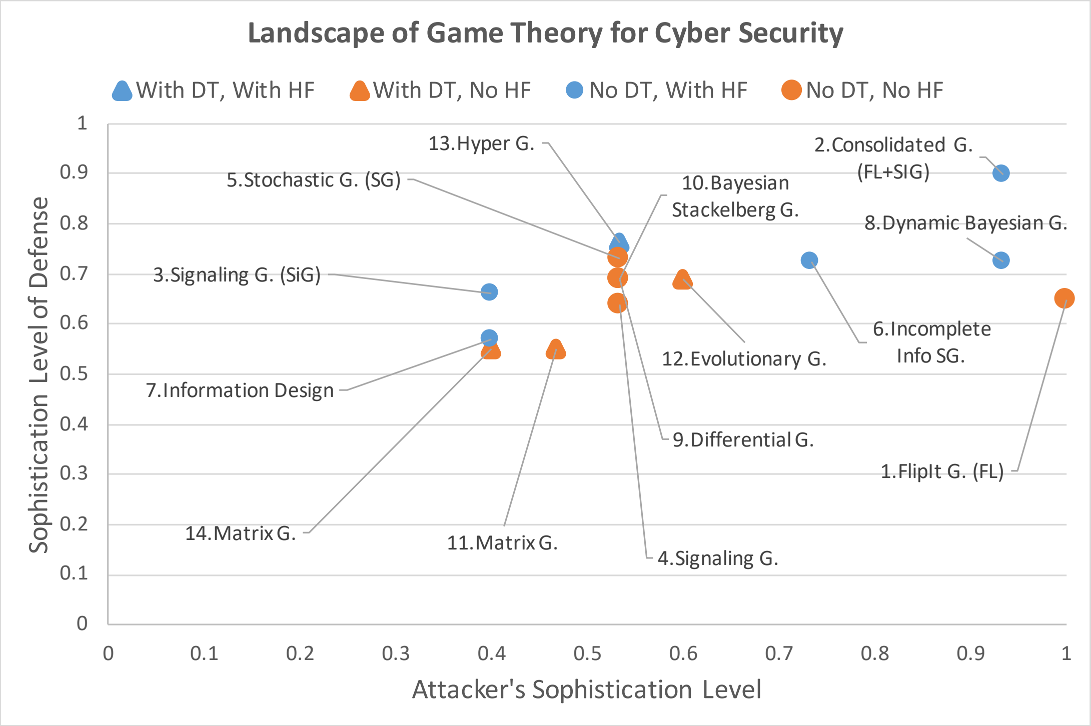


Figure 2.4: The landscape of security games concerning the sophistication levels of attacks and defense methods in x-axis and y-axis, respectively. We use triangles and circles to distinguish whether Deception Technologies (DT) are incorporated or not, respectively. We use blue and orange to distinguish whether Human Factors (HF) are incorporated or not, respectively.

can be of multiple types, and the designer can jointly design information, reward structure, and players' trust.

In Section 2.3.1, we present the standard feedback architecture. This architecture can be enriched and extended to several more sophisticated ones. One is the nested feedback loops, where one feedback loop is coupled to another feedback loop. This architecture is useful to separate and then fuse the learning of distinct system components of a CPS. For example, one feedback loop is used to acquire the attack footprint and learn its intent and capabilities. In contrast, the other feedback loop is used to acquire information regarding its system state. The two feedback loops can be fused for making online defense decisions in response to an unknown threat. Another architecture is a mixture of feedback and open-loop structures. Leveraging the ideas from moving-horizon control and estimation, we can make a moving-horizon plan by looking W stages into the future and optimizing for the W stages-to-go. This approach would require an open-loop prediction of the system under the attack, feedback-driven sensing of the environment, and a reasoning of the optimal moving-horizon strategies.

In Section 2.3.2, MDPs can be extended to many other frameworks. If the state is not fully observable but through an observation kernel g as shown in Fig. 2.1, then an MDP extends to a Partially Observable Markov Decision Process (POMDP). If there are multiple decision-makers, then a MDP extends to a stochastic game or a Markov game. If the transition happens at a random time that depends on the current and next state and the action, the MDP and the Markov game are extended to the Semi-Markov Decision Process (SMDP) and the Semi-Markov Game (SMG), respectively. We explore more about SMG in Chapter 4 and SMDP in Chapters 8 and 12, respectively.

Having been actively studied for decades, RL has a rich universe of algorithms that help the agent find a satisfactory policy. They can be classified as the model-based and the model-free RL based on whether the agent attempts to predict the environment parameters. The model-free RL also has two main categories for optimizing the policy: value-based methods and policy-based methods. Q-learning in Section 2.3.2 is a representative value-based model-free RL method.

Part II

Dynamic Protection of Critical Infrastructures

Chapter 3

Prevention and Response of Cascading Failures in Large-Scale Interdependent CINs

Following Section 1.3.2, the complex and large-scale interconnections between various critical infrastructure sectors illustrated in Fig. 1.2 make the System of Systems (SoS) vulnerable to natural disasters and cyber-physical attacks. As a result, the failure of one component can lead to a cascading failure over multiple infrastructures. To mitigate such cyber-physical threats, it is essential to design effective defense mechanisms to harden both the cyber and physical security at the nodes of the infrastructure to protect them from failures. To this end, we capture the system behaviors of the Critical Infrastructure Networks (CINs) under malicious attacks and the protection strategies by a zero-sum game. We further propose a computationally tractable approximation for large-scale networks which builds on the factored graph that exploits the dependency structure of the nodes of

CINs and the approximate dynamic programming tools for stochastic games. This work focuses on a localized information structure and the single-controller game solvable by linear programming. Numerical results illustrate the proper tradeoff of the approximation accuracy and computation complexity in the new design paradigm and show the proactive security at the time of unanticipated attacks.

3.1 Mathematical Model

This section introduces in Subsection 3.1.1 a zero-sum Markov game model over interdependent CINs to understand the interactions between an attacker and a defender at the nodes of infrastructures. The solution concept of the saddle-point equilibrium strategies is presented in Subsection 3.1.2 and the computational issues of the equilibrium is discussed in 3.1.3.

3.1.1 Network Game Model

The dynamic and complex CINs can be represented by nodes and links. For example, in an electric power system, a node can be a load bus or a generator and the links represent the transmission lines. Similarly, in a water distribution system, a node represents a source of water supply, storage or users, and the links can represent pipes for water delivery. Consider a system of I interdependent infrastructures. Let $\mathcal{G}^i = (\mathcal{N}^i, \mathcal{E}^i)$ be the graph representation of infrastructure $i \in \mathcal{I} := \{1, 2, \cdots, I\}$, where $\mathcal{N}^i = \{n_1^i, n_2^i, \cdots, n_{m_i}^i\}$ is the set of m_i nodes in the infrastructure and $\mathcal{E}^i = \{e_{j,k}^i\}$ is the set of directed links connecting nodes n_j^i and n_k^i. The directed link between two nodes indicates either physical, cyber or logical influences from one node to the other. For example, the state of node n_j^i in the electric power system can influence the state of node n_k^i through the physical connection or the market pricing. The dependencies across the infrastructures can be captured by adding interlinks. Let $\mathcal{E}^{i,j}$ be the set of directed interlinks between nodes in infrastructure i and infrastructure j. In particular, let $\varepsilon_{n_k^i, n_l^j} \in \mathcal{E}^{i,j}$ denote the interlink between n_k^i and n_l^j. Hence, the composed network can be represented by the graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, where $\mathcal{N} = \cup_{i=1}^{I} \mathcal{N}^i$ and $\mathcal{E} = \left(\cup_{i=1}^{I} \mathcal{E}^i\right) \bigcup \left(\cup_{i \neq j} \mathcal{E}^{i,j}\right)$.

Denote by $X_j^i \in \mathcal{X}_j^i$ the state of node n_j^i that can take values in the state space $\mathcal{X}_j^i$. We let $\mathcal{X}_j^i = \{0, 1\}$ be binary random variables for all $i = 1, 2, \cdots, I$ and $j \in \mathcal{N}^i$. Here, $X_j^i = 1$ means that node n_j^i is functional in a normal mode; $X_j^i = 0$ indicates that node n_j^i is in a failure mode. The state of infrastructure i can be thus denoted by $X^i = (X_1^i, X_2^i, \cdots, X_{m_i}^i) \in \mathcal{X}^i := \prod_{j=1}^{m_i} \mathcal{X}_j^i$ and the state of the whole system is denoted by $X = (X^1, X^2, \cdots, X^I) \in \prod_{i=1}^{I} \mathcal{X}^i$. The state transition of a node n_j^i from state $x_j^{i\prime} \in \mathcal{X}_j^i$ to state $x_j^i \in \mathcal{X}_j^i$ is governed by a stochastic kernel $\mathrm{Pr}_{i,j}(x_j^{i\prime} | x, d_j^i, a_j^i) := \mathrm{Pr}(X_j^i = x_j^{i\prime} | X = x, d_j^i, a_j^i)$, which depends on the protection policy $d_j^i \in \mathcal{D}_j^i$ adopted at node n_j^i as well as the adversarial behavior $a_j^i \in \mathcal{A}_j^i$, where $\mathcal{D}_j^i, \mathcal{A}_j^i$ are feasible sets for the infrastructure protection and the adversary, respectively. The state transition of a node depends on the entire system state of the interdependent infrastructure. It, in fact, captures the interdependencies between nodes in one infrastructure and across infrastructures. The infrastructure protection team or defender determine the protection policy with the goal of hardening the security and improving the resilience of the interdependent infrastructure. On the other hand, an adversary aims to create damage on the nodes that he can compromise and inflict maximum damage on the infrastructure in a stealthy manner, e.g., creating cascading and wide-area failures. Let $\mathcal{M}_a^i \subseteq \mathcal{N}^i$ and $\mathcal{M}_d^i \subseteq \mathcal{N}^i$ be the set of nodes that an adversary can control and the system action vector of the adversary is $\mathbf{a} = (a_j^i)_{j \in \mathcal{M}_a^i, i \in \mathcal{I}} \in \mathcal{A} := \prod_{i \in \mathcal{I}} \prod_{j \in \mathcal{N}^i} A_j^i$ with $|\mathcal{M}_a^i| = \bar{m}_{a,i}$. The system action vector for infrastructure protection is $\mathbf{d} = (d_j^i)_{j \in \mathcal{M}_d^i, i \in \mathcal{I}} \in \mathcal{D} := \prod_{i \in \mathcal{I}} \prod_{j \in \mathcal{N}^i} D_j^i$ with $|\mathcal{M}_d^i| = \bar{m}_{d,i}$. At every time $t = 1, 2, \cdots$, the pair of action profiles $(\mathbf{d}_t, \mathbf{a}_t)$ taken at t and the kernel Pr defined later determine the evolution of the system state trajectory. Here, we use add subscript t to denote the action taken time t. The conflicting objective of both players can be captured by a long-term cost J over an infinite horizon:

$$J := \sum_{i \in \mathcal{I}, j \in \mathcal{N}^i} \sum_{t=1}^{\infty} \gamma^t c_j^i(X_t, d_{j,t}^i, a_{j,t}^i), \tag{3.1}$$

where $\gamma \in (0, 1)$ is a discount factor; $X_t \in \mathcal{X}$ is the system state at time t; $c_j^i : \mathcal{X} \times \mathcal{D}_j^i \times \mathcal{A}_j^i \to \mathbb{R}_+$ is the stage cost function of the node n_j^i. Let $\mathcal{U}_j^i, \mathcal{V}_j^i$ be the sets of admissible strategies for the infrastructure and the adversary, respectively. Here, we consider a feedback protection policy $\mu_j^i \in \mathcal{U}_j^i$ as a function of the information structure $F_{j,t}^i$, i.e., $d_{j,t}^i = \mu_j^i(F_{j,t}^i)$. Likewise, we consider the same class of policies for the adversary, i.e., $a_{j,t}^i = \nu_j^i(F_{j,t}^i), \nu_j^i \in \mathcal{V}_j^i$.

The policy can take different forms depending on the information structure. For example, if $F_{j,t}^i = X_t$, i.e., each node can observe the whole state across infrastructures, then the policy is a global stationary policy, denoted by $\mu_j^{i,\text{GS}} \in \mathcal{U}_j^{i,\text{GS}}$, where $\mathcal{U}_j^{i,\text{GS}}$ is the set of all admissible global stationary policies.. If $F_{j,t}^i = X_{j,t}^i$, i.e., each node can only observe its local state, then the policy is a local stationary policy, denoted by $\mu_j^{i,\text{LS}} \in \mathcal{U}_j^{i,\text{LS}}$, where $\mathcal{U}_j^{i,\text{LS}}$ is the set of all admissible local stationary policies. If $F_{j,t}^i = X_t^i$, i.e., each node can observe the infrastructure-wide state, then the policy is an infrastructure-dependent stationary policy, denoted by $\mu_j^{i,\text{ID}} \in \mathcal{U}_j^{i,\text{ID}}$, where $\mathcal{U}_j^{i,\text{ID}}$ is the set of all admissible infrastructure-dependent stationary policies. Similarly, an adversary chooses a policy $\nu^i_{j,t}$, i.e., $a^i_{j,t} = \nu^i_j(F^i_{j,t})$. Denote by $\mu^i = (\mu^i_1, \mu^i_2, \cdots, \mu^i_{m_i})$, $\nu^i = (\nu^i_1, \nu^i_2, \cdots, \nu^i_{m_i})$ the protection and attack policies for infrastructure i, respectively, and let $\boldsymbol{\mu} = (\mu^1, \mu^2, \cdots, \mu^I)$ and $\boldsymbol{\nu} = (\nu^1, \nu^2, \cdots, \nu^I)$. Note that although both policies are determined only by the information structure and are independent of each other, the total cost function J depends on them both because of the coupling of the system stage cost $c(X_t, \mathbf{d}, \mathbf{a}) := \sum_{i,j} c^i_j(X_t, d^i_{j,t}, a^i_{j,t})$ and the system state transition probability $\Pr(X' = x'|X = x, \mathbf{d}, \mathbf{a}) := \prod_{i \in \mathcal{I}, j \in \mathcal{N}^i} \Pr_{i,j}(x^{i\prime}_j | x, d^i_j, a^i_j)$. Therefore, with $\mathcal{U} = \prod_{i \in \mathcal{I}, j \in \mathcal{N}_i} \mathcal{U}^i_j$ and $\mathcal{V} = \prod_{i \in \mathcal{I}, j \in \mathcal{N}_i} \mathcal{V}^i_j$, the total cost function $J : \mathcal{X} \times \mathcal{U} \times \mathcal{V} \to \mathbb{R}_+$ starting at initial state x^0 can be written as the expectation of the system stage cost regarding the system state transition probability, i.e.,

$$J(x^0, \boldsymbol{\mu}, \boldsymbol{\nu}) := \sum_{t=0}^{\infty} \gamma^t E_{\boldsymbol{\mu}, \boldsymbol{\nu}, x^0}[c(X_t, \mathbf{d}, \mathbf{a})]. \tag{3.2}$$

Remark 1. *Notice that there is a difference between policy* $\boldsymbol{\mu}, \boldsymbol{\nu}$ *and action* $\mathbf{d}, \mathbf{a}$. *A policy or strategy is a mapping and an action is the outcome of the mapping. Besides, since the information structure is the state information available to attackers or defenders, we can abstract it from the entire state information* X_t *at time* t. *Given a policy and an information structure, we can uniquely determine the action. Therefore, we write* $\mathbf{d}, \mathbf{a}$ *instead of* $\boldsymbol{\mu}, \boldsymbol{\nu}$ *in the Right-Hand Side (RHS) of* (3.2). *We use the same terminology in the following equations such as* (3.6) *where the solution provides us the optimal action pair* $\mathbf{d}^*, \mathbf{a}^*$ *at every state* x. *With the knowledge of the mapping outcome and corresponding information structure as the input of the mapping, the policy functions* $\boldsymbol{\mu}^*$, $\boldsymbol{\nu}^{\boldsymbol{*}}$ *are uniquely defined.*

Hence a security strategy for the infrastructure protection achieves the optimal solution $J^*(x^0)$ to the following minimax problem, which endeavors to minimize the system cost under the worst attacking situation $\max_{\boldsymbol{\nu}\in\mathcal{V}} J(x^0, \boldsymbol{\mu}, \boldsymbol{\nu})$, i.e.,

$$J^*(x^0) = \min_{\boldsymbol{\mu}\in\mathcal{U}} \max_{\boldsymbol{\nu}\in\mathcal{V}} J(x^0, \boldsymbol{\mu}, \boldsymbol{\nu}). \tag{3.3}$$

3.1.2 Zero-Sum Markov Games

The non-cooperative objective function (3.3) leads to the solution concept of *Saddle-Point Equilibrium* in game theory.

Definition 5. *A Saddle-Point Equilibrium (SPE) $(\boldsymbol{\mu^*}, \boldsymbol{\nu^*}) \in \mathcal{U}\times\mathcal{V}$ of the discounted zero-sum Markov games with two players satisfies the following inequalities:*

$$J(x^0, \boldsymbol{\mu}, \boldsymbol{\nu^*}) \geq J(x^0, \boldsymbol{\mu^*}, \boldsymbol{\nu^*}) \geq J(x^0, \boldsymbol{\mu^*}, \boldsymbol{\nu}), \forall \boldsymbol{\nu} \in \mathcal{V}, \boldsymbol{\mu} \in \mathcal{U}, \forall x^0 \in \prod_{i=1}^{I} \mathcal{X}^i. \tag{3.4}$$

The value $J^*(x^0)$ achieved under the saddle-point equilibrium of the game (3.3) for a given initial condition x^0 is called the value function of a two-player zero-sum game, i.e.,

$$J^*(x^0) := J(x^0, \boldsymbol{\mu}^*, \boldsymbol{\nu}^*) = \min_{\boldsymbol{\mu}\in\mathcal{U}} \max_{\boldsymbol{\nu}\in\mathcal{V}} J(x^0, \boldsymbol{\mu}, \boldsymbol{\nu}) = \max_{\boldsymbol{\nu}\in\mathcal{V}} \min_{\boldsymbol{\mu}\in\mathcal{U}} J(x^0, \boldsymbol{\mu}, \boldsymbol{\nu}). \tag{3.5}$$

By focusing on the class of global stationary policies, i.e., $\mu_j^{i,\mathrm{GS}} \in \mathcal{U}_j^{i,\mathrm{GS}}$ and $\nu_j^{i,\mathrm{GS}} \in \mathcal{V}_j^{i,\mathrm{GS}}$, the value function $J^*(x^0)$ can be characterized using dynamic programming principles. The action pairs $\mathbf{d}^*, \mathbf{a}^*$ with $d_j^{i*} = \mu_j^{i*,\mathrm{GS}}(x)$ and $a_j^{i*} = \nu_j^{i*,\mathrm{GS}}(x)$ satisfy the following Bellman equation:

$$J^*(x) = c(x, \mathbf{d}^*, \mathbf{a}^*) + \gamma \sum_{x' \in \prod_{i=1}^{I} \mathcal{X}^i} \Pr(x'|x, \mathbf{a}^*, \mathbf{d}^*) J^*(x'), \forall x. \tag{3.6}$$

The first term is the reward of current stage x and the second term is the expectation of the value function over all the possible next stage x'. The optimal action pairs $(\mathbf{d}^*, \mathbf{a}^*)$ guarantee that the value function starting from x equals the the current stage cost plus the expectation starting at the next stage x'. By solving the Bellman equation (3.6) for every state x, we can obtain the saddle-point equilibrium strategy pairs $(\boldsymbol{\mu}^*, \boldsymbol{\nu}^*)$ in global stationary policies.

The global stationary saddle-point policies in pure strategies may not always exist. The Bellman equation (3.7) can be solved under mixed-strategy action spaces. Let the mixed-strategy actions for the attacker and the defender be $\phi^a(x, \mathbf{a})$ and $\phi^d(x, \mathbf{d})$, where $\phi^d(x, \mathbf{d})$ (resp. $\phi^a(x, \mathbf{a})$) denotes the probability of taking action $\mathbf{d}$ (resp. $\mathbf{a}$) at the global state x for a defender (or an attacker). The saddle-point mixed-strategy action pair $(\phi^{a*}(x, \mathbf{a}), \phi^{d*}(x, \mathbf{d}))$ satisfies the following generalized Bellman equation, i.e., $\forall x$,

$$J^*(x) = \sum_{\mathbf{a}\in\mathcal{A}} \phi^{a*}(x, \mathbf{a}) \sum_{\mathbf{d}\in\mathcal{D}} \left[c(x, \mathbf{d}, \mathbf{a}) + \gamma \sum_{x' \in \prod_{i=1}^{I} \mathcal{X}^i} \Pr(x'|x, \mathbf{a}, \mathbf{d}) J^*(x') \right] \phi^{d*}(x, \mathbf{d}). \tag{3.7}$$

The existence of the mixed-strategy action pair is guaranteed when the action spaces $\mathcal{A}$ and $\mathcal{D}$ are finite. Hence solving (3.7) for every state x, we can obtain the mixed-strategy saddle-point equilibrium strategy pairs $(\hat{\boldsymbol{\mu}}^*, \hat{\boldsymbol{\nu}}^*)$ in global stationary policies, where $\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\nu}}$ are the mixed strategy extension of $\boldsymbol{\mu}, \boldsymbol{\nu}$, respectively.

3.1.3 Mathematical Programming Perspective

One way to compute the mixed-strategy equilibrium solutions for zero-sum games is to use a mathematical programming approach. Given a defender policy

$\phi^d(x, \mathbf{d})$, the attacker solves the following maximization problem for every state x:

$$J^*(x) = \max_{\phi^a(x,\mathbf{a})} \sum_{\mathbf{a}\in\mathcal{A}} \phi^a(x, \mathbf{a}) \sum_{\mathbf{d}\in\mathcal{D}} \left[c(x, \mathbf{d}, \mathbf{a}) + \gamma \sum_{x'} \Pr(x'|x, \mathbf{a}, \mathbf{d}) J^*(x') \right] \phi^d(x, \mathbf{d}), \forall x. \tag{3.8}$$

Define $f(x, \mathbf{a}) := \sum_{\mathbf{d}\in\mathcal{D}} \left[c(x, \mathbf{d}, \mathbf{a}) + \gamma \sum_{x'\in\prod_{i=1}^{I} \mathcal{X}^i} \Pr(x'|x, \mathbf{a}, \mathbf{d}) J^*(x') \right] \phi^d(x, \mathbf{d})$ and $f^*(x, \mathbf{a})$ when the defender policy is optimal. We have the following lemma:

Lemma 1. *The optimal attack's policy $\phi^{a*}(x, \mathbf{a})$ of (3.8) is a pure policy, denoted as $\phi^a(x, \mathbf{a})\mathbb{1}_{\{\mathbf{a}=\mathbf{a}^*\}}$, when the defender's policy is given, where $\mathbf{a}^* \in \arg\max_{\mathbf{a}} f(x, \mathbf{a})$.*

Proof. There exists an optimal action $\mathbf{a}^* \in arg \max_{\mathbf{a}} f(x, \mathbf{a})$. As a probability measure, all elements of $\phi^a(x, \mathbf{a})$ are positive and $\sum_{\mathbf{a}\in\mathcal{A}} \phi^a(x, \mathbf{a}) = 1, \forall x$. Multiply both sides of the equation $f(x, \mathbf{a}^*) \geq f(x, \mathbf{a})$ by $\phi^a(x, \mathbf{a})$ and sum over all possible $\mathbf{a}$, we arrive at

$$\begin{aligned} \sum_{\mathbf{a}\in\mathcal{A}} \phi^a(x, \mathbf{a}) f(x, \mathbf{a}) &\leq \sum_{\mathbf{a}\in\mathcal{A}} \phi^a(x, \mathbf{a}) f(x, \mathbf{a}^*) = 1 \cdot f(x, \mathbf{a}^*) \\ &= \sum_{\mathbf{a}\in\mathcal{A}} \phi^a(x, \mathbf{a}) \mathbb{1}_{\{\mathbf{a}=\mathbf{a}^*\}} f(x, \mathbf{a}), \quad \forall \mathbf{a}. \end{aligned}$$

Therefore, the optimal attacker policy is deterministic, i.e., $\phi^a(x, \mathbf{a})\mathbb{1}_{(\mathbf{a}=\mathbf{a}^*)}$. □

Lemma 1 is true for arbitrary defender policy, thus true for the optimal one.

Therefore, $J^*(x) = f^*(x, \mathbf{a}^*) \geq f^*(x, \mathbf{a}), \forall \mathbf{a}$. Now, we can form a bi-linear program:

$$
\begin{aligned}
&\min_{J^*(x), \phi^d(x,\mathbf{d})} \quad \sum_{x \in \prod_{i=1}^{I} \mathcal{X}^i} \alpha(x) J^*(x) \\
&\text{subject to :} \\
&(a)\ J^*(x) \geq \sum_{\mathbf{d} \in \mathcal{D}} \left[c(x, \mathbf{d}, \mathbf{a}) + \gamma \sum_{x' \in \prod_{i=1}^{I} \mathcal{X}^i} \Pr(x' | x, \mathbf{a}, \mathbf{d}) J^*(x') \right] \phi^d(x, \mathbf{d}), \quad \forall x, \mathbf{a} \\
&(b)\ \sum_{\mathbf{d} \in \mathcal{D}} \phi^d(x, \mathbf{d}) = 1, \qquad \forall x \\
&(c)\ \phi^d(x, \mathbf{d}) \geq 0, \qquad \forall x, \mathbf{d}
\end{aligned}
\tag{3.9}
$$

Constraints $(b)(c)$ reflect $\phi^d(x, \mathbf{d})$ as a probability measure. Constraint (a) guarantees that (3.8) is achieved under the optimal defender's policy. State-dependent weights $\alpha(x)$ are positive and satisfy $\sum_x \alpha(x) = 1$. Solutions of this program provide us the value function $J^*(x)$ and the optimal defender policy $\phi^{d*}(x, \mathbf{d})$.

3.1.4 Single-Controller Markov Game

In the single-controller game, one player's action entirely determines transition probabilities. This structure captures the fact that the failure probability of a node in the infrastructure depends on the action taken by the attacker once the node is attacked.

The single-controller assumption fits the infrastructure protection application because of the deficiency in real-time attack countermeasure after CINs are designed. Thus, defenders may not be capable of decreasing the probability of node failures under attacks once the network is established. However, the protection term can positively enhance the system security by mitigating the attack loss or increase the cost of an attacker. For example, defenders can apply for the cyber-insurance for high-risk nodes or setup ‘honeypot’ to increase the cost of the adversaries once trapped.

We focus on an attacker-controlled game Γ^a where the stochastic kernel for each node possesses $\Pr_{i,j}({x_j^i}'|x, d_j^i, a_j^i) = \Pr_{i,j}({x_j^i}'|x, a_j^i), \forall {x_j^i}', x, d_j^i, a_j^i$ and the system transition probability $\Pr(X' = x'|X = x, \mathbf{d}, \mathbf{a}) = \Pr(X' = x'|X = x, \mathbf{a})$. Because the system transition probability is independent of $\mathbf{d}$ and $\sum_{\mathbf{d}} \phi^{d*}(x, \mathbf{d}) \equiv 1$, the bi-linear program (3.9) can be reduced into a Linear Program (LP) where the primal LP is described as follows:

$$\begin{aligned}
&\min_{J^*(x),\phi^d(x,\mathbf{d})} \sum_{x' \in \prod_{i=1}^I \mathcal{X}^i} \alpha(x') J^*(x') \\
&\text{subject to :} \\
&(a)\ J^*(x) \geq \sum_{\mathbf{d}\in\mathcal{D}} c(x,\mathbf{d},\mathbf{a})\phi^d(x,\mathbf{d}) + \gamma \sum_{x' \in \prod_{i=1}^I \mathcal{X}^i} \Pr(x'|x,\mathbf{a})J^*(x') \quad \forall x, \mathbf{a} \\
&(b)\ \sum_{\mathbf{d}\in\mathcal{D}} \phi^d(x,\mathbf{d}) = 1, \qquad \forall x \\
&(c)\ \phi^d(x,\mathbf{d}) \geq 0, \qquad \forall x, \mathbf{d}
\end{aligned} \tag{3.10}$$

After solving (3.10), we obtain the value functions $J^*(x')$, the optimal defender's policy $\phi^{d*}(x, \mathbf{d})$, and we resort to the dual LP for the attacker's policy:

$$\begin{aligned}
&\max_{z(x),\phi^a(x,\mathbf{a})} \sum_{x \in \prod_{i=1}^I \mathcal{X}^i} z(x) \\
&\text{subject to :} \\
&(d)\ \sum_{\mathbf{a}\in\mathcal{A}} \phi^a(x',\mathbf{a}) - \sum_{x \in \prod_{i=1}^I \mathcal{X}^i} \sum_{\mathbf{a}\in\mathcal{A}} \gamma \Pr(x'|x,\mathbf{a})\phi^a(x,\mathbf{a}) = \alpha(x'), \quad \forall x' \\
&(e)\ z(x) \leq \sum_{\mathbf{a}\in\mathcal{A}} \phi^a(x,\mathbf{a})c(x,\mathbf{d},\mathbf{a}) \qquad \forall x, \mathbf{d} \\
&(f)\ \phi^a(x,\mathbf{a}) \geq 0, \qquad \forall x, \mathbf{a}
\end{aligned} \tag{3.11}$$

We normalize $\phi^{a*}(x,\mathbf{a}) = \frac{\phi^a(x,\mathbf{a})}{\sum_{\mathbf{a}} \phi^a(x,\mathbf{a})}$ to obtain the optimal policy for attacker. Analogous to the optimality principle of the value function (3.6), constraint (d) in the dual LP can be interpreted as the occupancy equality. The total occupancy frequency of state x', $\sum_{\mathbf{a}\in\mathcal{A}} \phi^a(x',\mathbf{a})$, is equal to the initial probability distribution of state x', $\alpha(x')$, plus the discounted expected visit from any other state x to state x', i.e., $\sum_{x\in\prod_{i=1}^{I}\mathcal{X}^i} \sum_{\mathbf{a}\in\mathcal{A}} \gamma \Pr(x'|x,\mathbf{a})\phi^a(x,\mathbf{a})$.

Theorem 1. *The optimal policy of attacker $\phi^a(x,\mathbf{a})$ solved by (3.11) is a pure policy, i.e., for each system state x, $\phi^a(x,\mathbf{a}^*) > 0$ and $\phi^a(x,\mathbf{a}) = 0, \forall \mathbf{a} \neq \mathbf{a}^*$. The explicit form is*

$$\mathbf{a}^* = arg \max_{\mathbf{a}\in\mathcal{A}} \left[\sum_{\mathbf{d}\in\mathcal{D}} c(x,\mathbf{d},\mathbf{a})\phi^{d*}(x,\mathbf{d}) + \gamma \sum_{x'} \Pr(x'|x,\mathbf{a}) J^*(x') \right].$$

Proof. Lemma 1 has shown that the optimal policy is deterministic, and thus here we only need to show that $\phi^{a*}(x,\mathbf{a}) = \frac{\phi^a(x,\mathbf{a})}{\sum_{\mathbf{a}} \phi^a(x,\mathbf{a})}$ is the optimal policy for the attacker. Following the proof of [50], we show that $\phi^{a*}(x,\mathbf{a})$ is the saddle point of the zero-sum game (3.4).

First, $\phi^{a*}(x,\mathbf{a})$ is well defined since the constraint (d) shows that $\sum_{\mathbf{a}} \phi^a(x,\mathbf{a}) \geq \alpha(x'), \forall x'$. By the complementary slackness of the dual LP, we require $J^*(x)$ strictly equal to $\sum\limits_{\mathbf{d}\in\mathcal{D}} c(x,\mathbf{d},\mathbf{a})\phi^{d*}(x,\mathbf{d}) + \gamma \sum\limits_{x'} \Pr(x'|x,\mathbf{a}) J^*(x')$ for all state x and the corresponding action $\mathbf{a}$ such that $\phi^a(x,\mathbf{a})$ is strictly positive, which is equivalent to $\phi^{a*}(x,\mathbf{a}) > 0$. Then, by multiplying both side by $\phi^{a*}(x,\mathbf{a})$ and summing over $\mathbf{a} \in \mathcal{A}$, we obtain the vector equation $\mathbf{J}^* = J(x^0, \boldsymbol{\mu}^*, \boldsymbol{\nu}^*)$. Next, we multiply an arbitrary $\phi^a(x,\mathbf{a})$ to both sides of constraints (a), sum over $\mathbf{a}$, and obtain a vector inequality $\mathbf{J}^* \geq J(x^0, \boldsymbol{\mu}^*, \boldsymbol{\nu})$. Therefore, we arrive at the RHS of saddle-point condition $J(x^0, \boldsymbol{\mu}^*, \boldsymbol{\nu}) \leq J(x^0, \boldsymbol{\mu}^*, \boldsymbol{\nu}^*)$. Similarly, the complementary slackness of the primal LP together with constraints (e) leads to $c(x, \mathbf{a}^*, \mathbf{d}^*) \leq c(x, \mathbf{a}^*, \mathbf{d})$. Because the transition probability is independent of defender policy, we can obtain the Left-Hand Side (LHS) of the saddle-point condition by computing (3.1). □

The major challenge to solve the LP is the large-scale nature of the CINs, which is known as the curse of dimension. Take (3.10) for an instance, we have $|\prod_{i=1}^{I} \mathcal{X}^i|$ variables in the LP objective and a constraints number of $|\prod_{i=1}^{I} \mathcal{X}^i| \times |\mathcal{A}| + |\prod_{i=1}^{I} \mathcal{X}^i| + |\prod_{i=1}^{I} \mathcal{X}^i| \times |\mathcal{D}|$. If we have n nodes in the CIN and all nodes can be attacked and defended, then we will have $N := 2^n$ variables and $N^2 + N + N^2$ constraints, which both grow exponentially with the number of nodes. The high computation cost prohibits the direct computation using the LP with a large number of nodes.

3.2 Factored Markov Game

To address the issue of the combinatorial explosion of the state size or the curse of dimensionality, we develop a factored Markov game framework in this section by leveraging the sparsity of the transition kernel. We first use factor graphs to represent the sparse structure of the probability transition matrix. Next, we introduce an approximation method for the value function and then reorganize terms and eliminate variables by exploiting the factored structure. We focus on the LP formulation of the attacker-controlled game. However, the technique can be extended to a bilinear form for the general zero-sum game to reduce computational complexity. Finally, we refer our reader to an overall structure diagram of this work in Fig. 3.1.

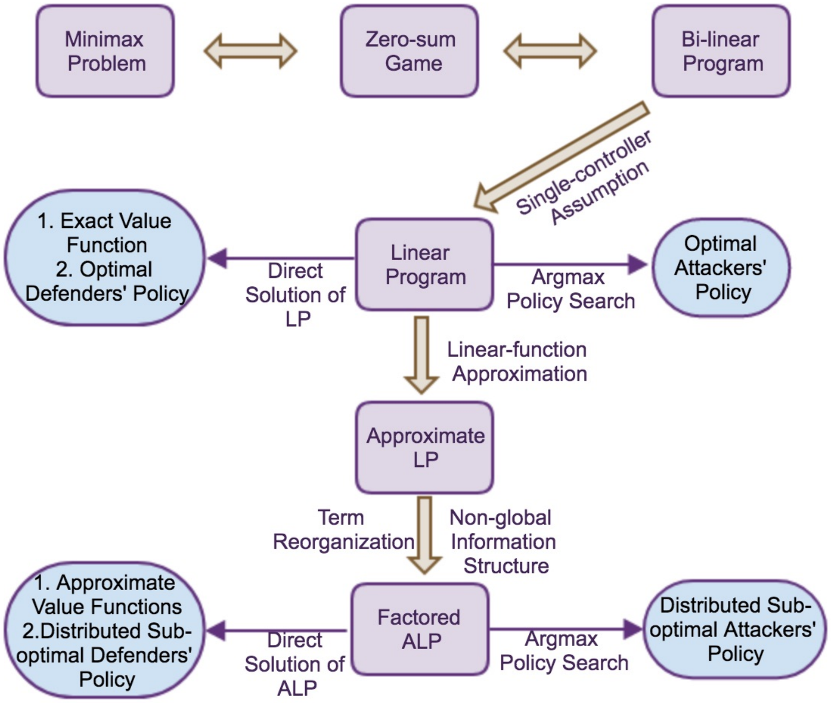

Minimax Problem
Zero-sum Game
Bi-linear Program
Single-controller Assumption
1. Exact Value Function 2. Optimal Defenders' Policy
Direct Solution of LP
Linear Program
Argmax Policy Search
Optimal Attackers' Policy
Linear-function Approximation
Approximate LP
Term Reorganization
Non-global Information Structure
1. Approximate Value Functions 2.Distributed Sub-optimal Defenders' Policy
Direct Solution of ALP
Factored ALP
Argmax Policy Search
Distributed Sub-optimal Attackers' Policy

Figure 3.1: In this overall structural diagram, blue squares show a sequence of techniques used in the problem formulation. The LP technique yields the exact value functions and the optimal defender's policy. The factored ALP yields an approximate value function and distributed sub-optimal defender's policy. The greedy search method solves for the attacker's policy.

3.2.1 Factored Structure

Define Ω_l as the set that contains all the parent nodes of node l. Parent nodes refer to the nodes that affect node l's state at the following time step through physical, cyber or logic interactions. The network example in Fig. 3.2 is a bi-directed graph that represent a 3-layer interdependent CIN. Then, Ω_l contains node l itself and all its neighbors, e.g., $\Omega_{1,1} = \{n_1^1, n_2^1, n_1^2, n_7^3\}$. Node l can affect itself because if, for instance, node l fails at time t, then it remains faulty in probability one without proper actions at next time step $t+1$. Note that we do not distinguish the dependence within (links in black) and across (links in blue) layers when considering the stochastic kernel. Recall m_i as the total number of nodes in layer i. We use a global index l to unify the 2D index of $\{i, j\}$, e.g., $l := \sum_{i'=1}^{i} i' m_{i'} + j$, which transforms the multi-layer network into a larger single network with $n = \sum_{i \in \mathcal{I}} m_i$ nodes. In this way, we can write $\Omega_{1,1} = \{n_1^1, n_2^1, n_1^2, n_7^3\}$ as $\Omega_1 = \{n_1, n_2, n_6, n_{17}\}$ and $\Pr_{i,j}(x_j^{i\prime} | x, d_j^i, a_j^i), \forall i \in \mathcal{I}, j \in \mathcal{N}^i$ equivalently as $\Pr_l(x_l{}' | x, d_l, a_l), \forall l = 1, 2, \cdots, n$. Define $x_{\Omega_l} := (x_l)_{l \in \Omega_l}$ as the state vector of the nodes inside set Ω_l, e.g., $x_{\Omega_1} = (x_1, x_2, x_6, x_{17})$. Then, each node's kernel will be $\Pr_{i,j}(x_j^{i\prime} | x, d_j^i, a_j^i) = \Pr_{i,j}(x_j^{i\prime} | x_j^i, x_{\Omega_{i,j}}, d_j^i, a_j^i)$ due to the sparsity, or in the global index $\Pr_l(x_l{}' | x, d_l, a_l) = \Pr_l(x_l' | x_l, x_{\Omega_l}, d_l, a_l)$.

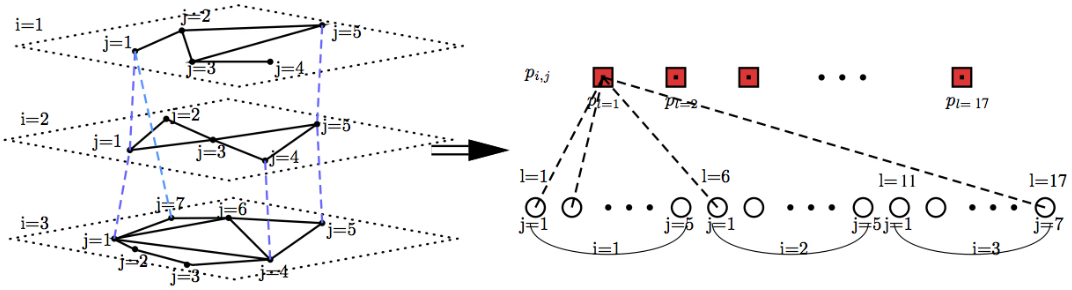


Figure 3.2: The left network shows a 3-layer CIN with blue lines representing the interdependencies across layers. The right bipartite graph shows a factor graph representation of the sparse transition probability. The total node number $n = \sum_{i=1,2,3} m_i = 5 + 5 + 7 = 17$.

3.2.2 Linear Function Approximation

We first approximate the high dimensional space spanned by the cost function vector $\mathbf{J} = (J^*(x'))_{x' \in \prod_{i=1}^{I} \mathcal{X}^i}$ through a weighted sum of basis functions $h_l(x'), l = 0, 1, \cdots, k$, where k is the number of 'features' and $h_0(x') \equiv 1, \forall x'$. Take CINs as an example. We choose a set of basis which serves as an indicator function of each node n_j^i's working state, e.g., $h_{i,j}(x') = x_j^{i\prime}, \forall i \in \mathcal{I}, j \in \mathcal{N}_j^i$. We unify the index with $l := \sum_{i'=1}^{i} i' m_{i'} + j$ and k equal to n, the total node number in the network.

To this end, we can substitute $J^*(x') = \sum_{l=0}^{k} w_l h_l(x')$ into (3.10) to obtain an Approximate Linear Program (ALP) with k variables $w_l, l = 0, 1, \cdots, k$.

$$\begin{aligned}
&\min_{\mathbf{w},\phi^d(x,\mathbf{d})} \sum_{x' \in \prod_{i=1}^{I} \mathcal{X}^i} \alpha(x') \sum_{l=0}^{k} w_l h_l(x') \\
&\text{subject to :} \\
&(a)\ \sum_{l=0}^{k} w_l h_l(x) \geq \sum_{\mathbf{d}\in\mathcal{D}} c(x,\mathbf{d},\mathbf{a})\phi^d(x,\mathbf{d}) + \gamma \sum_{x' \in \prod_{i=1}^{I} \mathcal{X}^i} \Pr(x'|x,\mathbf{a}) \sum_{l=0}^{k} w_l h_l(x'), \quad \forall x, \mathbf{a} \\
&(b)\ \sum_{\mathbf{d}\in\mathcal{D}} \phi^d(x,\mathbf{d}) = 1, \quad \forall x \\
&(c)\ \phi^d(x,\mathbf{d}) \geq 0, \quad \forall x, \mathbf{d}
\end{aligned} \tag{3.12}$$

The feature number k is often much smaller than the system state number 2^n. Hence the ALP reduces the involving variables in the LP objective. However, the exponentially growing number of constraints still makes the computation prohibitive. To address this issue, we further reduce the computational complexity in the following sections with similar techniques in [82].

Remark 2. *The ALP approximates $\min_{\boldsymbol{\mu}\in\mathcal{U}} \max_{\boldsymbol{\nu}\in\mathcal{V}} J(x^0, \boldsymbol{\mu}, \boldsymbol{\nu})$. The minimax strategy yields the optimal defensive strategy for the worst-case attacks. The strategy is achieved by searching the entire feasible attackers' actions of all possible system states in constraint (a) of (3.12). Thus, the approximate solution $\sum_{l=0}^{k} w_l h_l(x')$ is an upper bound to $J^*(x')$.*

3.2.3 Term Reorganization

The system transition matrix $\Pr(x'|x, \mathbf{a})$ has the dimension of $N \times N \times |\mathcal{A}|$ in constraint (a) of (3.10). Here, we choose indicator functions of each node $h_l(x') = x_l, \forall x', l = \{1, 2, \cdots, n\}$ as the set of basis functions, which yields a good trade off between the accuracy and computation complexity as shown in Section 3.3. We observe that the right-most term of constraint (a) of (3.10) can be rewritten as follows:

$$
\begin{aligned}
&\sum_{x' \in \prod_{i=1}^{I} \mathcal{X}^i} \Pr(x'|x, \mathbf{a}) \sum_{l=0}^{n} w_l h_l(x') \\
&\overset{(1)}{=} w_0 + \sum_{l=1}^{n} w_l \left[\sum_{x'_1, \cdots, x'_n} \prod_{k=1}^{n} \Pr_k(x'_k|x_k, a_k) x_l \right] \\
&\overset{(2)}{=} w_0 + \sum_{l=1}^{n} w_l \left[\sum_{x'_l} \Pr_l(x_l{}'|x_l, x_{\Omega_l}, a_l) x_l \sum_{\{x'_1, \cdots, x'_n\} \backslash \{x'_l\}} \prod_{k=1, k \neq l}^{n} \Pr_k(x'_k|x_k, a_k) \right] \\
&\overset{(3)}{=} w_0 + \sum_{l=1}^{n} w_l \left[\sum_{x'_l} \Pr_l(x_l{}'|x_l, x_{\Omega_l}, a_l) x_l \prod_{k=1, k \neq l}^{n} \sum_{x'_k} \Pr_k(x'_k|x_k, a_k) \right] \\
&\overset{(4)}{=} w_0 + \sum_{l=1}^{n} w_l \left[\sum_{x'_l} \Pr_l(x_l{}'|x_l, x_{\Omega_l}, a_l) x_l \right] \\
&= w_0 + \sum_{l=1}^{n} w_l \left[\Pr_l(x_l{}' = 1|x_l, x_{\Omega_l}, a_l) \cdot 1 + \Pr_l(x_l{}' = 0|x_l, x_{\Omega_l}, a_l) \cdot 0 \right] \\
&= w_0 + \sum_{l=1}^{n} w_l \left[\Pr_l(x_l{}' = 1|x_l, x_{\Omega_l}, a_l) \right] := w_0 + \sum_{l=1}^{n} w_l g_l(x_l, x_{\Omega_l}, a_l),
\end{aligned}
$$

where $g_l(x_l, x_{\Omega_l}, a_l) := \Pr_l(x_l{}' = 1|x_l, x_{\Omega_l}, a_l)$.

Equation (1) represents the vector x' with the set of its elements $\{x'_i\}$, writes the system transition probability in its factored form, and separates the first constant item w_0. The symbol $\sum_{\{x_1, \cdots, x_n\} \backslash \{x_l\}}$ in equation (2) means the summation over all variables except x_l. Equation (3) exchanges the summation and multiplication, and equation (4) is true because $\sum_{x'_k} \Pr_k(x'_k|x_k, a_k) \equiv 1$. To this end, we reduce $N = 2^n$ summations over the huge dimension system transition matrix into $n + 1$ summations over the local stochastic kernel.

3.2.4 Restricted Information Structure

The second step is to deal with $\sum_{\mathbf{d}} c(x, \mathbf{d}, \mathbf{a})\phi^d(x, \mathbf{d})$ in constraint (a) of (3.10). The saddle-point strategies studied in Section 3.1.2 belong to a class of global stationary policies in which the actions taken by the players are dependent on the global state information. The implementation of the policies is often restricted to the local information that is specific to the type of the infrastructure. For example, the Metropolitan Transportation Authority (MTA) may not be able to know the state of nodes in the power grid operated by Con Edison. Thus, MTA cannot make its policy based on the states of power nodes. Therefore, one way to approximate the optimal solution is to restrict the class of policies to stationary policies with local observations. We consider a time-invariant information structure of the defender $F^i_{j,t} \equiv F^i_j$. By unifying with the global index in Section 3.2.1, we let $l := \sum_{i'=1}^{i} i' m_{i'} + j$ and $F_l := F^i_j$. Define $\phi^d_l(x, d_l)$ as the probability of node l choosing d_l at state x. Therefore, $\phi^d(x, \mathbf{d}) = \prod_{l=1}^n \phi^d_l(x, d_l) = \prod_{l=1}^n \phi^d_l(F_l, d_l)$ and $F_l = (x_{\bar{\Omega}_l})$, where $\bar{\Omega}_l$ is the set of nodes which node l can observe. Note that not all nodes can be protected, i.e., $|\mathcal{D}| \leq N$. We let $d_l \equiv 0$ if node l cannot be defended.

$$
\begin{aligned}
\sum_{\mathbf{d}\in\mathcal{D}} c(x, \mathbf{d}, \mathbf{a})\phi^d(x, \mathbf{d}) &= \sum_{\mathbf{d}\in\mathcal{D}} \sum_{k=1}^{n} c_k(x_k, d_k, a_k) \prod_{l=1}^{n} \phi^d_l(F_l, d_l) \\
&= \sum_{k=1}^{n} \left[\sum_{d_w, w=1,\cdots,|D|} c_k(x_k, d_k, a_k)\phi^d_k(F_k, d_k) \prod_{l=1, l\neq k}^{n} \phi^d_l(F_l, d_l) \right] \\
&= \sum_{k=1}^{n} \left[\sum_{d_k} c_k(x_k, d_k, a_k)\phi^d_k(F_k, d_k) \prod_{l=1, l\neq k}^{n} \sum_{d_l} \phi^d_l(F_l, d_l) \right] \\
&= \sum_{k=1}^{n} \left[\sum_{d_k\in\{0,1\}} c_k(x_k, d_k, a_k)\phi^d_k(F_k, d_k) \right]. \qquad (3.13)
\end{aligned}
$$

Therefore, the ALP with the restricted information structure can be further rewritten as follows to form the factored ALP:

$$
\begin{aligned}
&\min_{\mathbf{w},\phi_l^d(F_l,d_l)} \sum_{l=0}^{n} \alpha_l w_l h_l(x) \\
&\text{subject to :} \\
&(a)\ 0 \geq \sum_{k=1}^{n} \sum_{d_k \in \{0,1\}} c_k(x_k, d_k, a_k)\phi_k^d(F_k, d_k) + \sum_{l=0}^{n} w_l[\gamma g_l(x_l, x_{\Omega_l}, a_l) - h_l(x)], \quad && \forall x, a_l \\
&(b)\ \sum_{d_i \in \{0,1\}} \phi_l^d(F_l, d_l) = 1, && \forall l, F_l \\
&(c)\ 0 \leq \phi_l^d(F_l, d_l) \leq 1, && \forall l, F_l, d_l
\end{aligned}
\tag{3.14}
$$

To this end, the number of constraints (b) $n \times |F_l|$ and (c) $n \times |F_l| \times 2$ relates only to the node number n and the domain of each node's information structure.

Remark 3. *For a general zero-sum game with bi-linear programming formulation (3.9), we can extend constraint (a) as follows with the same factored technique:*

$$
\begin{aligned}
0 \geq & \sum_{k=1}^{n} \sum_{d_k \in \{0,1\}} c_k(x_k, d_k, a_k)\phi_k^d(F_k, d_k) \\
& + \sum_{l=0}^{n} w_l[\gamma \sum_{d_l \in \{0,1\}} g_l(x_l, x_{\Omega_l}, a_l)\phi_l^d(F_l, d_l) - h_l(x)], \quad \forall x, a_l,
\end{aligned}
$$

where the second term is bi-linear in the variables of w_l and $\phi_l^d(F_l, d_l)$.

3.2.5 Variable Elimination

Constraint (a) of (3.14) can be further rewritten as one nonlinear constraint using the variable elimination method (see Section 4.2.2 of [69]) as follows:

$$0 \geq \max_{a_1,\cdots,a_n} \max_{x_1,\cdots,x_n} \sum_{k=1}^{n} \sum_{d_k \in \{0,1\}} c_k(x_k, d_k, a_k)\phi_k^d(F_k, d_k) + \sum_{l=0}^{n} w_l[\gamma g_l(x_l, x_{\Omega_l}, a_l) - h_l(x)]. \tag{3.15}$$

For simplicity, we have provided above an inequality for the case of a local information structure $\phi_l^d(F_l, d_l) = \phi_l^d(x_l, x_{\Omega_l}, d_l)$ and $|F_l| = 2^{|\Omega_l|+1}$.

First, we eliminate the variables of the attackers' action. Define $f_l(x_l, x_{\Omega_l}, a_l) := w_l[\gamma g_l(x_l, x_{\Omega_l}, a_l) - h_l(x_l)] + \sum_{d_l} c_l(x_l, d_l, a_l)\phi_l^d(x_l, d_l), l = 1, 2, \cdots, n$. We separate w_0, the weight of the constant basis, to the LHS and (3.15) becomes

$$\begin{aligned}(1-\gamma)w_0 &\geq \max_{x_1,\cdots,x_n} \max_{a_1,\cdots,a_n} \sum_{l=1}^{n} f_l(x_l, x_{\Omega_l}, a_l) \\ &= \max_{x_1,\cdots,x_n} \sum_{l=1}^{n} \max_{a_l} f_l(x_l, x_{\Omega_l}, a_l) \\ &:= \max_{x_1,\cdots,x_n} \sum_{l=1}^{n} e_l(x_l, x_{\Omega_l}).\end{aligned} \tag{3.16}$$

To achieve the global optimal solution of (3.14), we impose the following constraints for each l:

$$e_l(x_l, x_{\Omega_l}) \geq f_l(x_l, x_{\Omega_l}, a_l), \quad \forall x_l, x_{\Omega_l}, a_l. \tag{3.17}$$

Note that if node n_l cannot be attacked, we take $a_l \equiv 0$ and arrive at a simplified form:

$$e_l(x_l, x_{\Omega_l}) = f_l(x_l, x_{\Omega_l}, 1), \quad \forall x_l, x_{\Omega_l}. \tag{3.18}$$

The second step is to eliminate the variable of each node's state following a given order of $\mathcal{O} = \{p_1, p_2, \cdots, p_n\}$, where $\mathcal{O}$ is a permutation of $\{1, 2, \cdots, n\}$. The RHS of (3.16) is rewritten as:

$$\begin{aligned}
&\max_{x_1,\cdots,x_n} \sum_{l=1}^{n} e_l(x_l, x_{\Omega_l}) \\
=&\max_{x_{p_2},\cdots,x_{p_n}} \sum_{l=\{1,\cdots,n\}\backslash\mathcal{K}} e_k(x_k, x_{\Omega_k}) + \max_{x_{p_1}} \sum_{k\in\mathcal{K}} e_k(x_k, x_{\Omega_k}) \\
=&\max_{p_2,\cdots,p_n} \sum_{l=\{1,\cdots,n\}\backslash\mathcal{K}} e_k(x_k, x_{\Omega_k}) + E_1(\mathcal{E}),
\end{aligned} \tag{3.19}$$

where the set $\mathcal{K} := \{k : p_1 \in \{\Omega_k \cup \{k\}\}\}$ and E_1's domain $\mathcal{E} := \{x_j : j \in \{\{\cup_{k\in\mathcal{K}}\Omega_k\} \cup \{k\} \setminus \{p_1\}\}\}$. The variable x_{p_1} is eliminated and similar new constrains are generated to form the new LP, i.e., $E_1(\mathcal{E}) \geq \sum_{k\in\mathcal{K}} e_k(x_k, x_{\Omega_k})$ for all variables included in $\mathcal{E}$.

We repeat the above procedure of variable eliminations and constraints generation for n times following the order $\mathcal{O}$ and finally reach the equation $(1-\gamma)w_0 \geq E_n$, where E_n is a parameter independent of state and action variables. This method is suitable for a sparse network where each e_l has a domain involving a small set of node variables.

Example 1. *Consider a four-node example in Fig. 3.3 for the illustration of the variable elimination. With node 2 being immune to attacks, (3.18) can be reduced to $e_2(x_1, x_2) = f_1(x_1, x_2, 0), \forall x_1, x_2$. For node 1, (3.17) leads to four new inequality constraints $e_1(x_1) \geq f_1(x_1, a_1), \forall x_1, a_1$. Similarly, we have $2^4 = 16$ inequalities for node 3, i.e., $e_3(x_2, x_3, x_4) \geq f_3(x_2, x_3, x_4, a_3), \forall x_2, x_3, x_4, a_3$ and $2^3 = 8$ for node 4, i.e., $e_4(x_3, x_4) \geq f_3(x_3, x_4, a_4), \forall x_3, x_4, a_4$. After that, we eliminate all action* *variables and* (3.16) *becomes*

$$(1-\gamma)w_0 \geq \max_{x_1,x_2,x_3,x_4} e_1(x_1) + e_2(x_1,x_2) + e_3(x_2,x_3,x_4) + e_4(x_3,x_4). \tag{3.20}$$

With an elimination order $\mathcal{O} = \{3,2,4,1\}$, *the* RHS *of* (3.20) *can be rewritten as*

$$\begin{aligned}
&\max_{x_1,x_2,x_4} e_1(x_1) + e_2(x_1,x_2) + \max_{x_3} e_3(x_2,x_3,x_4) + e_4(x_3,x_4)\\
&= \max_{x_1,x_2,x_4} e_1(x_1) + e_2(x_1,x_2) + E_1(x_2,x_4).
\end{aligned}$$

The new constraints are generated, i.e., $E_1(x_2,x_4) \geq e_3(x_2,x_3,x_4) + e_4(x_3,x_4)$ *for all* x_2,x_3,x_4. *Then, we can repeat the above process and eliminate* x_2,x_4,x_1 *in sequence, i.e.,*

$$\begin{aligned}
&\max_{x_1,x_2,x_4} e_1(x_1) + e_2(x_1,x_2) + E_1(x_2,x_4)\\
&= \max_{x_1,x_4} e_1(x_1) + \max_{x_2} E_1(x_2,x_4) + e_2(x_1,x_2)\\
&= \max_{x_1,x_4} e_1(x_1) + E_2(x_1,x_4)\\
&= \max_{x_1} \max_{x_4} e_1(x_1) + E_2(x_1,x_4)\\
&= \max_{x_1} E_3(x_1) = E_4.
\end{aligned}$$

Along with the above process, new constraints appear $E_2(x_1,x_4) \geq E_1(x_2,x_4) + e_2(x_1,x_2), \forall x_1,x_2,x_4$; $E_3(x_1) \geq e_1(x_1) + E_2(x_1,x_4), \forall x_1,x_4$ *and* $E_4 \geq E_3(x_1), \forall x_1$. *Finally,* (3.20) *becomes* $(1-\gamma)w_0 \geq E_4$.

The new LP *in this example contains* 51 *constraints while the original constraint* (a) *includes* $2^{(4+3)} = 128$ *inequalities. With the increase of the node number*

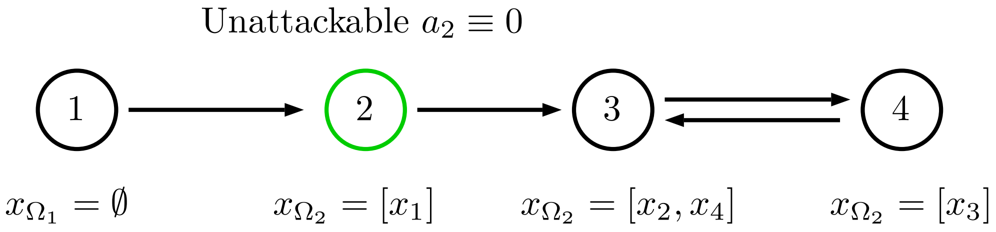

Unattackable $a_2 \equiv 0$
1
2
3
4
$x_{\Omega_1} = \emptyset$
$x_{\Omega_2} = [x_1]$
$x_{\Omega_2} = [x_2, x_4]$
$x_{\Omega_2} = [x_3]$

Figure 3.3: A four node example with node 2 unattackable. Assume a local information structure for each node $F_l = x_l, l = 1, 2, 3, 4$.

and a sparse topology, our factored framework greatly reduces the exponential computation complexity. Note that the order of $\{1, 2, 3, 4\}$ *introduces the least number of constraints in this case yet choosing the optimal order is shown to be NP-hard.*

3.2.6 Distributed Policy of Attacker

Similar to Lemma 1, we search for the approximate saddle-point policy of the attacker as follows:

$$\mathbf{a}^* \in \arg \max_{a_1,\cdots,a_n} \sum_{k=1}^{n} \sum_{d_k \in \{0,1\}} c_k(x_k, d_k, a_k)\phi_k^{d*}(F_k, d_k) + \sum_{l=0}^{n} w_l \gamma g_l(x_l, x_{\Omega_l}, a_l), \forall x_1, \cdots, x_n.$$

Separate w_0 in the second term and we obtain

$$\mathbf{a}^* \in \gamma w_0 + \arg \max_{a_1,\cdots,a_n} \sum_{k=1}^{n} \sum_{d_k} c_k(x_k, d_k, a_k)\phi_k^{d*}(F_k, d_k) + w_k \gamma g_k(x_k, x_{\Omega_k}, a_k), \forall x_1, \cdots, x_n.$$

Exchanging the argmax and the summation, we arrive at

$$\mathbf{a}^* \in \gamma w_0 + \sum_{k=1}^{n} \arg \max_{a_k} \sum_{d_k} c_k(x_k, d_k, a_k)\phi_k^{d*}(F_k, d_k) + w_k \gamma g_k(x_k, x_{\Omega_k}, a_k), \forall x_1, \cdots, x_n.$$

Therefore, we can obtain a distributed attack policy of node k which is fully determined by the state of itself and its parent nodes x_k, x_{Ω_k} and the state of nodes observable for the defender F_k, i.e.,

$$a_k = arg\max_{a_k} \sum_{d_k \in \{0,1\}} c_k(x_k, d_k, a_k)\phi_k^{d*}(F_k, d_k) + w_k \gamma g_k(x_k, x_{\Omega_k}, a_k), \forall x_k, x_{\Omega_k}, F_k.$$

Note that the approximate policy can be different from the optimal policy in Theorem 1. However, as long as the computation reduction surpasses the approximation error of the value function, it is worthwhile to equip with this sub-optimal policy.

Remark 4. *Under a local information structure with $F_l = x_l$, the defender decides its action at node l based on x_l and yet the attacker requires the state information of x_l and x_{Ω_l}. The difference in the structures of the policies is caused by the distinct factored structures of the cost function and the attacker-controlled transition probability matrix. The former $c_k(x_k, d_k, a_k)$ contains only x_k and the latter $g_l(x_l, x_{\Omega_l}, a_l)$ contains both x_l and x_{Ω_l}.*

3.2.7 Approximate Dual LP

We compute the dual of the ALP (3.12) by the Lagrange function. Our objective is to find a function $l(\mathbf{w}, \phi^a(x, \mathbf{a}), z(x))$ such that $l(\mathbf{w}, \phi^a(x, \mathbf{a}), z(x)) = 0$ when the constraints of (3.12) is satisfied and unbounded otherwise. Then, the following equation is equivalent to (3.12) :

$$\mathcal{L}(\mathbf{w}, \phi^a(x, \mathbf{a}), z(x)) = \min_{\mathbf{w}} \left[\sum_{x'} \alpha(x') \sum_{l=1}^{k} w_l h_l(x') + \max_{\phi^a(x,\mathbf{a}), z(x)} l(\mathbf{w}, \phi^a(x, \mathbf{a}), z(x)) \right].$$

Let $\phi^a(x, \mathbf{a}) \geq 0, \forall x, \mathbf{a}$ are multipliers for the inequality constraint (a), then

$$\begin{aligned} l(\mathbf{w}, \phi^a(x,\mathbf{a}), z(x)) = \sum_x z(x)(1 - \sum_{\mathbf{d}\in\mathcal{D}} \phi^d(x,\mathbf{d})) + \\ \sum_x \sum_{\mathbf{a}} \phi^a(x,\mathbf{a}) \Big[\sum_{\mathbf{d}\in\mathcal{D}} c(x,\mathbf{d},\mathbf{a}) \phi^d(x,\mathbf{d}) \\ + \sum_{x'} \gamma \Pr(x'|x,\mathbf{a})] \sum_{l=1}^{k} w_l h_l(x') - \sum_{l=1}^{n} w_l h_l(x) \Big]. \end{aligned} \tag{3.21}$$

Next, we reorganize the term and follow the minimax theorem to obtain:

$$\begin{aligned} &\mathcal{L}(\mathbf{w}, \phi^a(x,\mathbf{a}), z(x)) = \max_{z(x)} \sum_x z(x) \\ &+ \max_{\phi^a(x,\mathbf{a}), z(x)} \{ \sum_x \sum_{\mathbf{d}} \phi^d(x,\mathbf{d}) [\sum_{\mathbf{a}} \phi^a(x,\mathbf{a}) c(x,\mathbf{d},\mathbf{a}) - z(x)] \\ &+ \min_{\mathbf{w}} \sum_l w_l [\sum_{x'} \alpha(x') h_l(x') \\ &+ \gamma \sum_x \sum_{\mathbf{a}} \phi^a(x,\mathbf{a}) \sum_{x'} \Pr(x'|x,\mathbf{a}) h_l(x') - \sum_x \sum_{\mathbf{a}} \phi^a(x,\mathbf{a}) h_l(x)]\}. \end{aligned} \tag{3.22}$$

Finally, we can obtain the dual of (3.12) as follows:

$$\begin{aligned} &\max_{z(x), \phi^a(x,\mathbf{a})} \quad \sum_{x \in \prod_{i=1}^{I} \mathcal{X}^i} z(x) \\ &\text{subject to :} \\ &(a) \sum_x \alpha(x) h_l(x) + \gamma \sum_x \sum_{\mathbf{a}} \phi^a(x,\mathbf{a}) \sum_{x'} \Pr(x'|x,\mathbf{a}) h_l(x') \\ &\qquad\qquad = \textstyle\sum_x \sum_{\mathbf{a}} \phi^a(x,\mathbf{a}) h_l(x), \quad \forall l \\ &(b)\ z(x) \leq \sum_{\mathbf{a}} \phi^a(x,\mathbf{a}) c(x,\mathbf{d},\mathbf{a}), \qquad \forall x, \mathbf{d} \\ &(c)\ \phi^a(x,\mathbf{a}) \geq 0, \qquad \forall x, \mathbf{a} \end{aligned} \tag{3.23}$$

The dual of the ALP reveals the fact that constraint (a) approximates constraint

(d) of (3.10) while the objective and other constraints remain the same. The term $\gamma \sum_x \sum_{\mathbf{a}} \phi^a(x, \mathbf{a}) \sum_{x'} \Pr(x'|x, \mathbf{a}) h_l(x')$ sums over both x and x' in the same domain of $\prod_{i=1}^{I} \mathcal{X}^i$, and thus we can exchange x and x' in this term. Let $x^{(i)}, i = 1, \cdots, N$ be $N = 2^n$ possible values of the system state and $\mathbf{h}_l = (h_l(x^{(i)}))_{i=1,\cdots,N}$. Define $q^i(x^{(i)}) := \alpha(x^{(i)}) + \gamma \sum_{\mathbf{a}} \phi^a(x^{(i)}, \mathbf{a}) \sum_{x'} \Pr(x^{(i)}|x', \mathbf{a}) - \sum_{\mathbf{a}} \phi^a(x^{(i)}, \mathbf{a})$ and $\mathbf{q} := (q^1(x^{(1)}), \cdots, q^N(x^{(N)}))^T$.

Then, constraint (a) can be rewritten in matrix form as $\mathbf{Hq} = \mathbf{0}$, where $\mathbf{H} = (\mathbf{h}_1, \mathbf{h}_2, \cdots, \mathbf{h}_k)^T \in \mathcal{R}^{k \times N}$ and we can regard (3.23) as a relaxation of (3.11). If we select $k = N$ basis functions $h_l(x), l = \{1, 2, \cdots, |\prod_{i=1}^{I} \mathcal{X}^i|\}$, to be an indicator function of each possible value of the system state $x^{(l)}$, i.e., $h_l(x) = \mathbb{1}_{\{x = x^{(l)}\}}, l = 1, \cdots, N$, matrix $\mathbf{H}$ turns out to be an $N \times N$ identity matrix. Then, we arrive at $\mathbf{q} = \mathbf{0}$, i.e., N constraints $q^i(x^{(i)}) = 0, \forall i = 1, \cdots, N$, which is the same as constraint (a) in (3.11). Actually, as long as $k = N$ and $\mathbf{H}$ is of full rank, we have $\mathbf{q} = \mathbf{0}$. However, we obtain k constraints if we choose k less than N in the approximation form (3.23) with a reduced number of constraints. For each equation, according to the basis function selection, the corresponding elements in $\mathbf{q}$ sum up to 0.

Remark 5. *Analogous to the explanation in* (3.11)*, the constraint* (a) *in* (3.23) *achieves the occupancy equality for each feature rather than at each system state. For example, with the choice of the basis functions as* $h_l(x') = x_l$ *for all* $x', l = \{1, 2, \cdots, n\}$*, the* l^{th} *equation of constraint* (a) *in* (3.11) *is equivalent to the equation* $\sum_{x^{(i)} \in \mathcal{X}} q^i(x^{(i)}) = 0.$

3.3 Numerical Experiments

We implement our framework of the factored single-controller game and investigate the LP objective function as well as the policy of the attack and defender. Besides, we compare the approximation accuracy and the computation time. The LP objective shows in average the accuracy of the value functions starting at different initial states, which reflects the security level of the system. This risk analysis can have applications in areas such as cyber-insurance where risky systems have high premium rates. We use the pseudocode in Algorithm 1 to compute the saddle-point equilibrium policies for the factored single-controller game framework as follows.

3.3.1 Transition Probability and Cost

To illustrate the algorithm, we take one node's failure probability proportional to the failure number of its neighboring nodes. After one node is attacked, it can infect the connecting nodes and increase their failing risks. Besides, a node has a larger failure probability if it is targeted directly by attackers. In an attacker-controlled game, the defender cannot change the failure probability yet can positively affect the cost function.

The system stage cost is the sum of the local stage cost of each node $c(x, \mathbf{a}, \mathbf{d}) = \sum_{l=1}^{n} c_l(x_l, a_l, d_l)$, where $c_l(x_l, a_l, d_l) = \xi_1(1 - x_l) - \xi_2 a_l + \xi_3 d_l - \xi_4 a_l d_l$. The explicit form consists of four terms: the loss for faulty nodes, a cost of applying attacks, protection costs, and a reward of protecting a node which is being attacked. Since c_l is the cost function of node l in the defender's perspective and weights $\xi_i > 0, i = 1, 2, 3, 4$, the second and fourth terms are negative. The ordering of

Algorithm 1: Algorithm for computing the saddle-point equilibrium policies for the factored single-controller game framework as follows.

```
Initialize Initialize topology $\mathcal{G}$, elimination order $\mathcal{O}$, vector $aflag$ (and $dflag$)
  to indicate whether a node is controllable by attackers (and defenders);
// Note that $\phi_i^d(x_i, d_i)$ is a LP variable whose value depends on the
   value of $x_i, d_i$. Thus, we set up a $n \times n$ matrix to list all
   possible values for each $\phi_i^d(x_i, d_i)$.
Define ALP variables $w = \{w_0, w_1, \cdots, w_n\}, \phi^d = \{\phi_i^d(x_i, d_i)\}_{i=1,\cdots,n}$;
Determine the domain of $g = \{g_i(x_i, x_{\Omega_i}, a_i)\}, i = 1, \cdots, n$, based on the
  topology $\mathcal{G}$;
Set up an $n$-dimensional cell for functions $f_i(x_i, x_{\Omega_i}, a_i), i = 1, \cdots, n$;
foreach cell $i \leftarrow 1$ to $n$ do
    Create a table of $f_i$'s value based on the value of variables involved in $f_i$'s
      domain, i.e., $x_i, x_{\Omega_i}, a_i$;
    Compute the value of functions $g_i, h_i, c_i$ according to
      $f_i(x_i, x_{\Omega_i}, a_i) = w_l[\gamma g_l(x_l, x_{\Omega_l}, a_l) - h_l(x_l)] + \sum_{d_l} c_l(x_l, d_l, a_l)\phi_l^d(x_l, d_l)$ in
      Section 3.2.5;
    if $aflag(i) = 0$ (or $dflag(i) = 0$) then $a_i \leftarrow 0$ (or $d_i \leftarrow 0$) ;
end
Eliminate action variables $a_i$;
Generate $n$ new LP variables $e_i, i = 1, \cdots, n$ and set up a table based on the
  value of variables in its domain. Add constraints (3.17) or (3.18) according to
  aflag;
Eliminate state variables $x_i$ according to the elimination order $\mathcal{O}$;
Generate another $n$ new LP variables $E_i, i = 1, \cdots, n$, and setup a table
  based on the value of variables in its domain. Add constraints (3.19);
Solve the new ALP (3.14) to get the value function and the optimal defender
  policy;
Use greedy search for the distributed attacker policy (3.2.6);
```

$\xi_1 > \xi_4 > \xi_3 > \xi_2$ is assumed because the functionality of nodes serves as our primary goal. Protections are more costly than attacks, however, once an adversary attacks the node that possesses defensive strategies, e.g., a honeypot, which will create a significant loss for the attacker.

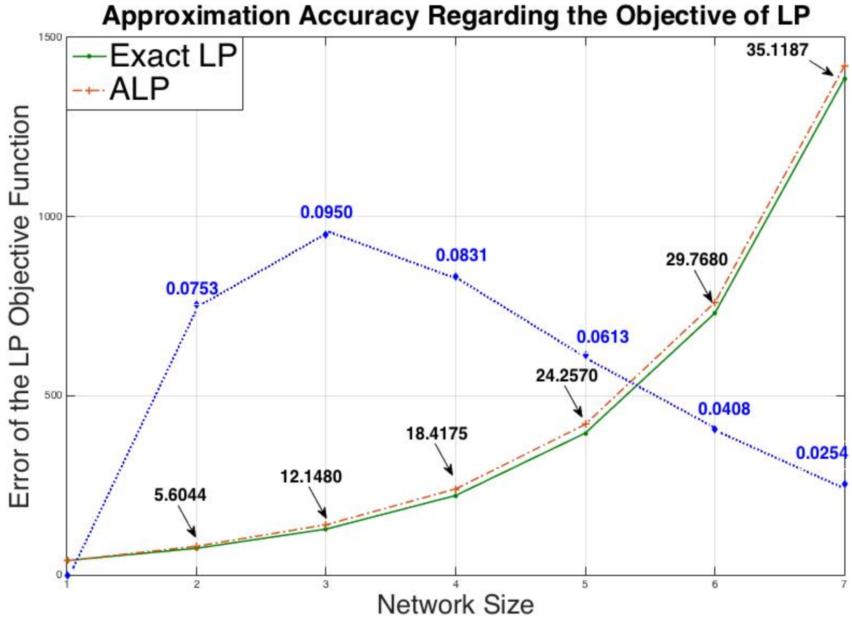


Figure 3.4: Approximation accuracy for a directed ring topology. The red and green lines are the value of the objective function of the LP and ALP, $obj(exact)$ and $obj(ALP)$ respectively. The black arrow shows the value of the absolute error while the blue number is the percentage of the relative error. The ALP achieves the upper bound for the exact LP as the size of network grows, i.e., $obj(ALP) \geq obj(exact)$ for the same network size.

3.3.2 Approximation Accuracy

We use a directed ring topology as shown in Fig. 3.5 to show the accuracy of the linear function approximation under the local information structure assumption. The comparison is limited to a network with 7 or fewer number of nodes due to the state explosion of the exact LP as shown in Table 3.1. The computational time is recorded by tic and toc function in $MATLAB$ and indicates the efficiency of the approximate algorithm as node number increases.

Fig. 3.4 illustrates the fact that the growth of the network size causes an increase of the absolute error $obj(ALP) - obj(exact) \geq 0$. This increasing absolute

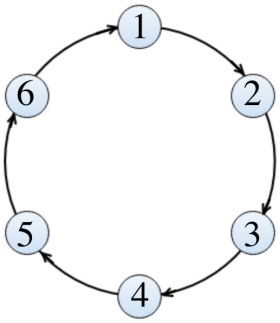


Figure 3.5: Directed ring topology of six nodes with index 1 to 6.

Table 3.1: Time cost (units: seconds) for the directed ring with an increasing node number.

Network Size	2	3	4	5	6	7
LP	0.21	0.63	3.33	34.52	178.68	1549.02
ALP	2.67	2.6	2.75	2.77	3.24	3.53

error is inevitable due to the growth of difference $2^n - n$ as n grows. In particular, the linear growth of the ALP variables $w_i, i \in \{0, 1, ..., n\}$ may not catch up with the exponential growth of the exact LP variables $v(\mathbf{x}), x \in \mathcal{X}$.

However, the linear function approximation remains suitable when we take a look at the relative error $(obj(ALP) - obj(exact))/obj(exact)$. We observe a decrease in the value of the objective function when the number of nodes in the network is larger than 3. Therefore, the error becomes negligible with a massive node number, which serves well for our large-scale CINs.

Besides accuracy, we see that for the ring topology, increasing the network size brings a higher cost to the attacker. Exponential function $f(x) = 18.25e^{0.6178x}$ provides a good fitting to the green line with the Root Mean Squared Error (RMSE) of 10.64.

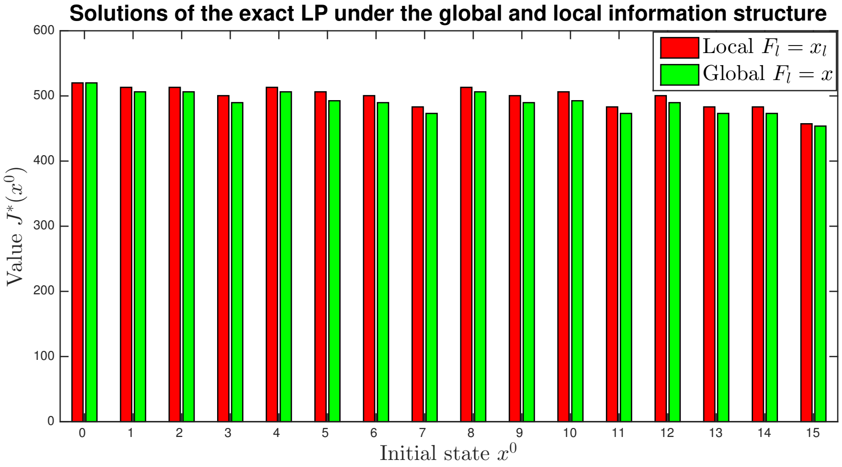


Figure 3.6: Value functions of different initial states in a four-node directed ring topology. State $0, 1, \cdots, 15$ is a decimalization of 2^4 different states from $(0, 0, 0, 0)$ to $(1, 1, 1, 1)$. Because the topology is symmetric, the number of working nodes determines the value. For example, state $1, 2, 4, 9$ share the same value in either global or local information structure because they all just have one working node. Besides, a better initial state $(1, 1, 1, 1)$ with all nodes working causes less loss of the system.

3.3.3 Various Information Structure

In Fig. 3.6, we compare the influence of global and local information structure of the defender to the exact LP. Recall that the y-axis shows the optimal cost of the system and a smaller value introduces a more secure system. Then, a local information structure in red brings a higher system cost than a global information structure in green for all initial states.

It shows that more knowledge can help defender better respond to the threat from the attacker. We can understand this with an example of the information structure of its neighboring nodes. Since the failure of its neighboring nodes increases its risk of being attacked, it tends to defend itself even when it is still working yet all his neighbors fail. Apparently, a defender with local information structure cannot achieve that. Besides, with the increasing of node number, the difference grows between global and local information structures.

3.3.4 Network Effect

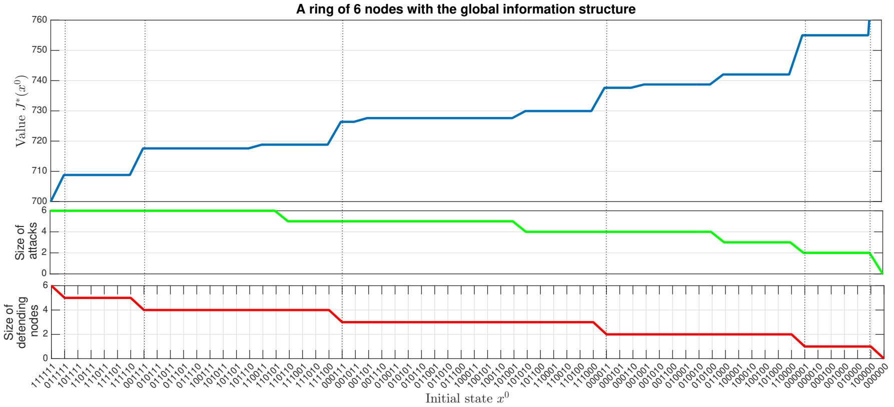


Figure 3.7: Value function $J^*(x^0)$ and the number of defending nodes at the optimal policy for different initial states x^0 in a 6-node ring example. From the value function (the blue line), the size of failures (the number of failure nodes) as well as the location of the failures affect the security level of the system. At the equilibrium policy, the size of defenses (the red line) is proportional to the number of the working nodes in the network. The attacker (the green line) decreases the number of nodes to attack as more nodes have been taken down but the green line is not aligned with the top two figures. The initial states between dotted lines share the same number of working nodes.

We reorganize the value-initial state pair $(J^*(x^0), x^0)$ of a 6-node ring topology in the top of Fig. 3.7 in an increasing order. Then, we see that the number of faulty nodes dominates the order of value. However, when the number of failures is the same, the location of the failure has an impact on $J^*(x^0)$, and a high degree of the failure aggregation results in a less secure system. For example, $J^*(x^0 = (1,1,1,0,0,0)) > J^*(x^0 = (1,1,0,0,1,0)) > J^*(x^0 = (1,0,1,0,1,0))$ because the dense pattern of the state vector $(1,1,1,0,0,0)$ is more likely to cause a cascading failure in the network than a sparse one $(1,0,1,0,1,0)$. These results suggest an alternating node protection if we cannot consolidate every node due to a limited budget. Specifically as shown in Fig. 3.5, we choose to consolidate every other connecting node in the 6-node ring network, i.e., node 1,3,5.

3.3.5 Optimal Policy

The global stationary policies of defenders and attackers for a 6-node ring topology is shown in Fig. 3.7 in red and green respectively. We observe that the size of defense is proportional to the number of working nodes in the network while the attacker compromises fewer nodes when the failure size increases.

Remark 6. *Since the defender can only affect the system through the reward function, the defense's policy follows an opposite pattern of the value function. The attacker, on the other hand, has a more irregular pattern because it can also influence the transition of the system.*

Other results of the approximated policy are summarized below. The local stationary defender policy is to defend a normal node with a higher probability. The defender does not defend the faulty nodes since the recovery of a failed node cannot mitigate the loss. Furthermore, if we reduce the cost of state failure ξ_1 or increase the defense cost ξ_3, we observe that the defender is less likely to defend. The sub-optimal distributed attacker policy avoids attacking node l when nodes in Ω_l are working. With an increase in ξ_4, the total number of attacks decreases to avoid attacking protected nodes. Thus, the presence of the defender results in fewer attacks. Besides, if node k cannot be attacked, then, naturally node k will not be defended, and attacker tends to decrease attack levels at the parent nodes of k.

Chapter 4

Time-Sensitive Attack Response in Nuclear Power Plants

As an example Industrial Control System (ICS) illustrated in Fig. 1.1, nuclear power plants play an essential role in energy supply and are under threat of sophisticated attacks. Once a cyber attack is detected, plant operators need to take actions to mitigate the consequences of the attack. However, this is a challenging task for the operators. First, the operators need to respond promptly. The high system complexity, the operators' knowledge limitations, and their increased stress during the incident may delay the response. Second, the dynamics of the system under a cyber attack depend on the actions of both system operators and the attacker. Therefore, the operators need not only to consider the effect of their own actions but also take into account the effect of the attacker's possible actions. Third, the operators are required to take actions to maximize long-term benefit rather than just take myopic actions that maximize the short-term benefit. Therefore, it is necessary to develop an attack response plan to support the plant operators. The importance of such a plan has also been emphasized in cyber-security regulation and guideline documents from the U.S. Nuclear Regulatory Commission [160], the Nuclear Energy Institute [153], the National Institute of Standards and Technology [28, 205], and the Department of Homeland Security [39]. For example, the U.S. Nuclear Regulatory Commission states [160]: "*The cyber security plan must include measures for incident response and recovery for cyber attacks. The cyber security plan must describe how the licensee will: (i) Maintain the capability for timely detection and response to cyber attacks; (ii) Mitigate the consequences of cyber attacks.*" Motivated by these needs, we propose a game-theoretic framework to determine the optimal actions for plant operators in responding to a cyber attack.

In addition to obtaining the optimal cyber-attack response strategy, we also focus on real-time assessment of the risk to nuclear power plants resulting from a cyber attack. This allows us to have a quantitative and objective estimate of the cyber-security risk to the plants, and to identify the gaps where improvements can be made.

4.1 Modeling of Defender-Attacker Interactions

Section 4.1.1 provides the modeling of the defender-attacker interaction as a finite-horizon Semi-Markov Game (SMG), and Section 4.1.2 introduces a systematic method for identifying feasible system states and transitions between the states under different action pairs in the game model. We integrate Probabilistic Risk Assessment (PRA) techniques into the modeling framework to quantify state transition probabilities in Section 4.1.3.

4.1.1 Finite-Horizon General-Sum Semi-Markov Game

Accompanied by the wider deployment of Information and Communication Technologies (ICTs) in nuclear power plants (e.g. smart sensors, wireless networks, PLC, SCADA) are the vulnerabilities embedded in these components. Such vulnerabilities can be potentially exploited by malicious attackers to cause damages to the physical components through attacks on cyber components. Such examples include the attack on Iran's nuclear facilities with Stuxnet [48] and the cyber event that occurred in the Davis-Besse nuclear power plant [133], where zero-day and n-day vulnerabilities in Windows systems were exploited. Zero-day vulnerabilities are particularly dangerous since no patches for the vulnerabilities have been developed. Even if a vulnerability has been identified, there is usually a time gap between patch release and patch installation [133], which provides n-day attackers with a time window to exploit the vulnerability. Besides, the patching process itself can be used as an attack vector. To improve the cyber-security posture of nuclear power plants, in addition to taking precautions to prevent cyber attacks from happening, it is also necessary to develop resilient attack response strategies in case certain components have been compromised.

In this research, we aim to develop a method for determining the optimal attack response strategy and for PRA. Suppose that a cyber attack is detected, then the reactor operator or the defender needs to respond to the attack to minimize its consequence. To develop an appropriate model for this problem and solve the problem, we need to take the following three aspects into consideration. First, because of the existence of the attacker, this is no longer a single-player decision-making optimization problem. One defender action paired with different attacker actions may lead to different consequences, and hence different payoffs for the defender. Therefore, in the modeling we need to consider both the defender's and the attacker's actions, as well as the effect of their interactive behavior on the system. Second, under a pair of defender and attacker actions, the system may transition from the current state to other states with certain probabilities, and the probabilities may vary with different defender-attacker action pairs. Besides, a system state transition is also stochastic with respect to time. Different system state trajectories correspond to different consequences and hence different payoffs for the defender. Therefore, the stochastic nature of state transitions needs to be considered in the modeling. Third, an attack normally will not last forever and the plant emergency support team should be able to terminate the attack after a certain amount of time. Therefore, it is preferable to model the problem as a finite-horizon process rather than an infinite-horizon process. Accordingly, this leads to a time-sensitive strategy for the defender, since the defender's optimal response may vary at different times in the finite horizon. To account for all the above aspects in one single modeling framework, we model the defender-attacker interaction as a SMG with a finite time horizon.

Specifically, the SMG has a finite time horizon $H \in (0, \infty)$ and a finite system state set $\mathbb{S}^e$. In the context of nuclear power plants, a state can be reactor core damage. We use $i \in \{1, 2\}$ in superscript to denote the defender and the attacker, respectively. The game can start from an arbitrary time point denoted as $T_0 = 0$. Both players take their first actions at $T_0 = 0$, and the initial time point corresponds to decision epoch $j = 0$. When the next state transition is detected at time point $T_1 > T_0$, both players take their second actions and T_1 corresponds to decision epoch $j = 1$. Since both players are not capable of observing the other player's actions, they are considered to take actions simultaneously at these discrete decision epochs. At each decision epoch j, player $i \in \{1,2\}$ under system state $s_j \in \mathbb{S}^e$ can take action $a_j^i \in \mathbb{A}^i(s_j), \forall s_j \in \mathbb{S}^e$, from a state-dependent action set $\mathbb{A}^i(s_j)$ which is discrete, finite, and commonly known by both players. In reality, one player may not possess the full information about his/her opposite's action set. But the assumption of complete information is the starting point of our research. As an example of action pairs in the context of nuclear power plants, the defender can switch to a backup component and the attacker can compromise a component in use. The action pair (a_j^1, a_j^2) results in the next state $s_{j+1} \in \mathbb{S}^e$ with transition probability $p(s_{j+1}|s_j, a_j^1, a_j^2)$. The transition can take place after sojourn time t_j, which is a random variable with support $[0, \infty)$ and distribution $q(\cdot|s_j, a_j^1, a_j^2, s_{j+1})$. Note that after taking the action pair (a_j^1, a_j^2) at the current decision epoch j, the players need to wait for the sojourn time t_j until the next decision epoch $j+1$ to take a new pair of actions (a_{j+1}^1, a_{j+1}^2). With a slight abuse of notation, we use $q(t_j|s_j, a_j^1, a_j^2, s_{j+1})$ to denote the probability density that the system remains in s_j at time t_j prior to the transition from s_j to s_{j+1} under the action pair (a_j^1, a_j^2). From a simple chain rule of conditional probability, we have the conditional probability of t_j and s_{j+1} as

$$\Pr\left(t_j, s_{j+1}|s_j, a_j^1, a_j^2\right) = q\left(t_j|s_j, a_j^1, a_j^2, s_{j+1}\right)p\left(s_{j+1}|s_j, a_j^1, a_j^2\right). \tag{4.1}$$

The process introduced above is a multi-agent SMDP, i.e., a SMG, because both the state transition and the sojourn time follow the Markov property only at decision epochs and the sojourn time distribution $q(\cdot|s_j, a_j^1, a_j^2, s_{j+1})$ can be arbitrary. If q is an exponential distribution, then the SMG reduces to a continuous-time Markov game which satisfies the Markov property at arbitrary time points.

If $\Pr(t_j, s_{j+1}|s_j, a_j^1, a_j^2)$ is independent of j, then the game process is said to be homogeneous and the probability becomes $\Pr(t, s'|s, a^1, a^2)$ where s, a^1, a^2, s' and t denote the current state, the defender action, the attacker action, the next state, and the sojourn time at the current state, respectively.

In real-world applications, the stochastic sojourn time and the semi-Markov assumption that players only take actions at discrete decision epochs can be explained from the following perspectives. Anytime the players observe a state transition, they take actions. For example, if a control computer is compromised, the defender may switch to a backup computer and the attacker may compromise another component in the system. However, it may take them a random amount of time to complete the actions. Besides, even if the actions are completed immediately, the observation of the state transition can experience a random amount of time delay. Therefore, the players would not change their actions between two decision epochs.

Semi-Markov decision processes where there is only one player or decision-maker have found wide applications in areas such as queuing control [174] and maintenance optimization [26]. In this work, we focus on SMGs of finite horizon, which can be represented as

$$(T_0 = 0, s_0, a_0^1, a_0^2, T_1, s_1, a_1^1, a_1^2, \dots, T_j, s_j, a_j^1, a_j^2, \dots, T_m, s_m, a_m^1, a_m^2). \tag{4.2}$$

The continuous-time state transition terminates when horizon H is reached and m is the number of state transitions. In (4.2), m is a random variable with support $\{0, 1, 2, \dots\}$ and its instantiation depends on both horizon H and the stochastic sojourn time at each decision epoch. Two samples of the process are shown in

Figure 4.1 for illustration.

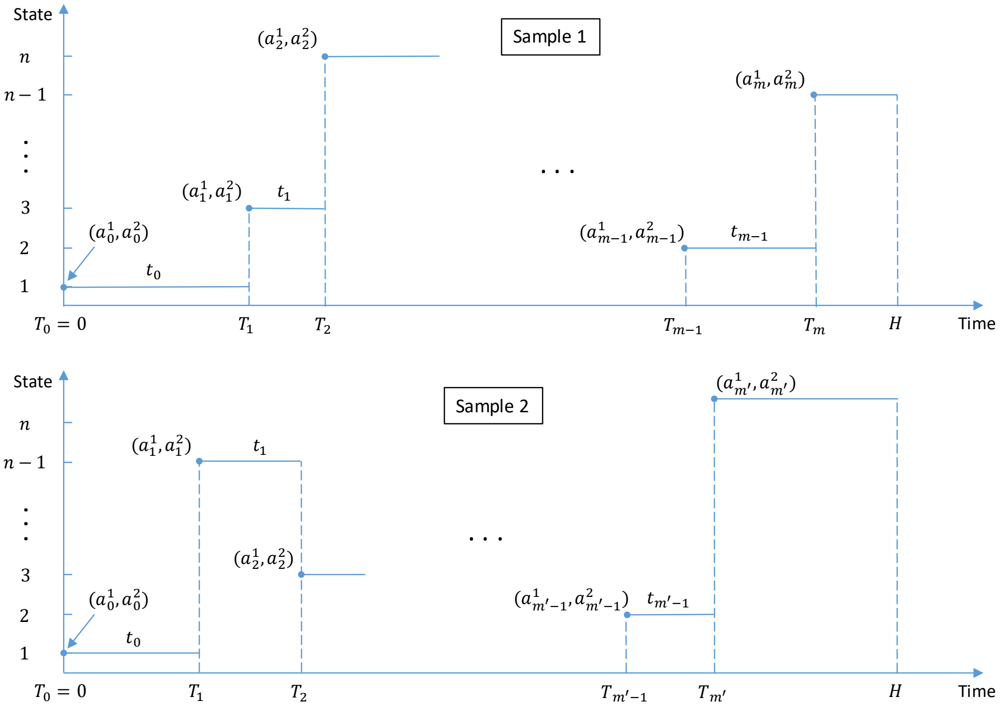


Figure 4.1: Two sample path realizations of the finite-horizon SMG. T_j: the time of the jth decision epoch; t_j: the sojourn time before the $(j+1)$th decision epoch, $t_j{=}T_{j+1}-T_j$; H: predetermined finite time horizon; a_j^1: the defender's action at the jth decision epoch (solid circle), which corresponds to time T_j; a_j^2: the attacker's action at the jth decision epoch.

Player i's action at each decision epoch is a realization of the mixed strategy $\sigma^i \in \Sigma^i$, which is defined in (4.3) and (4.4). A mixed strategy determines the probability (the probability is what makes it a mixed strategy) of an action which the player takes at any system state and any time of the game. As an example, a mixed strategy may determine that a player takes action 1 with probability 0.2 and action 2 with probability 0.8 at system state 3 and 10 min into the game. As a special case of a mixed strategy, a pure strategy determines the exact action (action 1 or action 2 with probability 1 in the aforementioned example). The finite time horizon renders the effect of both players' strategies time-dependent and the strategies should be adaptive to the amount of time that remains in the finite horizon. For example, as the amount of time remaining decreases, the player should reduce the probability of taking costly actions that produce benefits beyond the finite horizon of the game (therefore not counted). Formally, the mixed strategy for each player $i \in \{1, 2\}$ depends on both the system state and the time at the current decision epoch, i.e.,

$$\sigma^i \in \Sigma^i : \mathbb{S}^e \times [0, H] \mapsto \Delta\mathbb{A}^i(s), \tag{4.3}$$

$$\Delta\mathbb{A}^i(s) := \left\{ f : \mathbb{A}^i(s) \mapsto \mathbb{R}_+ \middle| \sum_{a^i \in \mathbb{A}^i(s)} f(a^i) = 1 \right\}, \forall s \in \mathbb{S}^e. \tag{4.4}$$

Since actions are applied at discrete decision epochs, each player $i \in \{1, 2\}$ receives an equivalent payoff $r^i(s_j, a^i_j, t_j, s_{j+1})$ at decision epoch $j \in \{0, 1, 2, \ldots, m\}$, which is decomposed into the following three terms to capture the payoff of the discrete decision, state transition, and the continuous sojourn time, respectively.

$$r^i(s_j, a^i_j, t_j, s_{j+1}) = r^{i,1}(s_j, a^i_j) + r^{i,2}(s_j, s_{j+1}) + r^{i,3}(s_j, t_j), i \in \{1, 2\}. \tag{4.5}$$

Here, $r^{i,1}(s_j, a^i_j)$ denotes the lump-sum payoff for taking action a^i_j at state s_j; $r^{i,2}(s_j, s_{j+1})$ denotes the lump-sum payoff for system transition from s_j to s_{j+1}; $r^{i,3}(s_j, t_j)$ denotes the duration payoff for staying in state s_j for time t_j. Each player $i \in \{1, 2\}$ aims to determine a mixed strategy $\sigma^{i*} \in \Sigma^i$ to maximize the cumulative payoff $u^i(s_0, H, \sigma^1, \sigma^2)$ in (4.6) expected (denoted by $\mathbb{E}$) over $a^1_j \sim \sigma^1$ (i.e., $\sim$ means that a^1_j is chosen according to the mixed strategy σ^1), $a^2_j \sim \sigma^2$, t_j, and s_j for all $j \in \{0, 1, \ldots, m\}$. Since there is no state transition after the last decision epoch, the lump-sum payoff for state transition is not considered in (4.6), as indicated by the last two terms.

$$
\begin{aligned}
u^i(s_0, H, \sigma^1, \sigma^2) &= \mathbb{E}\Bigg[\sum_{j=0}^{m-1}\bigg(r^i(s_j, a_j^i, t_j, s_{j+1})\bigg) + r^{i,1}(s_m, a_m^i) + r^{i,3}(s_m, H - T_m)\Bigg] \\
&= \mathbb{E}\Bigg[\sum_{j=0}^{m-1}\bigg(r^{i,1}(s_j, a_j^i) + r^{i,2}(s_j, s_{j+1}) + r^{i,3}(s_j, t_j)\bigg) \\
&\quad + r^{i,1}(s_m, a_m^i) + r^{i,3}(s_m, H - T_m)\Bigg], i \in \{1, 2\}.
\end{aligned}
\tag{4.6}
$$

Here, each player can only control his/her own strategy, which leads to the solution concept of NE. This will be introduced in Section 4.2.1.

4.1.2 Identification of System States and State Transitions

Suppose the system under study consists of M components. Each component $o \in \{1, \dots, M\}$ has a finite set of possible state values, denoted as $\mathbb{E}_o$ and the system state space $\mathbb{S}^e := \mathbb{E}_1 \times \mathbb{E}_2 \times \cdots \times \mathbb{E}_M$ can be represented by the Cartesian product of M sets, i.e., $\mathbb{E}_o$, $o = 1, 2, \dots, M$. We use $e_o \in \mathbb{E}_o$ to denote the instantiation of the component o's state. Then, each system state $s \in \mathbb{S}^e$ can be represented by the instantiation vector, i.e., $s = (e_1, e_2, \dots, e_M)$. Let $|\cdot|$ denote the cardinality of the set. It is clear that the size of the potential state space $|\mathbb{S}^e| = \prod_{o \in \{1,2,\dots,M\}} |\mathbb{E}_o| \geq \left(\min_o |\mathbb{E}_o|\right)^M$ increases at least exponentially with the number of components in a system. Thus, it is favorable to reduce the system state space and only consider the states that are of interest in the modeling and analysis of the game. For example, for a control system consisting of one main control computer and one backup control computer, we do not need to consider the state where both computers are used for control. However, it is not straightforward to identify these infeasible states for a system that consists of a large number of components.

Figure 4.2 presents the analysis for an example system to illustrate the algorithm. The example system consists of two components used for normal operation and backup, respectively. For simplicity, we have made the following assumptions. The defender can only switch from the first component for normal operation to the second one for backup, and hence $\mathbb{A}^1 = \{a^{1,1}, a^{1,2}\}$ as shown in Figure 4.2. The attacker can only compromise a component in use. We also assume that the defender's action is always effective, that is when the defender takes action $a^{1,2}$, the switch is always successful. However, the attacker's action may not always be effective. For example, when the attacker takes action $a^{2,2}$, component 1 is only compromised with a probability less than one. In Figure 4.2, a solid circle means that the analysis continues at the corresponding state, a solid square means that the analysis ends at the corresponding state, and an arrow denotes an action pair at the corresponding state. The analysis starts with the initial state $q_0 = (e_{11}, e_{21})$. At this state, the defender can take actions from $\mathbb{A}^1((e_{11}, e_{21})) = \{a^{1,1}, a^{1,2}\}$, and the attacker can take actions from $\mathbb{A}^2((e_{11}, e_{21})) = \{a^{2,1}, a^{2,2}\}$. Based on our assumption, action $a^{2,3}$ is not available to the attacker at this state because component 2 is not in use. Therefore, there are four possible action pairs and we can identify four successor states resulting from these four action pairs. Correspondingly, $\mathbb{Q}$ and $\mathbb{S}$ are updated. Note that one of the four successor states is exactly the initial state, and will not be analyzed again in the next step. This process continues and $\mathbb{Q}$ and $\mathbb{S}$ are updated until the end of the analysis, which is indicated by $\mathbb{Q} = \emptyset$. For this illustrative example, at the end of the analysis we have identified six system states, as shown in the rectangular box at the bottom of Figure 4.2. Once we have the tree-structure graph on the left side of Figure 4.2, the state transitions are readily identified. For example, we can see from Figure 4.2 that given start state (e_{11}, e_{21}) and under action pair $(a^{1,1}, a^{2,2})$, the system can remain in the same state or transition to (e_{12}, e_{21}).

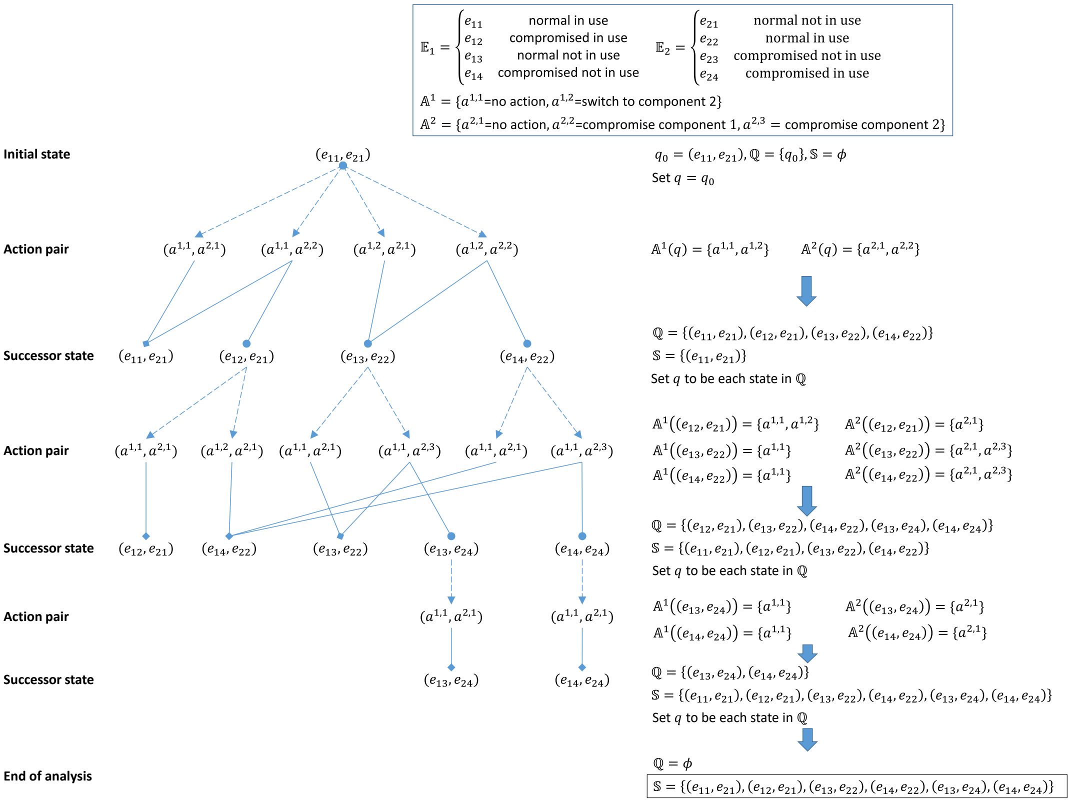

$\mathbb{E}_1 = \begin{cases} e_{11} & \text{normal in use} \\ e_{12} & \text{compromised in use} \\ e_{13} & \text{normal not in use} \\ e_{14} & \text{compromised not in use} \end{cases}$
$\mathbb{E}_2 = \begin{cases} e_{21} & \text{normal not in use} \\ e_{22} & \text{normal in use} \\ e_{23} & \text{compromised not in use} \\ e_{24} & \text{compromised in use} \end{cases}$
$\mathbb{A}^1 = \{a^{1,1}$=no action, $a^{1,2}$=switch to component 2$\}$
$\mathbb{A}^2 = \{a^{2,1}$=no action, $a^{2,2}$=compromise component 1, $a^{2,3}$ = compromise component 2$\}$
Initial state
(e_{11}, e_{21})
$q_0 = (e_{11}, e_{21}), \mathbb{Q} = \{q_0\}, \mathbb{S} = \phi$
Set $q = q_0$
Action pair
$(a^{1,1}, a^{2,1})$
$(a^{1,1}, a^{2,2})$
$(a^{1,2}, a^{2,1})$
$(a^{1,2}, a^{2,2})$
$\mathbb{A}^1(q) = \{a^{1,1}, a^{1,2}\}$
$\mathbb{A}^2(q) = \{a^{2,1}, a^{2,2}\}$
Successor state
(e_{11}, e_{21})
(e_{12}, e_{21})
(e_{13}, e_{22})
(e_{14}, e_{22})
$\mathbb{Q} = \{(e_{11}, e_{21}), (e_{12}, e_{21}), (e_{13}, e_{22}), (e_{14}, e_{22})\}$
$\mathbb{S} = \{(e_{11}, e_{21})\}$
Set q to be each state in $\mathbb{Q}$
Action pair
$(a^{1,1}, a^{2,1})$
$(a^{1,2}, a^{2,1})$
$(a^{1,1}, a^{2,1})$
$(a^{1,1}, a^{2,3})$
$(a^{1,1}, a^{2,1})$
$(a^{1,1}, a^{2,3})$
$\mathbb{A}^1((e_{12}, e_{21})) = \{a^{1,1}, a^{1,2}\}$
$\mathbb{A}^2((e_{12}, e_{21})) = \{a^{2,1}\}$
$\mathbb{A}^1((e_{13}, e_{22})) = \{a^{1,1}\}$
$\mathbb{A}^2((e_{13}, e_{22})) = \{a^{2,1}, a^{2,3}\}$
$\mathbb{A}^1((e_{14}, e_{22})) = \{a^{1,1}\}$
$\mathbb{A}^2((e_{14}, e_{22})) = \{a^{2,1}, a^{2,3}\}$
Successor state
(e_{12}, e_{21})
(e_{14}, e_{22})
(e_{13}, e_{22})
(e_{13}, e_{24})
(e_{14}, e_{24})
$\mathbb{Q} = \{(e_{12}, e_{21}), (e_{13}, e_{22}), (e_{14}, e_{22}), (e_{13}, e_{24}), (e_{14}, e_{24})\}$
$\mathbb{S} = \{(e_{11}, e_{21}), (e_{12}, e_{21}), (e_{13}, e_{22}), (e_{14}, e_{22})\}$
Set q to be each state in $\mathbb{Q}$
Action pair
$(a^{1,1}, a^{2,1})$
$(a^{1,1}, a^{2,1})$
$\mathbb{A}^1((e_{13}, e_{24})) = \{a^{1,1}\}$
$\mathbb{A}^2((e_{13}, e_{24})) = \{a^{2,1}\}$
$\mathbb{A}^1((e_{14}, e_{24})) = \{a^{1,1}\}$
$\mathbb{A}^2((e_{14}, e_{24})) = \{a^{2,1}\}$
Successor state
(e_{13}, e_{24})
(e_{14}, e_{24})
$\mathbb{Q} = \{(e_{13}, e_{24}), (e_{14}, e_{24})\}$
$\mathbb{S} = \{(e_{11}, e_{21}), (e_{12}, e_{21}), (e_{13}, e_{22}), (e_{14}, e_{22}), (e_{13}, e_{24}), (e_{14}, e_{24})\}$
Set q to be each state in $\mathbb{Q}$
End of analysis
$\mathbb{Q} = \phi$
$\mathbb{S} = \{(e_{11}, e_{21}), (e_{12}, e_{21}), (e_{13}, e_{22}), (e_{14}, e_{22}), (e_{13}, e_{24}), (e_{14}, e_{24})\}$

Figure 4.2: Illustration of the algorithm for identifying system states and state transitions.

For complex systems that have a large number of states, we still need to apply our knowledge of the system (e.g. the components comprising the system, the components states, the effect of defender-attacker actions on system state transitions) to manually perform the analysis following the proposed algorithm and identify system states and state transitions. Such analysis based on the analyst's knowledge is not unique to our method, but is commonly used in tasks related to the identification of system states and state transitions. However, the advantage of the proposed algorithm is that it provides us with a systematic way of exploring the system state space and identifying feasible system states and state transitions. Therefore, it can potentially reduce the system state space significantly. Taking this simple two-component system as an example, if we simply set the system state space as all the combinations of the components states, there would be 16 (i.e. 4×4) system states. However, we note that among the 16 system states, the state where the component for normal operation is "normal in use" and the component for backup is "normal in use" is not feasible in real-world applications, because normally only one of the two components is used at one single time point. By following the proposed algorithm, this system state is automatically eliminated. Besides, the proposed algorithm also helps us to avoid the situation where certain system states are ignored (considering the large number of system states, this is likely to happen if the analysis is not systematic).

4.1.3 Using PRA to Determine Transition Probabilities

Besides identifying system states and feasible state transitions, another task in developing the SMG model is to determine the probability for each feasible transition. Under certain conditions, the system may transition to states that correspond to physical damage. In this subsection, we introduce a method by integrating PRA techniques to determine such transition probabilities.

PRA [114] has found extensive applications in risk assessment for complex industrial systems, e.g., nuclear power plants, aerospace shuttles, and oil and gas systems. A PRA analysis combines two main elements: event tree analysis and fault tree analysis. Event tree analysis starts with an initiating event, generates sequences depending on whether safety functions succeed or not, and determines the consequence for each sequence. Based on the likelihood of the initiating event and safety functions failures, event tree analysis helps determine the likelihood of consequences of concern, e.g., reactor core damage. The probability of each safety function failure in an event tree can be obtained based on fault tree analysis. A fault tree analysis handles the safety function failure as a top event, decomposes it into a number of basic events, and links the top event and the basic events through logic gates, e.g., AND/OR gates. Based on the probabilities of the basic events, the probability of the top event, i.e., the safety function failure, can be obtained.

In a fault tree, the states of the basic events, i.e., TRUE or FALSE, can be determined by the states of the components that constitute the system under study. In the SMG model, as introduced in Section 4.1.2, the state of the system can also be represented by the states of these components. Therefore, the states of these components can be used to connect a PRA model to the SMG. Based on the identified components states, we can then assign TRUE or FALSE to the basic events in the fault tree and derive the probability that the system will transition to states that correspond to physical damage modeled in the PRA model. The connection of the PRA model and the SMG can be defined formally as follows.

We divide the identified state set $\mathbb{S} \subseteq \mathbb{S}^e$ into two new sub-sets $\mathbb{S} := \mathbb{S}_b \times \mathbb{S}_c$. The first sub-set $\mathbb{S}_b := \mathbb{E}_{b_1} \times \cdots \times \mathbb{E}_{b_g}$ is used to represent the states of the basic components, b_{o_b}, $o_b = 1, \ldots, g$. Examples of basic components are control computers, and sensors, which can be actually attacked by the attacker or controlled by the defender. The compromise of the basic components usually does not directly lead to the consequences of concern. Thus, we define the second sub-set

$S_c := \mathbb{E}_{c_1} \times \cdots \times \mathbb{E}_{c_z}$ to represent the states of consequential components, c_{o_c}, $o_c = 1, \ldots, z$. Consequential components are related to the consequences of concern. For instance, in the application of nuclear power plants, such a consequential component can be the reactor core, which has two states (core damage or core OK). These consequential components usually cannot be damaged directly but the damage can be realized through compromising the basic components, as described in a PRA model. We use s and s' to represent the value of identified state at the current and subsequent decision epochs, respectively. The same notation rule applies to s_b, s'_b, s_c, s'_c, a^1, $a^{1'}$, a^2, and $a^{2'}$. We have $s = (s_b, s_c)$ and $s' = (s'_b, s'_c)$.

The causal relationship between the states of the components is shown in Figure 4.3, which implies that states of basic components and the action pair at the current decision epoch will influence states of basic components at the next decision epoch. The states of basic components and the consequential components at the current decision epoch influence the state of consequential components at the next decision epoch.

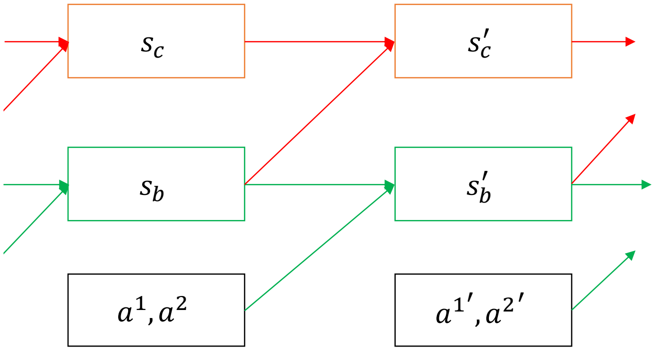


Figure 4.3: The causal relationship between the basic components states, consequential components states, and player actions.

By separating s into s_b and s_c, and s' into s'_b and s'_c, we can express the transition probability as:

$$\Pr(s'|s,a^1,a^2) = \Pr\Big((s'_b,s'_c)|(s_b,s_c),a^1,a^2\Big). \tag{4.7}$$

Based on the causal relationships in Figure 4.3 and the chain rule of probability, (4.7) can be simplified as

$$\Pr(s'|s,a^1,a^2) = \Pr(s'_b|s_b,a^1,a^2) \times \Pr(s'_c|s_c,s_b). \tag{4.8}$$

The first term on the right side of (4.8) quantifies the interaction between the defender and the attacker. This transition probability can be obtained based on expert judgments, statistics, simulations [134], or the Common Vulnerability Scoring System (CVSS) scores [170]. The second term on the right side of (4.8) quantifies the relationship between the states of basic components and consequential components, which can be obtained through a PRA model.

The method for integrating PRA to obtain transition probabilities is illustrated in Figure 4.4 through a simple example based on the main feedwater system and the reactor core. The state space can be represented by the states of the four components, i.e., main computer, backup computer, sensor, and reactor core. In this example, we would like to determine the transition probability from state s to state s' under action pair ($a^1 = \text{no action}, a^2 = \text{no action}$). The first three components (i.e., main computer, backup computer, sensor) are the basic components and the fourth one (i.e., reactor core) is the consequential component. The first term in (4.8), i.e., $\Pr(s'_b|s_b,a^1,a^2)$, is equal to unity because state s'_b represents the universe and includes any possible state of the basic components, i.e., s_b, no matter which defender and attacker actions are taken. To derive the second term in (4.8), i.e.,

$\Pr(s'_c|s_c, s_b)$, we first assign "TRUE" or "FALSE" to the basic events in the fault tree based on the states of basic components for state s in the SMG model. For instance, as the sensor is "compromised and used" in state s, we can assign "TRUE" to the basic event "sensor is used but compromised." In the second step, based on the states of the basic events, the probability of the top event (i.e., main feedwater system failure) can be obtained as one. In the third step, we obtain the probability of core damage using the event tree. Consider a transient caused by a cyber attack (i.e., the initiating event in the event tree) occurs, reactor shutdown succeeds, and the auxiliary feedwater system fails with a probability of 0.001, then the probability of core damage can be obtained as 0.001. In the fourth step, by multiplying the two terms (i.e., $\Pr(s'_b|s_b, a^1, a^2)$ and $\Pr(s'_c|s_c, s_b)$), we finally obtain the transition probability as 0.001.

4.2 Solution Concept and Technique

In this section, we first introduce the solution concept of the Nash Equilibrium (NE) for the game formalized in Section 4.1. Then, we introduce the dynamic programming technique to obtain the NE.

4.2.1 Nash Equilibrium

A mixed-strategy profile $(\sigma^{1^*} \in \Sigma^1, \sigma^{2^*} \in \Sigma^2)$ is a NE if for both the defender $(i = 1)$ and the attacker $(i = 2)$,

$$v^1(s, H-T) := u^1(s, H-T, \sigma^{1^*}, \sigma^{2^*}) \geq u^1(s, H-T, \sigma^1, \sigma^{2^*}), \\ \forall s \in \mathbb{S}, \forall \sigma^1 \in \Sigma^1, \forall T \in [0, H], \qquad (4.9)$$

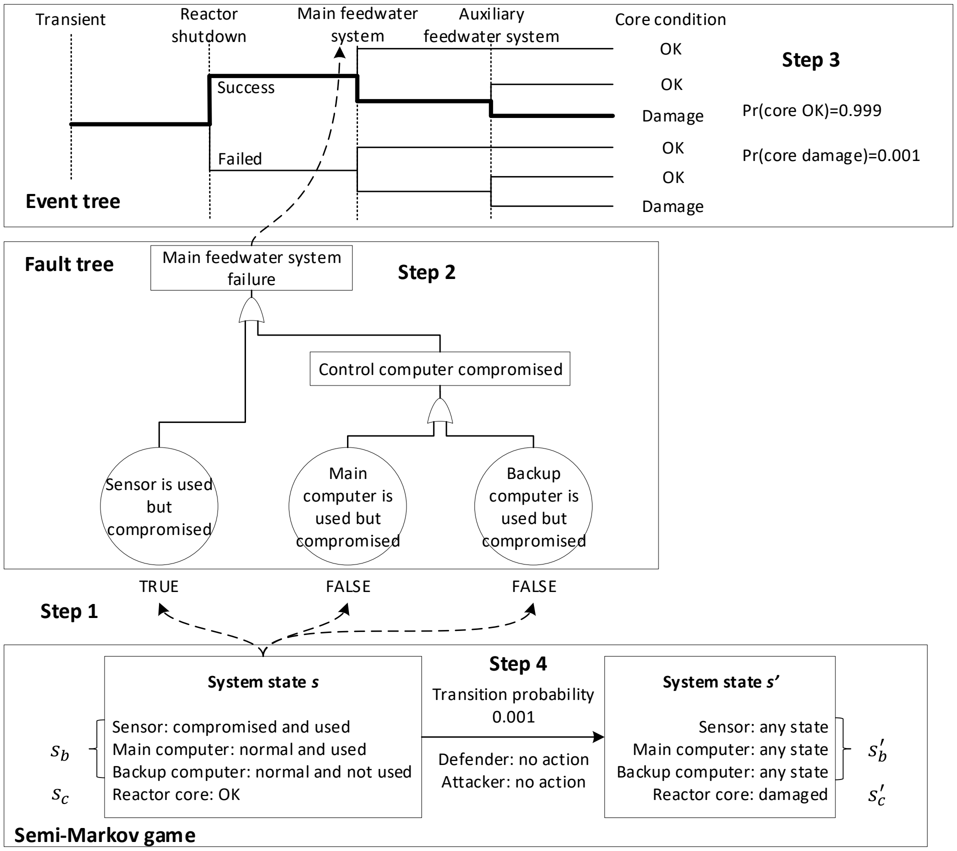


Figure 4.4: Illustration of the method for obtaining state transition probabilities.

$$v^2(s, H-T) := u^2(s, H-T, \sigma^{1*}, \sigma^{2*}) \geq u^2(s, H-T, \sigma^{1*}, \sigma^2),$$
$$\forall s \in \mathbb{S}, \forall \sigma^2 \in \Sigma^2, \forall T \in [0, H], \qquad (4.10)$$

where $v^i(s, H-T)$ is the value function for player i at state s and time T. Specifically, $v^i(s_0, H)$ describes the maximum payoff for player i if the system starts with s_0 at time 0 and the finite horizon is H. Since the SMG terminates at a finite horizon H, the boundary conditions are $v^i(s, 0) = \max_{a^i \in \mathbb{A}^i(s)} r^{i,1}(s, a^i), \forall s \in \mathbb{S}, \forall i \in \{1, 2\}$.

At a mixed-strategy NE, no player can gain by unilateral deviations. For the SMG with a finite horizon used in this research, a mixed strategy is defined as a function of both system states and time in the process. The optimal cyber-attack response strategy σ^{1^*} maximizes the payoff for the defender while considering the potential actions of the attacker at any state and any time.

4.2.2 Dynamic Programming

In dynamic programming, the cumulative payoff for player i in (4.6) can be written in the form in (4.11). In (4.11), $r^i(s_0, a_0^i, t_0, s_1)$ is the immediate payoff and $u^i(s_1, H - t_0, \sigma^1, \sigma^2)$ is the payoff-to-go with remaining time $H - t_0$. Since the first system transition can occur at any time between 0 and H, according to the sojourn time distribution q, we integrate $r^i(s_0, a_0^i, t_0, s_1) + u^i(s_1, H - t_0, \sigma^1, \sigma^2)$ in (4.11) in t_0 over $[0, H]$. The expectation is over the defender action $a_0^1 \sim \sigma^1$, the attacker action $a_0^2 \sim \sigma^2$, and state s_1 where the system arrives at decision epoch 1. We use $\mu^i(\cdot|s_0, 0)$ to denote the probability distribution $\sigma^i(s_0, 0)$ and therefore $\mu^i(a_0^i|s_0, 0)$ is the probability that player i takes action a_0^i at state s_0 and time 0. In (4.12), $u^i(s_j, H - T_j, \sigma^1, \sigma^2)$ at any decision epoch j are of the similar form as $u^i(s_0, H, \sigma^1, \sigma^2)$ which equals.

$$\begin{aligned}
&\underset{a_0^1, a_0^2, s_1}{\mathbb{E}}\left[\int_0^H \left(r^i(s_0, a_0^i, t_0, s_1) + u^i(s_1, H - t_0, \sigma^1, \sigma^2)\right) q(t_0|s_0, a_0^1, a_0^2, s_1) dt_0\right] \\
&= \sum_{a_0^1 \in \mathbb{A}^1(s_0)} \mu^1(a_0^1|s_0, 0) \sum_{a_0^2 \in \mathbb{A}^2(s_0)} \mu^2(a_0^2|s_0, 0) \sum_{s_1 \in \mathbb{S}} p(s_1|s_0, a_0^1, a_0^2) \Bigg[\int_0^H \Bigg(r^i(s_0, a_0^i, t_0, s_1) \\
&\quad + u^i(s_1, H - t_0, \sigma^1, \sigma^2)\Bigg) q(t_0|s_0, a_0^1, a_0^2, s_1) dt_0\Bigg], i \in \{1, 2\}.
\end{aligned} \tag{4.11}$$

$$\begin{aligned}
&u^i(s_j, H - T_j, \sigma^1, \sigma^2) = \mathop{\mathbb{E}}_{a_j^1, a_j^2, s_j} \Big[\int_0^{H - T_j} \Big(r^i(s_j, a_j^i, t_j, s_{j+1}) \\
&+ u^i(s_{j+1}, H - T_j - t_j, \sigma^1, \sigma^2) \Big) q(t_j | s_j, a_j^1, a_j^2, s_{j+1}) dt_j \Big] \\
&= \sum_{a_j^1 \in \mathbb{A}^1(s_j)} \mu^1(a_j^1 | s_j, T_j) \sum_{a_j^2 \in \mathbb{A}^2(s_j)} \mu^2(a_j^2 | s_j, T_j) \sum_{s_{j+1} \in \mathbb{S}} p(s_{j+1} | s_j, a_j^1, a_j^2) \\
&\int_0^{H - T_j} (r^i(s_j, a_j^i, t_j, s_{j+1}) \quad + u^i(s_{j+1}, H - T_j - t_j, \sigma^1, \sigma^2)) q(t_j | s_j, a_j^1, a_j^2, s_{j+1}) dt_j, \\
&j = 0, 1, 2, \ldots, i \in \{1, 2\}.
\end{aligned} \tag{4.12}$$

To obtain the value functions defined in (4.9) and (4.10) and then the mixed-strategy NE, one way is to find a contraction mapping. Contraction mapping is a commonly used property of an operator Γ to prove that a unique fixed point exists. With this property, equation $v = \Gamma v$ has the unique solution v_0 that can be found via value iteration. That is, independent of the initial value set for v, after applying Γ for an infinite number of times, v^∞ converges to v_0. Finite horizon semi-Markov decision processes (with only one player) possess the contraction property defined in [135], thus function approximation can be applied iteratively and is guaranteed to converge to the unique value of concern under any initial condition. However, the general-sum SMG with arbitrary distributions $q(\cdot | s_j, a_j^1, a_j^2, s_{j+1})$ and $p(\cdot | s_j, a_j^1, a_j^2)$ considered in this research does not satisfy this contraction property, and the integration in (4.12) makes it challenging to obtain an analytic solution.

To solve this problem, we discretize the continuous-time SMG of horizon H into N time steps with a constant time interval H/N and approximate the integration in (4.12) as in (4.13). Therefore, the accuracy of the analysis may depend on the number of time steps. Using a smaller time interval and hence more time steps increases the accuracy of the result, but also is more computationally expensive.

Therefore, we need to balance these two aspects. In a specific application, a range of time intervals can be tested, and the one that achieves high accuracy and is also computationally efficient can be identified. To avoid confusion, we have dropped index j which denotes the jth decision epoch. Instead, we use s to denote the state at the current decision epoch and s' the state at the subsequent decision epoch. In (4.13), h denotes the time step corresponding to the current decision epoch, $N-h$ denotes the number of remaining time steps, k denotes the sojourn time (represented by the number of time steps) at the current state, and $N-h-k$ is the number of remaining time steps at the subsequent decision epoch. In (4.13), $\tilde{q}(k|s,a^1,a^2,s') = \int_{\frac{H}{N}(k-1)}^{\frac{H}{N}k} q(\tau|s,a^1,a^2,s')d\tau$ is the probability that the sojourn time at the current state is k.

$$\begin{aligned}
&u^i(s, N-h, \sigma^1, \sigma^2) \\
&= \sum_{a^1 \in \mathbb{A}^1(s)} \mu^1(a^1|s,h) \sum_{a^2 \in \mathbb{A}^2(s)} \mu^2(a^2|s,h) \sum_{s' \in \mathbb{S}} p(s'|s,a^1,a^2) \sum_{k=1}^{N-h} \tilde{q}(k|s,a^1,a^2,s') \\
&\Big(r^i(s,a^i,k,s') + u^i(s', N-h-k, \sigma^1, \sigma^2) \Big), h = 0,1,\dots,N, i \in \{1,2\}.
\end{aligned} \tag{4.13}$$

Theorem 2. *A pair* $\Big(\sigma^{1*}(s,h), \sigma^{2*}(s,h)\Big)$ *constitutes a mixed-strategy NE solution to the bi-matrix game* $(R^1_{h,s}, R^2_{h,s})$ *if, and only if, there exists a pair* (w^{1*}, w^{2*}) *such that* $(\sigma^{1*}(s,h), \sigma^{2*}(s,h), w^{1*}, w^{2*})$ *is a solution of the following bilinear program:*

$$\max_{\sigma^1(s,h),\sigma^2(s,h),w^1,w^2} \sum_{a^1 \in \mathbb{A}^1(s)} \mu^1(a^1|s,h) \sum_{a^2 \in \mathbb{A}^2(s)} \mu^2(a^2|s,h) \sum_{i \in \{1,2\}} R^i_{h,s}(a^1,a^2) + \sum_{i \in \{1,2\}} w^i \tag{4.14}$$

such that

$$\sum_{a^2 \in \mathbb{A}^2(s)} \mu^2(a^2|s,h) R^1_{h,s}(a^1,a^2) \le -w^1(s,h), \forall a^1 \in \mathbb{A}^1(s), \forall \sigma^2(s,h) \in \Delta\mathbb{A}^2(s) \tag{4.15}$$

$$\sum_{a^1 \in \mathbb{A}^1(s)} \mu^1(a^1|s,h) R^2_{h,s}(a^1,a^2) \le -w^2(s,h), \forall a^2 \in \mathbb{A}^2(s), \forall \sigma^1(s,h) \in \Delta\mathbb{A}^1(s) \tag{4.16}$$

In (4.14), $\mu^1(\cdot|s,h) = \sigma^1(s,h), \mu^2(\cdot|s,h) = \sigma^2(s,h)$. $R^1_{h,s}$ and $R^2_{h,s}$ are the payoff matrices for the defender and the attacker, respectively. Matrix $R^i_{h,s}$ depends on h and s, and has a dimension of $|\mathbb{A}^1(s)| \times |\mathbb{A}^2(s)|$ where $R^i_{h,s}(a^1,a^2)$ is the element of row a^1 and column a^2. Notations w^1 and w^2 are decision variables in the bilinear program and w^{1*} and w^{2*} are solutions to the bilinear program. After solving the bilinear program, we can also obtain the value at state s and time step h for player i, i.e. $v^i(s, N-h) = -w^{i*}$.

In the case of multiple mixed-strategy Nash equilibria, we choose the equilibrium that maximizes the information entropy of both players' strategies. The strategy with a large entropy possesses a significant amount of uncertainties, and hence it is harder for the other player to learn the strategy. The extreme case of a pure strategy (the action in a pure strategy is deterministic, in contrast with the uncertain actions in a mixed strategy) has the minimal entropy of 0. Then, the observation of the action directly reveals the strategy.

4.3 Risk Assessment

In this section, we focus on the analysis of system evolution under the equilibrium strategies and the resulting risk to the system over the finite horizon H. This analysis is important as it provides us with direct information on the likelihood of the system surviving a cyber attack and the timing of system failures. The PRA model described in Section 4.1.3 is used to determine the state transition probabilities which constitute one essential element of the SMG model, while the risk assessment introduced in this section is based on the developed SMG model.

4.3.1 Risk Metrics

In this research, we focus on three risk metrics. Suppose that at the current decision epoch, there are $N - h^*$ remaining steps and the system is in state s^*, both of which are known to the players. The first metric is the probability that the system reaches a set of undesirable states (e.g., core damage) for the first time at a particular time step from the current step. The second metric is the probability that the system reaches the set of undesirable states before or at a particular time step from the current step. The third metric is the probability distribution of system states at each time step.

We denote the sets of desirable states and undesirable states by $D \subseteq \mathbb{S}$ and $U \subseteq \mathbb{S}$, respectively. We denote the first time step when the system reaches U by T_U. We also assume that at h^*, the system is at a desirable state in D. The first risk metric can be expressed as $\Pr(T_U = h | S_{h^*} = s^*)$, where $S_{h^*} = s^*$ means that at time step h^* the system state is s^*. Since we have discretized the continuous time horizon H into N discrete time steps, $\Pr(T_U = h | S_{h^*} = s^*)$ actually denotes the probability that the system reaches a set of undesirable states for the first time in the time interval between $\frac{H}{N}(h-1)$ and $\frac{H}{N}h$. The second risk metric can be expressed as $\Pr(T_U \leq h|S_{h^*} = s^*)$. The third risk metric can be expressed as $\Pr(s|h, S_{h^*} = s^*)$. We provide two methods for risk assessment: one is the exact analytical method and the other one is based on Monte Carlo simulation.

4.3.2 Exact Analytical Method

With fixed strategies of the defender and the attacker, the discrete time SMG formalized in Section 4.2 can be converted to a discrete-time semi-Markov chain [13]. Similar to $\Pr(s', k|s, a^1, a^2) = p(s'|s, a^1, a^2)\tilde{q}(k|s, a^1, a^2, s')$ in a discrete-time SMG, $\Pr(s', k|s, h)$ in a discrete-time semi-Markov chain is used to describe system transitions, which can be obtained as in (4.17).

$$\Pr(s', k|s, h) = \sum_{a^1 \in \mathbb{A}^1(s), a^2 \in \mathbb{A}^2(s)} \mu^{1^*}(a^1|s, h)\mu^{2^*}(a^2|s, h)p(s'|s, a^1, a^2)\tilde{q}(k|s, a^1, a^2, s'),$$
$$\forall s \in \mathbb{S}, \forall h \in \{0, \ldots, N\}, \forall k \in \{1, \ldots, N-h\}. \tag{4.17}$$

In (4.17), $\mu^{1^*}(\cdot|s, h) = \sigma^{1^*}(s, h)$ and $\mu^{2^*}(\cdot|s, h) = \sigma^{2^*}(s, h)$. Note that h is included in $\Pr(s', k|s, h)$ because the strategies of the defender and attacker are both functions of time over the finite horizon N. This means that the transition is non-stationary. Also, because of the finite horizon,

$$\sum_{s', k \leq N-h} \Pr(s', k|s, h) \leq 1, \tag{4.18}$$

where $1 - \sum\limits_{s',k \leq N-h} \Pr(s', k|s, h)$ is the probability that no transition happens before $N - h$ steps.

The sojourn time distribution also relies on h:

$$\Pr(k|s, h) = \sum_{s'} \Pr(s', k|s, h). \tag{4.19}$$

In contrast to the way of deriving the mixed-strategy NE in a backward fashion, $\Pr(T_U = h|S_{h^*} = s^*)$ can be obtained recursively in the forward fashion as presented in Algorithm 2. In the algorithm, $\Pr(s', h^* + j|S_{h^*} = s^*)$ is the probability that a state jump occurs at $h^* + j$ and the system arrives at s' given that at h^* the system state is s^* and the system has never reached a state in U before $h^* + j$. The algorithm takes into account the fact that the system may jump several times between states in D before the first time arriving at a state in U at $h^* + j$ which is subsequent to h^*. As the algorithm is implemented recursively, it is computationally efficient for arbitrarily large horizon N.

The second risk metric $\Pr(T_U \leq h|S_{h^*} = s^*)$ can be obtained simply by

$$\Pr(T_U \leq h|S_{h^*} = s^*) = \sum_{l=h^*}^{h} \Pr(T_U = l|S_{h^*} = s^*), \forall h \in \{h^*, \ldots, N\}, \tag{4.20}$$

where h^* is the current decision epoch and s^* is the system state at h^*. The derivation of the third risk metric, the probability distribution of system states at each time step, i.e., $\Pr(s|h, S_{h^*} = s^*)$, is less straightforward than the above two metrics. The algorithm for obtaining this risk metric is presented in Algorithm 3. In the algorithm, $\mathrm{Tr}(s', h^* + j|S_{h^*} = s^*)$ represents the probability that a state jump occurs at $h^* + j$ and the system enters state s', given that at h^* the system state is s^*. It takes into account all the possible state jump trajectories from h^*.

Algorithm 2: The algorithm for obtaining the probability mass function of the first arrival time.

```
Input:
$h^*, \forall h^* \in \{0, 1, \ldots, N\}$    // the current time step with $N - h^*$ remaining time steps in the game
$s^*, s^* \in D$ // the current system state and is assumed to be in $D$
$D \subseteq \mathbb{S}, U \subseteq \mathbb{S}$    // the sets of desirable states and undesirable states, respectively
$\Pr(s', k|s, h), h \in \{h^*, \ldots, N\}, s, s' \in \mathbb{S}$  // transition function in the semi-Markov chain
First arrival time:
set $\Pr(s^*, h^*|S_{h^*} = s^*) = 1$    // the state at the time step $h^*$ is $s^* \in D$
for $j = 1$ to $N - h^*$ do
    $\Pr(s', h^* + j|S_{h^*} = s^*) = \sum_{l=0}^{j-1} \sum_{s \in D} \Pr(s, h^* + l|S_{h^*} = s^*) \Pr(s', j - l|s, h^* + l), \forall s' \in \mathbb{S}$
    $\Pr(T_U = h^* + j|S_{h^*} = s^*) = \sum_{s' \in U} \Pr(s', h^* + j|S_{h^*} = s^*)$
Output:
$\Pr(T_U = h|S_{h^*} = s^*), \forall h \in \{h^*, \ldots, N\}$
```

In contrast, $\text{Nr}(s', h^* + j|S_{h^*} = s^*)$ represents the probability that the system stays in state s' at a previous time step and no state jump has occurred by $h^* + j$, given that at h^* the system state is s^*. It also takes into account all the possibilities; i.e., the system can stay in state s' at any of the previous time steps and no state jump has occurred.

4.3.3 Monte Carlo Simulation-Based Method

The Monte Carlo simulation results in a total number of M_s samples, each of which represents a possible system state trajectory starting from state s^* at time h^*.

Based on the samples, we can obtain three groups of values, i.e., $O_1(h|S_{h^*} = s^*)$,

Algorithm 3: The algorithm for obtaining the probability distribution of system states.

```
Input:
$h^*, \forall h^* \in \{0, 1, \ldots, N\}$          // the current time step with $N - h^*$ remaining time steps in the game
$s^*$                        // the current system state
$\Pr(s', k|s, h), h \in \{h^*, \ldots, N\}, s, s' \in \mathbb{S}$   // transition function in the semi-Markov chain
System state probability distribution:
set $\Pr(s^*|h^*, S_{h^*} = s^*) = \mathrm{Tr}(s^*, h^*|S_{h^*} = s^*) = 1$   // the state at $h^*$ is $s^*$
for $j = 1$ to $N - h^*$ do
    $\mathrm{Tr}(s', h^* + j|S_{h^*} = s^*) = \sum_{l=0}^{j-1} \sum_{s \in \mathbb{S}} \mathrm{Tr}(s, h^* + l|S_{h^*} = s^*) \Pr(s', j - l|s, h^* + l), \forall s' \in \mathbb{S}$
    $\mathrm{Nr}(s', h^* + j|S_{h^*} = s^*) = \sum_{l=0}^{j-1} \mathrm{Tr}(s', h^* + l|S_{h^*} = s^*) \left(1 - \sum_{k=1}^{j-l} \sum_{s'' \in \mathbb{S}} \Pr(s'', k|s', h^* + l)\right), \forall s' \in \mathbb{S}$
    $\Pr(s'|h^* + j, S_{h^*} = s^*) = \mathrm{Tr}(s', h^* + j|S_{h^*} = s^*) + \mathrm{Nr}(s', h^* + j|S_{h^*} = s^*)$
Output:
$\Pr(s|h, S_{h^*} = s^*), \forall s \in \mathbb{S}, \forall h \in \{h^*, \ldots, N\}$
```

$O_2(h|S_{h^*} = s^*)$, and $O_3(s|h, S_{h^*} = s^*)$:

$$O_1(h|S_{h^*} = s^*) = \sum_{i=1}^{M_s} \mathbf{1}\{T_U = h \, in \, sample \, i\}, \forall h \in \{h^*, \ldots, N\}, \tag{4.21}$$

$$O_2(h|S_{h^*} = s^*) = \sum_{i=1}^{M_s} \mathbf{1}\{T_U \leq h \, in \, sample \, i\}, \forall h \in \{h^*, \ldots, N\}, \tag{4.22}$$

$$O_3(s|h, S_{h^*} = s^*) = \sum_{i=1}^{M_s} \mathbf{1}\{S_h = s \, in \, sample \, i\}, \forall s \in \mathbb{S}, \forall h \in \{h^*, \ldots, N\}, \tag{4.23}$$

where $O_1(h|S_{h^*} = s^*)$ is the number of samples where the first arrival time is h

given that at h^* the system state is s^*, $O_2(h|S_{h^*} = s^*)$ is the number of samples where the first arrival time is before or at h given that at h^* the system state is s^*, and $O_3(s|h, S_{h^*} = s^*)$ is the number of samples where the system is at state s at time step h given that at h^* the system state is s^*. Note that in determining $O_3(s|h, S_{h^*} = s^*)$, for any time step between two adjacent decision epochs, the system is at the same state as the first of the two decision epochs since no state transition occurs before the second of the two decision epochs. As an example, assume two adjacent decision epochs are at time steps 10 and 20, respectively, and the system is in s and s' at the two decision epochs, respectively. Then for any time step between 10 and 20, the system is in state s. Based on $O_1(h|S_{h^*} = s^*)$, $O_2(h|S_{h^*} = s^*)$, and $O_3(s|h, S_{h^*} = s^*)$, the three risk metrics can be obtained as follows:

$$\Pr(T_U = h|S_{h^*} = s^*) = \frac{O_1(h|S_{h^*} = s^*)}{M_s}, \forall h \in \{h^*, \ldots, N\}, \tag{4.24}$$

$$\Pr(T_U \leq h|S_{h^*} = s^*) = \frac{O_2(h|S_{h^*} = s^*)}{M_s}, \forall h \in \{h^*, \ldots, N\}, \tag{4.25}$$

$$\Pr(s|h, S_{h^*} = s^*) = \frac{O_3(s|h, S_{h^*} = s^*)}{M_s}, \forall s \in \mathbb{S}, \forall h \in \{h^*, \ldots, N\}. \tag{4.26}$$

4.4 Case Study

In the case study, we focus on a simplified digital feedwater control system of a generic pressurized water reactor. During the normal operation or transients caused by abnormal events, this control system controls components in the main feedwater system (e.g., feedwater pump, feedwater flow regulating valves) to maintain sufficient water flow for the steam generator to cool the reactor core. Failures or compromises of this control system may lead to a dry-out of the steam generator and core damage. The system in this case study is taken from the U.S. NRC report NUREG/CR-6942 [7], which was originally used as a benchmark system for dynamic reliability analysis of digital systems in nuclear power plants. Simplifications and modifications have been made to the original system to simplify the analysis.

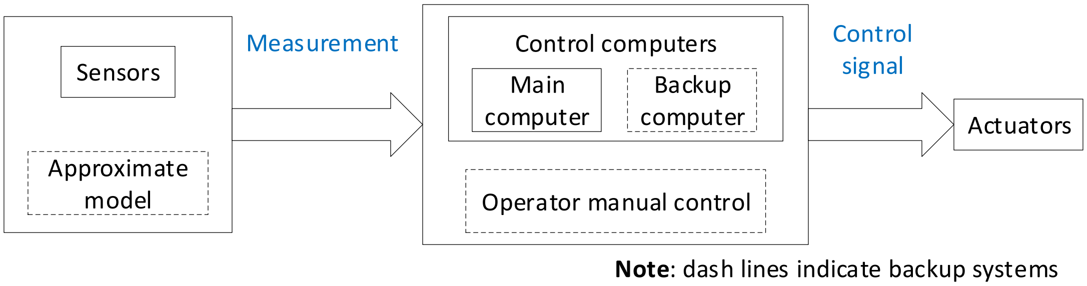


Figure 4.5: The digital feedwater control system.

The function of the system is similar to the one of a PLC. It consists of three main components as shown in Figure 4.5. The sensors provide the information on the state of the plant (e.g., water level in the steam generator, feedwater flow, steam flow). The information is then sent to the computer that implements the control algorithm. There are two computers that can be used. During the normal operation, the main computer is used. In the case of main computer failures or compromises, the backup computer takes over the control. In certain cases, the operators will take over the control, so the control transitions from the mode of automatic control to manual control. The control signal calculated in either the main or the backup computer, or obtained from the operators is then sent to field actuators (e.g., feedwater pump actuator, feedwater flow regulating valves actuators). In this research, we focus on cyber attacks on sensors, the main computer, and the backup computer. A potential type of attack on sensors is a false data injection attack [130]. The two computers can be attacked by installing malware, as in the attack on Iran's nuclear facilities [48]. Note that the proposed method is not limited to these attacks. The exact type of attack launched on a component may not matter because the objective of an attack is to compromise the function of the component and we only need to focus on whether the function of the component is compromised or not.

The benchmark system was originally used for reliability analysis, rather than cyber-security analysis, so there was no particular consideration of countermeasures against cyber attacks. In our research, an approximate linear model is considered as such a countermeasure to attacks on sensors [22], which approximates the dynamics of the plant and serves as a backup to the sensors. However, as discussed in [22], such an approximate model is a temporary solution because the adoption of the approximate model may cause disturbances to the plant due to the inaccuracy of the model output. In this case study, we assume the objective of the attacker is to damage the reactor core (as what will happen in a terrorist attack) and the defender aims to minimize the damage. Similarly, it is worth noting that the proposed approach is not limited to these assumptions and can be extended to other types of attack and defense objectives.

In this case study, we focus on a time horizon of $H = 1440$ min (i.e. 24 h). This time horizon is adopted for two reasons. First, in a typical PRA analysis, 24 h is usually considered as the mission time, because normally after 24 h the reactor core is either damaged or should have been maintained at a steady state. Second, we believe that the plant emergency support team should be able to terminate the attacker's attacks within 24 h, and hence the potential actions of the attacker beyond 24 h do not need to be considered. In other applications, the time horizon may be determined based on the specifics of the problem. We discretized H into

1440 time steps by using a time interval of 1 min. In this case study, we have also tested time intervals of 0.5 min and 0.1 min. The difference between the results is small, but using a smaller time interval requires more computation. Therefore, in this case study we used a time interval of 1 min in the discretization.

4.4.1 State Space and Action Space

Table 4.1 presents the elements that are used for defining system states in the case study. In this case study, we do not explicitly include the main feedwater system because its states can be defined by its components. Besides, in this research we have focused on the impact of the attacker's actions on the system. Therefore, we have focused on the normal and compromised states of each component, and the states of a component related to hardware failure, maintenance, test, etc., are not fully considered. The actions available to the defender and attacker are presented in Table 4.2 and Table 4.3, respectively.

All the possible combinations of states of the elements will lead to $|\mathbb{S}^e| = 4 \times 4 \times 4 \times 2 \times 2 \times 2 = 512$ system states. We followed the systematic investigation method introduced in Section 4.1.2 to reduce the system state space. For this system, the initial state is the normal system state, where all the elements in Table 4.1 are in their normal states, the sensors provide the measurements of the plant status and the main computer is used for automatic control. Based on this method, we finally identified sixteen system states, i.e., $|\mathbb{S}| = 16$. The identified system states are presented in Table 4.4, along with the states of the six elements that are used to define the system states. The actions available to the defender and attacker at each system state are presented in Table 4.5.

As discussed in Section 4.1.2, the analysis also leads to the information on the

Table 4.1: Elements considered for the system and the states of the elements.

Element No.	Element name	Element states
1 (SENS)	Sensors	1. Normal, used; 2. Normal, not used; 3. Compromised, used; 4. Compromised, not used.
2 (MC)	Main computer	1. Normal, used; 2. Normal, not used; 3. Compromised, used; 4. Compromised, not used.
3 (BC)	Backup computer	1. Normal, used; 2. Normal; not used; 3. Compromised, used; 4. Compromised, not used.
4 (CM)	Control mode	1. Automatic control; 2. Manual control.
5 (AM)	Approximate model	1. Not used; 2. Used.
6 (RC)	Reactor core	1. OK; 2. Damaged.

state transitions at each state under different action pairs. The transitions are presented in Figure 4.6. The nodes denote the system states and the directed links between the nodes denote the feasible transitions between states. Note that to make the figure readable, only one directed link is shown between two states. Actually, there might be multiple links between two states that correspond to different action pairs. In solving the SMG, all the state transitions (i.e. directed links) are used.

To explain the process of identifying the system states and state transitions, take the first step of the analysis in the case study as an example. In the first step, we start with the initial normal state, which is system state 1 in Table 4.4, and hence q here is system state 1. In this state, the sensor (SENS) is in a normal state and used in control, the main computer (MC) is in a normal state and used,

Table 4.2: The actions available to the defender.

No.	Action	Description
1	Use the output of the approximate model instead of the sensor measurements in automatic or manual control	The approximate model is able to provide information about the status of the plant, with a similar function as the sensors. However, the information is approximate, so the use of the model may cause disturbances and even lead to core damage. This model is specifically designed to defend against cyber-attacks, so in this research we assume it cannot be compromised by the attacker. Again, the proposed approach is not limited to this assumption.
2	Use the backup computer to implement the control algorithm	The backup computer will provide the same function as the main computer.
3	Control the system manually	Manual control is immune to cyber-attacks, but may cause disturbances to the system.
4	No action	This option is included to reflect the fact that the defender may not take any action.

the backup computer (BC) is in a normal state but not used, the control (CM) is in an automatic control mode, the approximate model (AM) is not used, and the reactor core (RC) is OK. In this state q, the feasible action set for the defender is $\mathbb{A}^1(q) = \{1, 2, 4\}$, and the feasible action set for the attacker is $\mathbb{A}^2(q) = \{1, 2, 4\}$, as shown in the row corresponding to system state 1 in Table 4.5. For a detailed explanation of each action, please refer to Tables 4.2 and 4.3. Assume that the action pair being considered is $(a^1 = 4, a^2 = 1)$, where action 4 by the defender is "no action" and action 1 by the attacker is "compromise sensors" as shown in Tables 4.2 and 4.3, respectively. The attacker's action may not be effective, and in this case the system stays in the initial state, which is system state 1. If the attacker's action is effective, then the sensors are compromised and used. This corresponds to system state 2 in Table 4.4. There is also a negligible probability that the reactor core will be damaged due to other component failures not considered in this model,

Table 4.3: The actions available to the attacker.

No.	Action	Description
1	Compromise sensors	False data can be injected into the sensors, which results in false inputs to the computers or the operator. Control signals generated or decisions made based on these false inputs will lead to inadequate feedwater flow to the steam generator which may lead to core damage.
2	Compromise main computer	A compromised computer will lead to inappropriate control signals sent to field actuators, even if the inputs from the sensors or the approximate model are correct.
3	Compromise backup computer	The same reasoning holds as for an attack on the main computer.
4	No action	The attacker may not take any action.

and this corresponds to system state 16 in Table 4.4. Therefore, in the case of $q = 1$ and $(a^1 = 4, a^2 = 1)$, we have identified three successor states $q' = 1$, $q' = 2$, or $q' = 16$. The analyses for any other action pairs are similar. Having identified all the successor states q', we can now set each of the q' as q and perform a similar analysis to identify all the successor states that can be reached from the new q. In identifying the 16 system states in Table 4.4, we also group some of the states. For example, if the reactor core is damaged, which indicates the end of the game, the states of the other elements (e.g. sensors) no longer matter. Therefore, we can group these states into one state, as shown in system state 16 in Table 4.4.

As will be expected, the state-space size of the SMG model is affected by the number of components (e.g. the number of sensors in the case study) that comprise the system under study. As the number of components in the system increases, the size of the state space does increase significantly, which is a common problem for discrete-state Markov models identified as the *curse of dimensionality*. This problem may be partially circumvented in three ways. First, by using the algorithm

Table 4.4: Sixteen system states identified for the case study.

System	State of the six elements					
state No.	1 (SENS)	2 (MC)	3 (BC)	4 (CM)	5 (AM)	6 (RC)
1	1	1	2	1	1	1
2	3	1	2	1	1	1
3	1	3	2	1	2	1
4	1	✓*	1	1	1	1
5	3	3	2	1	1	1
6	3	✓	1	1	1	1
7	✓	1	2	1	2	1
8	1	✓	3	1	1	1
9	✓	3	2	1	2	1
10	3	✓	3	1	1	1
11	1	✓	✓	2	1	1
12	✓	✓	1	1	2	1
13	3	✓	✓	2	1	1
14	✓	✓	3	1	2	1
15	✓	✓	✓	2	2	1
16	×**	×	×	×	×	2

* ✓ means that for the system state (row), the element (column) is not used and can be in any of the corresponding states (normal or compromised).

** × means that for the system state (row), the element (column) can be in any of its states.

introduced in Section 4.1.2, we are able to restrict the state space to a small subset of feasible states. As illustrated in the case study, we have reduced the system state space from 512 possible states to only 16 feasible states. Second, depending on the specific problem in consideration, the state-space size may be reduced by grouping components. For example, multiple sensors that provide redundant functions may be grouped and considered in the modeling as one single component. Third, the analysis for NE strategy and risk assessment can be performed in an offline fashion, and therefore the impact of state-space size increase is reduced.

Table 4.5: The actions available to the defender and attacker at each system state.

System	Defender action				Attacker action			
state No.	1	2	3	4	1	2	3	4
1	✓*	✓	×**	✓	✓	✓	×	✓
2	✓	✓	×	✓	×	✓	×	✓
3	✓	✓	×	✓	✓	×	×	✓
4	✓	×	✓	✓	✓	×	✓	✓
5	✓	✓	×	✓	×	×	×	✓
6	✓	×	✓	✓	×	×	✓	✓
7	×	✓	×	✓	×	✓	×	✓
8	✓	×	✓	✓	✓	×	×	✓
9	×	✓	×	✓	×	×	×	✓
10	✓	×	✓	✓	×	×	×	✓
11	✓	×	×	✓	✓	×	×	✓
12	×	×	✓	✓	×	×	✓	✓
13	✓	×	×	✓	×	×	×	✓
14	×	×	✓	✓	×	×	×	✓
15	×	×	×	✓	×	×	×	✓
16	×	×	×	✓	×	×	×	✓

* ✓ means that the action (column) for the defender or attacker is available at the system state (row).
** × means that the action (column) for the defender or attacker is not available at the system state (row).

4.4.2 State Transition Function and Payoffs

The transition probabilities are computed following the method introduced in Section 4.1.3. When the defender has taken an action, we assume that the action is always effective. However, it is worth noting that the proposed framework is not limited to this assumption. In the case where the system has been compromised to a significant extent, which is captured in the system state, we should consider the situation where the defender's action may not be effective even if the action has been executed. In this case, sensitivity analysis can be performed to assess the effect of this assumption on the results (i.e., cyber-attack response strategy and risk metrics) of the study, and this will be part of our future research. When the

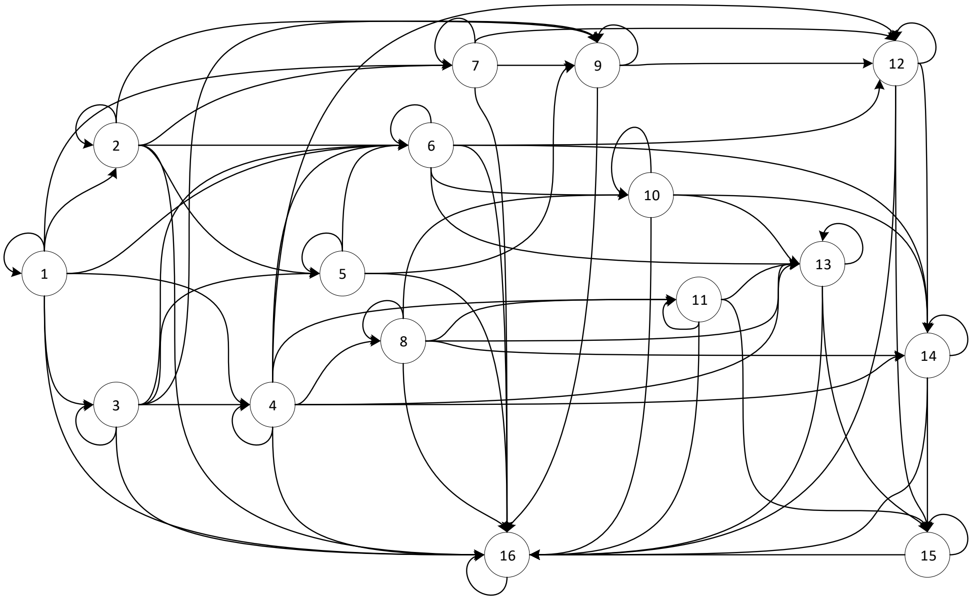


Figure 4.6: State transitions identified by systematic state investigation. Between two states, multiple directed links may exist under different action pairs, yet only one directed link between two states is shown to make the figure readable.

attacker takes an action, we compute the probability of success as follows [170]:

$$p = 2 \times S_{AV} \times S_{AC} \times S_{AU}, \tag{4.27}$$

where S_{AV}, S_{AC}, and S_{AU} represent the access vector, the access complexity, and the authentication that constitutes the exploitability subscore of the base group of metrics in the CVSS [139]. In this case study, we take the lowest values of the three factors (0.395, 0.35, and 0.45, respectively) to obtain a success probability of 0.12 for the attacker's action. As discussed before, sensitivity analysis can be performed for this model parameter. When the defender and attacker take actions on the same element (e.g., sensors, main computer, and backup computer), we assume that the defender's action dominates the attacker's action. For example, when the defender's action is to switch from sensors to the approximate model, the system will transition to the state where the approximate model is used with probability one regardless of the attacker's action on the sensors. Again, sensitivity analysis can be performed for this model parameter and will be part of our future research. For the sojourn time, we refer to the study in [29], where cyber-attack scenarios were studied based on modeling and simulation. Based on the timings in the sample scenarios in [29], we assume that the sojourn time for the transition between two states without core damage follows a uniform distribution between 0 and 10 min. However, as noted in [29], the timing of the scenarios vary significantly. Therefore, sensitivity analysis can be performed to assess the effect of the uncertainty in this model parameter on the results (i.e., cyber-attack response strategy and risk metrics) of this study.

The probability of transition from a state without core damage to a state with core damage is calculated based on the PRA model shown in Figure 4.7. The event tree in the PRA model refers to FIGURE I 4-11 in Appendix i of WASH-1400 [158]. In WASH-1400, the probabilities of "reactor shutdown failure" and "auxiliary feedwater system failure" are both 0.0001. To reflect the fact that these two systems may also be affected by the cyber attack, the probabilities of their failures are increased by one order of magnitude to 0.001 in this case study. This value is used in this paper for demonstration. In real-world applications, this value can be determined by subject matter experts or engineers based on the specific system configuration. The basic event "disturbance" in the PRA model reflects the fact that the main feedwater system can fail when the approximate model is adopted and does not capture the system dynamics. This basic event is assigned a probability of 0.01. This value is for demonstration in this paper and can be further determined by domain experts or engineers. The basic event "human error" reflects the fact that during the manual control, human error can cause the failure of the main feedwater system. The baseline human error probability, i.e., 0.011, from SPAR-H [59] is used in this case study. The probability of the basic event "hardware failure, maintenance, test" and the probabilities in the event tree are both taken from WASH-1400 [158]. These probabilities in gray are fixed values in this case study as they are not considered in the game. To determine the sojourn time for the transition from a state without core damage to the state with core damage, we refer to the accident analysis in WASH-1400 [159]. For transients, the time at which core damage begins ranges from 120 to 720 min following the accident. Therefore, we assume the sojourn time follows a uniform distribution between 120 and 720 min. For the state with core damage, as the reactor core is already damaged, the system state will no longer change.

As an example to illustrate the above computation, consider that the system state under study is state 1 in Table 4.4 and we aim to obtain the probabilities and sojourn time distributions for the system transitioning from state 1 to other states. According to the definition of state 1, we can assign the basic events in the PRA model as follows: "sensor used and compromised"-"FALSE"; "approximate model used"-"FALSE"; "main computer used and compromised"-"FALSE"; "backup computer used and compromised"-"FALSE"; and "manual operation"-"FALSE." The probabilities of main feedwater system failure and core damage can then be obtained as 0.01 and 10^{-5}, respectively. Therefore, the probability of the system state transition from state 1 to state 16, in which the core is damaged, can be determined as 10^{-5}. As discussed beforehand, the sojourn time follows a uniform distribution between 120 and 720 min. The remaining probability (i.e., $1 - 10^{-5}$) is split according to the defender's and the attacker's actions. For example, if the

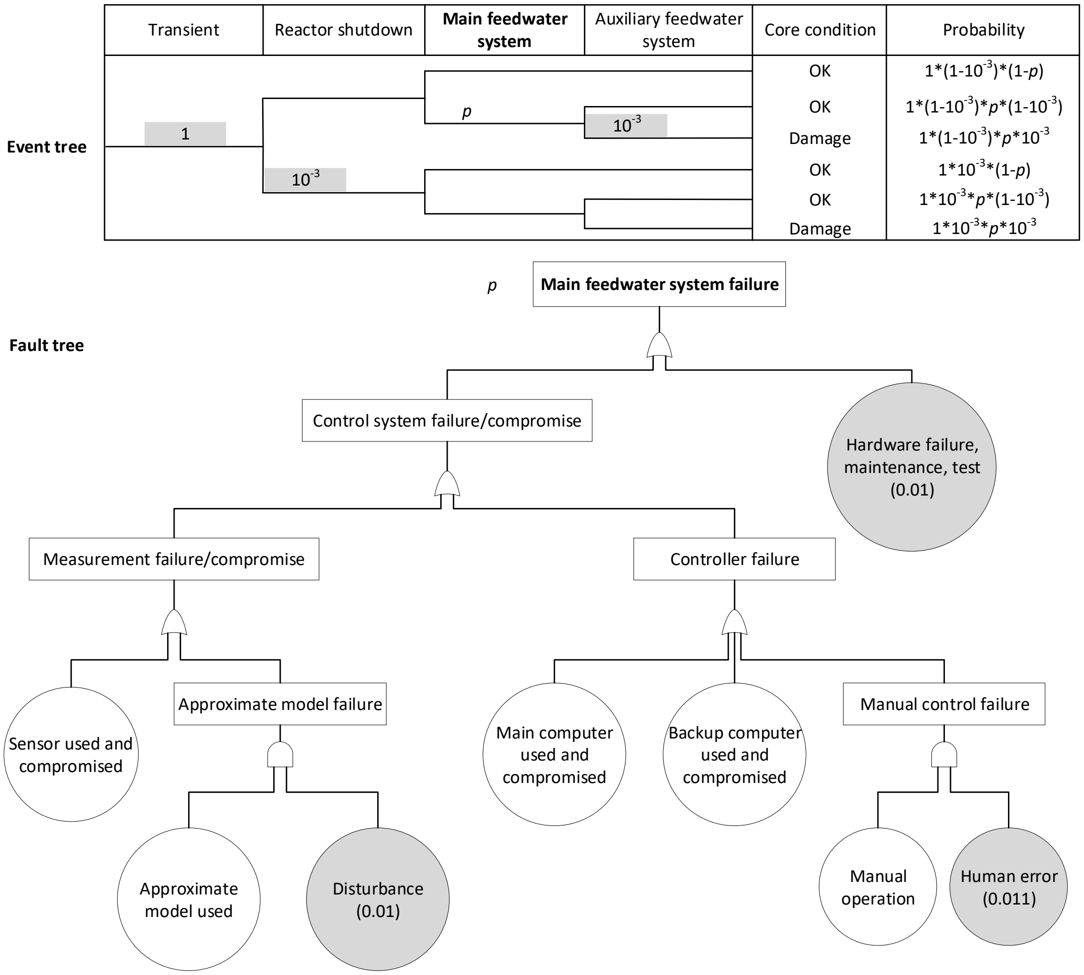

Transient
Reactor shutdown
Main feedwater system
Auxiliary feedwater system
Core condition
Probability
Event tree
1
10^{-3}
p
10^{-3}
OK
OK
Damage
OK
OK
Damage
$1*(1-10^{-3})*(1-p)$
$1*(1-10^{-3})*p*(1-10^{-3})$
$1*(1-10^{-3})*p*10^{-3}$
$1*10^{-3}*(1-p)$
$1*10^{-3}*p*(1-10^{-3})$
$1*10^{-3}*p*10^{-3}$
Fault tree
p
Main feedwater system failure
Control system failure/compromise
Hardware failure, maintenance, test (0.01)
Measurement failure/compromise
Controller failure
Sensor used and compromised
Approximate model failure
Approximate model used
Disturbance (0.01)
Main computer used and compromised
Backup computer used and compromised
Manual control failure
Manual operation
Human error (0.011)

Figure 4.7: The PRA model, i.e., event tree [158] and fault tree, used in this case study.

defender takes action 1 (i.e., switch to the approximate model) and the attacker takes action 1 (i.e., compromise sensors), the system will transition to state 7 with probability $1 - 10^{-5}$. The sojourn time for this transition follows a uniform distribution between 0 and 10 min as discussed earlier.

For the defender, we assume that the lump-sum payoff for taking actions, $r^{1,1}(s_j, a_j^1)$, in (4.5) is zero. When the state transition does not involve core damage, the lump-sum payoff for system transition, $r^{1,2}(s_j, s_{j+1})$, in (4.5) is zero. If the system state transition from a state without core damage to the state with core damage occurs, the lump-sum payoff for system transition, $r^{1,2}(s_j, s_{j+1})$, in (4.5) is assumed to be minus ten billion dollars. The payoff value refers to the studies on societal risk for nuclear power plants in [36, 150], which take into account both the onsite cost and the offsite cost due to nuclear accidents. Compared with these two types of payoffs, the duration payoff, $r^{1,3}(s_j, t_j)$, in (4.5) is negligible in our application and is assumed to be zero.

For the attacker, we assume that the lump-sum payoff for taking actions, $r^{2,1}(s_j, a_j^2)$ is minus ten thousand dollars for any action except the fourth one "no action." This value reflects the fact that any action during a cyber-attack may help the defender attribute the individual attacker or the group launching the attack [43], therefore inducing costs to the attacker. This value is assumed to be one hundredth of the lower bound of the statistical estimate of human life value [36, 230]. As for other model parameters, the effect of the uncertainty in this parameter on the results can be studied based on sensitivity analysis. For state transition payoffs, if the system state transition from a state without core damage to a state with core damage occurs, the attacker earns a payoff of ten billion dollars. Otherwise, the payoff for state transition is zero. In our application, the duration payoff, $r^{2,3}(s_j, t_j)$, in (4.5) is negligible and is assumed to be zero.

4.5 Results and Discussion

Section 4.5.1 presents the NE and the value of the game for the defender at each state. The real-time risk under the mixed-strategy equilibrium is calculated in Section 4.5.2. The comparison between the optimal strategy at the equilibrium and a baseline strategy is presented in Section 4.5.3.

4.5.1 Nash Equilibrium and State Value

The defender's equilibrium strategy at each system state is presented in Figure 4.8. The strategy can be analyzed as follows. Taking the strategy at state 1 as an example, based on this strategy, the defender's optimal response is to take action 1 (use approximate model) and action 2 (use backup computer) with probabilities of 0.52 and 0.48, respectively, for most time of the horizon from 0 to about 1280 min. Action 4 (no action) is preferable for a short time period at the late stage of the horizon. From 1320 min to the end of the game, the defender's optimal response is to choose from the three alternative actions randomly. There is no difference between the three actions in the very late stage of the game because each of them will lead to the same payoff. This can also be used to explain the indifference between available actions in the very late stage of the game for any other system state in Figure 4.8. Taking state 7 as another example, the optimal response is to take action 2 at the beginning. As the game approaches the end, the probability of taking action 4 (no action) increases. The defender's strategy at any other state can be explained in a similar way.

Note that in a finite-horizon game, the strategy is a function of state and time. From the result, we can see that the preferable action changes with states. For instance, at the beginning of the game, at state 2, action 1 is favorable compared to actions 2 and 4, while at state 3, action 2 is favorable compared to actions 1 and 4. Besides, we can see that for certain states the strategy is nondeterministic and the preferable action can change significantly as time evolves in the game. For instance, at states 11 and 12, action 4 is not preferable at the beginning but

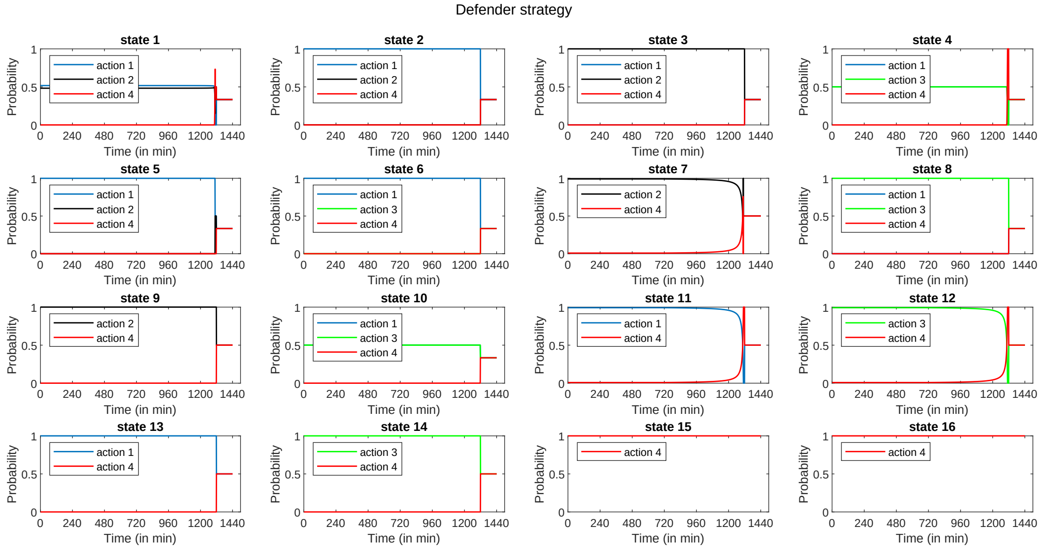


Figure 4.8: The defender's strategy over the entire horizon $H = 1440$ (in min).

becomes preferable as the game approaches the end. We also observe that we can completely ignore certain actions at certain states. For instance, at states 3 and 8, action 1 is never a better choice during the entire process of the game.

In real-world applications, the best way of using the defense strategy obtained from the game-theoretic analysis is to first determine the current system state and the time from the beginning of the game, and then to sample from the corresponding probability distribution described in the optimal defense strategy. For example, if the system is in state 1 and the game has just begun, then the defender (i.e. operator) has three optional actions, that is action 1 (i.e. "use approximate model"), action 2 (i.e. "use backup computer"), and action 4 (i.e. "no action"). According to the optimal strategy, the probabilities for the three options are about 0.52, 0.48, and 0, respectively. Then the defender should sample from this distribution to determine which action to take. In certain cases (e.g. system state 1), there are uncertainties in the strategy with respect to the actions, while in other cases (e.g. system state 2), the strategy is almost deterministic. By introducing uncertainty on purpose, the defender makes his/her defense less predictable and more effective against attacks. This kind of strategy is different from (deterministic) guidelines currently used in nuclear power plants.

The attacker's strategy is presented in Figure 4.9 and can be explained in a similar way as in the case of the defender's strategy. For instance, at state 1, the attacker's optimal strategy is to take action 1 (compromise sensors) and action 2 (compromise main computer) with probabilities of 0.28 and 0.72 respectively at the early stage of the game. As the game approaches the end, the probability of taking action 1 increases. From 1311 min, the optimal strategy is to take action 4 (no action). For simplicity, in this case study we have assumed that if a component is not being used, then the attacker can not compromise the component. Therefore, in states 5 and 9, the only action that is available to the attacker is action 4, i.e. "no action." For states 10, 13, 14, and 15, the only action that is available to the attacker is also "no action" because in these states all the digital components being used (i.e. sensors, main computer, and backup computer) have already been compromised.

The defender's value $v^1(s, N-h)$ at each state s as a function of time step h from the beginning of the game is presented in Figure 4.10. We can see that for states from 1 to 15, where there is no core damage, the defender's value increases from around -7×10^7 USD as time elapses. This is reasonable because when there is less time for the attacker to take actions, it is less likely that the attacker can compromise the system and cause damage to the core. The defender's value is constant at 0 for state 16 because at this state the core has already been damaged and there is no further reward or cost to the defender.

As a comparison of the values at different states, the values at states 1, 8,

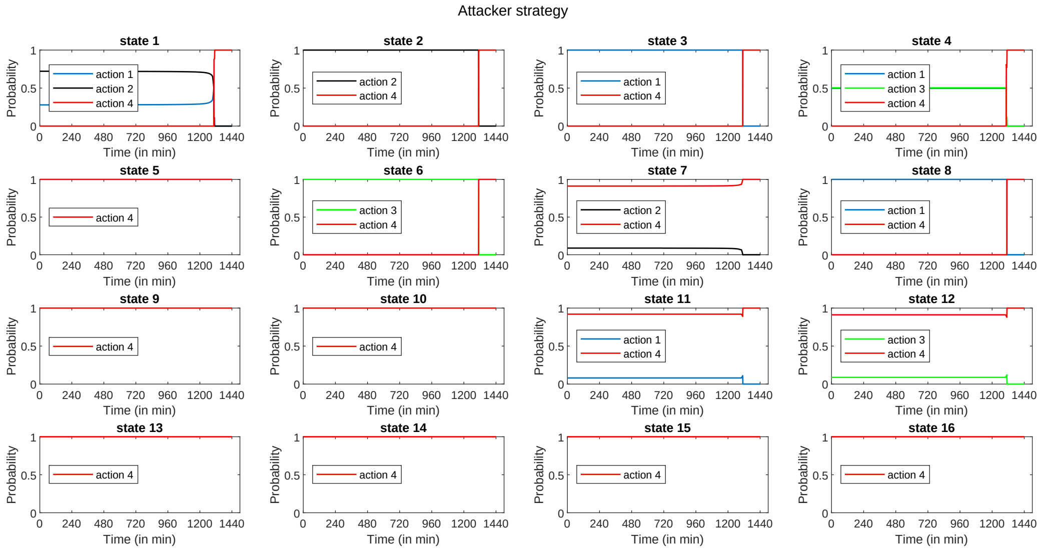


Figure 4.9: The attacker's strategy over the entire horizon $H = 1440$ (in min).

and 10 are plotted in Figure 4.11. The three states represent the cases where all the components being used are normal (state 1), one component being used is compromised (state 8), and two components being used are compromised (state 10). The result is reasonable as the value at state 1 is always greater than the value at state 8, which is always greater than the value at state 10, for the time period between 0 and 1319 min. From 1320 min, the values at the three states all become 0.

4.5.2 Risk Metrics

We performed two scenarios for risk assessment. In the first scenario, the game starts with state $s^* = 1$ at time step $h^* = 0$ (i.e. 0 min). In the second scenario, we assume the game has progressed to time step $h^* = 1200$ (i.e. 1200 min) and the system is at state $s^* = 1$.

The results for the three risk metrics, i.e., $\Pr(T_U = h|S_{h^*} = s^*)$, $\Pr(T_U \leq$

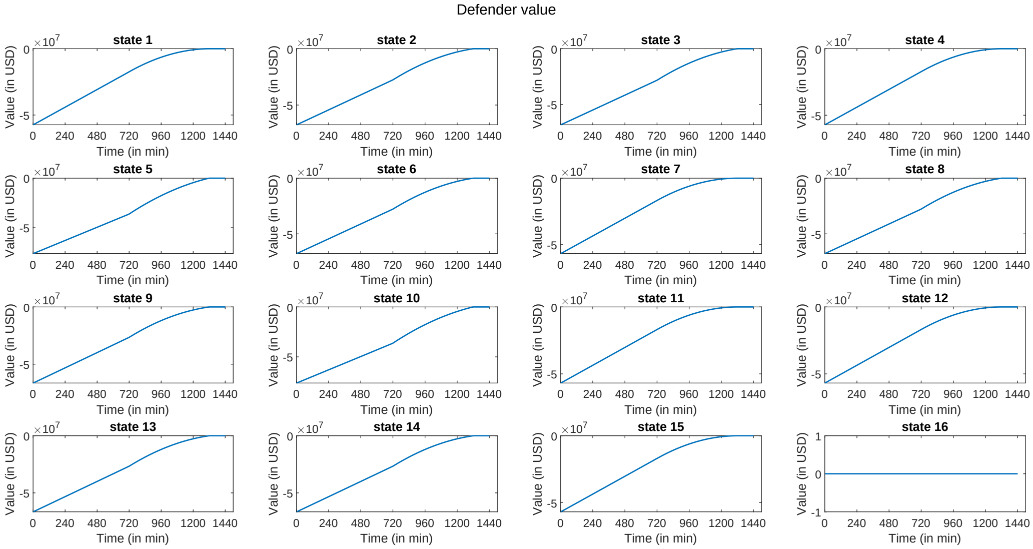


Figure 4.10: The defender's value at each state over the entire horizon $H = 1440$ (in min).

$h|S_{h^*} = s^*)$, and $\Pr(s|h, S_{h^*} = s^*)$, obtained based on the exact analytical method for the first scenario (where $h^* = 0$ and $s^* = 1$) are presented in Figure 4.12, Figure 4.13, and Figure 4.14, respectively. As a comparison and verification, the result for the second risk metric obtained based on Monte Carlo simulation with 500000 samples is also plotted in Figure 4.13. From Figure 4.12, we can see that the probability that the system arrives at the core damage state for the first time remains at 0 at the beginning of the game, increases as time evolves from 121 min to 740 min, and remains at an almost constant level of 5.6×10^{-6} from 741 min.

From Figure 4.13, we can see that the probability that the system reaches the core damage state before or at a time point increases slowly at the beginning and then almost linearly in the late stage of the game. Given the assumption that the system starts with state 1 at time 0, we can see that the core will be damaged with probability of 0.0057 before or at the end of the game. The results obtained based on the exact method and Monte Carlo simulation are very close to each

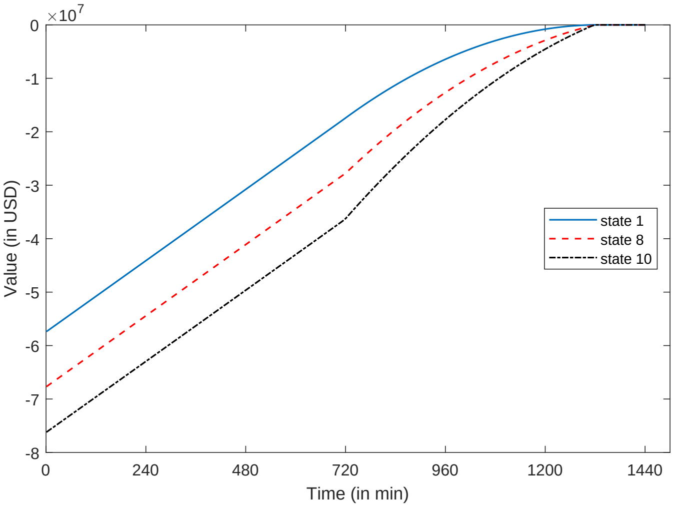


Figure 4.11: Comparison of the defender's values at states 1, 8, and 10.

other, which provides verification to each method.

From Figure 4.14, we can see that the probability of state 1 drops quickly to 0 at the beginning of the game. This is because at state 1, the defender's optimal strategy is to switch from the sensors to the approximate model or to switch from the main computer to the backup computer as shown in Figure 4.8, and the attacker's optimal strategy is to compromise the sensors or to compromise the main computer as shown in Figure 4.9. Correspondingly, the probabilities of states 4, 6, 7 and 9, which are the states resulting from the actions of the players, increase quickly at the beginning of the game. From these four states and based on the strategies of the players, the successor states can be analyzed to explain the state distribution versus time in Figure 4.14. From Figure 4.14, we can observe that after leaving state 1, it is very likely that the system will shortly visit states 4, 6 7, 9, 11, and 12, and will spend most of the time in state 15. It is also easy to observe that the system will never visit certain states according to the strategies of the players, for instance state 5.

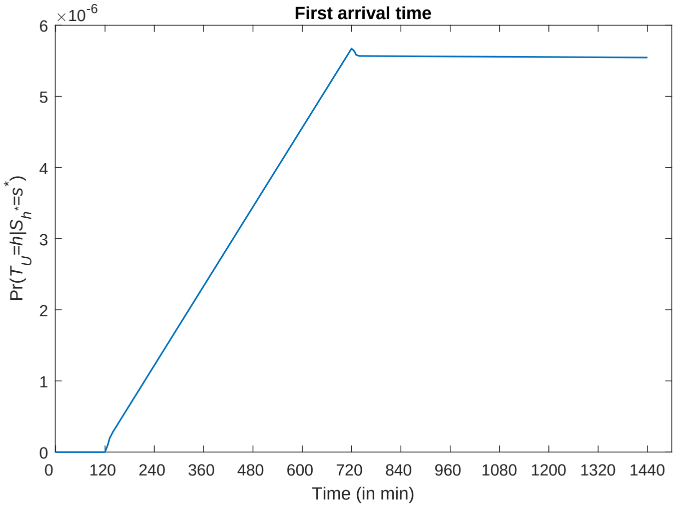


Figure 4.12: The result for the first risk metric, i.e., $\Pr(T_U = h | S_{h^*} = s^*)$, obtained based on the exact method.

As introduced in Section 4.3.2, we can analyze the risk metrics starting at any time in the game and the risk metrics update persistently according to the amount of time remaining. In the analysis presented above, we assume that the game has just started at time 0 and the initial state is 1. As a comparison, in the second scenario, we assume the game has progressed to the time 1200 min (i.e. time step 1200) and the current state is 1. The result for state distribution is presented in Figure 4.15. We can see that now the probability of core damage at the end of the game is 1.22×10^{-6}, in contrast to 0.0057 in the first scenario. This capability of real-time risk assessment is important because as we have moved to a time point in the middle of the game and we observe the state of the system, we would like to predict the state distribution starting from the current time and with the current state.

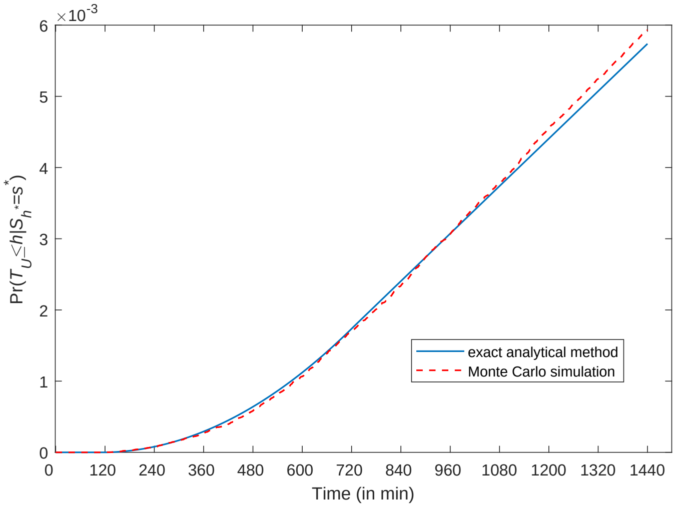


Figure 4.13: The result for the second risk metric, i.e., $\Pr(T_U \leq h|S_{h^*} = s^*)$, obtained based on the exact method and Monte Carlo simulation.

4.5.3 Comparison between the Equilibrium Strategy and a Baseline Strategy

In this subsection, we compare the equilibrium strategy and a baseline strategy of the defender to illustrate the benefit gained from the game-theoretic analysis. The equilibrium strategy is the one presented in Figure 4.8 and the baseline strategy is presented in Table 4.6. This baseline strategy is stationary, meaning that it does not vary with time in the game. It can be summarized as the following decision rules: 1) in the cases where no component in use is compromised, the defender takes action 4 (no action); 2) in the cases where either sensors, the main computer, or the backup computer is used but compromised, the defender takes action 1 (switch to the approximate model), action 2 (switch to the backup computer), or action 3 (switch to manual control), respectively; 3) in the cases where the sensors and the main computer are used but compromised, or the sensors and backup computer are used but compromised, the defender takes action 2 or action 3, respectively; 4)

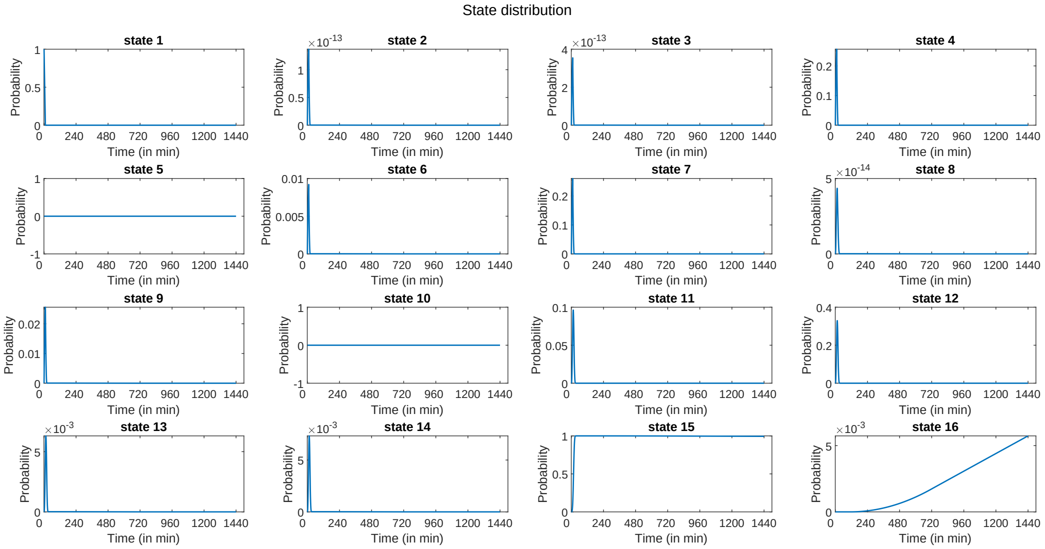


Figure 4.14: The result for the third risk metric, i.e., $\Pr(s|h, S_{h^*} = s^*)$, obtained based on the exact method.

in all other cases, the defender takes action 4. This baseline strategy is similar to procedures used in current practices in cyber-attack response.

Table 4.6: The baseline strategy of the defender.

System	Defender action				System	Defender action			
state No.	1	2	3	4	state No.	1	2	3	4
1				✓	9		✓		
2	✓				10			✓	
3		✓			11				✓
4				✓	12				✓
5		✓			13	✓			
6	✓				14			✓	
7				✓	15				✓
8			✓		16				✓

Note that in this comparison study, no matter what strategy the defender takes, the attacker always takes the mixed strategy at the NE presented in Figure 4.9. This can be explained and justified as follows. First, the focus in this comparison study is on the defender's strategy instead of the formulation of the game. This is

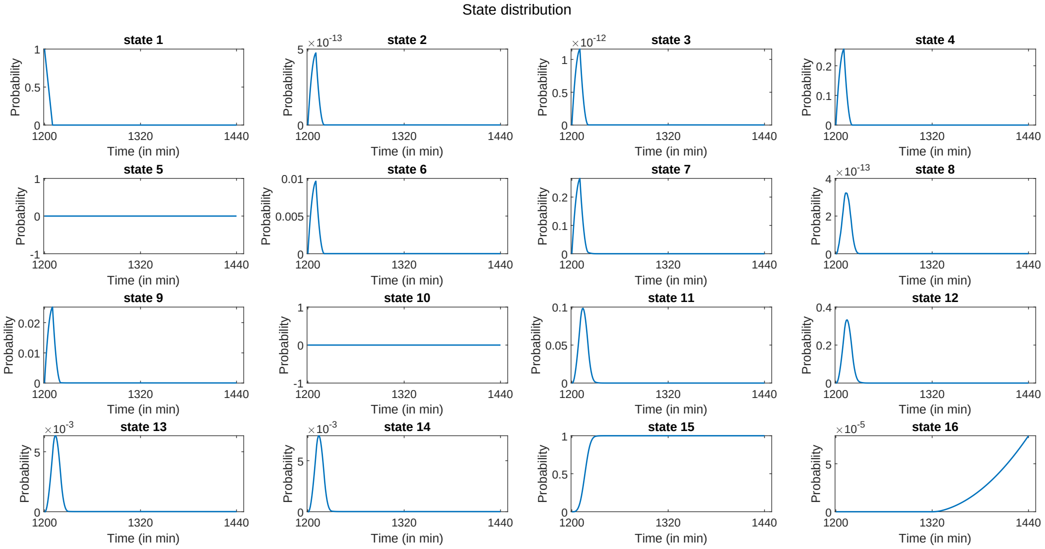


Figure 4.15: State distribution assuming the current time is 1200 min from the beginning of the game and the current state is 1.

part of the reason we fix the attacker's strategy at the NE in both cases. Second, each player actually does not know how the opposite is thinking of the game and what actions the opposite is going to take. Instead, each player just assumes how the opposite thinks of the game and what actions the opposite is going to take to maximize his/her payoff. If either player makes decisions by taking the opposite's thinking and decision-making into consideration, then the solution concept of NE is followed, which is the case in this research. It is also possible that the attacker makes decisions by taking the defender's thinking and decision-making (based on the attacker's assumption) into consideration, while the defender makes decisions without such considerations, which is the case of the baseline defender strategy.

In this comparison study, the game starts at time 0 and state 1. We compare the optimal strategy and the baseline strategy based on two measures. The first measure is the expected cumulative payoff of the defender over the 24 h finite horizon that can be calculated following (4.13). The results with the two strategies are presented in Figure 4.16. The results can be interpreted as follows. For example, if the defender adopts the optimal strategy at time 0 and state 1, the expected cumulative payoff that can be obtained after 1440 min (24 h) is -5.74×10^7 USD, in contrast with -7.19×10^7 USD if the baseline strategy is adopted, as marked by the circles in Figure 4.16. If the game has evolved to 240 min and the system is in state 1, the expected cumulative payoff that can be obtained after 1200 min (i.e., 1440-240) is -4.41×10^7 USD if the optimal strategy is adopted, and is -5.87×10^7 USD if the baseline strategy is adopted, as marked by the stars in Figure 4.16.

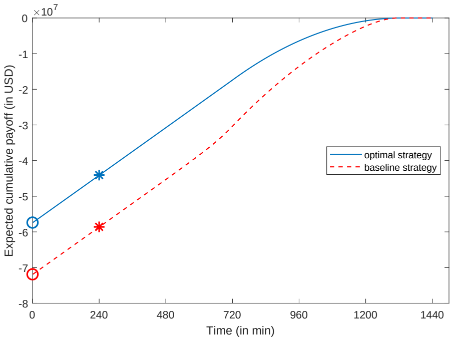


Figure 4.16: The expected cumulative payoffs of the defender with the optimal strategy and the baseline strategy.

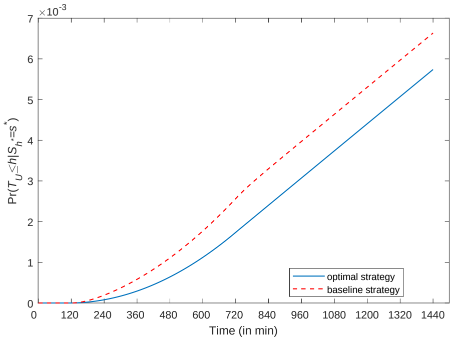


Figure 4.17: The comparison between $\Pr(T_U \leq h|S_{h^*} = s^*)$ with the equilibrium strategy and the baseline strategy.

The second measure in the comparison is the second risk metric, i.e., $\Pr(T_U \leq h|S_{h^*} = s^*)$, where $h^* = 0$ and s^* is state 1. The results for the optimal strategy and the baseline strategy are presented in Figure 4.17. It is easy to see that the equilibrium strategy obtained based on the proposed method reduces the probability of core damage in the finite horizon of 24 h from 0.0066 when the baseline strategy is used to 0.0057. The results illustrate the capability of the proposed method in reducing the impact and the risk and improving the resilience of nuclear power plants against malicious cyber attacks.

Part III

Counter-Deception Technologies

Chapter 5

Zero-Trust Defense against Advanced Persistent Threats

Following Section 1.3.2, Advanced Persistent Threats (APTs) are emerging security challenges for CPSs as the attacker can stealthily enter, persistently stay in, and strategically interact with the system. The multi-phase feature of APTs illustrated in Fig. 1.3 results in the concept of Defense in Depth (DiD), i.e., multi-stage cross-layer defense policies. A system defender should adopt defensive countermeasures across the phases of APTs and holistically consider interconnections and interdependencies among these layers. To formally describe the interaction between an APT attacker and a defender with the DiD strategy, we map the sequential phases of APTs into a game of multiple stages. Each stage describes a local interaction between the attacker and the defender where the outcome leads to the next stage of interactions. The goal of the attacker is to stealthily reach the targeted physical or informational assets, while the defender aims to take defensive actions at multiple phases to thwart the attack or reduce its impact.

Detecting APTs timely (i.e., before attackers have reached the final stage) and effectively (i.e., with a low rate of false alarms and missed detections) is still an open problem due to their stealthy and deceptive characteristics. Stuxnet-like APT attacks can conceal themselves in a ICS for years and inconspicuously increase the failure probability of physical components. Due to the insufficiency of timely and effective detection systems for APTs, the defender remains uncertain about the user's type, i.e., either legitimate or adversarial, throughout stages. In this work, we adopt a *zero-trust* framework [181], where the defender does not trust any user and adopts precautions and proactive defense measures for all users. By observing these users' behaviors, the defender updates his trust (or belief of the user's type) and revises the defense measures accordingly. Since these defense measures may impair the user experience and reduce the utility of a legitimate user. Therefore, the defender needs to strategically balance the tradeoff between security and usability when the user's type remains private.

5.1 Dynamic Game Modelling of APT Attacks

There are two players in the game, player 1 is the user and player 2 is the defender. The stealthy, persistent, and deceptive features of APTs result in incomplete information of the user's type to the defender. We use a finite set Θ_2 to accommodate all possible types of the user. For example, we consider a binary type set for the case study in Sections 5.4 and 5.5 where the user's type θ_2 is either adversarial θ_2^b or legitimate θ_2^g. The APT attacker, i.e., the adversarial user, disguises himself as the legitimate user, thus the defender does not know the type of the user. The set of the user's type can also be non-binary and incorporate different APT groups when their attack tools and targeted assets are different [51].

The defender can also be classified into different levels of sophistication based on various factors such as her level of security awareness, detection techniques she adopted, and the completeness of her virus signature database. The discrete type θ_1 distinguishes defenders of different sophistication levels and all the possible type values constitute the defender's type set Θ_1. For example, in our case study, the defender's type θ_1 is either sophisticated θ_1^H or primitive θ_1^L. The defender can apply defensive deception techniques and keep her type private to the user. We assume that both players' type sets are commonly known. Each player knows his/her own type, yet not the other player's type. Thus, each player i should treat the other player's type as a random variable with an initial distribution b_i^0 and update the distribution to b_i^k when obtaining new information at each stage k. We present the above belief update formally in Section 5.1.3.

5.1.1 Multi-Stage Transition

We formulate the interaction between the multi-stage APT attack and the cross-stage proactive defense into K stages of sequential games with incomplete information, as shown in Fig. 5.1. At each stage $k \in \{0, 1, \cdots, K\}$, player $i \in \{1, 2\}$ takes an action $a_i^k \in A_i^k$ from a finite and discrete set A_i^k. An Intrusion Detection System (IDS) generates alerts based on the user's actions. However, since legitimate users can also trigger these alerts, each alert itself does not reveal the user's type. For example, an APT attacker uses the Tor network connection for data exfiltration, yet a legitimate user can also use it legally for the traffic confidentiality as shown in [141]. Another example is that code obfuscation can be either used legitimately to

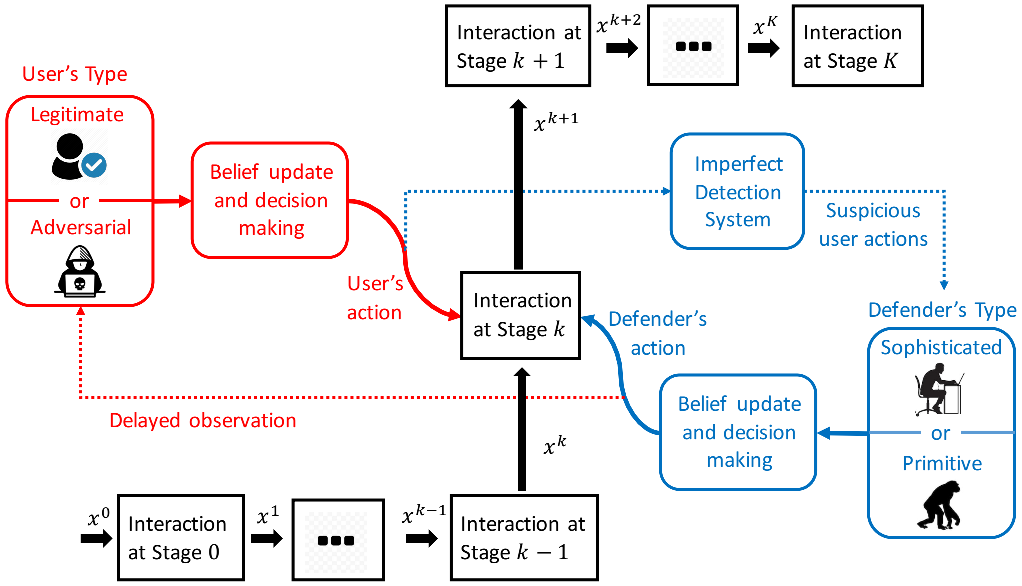


Figure 5.1: A block diagram of applying the defense-in-depth approach against multi-stage APT attacks. We denote the user, the defender, and the system states in red, blue, and black, respectively. The defender interacts with the user from stage 0 to stage K in sequence where the output state of stage $k-1$ becomes the input state of stage k. At each stage k, the user observes the defender's actions at previous stages, forms a belief on the defender's type, and takes an action. At the same time, the defender makes decisions based on the output of an imperfect detection system. The dotted line means that the observation is not in real time, i.e., both players can only observe the previous-stage actions of the other player.

prevent reverse engineering or illegally to conceal malicious JavaScript code from being recognized by signature-based detectors or human analysts as shown in [157]. We assume that the user can observe the defender's stage-k action at stage $k+1$. The observation of the defender's action at a single stage also does not reveal the defender's type.

In this paper, each player obtains a one-stage delayed observation of the other player's actions, i.e., at each stage k, the action history available to both players is $h^k = \{a_1^0, \cdots, a_1^{k-1}, a_2^0, \cdots, a_2^{k-1}\} \in H^k := \prod_{i=1}^{2} \prod_{\bar{k}=0}^{k-1} A_i^{\bar{k}}$. Given history h^k at the current stage k, players at stage $k+1$ obtain an updated history $h^{k+1} = h^k \cup \{a_1^k, a_2^k\}$ after the observation of both players' actions at stage k. At each stage k, we further define a state $x^k \in X^k$ which summarizes information about both players' actions in previous stages so that the initial state $x^0 \in X^0$ and the history at stage k uniquely determine x^k through a known state transition function f^k, i.e., $x^{k+1} = f^k(x^k, a_1^k, a_2^k), \forall k \in \{0, 1, \cdots, K-1\}$. States at different stages can have different meanings such as the reconnaissance outcome, the user's location, the privilege level, and the sensor status.

5.1.2 Behavioral Strategy

A defender should behave differently when interacting with adversarial users and legitimate ones. The defensive measure should also vary for attackers who adopt different code families and tools. However, since the defender is uncertain about the user's type throughout the entire stages of games, she has to make judicious decisions at each stage to balance usability versus security. The user's action should also adapt to the type of the defender. For example, if the defender is primitive, an attacker prefers to take aggressive adversarial actions to achieve a quicker and low-cost compromise. However, if the defender is sophisticated and can detect the malware with better accuracy, an attacker has to take conservative actions to remain stealthy. Since the proactive defense actions across the entire stages can affect legitimate users, they also need to be designed to avoid collateral damage.

Thus, the decision-making problem of the defender or the user boils down to the determination of a behavioral strategy $\sigma_i^k \in \Sigma_i^k : L_i^k \mapsto \Delta(A_i^k)$, i.e., player i at each stage k needs to decide which action to take or take an action with what probability based on the information $l_i^k \in L_i^k$ available to him/her at stage k. We present two different information structures in Sections 5.1.3 and 5.1.3. The strategy is called 'behavioral' as the strategy depends on the information available at the time the players make their decisions. In this work, players are allowed to take *mixed strategies*, thus the co-domain of the strategy function σ_i^k is $\Delta(A_i^k)$, a probability distribution over the action space A_i^k. With a slight abuse of notation, we denote $\sigma_i^k(a_i^k|l_i^k)$ as the probability of player i taking action $a_i^k \in A_i^k$ given the available information $l_i^k \in L_i^k$. The actual action of player i taken at stage k, i.e., a_i^k, is a realization of the behavioral strategy σ_i^k. Note that the values of the other player's type θ_j and action a_j^k are not observable for player i at stage k, thus do not affect player i's behavioral strategy σ_i^k, i.e., $\Pr(a_i^k|a_j^k, \theta_j, l_i^k) = \sigma_i^k(a_i^k|l_i^k)$. Therefore, σ_1^k and σ_2^k are conditionally independent, i.e., $\Pr(a_i^k, a_j^k|l_i^k, l_j^k) = \sigma_i^k(a_i^k|l_i^k)\sigma_j^k(a_j^k|l_j^k)$.

5.1.3 Belief and Bayesian Update

To quantify the uncertainty of the other player's type throughout the entire stages, each player i forms a belief $b_i^k : L_i^k \mapsto \Delta(\Theta_j), j \neq i$. Likewise, $b_i^k(\theta_j|l_i^k)$ means that given information $l_i^k \in L_i^k$ at stage k, player i forms a belief that the other player j is of type $\theta_j \in \Theta_j$ with probability $b_i^k(\theta_j|l_i^k)$. At the initial stage $k = 0$, the only information available to player i is his/her own type, i.e., $l_i^0 = \theta_i$. We assume that player i has a prior belief distribution b_i^0 based on the past experiences with the other player. If no previous experiences are available to player i, player i can take the uniform distribution as an unbiased prior belief. As each player i obtains new information when arriving at the next stage, his or her belief can be updated using the Bayesian rule. We present the Bayesian update under two different information structures L_i^k at stage $0 < k \leq K$ in the following two subsections.

Timely Observations

The most straightforward information structure is $L_i^k = H^k \times \Theta_i$, i.e., the information available to player i at stage k is the action history h^k and player i's own type θ_i, which leads to the belief update in (5.1), i.e., for all $i, j \in \{1, 2\}, j \neq i$,

$$b_i^{k+1}(\theta_j | h^k \cup \{a_i^k, a_j^k\}, \theta_i) = \frac{\sigma_i^k(a_i^k | h^k, \theta_i) \sigma_j^k(a_j^k | h^k, \theta_j) b_i^k(\theta_j | h^k, \theta_i)}{\sum_{\bar{\theta}_j \in \Theta_j} \sigma_i^k(a_i^k | h^k, \theta_i) \sigma_j^k(a_j^k | h^k, \bar{\theta}_j) b_i^k(\bar{\theta}_j | h^k, \theta_i)}. \quad (5.1)$$

Here, player i updates the belief b_i^k based on the observation of the action a_i^k, a_j^k. When the denominator is 0, the history h^{k+1} is not reachable from h^k, and the Bayesian update does not apply. In this case, we let $b_i^{k+1}(\theta_j | h^k \cup \{a_i^k, a_j^k\}, \theta_i) := b_i^0(\theta_j | \theta_i)$.

Markov Belief

If the information available to player i at stage k is the state value x^k and player i's own type θ_i, then the information set is taken to be $L_i^k = X^k \times \Theta_i$. With the Markov property that $\Pr(x^{k+1} | \theta_j, x^k, \cdots, x^1, x^0, \theta_i) = \Pr(x^{k+1} | \theta_j, x^k, \theta_i)$, the Bayesian update between two consequent states is

$$b_i^{k+1}(\theta_j | x^{k+1}, \theta_i) = \frac{\Pr(x^{k+1} | \theta_j, x^k, \theta_i) b_i^k(\theta_j | x^k, \theta_i)}{\sum_{\bar{\theta}_j \in \Theta_j} \Pr(x^{k+1} | \bar{\theta}_j, x^k, \theta_i) b_i^k(\bar{\theta}_j | x^k, \theta_i)}, i, j \in \{1, 2\}, j \neq i. \quad (5.2)$$

With the conditional independence of σ_1^k and σ_2^k,

$$\Pr(x^{k+1} | \theta_j, x^k, \theta_i) = \sum_{\{a_1^k, a_2^k\} \in \bar{A}^k} \sigma_1^k(a_1^k | x^k, \theta_1) \sigma_2^k(a_2^k | x^k, \theta_2), \quad (5.3)$$

where $\bar{A}^k := \{a_1^k \in A_1^k, a_2^k \in A_2^k | x^{k+1} = f^k(x^k, a_1^k, a_2^k)\}$ contains all the action pairs that change the system state from x^k to x^{k+1}. Equation (5.3) shows that the Bayesian update in (5.2) can be obtained from (5.1) by clustering all the action pairs in set $\bar{A}^k$. Thus, the Markov belief update (5.2) can also be regarded as an approximation of (5.1) using action aggregations. Unlike the history set H^k, the dimension of the state set, $|X^k|$, does not grow with the number of stages. Hence, the Markov approximation significantly reduces the memory and computational complexity. The following sections adopt the Markov belief update.

5.1.4 Stage and Cumulative Utility

The player's utility can vary under the same action taken by different types of users or defenders. For example, the remote access from a legitimate teleworker brings a reward to the defender while the one from an adversarial user inflicts a loss. Therefore, at each stage k, player i's stage utility $\bar{J}_i^k : X^k \times A_1^k \times A_2^k \times \Theta_1 \times \Theta_2 \times \mathbb{R} \mapsto \mathbb{R}$ can depend on both players' types and actions, the current state $x^k \in X^k$, and an external noise $w_i^k \in \mathbb{R}$ with a known probability density function ϖ_i^k. The noise term models unknown or uncontrolled factors that can affect the value of the stage utility. The existence of the external noise makes it impossible for player i, after reaching stage $k+1$, to infer the value of the other player's type θ_j based on the knowledge of the input parameters $x^k, a_1^k, a_2^k, \theta_i$, together with the output of the utility function $\bar{J}_i^k$ at stage k. We denote the expected stage utility as

$$J_i^k(x^k, a_1^k, a_2^k, \theta_1, \theta_2) := \mathbb{E}_{w_i^k \sim \varpi_i^k}[\bar{J}_i^k(x^k, a_1^k, a_2^k, \theta_1, \theta_2, w_i^k)], \forall x^k, a_1^k, a_2^k, \theta_1, \theta_2.$$

Given the type $\theta_i \in \Theta_i$, the initial state $x^{k_0} \in X^{k_0}$, and both players' strategies $\sigma_i^{k_0:K} := [\sigma_i^k(a_i^k|x^k, \theta_i)]_{k=k_0,\cdots,K} \in \prod_{k=k_0}^K \Sigma_i^k$ from stage k_0 to K, we can determine the expected cumulative utility $U_i^{k_0:K}$ for player i, i.e., for all $j \neq i$,

$$U_i^{k_0:K}(\sigma_i^{k_0:K}, \sigma_j^{k_0:K}, x^{k_0}, \theta_i) := \sum_{k=k_0}^{K} \mathbb{E}_{\theta_j \sim b_i^k, a_i^k \sim \sigma_i^k, a_j^k \sim \sigma_j^k}[J_i^k(x^k, a_1^k, a_2^k, \theta_1, \theta_2)]$$
$$= \sum_{k=k_0}^{K} \sum_{\theta_j \in \Theta_j} b_i^k(\theta_j | x^k, \theta_i) \sum_{a_i^k \in A_i^k} \sigma_i^k(a_i^k | x^k, \theta_i) \cdot \sum_{a_j^k \in A_j^k} \sigma_j^k(a_j^k | x^k, \theta_j) J_i^k(x^k, a_1^k, a_2^k, \theta_1, \theta_2). \tag{5.4}$$

5.2 PBNE and Dynamic Programming

The user and the defender use the Bayesian update to reduce their uncertainties on the other player's type. Since their actions affect the belief update, both players at each stage should optimize their expected cumulative utilities concerning the updated beliefs at the future stages, which leads to the Perfect Bayesian Nash Equilibrium (PBNE) in Definition 6.

Definition 6. *Consider the two-person K-stage game with double-sided incomplete information (i.e., each player's type is not known to the other player), a sequence of beliefs $b_i^k, \forall k \in \{0, \cdots, K\}$, an expected cumulative utility $U_i^{0:K}$ in (5.4), and a given scalar $\varepsilon \geq 0$. A sequence of strategies $\sigma_i^{*,0:K} \in \prod_{k=0}^{K} \Sigma_i^k$ is called ε-dynamic Bayesian Nash equilibrium for player i if condition (C2) is satisfied. If condition (C1) is also satisfied, $\sigma_i^{*,0:K}$ is further called ε-Perfect Bayesian Nash Equilibrium.*

(C1) Belief consistency: under strategy pair $(\sigma_1^{*,0:K}, \sigma_2^{*,0:K})$, each player's belief b_i^k at each stage $k = 0, \cdots, K$ satisfies (5.2).

(C2) Sequential rationality: for all given initial state $x^{k_0} \in X^{k_0}$ at every initial

stage $k_0 \in \{0, \cdots, K\}$,

$$U_1^{k_0:K}(\sigma_1^{*,k_0:K}, \sigma_2^{*,k_0:K}, x^{k_0}, \theta_1) + \varepsilon \geq U_1^{k:K}(\sigma_1^{k_0:K}, \sigma_2^{*,k_0:K}, x^{k_0}, \theta_1), \forall \sigma_1^{k_0:K} \in \prod_{k=0}^{K} \Sigma_1^k;$$
$$U_2^{k_0:K}(\sigma_1^{*,k_0:K}, \sigma_2^{*,k_0:K}, x^{k_0}, \theta_2) + \varepsilon \geq U_2^{k:K}(\sigma_1^{*,k_0:K}, \sigma_2^{k_0:K}, x^{k_0}, \theta_2), \forall \sigma_2^{k_0:K} \in \prod_{k=0}^{K} \Sigma_2^k. \tag{5.5}$$

When $\varepsilon = 0$*, the two* ε*-equilibria are called Dynamic Bayesian Nash Equilibrium (DBNE) and Perfect Bayesian Nash Equilibrium (PBNE), respectively.*

The belief consistency emphasizes that when strategic players make long-term decisions, they have to consider the impact of their actions on their opponent's beliefs at future stages. The PBNE is a refinement of the DBNE with the additional requirement of the belief consistency property. When the horizon $K = 0$, the multi-stage game of incomplete information defined in Section 5.1 degenerates to a one-stage (static) Bayesian game with the one-stage belief pairs (b_1^K, b_2^K) and the solution concept of the DBNE/PBNE degenerates to the Static Bayesian Nash Equilibrium (SBNE) in Definition 7.

The sequential rationality property in (5.5) guarantees that unilateral deviations from the equilibrium at any states do not benefit the deviating player. Thus, the equilibrium strategy can be a reasonable prediction of both players' multi-stage behaviors. DBNE strategies have the property of *strongly time consistency* because (5.5) holds for any possible initial states, even for states that are not on the equilibrium path, i.e., those states would not be visited under DBNE strategies. The *strongly time consistency* property makes the DBNE adapt to unexpected changes. Solutions obtained by dynamic programming naturally satisfy *strongly time consistency*. Hence, in the following, we introduce algorithms based on dynamic programming techniques.

Define the value function $V_i^{k_0}(x^{k_0}, \theta_i) := U_i^{k_0:K}(\sigma_1^{*,k_0:K}, \sigma_2^{*,k_0:K}, x^{k_0}, \theta_i)$ as the utility-to-go from any initial stage $k_0 \in \{0, \cdots, K\}$ under the DBNE strategy pair $(\sigma_1^{*,k_0:K}, \sigma_2^{*,k_0:K})$. Then, at the final stage K, the value function for player $i \in \{1, 2\}$ with type θ_i at state x^K is

$$V_i^K(x^K, \theta_i) = \sup_{\sigma_i^K \in \Sigma_i^K} \mathbb{E}_{\theta_j \sim b_i^K, a_i^K \sim \sigma_i^K, a_j^K \sim \sigma_j^{*,K}}[J_i^K(x^K, a_1^K, a_2^K, \theta_1, \theta_2)]. \tag{5.6}$$

For any feasible sequence of belief pairs $(b_1^k, b_2^k), k = 0, \cdots, K-1$, we have the following recursive system equations for player i to find the equilibrium strategy pairs $(\sigma_1^{*,k}, \sigma_2^{*,k})$ backwardly from stage $K-1$ to the initial stage 0, i.e., $\forall k \in \{0, \cdots, K-1\}, \forall i, j \in \{1, 2\}, j \neq i$,

$$V_i^k(x^k, \theta_i) = \sup_{\sigma_i^k \in \Sigma_i^k} \mathbb{E}_{\theta_j \sim b_i^k, a_i^k \sim \sigma_i^k, a_j^k \sim \sigma_j^{*,k}}[V_i^{k+1}(f^k(x^k, a_1^k, a_2^k), \theta_i) + J_i^k(x^k, a_1^k, a_2^k, \theta_1, \theta_2)]. \tag{5.7}$$

If we assume a virtual termination value $V_i^{K+1}(f^K(x^K, a_1^K, a_2^K), \theta_i) \equiv 0$, we can obtain (5.6) by letting stage $k = K$ in (5.7). The second term in (5.7) represents the immediate stage utility and the first term represents the expected utility under the future state $x^{k+1} = f^k(x^k, a_1^k, a_2^k), k \in \{0, \cdots, K-1\}$. Since a_i^k affects both terms, players should adopt a long-term perspective and avoid myopic behaviors to balance between the immediate utility and the expected future utility.

5.3 Computational Algorithms

In 5.3.1, we formulate a constrained optimization problem to compute the SBNE and V_i^K for the one-stage game. In 5.3.2, we use the proposed optimization problem as building blocks to compute the DBNE and $V_i^k, \forall k \in \{0, \cdots, K-1\}$. Finally, we propose an iterative algorithm to solve for the PBNE. Efficient algorithms to compute the PBNE lay a solid foundation to quantify the risk of cyber-physical attacks and guide the design of proactive DiD strategies.

5.3.1 One-Stage Bayesian Game and SBNE

Since both players' actions at the final stage $k = K$ only affect the immediate utility J_i^K and there is no future state transition, we can treat the final-stage game at each state $x^K \in X^K$ as an equivalent one-stage Bayesian game with the belief b_i^K and obtain the SBNE.

Definition 7. *A pair of mixed-strategies $(\sigma_1^{*,K} \in \Sigma_1^K, \sigma_2^{*,K} \in \Sigma_2^K)$ is said to constitute a Static Bayesian Nash Equilibrium (SBNE) under the given belief pair (b_1^K, b_2^K) and the state $x^K \in X^K$, if $\forall \theta_1 \in \Theta_1, \theta_2 \in \Theta_2$,*

$$
\begin{aligned}
&\mathbb{E}_{\theta_2 \sim b_1^K, a_1^K \sim \sigma_1^{*,K}, a_2^K \sim \sigma_2^{*,K}}[J_1^K(x^K, a_1^K, a_2^K, \theta_1, \theta_2)] \\
&\geq \mathbb{E}_{\theta_2 \sim b_1^K, a_1^K \sim \sigma_1^K, a_2^K \sim \sigma_2^{*,K}}[J_1^K(x^K, a_1^K, a_2^K, \theta_1, \theta_2)], \forall \sigma_1^K \in \Sigma_1^K; \\
&\mathbb{E}_{\theta_1 \sim b_2^K, a_1^K \sim \sigma_1^{*,K}, a_2^K \sim \sigma_2^{*,K}}[J_2^K(x^K, a_1^K, a_2^K, \theta_1, \theta_2)] \\
&\geq \mathbb{E}_{\theta_1 \sim b_2^K, a_1^K \sim \sigma_1^{*,K}, a_2^K \sim \sigma_2^K}[J_2^K(x^K, a_1^K, a_2^K, \theta_1, \theta_2)], \forall \sigma_2^K \in \Sigma_2^K.
\end{aligned}
\tag{5.8}
$$

In Theorem 3, we propose a constrained optimization program C^K to compute the SBNE. We suppress the superscript of K without any ambiguity in one-stage games.

Theorem 3. *A strategy pair $(\sigma_1^* \in \Sigma_1, \sigma_2^* \in \Sigma_2)$ constitutes a SBNE to the one-stage bi-matrix Bayesian game (J_1, J_2) under private type $\theta_i \in \Theta_i, \forall i \in \{1,2\}$, belief $b_i, \forall i \in \{1,2\}$, and a given state x, if and only if the strategy pair is a solution to C^K:*

$$\begin{aligned}
[C^K] : \max_{\sigma_1,\sigma_2,s_1,s_2} & \sum_{\theta_1\in\Theta_1} \alpha_1(\theta_1)s_1(x,\theta_1) + \sum_{\theta_2\in\Theta_2} \alpha_2(\theta_2)s_2(x,\theta_2) \\
& + \sum_{\theta_1\in\Theta_1} \alpha_1(\theta_1)\mathbb{E}_{\theta_2\sim b_1, a_1\sim\sigma_1, a_2\sim\sigma_2}[J_1(x,a_1,a_2,\theta_1,\theta_2)] \\
& + \sum_{\theta_2\in\Theta_2} \alpha_2(\theta_2)\mathbb{E}_{\theta_1\sim b_2, a_1\sim\sigma_1, a_2\sim\sigma_2}[J_2(x,a_1,a_2,\theta_1,\theta_2)] \\
s.t. \qquad & (a) \quad \mathbb{E}_{\theta_1\sim b_2, a_1\sim\sigma_1}[J_2(x,a_1,a_2,\theta_1,\theta_2)] \le -s_2(x,\theta_2), \forall\theta_2, \forall a_2, \\
& (b) \quad \sum_{a_1\in A_1} \sigma_1(a_1|x,\theta_1) = 1, \sigma_1(a_1|x,\theta_1) \ge 0, \forall\theta_1, \\
& (c) \quad \mathbb{E}_{\theta_2\sim b_1, a_2\sim\sigma_2}[J_1(x,a_1,a_2,\theta_1,\theta_2)] \le -s_1(x,\theta_1), \ \forall\theta_1, \forall a_1, \\
& (d) \quad \sum_{a_2\in A_2} \sigma_2(a_2|x,\theta_2) = 1, \sigma_2(a_2|x,\theta_2) \ge 0, \forall\theta_2.
\end{aligned}$$

The dimensions of decision variables $\sigma_1(a_1|x,\theta_1), \forall\theta_1 \in \Theta_1$, and $\sigma_2(a_2|x,\theta_2), \forall\theta_2 \in \Theta_2$, are $|A_1| \times |\Theta_1|$ and $|A_2| \times |\Theta_2|$, respectively. Besides, $s_1(x,\theta_1), \forall\theta_1 \in \Theta_1$, and $s_2(x,\theta_2), \forall\theta_2 \in \Theta_2$, are scalar decision variables for each given $\theta_i, i \in \{1,2\}$. The non-decision variables $\alpha_1(\theta_1), \forall\theta_1$ and $\alpha_2(\theta_2), \forall\theta_2$, can be any strictly positive and finite numbers. The solution to C^K exists and is achieved at the equality of constraints $(a), (c)$, i.e., $s_2^*(x,\theta_2) = -V_2(x,\theta_2), s_1^*(x,\theta_1) = -V_1(x,\theta_1)$.

Proof. The finiteness and discreteness of the action and the type spaces guarantee the existence of the SBNE in mixed strategies as shown in [196], which further guarantee that program C^K has solutions. To show the equivalence between the solution to C^K and the SBNE, we first show that every SBNE is a solution of C^K. If $(\sigma_1^* \in \Sigma_1, \sigma_2^* \in \Sigma_2)$ is a SBNE pair, then the quadruple $\sigma_1^*(\theta_1), \sigma_2^*(\theta_2), s_2^*(x, \theta_2) = -V_2(x, \theta_2), s_1^*(x, \theta_1) = -V_1(x, \theta_1), \forall \theta_i \in \Theta_i, \forall i \in \{1, 2\}$, is feasible because it satisfies constraints $(a), (b), (c), (d)$. Constraints (a) and (c) imply a non-positive objective function of C^K. Since the value of the objective function achieved under this quadruple is 0, this quadruple is also optimal. Second, we show that $\sigma_1^*(\theta_1), \sigma_2^*(\theta_2), s_2^*(x, \theta_2), s_1^*(x, \theta_1)$, the result of C^K is a SBNE. The solution of C^K should satisfy all the constraints, i.e.,

$$\begin{aligned} &\mathbb{E}_{\theta_1 \sim b_2, a_1 \sim \sigma_1^*, a_2 \sim \sigma_2}[J_2(x, a_1, a_2, \theta_1, \theta_2)] \leq -s_2^*(x, \theta_2), \forall \theta_2, \forall \sigma_2 \in \Sigma_2, \\ &\mathbb{E}_{\theta_2 \sim b_1, a_1 \sim \sigma_1, a_2 \sim \sigma_2^*}[J_2(x, a_1, a_2, \theta_1, \theta_2)] \leq -s_1^*(x, \theta_1), \forall \theta_1, \forall \sigma_1 \in \Sigma_1. \end{aligned} \tag{5.9}$$

In particular, if we pick $\sigma_i(\theta_i) = \sigma_i^*(\theta_i), \forall \theta_i, \forall i \in \{1, 2\}$, and combine the fact that the optimal value is achieved at 0, the inequality turns out to be an equality and equation (5.9) becomes (5.8), which shows that $(\sigma_1^* \in \Sigma_1, \sigma_2^* \in \Sigma_2)$ is a SBNE. □

Theorem 3 focuses on the double-sided Bayesian game where each player player i has a private type $\theta_i \in \Theta_i$. To accommodate the one-sided Bayesian game where player i's type $\theta_i \in \Theta_i$ is known by both players and player j's type remains unknown to player i, we can modify program C^K by letting $\alpha_i(\theta_i) > 0$ and $\alpha_i(\tilde{\theta}_i) = 0, \forall \tilde{\theta}_i \in \Theta_i \setminus \{\theta_i\}$.

5.3.2 Multi-Stage Bayesian Game and PBNE

From (5.7), we can see that at stages $k < K$, each player optimizes the sum of the immediate utility J_i^k and the utility-to-go V_i^k. Thus, we can replace the original stage utility J_i^K in program C^K with $V_i^k + J_i^k$ in program C^k to compute the DBNE in a multi-stage Bayesian game.

Theorem 4. *Given a sequence of beliefs b_i^k for each player $i \in \{1,2\}$ at each stage $k \in \{0,1,\cdots,K-1\}$, a strategy pair $(\sigma_1^{*,0:K-1}, \sigma_2^{*,0:K-1})$ constitutes a DBNE of the K-stage Bayesian game under double-sided incomplete information with the expected cumulative utility $U_i^{0:K}$ in (5.4), if and only if $\sigma_1^{*,k}, \sigma_2^{*,k}$, $s_1^{*,k}(x^k,\theta_1)$, and $s_2^{*,k}(x^k,\theta_2)$ are the optimal solutions to the following constrained optimization problem C^k for each $k \in \{0,1,\cdots,K-1\}$:*

$$
\begin{aligned}
[C^k] : \max_{\sigma_1^k,\sigma_2^k,s_1^k,s_2^k} \quad & \sum_{i=1}^{2} \sum_{\theta_i \in \Theta_i} \alpha_i(\theta_i)\{s_i^k(x^k,\theta_i) + \sum_{\theta_j \in \Theta_j} b_i^k(\theta_j|x^k,\theta_i) \sum_{a_1^k \in A_1^k} \sigma_1^k(a_1^k|x^k,\theta_1) \\
& \cdot \sum_{a_2^k \in A_2^k} \sigma_2^k(a_2^k|x^k,\theta_2)[J_i^k(x^k,a_1^k,a_2^k,\theta_1,\theta_2) + V_i^{k+1}(f^k(x^k,a_1^k,a_2^k),\theta_i)]\} \\
s.t. \ (a) \quad & \sum_{\theta_1 \in \Theta_1} b_2^k(\theta_1|x^k,\theta_2) \sum_{a_1^k \in A_1^k} \sigma_1^k(a_1^k|x^k,\theta_1) \cdot [J_2^k(x^k,a_1^k,a_2^k,\theta_1,\theta_2) \\
& + V_2^{k+1}(f^k(x^k,a_1^k,a_2^k),\theta_2)] \leq -s_2^k(x^k,\theta_2), \forall \theta_2 \in \Theta_2, \forall a_2^k \in A_2^k, \\
(b) \quad & \sum_{\theta_2 \in \Theta_2} b_1^k(\theta_2|x^k,\theta_1) \sum_{a_2^k \in A_2^k} \sigma_2^k(a_2^k|x^k,\theta_2) \cdot [J_1^k(x^k,a_1^k,a_2^k,\theta_1,\theta_2) \\
& + V_1^{k+1}(f^k(x^k,a_1^k,a_2^k),\theta_1)] \leq -s_1^k(x^k,\theta_1), \forall \theta_1 \in \Theta_1, \forall a_1^k \in A_1^k.
\end{aligned}
$$

Similarly, $\alpha_1(\theta_1), \alpha_2(\theta_2)$ can be any strictly positive and finite numbers, and $(s_1^k(x^k,\theta_1), s_2^k(x^k,\theta_2))$ is a sequence of scalar variables for each $x^k \in X^k, \theta_i \in \Theta_i, i \in \{1,2\}$. The optimum exists and is achieved at the equality of constraints $(a),(b)$, i.e., $s_i^{*,k}(x^k,\theta_i) = -V_i^k(x^k,\theta_i), \forall \theta_i \in \Theta_i, \forall i \in \{1,2\}$.

The proof is similar to the one for Theorem 3. The decision variables σ_i^k are of size $|A_i^k| \times |X^k| \times |\Theta_i|$. By letting stage $k = K$ and $V_i^{K+1} = 0$, program C^K for the static Bayesian game is a special case of C^k for the multi-stage Bayesian game.

We can solve program C^{k+1} to obtain the DBNE strategy pair $(\sigma_1^{k+1}, \sigma_2^{k+1})$ and the value of V_i^{k+1}. Then, we apply V_i^{k+1} in program C^k to obtain a DBNE strategy pair (σ_1^k, σ_2^k) and the value of V_i^k. Thus, for any given sequences of type belief pairs $b_i^k, \forall i \in \{1, 2\}, \forall k \in \{0, 1, \cdots, K\}$, we can solve C^k from $k = K$ to $k = 0$ recursively to obtain the DBNE pair $(\sigma_1^{*,0:K-1}, \sigma_2^{*,0:K-1})$.

Given a sequence of beliefs, we can obtain the corresponding DBNE via C^k in a backward fashion. However, given a sequence of policies, both players forwardly update their beliefs at each stage by (5.2). Thus, we need to find a consistent pair of belief and policy sequences as required by the PBNE. As summarized in Algorithm 4, we iteratively alternate between the forward belief update and the backward policy computation to find the PBNE. We resort to ε-PBNE solutions when the existence of PBNE is not guaranteed.

Algorithm 4 provides a computational approach to find ε-PBNE with the following procedure. First, both players initialize their beliefs b_i^k for every state x^k at stage $k \in \{0, 1, \cdots, K\}$, according to their types. Then, they compute the DBNE strategy pair $\sigma_i^{*,0:K}, \forall i \in \{1, 2\}$, under the given belief sequence at each stage by solving program C^k from stage K to stage 0 in sequence. Next, they update their beliefs at each stage according to the strategy pair $\sigma_i^{*,0:K-1}, \forall i \in \{1, 2\}$, via the Bayesian update (5.2). If the strategy pair $\sigma_i^{*,0:K-1}, \forall i \in \{1, 2\}$, satisfies (5.5) under the updated belief, we find the ε-PBNE and terminate the iteration. Otherwise, we repeat the backward policy computation in step two and the forward belief update in step three.

Algorithm 4: Numerical Solution of ε-PBNE

```
Initialization beliefs $b_i^k$ at each stage $k \in \{0, 1, \cdots, K\}$, IterNum$> 0$, $\varepsilon \geq 0$.
while the $t <$IterNum do
    $t := t + 1$;
    for each $x^K \in X^K$ do
        Compute SBNE strategy $\sigma_i^{*,K}$ and $V_i^K(x^K, \theta_i)$ via $C^K$.
    end
    for $k \leftarrow K - 1$ to $0$ do
        for each $x^k \in X^k$ do
            Compute DBNE strategy $\sigma_i^{*,k}$ and $V_i^k(x^k, \theta_i)$ via $C^k$.
        end
    end
    for $k \leftarrow 0$ to $K - 1$ do
        Update $b_i^k$ with $\sigma_i^{*,0:K-1}$ via (5.2).
    end
    if $\sigma_i^{*,0:K-1}, \forall i \in \{1, 2\}$, satisfy (5.5) then
        Terminate
end
Output $\varepsilon$-PBNE strategy pair $(\sigma_1^{*,0:K-1}, \sigma_2^{*,0:K-1})$ and consistent beliefs
  $b_i^k, \forall k \in \{0, \cdots, K\}$.
```

5.4 Case Study

The model presented in Section 5.1 can be applied to various APT scenarios. To illustrate the framework, this section presents a specific attack scenario where the attacker stealthily initiates infection and escalates privileges in the cyber network, aiming to launch attacks on the physical plant as shown in Fig. 5.2. Three vertical columns in the left block illustrate the state transitions across three stages: the initial compromise, the privilege escalation, and the sensor compromise of a physical system. The red squares at each column represent possible states at that stage. The right block illustrates a simplified flow chart of the Tennessee Eastman Process. We use the Tennessee Eastman process as a benchmark of industrial control systems to show that attackers can strategically compromise the SCADA system and decrease the operational efficiency of a physical plant without triggering the alarm.

In this case study, we adopt the binary type space $\Theta_2 = \{\theta_2^b, \theta_2^g\}$ and $\Theta_1 = \{\theta_1^H, \theta_1^L\}$ for the user and the defender, respectively. In particular, θ_2^b and θ_2^g denote the adversarial and legitimate user, respectively; θ_1^H and θ_1^L denote the sophisticated and primitive defender, respectively. The bi-matrices in Table 5.1, 5.2, and 5.3 represent both players' expected utilities at three stages, respectively. In these matrices, the defender is the *row player* and the user is the *column player*. Each entry of the matrix corresponds to players' payoffs under their action pairs, types, and the state. In particular, the two red numbers in the parenthesis before the semicolon are the payoffs of the defender and the user, respectively, under type θ_2^b, while the parenthesis in blue after the semicolon presents the payoff of the defender and the user, respectively, under type θ_2^g.

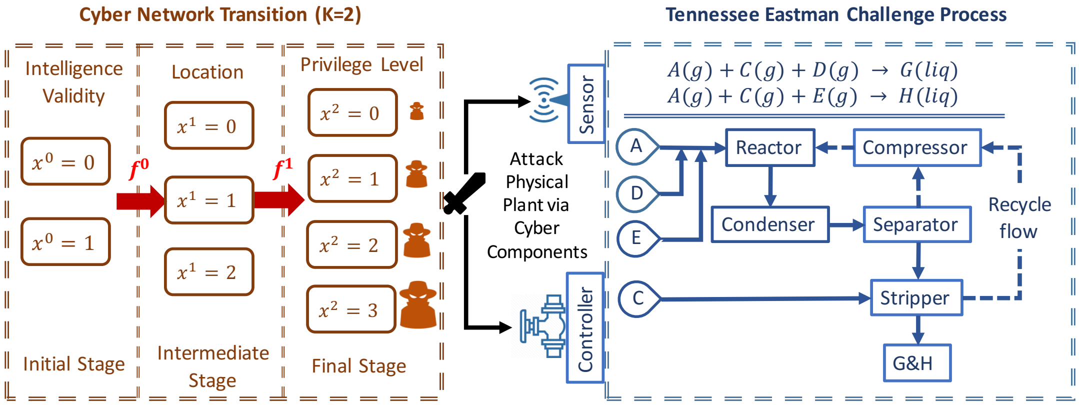


Figure 5.2: The diagram of the cyber state transition (denoted by the left block in orange) and the physical attack on Tennessee Eastman process via the compromise of the SCADA system (denoted by the right block in blue). APTs can damage the normal industrial operation by falsifying controllers' setpoints, tampering sensor readings, and blocking communication channels to cause delays in either the control message or the sensing data.

5.4.1 Initial Stage: Phishing Emails

We use a binary set to represent whether the reconnaissance is effectual $x^0 = 1$ or not $x^0 = 0$. Effectual reconnaissance collects essential intelligence that can better support APTs for an initial entry through phishing emails. To penalize the adversarial exploitation of the open-source intelligence (OSINT) data, the defender can create avatars (fake personal profiles) on the social network or the company website as shown in [145].

At the initial stage of interaction, a user can send emails with non-executable attachments and shortened URLs to the accounts of entry-level employees, managers, or avatars. These three action options of the user are represented by $a_2^0 = 0, 1, 2$, respectively. Non-executable files such as PDF and MS Office are widely used in organizations yet an APT attacker can exploit them to execute malicious actions on the victim's computer. The shortened URL is created by legitimate service providers such as *Google URL shortener* yet can redirect to malicious links. The existing email security mechanisms are not completely effective for identifying malicious PDF files (see [157]) and malicious links behind shortened URLs (see [185]). As a supplement to technical countermeasures, security training should be emphasized to increase employees' security awareness and protect them from web phishing. For example, after receiving suspicious links or attachments with strange names at unexpected times, the entry-level employee and the manager should be aware of the potential risk and apply extra security measures such as a digital signature request from the sender before clicking the link or opening the attachment. They should also be sufficiently alert and report immediately if a PDF does not contain the information that it claims to have. Then isolation can be applied to prevent the attacker from the potential lateral movement. Since employees' awareness and alertness diminish over time, the security training needs to be repeated at reasonable intervals as argued in [143], which can be costly. With a limited budget, the defender can choose to educate entry-level employees, manager-level employees, or no training to avoid the prohibitive training cost c^0. These three action options of the defender are represented by $a_1^0 = 1, 2, 0$, respectively. The utility matrix of the initial infection is given in Table 5.1. If the user is legitimate, i.e., $\theta_2 = \theta_2^g$, then as denoted in the blue color, he receives an immediate reward r_1^0 if he successfully communicates with the employee or the manager by email, but receives a substantial penalty $r_{g,f}^0 < 0$ if he emails the avatars because he should not contact a non-existing person. If the user is adversarial, i.e., $\theta_2 = \theta_2^b$, then as denoted in the red color, he receives an immediate attack reward r_2^0 if the email receiver does not have proper security training, but an additional attack cost r^0 if the receiver has been trained properly. The adversarial user receives a faked reward $r_{b,f}^0 > 0$ when contacting the avatar, yet arrives at an unfavorable state at stage $k = 1$ and receives few rewards in the future stages. The training cost and the attack cost are both different for the primitive and the sophisticated defender, i.e., $c^0 := c_L^0 \cdot \mathbf{1}_{\{\theta_1 = \theta_1^L\}} + c_H^0 \cdot \mathbf{1}_{\{\theta_1 = \theta_1^H\}}$ and $r^0 := r_L^0 \cdot \mathbf{1}_{\{\theta_1 = \theta_1^L\}} + r_H^0 \cdot \mathbf{1}_{\{\theta_1 = \theta_1^H\}}$. The sophisticated defender holds the security training with a higher frequency, which incurs a higher cost, i.e., $c_H^0 > c_L^0$, but is also more effective in mitigating web phishing, i.e., $r_H^0 > r_L^0$.

5.4.2 Intermediate Stage: Privilege Escalation

The state at the intermediate stage can be interpreted as the location of the user where $x^1 = 1$ refers to the employee's computer, $x^1 = 2$ refers to the manager's computer, and $x^1 = 0$ refers to the quarantine area. After the initial access, the user operates within a process of low privilege. To access certain resources, the user

Table 5.1: The expected utilities of the defender and the user at the initial stage, i.e., J_1^0 and J_2^0, respectively.

θ_2^b;θ_2^g	**Email Employees**	**Email Managers**	**Email Avatars**
No Training	$(-r_2^0, r_2^0)$;$(0, r_1^0)$	$(-r_2^0, r_2^0)$;$(0, r_1^0)$	$(0, r_{b,f}^0)$;$(0, r_{g,f}^0)$
Train Employees	$(-c^0, -r^0)$;$(-c^0, r_1^0)$	$(-c^0, r_2^0)$;$(-c^0, r_1^0)$	$(-c^0, r_{b,f}^0)$;$(-c^0, r_{g,f}^0)$
Train Managers	$(-c^0, r_2^0)$;$(-c^0, r_1^0)$	$(-c^0, -r^0)$;$(-c^0, r_1^0)$	$(-c^0, r_{b,f}^0)$;$(-c^0, r_{g,f}^0)$

needs to gain higher-level privileges. An attacker can utilize the process injection to execute malicious code in the address space of a live process and masquerade as legitimate programs to evade detection as shown in [210]. A mitigation method for the defender is to prevent certain endpoint behaviors that can occur during the process injection. Table 5.2 presents this game of privilege escalation.

The user can choose to escalate his privileges, or choose '*no operation performed (NOP)*'. The two action options are denoted by $a_2^1 = 1$ and $a_2^1 = 0$, respectively. The defender can choose to either restrict or permit an escalation, which are denoted by $a_1^1 = 1$ and $a_1^1 = 0$, respectively. If the legitimate user escalates his privilege and the defender permits escalation, then both players obtain a reward of r_1^1. If the legitimate user escalates his privilege and the defender restricts escalation, then the efficiency reduction brings a loss of r_1^1 to both players. On the other hand, if the adversarial user escalates his privilege and the defender permits escalation, the defender receives a loss of r_2^1. If the adversarial user escalates his privilege and the defender restricts escalation, then the adversarial user has to resort to other attack techniques which lead to a higher rate of detection. Thus, the defender obtains a reward while the attacker receives an additional cost. We assume that

Table 5.2: The expected utilities of the defender and the user at the intermediate stage, i.e., J_1^1 and J_2^1, respectively.

θ_2^b;θ_2^g	**NOP**	**Escalate Privilege**
Permit Escalation	$(0,0)$;$(0,0)$	$(-r_2^1, r_2^1)$;(r_1^1, r_1^1)
Restrict Escalation	$(0,0)$;$(0,0)$	$(r^1, -r^1)$;$(-r_1^1, -r_1^1)$

the reward and the additional cost are both r_L^1 if the defender is primitive, and r_H^1 if the defender is sophisticated, i.e., $r^1 = r_L^1 \cdot \mathbf{1}_{\{\theta_1 = \theta_1^L\}} + r_H^1 \cdot \mathbf{1}_{\{\theta_1 = \theta_1^H\}}$.

5.4.3 Final Stage: Sensor Compromise

The state at the final stage represents four possible privilege levels, denoted by $x^2 = \{0, 1, 2, 3\}$, respectively. The privilege level affects the result of the physical attack at the final stage. The defender's and the user's actions, and the state at the intermediate stage determine the state at the final stage. For example, if the user is at the quarantine area during the intermediate stage, then he ends up with a level-zero privilege regardless of actions taken by the defender and himself. Users who take control of the manager's computer at the intermediate stage can obtain a higher privilege level than those who start from the entry-level employee's computer, yet the degree of escalation is reduced if the defender chooses to restrict escalation.

We modify the Simulink model in [16] to quantify the monetary loss of the Tennessee Eastman process under sensor compromises. Our attack model of sensor compromise is presented in Section 5.4.3. A new performance metric to quantify the operational efficiency of the Tennessee Eastman process is proposed in Section 5.4.3 and applied in the game matrix in Section 5.4.3.

Performance Metric

The Tennessee Eastman process involves two irreversible reactions to produce two liquid (liq) products G, H from four gaseous (g) reactants A, C, D, E, as shown in the right block of Fig. 5.2. The control objective is to maintain a desired production rate as well as quality while stabilizing the whole system under the Gaussian noise to avoid violating safety constraints such as a high reactor pressure, a high reactor temperature, and a high/low separator/stripper liquid level. Previous studies on the security of the Tennessee Eastman process have mostly focused on how an attacker can cause the shortest shutdown time (see [120]), or a serious violation of a setpoint, e.g., the reactor pressure exceeds $3,000$ kpa (see [22]). These attacks successfully cause the shutdown of the plant and a few days of shutdowns can incur a considerable financial loss. However, the shutdown also discloses the attack and leads to an immediate patch and a defense strategy update. Thus, it becomes harder for the same kind of attacks to succeed after the plant recovers from the shutdown.

In our APT scenario, the attacker aims to stealthily decrease the operational efficiency of the plant, i.e., deviate the normal operation state of the plant without triggering the safety alarm or shutting down the plant. By compromising the SCADA system and generating fraudulent sensor readings, the attacker can stealthily make the plant operates at a non-optimal state with reduced utilities. The following economic metrics affect the operational utility of the Tennessee Eastman process:

- Hourly operating cost C_o with the unit $(\$/h)$ is taken as the sum of purge costs, product stream costs, compressor costs, and stripper steam costs.

- Production rate R_p with the unit (m^3/h) is the volume of total products per hour.

- Quality of products Q_p with the unit $(G\ mole\%)$, is the percentage of G among total products.

- P_G with the unit $(\$/m^3)$ is the price of product G.

We propose a new performance metric U_{TE}, the *per-hour utility* to quantify the operational efficiency of the Tennessee Eastman process as follows:

$$U_{TE} = R_p \times Q_p \times P_G - C_o. \tag{5.10}$$

Attack Model

An attack model is characterized by two separate parts, *information* and *capacity*. First, the information available to the attacker such as readings of different sensors can affect the performance of the attack differently. For example, observing the input rate of the raw material in the Tennessee Eastman process is less beneficial for the attacker than the direct measurements of P_G, R_p, Q_p, C_o that affect the utility metric in (5.10). Second, attackers can have different capacities in accessing and revising controllers and sensors. An attacker may change the parameters of the proportional-integral-derivative controller, directly falsify the controller output, or indirectly deviate the setpoint by tampering, blocking or delaying sensor readings.

In this experiment, we assume a reading manipulation of sensor XMEAS(40) and XMEAS(17) in loop 8 and loop 13 of Tennessee Eastman process (see [177]), respectively. Sensor XMEAS(40) measures the composition of component G and sensor XMEAS(17) measures the stripper underflow. A higher privilege state

$x^2 \in \{0,1,2,3\}$ means that the user can access more sensors for a longer time, which results in a larger loss and thus a smaller utility of $r_1^2(x^2)$ to the defender if the user is adversarial. Fig. 5.3 shows the variation of U_{TE} versus the simulation time under four different privilege states. We use the time average of these utilities to obtain the normal operational utility r_4^2 and compromised utilities $r_1^2(x^2)$ under four different privilege states $x^2 \in \{0,1,2,3\}$. The attacker compromises the sensor

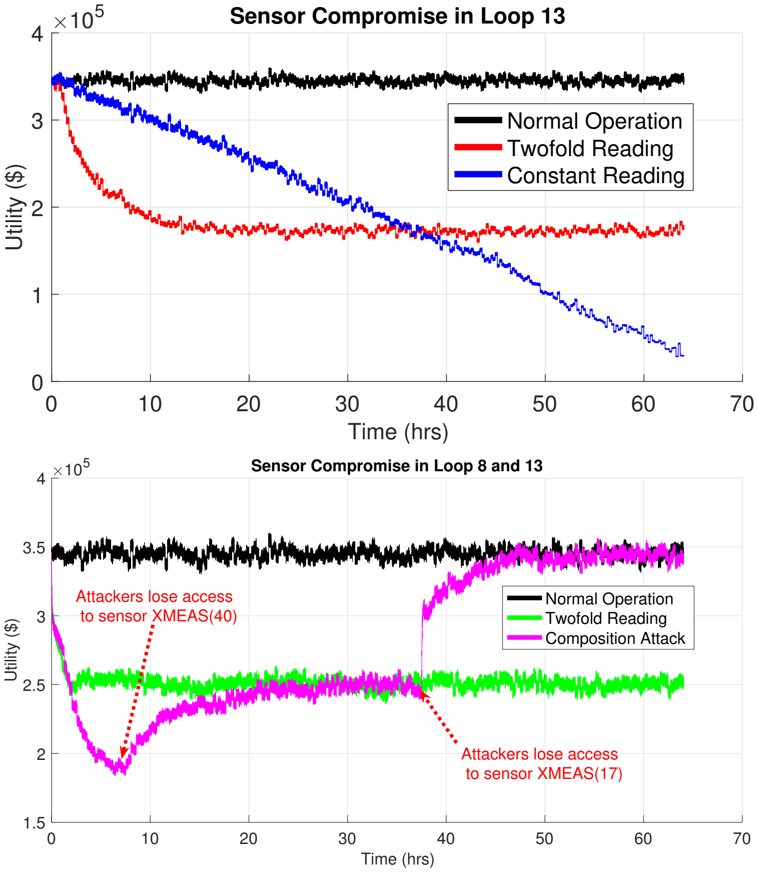


Figure 5.3: The economic impact of sensor compromise in the Tennessee Eastman process. The black line represents the utility of Tennessee Eastman process under the normal operation while the other four lines represent the utility of Tennessee Eastman process under attacks with four possible privilege levels. We use the time average of these utilities to obtain the normal operational utility r_4^2 and compromised utilities $r_1^2(x^2), \forall x^2 \in \{0,1,2,3\}$, under four different states of privilege levels in Table 5.3.

and generates fraudulent readings. The fraudulent reading can be a constant, denoted by the blue line, or a double of the real readings, denoted by the red or green lines. The pink line represents a composition attack with a limited control time. Initially, the attacker manages to compromise both sensors by doubling their readings. After the attacker loses access to XMEAS(40) at the 6^{th} hour, the system is sufficiently resilient to recover partially in about 16 hours and achieve the same level of utility as the single attack in green. When the attacker also loses access to XMEAS(17) at the 36^{th} hour, the utility goes back to normal in about 13 hours.

Utility Matrix

Attacks against SCADA system can apply command injection attacks to inject false control and compromise sensor readings as shown in [149]. Encryption can be introduced to conceal these malicious commands. However, a legitimate user may also encrypt his communication with the sensor to avoid eavesdropping and enhance privacy.

Therefore, at the final stage, the user has two options, sends commands to the sensor with or without encryption, which are denoted by $a_2^2 = 1$ and $a_2^2 = 0$, respectively. The defender chooses to apply either a complete or selective monitoring, denoted by $a_1^2 = 1$ and $a_1^2 = 0$, respectively. The complete monitoring stores all sets of communication data and analyzes them elaborately to identify malicious commands despite encryption. The selective monitoring cannot identify malicious commands if they are encrypted. The implementation of the complete monitoring incurs an additional cost c^2 compared to the selective one. The last-stage utility matrix of both players is defined in Table 5.3. If the user is legitimate, as denoted in blue, both the defender and the user can receive a reward of r^4 when the

Table 5.3: The expected utilities of the defender and the user at the final stage, i.e., J_1^2 and J_2^2, respectively.

θ_2^b;θ_2^g	**Unencrypted Command (UC)**	**Encrypted Command (EC)**
Selective Monitoring (SM)	$(r_4^2, 0)$;$(r_4^2, r_4^2/2)$	$(r_1^2(x^2), r_4^2 - r_1^2(x^2))$;$(r_4^2, r_4^2)$
Complete Monitoring (CM)	$(r_4^2 - c^2, 0)$;$(r_4^2 - c^2, r_4^2/2)$	$(r^2 - c^2, -r^2)$;$(r_4^2 - c^2, r_4^2)$

Tennessee Eastman process operates normally. Legitimate users further receive a utility reduction of $r^4/2$ for the potential privacy loss if they choose unencrypted commands. For adversarial users, they send malicious commands only when the communication is encrypted to evade detection. Thus, if they choose not to encrypt the communication, they receive 0 utility and the defender receives a reward of r^4 for the normal operation. However, if they choose to send encrypted malicious commands, both players' rewards depend on whether the defender chooses the selective or complete monitoring. If the defender chooses the selective monitoring, then the adversarial user can successfully compromise the sensor, which results in a reduced utility of $r_1^2(x^2)$. In the meantime, the attacker benefits from the reward reduction of $r_4^2 - r_1^2(x^2)$. If the defender chooses the complete monitoring, then the adversarial user suffers a loss of r^2 for being detected. The detection reward and the implementation cost for two types of defenders are r_L^2, r_H^2 and c_L^2, c_H^2, respectively. Let $r^2 := r_L^2 \cdot \mathbf{1}_{\{\theta_1 = \theta_1^L\}} + r_H^2 \cdot \mathbf{1}_{\{\theta_1 = \theta_1^H\}}$ and $c^2 := c_L^2 \cdot \mathbf{1}_{\{\theta_1 = \theta_1^L\}} + c_H^2 \cdot \mathbf{1}_{\{\theta_1 = \theta_1^H\}}$.

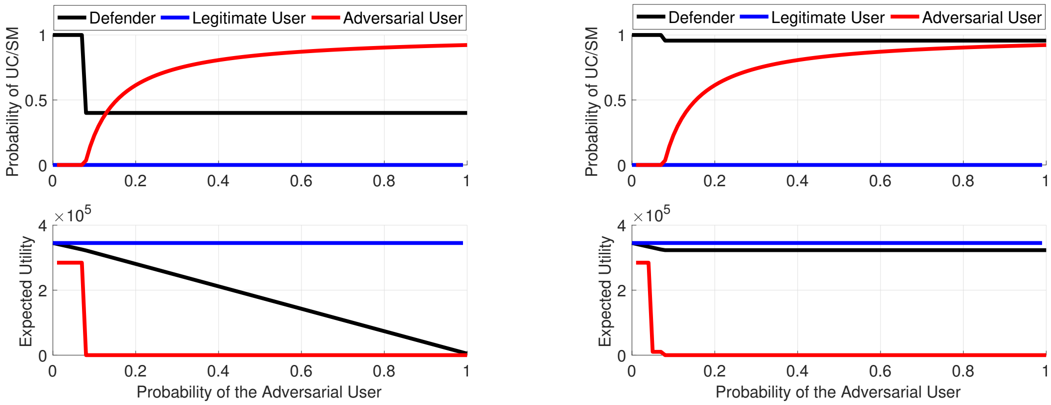
5.5 Computation Results

In this section, we apply the algorithms introduced in Section 5.3 to compute both players' strategies and utilities at the equilibrium. We implement our algorithms in MATLAB and use YALMIP (see [132]) as the interface to call external solvers such as BARON (see [207]) to solve the optimization problems. We present elaborate results from the concrete case study and provide meaningful insights of the proactive cross-layer defense against multi-stage APT attacks that are stealthy and deceptive.

For the static Bayesian game at the final stage in Section 5.5.1, we focus on illustrating how two players' private types affect their policies and utilities under different information structures. We further apply sensitivity analysis to show how the value of the key parameter affects the defender's and the attacker's utilities. For the multi-stage Bayesian game in 5.5.2, we focus on the dynamic of the belief update and state transition under the interaction of the stealthy attacker and the proactive defender. Moreover, we investigate how the adversarial and defensive deception, and how the initial state can affect the stage utility and the cumulative utility of the user and the defender.

5.5.1 Final Stage and SBNE

Players' beliefs affect their policies and expected utilities at the final stage. We discuss three different scenarios as follows. In Fig 5.4a, the defender does not know the user's type. In Fig. 5.5, the user does not know the defender's type. In Fig. 5.4b, both the user and the defender do not know the other's type. In all three scenarios, the x-axis represents the belief of either the user or the defender. The y-axis of the upper figure represents the probability of either the user taking action '*selective monitoring (SM)*' or the defender taking action '*unencrypted command (UC)*'. Fig. 5.4a shows the following trends as the user becomes more likely to be adversarial.



(a) The user knows that the defender is primitive, yet the defender only knows the probability of the user being adversarial.

(b) Both players' types are private, and each player only knows the probability of the other player's type.

Figure 5.4: The SBNE strategy and the expected utility of the primitive defender and the user who is either legitimate or adversarial. The x-axis represents the probability of the user being adversarial. The y-axis of the upper figure represents the probability of either the user taking action '*selective monitoring (SM)*' or the defender taking action '*unencrypted command (UC)*'.

First, two black lines show that the expected utility of the defender decreases and the defender is more inclined to apply action '*complete monitoring*' after her belief exceeds a threshold. Second, two red lines show that the adversarial user takes action '*unencrpted command*' with a higher probability and only gains a reward when the probability of adversarial users is sufficiently small. Thus, we conclude that when the probability of the adversarial user increases, the defender tends to invest more in cyber defense so that the attacker behaves more conservatively and inflicts fewer losses. Third, the two blue lines show that the legitimate user always chooses '*encrypt command*' and receives a constant utility, which indicates that the proactive defense does not affect the behavior and the utility of legitimate users at this stage.

Fig. 5.5 shows that the defender benefits from introducing defensive deception.

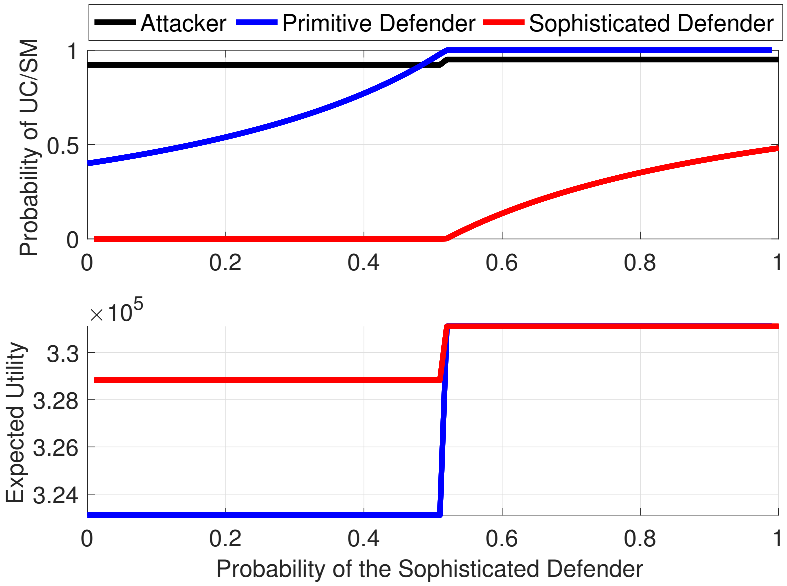


Figure 5.5: The SBNE strategy and the expected utility of the adversarial user and the defender who is either primitive or sophisticated. The defender knows that the user is adversarial while the adversarial user only knows the probability of the defender being primitive. The x-axis represents the probability of the defender being sophisticated. The y-axis of the upper figure represents the probability of either the user taking action '*selective monitoring (SM)*' or the defender taking action '*unencrypted command (UC)*'.

When the defender becomes more likely to a sophisticated one, both types of defenders can have a higher probability to apply the selective monitoring and save the extra surveillance cost of the complete monitoring. The attacker with incomplete information has a threshold policy and switches to a lower attacking probability after reaching the threshold of 0.5 as shown in the black line. When the probability goes beyond the threshold, the primitive defender can pretend to be a sophisticated one and take action '*selective monitoring*'. Meanwhile, a sophisticated defender can reduce the security effort and take action '*selective monitoring*' with a higher probability since the attacker becomes more cautious in taking adversarial actions after identifying the defender as more likely to be sophisticated. It is also observed that the sophisticated defender receives a higher payoff before the attacker's belief reaches the 0.5 threshold. After the belief reaches the threshold, the attacker is threatened to take less aggressive actions, and both types of defenders share the same payoff.

Finally, we consider the double-sided incomplete information where both players' types are private information, and each player only has the belief of the other player's type. Compared with the defender in Fig. 5.4a who takes action '*selective monitoring*' with a probability less than 0.5 and receives a decreasing expected payoff, the defender in Fig. 5.4b can take '*selective monitoring*' with a probability closed to 1 and receive a constant payoff in expectation after the user's belief exceeds the threshold. Thus, the defender can spare defense efforts and mitigate risks by introducing uncertainties on her type as a countermeasure to the adversarial deception.

Sensitivity Analysis

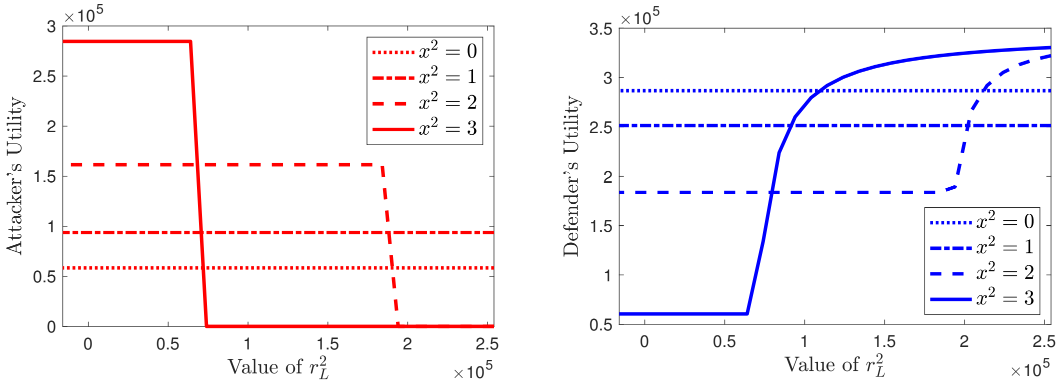


Figure 5.6: Utilities of the primitive defender and the attacker versus the value of r_L^2 under different states $x^2 \in \{0, 1, 2, 3\}$.

As shown in Fig. 5.6, if the value of the penalty r_L^2 is close to 0, i.e., the defense at the final stage is ineffective, then an arrival at state $x^2 = 3$, the highest privilege level can significantly increase the attacker's payoff and cause the most damage to the defender. As more effective defensive methods are employed at the final stage, i.e., the value of r_L^2 increases, the attacker becomes more conservative and strategic in taking adversarial behaviors. Then, the state with the highest privilege level may not be the most favorable state for the attacker.

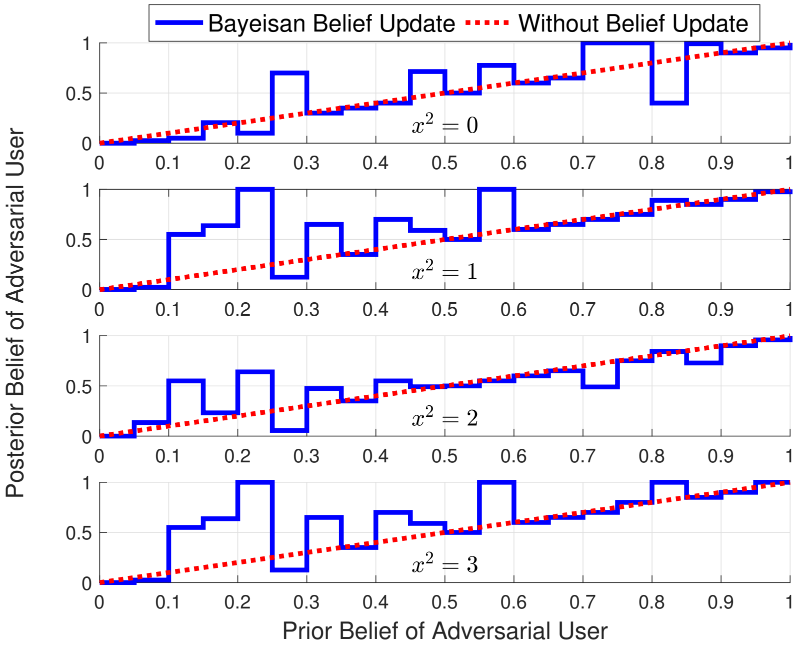


Figure 5.7: The defender's prior and posterior beliefs of the user being adversarial.

5.5.2 Multi-Stage and PBNE

We show in Fig. 5.7 that the Bayesian belief update leads to a more accurate estimate of users' types. Without the belief update, the posterior belief is the same as the prior belief in red and is used as the baseline. As the prior belief increases in the x-axis, the posterior belief after the Bayesian update also increases in blue. The blue line is in general above the red line, which means that with the Bayesian

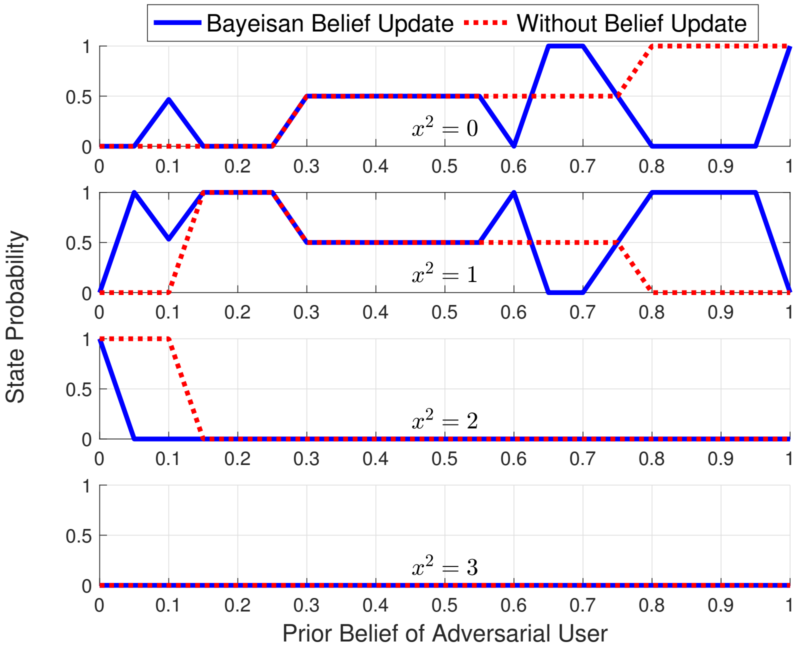


Figure 5.8: The probability of different states $x^2 \in \{0, 1, 2, 3\}$.

update, the defender's belief becomes closer to the right type. Also, we find that the belief update is the most effective when an inaccurate prior belief is used as it corrects the erroneous belief significantly.

In Fig. 5.8, we show that the proactive defense, i.e., defensive methods in intermediate stages can affect the state transition and reduce the probability of attackers reaching states that can result in huge damage at the final stage. As the prior belief of the user being adversarial increases, the attacker is more likely to arrive at state $x^2 = 0$ and $x^2 = 1$, and reduce the probability of visiting $x^2 = 2$ and $x^2 = 3$.

Adversarial and Defensive Deception

Fig. 5.9 investigates the adversarial deception where the attacker takes full control of the defense system and manipulates the defender's belief. As shown in

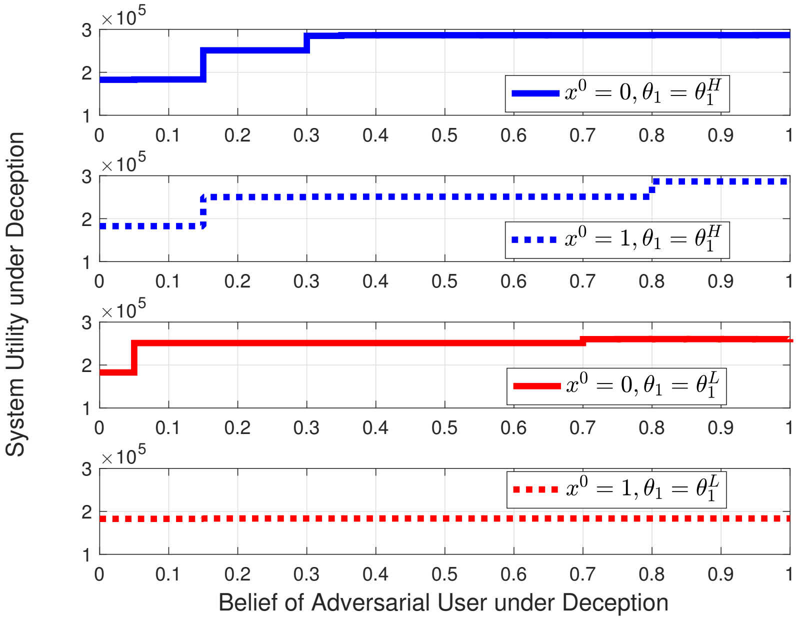


Figure 5.9: The defender's utility under deceived beliefs.

the figure, the defender's utilities all increase when the belief under the deception approaches the correct belief that the user is adversarial. Also, the increase is stair-wise, i.e., the defender only alternates her policy when the manipulated belief is beyond certain thresholds. Under the same manipulated belief, a sophisticated defender benefits no less than a primitive one. The defender receives a lower payoff when the reconnaissance provides effectual intelligence.

Incapable of revealing the adversarial deception completely, the defender can alternatively introduce defensive deceptions, e.g., a primitive defender can disguise himself as a sophisticated one to confuse the attacker. Defensive deceptions introduce uncertainties to attackers, increase their costs, and increase the defender's utility. Fig. 5.10 investigates the defender's and the attacker's utilities under three different scenarios. The complete information refers to the scenario where both players know the other player's type. The deception with the H-type or the

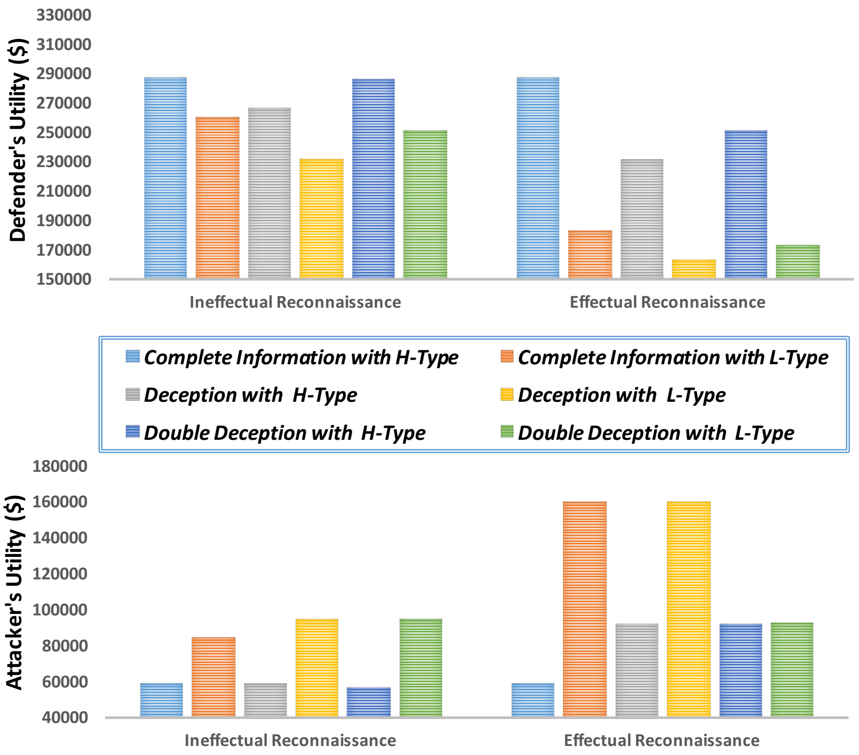


Figure 5.10: The cumulative utilities of the attacker and the defender under the complete information, the adversarial deception, and the defensive deception. In the legend, the left three represent the utilities for a sophisticated defender and the right three represent the ones for a primitive defender.

L-type means that the attacker knows the defender's type to be sophisticated or primitive, respectively, yet the defender has no information about the user's type. The double-sided deception indicates that both players do not know the other player's type. The results from Fig. 5.10 are summarized as follows. First, the sophisticated defender's payoffs can increase as much as 56% than those of the primitive defender. Also, a prevention of effectual reconnaissance increases the defender's utility by as much as 41% and reduces the attacker's utility by as much as 38%. Second, the defender and the attacker receive the highest and the lowest payoff, respectively, under the complete information. When the attacker introduces deceptions over his type, the attacker's utility increases and the defender's utility decreases. Third, when the defender adopts defensive deceptions to introduce double-sided incomplete information, we find that the decrease of the sophisticated defender's utilities is reduced by at most 64%, i.e., changes from $\$55,570$ to $\$35,570$ when the reconnaissance is effectual. The double-sided incomplete information also brings lower utilities to the attacker than the one-sided adversarial deception. However, the defender's utility under the double-sided deception is still less than the complete information case, which concludes that acquiring complete information of the adversarial user is the most effective defense. However, if the complete information cannot be obtained, the defender can mitigate her loss by introducing defensive deceptions.

Chapter 6

Rational and Persistent Deception among Intelligent Robots

Recent advances in automation and adaptive control in multi-agent systems enable robots to use deception to accomplish their objectives. Since robots are critical components of CPSs for life-critical tasks, technologies to counteract adversarial robot deception are indispensable to achieving high-confidence robot systems. Deception involves intentional information hiding to compromise the security and operational efficiency of the robotic systems. This work proposes a dynamic game framework to quantify the impact of deception, understand the robots' behaviors and intentions, and design cost-efficient strategies under the deception that persists over stages. Existing researches on robot deception have relied on experiments while this work aims to lay a theoretical foundation of deception with quantitative metrics, such as deceivability and the price of deception. The proposed model has wide applications, including cooperative robots, pursuit and evasion, and human-robot teaming. The pursuit-evasion games are used as case studies to show how the

Table 6.1: Summary of variables and their meanings.

Variable	**Meaning**
$\mathcal{N} := \{1, 2, \cdots, N\}$	Set of N players in the dynamic game
$\mathcal{K} := \{0, 1, 2, \cdots, K\}$	Set of K discrete stages in the dynamic game
$\Theta_i := \{\theta_i^1, \theta_i^2, \cdots, \theta_i^{N_i}\}$	Set of N_i possible types for player $i \in \mathcal{N}$
$\theta_i \in \Theta_i$	Type of player $i \in \mathcal{N}$
$\theta := [\theta_1, \cdots, \theta_N]$	N players' joint type
$\Theta_{-i} := \prod_{j \in \mathcal{N} \setminus \{i\}} \Theta_j$	Set of types of all players except for player i
$\theta_{-i} := [\theta_j]_{j \in \mathcal{N} \setminus \{i\}} \in \Theta_{-i}$	Types of all players except for player i
$\Delta(\Theta_{-i})$	Set of probability distributions over set Θ_{-i}
$\Xi_i(\cdot)$	Probability distribution of player i's type
$\Xi = [\Xi_i]_{i \in \mathcal{N}}$	Probability distribution of the joint type θ
$\Xi_w(\cdot)$	Probability distribution of noise $w^k, \forall k \in \mathcal{K}$
$x^k \in \mathbb{R}^{n \times 1}$	System state of dimension n at stage k
$x_i^k \in \mathbb{R}^{n_i \times 1}$	Player i's state of dimension n_i at stage k
$[\hat{x}_i^k(\theta_i)]_{k \in \mathcal{K}}$	Reference trajectory for player i of type θ_i
$\beta_i^k \in \Lambda_i \subseteq [0, 1]^{\lvert\Theta_{-i}\rvert \times \lvert\Theta_i\rvert}$	Player i's belief state at stage k
$\beta^k = [\beta_i^k]_{i \in \mathcal{N}} \in \Lambda$	N players' joint belief state at stage k
$h^k := [x^0, \cdots, x^k] \in \mathcal{H}^k$	State history
f^k	State transition function at stage k
Γ_i^k, g_i^k	Player i's belief transition and cost at stage k
$V_i^k(\beta^k, x^k, \theta_i)$	Player i's PBNE cost
$\bar{V}_i^k(x^k, \theta)$	Player i's PBNE cost when all players' types are *common knowledge*
$l_i^k(\theta_{-i} \mid h^k, \theta_i)$	Player i's belief at stage k, i.e., the probability of other players' types being θ_{-i} based on player i's available information of h^k, θ_i

deceiver can amplify the deception by belief manipulation and how the deceived robots can reduce the negative impact of deception by enhanced maneuverability and Bayesian learning. We summarize main notations in Table 6.1.

6.1 Dynamic Game with Private Types

We model deception as a K-stage game consisting of N robots as players and each robot has asymmetric information. Let $\mathcal{N} := \{1, \cdots, N\}$ be the set of N players and $\mathcal{K} := \{0, 1, 2, \cdots, K\}$ be the set of K discrete stages. Private information of player $i \in \mathcal{N}$, i.e., his type θ_i, is modeled as the realization of a discrete random variable with a finite support $\Theta_i := \{\theta_i^1, \theta_i^2, \cdots, \theta_i^{N_i}\}$ and a prior probability distribution $\Xi_i(\cdot)$. Hence, N_i is the number of possible types for player i and $\Xi_i(\theta_i)$ is the probability that player i's type is θ_i. Define shorthand notation $\Xi := [\Xi_i]_{i\in\mathcal{N}}$ and let $\Theta_{-i} := \prod_{j\in\mathcal{N}\setminus\{i\}} \Theta_j$ be the set of types of all players except for player $i \in \mathcal{N}$. Each player i knows the value of his own type θ_i, but does not know the values of other players' types $\theta_{-i} := [\theta_j]_{j\in\mathcal{N}\setminus\{i\}} \in \Theta_{-i}$, throughout K stages of the game. The system state dynamics under N players' joint action $u^k := [u_1^k, \cdots, u_N^k]$, joint type $\theta := [\theta_1, \cdots, \theta_N]$, and an additive external noise $w^k \in \mathbb{R}^{n\times 1}$ are shown in (6.1):

$$x^{k+1} = f^k(x^k, u_1^k, \cdots, u_N^k, \theta_1, \cdots, \theta_N) + w^k, k \in \mathcal{K} \setminus \{K\}. \tag{6.1}$$

The dynamics in (6.1) can have different interpretations based on applications. In the pursuit-evasion scenario as in [124], $x_i^k \in \mathbb{R}^{n_i\times 1}$ represents robot i's local states such as its location and speed. The system state $x^k \in \mathbb{R}^{n\times 1}$ can be explicitly represented by N robots' joint state $[x_1^k, \cdots, x_N^k]$ with $n = \sum_{i=1}^N n_i$. In the application where N robots cooperatively transport a payload, e.g., [72, 202], system state $x^k \in \mathbb{R}^{n\times 1}$ represents the payload's location and posture, which does not explicitly relate to robots' local states. The noise sequence $[w^k]_{k\in\mathcal{K}}$ assumed to be independent with probability density function $\Xi_w(\cdot)$, i.e., $\mathbb{E}_{w^k,w^h\sim\Xi_w}[w^k(w^h)'] = 0, \forall k \in \mathcal{K}, h \in \mathcal{K}\setminus\{k\}$. The noise is not necessarily Gaussian distributed but is assumed to have a zero mean, i.e., $\mathbb{E}_{w^k\sim\Xi_w}[w^k] = 0, \forall k \in \mathcal{K}$. We assume that system dynamics (6.1) are multi-agent controllable as defined in Definition 8 so that players can design their deceptive actions to reach the entire state space in finite stages.

Definition 8 (Multi-Agent Controllability). *System dynamics (6.1) are called multi-agent controllable if for any target state $x^k \in \mathbb{R}^{n\times 1}$ at stage $k \in \mathcal{K} \setminus \{0\}$, initial state $x^0 \in \mathbb{R}^{n\times 1}$, and joint type $\theta \in \Theta$, there exists a sequence of finite joint actions $u^{0:k}$ that drive the system state from x^0 to x^k in expectation.*

6.1.1 Forward Belief Dynamics

At each stage $k \in \mathcal{K}$, the information available to player i compromises all players' state history $h^k := [x^0, \cdots, x^k] \in \mathcal{H}^k$ as well as his own type value θ_i. Define $\Delta(\Theta_{-i})$ as the set of probability distributions over set Θ_{-i}. Each player i at stage k forms a belief $l_i^k : \mathcal{H}^k \times \Theta_i \mapsto \triangle\Theta_{-i}$ based on his available information. Thus, $l_i^k(\cdot|h^k, \theta_i)$ is a probability measure of other players' types, i.e., $\sum_{\theta_{-i}\in\Theta_{-i}} l_i^k(\theta_{-i}|h^k, \theta_i) = 1, \forall h^k \in \mathcal{H}^k, \theta_i \in \Theta_i$. Define a vector

$$\beta_i^k := [l_i^k(\theta_{-i}|h^k, \theta_i^1), l_i^k(\theta_{-i}|h^k, \theta_i^2), \cdots, l_i^k(\theta_{-i}|h^k, \theta_i^{N_i})]_{\theta_{-i}\in\Theta_{-i}}$$

as player i's belief state at stage $k \in \mathcal{K}$. We assume that the set of belief states is independent of stages, i.e., $\beta_i^k \in \Lambda_i \subseteq [0,1]^{|\Theta_{-i}|\times|\Theta_i|}$. Then, we can represent player i's belief dynamics as

$$\beta_i^{k+1} := \Gamma_i^k(\beta_i^k, u^k, w^k, \theta_i), \forall k \in \{0, \cdots, K-1\}. \tag{6.2}$$

Note that the belief transition function Γ_i^k can be different for each i and k, i.e., players' belief updates can be heterogeneous and time-varying. Define $\beta^k := [\beta_i^k]_{i\in\mathcal{N}} \in \Lambda := \prod_{i\in\mathcal{N}} \Lambda_i$. In this work, we assume that the initial beliefs of all players of all types β^0 and the belief update rules $\Gamma_i^k, \forall i \in \mathcal{N}, \forall k \in \{0, \cdots, K-1\}$, are *common knowledge*. In the next two subsections, we provide two specific forms of Γ_i^k that rely on *intrinsic* and *extrinsic* information, respectively.

Bayesian Belief Dynamics

The most common belief update rule Γ_i^k in (6.2) for player i at stage $k+1$ uses Bayesian inference. Given the knowledge of the sequential state observations x^k, x^{k+1} and all players' actions u^k, each player i of type $\theta_i \in \Theta_i$ at stage $k+1$ can update his belief as follows: $\forall \theta_{-i} \in \Theta_{-i}$,

$$l_i^{k+1}(\theta_{-i}|h^{k+1}, \theta_i) = \frac{l_i^k(\theta_{-i}|h^k, \theta_i)\,\Pr(x^{k+1}|\theta_{-i}, x^k, \theta_i)}{\sum_{\bar{\theta}_{-i} \in \Theta_{-i}} l_i^k(\bar{\theta}_{-i}|h^k, \theta_i)\,\Pr(x^{k+1}|\bar{\theta}_{-i}, x^k, \theta_i)}. \tag{6.3}$$

In (6.3), we use the Markov property, i.e.,

$$\Pr(x^{k+1}|\theta_{-i}, h^k, \theta_i) = \Pr(x^{k+1}|\theta_{-i}, x^k, \theta_i) = \Xi_w(x^{k+1} - f^k(x^k, u^k, \theta)).$$

The denominator is positive as $w^k \in \mathbb{R}^{n \times 1}$.

Remark 7 (**Actions Reveal Type Information**). *Even if the state dynamics f^k in (6.1) are independent of $\theta_j, \forall j \in \mathcal{N} \setminus \{i\}$, player $i \in \mathcal{N}$ can still learn player j' type via (6.3) as player j's action u_j^k is a function*[1] *of his type θ_j.*

Markov-Chain Belief Dynamics

In section 6.1.1, we assume that players can exploit the *intrinsic* information of state dynamics f^k, state observations x^k, x^{k+1}, and the prediction of

[1] Each player's action is a function of his type as his cost is related to his type and the action aims to minimize his cost.

all players' actions u^k. Since the above *intrinsic* information may not be available in practice, we consider the belief dynamics with *extrinsic* information in this subsection. In particular, we assume that each player i's belief dynamics $\beta_i^{k+1} := \Gamma_i^k(\beta_i^k, w^k, \theta_i), \forall k \in \{0, \cdots, K-1\}$, are a discrete-time Markov chain where the *extrinsic* information at stage k is characterized by the transition function $\Gamma_i^k(\cdot, w^k, \theta_i)$. Note that the transition function only characterizes how players update their beliefs at each stage yet does not guarantee that a player can learn the true types of others. The following example illustrates a class of players whose belief dynamics exhibit the confirmation bias [155] where players ignore intrinsic evidence such as u^k and preserve their belief update rules Γ_i^k at each stage k.

Example 2. *Consider a two-person game $N = 2$ where the first player has two types $N_1 = 2, \Theta_1 = \{\theta_1^1, \theta_1^2\}$ and the second player only has one type $N_1 = 1, \Theta_2 = \{\theta_2^1\}$. The second player's belief state $\beta_2^k = [l_2^k(\theta_1^1|\theta_2^1), l_2^k(\theta_1^2|\theta_2^1)]$ toward the first player's type belongs to a finite set $\Lambda_2 = \{[0.2, 0.8], [0.5, 0.5], [0.8, 0.2]\}$. The transition function Γ_2^k is independent of k: if the current belief state is $[0.5, 0.5]$, then the belief at the next stage is $[0.2, 0.8], [0.5, 0.5]$, or $[0.8, 0.2]$ with probability $0.4, 0.2, 0.4$, respectively. If the current belief state is $[0.8, 0.2]$ (resp. $[0.2, 0.8]$), then the belief at the next stage is $[0.8, 0.2]$ (resp. $[0.2, 0.8]$) or $[0.5, 0.5]$ with probability 0.9 and 0.1, respectively. The above transition function Γ_2^k means that the second player tends to interpret the extrinsic information of the first player's type based on his current belief. If the second player already believes that the first player is of type θ_1^1 with a high probability of 0.8 at stage k, i.e., $\beta_2^k = [0.8, 0.2]$, then the second player is more inclined to enhance his current belief, i.e., his belief state at the next stage, i.e., β_2^{k+1}, will remain to be $[0.8, 0.2]$ with a high probability of 0.9. The above transition function represents the phenomena of attitude polarization and confirmation bias* *where players preserve their existing beliefs and the disagreement becomes more extreme at each stage even when players are exposed to the same evidence.*

6.1.2 Nonzero-Sum Cost Function and Equilibrium

At non-terminal stage $k \in \mathcal{K} \setminus \{K\}$, player i's cost function is $g_i^k : \mathbb{R}^{n\times 1} \times \prod_{j=1}^{N} \mathbb{R}^{m_j \times 1} \times \Theta_i \mapsto \mathbb{R}$. The final stage cost is $g_i^K : \mathbb{R}^{n\times 1} \times \Theta_i \mapsto \mathbb{R}$. Define $u_i^{k_0:K-1} := [u_i^{k_0}, \cdots, u_i^{K-1}]$ as player i's action sequence from stage k_0 to $K-1$ and $u^{k_0:K-1} := [u_i^{k_0:K-1}, u_{-i}^{k_0:K-1}]$ as player i's and all other players' action sequences from stage k_0 to $K-1$. Player i's expected cumulative cost from arbitrary initial stage $k_0 \in \mathcal{K}$ to the terminal stage K is defined as

$$\begin{aligned} J_i^{k_0}(l_i^{k_0:K-1}, u^{k_0:K-1}, x^{k_0}, \theta_i) &= \mathbb{E}_{w^{K-1}\sim\Xi_w}[g_i^K(x^K, \theta_i)] \\ &+ \sum_{k=k_0}^{K-1} \mathbb{E}_{w^{k-1}\sim\Xi_w}\left[\mathbb{E}_{\theta_{-i}\sim l_i^k}[g_i^k(x^k, u^k, \theta_i)]\right]. \end{aligned} \tag{6.4}$$

The expectations are taken first over the external noise sequence w^k and then over other players' internal type uncertainty. We cannot exchange the order of these two expectations as l_i^k is a function of w^{k-1}. Each player i at stage $k_0 \in \mathcal{K}$ aims to minimize $J_i^{k_0}$ by choosing only his action sequence $u_i^{k_0:K-1}$ but not other players' action sequence $u_{-i}^{k_0:K-1}$. The following definition of sequential rationality in Definition 9 guarantees that each player i has no motivation to deviate from the sequentially rational action at any stage $k \in \{k_0, \cdots, K-1\}$ during the interaction if all other players adopt the sequentially rational actions.

Definition 9. *An action sequence $u^{*,k_0:K-1} := \{u_i^{*,k_0:K-1}, u_{-i}^{*,k_0:K-1}\}$ is called sequentially rational for player i under the belief sequence $l_i^{k_0:K-1}$, state x^{k_0}, and type θ_i, if for any state x^k at stage $k \in \{k_0, \cdots, K-1\}$, player i does not benefit from* *taking any other action sequence* $u_i^{k:K-1}$*, i.e.,* $J_i^k(l_i^{k:K-1}, u_i^{*,k:K-1}, u_{-i}^{*,k:K-1}, x^k, \theta_i) \leq J_i^k(l_i^{k:K-1}, u_i^{k:K-1}, u_{-i}^{*,k:K-1}, x^k, \theta_i), \forall u_i^{k:K-1}$.

Since players' actions may affect their future beliefs as captured by the belief dynamics Γ_i^k in (6.2), we further require the equilibrium action $u^{*,k_0:K-1}$ in Definition 9 to be consistent with the belief dynamics, which leads to the following definition of Perfect Bayesian Nash Equilibrium (PBNE).

Definition 10. *Consider the* N*-player dynamic game of private types and asymmetric information defined by the state dynamics* (6.1) *and the expected cumulative cost* (6.4)*. The action sequence* $u^{*,0:K-1} := \{u_i^{*,0:K-1}, u_{-i}^{*,0:K-1}\}$ *of* N *players over* K *stages compromises the Perfect Bayesian Nash Equilibrium (PBNE) if, regardless of each player* i*'s type* $\theta_i \in \Theta_i$*, the following statements hold.*

1. ***Sequential rationality****:* $u^{*,0:K-1}$ *is sequential rational for each player* $i \in \mathcal{N}$ *under his belief sequence* $l_i^{*,0:K-1}$*;*

2. ***Belief consistency****: each player* i*'s belief sequence* $l_i^{*,0:K-1}$ *is consistent with* (6.2) *under* $u^{*,0:K-1}$*.*

Proposition 1. *It is sufficient to represent player* i*'s equilibrium cost, denoted as* $J_i^k(l_i^{*,k:K-1}, u^{*,k:K-1}, x^k, \theta_i)$*, under the PBNE action* $u^{*,k:K-1}$ *at stage* $k \in \mathcal{K}$ *as a function of* β^k*,* x^k *and* θ_i*, which is defined as* $V_i^k(\beta^k, x^k, \theta_i)$*. Under the boundary condition* $V_i^K(\beta^K, x^K, \theta_i) := g_i^K(x^K, \theta_i)$*, the following holds for all* $k \in \{0, \cdots, K-1\}$ *and all* $x^k \in \mathbb{R}^{n\times 1}, \beta^k \in \Lambda$*, i.e.,*

$$V_i^k(\beta^k, x^k, \theta_i) = \min_{u_i^k} \sum_{\theta_{-i}} l_i^k(\theta_{-i}|h^k, \theta_i)\{g_i^k(x^k, u^k, \theta_i) + \\ \mathbb{E}_{w^k\sim\Xi_w}[V_i^{k+1}(\beta^{k+1}, x^{k+1}, \theta_i)]\}, \forall \theta_i \in \Theta_i, \forall i \in \mathcal{N}, \tag{6.5}$$

where β^{k+1} and x^{k+1} satisfy (6.2) *and* (6.1)*, respectively.*

Proof. According to the definition of PBNE, at the second last stage $k = K - 1$, each player i's equilibrium action

$$u_i^{*,k} = arg \min_{u_i^k} \mathbb{E}_{\theta_{-i} \sim l_i^k}[g_i^k(x^k, u^k, \theta_i)] + \mathbb{E}_{w^k \sim \Xi_w}[g_i^K(x^K, \theta_i)]$$

is in general a function of $\theta_i, x^k, l_i^{*,k}, u_{-i}^{*,k}$. Due to the coupling between $u_i^{*,k}$ and $u_{-i}^{*,k}$, we need to solve a set of system equations for all $i \in \mathcal{N}$ and $\theta_i \in \Theta_i$. Then, $u_i^{*,k}$ will be a function of β^k, x^k, θ_i and we obtain (6.5) at stage $k = K - 1$. We can repeat the above procedure from $k = K - 2$ to $k = 0$ to obtain the recursive form in (6.5). □

Proposition 1 characterizes the structure of the equilibrium action $u_i^{*,k}$ and the equilibrium cost $V_i^k(\beta^k, x^k, \theta_i)$ for each player i of type θ_i under the solution concept of PBNE; i.e., both terms are feedback functions of the belief state β^k, the physical state x^k, and the player' type θ_i. Although J_i^k is a function of beliefs $l_i^{k:K-1}$ over all the remaining stages, $V_i^k(\beta^k, x^k, \theta_i)$ only depends on the belief state at the current stage k. If all players' types are *common knowledge*, PBNE still applies and we can define a new function $\bar{V}_i^k(x^k, \theta)$ to represent the resulting equilibrium cost $V_i^k(\beta^k, x^k, \theta_i)$ for all $k \in \mathcal{K}$ without loss of generality.

6.1.3 Offline Evaluation of Equilibrium Cost

If each player i's initial belief confirms to the prior distribution of other players' types, i.e., $l_i^0(\theta_j|x^0, \theta_i) = \Xi_j(\theta_j), \forall \theta_i \in \Theta_i, j \in \mathcal{N}, \theta_j \in \Theta_j, \forall x^0$, then each player i at system state x^0 with belief state β^0 can use his expected equilibrium cost $\mathbb{E}_{\theta_i \sim \Xi_i}[V_i^0(\beta^0, x^0, \theta_i)]$ over his type uncertainty Ξ_i as an offline performance measure of the equilibrium action $u^{*,0:K}$. As a comparison, player i's expected equilibrium cost $\mathbb{E}_{\theta\sim\Xi}[\bar{V}_i^0(x^0,\theta)]$ under the complete information game serves as a benchmark. Note that player i does not need to know the realization of the joint type θ to compute $\mathbb{E}_{\theta\sim\Xi}[\bar{V}_i^0(x^0,\theta)]$. Due to the coupling in dynamics, costs, and cognition among N players, obtaining more information and knowing the type of another player $j \in \mathcal{N} \setminus \{i\}$ may not always improve player i's performance; i.e., there is no guarantee that $\mathbb{E}_{\theta_i\sim\Xi_i}[V_i^0(\beta^0,x^0,\theta_i)] \geq \mathbb{E}_{\theta\sim\Xi}[\bar{V}_i^0(x^0,\theta)]$. Besides the above performance evaluation for an individual player $i \in \mathcal{N}$ under deception, we may also aim to evaluate the overall performance of multiple players or all N players. We define the Price of Deception (PoD) in Definition 11 with a set of coefficients $\eta_i \in [0,1], \forall i \in \mathcal{N}, \sum_{i\in\mathcal{N}} \eta_i = 1$. Since the equilibrium cost can be negative, we let $\eta_0(\Xi) := -\min(0, \{\mathbb{E}_{\theta_i\sim\Xi_i}[V_i^0(\beta^0,x^0,\theta_i)]\}_{i\in\mathcal{N}}, \{\mathbb{E}_{\theta\sim\Xi}[\bar{V}_i^0(x^0,\theta)]\}_{i\in\mathcal{N}})$ be the normalizing constant to guarantee that $p^\eta(\Xi)$ is non-negative for all chosen coefficients $\eta_i, i \in \mathcal{N}$.

Definition 11. *For a given set of coefficients $\eta := \{\eta_i\}_{i\in\mathcal{N}\cup\{0\}}$, the Price of Deception (PoD) of the N-player K-stage game defined by (6.1), (6.4), and (6.2) under the prior probability distribution $\Xi = [\Xi_i]_{i\in\mathcal{N}}$ is*

$$p^\eta(\Xi) := \frac{\sum_{i\in\mathcal{N}} \eta_i \mathbb{E}_{\theta\sim\Xi}[\bar{V}_i^0(x^0,\theta)] + \eta_0(\Xi)}{\sum_{i\in\mathcal{N}} \eta_i \mathbb{E}_{\theta_i\sim\Xi_i}[V_i^0(\beta^0,x^0,\theta_i)] + \eta_0(\Xi)} \in [0,\infty).$$

The PoD is a crucial evaluation and design metric. We can endow PoD with different meanings by properly choosing the weighting coefficients $\eta_i, i \in \mathcal{N}$. For example, if besides N players, there is a central planner who aims to minimize the total cost of all N players under their deceptive interaction. Then, we can pick $\eta_i = 1/N, i \in \mathcal{N}$, to represent the overall system performance. Although the central planner cannot control players' state dynamics, costs, and belief dynamics directly, he can still affect their deceptive interaction if he can design the prior probability distribution Ξ of the joint type θ. If the central planner instead only aims to reduce the cost of one player $j \in \mathcal{N}$, then we can pick $\eta_j = 1$ and $\eta_h = 0, \forall h \in \mathcal{N} \setminus \{j\}$. With a given weighting parameters η, a larger value of $p_\eta(\Xi)$ indicates a better accomplishment of the above goals. Note that individual deception may improve the system performance, i.e., $p^\eta(\Xi) > 1$.

6.2 Linear-Quadratic Specification

Linear-Quadratic (LQ) game is an important class of dynamic games. They can also be applied iteratively to approximate nonlinear stochastic systems with general cost functions and obtain equilibrium actions [54]. In the following sections, we consider linear state dynamics

$$f^k(x^k, u^k, \theta) := A^k(\theta)x^k + \sum_{i=1}^{N} B_i^k(\theta_i)u_i^k, \tag{6.6}$$

with stage-varying matrices $A^k(\theta) \in \mathbb{R}^{n\times n}$, $B_i^k(\theta_i) \in \mathbb{R}^{n\times m_i}$.

Remark 8. *System (6.6) is multi-agent controllable if and only if matrices, denoted as $H_i^k(\theta) := [B_i^{k-1}(\theta_i), \cdots, \prod_{h=2}^{k-1} A^h(\theta)B_i^1(\theta_i), \prod_{h=1}^{k-1} A^h(\theta)B_i^0(\theta_i)], \forall i \in \mathcal{N}, \forall \theta \in \Theta, \forall k \in \mathcal{K}$, are of full rank as noise w^k has zero mean and we can obtain $\mathbb{E}[x^k] = \prod_{h=0}^{k-1} A^h(\theta)x^0 + \sum_{r=1}^{N} H_r^k(\theta)[u_r^{k-1}; \cdots; u_r^0]$ by induction.*

Each player i's cost is quadratic in both x^k and u^k; i.e.,

$$\begin{aligned} g_i^k(x^k,u^k,\theta_i) &= (x^k-\hat{x}_i^k(\theta_i))'D_i^k(\theta_i)(x^k-\hat{x}_i^k(\theta_i)) \\ &+\hat{f}_i^k(\hat{x}_i^k(\theta_i))+\sum_{j=1}^{N}(u_j^k)'F_{ij}^k(\theta_i)u_j^k, \forall k\in\mathcal{K}, \end{aligned} \tag{6.7}$$

where $[\hat{x}_i^k(\theta_i)]_{k\in\mathcal{K}}$ is a known type-dependent reference trajectory for player $i\in\mathcal{N}$ and $\hat{f}_i^k$ is a known function of $\hat{x}_i^k(\theta_i)$. The cost matrices $D_i^k(\theta_i)\in\mathbb{R}^{n\times n}, F_{ij}^k(\theta_i)\in\mathbb{R}^{m_i\times m_i}, \forall i,j\in\mathcal{N},k\in\mathcal{K}$, are symmetric. At the final stage, $F_{ij}^K(\theta_i)\equiv\mathbf{0}_{m_i,m_i}$, $\forall i,j\in\mathcal{N},\forall\theta_i\in\Theta_i$. We introduce the following three sets of notations for the belief matrix, the extended Riccati equations, and the matrix-form equilibrium action, respectively.

Belief Matrix With a little abuse of notation, we can define the marginal probability $l_i^k(\theta_j|h^k,\theta_i):=\sum_{\theta_r\in\Theta_r,r\in\mathcal{N}\setminus\{i,j\}}l_i^k(\theta_{-i}|h^k,\theta_i),\forall j\in\mathcal{N}\setminus\{i\}$, as the player i's belief toward the player j's type at stage k. Define the belief matrix for all $i\in\mathcal{N},j\in\mathcal{N}\setminus\{i\},k\in\{0,\cdots,K-1\}$, as

$$\mathbf{L}_{ij}^k:=\begin{bmatrix} \mathbf{L}_i^k(\theta_j^1|h^k,\theta_i^1), & \cdots & \mathbf{L}_i^k(\theta_j^{N_j}|h^k,\theta_i^1) \\ \mathbf{L}_i^k(\theta_j^1|h^k,\theta_i^2), & \cdots & \mathbf{L}_i^k(\theta_j^{N_j}|h^k,\theta_i^2) \\ \vdots & \ddots & \vdots \\ \mathbf{L}_i^k(\theta_j^1|h^k,\theta_i^{N_i}), & \cdots & \mathbf{L}_i^k(\theta_j^{N_j}|h^k,\theta_i^{N_i}) \end{bmatrix}, \tag{6.8}$$

where each block element $\mathbf{L}_i^k(\theta_j^r|h^k,\theta_i^h)=\text{Diag}[l_i^k(\theta_j^r|h^k,\theta_i^h),\cdots,l_i^k(\theta_j^r|h^k,\theta_i^h)]\in\mathbb{R}^{n\times n},\forall r\in\{1,\cdots,N_j\},\forall h\in\{1,\cdots,N_i\}$. Since all its elements are positive and all rows sum to one, the belief matrix $\mathbf{L}_{ij}^k$ is a *right stochastic matrix.*

Extended Riccati Equations A sequence of symmetric matrices $S_i^k(\beta^k, \theta_i) \in \mathbb{R}^{n\times n}$, vectors $N_i^k(\beta^k, \theta_i) \in \mathbb{R}^{n\times 1}$, and scalars $q_i^k(\beta^k, \theta_i) \in \mathbb{R}$ satisfies the following extended Riccati equations for all $\beta^k \in \Lambda, i \in \mathcal{N}, \theta_i \in \Theta_i, k \in \{0, \cdots, K-1\}$:

$$\begin{aligned} S_i^k =& D_i^k + \mathbb{E}_{\theta_{-i}\sim l_i^k}\Big[(A^k + \sum_{j=1}^N B_j^k \Psi_j^{1,k})' \mathbb{E}_{w^k\sim\Xi_w}[S_i^{k+1}] \\ & \cdot (A^k + \sum_{j=1}^N B_j^k \Psi_j^{1,k}) + \sum_{j=1}^N (\Psi_j^{1,k})' F_{ij}^k \Psi_j^{1,k}\Big], \end{aligned} \tag{6.9}$$

$$\begin{aligned} N_i^k =& -2D_i^k \hat{x}_i^k + \mathbb{E}_{\theta_{-i}\sim l_i^k}\Big[(\sum_{j=1}^N B_j^k \Psi_j^{1,k} + A^k)'(\mathbb{E}_{w^k\sim\Xi_w}[N_i^{k+1}] \\ & + 2\mathbb{E}_{w^k\sim\Xi_w}[S_i^{k+1}] \sum_{j=1}^N B_j^k \Psi_j^{2,k}) + 2\sum_{j=1}^N (\Psi_j^{1,k})' F_{ij}^k \Psi_j^{2,k}\Big], \end{aligned} \tag{6.10}$$

$$\begin{aligned} q_i^k =& (\hat{x}_i^k)' D_i^k \hat{x}_i^k + \hat{f}_i^k(\hat{x}_i^k) + \mathbb{E}_{w^k\sim\Xi_w}[(w^k)' S_i^{k+1} w^k + q_i^{k+1}] \\ & + \mathbb{E}_{\theta_{-i}\sim l_i^k}\Big[(\sum_{j=1}^N B_j^k \Psi_j^{2,k})' \mathbb{E}_{w^k\sim\Xi_w}[S_i^{k+1}] \sum_{j=1}^N B_j^k \Psi_j^{2,k} \\ & + (\sum_{j=1}^N B_j^k \Psi_j^{2,k})' \mathbb{E}_{w^k\sim\Xi_w}[N_i^{k+1}] + \sum_{j=1}^N (\Psi_j^{2,k})' F_{ij}^k \Psi_j^{2,k}\Big], \end{aligned} \tag{6.11}$$

where functions $\Psi_i^{1,k}, \Psi_i^{2,k}, \forall i \in \mathcal{N}$, are defined below. The boundary conditions of the extended Riccati equations are

$$S_i^K = D_i^K;\ N_i^K = -2D_i^K \hat{x}_i^K;\ q_i^K = (\hat{x}_i^K)' D_i^K \hat{x}_i^K + \hat{f}_i^K(\hat{x}_i^K). \tag{6.12}$$

Equilibrium Action in Matrix Form We need to represent the equilibrium action of all players under all types in matrix form as each player's action is coupled with other players' actions under PBNE. Since each player i has different equilibrium actions under different types, with a little abuse of notation, we write each player i's action as a function of his type θ_i and define two action vectors $\mathbf{u}_i^k := [u_i^k(\theta_i^1), \cdots, u_i^k(\theta_i^{N_i})]' \in \mathbb{R}^{m_i N_i \times 1}$ and $\mathbf{u}^k := [\mathbf{u}_1^k, \mathbf{u}_2^k \cdots, \mathbf{u}_N^k]' \in \mathbb{R}^{\sum_{r=1}^N m_r N_r \times 1}$. For all $i \in \mathcal{N}, l_i^k, \theta_i \in \Theta_i, k \in \{0, \cdots, K-1\}$, define a series of (m_i)-by-(m_i) square matrices

$$R_i^k(\beta^k, \theta_i) := F_{ii}^k(\theta_i) + (B_i^k(\theta_i))' S_i^{k+1}(\beta^k, \theta_i) B_i^k(\theta_i).$$

Let $\mathbf{B}_i^k := \text{Diag}[B_i^k(\theta_i^1) \cdots, B_i^k(\theta_i^{N_i})]$ be $(N_i n)$-by-$(N_i m_i)$ block matrices. Let $\mathbf{S}_i^k(\beta^k) := \text{Diag}[S_i^k(\beta^k, \theta_i^1), \cdots, S_i^k(\beta^k, \theta_i^{N_i})]$ be $(N_i n)$-by-$(N_i n)$ block matrices. Finally, for any $\beta^k \in \Lambda$, we define three categories of parameter matrices $\mathbf{W}^{1,k}(\beta^k) = [W_1^{1,k}(\beta^k); \cdots; W_N^{1,k}(\beta^k)] \in \mathbb{R}^{\sum_{r=1}^N m_r N_r \times n}$, $\mathbf{W}^{2,k}(\beta^k) = [W_1^{2,k}(\beta^k); \cdots; W_N^{2,k}(\beta^k)] \in \mathbb{R}^{\sum_{r=1}^N m_r N_r \times 1}$, and $\mathbf{W}^{0,k}(\beta^k) := [W_{ij}^{0,k}(\beta^k) \in \mathbb{R}^{m_i N_i \times m_j N_j}]_{i,j \in \mathcal{N}}$. Their elements are given as follows; i.e., $\forall i \in \mathcal{N}, \forall k \in \{0, \cdots, K-1\}$,

$$\begin{aligned}
W_i^{1,k}(\beta^k) &= \Big[(B_i^k(\theta_i^1))' S_i^{k+1}(\beta^k, \theta_i^1) \mathbb{E}_{\theta_{-i} \sim l_i^k}[A^k(\theta_i^1, \theta_{-i})]; \\
&\qquad \cdots; (B_i^k(\theta_i^{N_i}))' S_i^{k+1}(\beta^k, \theta_i^{N_i}) \mathbb{E}_{\theta_{-i} \sim l_i^k}[A^k(\theta_i^{N_i}, \theta_{-i})]\Big], \\
W_i^{2,k}(\beta^k) &= \frac{1}{2}\Big[(B_i^k(\theta_i^1))' N_i^{k+1}(\beta^k, \theta_i^1); \\
&\qquad \cdots; (B_i^k(\theta_i^{N_i}))' N_i^{k+1}(\beta^k, \theta_i^{N_i})\Big], \\
W_{ii}^{0,k}(\beta^k) &= \text{Diag}[R_i^k(\beta^k, \theta_i^1), \cdots, R_i^k(\beta^k, \theta_i^{N_i})], \\
W_{ij}^{0,k}(\beta^k) &= (\mathbf{B}_i^k)' \mathbf{S}_i^{k+1}(\beta^k) \mathbf{L}_{ij}^k \mathbf{B}_j^k, \forall j \in \mathcal{N} \setminus \{i\}.
\end{aligned}$$

Let matrix $\mathbf{M}_i^k(\beta^k, \theta_i^l) \in \mathbb{R}^{m_i \times \sum_{r=1}^N m_r N_r}, l \in \{1, 2, \cdots, N_i\}, i \in \mathcal{N}, k \in \{0, \cdots, K-1\}$, be the truncated row block, i.e., from the row $\sum_{r=1}^{i-1} m_r N_r + m_i(l-1)$ to $\sum_{r=1}^{i-1} m_r N_r + m_i l$, of matrix $(-\mathbf{W}^{0,k}(\beta^k))^{-1}$. We define the following shorthand notations $\Psi_i^{1,k}(\beta^k, \theta_i) := \mathbf{M}_i^k(\beta^k, \theta_i)\mathbf{W}^{1,k}(\beta^k)$ and $\Psi_i^{2,k}(\beta^k, \theta_i) := \mathbf{M}_i^k(\beta^k, \theta_i)\mathbf{W}^{2,k}(\beta^k)$.

6.2.1 Extrinsic Belief Dynamics and the Extended Riccati Equations

In this section, we focus on the extrinsic belief dynamics where Γ_i^k is independent of players' actions u^k for all $i \in \mathcal{N}, k \in \{0, \cdots, K-1\}$. The proof of Theorem 5 generalizes the one of classical LQ games (e.g., Chapter 5.5 and 6.2 in [14]) where we further incorporate players' asymmetric belief dynamics into their objective functions to minimize their expected costs under deception. We apply *dynamic programming* from stage $K-1$ backward to stage 0 to obtain a closed-form solution of PBNE.

Theorem 5. *An N-player K-stage LQ game of incomplete information defined by (6.6), (6.7), and extrinsic belief dynamics $\beta_i^{k+1} = \Gamma_i^k(\beta_i^k, w^k, \theta_i), \forall i \in \mathcal{N}, \forall k \in \{0, \cdots, K-1\}$, admits a unique state-feedback PBNE*

$$u_i^{*,k}(\beta^k, x^k, \theta_i) = \Psi_i^{1,k}(\beta^k, \theta_i)x^k + \Psi_i^{2,k}(\beta^k, \theta_i), \tag{6.13}$$

if and only if $R_i^k(\beta^k, \theta_i)$ is positive definite and $\mathbf{W}^{0,k}(\beta^k)$ is non-singular for all $\beta^k \in \Lambda, i \in \mathcal{N}, \theta_i \in \Theta_i, k \in \{0, \cdots, K-1\}$. The equilibrium cost V_i^k is quadratic *in* x^k, *i.e.,*

$$V_i^k(\beta^k, x^k, \theta_i) = q_i^k(\beta^k, \theta_i) + (x^k)' N_i^k(\beta^k, \theta_i)$$
$$+ (x^k)' S_i^k(\beta^k, \theta_i) x^k, \forall i \in \mathcal{N}, k \in \mathcal{K}. \tag{6.14}$$

Proof. We use backward induction to prove the result. At the final stage K, the value function $V_i^K(\beta^K, x^K, \theta_i) = (x^K - \hat{x}_i^K(\theta_i))' D_i^K(\theta_i)(x^K - \hat{x}_i^K(\theta_i)) + \hat{f}_i^K(\hat{x}_i^K(\theta_i))$ is quadratic in x^K and we obtain the boundary conditions for S_i^K, N_i^K, q_i^K in (6.12) by matching the Right-Hand Side (RHS) of (6.14). At any stage $k \in \{0, \cdots, K-1\}$, if (6.14) is true at stage $k+1$, we can expand $\mathbb{E}_{w^k \sim \Xi_w}[V_i^{k+1}(\beta^{k+1}, x^{k+1}, \theta_i)]$ by plugging in the state dynamics $x^{k+1} = A^k(\theta) x^k + \sum_{i=1}^N B_i^k(\theta_i) u_i^k + w^k$ and the belief dynamics $\beta_i^{k+1} = \Gamma_i^k(\beta_i^k, w^k, \theta_i)$. Then, the RHS of (6.5) is quadratic in u_i^k for each player i. If the coefficient matrix R_i^k of the quadratic form $(u_i^k)' R_i^k u_i^k$ is positive definite, then the first-order necessary conditions for minimization are also sufficient and we obtain the following unique set of equations for the equilibrium action $u^{*,k}$ by differentiating the RHS of (6.5) and setting it to zero, i.e., $\forall \theta_i \in \Theta_i$,

$$-R_i^k u_i^{*,k}(\theta_i) = (B_i^k)' S_i^{k+1} \mathbb{E}_{\theta_{-i} \sim l_i^k}[A^k] x^k + \frac{1}{2}(B_i^k)' N_i^{k+1}$$
$$+ (B_i^k)' S_i^{k+1} \sum_{j \neq i} \mathbb{E}_{\theta_j \sim l_i^k}[B_j^k(\theta_j) u_j^{*,k}(\theta_j)], \forall i \in \mathcal{N}. \tag{6.15}$$

Due to the coupling in players' actions and beliefs, we rewrite (6.15) in matrix form, i.e., $-\mathbf{W}^{0,k}(\beta^k)\mathbf{u}^{*,k} = \mathbf{W}^{1,k}(\beta^k) x^k + \mathbf{W}^{2,k}(\beta^k)$, to solve the set of equations. Given the existence of $(-\mathbf{W}^{0,k}(\beta^k))^{-1}$, each player i's equilibrium action is an affine function in x^k, i.e., $u_i^{*,k}(\beta^k, x^k, \theta_i) = \Psi_i^{1,k}(\beta^k, \theta_i) x^k + \Psi_i^{2,k}(\beta^k, \theta_i)$. Note that the coefficients $\Psi_i^{1,k}, \Psi_i^{2,k}$ for player i are functions of β^k, i.e., the beliefs of all players under all types at stage k. Finally, after substituting the equilibrium action

$u_i^{*,k}(\beta^k, x^k, \theta_i) = \Psi_i^{1,k}(\beta^k, \theta_i)x^k + \Psi_i^{2,k}(\beta^k, \theta_i)$ into the RHS of (6.5) and representing V_i^k in the Left-Hand Side (LHS) in its quadratic form of x^k, we can match the coefficients of quadratic, linear, and constant terms in the LHS and RHS to obtain the extended Riccati equations (6.9), (6.10), and (6.11). □

Remark 9 (**Positive Definiteness**). *If $D_i^k(\theta_i)$ and $F_{ij}^k(\theta_i), \forall j \in \mathcal{N}$, are positive definite for all $k \in \mathcal{K}$, then $R_i^k(\beta^k, \theta_i)$ is positive definite for all $k \in \mathcal{K}, \beta^k \in \Lambda$, because the linear combination of positive definite matrices in* (6.9) *preserves positive definiteness. Note that the above condition is only a necessary condition; i.e., D_i^k and F_{ij}^k do not need to be positive definite to make R_i^k positive definite as shown in Section 6.3.*

Remark 10 (**Cognitive Coupling**). *Compared with the classical LQ games (e.g., Chapter* 6 *in [14]), the deception of players' types results in a unique feature of cognitive coupling represented by the belief matrix in* (6.8)*; i.e., each player's action hinges on not only his own belief but also all other players' beliefs as these beliefs can affect their actions and further the outcome of the interaction. Thus, player i can change other players' actions by manipulating their beliefs of his type θ_i, i.e., $l_j^k, \forall j \in \mathcal{N} \setminus \{i\}$, or making them believe that his belief l_i^k on their types θ_{-i} has changed.*

We introduce matrix block partitions as follows. For each type $\theta_i \in \Theta_i$, we divide $A^k(\theta), D_i^k(\theta_i), S_i^k(\theta_i)$ into N-by-N blocks where the (i,i) block is denoted as $A_i^k(\theta), \bar{D}_i^k(\theta_i), \bar{S}_i^k(\theta_i) \in \mathbb{R}^{n_i \times n_i}$, respectively. The *i-th* row block of $N_i^k(\theta_i), \hat{x}_i^k(\theta_i)$ is $\bar{N}_i^k(\theta_i), \bar{x}_i^k(\theta_i) \in \mathbb{R}^{n_i \times 1}$, respectively. The *i-th* row block of $B_i^k(\theta_i)$ is $\bar{B}_i^k(\theta_i) \in \mathbb{R}^{n_i \times m_i}$. When the system state x^k can be represented by players' joint states $[x_i^k]_{i \in \mathcal{N}}$, Corollary 1 shows that the LQ game of asymmetric information degenerates to an

LQ control problem if players have decoupled cost and state dynamics defined as follows.

Definition 12 (**Decoupled Dynamics and Cost**). *Player $i \in \mathcal{N}$ has decoupled dynamics if for all $k \in \mathcal{K}$, $A_i^k(\theta) = \bar{A}_i^k(\theta_i), \forall \theta \in \Theta$, while all other elements in the i-th row block and the i-th column block of $A^k(\theta)$ are 0. Besides, all elements of $B_i^k(\theta_i)$ except for the row block $\bar{B}_i^k(\theta_i)$ are required to be 0. Player $i \in \mathcal{N}$ has a decoupled cost if for all stage $k \in \mathcal{K}$, $F_{ij}^k(\theta_i) = \mathbf{0}_{m_i,m_i}, \forall \theta_i \in \Theta_i, j \in \mathcal{N} \setminus \{i\}$, and all elements of $D_i^k(\theta_i)$ equal 0 except for $\bar{D}_i^k(\theta_i)$.*

Corollary 1 (**Degeneration to LQ Control**). *If $x^k = [x_i^k]_{i \in \mathcal{N}}$ for all stage $k \in \mathcal{K}$ and player i has both decoupled cost and state dynamics, then his action under PBNE is independent of other players' actions, types, and beliefs, i.e., $u_i^{*,k} = -(R_i^k)^{-1}(\bar{B}_i^k)'\bar{S}_i^{k+1}A_i^k x_i^k - \frac{1}{2}(R_i^k)^{-1}(\bar{B}_i^k)'\bar{N}_i^{k+1}$, where we have $R_i^k = F_{ii}^k + (\bar{B}_i^k)'\bar{S}_i^{k+1}\bar{B}_i^k$, $(G_i^k)' = \mathbf{I}_n - \bar{S}_i^{k+1}\bar{B}_i^k(R_i^k)^{-1}(\bar{B}_i^k)'$, $\bar{S}_i^k = (A_i^k)'(G_i^k)'\bar{S}_i^{k+1}A_i^k + \bar{D}_i^k$, and $\bar{N}_i^k = (A_i^k)'(G_i^k)'\bar{N}_i^{k+1} - 2\bar{D}_i^k\bar{x}_i^k$.*

Proof. We show by induction that $S_i^k, N_i^k, \forall k \in \mathcal{K}$, satisfy the *sparsity condition* that only the (i,i) block of S_i^k and the *i-th* row block of N_i^k are nonzero. At stage K, $S_i^K = D_i^K$ and $N_i^K = -2D_i^K \hat{x}_i^K$ satisfy the above condition. At stage $k \in \{0, \cdots, K-1\}$, if S_i^{k+1}, N_i^{k+1} satisfy the sparsity condition, $\mathbf{W}^{0,k}(\beta^k)$ becomes a diagonal block matrix where $W_{ij}^{0,k}(\beta^k) = \mathbf{0}_{m_i N_i, m_j N_j}$ and $\mathbf{M}_i^k(\beta^k, \theta_i) = -(R_i^k(\beta^k, \theta_i))^{-1}$ for all $\beta^k \in \Lambda$. Then, S_i^k, N_i^k satisfy the condition based on (6.9) and (6.10). □

6.2.2 Intrinsic Belief Dynamics and the Receding-Horizon Control

If there exists a player $i \in \mathcal{N}$ whose belief dynamics Γ_i^k depend on intrinsic information at some stage $k \in \{0, \cdots, K-1\}$ as shown in (6.2), then the equilibrium action $u_i^{*,k}$ is in general a nonlinear function of x^k and the equilibrium cost V_i^k is not quadratic in x^k even under the LQ setting of (6.6) and (6.7). Besides the *static cognitive coupling* among N players in Remark 10, the intrinsic information of u^k in the belief update introduces another *dynamic cognitive coupling* between the forward belief dynamics via (6.2) and the backward equilibrium computation via (6.5), which makes it challenging to compute PBNE. To reduce the computational complexity and further obtain implementable actions, we adopt a receding-horizon approach that computes the sequentially rational action sequence of all the future stages $u^{*,k:K-1}$ at current stage $k \in \{0, \cdots, K-1\}$ assuming $\beta^{\bar{k}} = \beta^k, \forall \bar{k} \in \{k, ..., K-1\}$, yet only implements the current-stage action $u^{*,k}$. Then, at the new stage $k+1$, each player observes the new system state x^{k+1} and updates the belief to β^{k+1} and recomputes the entire action sequence $u^{*,k+1:K-1}$ under assumption of $\beta^{\bar{k}} = \beta^{k+1}, \forall \bar{k} \in \{k+1, ..., K-1\}$, yet still only implements the new current-stage action $u^{*,k+1}$. Players repeat the above procedure until they reach the final stage of the interaction.

Compared with PBNE, which produces an offline planning for all future stages under all possible scenarios before the game has taken place, the receding-horizon approach enables an online replanning of their actions repeatedly at the beginning of each new stage as the interaction continues. Although we assume that players' beliefs at the future stages are the same as the current beliefs during the phase of equilibrium computation, players can correct and update their beliefs and actions based on the online observation of x^k during each replanning phase. Thus, the receding-horizon approach provides a reasonable approximation of the PBNE action and is more adaptive to unexpected environmental changes of the state dynamics f^k and cost structure $g_i^k, \forall i \in \mathcal{N}$.

Under the LQ specification in (6.6) and (6.7) and Bayesian belief dynamics in (6.3), we summarize the computation phase and online implementation phase in Algorithm 5 and 6, respectively. To investigate the scalability of our algorithms, we analyze the temporal and spatial complexity concerning N, K, and N_i. To simplify the notation and enhance readability, we focus on the symmetric setting where $N_i = N_0 \in \mathbb{Z}^+, \forall i \in \mathcal{N}$. For each player $i \in \mathcal{N}$ of type $\theta_i \in \Theta_i$ at the beginning of the interaction, i.e., $k = 0$, he needs to store the game parameters $A^0, B_r^0(\theta_r), D_r^0(\theta_r), F_{rh}^0(\theta_r), \forall \theta_r \in \Theta_r$, and the belief matrix $\mathbf{L}_{rh}^0$ for all $r, h \in \mathcal{N}$, which are *common knowledge*. The spatial complexity to store the game parameters and the belief matrix is $O(N^2 N_0)$ and $O(N^2 N_0^2)$, respectively. Note that in general, player i has coupled cognition as shown in Remark 10 and has to keep track of not only his belief $\mathbf{L}_{i,j}^k, \forall j \in \mathcal{N}$, but also other players' beliefs $\mathbf{L}_{r,h}^k, \forall r \in \mathcal{N} \setminus \{i\}, h \in \mathcal{N}$, to decide his equilibrium action under deception at each stage k. During the K-stage interaction, each player $i \in \mathcal{N}$ of type $\theta_i \in \Theta_i$ observes the system state x^k and computes his equilibrium action $u_i^{*,k}(\beta^k, x^k, \theta_i)$ at stage k based on Algorithm 5. After all players implement their equilibrium actions at stage k, the system state evolves to x^{k+1}. Based on the new state observation x^{k+1}, each player i updates the belief matrix in (6.8) via (6.3). Since player i can delete the game parameters and the belief matrices of previous stages, the spatial complexity remains the same as the real-time stage index k increases. Thus, our algorithm can handle the interaction of long duration. All players repeat the above procedure stated in lines 14-17 of Algorithm 6 until reaching the terminal stage $k = K$.

Algorithm 5: PBNE computation with $\beta^{\bar{k}} = \beta^k, \forall \bar{k} \in \{k, ..., K-1\}$ at stage $k \in \{0, \cdots, K-1\}$ for player $i \in \mathcal{N}$ of type $\theta_i \in \Theta_i$

```
Load game parameters $A^k, B_r^k(\bar{\theta}_r), D_r^k(\bar{\theta}_r), F_{rh}^k(\bar{\theta}_r), \forall \bar{\theta}_r \in \Theta_r$ and the belief
  matrix $\mathbf{L}_{r,h}^k$ for all $r, h \in \mathcal{N}$;
Input state observation $x^k$;
for $\bar{k} \leftarrow K-1$ to $k$ do
    for $j \leftarrow 1$ to $N$ do
        for $\theta_j \leftarrow \theta_j^1$ to $\theta_j^{N_j}$ do
            Compute $S_j^{\bar{k}}, N_j^{\bar{k}}$ via (6.9), (6.10) with $\beta^{\bar{k}} = \beta^k$;
        end
    end
end
Return his equilibrium action $u_i^{*,k}(l_i^k, x^k, \theta_i)$ via (6.13);
```

The computational complexity of the belief matrix update in the line 15 of Algorithm 6 is $O(N_0^N N)$. For any β^k, the term $\mathbf{W}^{0,k}(\beta^k)$ has computational complexity $O(N_0^N N) + O(N_0^3 N^2)$, which is determined by the belief matrix update and the matrix chain multiplication of $W_{ij}^{0,k}(\beta^k)$, respectively. Then, the computational complexity of $(\mathbf{W}^{0,k}(\beta^k))^{-1}$ and $\mathbf{W}^{1,k}(\beta^k)$ is $O(N_0^N N) + O(N_0^3 N^3)$ and $O(N_0^N N) + O(N_0^3 N^2)$, respectively. Given β^k and θ_i, the computational complexity of $S_i^k(\beta^k, \theta_i)$ in (6.9) is $O(N_0^N N) + O(N_0^3 N^3) + O(N_0^3 N^2) + O(N_0 N) = O(\max(N_0^N N, N_0^3 N^3))$, which hinges on the computational complexity of the matrix $\mathbf{M}_i^k(\beta^k, \theta_i)$ (or $(\mathbf{W}^{0,k}(\beta^k))^{-1}$), $\mathbf{W}^{1,k}(\beta^k)$, and the matrix chain multiplication in (6.9). Similarly, $N_i^k(\beta^k, \theta_i)$ and $W^{2,k}(\beta^k)$ both have computational complexity of $O(N_0^N N) + O(N_0 N)$. Therefore, player i's temporal complexity at each stage

Algorithm 6: K-stage receding-horizon control for player $i \in \mathcal{N}$ of type $\theta_i \in \Theta_i$

```
Initialize $k = 0$;
Store game parameters $A^k, B_r^k(\bar{\theta}_r), D_r^k(\bar{\theta}_r), F_{rh}^k(\bar{\theta}_r), \forall \bar{\theta}_r \in \Theta_r$ and the belief
  matrix $\mathbf{L}_{r,h}^k$ for all $r, h \in \mathcal{N}$;
while $k < K$ do
    Call Algorithm 5 to implement $u_i^{*,k}(l_i^k, x^k, \theta_i)$;
    Observe state $x^{k+1}$ and update all elements of the belief matrix via (6.3)
      to obtain $\mathbf{L}_{r,h}^{k+1}, \forall r, h \in \mathcal{N}$;
    Delete $A^k, B_r^k(\bar{\theta}_r), D_r^k(\bar{\theta}_r), F_{rh}^k(\bar{\theta}_r), \mathbf{L}_{r,h}^k$ and Store
      $A^{k+1}, B_r^{k+1}(\bar{\theta}_r), D_r^{k+1}(\bar{\theta}_r), F_{rh}^{k+1}(\bar{\theta}_r), \mathbf{L}_{r,h}^{k+1}$ for all $\bar{\theta}_r \in \Theta_r$ and for all
      $r, h \in \mathcal{N}$;
    Update stage index $k \leftarrow k + 1$;
end
```

$k \in \{0, 1, \cdots, K-1\}$ is

$$O((K-k) \cdot N_0 N \cdot \max(N_0^N N, N_0^3 N^3)).$$

The temporal complexity has the maximum value of $O(K \cdot \max\{N_0^{N+1} N^2, N_0^4 N^4\})$ at the initial stage $k = 0$ where each player has to predict the entire K future stages to act optimally under the deception. Since the temporal complexity decreases as the real-time stage index k increases, a player who can compute the equilibrium action within the required time at the initial stage $k = 0$ is guaranteed to meet the real-time requirement in the following stages of interaction. If the number of types and agents are on the same scale, e.g., $N_0 = N$, then $\lim_{N \to \infty}(N_0^{N+1} N^2)/(N_0^4 N^4) \to \infty$ and the computation of belief matrix update plays a dominant role as each player keeps track of all players' beliefs to obtain the equilibrium action under deception. If $N_0 \ll N$, e.g., $N_0 = N^{1/N}$, then $\lim_{N \to \infty}(N_0^{N+1} N^2)/(N_0^4 N^4) \to 0$ and the inverse of $\mathbf{W}^{0,k}(\beta^k)$ becomes the most time-consuming operation due to the coupling in dynamics, costs, and cognition.

Effective deception can prevent or delay other players from learning the deceiver's private type. We define the criterion of successful learning of the deceiver's type in Definition 13 and ϵ-deceviability and ϵ-learnability in Definition 14.

Definition 13 (**Stage of Truth Revelation**)**.** *Consider two players $i, j \in \mathcal{N}$ with type θ_i and θ_j, respectively. Stage $k_{i,j}^{tr} \in \mathcal{K} \cup \{K+1\}$ is said to be player i's truth-revealing stage with accuracy $\delta \in (0,1]$[2] if it satisfies the following two conditions.*

- ***The bounded mismatch condition**: player i's belief mismatch remains less than δ after stage $k_{i,j}^{tr} \in \mathcal{K}$, i.e.,*

$$1 - l_i^k(\theta_j | h^k, \theta_i) \leq \delta, \forall k \geq k_{i,j}^{tr}. \tag{6.16}$$

- ***The first-hitting-time condition**: $k_{i,j}^{tr} \in \mathcal{K}$ is the first stage satisfying (6.16), i.e., $1 - l_i^{k_{i,j}^{tr}-1}(\theta_j | h^{k_{i,j}^{tr}-1}, \theta_i) > \delta, k_{i,j}^{tr} > 1$.*

If there does not exist $k_{i,j}^{tr} \in \mathcal{K}$ that satisfies (6.16), we define $k_{i,j}^{tr} := K+1$. If there are only two players $N = 2$, we write $k_{i,j}^{tr}$ as k_i^{tr} without ambiguity.

Due to deceivers' deceptive actions and the external noises, the belief sequence may be fluctuant; i.e., there can exist $k < k_{i,j}^{tr}$ such that $1 - l_i^k(\theta_j | h^k, \theta_i) \leq \delta$. Thus, as shown in Definition 13, a player should only claim a successful learning of other players' types if his belief mismatch remains less than δ for the remaining stages.

Definition 14 (**Deceviability and Learnability**)**.** *Consider players $i, j \in \mathcal{N}$ with type θ_i and θ_j, thresholds $\delta \in (0,1], \epsilon \in [0,1]$, and a given stage index $\tilde{k} \in \mathcal{K} \cup \{K+1\}$.*

[2] Since the belief mismatch does not reduce to 0 in finite stages with initial belief $l_i^0 \in (0,1)$, the accuracy threshold $\delta \neq 0$.

Player i is $\tilde{k}$-stage ϵ-deceivable if the probability $\Pr(k^{tr}_{i,j} < \tilde{k})$*, or equivalently* $\Pr(l^{\tilde{k}}_i(\theta_j|x^{\tilde{k}}, \theta_i) > 1-\delta)$*, is not greater than ϵ for all $l^0_i \in (0,1)$. If the above does not hold, player j's type is said to be $\tilde{k}$-stage ϵ-learnable by player i.*

Since robot deception involves only a finite number of stages, it is essential that the deceived robot can learn the deceiver's type as quickly as possible so that he has sufficient stages to plan on and mitigate the deception impact from the previous stages. Therefore, the definition of learnability, i.e., non-deceviability in Definition 14, not only requires the deceived player to be capable of learning the deceiver's private information, but also learning it in a desirable rate, i.e., within $\tilde{k}$ stage. Due to the external noise, $k^{tr}_{i,j}$ is a random variable. Thus, the definition of learnability requires $\Pr(k^{tr}_{i,j} < \tilde{k}) > \epsilon$; i.e., player i has a large probability to correctly learn the type of player j before stage $\tilde{k}$.

6.3 Dynamic Target Protection under Deception

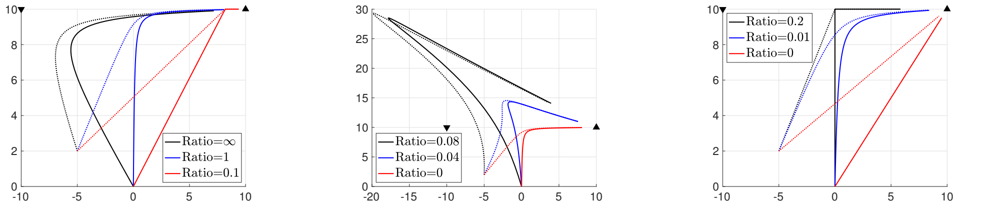
We investigate a pursuit-evasion scenario that contains two UAVs with the decoupled linear time-invariant state dynamics, i.e., $A^k(\theta) = \mathbf{I}_4$ and $\bar{B}^k_i(\theta_i) = [\tilde{B}_i(\theta_i), 0; 0, \tilde{B}_i(\theta_i)] \in \mathbb{R}^{2\times 2}, \forall k \in \mathcal{K}$. We use 'she' for UAV 1, the pursuer, and 'he' for UAV 2, the evader. UAV i's state $x^k_i := [x^k_{i,x}, x^k_{i,y}]' \in \mathbb{R}^{2\times 1}$ represents i's location $(x^k_{i,x}, x^k_{i,y})$ in the $2D$ space, and action $u^k_i = [u^k_{i,x}, u^k_{i,y}] \in \mathbb{R}^{2\times 1}$ affects i's speed in x and y directions.

UAV 2 as the evader selects either the harbor in 'Normandy' or 'Calais' as his final location based on his type $\theta_2 \in \{\theta^g_2, \theta^b_2\}$. He aims to reach 'Normandy' located at $\gamma(\theta^g_2) := (x^g, y^g)$ in $K = 40$ stages if his type is θ^g_2, otherwise 'Calais' located at $\gamma(\theta^b_2) := (x^b, y^b)$ if his type is θ^b_2. UAV 1 as the pursuer can make interfering signals and aims to be close to UAV 2 at the final stage to protect the harbor targeted by the evader, i.e., $g_1^k(x^k, u^k, \theta_1) = d_{12}^k(\theta_1)((x_{2,y}^k - x_{1,y}^k)^2 + (x_{2,x}^k - x_{1,x}^k)^2) + f_{11}^k(\theta_1)((u_{1,x}^k)^2 + (u_{1,y}^k)^2) - f_{12}^k(\theta_1)((u_{2,x}^k)^2 + (u_{2,y}^k)^2), \forall k \in \mathcal{K}$, where $d_{12}^k(\theta_1) \in \mathbb{R}_{\geq 0}$ penalizes her distance from the evader at stage $k \in \mathcal{K}$, $f_{11}^k(\theta_1) \in \mathbb{R}_{\geq 0}$ prevents her from a high action cost, and $f_{12}^k(\theta_1) \in \mathbb{R}_{\geq 0}$ incites her opponent, i.e., the evader, to take costly actions. We classify UAV 1 into two types, i.e., $\Theta_1 = \{\theta_1^H, \theta_1^L\}$, based on her maneuverability represented by the value of $\tilde{B}_1(\theta_1)$. Given higher maneuverability $\tilde{B}_1(\theta_1^H) > \tilde{B}_1(\theta_1^L)$, the pursuer of type θ_1^H can obtain a higher speed under the same action u_1^k and thus cover a longer distance.

The evader's goals of deceptive target reaching and pursuit evasion are incorporated into the cost structure $g_2^k(x^k, u^k, \theta_2) = d_{2,b}^k(\theta_2)((x_{2,y}^k - y^b)^2 + (x_{2,x}^k - x^b)^2) + d_{2,g}^k(\theta_2)((x_{2,y}^k - y^g)^2 + (x_{2,x}^k - x^g)^2) - d_{21}^k(\theta_2)((x_{1,y}^k - x_{2,y}^k)^2 + (x_{1,x}^k - x_{2,x}^k)^2) + f_{22}^k(\theta_2)((u_{2,x}^k)^2 + (u_{2,y}^k)^2) - f_{21}^k(\theta_2)((u_{1,x}^k)^2 + (u_{1,y}^k)^2), \forall k \in \mathcal{K}$. Similar to the pursuer's cost parameters, $d_{21}^k(\theta_2) \in \mathbb{R}_{\geq 0}$ represents the evader's level of *evasion determination* to keep a distance from the pursuer along the trajectory. The action costs of the evader and the pursuer are regulated by $f_{22}^k(\theta_2) \in \mathbb{R}_{\geq 0}$ and $f_{21}^k(\theta_2) \in \mathbb{R}_{\geq 0}$, respectively. The parameters $d_{2,b}^k(\theta_2)$ and $d_{2,g}^k(\theta_2)$ represent the evader's attempt to head toward 'Normandy' and 'Calais', respectively, at stage $k \in \mathcal{K}$ under type $\theta_2 \in \Theta_2$. We use the ratio $d_{2,g}^k(\theta_2)/d_{2,b}^k(\theta_2)$ to represent the evader's level of *trajectory deception*. Since the pursuer can learn the evader's type based on the real-time observations of state x_2^k, the evader attempts to make his target ϵ_0-ambiguous at all previous stages, i.e., $|d_{2,b}^k(\theta_2)/d_{2,g}^k(\theta_2) - 1| \leq \epsilon_0, \forall \theta_2, \forall k \neq K$, and reveal his true target only at the final stage, i.e., $d_{2,g}^K(\theta_2^b) = 0$ and $d_{2,b}^K(\theta_2^g) = 0$. The evader chooses a small $\epsilon_0 \geq 0$ and achieves the maximum ambiguity when $\epsilon_0 = 0$. Two blue lines in Fig. 6.1a illustrate how the evader manages to remain ambiguous in a cost-effective manner from two different initial locations. Instead of keeping an equal distance to both potential targets, the evader heads toward the midpoint $((x^g + x^b)/2, (y^g + y^b)/2)$ at the early stages to confuse the pursuer. However, the evader starts to head toward the true target at around half of K stages rather than the last few stages so that he can reach the target with a moderate control cost $(u_2^k)' F_{22}^k(\theta_2) u_2^k$. Fig. 6.1a also shows that for a given initial location, the evader who adopts a higher level of *trajectory deception* heads more toward the misleading target at the early stages.

In this case study, we suppose that the evader's true target is Calais and let θ_2^b be his *true type* and θ_2^g be the *misleading type*. The following two ratios capture the evader's tradeoff of being deceptive, effective, and evasive. On one hand, the ratio $d_{2,b}^k(\theta_2^b)/d_{2,b}^K(\theta_2^b), k \neq K$, reflects the evader's tradeoff between applying deception along the trajectory and staying close to the true target at the final stage. Fig. 6.1b shows that as the evader focuses more on a deceptive trajectory represented by a larger value of $d_{2,b}^k(\theta_2^b)/d_{2,b}^K(\theta_2^b), k \neq K$, his trajectory remains ambiguous for longer stages while his final location is farther away from the true target. On the other hand, the ratio $d_{21}^k(\theta_2^b)/d_{2,b}^K(\theta_2^b), k \neq K$, reflects the evader's tradeoff between evasion and target-reaching. As the evader focuses more on keeping a distance from the pursuer along the trajectory, he takes a bigger detour and stays farther away from his true target at the final stage as shown in Fig. 6.1c.

Finally, we transform UAV i's coupled cost g_i^k into the matrix form given in Section 6.2, i.e., $\hat{x}_1^k(\theta_1) = \mathbf{0}_{4,1}, \hat{f}_1^k(\hat{x}_1^k(\theta_1)) = 0, F_{ii}^k(\theta_1) = f_{ii}^k(\theta_1) \cdot \mathbf{I}_2, F_{ij}^k(\theta_1) =$



(a) Ratio represents $d^k_{2,g}(\theta^b_2)/d^k_{2,b}(\theta^b_2)$.

(b) Ratio represents $d^k_{2,b}(\theta^b_2)/d^K_{2,b}(\theta^b_2)$.

(c) Ratio represents $d^k_{21}(\theta^b_2)/d^K_{2,b}(\theta^b_2)$.

Figure 6.1: The evader's trajectories from $x^0_2 = [0,0]$ and $x^0_2 = [-5,2]$ in solid and the dashed lines, respectively. The black downward and upward triangles represent the location of Calais $(x^b, y^b) = (-10, 10)$ and Normandy $(x^g, y^g) = (10, 10)$, respectively. The ratios capture the evader's tradeoff of forming a deceptive trajectory, reaching the true target, and evading the pursuit.

$-f^k_{ij}(\theta_1)\cdot \mathbf{I}_2, j \neq i, D^k_1(\theta_1) = d^k_{12}(\theta_1)\cdot [1,0,-1,0;0,1,0,-1;-1,0,1,0;0,-1,0,1]$,

$$D^k_2(\theta_2) = \begin{bmatrix} -d^k_{21} & 0 & d^k_{21} & 0 \\ 0 & -d^k_{21} & 0 & d^k_{21} \\ d^k_{21} & 0 & d^k_{2,b}+d^k_{2,g}-d^k_{21} & 0 \\ 0 & d^k_{21} & 0 & d^k_{2,b}+d^k_{2,g}-d^k_{21} \end{bmatrix},$$

$\hat{x}^k_2(\theta_2) = \frac{1}{d^k_{2,b}+d^k_{2,g}} \cdot [d^k_{2,b}x^b + d^k_{2,g}x^g \ ; \ d^k_{2,b}y^b + d^k_{2,g}y^g \ ; \ d^k_{2,b}x^b + d^k_{2,g}x^g \ ; \ d^k_{2,b}y^b + d^k_{2,g}y^g]$,
$\hat{f}^k_2(\hat{x}^k_2(\theta_2)) = \frac{d^k_{2,b}d^k_{2,g}((x^b-x^g)^2+(y^b-y^g)^2)}{d^k_{2,b}+d^k_{2,g}}$.

6.3.1 Deceptive Evader with Decoupled Cost Structure

We first investigate the scenario where the evader has a decoupled cost structure[3] defined in Definition 12, i.e., $d^k_{21}(\theta_2) = 0, \forall \theta_2 \in \Theta_2, \forall k \in \mathcal{K}$. According to Corollary 1, the evader's trajectory is then independent of the pursuer's action, type, and

[3] This paper has supplementary downloadable materials available at http://ieeexplore.ieee.org, provided by the authors. This includes a video demo of two UAVs' trajectories and belief updates under the decoupled structure.

belief. Fig. 6.2 visualizes the pursuer's trajectories. Although the pursuer only aims to be close to the evader at the final stage, she also takes proactive actions in the previous stages to be cost-efficient. If the pursuer knows the evader's type, then she can head toward the true target directly and will not be misled by the evader's trajectory ambiguity at the early stages as illustrated by the black dashed line in Fig. 6.2. If the evader's type is private, then a larger initial belief mismatch $1 - l_1^0(\theta_2^b|x^0, \theta_1^H)$ makes the pursuer head more toward the misleading target at the early stages as illustrated by the three solid lines in Fig. 6.2. However, due to the pursuer's online learning, which is compatible, efficient, and robust as shown in Section 6.3.1, she manages to approach the evader at the final stage regardless of her initial belief mismatch. Fig. 6.3 shows the pursuer's K-stage belief variation. The evader's ambiguous trajectory results in belief fluctuations at the early stages, yet the pursuer can quickly reduce the belief mismatch when the evader starts to head toward the true target. After the pursuer has corrected her initial belief mismatch at around stage $k = 16$, she can head toward the true target in the cost-efficient way; i.e, she attempts to keep a uniform linear motion under the external noise as shown in the upper right region of Fig. 6.2.

Finite-Horizon Analysis of Bayesian Update

In this subsection, we illustrate the compatibility, efficiency, and robustness of the finite-horizon Bayesian update in (6.3) to reduce the initial belief mismatch. The pursuer is of high-maneuverability and the evader's true type is θ_2^b. Define the likelihood function of θ_2^b and θ_2^g as $a^k := \Pr(x^{k+1}|\theta_2^b, x^k, \theta_1^H)$ and $c^k := \Pr(x^{k+1}|\theta_2^g, x^k, \theta_1^H)$, respectively. As $w^k \in \mathbb{R}^{n\times 1}$, a^k and c^k are positive. With an initial belief $l_1^0 \in (0, 1)$ and a finite likelihood ratio $e^k := c^k/a^k \in (0, \infty)$, we can represent (6.3) in the

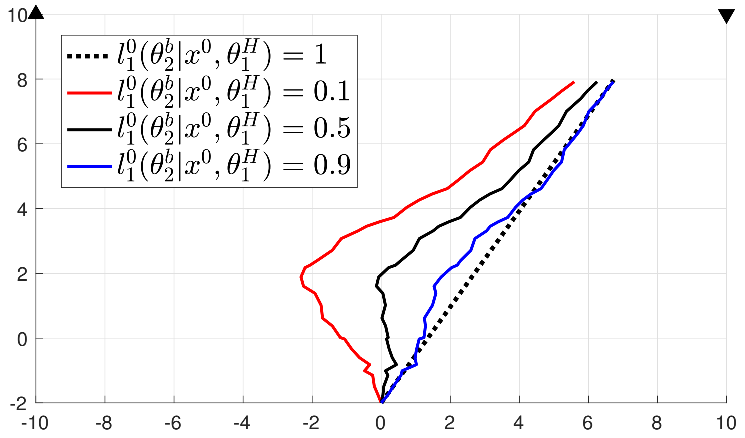


Figure 6.2: The pursuer's trajectories under different initial beliefs.

following form with three properties:

$$l_1^{k+1} = \frac{l_1^k \cdot a^k}{l_1^k \cdot a^k + (1-l_1^k)\cdot c^k} = \frac{1}{1+(\frac{1}{l_1^0}-1)\prod_{\bar{k}=0}^{k} e^{\bar{k}}} \in (0,1).$$

1. (**Compatibility**): For all $l_1^k \in (0,1)$, the belief update at stage k is compatible to the evidence represented by the ratio e^k. In particular, if $e^k < 1$, then $l_1^{k+1} > l_1^k$; if $e^k > 1$, then $l_1^{k+1} < l_1^k$; if $e^k = 1$, then $l_1^{k+1} = l_1^k$.

2. (**Efficiency**): If the evidence of state observation x^{k+1} indicates that the type is more likely to be the true type θ_2^b, i.e., $e^k < 1$, then the function $l_1^{k+1}/l_1^k = 1/(l_1^k + (1-l_1^k)e^k)$ at stage k is monotonically decreasing over l_1^k. If the evidence indicates that the type is more likely to be the misleading type θ_2^g, i.e., $e^k > 1$, then the function l_1^{k+1}/l_1^k is monotonically increasing over l_1^k.

3. (**Robustness**): The order of the evidence sequence $e^{\bar{k}}, \bar{k} = 0, \cdots, k$, has no impact on the belief l_1^{k+1}.

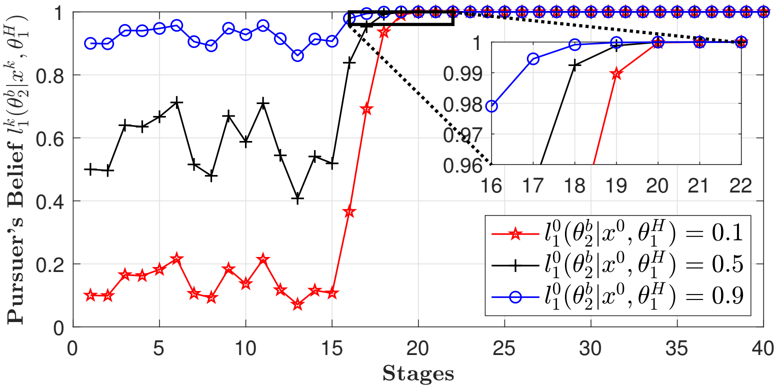


Figure 6.3: The pursuer's belief update over K stages under three different initial beliefs and the same noise sequence $[w^k]_{k\in\mathcal{K}}$. The inset black box magnifies the selected area.

Property one shows that although the external noise can result in the fluctuations of the belief update, the belief mismatch, i.e., $1-l_1^k$, will decrease when $e^k<1$, regardless of the prior belief $l_1^k\in(0,1)$. Property two shows the efficiency of the belief update. The belief changes more under a larger belief mismatch, which results in a quick correction. Property three shows the robustness of the belief update. The erroneous belief update caused by a heavy noise can be corrected in the later stages when the noise fades.

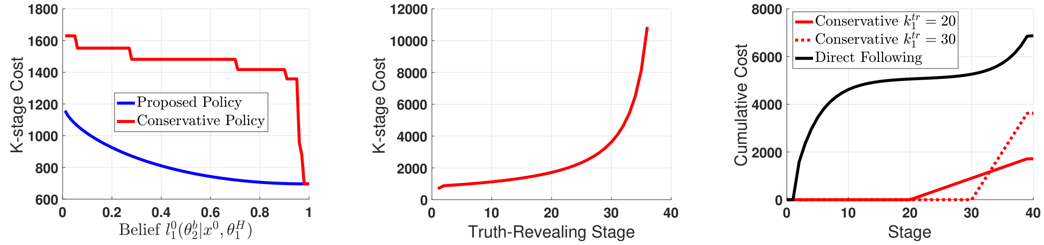
Comparison with Heuristic Policies

We compare the proposed pursuer's control policy with two heuristic ones to demonstrate its efficacy in counter-deception[4]. The first heuristic policy is to repeat the attacker's trajectory with a one-stage delay; i.e., the pursuer applies the action so that $x_1^{k+1}=x_2^k, \forall k\in\mathcal{K}\setminus\{K\}$. The pursuer does not need to apply Bayesian learning and we name this policy as *direct following*. The second heuristic policy for

[4] The supplementary materials include a video demo that compares the proposed policy's trajectory and performance with two heuristic policies.

the pursuer is to stay at the initial location until her truth-revealing stage k_1^{tr} and then head toward the evader's expected final-stage location in the remaining stages. The second policy is *conservative* because the pursuer does not take proactive actions until she identifies the evader's type.

Let player i's *ex-post cumulative cost* $\hat{V}_i^{0:k} := \sum_{h=0}^{k} g_i^h, \forall k \in \mathcal{K}$, be a real-time evaluation of the online algorithm. Although a pursuer under both heuristic policies manages to stay close to the evader at the final stage, Fig. 6.4 shows that both heuristic policies are more costly than the proposed equilibrium strategy in the long run. The conservative policy avoids potential trajectory deviations under deception but results in less planning stages for the pursuer to achieve the capture goal. We visualize the accumulation of the pursuer's cost in Fig. 6.4c. The red lines show that the pursuer who adopts the conservative policy spends no action costs before the truth-revealing stage k_1^{tr}, i.e., $(u_1^k)' F_{11}^k(\theta_1) u_1^k = 0, \forall k \leq k_1^{tr}$, but huge costs in the remaining stages to fulfill her capture goal. The total cumulative cost $\hat{V}_i^{0:K}$ at the final stage increases exponentially with the value of k_1^{tr} as shown in Fig. 6.4b. The black line in Fig. 6.4c illustrates the accumulation of $\hat{V}_i^{0:k}$ when the pursuer direct follows the evader's trajectory. Only under extreme deception scenarios where



(a) The K-stage cumulative cost $\hat{V}_i^{0:K}$ versus different initial beliefs.

(b) The K-stage cumulative cost $\hat{V}_i^{0:K}$ versus k_1^{tr} under *conservative policy*.

(c) The accumulation of the pursuer's cost $\hat{V}_i^{0:k}, \forall k \in \mathcal{K}$.

Figure 6.4: The pursuer's ex-post cumulative cost under two heuristic policies and the proposed policy.

$k_1^{tr} > 34$, the direct following policy results in a lower cost than the conservative policy does. Since the initial belief l_1^0 affects both the truth-revealing stage and the proposed policy, we plot $\hat{V}_i^{0:K}$ versus l_1^0 under the conservative policy and the proposed policy in Fig. 6.4a. When there is no belief mismatch $l_1^0(\theta_2^b|x^0, \theta_1^H) = 1$, we have $k_1^{tr} = 1$ and the conservative policy is equivalent to the proposed policy. As the belief mismatch increases, the cost $\hat{V}_i^{0:K}$ under the proposed policy (resp. the conservative policy) increases due to the larger deviation along the x-axis (resp. the larger k_1^{tr}). The proposed policy always results in a lower cost $\hat{V}_i^{0:K}$ than the conservative policy does. The results in Fig. 6.4 lead to the following two principles for the pursuer to behave under deception. First, Bayesian learning is a more effective countermeasure than the direct following of the evader's deceptive trajectory. Second, if learning the evader's type takes a long time, the pursuer is better to act proactively based on her current belief than to delay actions until the truth-revealing stage.

6.3.2 Dynamic Game for Counter-Deception

In this section, the evader has a coupled cost[5] defined in Definition 12 and the level of *evasion determination* increases with a constant rate $\alpha > 0$; i.e., $d_{21}^k(\theta_2) = \alpha k, \forall \theta_2 \in \Theta_2, \forall k \in \mathcal{K}$. The evader deceives the pursuer by hiding his true target. The pursuer can adopt the following two countermeasures to reduce her cost under the evader's deception. Section 6.3.2 investigates the effectiveness of adaptive learning. We find that the pursuer manages to approach the true target at the final stage by updating her belief and taking actions accordingly based on

[5] A video demo of two UAVs' real-time trajectories and belief updates under the coupled structure is included in the supplementary materials.

the real-time trajectory observation. Section 6.3.2 further allows the pursuer to introduce additional deception, i.e., obfuscate her maneuverability, to counteract the evader's information advantage and his deception impact.

Pursuer with a Public Type

When the pursuer's type is *common knowledge*, we plot both UAVs' trajectories under two initial beliefs and two types of pursuers in Fig. 6.5. The solid lines show

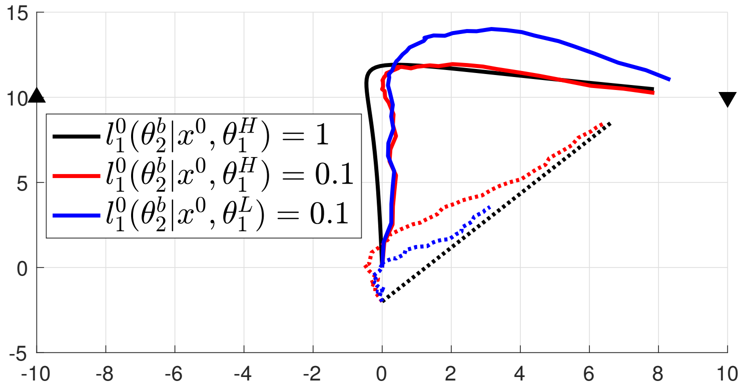
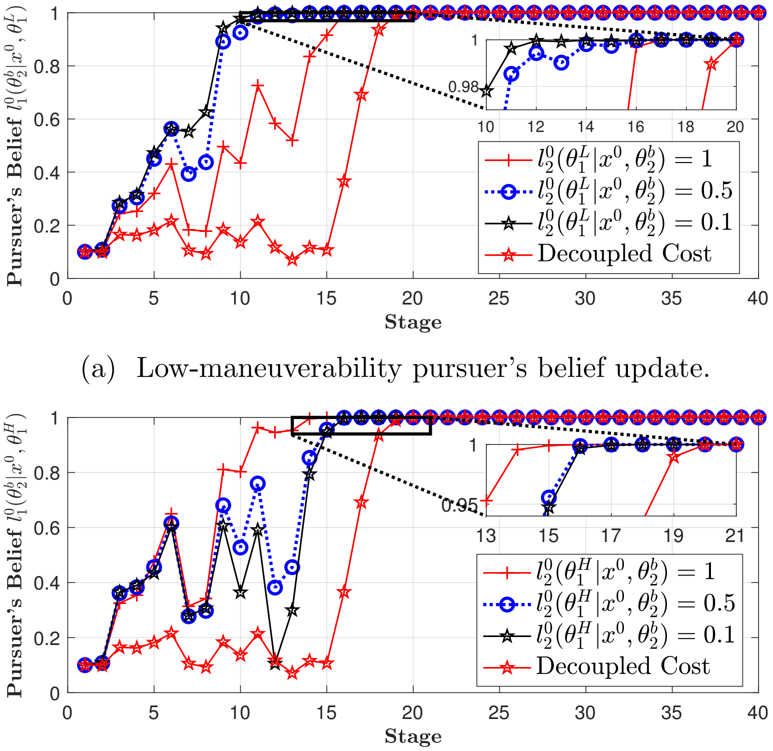


Figure 6.5: The K-stage trajectory of the evader and the pursuer in solid and dashed lines, respectively. If the evader's type is *common knowledge* and the pursuer is of high-maneuverability, we represent their noise-free trajectories in black. If the evader's type is private and the pursuer's initial belief mismatch is 0.9, two UAVs' trajectories are in red (resp. blue) when the pursuer's maneuverability is high (resp. low).

that the evader with the coupled cost detours to stay further from the pursuer. The initial belief mismatch causes a deviation along the x-axis for both high- and low-maneuverability pursuers as shown in red and blue, respectively. However, the deviation has a smaller magnitude and lasts shorter than the one represented by the red line in Fig. 6.2 due to the coupled cost structure of the evader. The pursuer with a high maneuverability stays closer to the evader at the final stage.

Deception to Counteract Deception

When the pursuer's type is also private, Fig. 6.6 shows that she can manipulate the evader's initial belief l_2^0 to obtain a smaller k_1^{tr} and a belief update with less fluctuation. The red line with stars is the same as the one in Fig. 6.3. It shows that the pursuer's belief learning is slower and fluctuates more when she interacts with the evader who has a decoupled cost. The reason is that her manipulation of the initial belief l_2^0 does not affect the evader's decision making as shown in Corollary 1. A comparison between Fig. 6.6a and Fig. 6.6b shows that it is beneficial for a



(a) Low-maneuverability pursuer's belief update.

(b) High-maneuverability pursuer's belief update.

Figure 6.6: The pursuer's belief update over K stages under the same initial belief $l_1^0(\theta_2^b|x^0,\theta_1) = 0.1$. The inset black box magnifies the selected area.

low-maneuverability pursuer to disguise as a high-maneuverability pursuer but not vice versa. Thus, introducing additional deception to counteract existing deception is not always effective.

6.3.3 Multi-Dimensional Deception Metrics

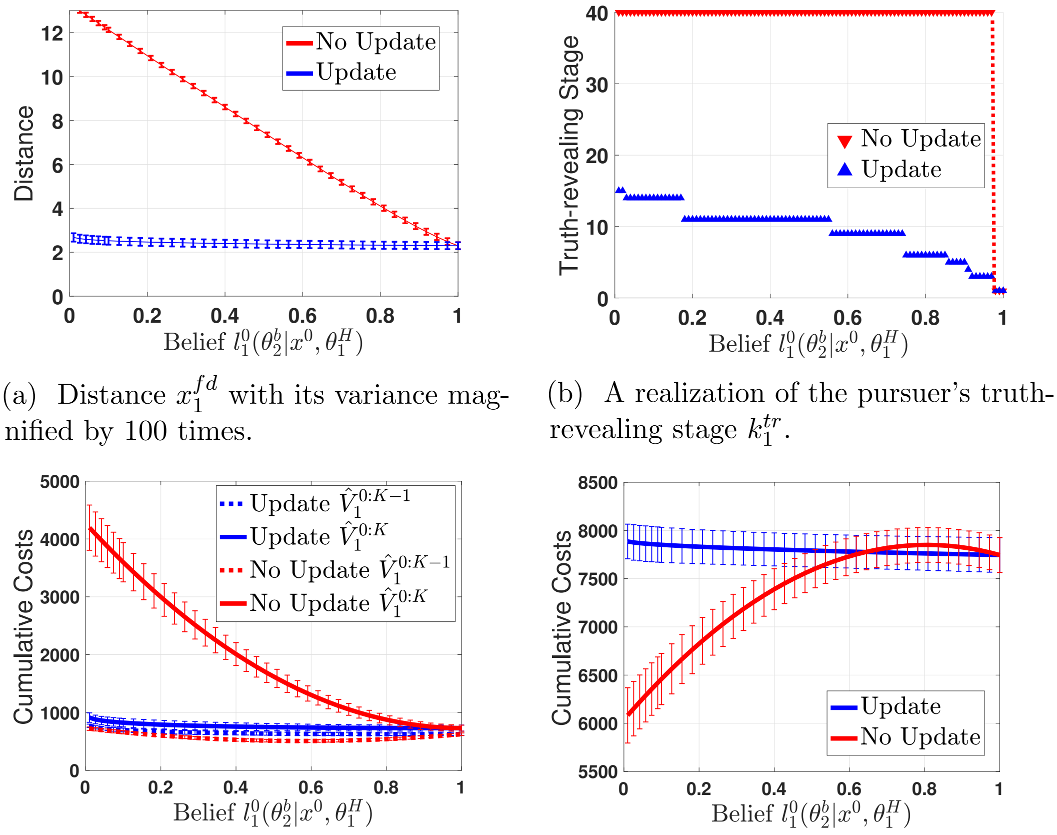
The impact of the evader's deception can be measured by metrics such as the *endpoint distance* $x_2^{fd} := ||x_2^K - \gamma(\theta_2)||_2$ between the evader and the true target, the *endpoint distance* $x_1^{fd} := ||x_2^K - x_1^K||_2$ between two UAVs, both UAVs' truth-revealing stages k_i^{tr}, and their ex-post cumulative costs $\hat{V}_i^{0:k}, \forall k \in \mathcal{K}$. In this pursuit-evasion case study, we define ϵ-reachability and ϵ-capturability in Definition 15. Although $x_i^{fd}, \forall i \in \{1, 2\}$, is a random variable, we can obtain a good estimate of the reachability and capturability due to the negligible variance of x_i^{fd} as shown in Fig. 6.7a and Fig. 6.8a.

Definition 15 (**Reachability and Capturability**). *Consider the proposed scenario of pursuit evasion with a given $\epsilon \geq 0$, a threshold $\bar{x}^{fd} \geq 0$, and all initial beliefs $l_i^0 \in (0, 1)$. The target is said to be ϵ-reachable if* $\Pr(x_2^{fd} \geq \bar{x}^{fd}) \leq \epsilon$. *The evader is said to be ϵ-capturable if* $\Pr(x_1^{fd} \geq \bar{x}^{fd}) \leq \epsilon$.

In Section 6.3.3, we investigate how the evader can manipulate the pursuer's initial belief $l_1^0(\theta_2^b|x^0, \theta_1^H)$ to influence the deception. In Section 6.3.3, we investigate how the pursuer's maneuverability plays a role in deception. In both sections, the evader has a coupled cost structure. The pursuer either applies the Bayesian update or not, which is denoted by blue and red lines, respectively, in both Fig. 6.7 and Fig. 6.8. In Section 6.3.3, we study other metrics, such as deceivability, distinguishability, and PoD.

The Impact of the Evader's Belief Manipulation

Both UAVs determine their initial beliefs based on the intelligence collected before their interactions. By falsifying the pursuer's intelligence, the evader can manipulate the pursuer's initial belief l_1^0 and further influence the deception as shown in Fig. 6.7. In the x-axis, an initial belief $l_1^0(\theta_2^b|x^0, \theta_1^H)$ closer to 1 indicates a smaller belief mismatch. Fig. 6.7a shows that the pursuer's distance to the evader at the final stage decreases as the belief mismatch decreases regardless of



(a) Distance x_1^{fd} with its variance magnified by 100 times.

(b) A realization of the pursuer's truth-revealing stage k_1^{tr}.

(c) The costs $\hat{V}_1^{0:K-1}$ and $\hat{V}_1^{0:K}$ of the pursuer under type θ_1^H.

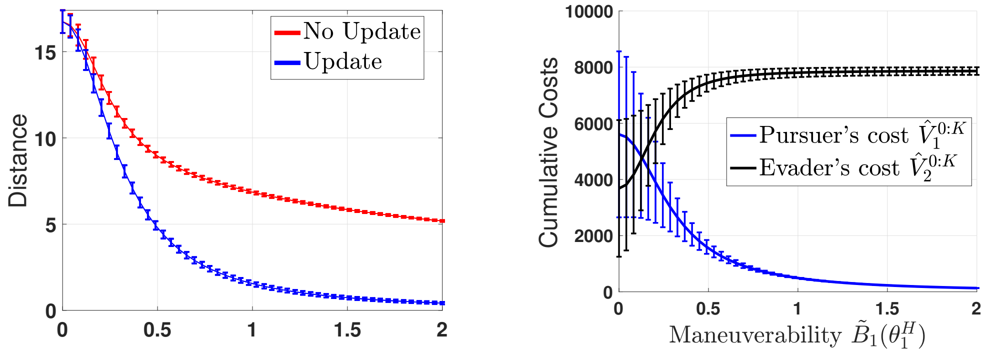
(d) The evader's K-stage ex-post cumulative cost $\hat{V}_2^{0:K}$.

Figure 6.7: The influence of the initial belief mismatch on deception. Error bars represent variances of the random variables.

the existence of Bayesian learning. However, the initial belief manipulation has a much less influence on the endpoint distance x_1^{fd} when Bayesian learning is applied. Fig. 6.7b shows that for each realization of the noise sequence w^k, the pursuer's truth-revealing stage steps down as the belief mismatch decreases when Bayesian update is applied. Fig. 6.7c illustrates the pursuer's ex-post cumulative cost $\hat{V}_1^{0:K}$ and $\hat{V}_1^{0:K-1}$ at the last and the second last stage, respectively. Without Bayesian update, the evader's deception significantly increases the pursuer's cost at the second last stage due to the large endpoint distance x_1^{fd}. The red lines show that the cost increase is higher under a larger belief mismatch. Fig. 6.7d illustrates the evader's ex-post cumulative cost at the last stage. If the pursuer does not apply Bayesian learning, then the evader can decrease his cost by increasing the pursuer's belief mismatch. If the pursuer applies Bayesian learning, then the evader's cost increases slightly if the pursuer's belief mismatch is increased. When the belief mismatch is small (i.e., $1 - l_1^0 \in (0, 0.35)$), we observe a win-win situation; i.e., Bayesian learning not only reduces the pursuer's ex-post cumulative cost, but also the evader's.

The Impact of the Pursuer's Maneuverability

The pursuer's maneuverability can also affect deception as shown in Fig. 6.8. The pursuer has an initial belief $l_1^0(\theta_2^b | x^0, \theta_1^H) = 0.5$ and the evader knows the pursuer's type. Fig. 6.8a illustrates that the pursuer can exponentially decrease her distance to the evader at the final stage as her maneuverability increases. Fig. 6.8b demonstrates that the maneuverability increase can decrease and increase the pursuer's and the evader's ex-post cumulative costs at the final stage, respectively. The variance grows as maneuverability decreases because the pursuer's trajectory



(a) Distance x_1^{fd} with its variance magnified by 100 times.

(b) Two UAVs' K-stage costs $\hat{V}_1^{0:K}$ and $\hat{V}_2^{0:K}$.

Figure 6.8: The influence of the pursuer's maneuverability on deception. Error bars represent variances of the random variables.

will become largely affected by the external noise. In both figures, we observe the phenomenon of the *marginal effect*; i.e., the change rates of both the endpoint distance x_1^{fd} and the cost $\hat{V}_i^{0:K}$ decrease as the maneuverability increases. Thus, we conclude that higher maneuverability can improve the pursuer's performance under the evader's deception as measured by the distance x_1^{fd} and the cost $\hat{V}_1^{0:K}$. Moreover, the improvement rate is higher with low maneuverability.

Deceivability, Distinguishability, and PoD

Deceivability defined in Definition 14 is highly related to the distinguishblity among different types. In this case study, a larger distance between targets, i.e., $||\gamma(\theta_2^g) - \gamma(\theta_2^b)||_2$, makes it easier for the pursuer to distinguish between evaders of type θ_2^b and type θ_2^g. A larger maneuverability difference $|\tilde{B}_1(\theta_1^H) - \tilde{B}_1(\theta_1^L)|$ makes it easier for the evader to distinguish between pursuers of type θ_1^H and type θ_1^L. We visualize two UAVs' truth-revealing stages k_i^{tr} versus the distance between targets and the maneuverability difference in Fig. 6.9. The evader has a coupled cost and both players' initial belief mismatches are 0.5. The dashed black line indicates $\tilde{B}_1(\theta_1^L) = 0.3$. When the maneuverability difference is negligible

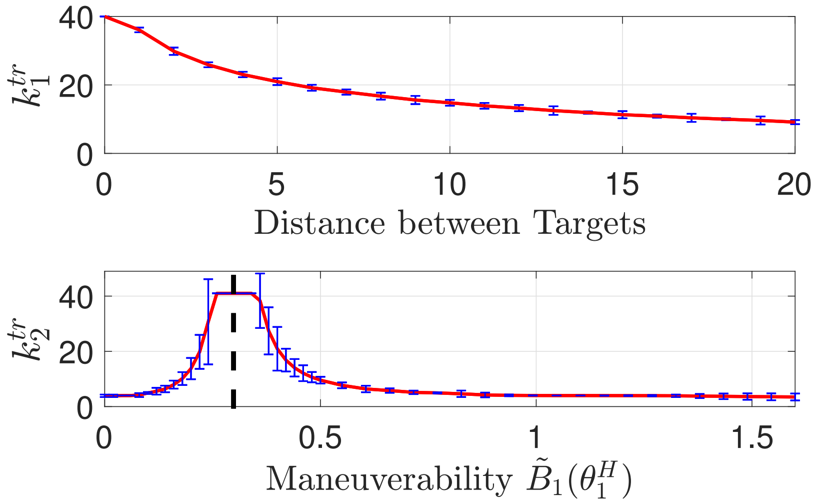


Figure 6.9: The deceived robot's truth-revealing stage versus the deceiver's type distinguishability. Error bars represent their variances, which are magnified by 5 times.

$\tilde{B}_1(\theta_1^H) \in (0.26, 0.36)$, the pursuer's type cannot be learned correctly in K stages; i.e., the pursuer is $(K+1)$-stage 0-deceivable. When the maneuverability difference is small, i.e., $\tilde{B}_1(\theta_1^H) \in (0.1, 0.5)$, yet not negligible, i.e., $\tilde{B}_1(\theta_1^H) \notin (0.26, 0.36)$, the variance of k_2^{tr} is large.

Let $\theta_2 = \theta_2^b$ be *common knowledge* and assume that the evader's belief confirms to the prior distribution of the pursuer's type for all stages, i.e., $l_2^k(\theta_1 | h^k, \theta^b) = \Xi_1(\theta_1), \forall \theta_1 \in \Theta_1, \forall k \in \mathcal{K}$. Then, Fig. 6.10 illustrates how the prior distribution of the pursuer's type affects the value of PoD under three scenarios:

- $\eta_1 = 1$, i.e., the central planner only evaluates UAV 1's performance under deception.

- $\eta_1 = 0$, i.e., the central planner only evaluates UAV 2's performance under deception.

- $\eta_1 = 0.5$, i.e., the central planner evaluates the average performance of two UAVs under deception.

When the pursuer's type is also *common knowledge*, i.e., $\Xi_1(\theta_1^H) = 0$ (i.e., the pursuer has type θ_1^L) and $\Xi_1(\theta_1^H) = 1$ (i.e., the pursuer has type θ_1^H), the game is of complete information and the value of PoD equals 1. Since PoD takes continuous values over $\Xi_1(\theta_1^H) \in [0, 1]$ and has a value of 1 at two endpoints for all feasible η_1, we refer to the plots in Fig. 6.10 as *jump rope* plots. They corroborate that the PoD can be bigger than 1; i.e., deception among players may not only benefit the deceiver but also the deceivee.

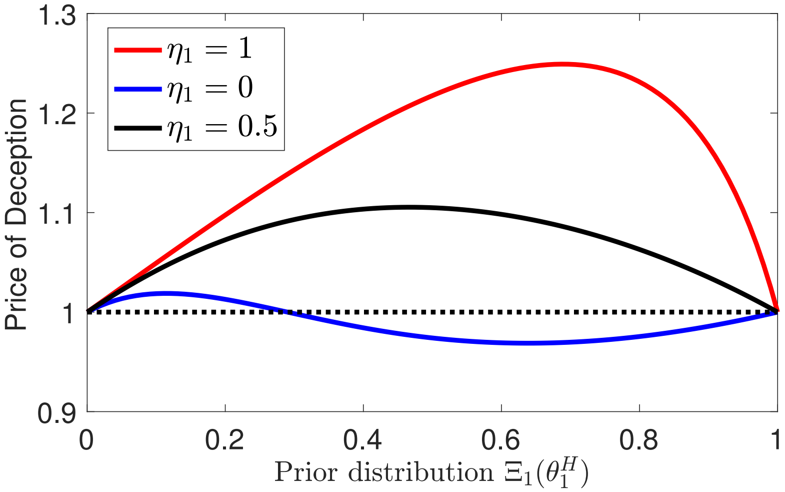


Figure 6.10: PoD vs. prior type distribution for three values of η_1.

Part IV

Defensive Deception Technologies

Chapter 7

Cognitive Honeypots for Lateral Movement Mitigation

Following Section 1.3.2, it is challenging to detect and terminate the adversarial lateral movement of APTs in Fig. 1.3 timely. Since APTs attackers can remain undetected in compromised nodes for a long time, a network that is secure at any separate time may become insecure if the times and the spatial locations are considered holistically. Therefore, the defender needs to reduce the Long-Term Vulnerability (LTV) of valuable assets. Honeypots, as a promising deceptive defense method, can detect lateral movement attacks at their early stages. Since advanced attackers can identify the honeypots located at fixed machines that are segregated from the production system, we propose a cognitive honeypot mechanism which reconfigures idle production nodes as honeypot at different stages based on the probability of service links and successful compromise. We use time-expanded networks in Section 7.1 to model the time of the random service occurrence and the adversarial compromise explicitly.

7.1 Chronological Enterprise Network Model

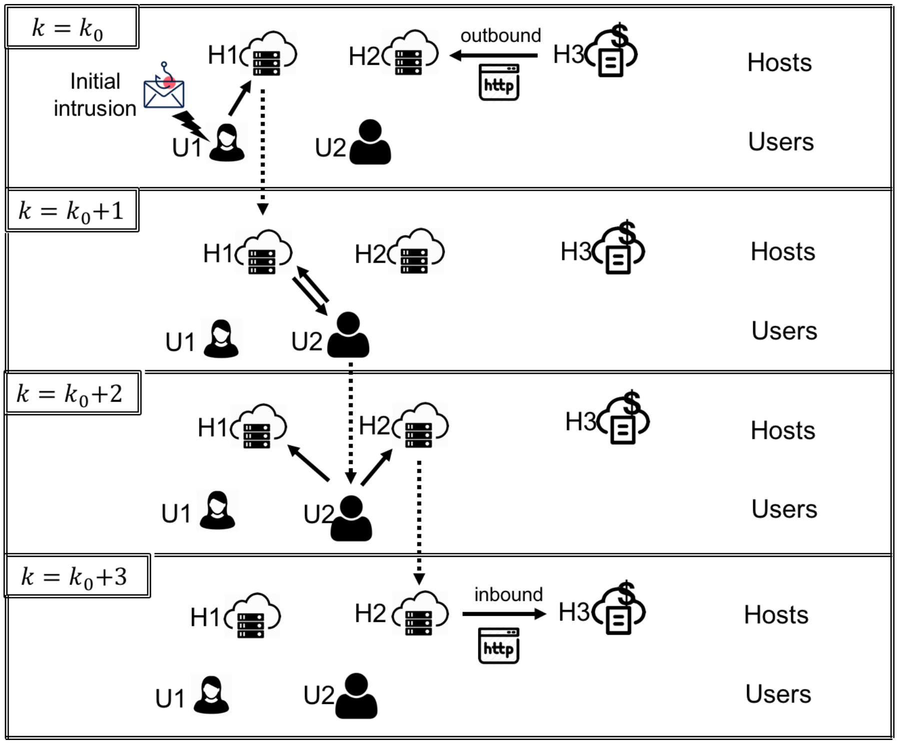


Figure 7.1: A sequence of user-host networks with service links in chronological order under discrete stage-index k. The initial stage k_0 is the stage of the attacker's initial intrusion yet the defender does not know the value of k_0. The solid arrows show the direction of the user-host and host-host network flows. By incorporating part of temporal links denoted by the dashed arrows, we reveal the attack path over a long period explicitly.

We model the normal operation of an enterprise network over a continuous period as a sequence of user-host networks in chronological order. As shown in Fig. 7.1, nodes U1 and U2 represent the two users' client computers. Nodes H1, H2, and H3 represent three hosts in the network. In particular, host H3 stores confidential information or controls a critical actuator, thus the defender needs to protect H3 from attacks. Define $\mathcal{V} := \{\mathcal{V}_U, \mathcal{V}_H\}$ as the node set where $\mathcal{V}_U, \mathcal{V}_H$ are the sets of the user nodes and hosts, respectively. The solid arrows represent two types of service links, i.e., the user-host connections and the host-host communications through an application such as HTTP [27]. Users such as U1 and U2 can access non-confidential hosts, such as H1 and H2, through their client computers for upload and/or download. However, to prevent data theft and physical damages, host H3 is inaccessible to users; e.g., there are no service links from U1 or U2 to H3 at any stage k. Since the normal operation requires data exchanges among hosts, directed network flows exist among hosts at different stages; e.g., H3 has an outbound connection to H2 at stage $k = k_0$ and an inbound connection from H2 at stage $k = k_0 + 3$. We assume that both types of service links occur randomly and last for a random but finite duration. Whenever there is a change of network topology, i.e., adding or deleting the user-host and host-host links, we define it as a new stage. We can characterize the chronological network as a series of user-host networks at discrete stages $k = k_0, k_0 + 1, \cdots, k_0 + \Delta k$, where the initial stage $k_0 \in \mathbb{Z}^+$ and $\Delta k \in \mathbb{Z}_0^+$. Since APTs are stealthy, the defender may not know the value of k_0, i.e., when the initial intrusion happens or has already happened. The lack of accurate and timely identification of the initial intrusion brings a significant challenge to detect and deter the lateral movement.

7.1.1 Time-Expanded Network and Random Service Links

We abstract the discrete series of networks in Fig. 7.1 from $k \in \{k_0, \cdots, k_0+\Delta k\}$ as a time-expanded network $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \Delta k)$ in Fig. 7.2. In the time-expanded network, we distinguish the same user or host node by the stage k and define $n_i^k \in \mathcal{V}$ as the i-th node in set $\mathcal{V}$ at stage $k \in \{k_0, \cdots, k_0 + \Delta k\}$. We drop the superscript k if we refer to the node rather than the node at stage k or the time does not matter. We can assume without loss of generality that the number of nodes $N := |\mathcal{V}|$ does

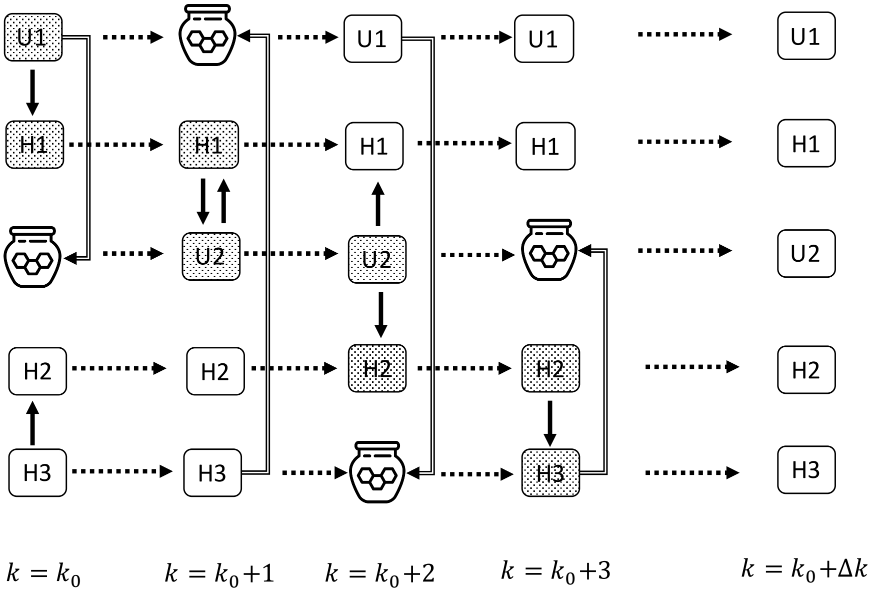


Figure 7.2: Time-expanded network $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \Delta k\}$ for the adversarial lateral movement and the cognitive honeypot configuration. The solid, dashed, double-lined arrows represent the service links, the temporal connections, and the honey links to honeypots, respectively. The shadowed nodes reveal the attack path from U1 to H3 explicitly over $\Delta k = 3$ stages.

not change with time as we can let $\mathcal{V}$ contain all the potential users and hosts in the enterprise network over Δk stages. The link set $\mathcal{E} := \{\mathcal{E}^{k_0}, \cdots, \mathcal{E}^{k_0+\Delta k}\} \cup \{\mathcal{E}_C^{k_0}, \cdots, \mathcal{E}_C^{k_0+\Delta k-1}\}$ consists of two parts. On the one hand, the user-host and host-host connections at each stage $k \in \{k_0, \cdots, k_0+\Delta k\}$ are represented by the set $\mathcal{E}^k = \{e(n_i^k, n_j^k) \in \{0,1\} | n_i^k, n_j^k \in \mathcal{V}, i \neq j, \forall i, j \in \{1, \cdots, N\}\}$. On the other hand, set $\mathcal{E}_C^k := \{e(n_i^k, n_i^{k+1}) = 1 | n_i^k, n_i^{k+1} \in \mathcal{V}, \forall i \in \{1, \cdots, N\}\}$ contains the virtual temporal links from stage k to $k+1$. A link exists if $e(\cdot,\cdot) = 1$ and does not if $e(\cdot,\cdot) = 0$. The time-expanded network $\mathcal{G}$ is a directed graph due to the temporal causality represented by the set $\mathcal{E}_C^k, k \in \{k_0, \cdots, k_0 + \Delta k - 1\}$.

Since the user-host and the host-host connections happen randomly at each stage, we assume that a service link from node $n_i^k \in \mathcal{V}$ to node $n_j^k \in \mathcal{V} \setminus \{n_i^k\}$ exists with probability $\beta_{i,j} \in [0,1]$ for any stage $k \in \{k_0, \cdots, k_0 + \Delta k\}$. If a connection from node n_i^k to n_j^k is prohibitive; e.g., U1 cannot access H3 in Fig. 7.1, then $\beta_{i,j} = 0$. We can define $\beta := \{\beta_{i,j}\}, i, j \in \{1, \cdots, N\}$, as the service-link generating matrix without loss of generality by letting $\beta_{i,i} = 0, \forall i \in \{1, \cdots, N\}$. In this work, we consider a time-invariant β whose value can be estimated empirically from long-term historical data[1]. The service links at each stage may only involve a small number of nodes and leave other nodes idle.

Definition 16. *A node $n_i^k \in \mathcal{V}$ is said to be **idle** at stage k if it is neither the source nor the sink node of any service link at stage k, i.e., $e(n_i^k, n_j^k) = 0, e(n_j^k, n_i^k) = 0, \forall n_j^k \in \mathcal{V}$.*

7.1.2 Attack Model of Lateral Movement

We assume that the initial intrusion can only happen at a subset of N nodes $\mathcal{V}_I \subseteq \mathcal{V}$ due to the network segregation. We can refer to $\mathcal{V}_I$ as the Demilitarized Zone (DMZ). Take Fig. 7.1 as an example, if all hosts in the enterprise network are segregated from the Internet, the initial intrusion can only happen to the client computer of U1 or U2 through phishing emails or social engineering. Although network segregation narrows down the potential location of initial intrusion from $\mathcal{V}$ to the subset $\mathcal{V}_I$ that may contain only one node, it is still challenging for the defender to prevent the nodes in $\mathcal{V}_I$ from an initial intrusion as the defender cannot determine *when* the initial intrusion happens; i.e., the value of k_0 is unknown. In this work, we assume that the initial intrusion only happens to one node in set $\mathcal{V}_I$ at a time; i.e., no concurrent intrusions happen. Once the attacker has entered

[1] For example, we can use the user-computer authentication dataset from the Los Alamos National Laboratory enterprise network [71] to estimate the probability of user-host service links over a long period. The dataset is available at https://csr.lanl.gov/data/auth/.

the enterprise network via the initial intrusion from an external network domain, he does not launch new intrusions from the external domain to compromise more nodes in $\mathcal{V}_I$. Instead, the attacker can exploit the internal service links to move laterally over time, which is much stealthier than intrusions from external network domains. For example, after the attacker has controlled U1's computer by phishing emails, he would not send phishing emails to other users from the external network domain, which increases his probability of being detected. We define $\rho_i \in [0,1]$ as the probability that the initial intrusion happens at node $n_i^{k_0} \in \mathcal{V}_I, \forall k_0 \in \mathbb{Z}^+$. The probability satisfies $\sum_{i \in \mathcal{V}_I} \rho_i = 1$ and is assumed to be independent of the stage k_0. This probability of initial intrusion can be estimated based on the node's vulnerability assessed by historical data, red team exercises, and the Common Vulnerability Scoring System (CVSS) [139].

After the initial intrusion, the attacker can exploit service links at different stages by various techniques to move laterally, such as Pass the Hash (PtH), taint shared content, and remote service session hijacking [31]. Take PtH as an example, when a user enters the password and logs into host H1 from a compromised client computer U1 at stage k_0 as shown in Fig. 7.1, the attacker at U1 can capture the valid password hashes for accessing host H1 by credential access technique. Then, the attacker can use the captured hashes to access the host H1 for all the future stage $k > k_0$. The attacker can also compromise a user node from a compromised host by tainting the shared content, i.e., adding malicious scripts to valid files in the host. Then, the malicious code can be executed when user U2 downloads those files from H1 at stage $k_0 + 1$. PtH (resp. tainting shared content) enables an adversarial lateral movement from a user node (resp. host node) to a host node (resp. user node). The attacker can also use remote service session hijacking, such as Secure

Shell (SSH) hijacking and Remote Desktop Protocol (RDP) hijacking, to move laterally between hosts by hijacking the inbound or outbound network flows. In this work, we assume that once the attacker compromises a node, he retains the control of the node for the given length of time window Δk determined by the defender. For example, the defender can require users to update their password every Δk days to invalidate the PtH attack. During the time window, i.e., from the initial intrusion $k = k_0$ to $k = k_0 + \Delta k$, the attacker can launch simultaneous attacks from all the compromised nodes to move laterally whenever there are outbound service links from them. If there are multiple service links from one compromised node, the attacker can also compromise all the sink nodes of these service links within the stage. Note that the only objective of the attacker is to search for valuable nodes (e.g., H3), compromise it, and then launch subversive attacks for data theft and physical damages. Thus, we assume that the attack does not launch any subversive attacks in all the compromised nodes except at the target node to remain stealthy. That is, even though the attacker retains the control of the compromised nodes, he only uses them as stepping stones to reach the target node.

The persistent lateral movement over a long time period enables the attacker to reach and compromise segregated nodes that are not in the DMZ $\mathcal{V}_I$. In both Fig. 7.1 and Fig. 7.2, although the network has no direct service links, represented by solid arrows, from U1 to H3 at each stage, the cascade of *static security* in all stages does not result in *long-term security* over $\Delta k = 3$ stages. After we add the temporal links represented by the dashed arrows and consider stages and spatial locations holistically, we can see the attack path from the initial intrusion node U1 to the target node H3 over $\Delta k = 3$ stages as highlighted by the shadows in Fig. 7.2. The temporal order of the service links affects the likelihood that the attacker can compromise the target node. For example, if we exchange the services links that happen at stage $k_0 + 1$ and stage $k_0 + 2$, then the attacker from node U1 cannot reach H3 in $\Delta k = 3$ stages. Since the attacker can launch simultaneous attacks from multiple compromised nodes to move laterally, there can exist multiple attack paths from an initial intrusion node to the target node.

The adversarial exploitation of service links is not always successful due to the defender's mitigation technologies against lateral movement techniques [31]. For example, the firewall rules to block RDP traffic between hosts can invalidate RDP hijacking. If the attacker has compromised nodes $n_i^{k'} \in \mathcal{V}$ before stage $k > k'$ and a service link from n_i^k to $n_j^k \in \mathcal{V} \setminus \{n_i^k\}$ exists at stage k, i.e., $e(n_i^k, n_j^k) = 1$, we can define $\lambda_{i,j} \in [0, 1]$ as the probability that the attacker at node n_i^k successfully compromises node n_j^k, which is assumed to be independent of stage k.

7.1.3 Cognitive Honeypot

The lateral movement of persistent and stealthy attacks makes the enterprise network insecure in the long run. The high rates of false alarms and the miss detection of both the initial external intrusion and the following internal compromise make it challenging for the defender to identify the set of nodes that have been compromised. Thus, the defender needs to patch and reset all suspicious nodes at all stages to deter the attacks, which can be cost-prohibitive.

Honeypots are a promising active defense method to detect and deter these persistent and stealthy attacks by deception [152]. In this paper, the connection from a service node to a honeypot is referred to as a honey link. The defender disguises a honey link as a service link to attract attackers. For example, the defender can start a session with remote services from a host to a honeypot. The attacker who has compromised the host will be detected once he hijacks the remote service session and carries out actions in the honeypots. Since regular honeypots are implemented at fixed locations and on machines that are never involved in the regular operation, advanced attacks like APTs can identify the honeypots and avoid accessing them. Motivated by the roaming honeypot [115] and the fact that the service links at each stage only involve a small number of nodes, we develop the following cognitive honeypot configuration that utilizes and reconfigures different idle nodes at different stages as honeypots. Let $\mathcal{V}_D \subseteq \mathcal{V}$ be the subset of nodes that can be reconfigured as honeypots when idle. At each stage k, the defender randomly selects a node $n_w^k \in \mathcal{V}_D$ to be the potential honeypot and creates a random honey link from other nodes to n_w^k. Since disguising a honeypot as a normal node requires emulating massive services and the continuous monitoring of all inbound network flows are costly, we assume that the defender sets up at most one honeypot and monitors one honey link at each stage.

As shown in Fig. 7.2, U1, H2, and H3 are idle at stage $k_0 + 1$ and U1 is reconfigured as the honeypot. The link from H3 to U1 is the honey link which is monitored by the defender. At stage k_0, U2 is the only idle node and is reconfigured as the honeypot with a honey link from U1 to U2. As stated in Section 7.1.2, the attacker who has compromised U1 at stage k_0 remains stealthy and does not sabotage any normal operations. Thus, the defender can reconfigure U1 as a honeypot at stage k_0+1. However, the honeypot of U1 at stage k_0+1 cannot identify the attacker by monitoring all the inbound traffic as he has already compromised U1. On the contrary, the honeypots at stage k_0 and $k_0 + 2$ can trap the attackers who have compromised U1 and mistaken the honey links as service links[2]. Theoretically,

[2] The defender would avoid configuring honey links from the target node to the honeypot. If the attacker has not compromised the target node H3 as shown in stage $k_0 + 1$, the honeypot

the honeypot can achieve zero false alarms as the legitimate network flows should occur only at the service links. For example, although the existence of the honey link at stage k_0 enables legitimate users at U1 to access another user's computer U2, a legitimate user aiming to finish the service link from U1 to H1 should not access any irrelevant nodes other than host H1. On the other hand, an attacker at U1 cannot tell whether the links from U1 to H1 and U2 are service links or honey links. Thus, only an attacker at U1 can access the honeypot U2 at stage k_0.

Random Honeypot Configuration and Detection

Since the defender can neither predict future service links nor determine the set of compromised nodes at the current stage, she needs to develop a time-independent policy $\gamma := \{\gamma_{l,w}\}, \forall n_l^k, n_w^k \in \mathcal{V}$, to determine the honeypot location and the honey link at each stage k to minimize the risk that an attacker from the node of the initial intrusion can compromise the target node after Δk stages. Each policy element $\gamma_{l,w}$ is the probability that the honeypot is node n_w^k and the honey link is from node n_l^k to n_w^k at stage $k \in \{k_0, \cdots, k_0 + \Delta k\}$. Note that $\gamma_{i,i} = 0, \forall i \in \mathcal{V}$, and we can let n_l, n_w belong to the entire node set $\mathcal{V}$ without loss of generality because if a node $n_w \notin \mathcal{V}_D$ is not reconfigurable, then we can let the probability $\gamma_{l,w}$ be zero. Define $n_{j_0} \in \mathcal{V} \setminus \mathcal{V}_I$ as the target node to protect for all stages and the target node is segregated from the set of potential initial intrusion. Then, defender should avoid honey links from node n_{j_0} for all stages, i.e., $\gamma_{j_0,w} = 0, \forall n_w \in \mathcal{V}$. If a honey link from n_l to n_w, e.g., the link from U1 to H3, is not available for all stages due to segregation, then $\gamma_{l,w} = 0$. Since at most one link is allowed, we have the constraint $\sum_{n_l,n_w \in \mathcal{V}} \gamma_{l,w} = 1$. In this work, we assume that the honeypot policy

cannot capture the attacker. If the attacker has compromised the target node as shown in stage $k_0 + 3$, then the late detection cannot reduce the loss that has already been made.

γ is not affected by the realization of the service links at each stage and thus can interfere with the service links that are not idle as defined in Definition 16. If the honeypot n_w^k selected by the policy γ is interfering, i.e., not *idle*, then the defender neither monitors nor filters the inbound network flows to avoid any interference with the normal operation.

Although we increase the difficulty for the attacker to identify the honeypot by applying it to idle nodes in the network and change its location at every stage, we cannot eliminate the possibility of advanced attackers identifying the honeypot [118]. If the attacker has compromised node n_i before stage k and there is a honey link from node n_i^k to n_j^k at stage k, then we assume that the attacker has probability $q_{i,j} \in [0, 1]$ to identify the honey link and choose not to access the honeypot. If the honeypot is not identified, then the attacker accesses the honeypot and he is detected by the defender. We assume the defender can deter the lateral movement completely after a detection from any single honeypot by patching or resetting all nodes at that stage. As stated in Section 7.1.2, the attacker can move simultaneously from all the compromised nodes to multiple nodes through service links that connect them. For example, the attacker at stage $k_0 + 2$ can compromise H2 and H1 through the two service links and may also reach the honeypot if the attacker attempts to compromise H3 from U1. However, we assume that the attacker at a compromised node does not move consecutively through multiple service links (or honey links defined in Section 7.1.3 as the attacker cannot distinguish honey links from service ones) in a single stage to remain stealthy. Contrary to the persistent lateral movement over a long time period, consecutive attack moves within one stage make it easier for the defender to connect all Indicators of Compromise (IoCs) and attribute the attacker. Take Fig. 7.2 as an example. Suppose that there are two links, e.g., H1 to U2 and U2 to H2 at a stage k, where each link can be either a service link or a honey link. If the attacker has only compromised H1 among these three nodes, then he only attempts to compromise node U2 rather than both U2 and H2 during stage k.

Interference, Stealthiness, and Cost of Roaming

In this section, we define three critical security metrics for a cognitive honeypot to achieve low interference, low cost, and high stealthiness. Define $\mathcal{V}_S$ as the set of all the subsets of $\mathcal{V}$. Define a series of binary random variables $x^k_{v,w,v'} \in \{0,1\}, v, v' \in \mathcal{V}_S, n^k_w \in \mathcal{V}$, where $x^k_{v,w,v'} = 1$ means that there are no direct service links from any node $n^k_l \in v$ to node n^k_w and from n^k_w to $n^k_l \in v'$ at stage k. Thus, $\Pr(x^k_{v,w,v'} = 1) = \prod_{n^k_l \in v}(1-\beta_{l,w}) \prod_{n^k_{l'} \in v'}(1-\beta_{w,l'})$ represents the probability that the honeypot at n^k_w does not interfere with any service link whose source node is in set v and sink node is in v'. Then, we can define $H_{PoI}(\gamma)$ as the probability of interference in Definition 17. Since the defender can only apply cognitive honeypots to idle nodes, a low probability of interfering can increase efficiency. To reduce $H_{PoI}(\gamma)$, the defender can design γ based on the value of β, i.e., the frequency/probability of all potential service links.

Definition 17. *The* ***probability of interference*** *(PoI) for any honeypot policy γ is*

$$\begin{aligned} H_{PoI}(\gamma) &:= \sum_{n_h \in \mathcal{V}} \sum_{n_w \in \mathcal{V} \setminus \{n_h\}} \gamma_{h,w}(1 - \Pr(x^k_{\mathcal{V}\setminus\{n_w\},w,\mathcal{V}\setminus\{n_w\}} = 1)) \\ &= \sum_{n_w \in \mathcal{V}} (1 - \Pr(x^k_{\mathcal{V}\setminus\{n_w\},w,\mathcal{V}\setminus\{n_w\}} = 1)) \sum_{n_h \in \mathcal{V}\setminus\{n_w\}} \gamma_{h,w}. \end{aligned} \tag{7.1}$$

Since the attacker can learn the honeypot policy γ, the defender prefers the policy to be as random as possible to increase the stealthiness of the honeypot. A fully random policy that assigns equal probability to all possible honey links provides forward and backward security; i.e., even if an attacker identifies the honeypot at stage k, he cannot use that information to deduce the location of the honeypots in the following and previous stages. We use $H_{SL}(\gamma)$, the entropy of γ in Definition 18 as a measure for the stealthiness level of the honeypot policy where we define $0 \cdot \log 0 = 0$.

Definition 18. *The stealthiness level (SL) for any γ is defined as $H_{SL}(\gamma) := \sum_{n_h, n_w \in \mathcal{V}} \gamma_{h,w} \log(\gamma_{h,w})$.*

A tradeoff of roaming honeypots hinges on the cost to reconfigure the idle nodes when the defender changes the location of the honeypot and the honey link. Define the term $C(\gamma_{h_1,w_1}, \gamma_{h_2,w_2}), \forall n_{h_1}, n_{h_2}, n_{w_1}, n_{w_2} \in \mathcal{V}$, as the cost of changing a $(n_{h_1} - n_{w_1})$ honey link to a $(n_{h_2} - n_{w_2})$ honey link. Note that this cost captures the cost of changing the honeypot location from w_1 to w_2. If only the location change of honeypots incurs a cost, we can let $C(\gamma_{h_1,w}, \gamma_{h_2,w}) = 0, \forall h_1 \neq h_2, \forall n_w \in \mathcal{V}$, without loss of generality. We define the cost of roaming in Definition 19.

Definition 19. *The* ***cost of roaming*** *(CoR) for any honeypot policy γ is*

$$H_{CoR}(\gamma) := \sum_{n_{h_1} \in \mathcal{V}} \sum_{n_{w_1} \in \mathcal{V} \setminus \{n_{h_1}\}} \gamma_{h_1,w_1} (1 - \Pr(x^k_{\mathcal{V} \setminus \{n_{w_1}\}, w_1, \mathcal{V} \setminus \{n_{w_1}\}} = 1))$$
$$\cdot \sum_{n_{h_2} \in \mathcal{V}} \sum_{n_{w_2} \in \mathcal{V} \setminus \{h_2\}} \gamma_{h_2,w_2} (1 - \Pr(x^k_{\mathcal{V} \setminus \{n_{w_2}\}, w_2, \mathcal{V} \setminus \{n_{w_2}\}} = 1)) \cdot C(\gamma_{h_1,w_1}, \gamma_{h_2,w_2}) \quad (7.2)$$

7.2 Farsighted Vulnerability Mitigation

Throughout the entire operation of the enterprise network, the defender does not know whether, when, and where the initial intrusion has happened. The defender also cannot know attack paths until a honeypot detects the lateral movement attack. Therefore, instead of reactive policies to mitigate attacks that have happened at known stages, we aim at proactive and persistent policies that prepare for the initial intrusion at any stage k_0 over a time window of length Δk. That means that the honeypot should roam persistently at all stages according to the policy γ to reduce LTV, i.e., the probability that an initial intrusion can reach and compromise the target node within Δk stages.

Given the target node $n_{j_0} \in \mathcal{V} \setminus \mathcal{V}_I$, a subset $v \in \mathcal{V}_S$, and the defender's honeypot policy γ, we define $g_{j_0}(v, \gamma, \Delta k)$ as the probability that an attacker who has compromised the set of nodes v can compromise the target node n_{j_0} within Δk stages. Since the initial intrusion happens to a single node $n_i \in \mathcal{V}_I$ with probability ρ_i as argued in Section 7.1.2, the Δk-stage vulnerability of the target node n_{j_0} defined in Definition 20 equals $\bar{g}^{\Delta k}_{j_0, \mathcal{V}_I}(\gamma) := \sum_{n_i \in \mathcal{V}_I} \rho_i g_{j_0}(\{n_i\}, \gamma, \Delta k)$. In this paper, we refer to Δk-stage vulnerability as LTV when $\Delta k > 1$.

Definition 20 (Long-Term Vulnerability)**.** *The Δk-**stage vulnerability** of the target node n_{j_0} is the probability that an attacker in the DMZ $\mathcal{V}_I$ can compromise the target node n_{j_0} within a time window of Δk stages.*

The length of the time window represents the attack's time-effectiveness which is determined by the system setting and the defender's detection efficiency. For example, Δk can be the time-to-live (typically on the order of days [173]) for re-authentication to invalidate the PtH attack. For another example, suppose that the defender can detect and deter the attacker after the initial intrusion yet with a delay due to the high rate of false alarms. If the delay can be contained within Δk_0 stages, then the defender should choose the honeypot policy to minimize the Δk_0-stage vulnerability. Consider a given threshold $T_0 \in [0, 1]$, we define the concept of level-T_0 stage-Δk security for node n_{j_0} and honeypot policy γ in Definition 21.

Definition 21 (Long-Term Security)**.** *Policy γ achieves **level-T_0 stage-Δk security** for node n_{j_0} if the Δk-stage vulnerability is less than the threshold, i.e., $\bar{g}^{\Delta k}_{j_0,\mathcal{V}_I}(\gamma) \leq T_0$.*

Finally, we define the defender's decision problem of a cognitive honeypot that can minimize the LTV for the target node with a low PoI, a high SL, and a low CoR in (7.3). The coefficients $\alpha_{PoI}, \alpha_{SL}, \alpha_{CoR}$ represent the tradeoffs of Δk-stage vulnerabilities with PoI, SL, and CoR, respectively.

$$\begin{aligned}
\min_{\gamma} \quad & \bar{g}^{\Delta k}_{j_0,\mathcal{V}_I}(\gamma) + \alpha_{PoI} H_{PoI}(\gamma) - \alpha_{SL} H_{SL}(\gamma) + \alpha_{CoR} H_{CoR}(\gamma) \\
\text{s.t.} \quad & \sum_{n_h, n_w \in \mathcal{V}} \gamma_{h,w} = 1, \\
& \gamma_{h,w} = 0, \forall n_h \in \mathcal{V}, n_w \in \mathcal{V} \setminus \mathcal{V}_D.
\end{aligned} \tag{7.3}$$

7.2.1 Imminent Vulnerability

We first compute the probability that an initial intrusion at node $n_i \in \mathcal{V}_I$ can compromise the target node $n_{j_0} \in \mathcal{V} \setminus \mathcal{V}_I$ within $\Delta k = 0$ stages. The term $\gamma_{i,w}(1 - q_{i,w})$ is the *Probability of Immediate Capture (**PoIC**)*, i.e., the attacker with initial intrusion at node n_i is directly trapped by the honeypot n_w. Since the attacker does not take consecutive movements in one stage to remain stealthy as stated in Section 7.1.2, $g_{j_0}(\{n_i\}, \gamma, 0)$ equals the product of the probability that attacker exploits the service link from n_i to n_{j_0} successfully and the probability that the attacker is not trapped by the honeypot, i.e., $\forall n_i \in \mathcal{V}_I$,

$$g_{j_0}(\{n_i\}, \gamma, 0) = \beta_{i,j_0} \lambda_{i,j_0} \Big(1 - \sum_{w \neq i, j_0} \gamma_{i,w}(1 - q_{i,w}) \Pr(x^k_{\mathcal{V}\setminus\{n_w\},w,\mathcal{V}\setminus\{n_w\}} = 1)\Big). \tag{7.4}$$

7.2.2 Long-Term Vulnerability

Define $\mathcal{V}_{i,j_0} \subseteq \mathcal{V}_S$ as the set of all the subsets of $\mathcal{V} \setminus \{n_i, n_{j_0}\}$. For each $v \in \mathcal{V}_{i,j_0}$, define $\mathcal{V}^v_{i,j_0}$ as the set of all the subsets of $\mathcal{V} \setminus \{n_i, n_{j_0}, v\}$. Define the shorthand notation $f_{v,u}(\beta, \lambda) := \prod_{n_{h_1} \in v} \beta_{i,h_1} \lambda_{i,h_1} \prod_{n_{h_2} \in u} \beta_{i,h_2}(1 - \lambda_{i,h_2}) \prod_{n_{h_3} \in \mathcal{V} \setminus \{n_i, n_{j_0}, v, u\}}(1 - \beta_{i,h_3})$ as the *probability of partial compromise*, i.e., the attacker with initial intrusion at node n_i has compromised the service links from n_i to all nodes in set $v \in \mathcal{V}_{i,j_0}$, yet fails to compromise the remaining service links from n_i to all nodes in set $u \in \mathcal{V}^v_{i,j_0}$. We can compute $g_{j_0}(\{n_i\}, \gamma, \Delta k)$ based on the following induction, i.e.,

$$
\begin{aligned}
g_{j_0}(\{n_i\}, \gamma, \Delta k) = g_{j_0}(\{n_i\}, \gamma, 0) + (1 - \beta_{i,j_0}\lambda_{i,j_0}) \sum_{v \in \mathcal{V}_{i,j_0}} \sum_{u \in \mathcal{V}^v_{i,j_0}} f_{v,u}(\beta, \lambda)(1- \\
\sum_{n_w \in \mathcal{V} \setminus \{n_i, v, u\}} \gamma_{i,w}(1 - q_{i,w}) \Pr(x^k_{\mathcal{V} \setminus \{n_i, n_w\}, w, \mathcal{V} \setminus \{n_w\}} = 1)) g_{j_0}(\{n_i\} \cup v, \gamma, \Delta k - 1).
\end{aligned}
\tag{7.5}
$$

7.2.3 Curse of Multiple Attack Paths and Sub-Optimal Honeypot Policies

For a given γ, we can write out the explicit form of $g_{j_0}(\{n_i\} \cup v, \gamma, \Delta k - 1)$ for all $\Delta k \in \mathbb{Z}^+$ as in (7.4) and (7.5). However, the complexity increases dramatically with the cardinality of set v due to the *curse of multiple attack paths*; i.e., the event that the attacker can compromise target node n_{j_0} within Δk stages from node n_i is not independent of the event that the attacker can achieve the same compromise from node $n_h \neq n_i$. Thus, we use the union bound

$$
\begin{aligned}
g_{j_0}(\{n_i\} \cup v, \gamma, \Delta k) &\geq \max_{n_j \in \{n_i\} \cup v} g_{j_0}(\{n_j\}, \gamma, \Delta k), \\
g_{j_0}(\{n_i\} \cup v, \gamma, \Delta k) &\leq \min(1, \sum_{n_j \in \{n_i\} \cup v} g_{j_0}(\{n_j\}, \gamma, \Delta k)),
\end{aligned}
$$

to simplify the computation and provide an upper bound and a lower bound for $g_{j_0}(\{n_i\}\cup v, \gamma, \Delta k), v \neq \emptyset, \forall \Delta k \in \mathbb{Z}^+$, in (7.6) and (7.7), respectively.

$$g_{j_0}^{lower}(\{n_i\}, \gamma, \Delta k) = g_{j_0}(\{n_i\}, \gamma, 0) + (1-\beta_{i,j_0}\lambda_{i,j_0}) \sum_{v\in\mathcal{V}_{i,j_0}} \sum_{u\in\mathcal{V}_{i,j_0}^v} f_{v,u}(\beta,\lambda)(1- \sum_{n_w\in\mathcal{V}\setminus\{n_i,v,u\}} \gamma_{i,w}(1-q_{i,w}) \Pr(x^k_{\mathcal{V}\setminus\{n_i,n_w\},w,\mathcal{V}\setminus\{n_w\}} = 1)) \max_{n_j\in\{n_i\}\cup v} g_{j_0}^{lower}(\{n_j\}, \gamma, \Delta k - 1). \tag{7.6}$$

$$\begin{aligned} g_{j_0}^{upper}(\{n_i\}, \gamma, \Delta k) =& g_{j_0}(\{n_i\}, \gamma, 0) + (1-\beta_{i,j_0}\lambda_{i,j_0}) \sum_{v\in\mathcal{V}_{i,j_0}} \sum_{u\in\mathcal{V}_{i,j_0}^v} f_{v,u}(\beta,\lambda) \\ & \cdot (1 - \sum_{n_w\in\mathcal{V}\setminus\{n_i,v,u\}} \gamma_{i,w}(1-q_{i,w}) \Pr(x^k_{\mathcal{V}\setminus\{n_i,n_w\},w,\mathcal{V}\setminus\{n_w\}} = 1)) \\ & \cdot \min(1, \sum_{n_j\in\{n_i\}\cup v} g_{j_0}^{upper}(\{n_j\}, \gamma, \Delta k - 1)). \end{aligned} \tag{7.7}$$

The initial condition at $\Delta k = 0$ is

$$g_{j_0}^{lower}(\{n_j\}, \gamma, 0) = g_{j_0}^{upper}(\{n_j\}, \gamma, 0) = g_{j_0}(\{n_j\}, \gamma, 0), \forall n_j \in \{n_i\} \cup v.$$

Define

$$\bar{g}_{j_0,\mathcal{V}_I}^{\Delta k, lower}(\gamma) := \sum_{n_i\in\mathcal{V}_I} \rho_i g_{j_0}^{lower}(\{n_i\}, \gamma, \Delta k)$$

and $\bar{g}_{j_0,\mathcal{V}_I}^{\Delta k, upper}(\gamma) := \sum_{n_i\in\mathcal{V}_I} \rho_i g_{j_0}^{upper}(\{n_i\}, \gamma, \Delta k)$ as the lower and upper bounds of the Δk-stage vulnerability of the target node n_{j_0} under any given policy γ, respectively. Then, replacing $\bar{g}_{j_0,\mathcal{V}_I}^{\Delta k}(\gamma)$ in (7.3) with $\bar{g}_{j_0,\mathcal{V}_I}^{\Delta k, lower}(\gamma)$ and $\bar{g}_{j_0,\mathcal{V}_I}^{\Delta k, upper}(\gamma)$, we obtain the optimal risky and conservative honeypot policy $\gamma^{*,risky}$ and $\gamma^{*,cons}$, respectively. Both sub-optimal honeypot policies approximate the optimal policy that is hard to compute explicitly. A risky defender can choose $\gamma^{*,risky}$ to minimize the lower bound of LTV while a conservative defender can choose $\gamma^{*,cons}$ to minimize the upper bound.

We propose the following iterative algorithm to compute these two honeypot policies. We use $\gamma^{*,risky}$ as an example and $\gamma^{*,cons}$ can be computed in the same fashion. At iteration $t \in \mathbb{Z}_0^+$, we consider any feasible honeypot policy γ^t and compute $g_{j_0}^{lower}(\{n_i\}, \gamma^t, \Delta k'), \forall n_i \in \mathcal{V}_I, \forall \Delta k' \in \{1, \cdots, \Delta k\}$, via (7.6). Then, we solve (7.3) by replacing $\bar{g}_{j_0,\mathcal{V}_I}^{\Delta k}(\gamma^t)$ with $\bar{g}_{j_0,\mathcal{V}_I}^{\Delta k,lower}(\gamma^t)$ and plugging in $g_{j_0}^{lower}(\{n_i\}, \gamma^t, \Delta k), \forall n_i \in \mathcal{V}_I$, as constants. Since $\bar{g}_{j_0,\mathcal{V}_I}^{\Delta k,lower}(\gamma^t), H_{PoI}(\gamma^t), H_{CoR}(\gamma^t)$ are all linear with respect to γ^t, the objective function of the constrained optimization in (7.3) is a linear function of γ^t plus the entropy regularization $H_{SL}(\gamma^t)$. Then, we can solve the constrained optimization in closed form and update the honeypot policy from γ^t to γ^{t+1}. Given a small error threshold $\epsilon > 0$, the above iteration process can be repeated until there exists a $T_1 \in \mathbb{Z}_0^+$ such that a proper matrix norm is less than the error threshold, i.e., $||\gamma^{T_1+1} - \gamma^{T_1}|| \leq \epsilon$. Then, we can output γ^{T_1+1} as the optimal risky honeypot policy $\gamma^{*,risky}$.

7.2.4 LTV Analysis under two Heuristic Policies

In this section, we consider the scenario where the initial intrusion set $\mathcal{V}_I = \{n_i\}$ contains only one node n_i, i.e., the attacker cannot compromise other nodes directly from the external network at stage k_0. Then, a reasonable heuristic policy is to set up the honeypot at a fixed node $n_{w_0} \in \mathcal{V} \setminus \{n_i, n_{j_0}\}$ whenever the node is idle and also a direct honey link from n_i to n_{w_0}. We refer to these deterministic policies with $\gamma_{i,w_0} = 1$ as the direct policies in Section 7.2.4.

In the second scenario, the defender further segregates node n_i from the external network to form a *air gap* so that she chooses to apply no direct honey links from

Algorithm 7: Optimal Risky (and Conservative) Honeypot Policy

```
Initialization $\mathcal{V}_I, n_{j_0} \in \mathcal{V} \setminus \mathcal{V}_I, \Delta k \in \mathbb{Z}^+, \epsilon > 0, \gamma^0, t = 0$;
while $||\gamma^{t+1} - \gamma^t|| > \epsilon$ do
    for $\Delta k' = 1, \cdots, \Delta k$ do
        for $i \in \mathcal{V}_I$ do
            Compute $g_{j_0}^{lower}(\{n_i\}, \gamma^t, \Delta k')$ via (7.6);
        end
    end
    Replace $\bar{g}^{\Delta k}_{j_0,\mathcal{V}_I}(\gamma^t)$ with $\bar{g}^{\Delta k,lower}_{j_0,\mathcal{V}_I}(\gamma^t)$ and plug in
      $g_{j_0}^{lower}(\{n_i\}, \gamma^t, \Delta k), \forall n_i \in \mathcal{V}_I$;
    Obtain $\gamma^{t+1}$ as the solution of (7.3);
    if $||\gamma^{t+1} - \gamma^t|| \leq \epsilon$ then
        $T_1 = t$;
        Terminate
    $t := t + 1$;
end
Output $\gamma^{*,risky} = \gamma^{T_1+1}$.
```

n_i to any honeypot at all stages, i.e.,$\gamma_{i,w} = 0, \forall n_w \in \mathcal{V}$. However, advanced attacks, such as Stuxnet, can cross the air gap by an infected USB flash drive to accomplish the initial intrusion to the air-gap node n_i and then move laterally to the entire network $\mathcal{V}$. Although the defender mistakenly sets up no honey links from n_i to the honeypot at all stages, other indirect honey links with source nodes other than n_i may also detect the lateral movement in Δk stages. Unlike the deterministic direct policies, we refer to these stochastic policies with $\gamma_{i,w} = 0, \forall n_w \in \mathcal{V}$, as the indirect policies in Section 7.2.4.

Since the defender may adopt these heuristic policies in the listed scenarios, this section aims to analyze the LTV under the direct and indirect policies to answer the following security questions. How effective is the lateral movement for a different length of duration time under heuristic policies? What are the limit and the bounds of the vulnerability when the window length goes to infinity? How much additional vulnerability is introduced by adopting improper indirect policies rather than the direct policies? How to change the value of parameters, such as β and λ, to reduce LTV if they are designable?

Indirect Honeypot Policies

Since the defender overestimates the effectiveness of air gap and chooses the improper honeypot policies that $\gamma_{i,w} = 0, \forall n_w \in \mathcal{V}$, the vulnerability of any target node n_{j_0} is non-decreasing with the length of the time window as shown in Proposition 2.

Proposition 2 (Non-Decreasing Vulnerability over Stages)**.** *If the PoIC is zero, i.e., $\gamma_{i,w}(1-q_{i,w}) = 0, \forall n_w \in \mathcal{V}$, then the vulnerability $g_{j_0}(\{n_i\}, \gamma, \Delta k) \in [0,1]$ is an non-decreasing function regarding Δk for all target node $n_{j_0} \in \mathcal{V} \setminus \mathcal{V}_I, n_i \in \mathcal{V}_I$. The value of $g_{j_0}(\{n_i\}, \gamma, \Delta k)$ does not increase to 1 as Δk increases to infinity if and only if $\beta_{i,j_0}\lambda_{i,j_0} = 0$ and $g_{j_0}(\{n_i\} \cup v, \gamma, \Delta k - 1) = g_{j_0}(\{n_i\}, \gamma, \Delta k - 1), \forall v \in \mathcal{V}_S, \forall \Delta k \in \mathbb{Z}^+$.*

Proof. If $\gamma_{i,w}(1-q_{i,w}) = 0, \forall n_w \in \mathcal{V}$, we can use the facts that $g_{j_0}(\{n_i\} \cup v, \gamma, \Delta k - 1) \geq g_{j_0}(\{n_i\}, \gamma, \Delta k - 1), \forall \gamma, n_{j_0} \in \mathcal{V}, n_i \in \mathcal{V}_I, \Delta k \geq 0, \forall v \in \mathcal{V}_S$, and

$$\sum_{v \in \mathcal{V}_{i,j_0}} \sum_{u \in \mathcal{V}_{i,j_0}^v} f_{v,u}(\beta, \lambda) \equiv 1, \forall \beta, \lambda,$$

to obtain $g_{j_0}(\{n_i\}, \gamma, \Delta k)$ as

$$\begin{aligned} &\beta_{i,j_0}\lambda_{i,j_0} + (1-\beta_{i,j_0}\lambda_{i,j_0}) \sum_{v \in \mathcal{V}_{i,j_0}} \sum_{u \in \mathcal{V}_{i,j_0}^v} f_{v,u}(\beta, \lambda) g_{j_0}(\{n_i\} \cup v, \gamma, \Delta k - 1) \\ &\geq \beta_{i,j_0}\lambda_{i,j_0} + (1-\beta_{i,j_0}\lambda_{i,j_0}) g_{j_0}(\{n_i\}, \gamma, \Delta k - 1) \geq g_{j_0}(\{n_i\}, \gamma, \Delta k - 1), \end{aligned} \tag{7.8}$$

for all $\Delta k \in \mathbb{Z}^+$. The inequality is an equality if and only if $\beta_{i,j_0}\lambda_{i,j_0} = 0$ and

$g_{j_0}(\{n_i\} \cup v, \gamma, \Delta k - 1) = g_{j_0}(\{n_i\}, \gamma, \Delta k - 1), \forall v \in \mathcal{V}_S, \forall \Delta k \in \mathbb{Z}^+$. □

The equation $g_{j_0}(\{n_i\} \cup v, \gamma, \Delta k - 1) = g_{j_0}(\{n_i\}, \gamma, \Delta k - 1), \forall v \in \mathcal{V}_S, \forall \Delta k \in \mathbb{Z}^+$, holds only under very unlikely conditions such as there is only one node in the network, i.e., $N = 1$ or service links occur only from node n_i, i.e., $\lambda_{i',j} = 0, \forall i' \neq i, \forall n_j \in \mathcal{V}$. Thus, except for these rare special cases, the vulnerability $g_{j_0}(\{n_i\}, \gamma, \Delta k)$ always increases to the maximum value of 1 under indirect policies.

Remark 11. *Proposition 2 shows that without a proper mitigation strategy, e.g., no direct honey link from the initial intrusion node to the honeypot, the vulnerability of a target node never decreases over stages. Moreover, except from rare special cases, the target node will be compromised with probability* 1 *as time goes to infinity.*

Proposition 2 demonstrates the disadvantaged position of the defender against persistent lateral movement without proper honeypot policies. Under these disadvantageous situations, the defender may need alternative security measures to mitigate the LTV. For example, the defender may reduce the arrival frequency of the service link from n_{j_1} to n_{j_2}, i.e., β_{j_1,j_2}, to delay lateral movement at the expenses of operational efficiency. Also, the defender may attempt to reduce the probability of a successful compromise from node n_{j_1} to n_{j_2}, i.e., λ_{j_1,j_2}, by filtering the service link from n_{j_1} to n_{j_2} with more stringent rules or demotivate the attacker to initiate the link compromise by disguising the service link as a honey link. In the rest of this subsection, we briefly investigate the influence of β and λ on the Δk-stage vulnerability under indirect policies.

The probability of no direct link from the initial intrusion node n_i to target n_{j_0}, i.e., $1 - \beta_{i,j_0}\lambda_{i,j_0}$, and the probability that the attacker at node n_i is demotivated to or fails to compromise the service links from node n_i, i.e., $\sum_{u \in \mathcal{V}^{\emptyset}_{i,j_0}} f_{\emptyset,u}(\beta, \lambda)$, defines the *Probability of Movement Deterrence (**PoMD**)*

$$r := (1 - \beta_{i,j_0}\lambda_{i,j_0}) \sum_{u \in \mathcal{V}^{\emptyset}_{i,j_0}} f_{\emptyset,u}(\beta, \lambda)$$

. In (7.8) where the PoIC is 0, i.e., $\gamma_{i,w}(1 - q_{i,w}) = 0, \forall n_w \in \mathcal{V}$, we can upper bound the term $g_{j_0}(\{n_i\} \cup v, \gamma, \Delta k - 1)$ by 1 for all $v \neq \emptyset$, which leads to

$$\begin{aligned} g_{j_0}(\{n_i\}, \gamma, \Delta k) &= (1 - r) \cdot g_{j_0}(\{n_i\} \cup v, \gamma, \Delta k - 1) + r \cdot g_{j_0}(\{n_i\}, \gamma, \Delta k - 1) \\ &\leq (1 - r) + r \cdot g_{j_0}(\{n_i\}, \gamma, \Delta k - 1) \\ &= 1 - r^{\Delta k} + r^{\Delta k} g_{j_0}(\{n_i\}, \gamma, 0) = 1 - r^{\Delta k}(1 - \beta_{i,j_0}\lambda_{i,j_0}), \end{aligned} \tag{7.9}$$

where the final line results from solving the first-order linear difference equation iteratively by $\Delta k - 1$ times.

Equation (7.9) shows that the upper bound of LTV increases exponentially concerning the duration of lateral movement Δk yet decreases in a polynomial growth rate as PoMD increases. Note that letting PoMD be 1 can completely deter lateral movement and achieve zero LTV for any $\Delta k \in \mathbb{Z}^+$. However, it is challenging to attain it as it requires the attacker do not succeed from n_i to any node n_j with probability 1, i.e., $\lambda_{i,j} = 0, \forall n_j \in \mathcal{V}$. Since increasing PoMD incurs a higher cost (e.g., reducing the compromise rate λ) and lower operational efficiency (e.g., reducing the frequency of service links β), we aim to find the minimum PoMD to mitigate LTV even when the duration of lateral movement Δk goes to infinity. In Proposition 3, we characterize the critical *Threshold of Compromisability (**ToC**)* $T^{ToC}_m := 1 - m/\Delta k$ for a positive $m \ll \Delta k$ to guarantee a level-$(\beta_{i,j_0}\lambda_{i,j_0})$, stage-$\infty$ security defined in Definition 21. The proof follows directly from a limit analysis based on (7.9).

Proposition 3 (ToC). *Consider the scenario where $\gamma_{i,w}(1-q_{i,w}) = 0, \forall n_w \in \mathcal{V}$, and r as a function of Δk has the form $r = 1 - m\Delta k^{-n}$, where $n, m \in \mathbb{R}^+$ and $m \ll \Delta k$.*

(1). If $(1-r)/m$ is of the same order with $1/\Delta k$, i.e., $n = 1$, then the limit of the upper bound $\lim_{\Delta k \to \infty} 1 - r^{\Delta k}(1 - \beta_{i,j_0}\lambda_{i,j_0})$ is a constant $1 - e^{-m}(1 - \beta_{i,j_0}\lambda_{i,j_0})$.

(2). If $(1-r)/m$ is of higher order, i.e., $n > 1$, then the limit of the upper bound is $g_{j_0}(\{n_i\}, \gamma, 0) = \beta_{i,j_0}\lambda_{i,j_0}$. If $\beta_{i,j_0}\lambda_{i,j_0} = 0$, zero LTV is achieved $g_{j_0}(\{n_i\}, \gamma, \infty) = 0$.

(3). If $(1-r)/m$ is of lower order, i.e., $n < 1$, then the limit of the upper bound is 1.

Based on the fact that $1 - e^{-m}(1 - \beta_{i,j_0}\lambda_{i,j_0}) \geq \beta_{i,j_0}\lambda_{i,j_0}$ where the equality holds if and only if $\beta_{i,j_0}\lambda_{i,j_0} = 1$, we can conclude that if $r \geq T_m^{ToC}$ for a positive $m \ll \Delta k$, then the ∞-stage vulnerability of target node n_{j_0} is upper bounded by $\beta_{i,j_0}\lambda_{i,j_0}$ and thus achieves the level-($\beta_{i,j_0}\lambda_{i,j_0}$), stage-$\infty$ security as defined in Definition 21. Note that if the target node is segregated from nodes in DMZ $\mathcal{V}_I$ for the sake of security, then there is no direct service link from node n_i to the target node n_{j_0} and $\beta_{i,j_0}\lambda_{i,j_0} = 0$. In that case, the target node n_{j_0} can achieve a zero vulnerability for an infinite duration of lateral movement, i.e., $g_{j_0}(\{n_i\}, \gamma, \infty) = 0$, because the upper bound is 0 and LTV is always non-negative.

Direct Honeypot Policies

For the direct policies $\gamma_{i,w_0} = 1, n_{w_0} \in \mathcal{V} \setminus \{n_i, n_{j_0}\}$, we obtain the corresponding Δk-stage vulnerability and an explicit lower bound in (7.10) based on (7.5) by using the inequality $g_{j_0}(\{n_i\} \cup v, \gamma, \Delta k - 1) \geq g_{j_0}(\{n_i\}, \gamma, \Delta k - 1)$. Define shorthand notations $k_1 := \prod_{l \neq w_0}(1-\beta_{l,w_0})(1-\beta_{w_0,l})(1-q_{i,w_0}) \in [0,1]$ and $k_2 := \sum_{v \in \mathcal{V}_{i,j_0} \setminus \{n_{w_0}\}} \sum_{u \in \mathcal{V}^v_{i,j_0} \setminus \{w_0\}} f_{v,u}(\beta, \lambda) \leq \sum_{v \in \mathcal{V}_{i,j_0}} \sum_{u \in \mathcal{V}^v_{i,j_0}} f_{v,u}(\beta, \lambda) = 1$. Note that $k_1 = 0$ is a very restrictive condition as it requires that the honeypot n_{w_0} is not interfering, i.e., node n_{w_0} is *idle* and the attacker never identify the honey link from n_i to n_{w_0}, i.e., $q_{i,w_0} = 0$.

$$
\begin{aligned}
&g_{j_0}(\{n_i\}, \gamma, \Delta k) = \beta_{i,j_0}\lambda_{i,j_0}[1 - \prod_{l \neq w_0}(1-\beta_{l,w_0})(1-\beta_{w_0,l})(1-q_{i,w_0})] + \\
&(1-\beta_{i,j_0}\lambda_{i,j_0})[\sum_{v \in \mathcal{V}_{i,j_0}} \sum_{u \in \mathcal{V}^v_{i,j_0}} f_{v,u}(\beta,\lambda) g_{j_0}(\{n_i\} \cup v, \gamma, \Delta k - 1) - \sum_{v \in \mathcal{V}_{i,j_0} \setminus \{n_{w_0}\}} \sum_{u \in \mathcal{V}^v_{i,j_0} \setminus \{n_{w_0}\}} \\
&f_{v,u}(\beta,\lambda) \cdot \prod_{l \neq i, w_0}(1-\beta_{l,w_0}) \prod_{l' \neq w_0}(1-\beta_{w_0,l'})(1-q_{i,w_0}) g_{j_0}(\{n_i\} \cup v, \gamma, \Delta k - 1)] \\
&\geq \beta_{i,j_0}\lambda_{i,j_0}(1-k_1) + (1-\beta_{i,j_0}\lambda_{i,j_0})[1 - k_1 k_2 (1-\beta_{i,w_0})] g_{j_0}(\{n_i\}, \gamma, \Delta k - 1).
\end{aligned}
\tag{7.10}
$$

Define a shorthand notation $r_2 := (1-\beta_{i,j_0}\lambda_{i,j_0})[1 - k_1 k_2 (1-\beta_{i,w_0})]$, we can solve the linear difference equation in the final step of (7.10) to obtain an lower bound, i.e., $g_{j_0}(\{n_i\}, \gamma, \Delta k) \geq T_2^{lower,1} := \beta_{i,j_0}\lambda_{i,j_0}(1-k_1)\frac{1-(r_2)^{\Delta k+1}}{1-r_2}$ for all $\Delta k \in \mathbb{Z}^+$. According to the first equality in (7.10), we also obtain an upper bound T_2^{upper} for $g_{j_0}(\{n_i\}, \gamma, \Delta k), \forall \Delta k \in \mathbb{Z}^+$, in Lemma 2 by using the inequality $g_{j_0}(\{n_i\} \cup v, \gamma, \Delta k) \leq 1, \forall v \in \mathcal{V}_{i,j_0}$[3]. The bound $T_2^{upper} < 1$ is non-trivial if $\beta_{i,j_0}\lambda_{i,j_0} \neq 0, \beta_{i,j_0}\lambda_{i,j_0} \neq 1$, and $k_1 k_2 (1-\beta_{i,w_0}) \neq 0$.

Lemma 2. *If $\gamma_{i,w_0} = 1, w_0 \neq i, j_0$, then $g_{j_0}(\{n_i\}, \gamma, \Delta k)$ is lower and upper bounded by $T_2^{lower,1}$ and $T_2^{upper} := 1 - \beta_{i,j_0}\lambda_{i,j_0}k_1 - (1-\beta_{i,j_0}\lambda_{i,j_0})k_1 k_2 (1-\beta_{i,w_0}) \in [0,1]$ for all $\Delta k \in \mathbb{Z}^+$, respectively.*

[3] Since we can compute $g_{j_0}(\{n_i\} \cup v, \gamma, \Delta k - 1)$ explicitly when v is empty, we can obtain a tighter upper bound by using the inequality $g_{j_0}(\{n_i\} \cup v, \gamma, \Delta k) \leq 1, \forall v \in \mathcal{V}_{i,j_0} \setminus \emptyset$.

Lemma 2 shows that if the defender applies a direct honeypot from n_i in a deterministic fashion, then the Δk-stage vulnerability is always upper bounded. However, these direct policies cannot reduce the ∞-stage vulnerability to zero as shown in Proposition 4.

Proposition 4 (Vulnerability Residue)**.** *If $\beta_{i,j_0}\lambda_{i,j_0} \neq 0$ and $\gamma_{i,w_0} = 1, w_0 \neq i, j_0$, then*

(1). The term $T_2^{lower,2} := \frac{\beta_{i,j_0}\lambda_{i,j_0}(1-k_1)}{(1-\beta_{i,j_0}\lambda_{i,j_0})k_1k_2(1-\beta_{i,w_0})+\beta_{i,j_0}\lambda_{i,j_0}} \in [0,1)$ is strictly less than 1.

(2). If $g_{j_0}(\{n_i\}, \gamma, \Delta k - 1) < T_2^{lower,2}$, then $g_{j_0}(\{n_i\}, \gamma, \Delta k) > g_{j_0}(\{n_i\}, \gamma, \Delta k - 1)$.

(3). $\lim_{\Delta k \to \infty} g_{j_0}(\{n_i\}, \gamma, \Delta k)$ is lower bounded by $\max(T_2^{lower,1}, T_2^{lower,2})$.

Proof. Based on the inequality in (7.10), we obtain that if $g_{j_0}(\{n_i\}, \gamma, \Delta k - 1) < T_2^{lower,2}$, then $g_{j_0}(\{n_i\}, \gamma, \Delta k) > g_{j_0}(\{n_i\}, \gamma, \Delta k - 1)$. Since the above is true for all $\Delta k \in \mathbb{Z}^+$, we know that the Δk-stage vulnerability increases with Δk strictly until it has reach $T_2^{lower,2}$. If $\beta_{i,j_0}\lambda_{i,j_0} \neq 0$ and $k_1 \neq 1$, then $T_2^{lower,2} > 0$ is a non-trivial lower bound. The other lower bound $T_2^{lower,1}$ comes from Lemma 2. □

Remark 12. *Proposition 4 defines a vulnerability residue*

$$T^{VR} := \max(T_2^{lower,1}, T_2^{lower,2})$$

under direct honeypot policies. A nonzero T^{VR} characterizes the limitation of security policies against lateral movement attacks, i.e., LTV cannot be reduced to 0 as $\Delta k \to \infty$.

Chapter 8

Adaptive Honeypot Engagement for Threat Intelligence

Following Section 1.4.2, defenders adopting reactive defense mechanisms suffer from information disadvantages. Off-the-shelf defense can detect low-level Indicators of Compromise (IoCs), such as hash values, IP addresses, and domain names. However, they can hardly disclose high-level indicators such as attack tools and Tactics, Techniques, and Procedures (TTPs) of the attacker, which induces the attacker fewer pains to adapt to the defense mechanism, evade the indicators, and launch revised attacks, as shown in Figure 8.1. Since high-level threat intelligence is more effective in deterring emerging advanced attacks yet harder to acquire through the traditional passive mechanism, defenders need to adopt proactive defense paradigms to learn these fundamental characteristics of the attacker, attribute cyber attacks [178], and design defensive countermeasures correspondingly.

Honeypots are one of the most frequently employed active defense techniques to gather information on threats. A honeynet is a network of honeypots, which

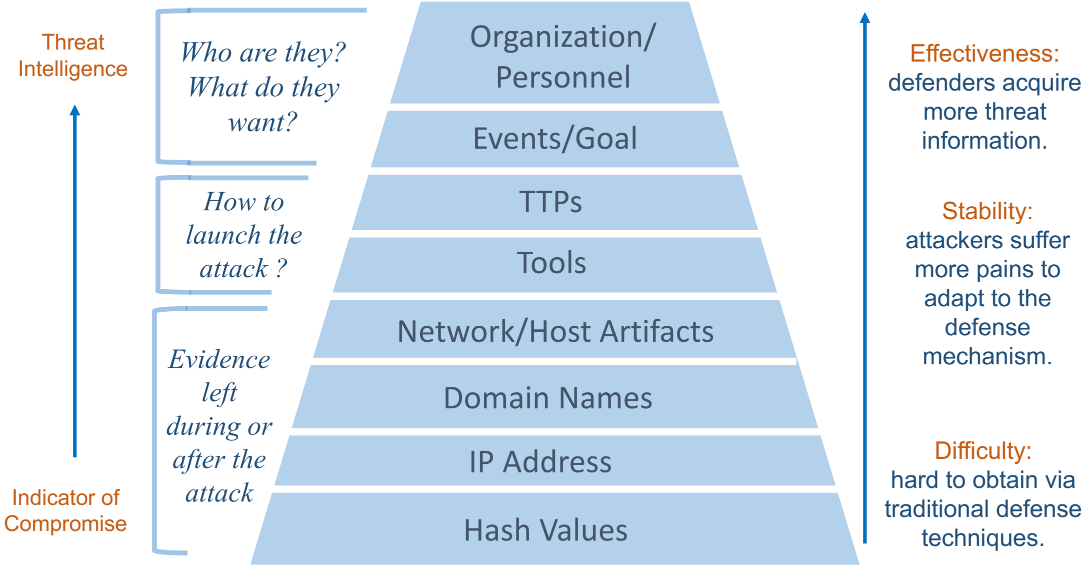

Threat Intelligence
Indicator of Compromise
Who are they? What do they want?
How to launch the attack ?
Evidence left during or after the attack
Organization/ Personnel
Events/Goal
TTPs
Tools
Network/Host Artifacts
Domain Names
IP Address
Hash Values
Effectiveness: defenders acquire more threat information.
Stability: attackers suffer more pains to adapt to the defense mechanism.
Difficulty: hard to obtain via traditional defense techniques.

Figure 8.1: The transformation from IoCs to threat intelligence increases the difficulty, stability, and effectiveness. IoCs focus on the evidence during or after the attack, while threat intelligence identifies the attack tools, attack goals, and the personnel who launches the attack.

emulates the real production system but has no production activities nor authorized services. Thus, an interaction with a honeynet, e.g., unauthorized inbound connections to any honeypot, directly reveals malicious activities. On the contrary, traditional passive techniques, such as firewall logs or IDSs, have to separate attacks from a ton of legitimate activities, thus providing many more false alarms and may still miss some unknown attacks.

Besides a more effective identification and denial of adversarial exploitation through low-level indicators such as the inbound traffic, a honeynet can also help defenders to achieve the goal of identifying attackers' TTPs under proper engagement actions. The defender can interact with attackers and allow them to probe and perform in the honeynet until she has learned the attacker's fundamental characteristics. More services a honeynet emulates, more activities an attacker is allowed to perform, and a higher degree of interactions together result in a larger revelation probability of the attacker's TTPs. However, the additional services and reduced restrictions also bring extra risks. Attacks may use some honeypots as pivot nodes to launch attackers against other production systems [200].

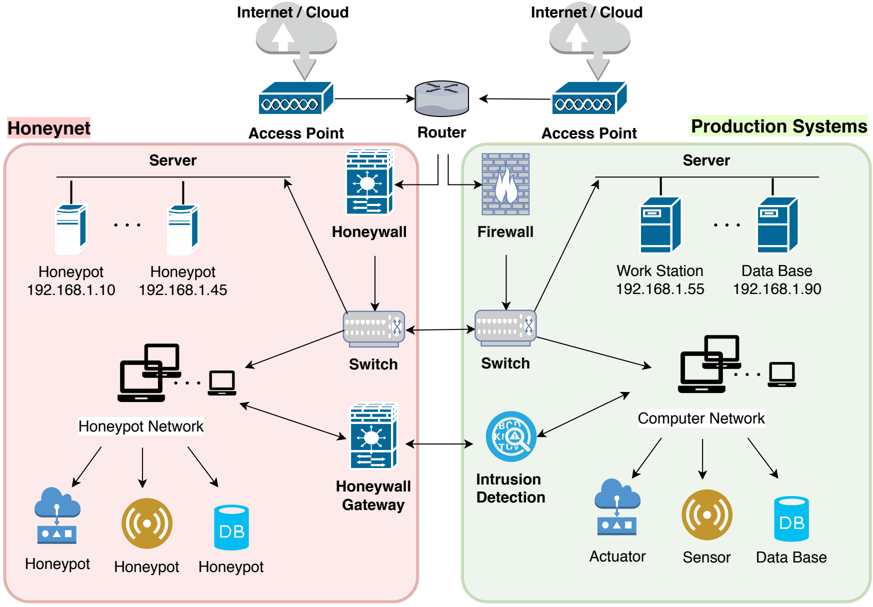


Figure 8.2: The honeynet in red mimics the targeted production system in green. The honeynet shares the same structure as the production system yet has no authorized services.

The current honeynet applies the honeywall as a gateway device to supervise outbound data and separate the honeynet from other production systems, as shown in Fig. 8.2. However, to avoid attackers' identification of the data control and the honeynet, a defender cannot block all outbound traffics from the honeynet, which leads to a trade-off between the rewards of learning high-level IoCs and the following three types of risks.

T1: Attackers identify the honeynet and thus either terminate on their own or generate misleading interactions with honeypots.

T2: Attackers circumvent the honeywall to penetrate production systems [172].

T3: Defender's engagement costs outweigh the investigation reward.

We quantify risk T1 in Section 8.1.3, T2 in Section 8.1.5, and T3 in Section 8.1.4, respectively. In particular, risk T3 brings the problem of timeliness and optimal decisions on timing. Since a persistent traffic generation to engage attackers is costly and the defender aims to obtain timely threat information, the defender needs cost-effective policies to lure the attacker quickly to the target honeypot and reduce attacker's sojourn time in honeypots of low-investigation value.

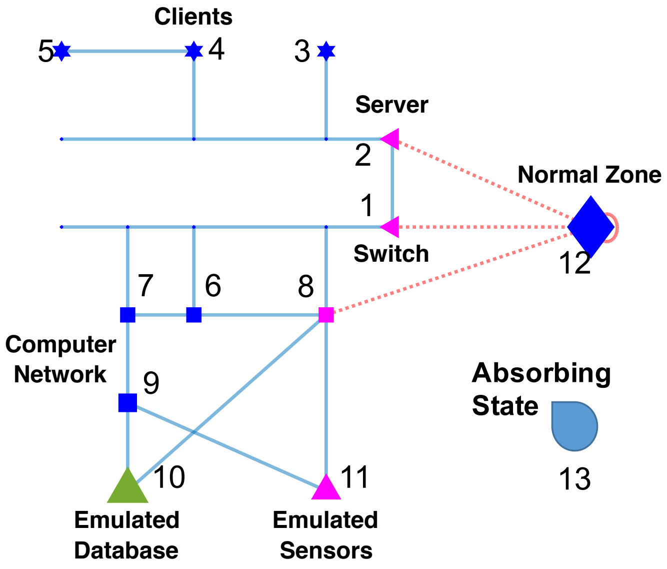


Figure 8.3: Honeypots emulate different components of the production system.

To achieve the goal of long-term, cost-effective policies, we construct the Semi-Markov Decision Process (SMDP) in Section 8.1 on the network shown in Fig. 8.3.

Nodes 1 to 11 represent different types of honeypots, nodes 12 and 13 represent the domain of the production system and the virtual absorbing state, respectively. The attacker transits between these nodes according to the network topology in Fig. 8.2 and can remain at different nodes for an arbitrary period of time. The defender can dynamically change the honeypots' engagement levels (e.g., the amount of outbound traffic) to affect the attacker's sojourn time, engagement rewards, and the probabilistic transition in that honeypot.

8.1 Problem Formulation

To obtain optimal engagement decisions at each honeypot under the probabilistic transition and the continuous sojourn time, we introduce the continuous-time infinite-horizon discounted SMDPs, which can be summarized by the tuple $\{t \in [0,\infty), \mathcal{S}, \mathcal{A}(s_j), tr(s_l|s_j,a_j), z(\cdot|s_j,a_j,s_l), r^{\gamma}(s_j,a_j,s_l), \gamma \in [0,\infty)\}$. We describe each element of the tuple in this section.

8.1.1 Network Topology

We abstract the structure of the honeynet as a finite graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$. The node set $\mathcal{N} := \{n_1, n_2, \cdots, n_N\} \cup \{n_{N+1}\}$ contains N nodes of hybrid honeypots. Take Fig. 8.3 as an example, a node can be either a virtual honeypot of an integrated database system or a physical honeypot of an individual computer. These nodes provide different types of functions and services, and are connected following the topology of the emulated production system. Since we focus on optimizing the value of investigation in the honeynet, we only distinguish between different types of honeypots in different shapes, yet use one extra node n_{N+1} to represent the entire domain of the production system. The network topology $\mathcal{E} := \{e_{jl}\}, j, l \in \mathcal{N}$, is the set of directed links connecting node n_j with n_l, and represents all possible transition trajectories in the honeynet. The links can be either physical (if the connecting nodes are real facilities such as computers) or logical (if the nodes represent integrated systems). Attackers cannot break the topology restriction. Since an attacker may use some honeypots as pivots to reach a production system, and it is also possible for a defender to attract attackers from the normal zone to the honeynet through these bridge nodes, there exist links of both directions between honeypots and the normal zone.

8.1.2 States and State-Dependent Actions

At time $t \in [0, \infty)$, an attacker's state belongs to a finite set $\mathcal{S} := \{s_1, \cdots, s_N, s_{N+1}, s_{N+2}\}$ where $s_i, i \in \{1, \cdots, N+1\}$, represents the attacker's location at time t. Once attackers are ejected or terminate on their own, we use the extra absorbing state s_{N+2} to represent the virtual location. The attacker's state reveals the adversary visit and exploitation of the emulated functions and services. Since the honeynet provides a controlled environment, we assume that the defender can monitor the state and transitions persistently without uncertainties. The attacker can visit a node multiple times for different purposes. A stealthy attacker may visit the honeypot node of the database more than once and revise data progressively (in a small amount each time) to evade detection. An attack on the honeypot node of sensors may need to frequently check the node for the up-to-date data. Some advanced honeypots may also emulate anti-virus systems or other protection mechanisms such as setting up an authorization expiration time, then the attacker has to compromise the nodes repeatedly.

At each state $s_i \in \mathcal{S}$, the defender can choose an action a_i from a state-dependent finite set $\mathcal{A}(s_i)$. For example, at each honeypot node, the defender can conduct action a_E to eject the attacker, action a_P to purely record the attacker's activities, low-interactive action a_L, or high-interactive action a_H to engage the attacker, i.e., $\mathcal{A}(s_i) := \{a_E, a_P, a_L, a_H\}, i \in \{1, \cdots, N\}$. The high-interactive action is costly to implement yet both increases the probability of a longer sojourn time at honeypot n_i, and reduces the probability of attackers penetrating the normal system from n_i if connected. If the attacker resides in the normal zone either from the beginning or later through the pivot honeypots, the defender can choose either action a_E to eject the attacker immediately, or action a_A to attract the attacker to the honeynet by exposing some vulnerabilities intentionally, i.e., $\mathcal{A}(s_{N+1}) := \{a_E, a_A\}$. Note that the instantiation of the action set and the corresponding consequences are not limited to the above scenario. For example, the action can also refer to a different degree of outbound data control. A strict control reduces the probability of attackers penetrating the normal system from the honeypot, yet also brings less investigation value.

8.1.3 Continuous-Time Process and Discrete Decision

Based on the current state $s_j \in \mathcal{S}$, the defender's action $a_j \in \mathcal{A}(s_j)$, the attacker transits to state $s_l \in \mathcal{S}$ with a probability $tr(s_l|s_j, a_j)$ and the sojourn time at state s_j is a continuous random variable with a probability density $z(\cdot|s_j, a_j, s_l)$. Note that the risk T1 of the attacker identifying the honeynet at state s_j under action $a_j \neq A_E$ can be characterized by the transition probability $tr(s_{N+2}|s_j, a_j)$ as well as the duration time $z(\cdot|s_j, a_j, s_{N+2})$. Once the attacker arrives at a new honeypot n_i, the defender dynamically applies an interaction action at honeypot

n_i from $\mathcal{A}(s_i)$ and keeps interacting with the attacker until he transits to the next honeypot. The defender may not change the action before the transition to reduce the probability of attackers detecting the change and become aware of the honeypot engagement. Since the decision is made at the time of transition, we can transform the above continuous time model on horizon $t \in [0, \infty)$ into a discrete decision model at decision epoch $k \in \{0, 1, \cdots, \infty\}$. The time of the attacker's k^{th} transition is denoted by a random variable T^k, the landing state is denoted as $s^k \in \mathcal{S}$, and the adopted action after arriving at s^k is denoted as $a^k \in \mathcal{A}(s^k)$.

8.1.4 Investigation Value

The defender gains a reward of investigation by engaging and analyzing the attacker in the honeypot. To simplify the notation, we divide the reward during time $t \in [0, \infty)$ into ones at discrete decision epochs $T^k, k \in \{0, 1, \cdots, \infty\}$. When $\tau \in [T^k, T^{k+1}]$ amount of time elapses at stage k, the defender's reward of investigation

$$r(s^k, a^k, s^{k+1}, T^k, T^{k+1}, \tau) = r_1(s^k, a^k, s^{k+1})\mathbf{1}_{\{\tau=0\}} + r_2(s^k, a^k, T^k, T^{k+1}, \tau), \quad (8.1)$$

at time τ of stage k, is the sum of two parts. The first part is the immediate cost of applying engagement action $a^k \in \mathcal{A}(s^k)$ at state $s^k \in \mathcal{S}$ and the second part is the reward rate of threat information acquisition minus the cost rate of persistently generating deceptive traffics. Due to the randomness of the attacker's behavior, the information acquisition can also be random, thus the actual reward rate r_2 is perturbed by an additive zero-mean noise w_r.

Different types of attackers target different components of the production system. For example, an attacker who aims to steal data will take intensive adversarial actions at the database. Thus, if the attacker is actually in the honeynet and adopts the same behavior as he is in the production system, the defender can identify the target of the attack based on the traffic intensity. We specify r_1 and r_2 at each state properly to measure the risk T3. To maximize the value of the investigation, the defender should choose proper actions to lure the attacker to the honeypot emulating the target of the attacker in a short time and with a large probability. Moreover, the defender's action should be able to engage the attacker in the target honeypot actively for a longer time to obtain more valuable threat information. We compute the optimal long-term policy that achieves the above objectives in Section 8.1.5.

As the defender spends longer time interacting with attackers, investigating their behaviors and acquires better understandings of their targets and TTPs, less new information can be extracted. In addition, the same intelligence becomes less valuable as time elapses due to the timeliness. Thus, we use a discounted factor of $\gamma \in [0, \infty)$ to penalize the decreasing value of the investigation as time elapses.

8.1.5 Optimal Long-Term Policy

The defender aims at a policy $\pi \in \Pi$ which maps state $s^k \in \mathcal{S}$ to action $a^k \in \mathcal{A}(s^k)$ to maximize the long-term expected utility starting from state s^0, i.e.,

$$u(s^0, \pi) = \mathbb{E}\left[\sum_{k=0}^{\infty} \int_{T^k}^{T^{k+1}} e^{-\gamma(\tau+T^k)}(r(S^k, A^k, S^{k+1}, T^k, T^{k+1}, \tau) + w_r)d\tau\right].$$

At each decision epoch, the value function $v(s^0) = \sup_{\pi \in \Pi} u(s^0, \pi)$ can be represented by dynamic programming, i.e.,

$$v(s^0) = \sup_{a^0 \in \mathcal{A}(s^0)} \mathbb{E}\left[\int_{T^0}^{T^1} e^{-\gamma(\tau+T^0)} r(s^0, a^0, S^1, T^0, T^1, \tau)d\tau + e^{-\gamma T^1} v(S^1)\right]. \quad (8.2)$$

We assume a constant reward rate $r_2(s^k, a^k, T^k, T^{k+1}, \tau) = \bar{r}_2(s^k, a^k)$ for simplicity. Then, (8.2) can be transformed into an equivalent MDP form, i.e., $\forall s^0 \in \mathcal{S}$,

$$v(s^0) = \sup_{a^0 \in \mathcal{A}(s^0)} \sum_{s^1 \in \mathcal{S}} tr(s^1|s^0, a^0)(r^\gamma(s^0, a^0, s^1) + z^\gamma(s^0, a^0, s^1)v(s^1)), \quad (8.3)$$

where $z^\gamma(s^0, a^0, s^1) := \int_0^\infty e^{-\gamma\tau} z(\tau|s^0, a^0, s^1)d\tau \in [0, 1]$ is the Laplace transform of the sojourn probability density $z(\tau|s^0, a^0, s^1)$ and the equivalent reward

$$r^\gamma(s^0, a^0, s^1) := r_1(s^0, a^0, s^1) + \frac{\bar{r}_2(s^0, a^0)}{\gamma}(1 - z^\gamma(s^0, a^0, s^1)) \in [-m_c, m_c]$$

is assumed to be bounded by a constant m_c.

A classical regulation condition of SMDP to avoid the probability of an infinite number of transitions within a finite time is stated as follows: there exists constants $\theta \in (0, 1)$ and $\delta > 0$ such that

$$\sum_{s^1 \in \mathcal{S}} tr(s^1|s^0, a^0)z(\delta|s^0, a^0, s^1) \leq 1 - \theta, \forall s^0 \in \mathcal{S}, a^0 \in \mathcal{A}(s^0). \quad (8.4)$$

It is shown in [80] that condition (8.4) is equivalent to

$$\sum_{s^1 \in \mathcal{S}} tr(s^1|s^0, a^0)z^\gamma(s^0, a^0, s^1) \in [0, 1),$$

which serves as the equivalent stage-varying discounted factor for the associated

MDP. Then, the right-hand side of (8.2) is a contraction mapping and there exists a unique optimal policy $\pi^* = arg \max_{\pi \in \Pi} u(s^0, \pi)$ which can be found by value iteration, policy iteration or linear programming.

Cost-Effective Policy

The computation result of our 13-state example system is illustrated in Fig. 8.3. The optimal policies at honeypot nodes n_1 to n_{11} are represented by different colors. Specifically, actions a_E, a_P, a_L, a_H are denoted in red, blue, purple, and green, respectively. The size of node n_i represents the state value $v(s_i)$.

In the example scenario, the honeypot of database n_{10} and sensors n_{11} are the main and secondary targets of the attacker, respectively. Thus, defenders can obtain a higher investigation value when they manage to engage the attacker in these two honeypot nodes with a larger probability and for a longer time. However, instead of naively adopting high interactive actions, a savvy defender also balances the high implantation cost of a_H. Our quantitative results indicate that the high interactive action should only be applied at n_{10} to be cost-effective. On the other hand, although the bridge nodes n_1, n_2, n_8 which connect to the normal zone n_{12} do not contain higher investigation values than other nodes, the defender still takes action a_L at these nodes. The goal is to either increase the probability of attracting attackers away from the normal zone or reduce the probability of attackers penetrating the normal zone from these bridge nodes.

Engagement Safety versus Investigation Values

Restrictive engagement actions endow attackers less freedom so that they are less likely to penetrate the normal zone. However, restrictive actions also decrease the probability of obtaining high-level IoCs, thus reduces the investigation values.

To quantify the system value under the trade-off of the engagement safety and the reward from the investigation, we visualize the trade-off surface in Fig. 8.4. In the x-axis, a larger penetration probability $p(s_{N+1}|s_j, a_j), j \in \{s_1, s_2, s_8\}, a_j \neq a_E$, decreases the value $v(s_{10})$. In the y-axis, a larger reward $r^{\gamma}(s_j, a_j, s_l), j \in \mathcal{S} \setminus \{s_{12}, s_{13}\}, l \in \mathcal{S}$, increases the value. The figure also shows that value $v(s_{10})$ changes in a higher rate, i.e., are more sensitive when the penetration probability is small and the reward from the investigation is large. In our scenario, the penetration probability has less influence on the value than the investigation reward, which motivates a less restrictive engagement.

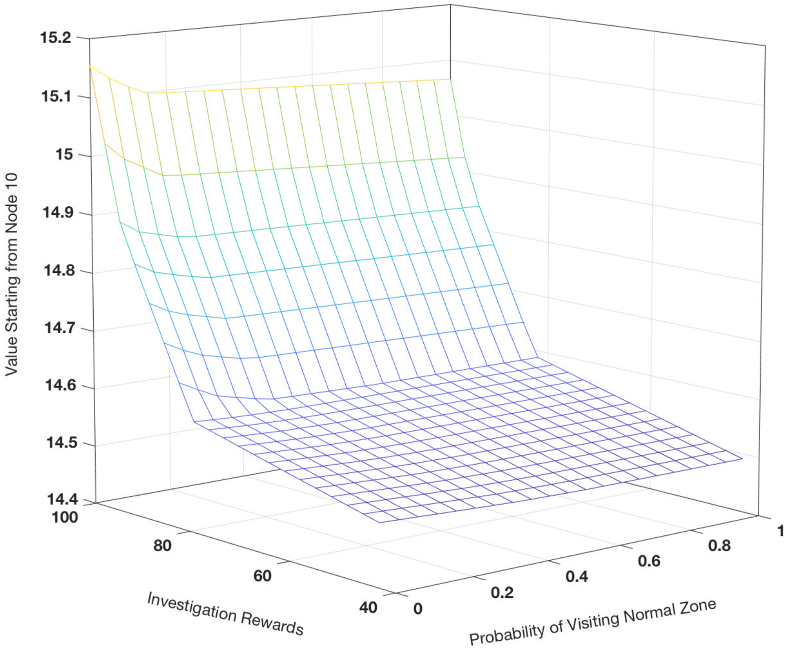


Figure 8.4: The trade-off surface of $v(s_{10})$ in z-axis under different values of penetration probability $p(s_{N+1}|s_j, a_j), j \in \{s_1, s_2, s_8\}, a_j \neq a_E$, in x-axis, and the reward $r^{\gamma}(s_j, a_j, s_l), j \in \mathcal{S} \setminus \{s_{12}, s_{13}\}, l \in \mathcal{S}$, in y-axis.

8.2 Risk Assessment

Given any feasible engagement policy $\pi \in \Pi$, the SMDP becomes a semi-Markov process [151]. We analyze the evolution of the occupancy distribution and first passage time in Section 8.2.1 and 8.2.2, respectively, which leads to three security metrics during the honeypot engagement. To shed lights on the defense of APTs, we investigate the system performance against attackers with different levels of persistence and intelligence in Section 8.2.3.

8.2.1 Transition Probability of Semi-Markov Process

Define the cumulative probability $q_{ij}(t)$ of the one-step transition from $\{S^k = i, T^k = t^k\}$ to $\{S^{k+1} = j, T^{k+1} = t^k + t\}$ as $\Pr(S^{k+1} = j, T^{k+1} - t^k \leq t | S^k = i, T^k = t^k) = tr(j|i,\pi(i)) \int_0^t z(\tau|i,\pi(i),j)d\tau, \forall i,j \in \mathcal{S}, t \geq 0$. Based on a variation of the forward Kolmogorov equation where the one-step transition lands on an intermediate state $l \in \mathcal{S}$ at time $T^{k+1} = t^k + u, \forall u \in [0,t]$, the transition probability of the system in state j at time t, given the initial state i at time 0 can be represented as

$$
\begin{aligned}
p_{ii}(t) &= 1 - \sum_{h\in\mathcal{S}} q_{ih}(t) + \sum_{l\in\mathcal{S}} \int_0^t p_{li}(t-u)dq_{il}(u), \\
p_{ij}(t) &= \sum_{l\in\mathcal{S}} \int_0^t p_{lj}(t-u)dq_{il}(u) = \sum_{l\in\mathcal{S}} p_{lj}(t) \star \frac{dq_{il}(t)}{dt}, \forall i,j \in \mathcal{S}, j \neq i, \forall t \geq 0,
\end{aligned}
$$

where $1 - \sum_{h\in\mathcal{S}} q_{ih}(t)$ is the probability that no transitions happen before time t. We can easily verify that $\sum_{l\in\mathcal{S}} p_{il}(t) = 1, \forall i \in \mathcal{S}, \forall t \in [0,\infty)$. To compute $p_{ij}(t)$ and $p_{ii}(t)$, we can take Laplace transform and then solve two sets of linear equations.

For simplicity, we specify $z(\tau|i,\pi(i),j)$ to be exponential distributions with parameters $\lambda_{ij}(\pi(i))$, and the semi-Markov process degenerates to a continuous time Markov chain. Then, we obtain the infinitesimal generator via the Leibniz integral rule, i.e.,

$$\bar{q}_{ij} := \left.\frac{dp_{ij}(t)}{dt}\right|_{t=0} = \lambda_{ij}(\pi(i)) \cdot tr(j|i,\pi(i)) > 0, \forall i,j \in \mathcal{S}, j \neq i,$$
$$\bar{q}_{ii} := \left.\frac{dp_{ii}(t)}{dt}\right|_{t=0} = -\sum_{j\in\mathcal{S}\setminus\{i\}} \bar{q}_{ij} < 0, \forall i \in \mathcal{S}.$$

Define matrix $\bar{\mathbf{Q}} := [\bar{q}_{ij}]_{i,j\in\mathcal{S}}$ and vector $\mathbf{P}_i(t) = [p_{ij}(t)]_{j\in\mathcal{S}}$, then based on the forward Kolmogorov equation,

$$\frac{d\mathbf{P}_i(t)}{dt} = \lim_{u\to 0^+} \frac{\mathbf{P}_i(t+u) - \mathbf{P}_i(t)}{u} = \lim_{u\to 0^+} \frac{\mathbf{P}_i(u) - \mathbf{I}}{u}\mathbf{P}_i(t) = \bar{\mathbf{Q}}\mathbf{P}_i(t).$$

Thus, we can compute the first security metric, the *occupancy distribution* of any state $s \in \mathcal{S}$ at time t starting from the initial state $i \in \mathcal{S}$ at time 0, i.e.,

$$\mathbf{P}_i(t) = e^{\bar{\mathbf{Q}}t}\mathbf{P}_i(0), \forall i \in \mathcal{S}. \tag{8.5}$$

We plot the evolution of $p_{ij}(t), i = s_{N+1}, j \in \{s_1, s_2, s_{10}, s_{12}\}$, versus $t \in [0,\infty)$ in Fig. 8.5 and the limiting occupancy distribution $p_{ij}(\infty), i = s_{N+1}$, in Fig. 8.6. In Fig. 8.5, although the attacker starts at the normal zone $i = s_{N+1}$, our engagement policy can quickly attract the attacker into the honeynet. Fig. 8.6 demonstrates that the engagement policy can keep the attacker in the honeynet with a dominant probability of 91% and specifically, in the target honeypot n_{10} with a high probability of 41%. The honeypots connecting the normal zone also have a higher occupancy probability than nodes $n_3, n_4, n_5, n_6, n_7, n_9$, which are less likely to be explored by the attacker due to the network topology.

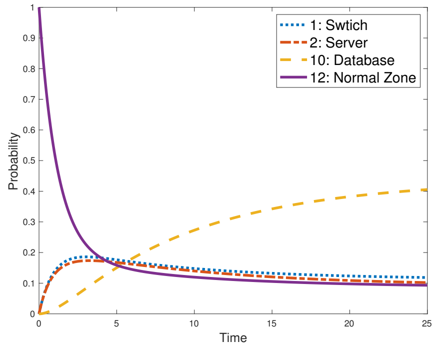


Figure 8.5: Evolution of $p_{ij}(t), i = s_{N+1}$.

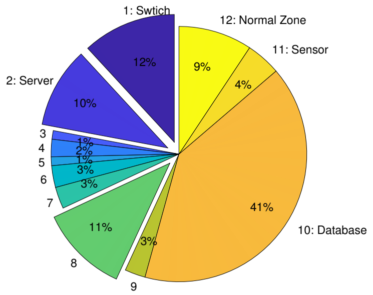


Figure 8.6: The limiting occupancy distribution.

8.2.2 First Passage Time

Another quantitative measure of interest is the first passage time $T_{i\mathcal{D}}$ of visiting a set $\mathcal{D} \subset \mathcal{S}$ starting from $i \in \mathcal{S} \setminus \mathcal{D}$ at time 0. Define the cumulative probability function $f^c_{i\mathcal{D}}(t) := \Pr(T_{i\mathcal{D}} \leq t)$, then $f^c_{i\mathcal{D}}(t) = \sum_{h \in \mathcal{D}} q_{ih}(t) + \sum_{l \in \mathcal{S} \setminus \mathcal{D}} \int_0^t f^c_{l\mathcal{D}}(t - u) dq_{il}(u)$. In particular, if $\mathcal{D} = \{j\}$, then the probability density function $f_{ij}(t) := \frac{df^c_{ij}(t)}{dt}$ satisfies

$$p_{ij}(t) = \int_0^t p_{jj}(t-u) df^c_{ij}(u) = p_{jj}(t) \star f_{ij}(t), \forall i, j \in \mathcal{S}, j \neq i.$$

Take Laplace transform $\bar{p}_{ij}(s) := \int_0^\infty e^{-st} p_{ij}(t) dt$, and then take inverse Laplace transform on $\bar{f}_{ij}(s) = \frac{\bar{p}_{ij}(s)}{\bar{p}_{jj}(s)}$, we obtain

$$f_{ij}(t) = \int_0^\infty e^{st} \frac{\bar{p}_{ij}(s)}{\bar{p}_{jj}(s)} ds, \forall i, j \in \mathcal{S}, j \neq i. \tag{8.6}$$

We define the second security metric, the *attraction efficiency* as the probability of the first passenger time $T_{s_{12},s_{10}}$ less than a threshold t_{th}. Based on (8.5) and (8.6), the probability density function of $T_{s_{12},s_{10}}$ is shown in Fig. 8.7. We take the mean denoted by the orange line as the threshold t_{th} and the attraction efficiency is 0.63, which means that the defender can attract the attacker from the normal zone to the database honeypot in less than $t_{th} = 20.7$ with a probability of 0.63.

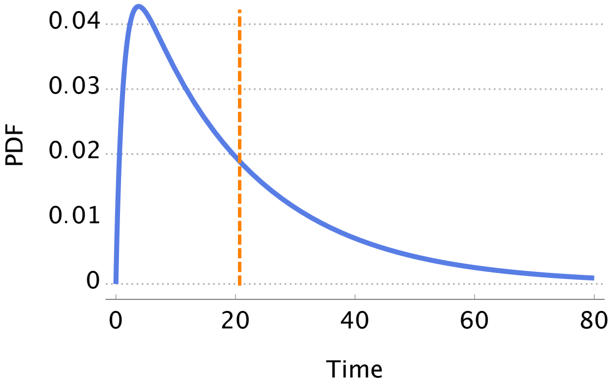


Figure 8.7: Probability density function of $T_{s_{12},s_{10}}$.

Mean First Passage Time

The third security metric of concern is the *average engagement efficiency* defined as the Mean First Passage Time (MFPT) $t^m_{i\mathcal{D}} = E[T_{i\mathcal{D}}], \forall i \in \mathcal{S}, \mathcal{D} \subset \mathcal{S}$. Under the exponential sojourn distribution, MFPT can be computed directly through the a system of linear equations, i.e.,

$$\begin{aligned} & t^m_{iD} = 0, i \in \mathcal{D}, \\ & 1 + \sum_{l \in \mathcal{S}} \bar{q}_{il} t^m_{l\mathcal{D}} = 0, i \notin \mathcal{D}. \end{aligned} \tag{8.7}$$

In general, the MFPT is asymmetric, i.e., $t^m_{ij} \neq t^m_{ji}, \forall i, j \in \mathcal{S}$. Based on (8.7), we compute the MFPT from and to the normal zone in Fig. 8.8 and Fig. 8.9, respectively. The color of each node indicates the value of MFPT. In Fig. 8.8, the honeypot nodes that directly connect to the normal zone have the shortest MFPT, and it takes attackers much longer time to visit the honeypots of clients due to the network topology. Fig. 8.9 shows that the defender can engage attackers in the target honeypot nodes of database and sensors for a longer time. The engagements at the client nodes are yet much less attractive. Note that two figures have different time scales denoted by the color bar value, and the comparison shows that it generally takes the defender more time and efforts to attract the attacker from the normal zone.

The MFPT from the normal zone $t^m_{s_{12},j}$ measures the average time it takes to attract attacker to honeypot state $j \in \mathcal{S} \setminus \{s_{12}, s_{13}\}$ for the first time. On the contrary, the MFPT to the normal zone $t^m_{i,s_{12}}$ measures the average time of the attacker penetrating the normal zone from honeypot state $i \in \mathcal{S} \setminus \{s_{12}, s_{13}\}$ for the first time. If the defender pursues absolute security and ejects the attack once it goes to the normal zone, then Fig. 8.9 also shows the attacker's average sojourn time in the honeynet starting from different honeypot nodes.

8.2.3 Advanced Persistent Threats

In this section, we quantify three engagement criteria on attackers of different levels of persistence and intelligence in Fig. 8.10 and Fig. 8.11, respectively. The criteria are the stationary probability of normal zone $p_{i,s_{12}}(\infty), \forall i \in \mathcal{S} \setminus \{s_{13}\}$, the utility of normal zone $v(s_{12})$, and the expected utility over the stationary probability, i.e., $\sum_{j \in \mathcal{S}} p_{ij}(\infty) v(j), \forall i \in \mathcal{S} \setminus \{s_{13}\}$.

As shown in Fig. 8.10, when the attacker is at the normal zone $i = s_{12}$ and

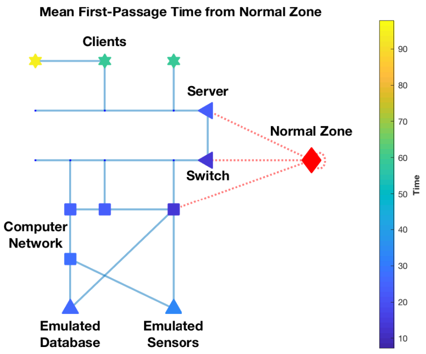


Figure 8.8: MFPT from the normal zone $t^m_{s_{12},j}$.

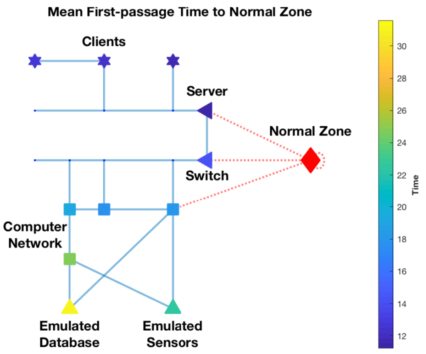


Figure 8.9: MFPT to the normal zone $t^m_{i,s_{12}}$.

the defender chooses action $a = a_A$, a larger $\lambda := \lambda_{ij}(a_A), \forall j \in \{s_1, s_2, s_8\}$, of the exponential sojourn distribution indicates that the attacker is more inclined to respond to the honeypot attraction and thus less time is required to attract the attacker away from the normal zone. As the persistence level λ increases from 0.1 to 2.5, the stationary probability of the normal zone decreases and the expected utility over the stationary probability increases, both converge to their stable values. The change rate is higher during $\lambda \in (0, 0.5]$ and much lower afterward. On the other hand, the utility loss at the normal zone decreases approximately linearly during the entire period $\lambda \in (0, 2.5]$.

As shown in Fig. 8.11, when the attacker becomes more advanced with a larger failure probability of attraction, i.e., $p := p(j|s_{12}, a_A), \forall j \in \{s_{12}, s_{13}\}$, he can stay in the normal zone with a larger probability. A significant increase happens after $p \geq 0.5$. On the other hand, as p increases from 0 to 1, the utility of the normal zone reduces linearly, and the expected utility over the stationary probability remains approximately unchanged until $p \geq 0.9$.

Fig. 8.10 and Fig. 8.11 demonstrate that the expected utility over the stationary

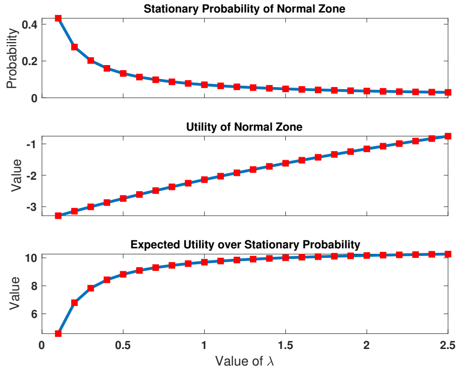


Figure 8.10: Three engagement criteria under different persistence levels $\lambda \in (0, 2.5]$.

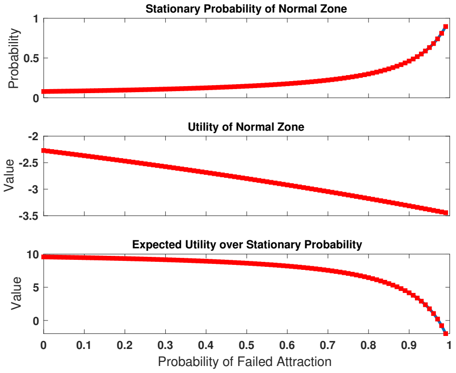


Figure 8.11: Three engagement criteria under different intelligence levels $p \in [0, 1]$.

probability receives a large decrease only at the extreme cases of a high transition frequency and a large penetration probability. Similarly, the stationary probability of the normal zone remains small for most cases except for the above extreme cases. Thus, our policy provides a robust expected utility as well as a low-risk engagement over a large range of changes in the attacker's persistence and intelligence.

8.3 Reinforcement Learning of SMDP

Due to the absent knowledge of an exact SMDP model, i.e., the investigation reward, the attacker's transition probability (and even the network topology), and the sojourn distribution, the defender has to learn the optimal engagement policy based on the actual experience of the honeynet interactions. As one of the classical model-free Reinforcement Learning (RL) methods, the Q-learning algorithm for

SMDP has been stated in [21], i.e.,

$$\begin{aligned}Q^{k+1}(s^k, a^k) :=&(1-\alpha^k(s^k,a^k))Q^k(s^k,a^k) + \alpha^k(s^k,a^k)[\bar{r}_1(s^k,a^k,\bar{s}^{k+1})\\ &+ \bar{r}_2(s^k,a^k)\frac{(1-e^{-\gamma\bar{\tau}^k})}{\gamma} - e^{-\gamma\bar{\tau}^k}\max_{a'\in\mathcal{A}(\bar{s}^{k+1})} Q^k(\bar{s}^{k+1},a')],\end{aligned} \tag{8.8}$$

where s^k is the current state sample, a^k is the current selected action, $\alpha^k(s^k, a^k) \in (0, 1)$ is the learning rate, $\bar{s}^{k+1}$ is the observed state at next stage, $\bar{r}_1, \bar{r}_2$ is the observed investigation rewards, and $\bar{\tau}^k$ is the observed sojourn time at state s^k. When the learning rate satisfies $\sum_{k=0}^{\infty} \alpha^k(s^k, a^k) = \infty, \sum_{k=0}^{\infty}(\alpha^k(s^k, a^k))^2 < \infty, \forall s^k \in \mathcal{S}, \forall a^k \in \mathcal{A}(s^k)$, and all state-action pairs are explored infinitely, $\max_{a'\in\mathcal{A}(s^k)} Q^k(s^k, a'), k \to \infty$, in (8.8) converges to value $v(s^k)$ with probability 1.

At each decision epoch $k \in \{0, 1, \cdots\}$, the action a^k is chosen according to the ϵ-greedy policy, i.e., the defender chooses the optimal action denoted as $arg \max_{a'\in\mathcal{A}(s^k)} Q^k(s^k, a')$ with a probability $1 - \epsilon$, and a random action with a probability ϵ. Note that the exploration rate $\epsilon \in (0, 1]$ should not be too small to guarantee sufficient samples of all state-action pairs. The Q-learning algorithm under a pure exploration policy $\epsilon = 1$ still converges yet at a slower rate.

In our scenario, the defender knows the reward of ejection action a_A and $v(s_{13}) = 0$, thus does not need to explore action a_A to learn it. We plot one learning trajectory of the state transition and sojourn time under the ϵ-greedy exploration policy in Fig. 8.12, where the chosen actions a_E, a_P, a_L, a_H are denoted in red, blue, purple, and green, respectively. If the ejection reward is unknown, the defender should be restrictive in exploring a_A which terminates the learning process. Otherwise, the defender may need to engage with a group of attackers who share similar behaviors to obtain sufficient samples to learn the optimal engagement

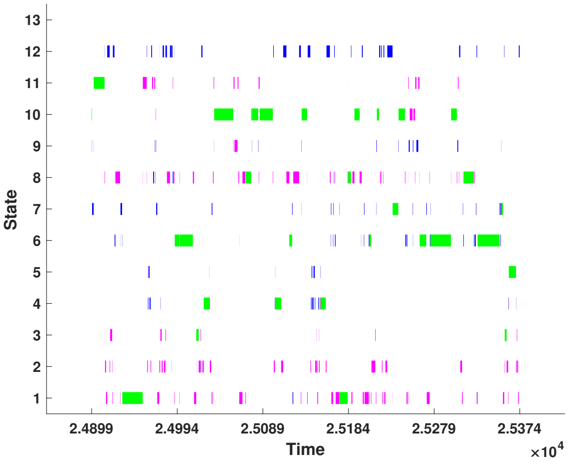


Figure 8.12: One instance of Q-learning on SMDP where the x-axis shows the sojourn time and the y-axis represents the state transition. The chosen actions a_E, a_P, a_L, a_H are denoted in red, blue, purple, and green, respectively.

policy.

In particular, we choose $\alpha^k(s^k, a^k) = \frac{k_c}{k_{\{s^k,a^k\}}-1+k_c}, \forall s^k \in \mathcal{S}, \forall a^k \in \mathcal{A}(s^k)$, to guarantee the asymptotic convergence, where $k_c \in (0, \infty)$ is a constant parameter and $k_{\{s^k,a^k\}} \in \{0, 1, \cdots\}$ is the number of visits to state-action pair $\{s^k, a^k\}$ up to stage k. We need to choose a proper value of k_c to guarantee a good numerical performance of convergence in finite steps as shown in Fig. 8.13. We shift the green and blue lines vertically to avoid the overlap with the red line and represent the corresponding theoretical values in dotted black lines. If k_c is too small as shown in the red line, the learning rate decreases so fast that new observed samples hardly update the Q-value and the defender may need a long time to learn the right value. However, if k_c is too large as shown in the green line, the learning rate decreases so slow that new samples contribute significantly to the current Q-value. It causes a large variation and a slower convergence rate of $\max_{a' \in \mathcal{A}(s_{12})} Q^k(s_{12}, a')$.

We show the convergence of the policy and value under $k_c = 1, \epsilon = 0.2$, in the

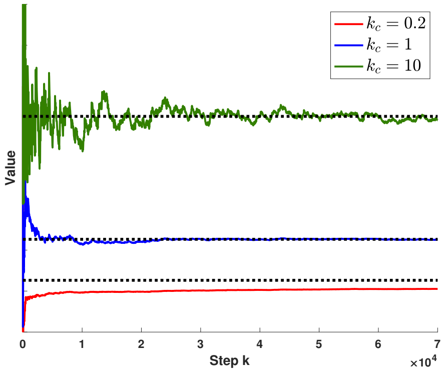


Figure 8.13: The convergence rate under different values of k_c.

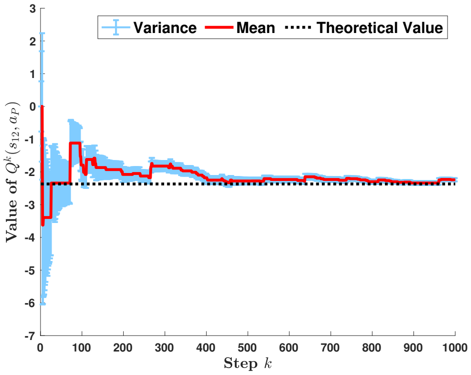


Figure 8.14: The evolution of the mean and the variance of $Q^k(s_{12}, a_P)$.

video demo (See URL: https://bit.ly/2QUz3Ok). In the video, the color of each node n^k distinguishes the defender's action a^k at state s^k and the size of the node is proportional to $\max_{a' \in \mathcal{A}(s^k)} Q^k(s^k, a')$ at stage k. To show the convergence, we decrease the value of ϵ gradually to 0 after 5000 steps.

Since the convergence trajectory is stochastic, we run the simulation for 100 times and plot the mean and the variance of $Q^k(s_{12}, a_P)$ of state s_{12} under the optimal policy $\pi(s_{12}) = a_P$ in Fig. 8.14. The mean in red converges to the theoretical value in about 400 steps and the variance in blue reduces dramatically as step k increases.

Part V

Incentive Mechanisms against Insider Threats

Chapter 9

ZETAR: Strategic and Trustworthy Recommendations for Compliance Improvement

Following Section 1.3.2, insider threats have resulted in significant operational disruptions, data loss, and reputation damage. Many studies [74, 146, 213] have recognized the critical role of incentives in mitigating insider threats. An incentive-based insider threat program can influence insiders' incentives and elicit proper behaviors of the insiders that align with the organization's security objectives. Its design can proactively improve the cyber hygiene of an organization by deterring noncompliance and misbehavior.

It is, however, challenging for a defender to model, characterize, and further control the insiders' incentives quantitatively. First, insiders' incentives are not directly controllable but indirectly and restrictively affected through extrinsic factors such as reward, penalty, and information. Systematically quantifying the impact of these extrinsic factors on the insiders' incentives and behaviors is the precondition of designing them to motivate (rather than command) an employee to act in the organization's interests. Second, it is costly to customize an incentive mechanism for a large population of insiders with varied incentives. The automated and customized design of the incentive mechanism is a desideratum to achieve population-level compliance. Third, insiders' incentives are not directly observable but have to be gradually learned based on insiders' behaviors and responses to the incentive mechanism. It is instrumental for the learning process to be fast and adaptive to mitigate insider threats timely.

To this end, we develop a modeling and computational framework called ZETAR[1] for the defender to improve compliance and organizational cyber hygiene. As illustrated in Fig. 9.1, ZETAR conducts two major roles of zero-trust audit and

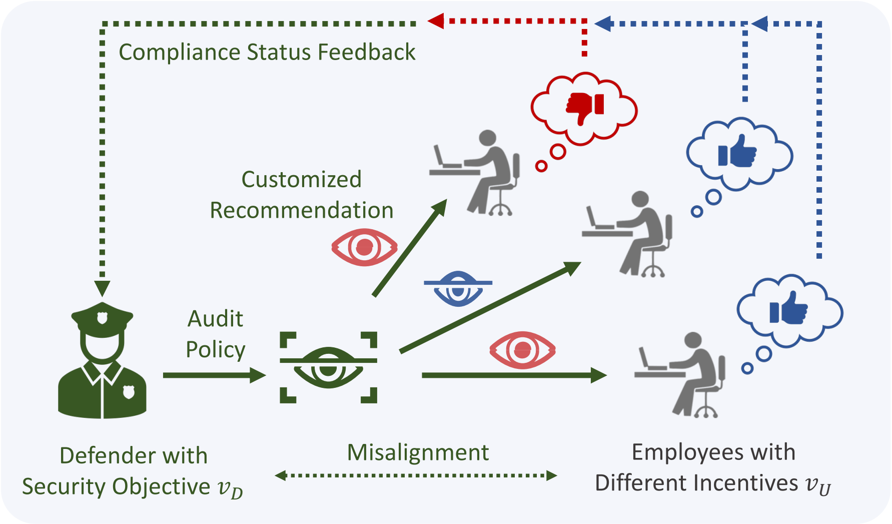


Figure 9.1: An illustration of ZETAR feedback system: the defender of a corporate network conducts a stochastic audit on employees' compliance status and provides customized recommendations to employees based on the learning of their incentives. ZETAR reduces the incentive misalignment between employees and the defender.

[1] ZETAR stands for ZEro-Trust Audit with strategic Recommendation.

strategic recommendation. The zero-trust audit mechanism inspects each insider's behaviors to attribute accountable insiders, penalize non-compliant behaviors, and reward compliant behaviors. It is challenging to customize the audit mechanism under time and budget constraints. ZETAR introduces a strategic recommendation mechanism to reveal information that adapts to the insiders' different incentives.

9.1 System Model of ZETAR

ZETAR provides customized designs for employees with different incentives. Each design involves two players, the organizational defender D and an employee U. The defender can assess the organization's security posture and audit insiders' behaviors either by himself or through a third-party service provider. An audit monitors the compliance of the insiders. The defender is informed of the audit outcomes and improves the compliance by appropriate management of incentives and strategic recommendations.

9.1.1 An Organization's Security Posture

Security Posture (SP) reflects an enterprise's overall cybersecurity strength and capacities to deter, detect, and respond to the dynamic threat landscape [42]. Based on different scoring and categorization methodologies [5], SP can be classified into finite categories (e.g., high-risk SP and low-risk SP). In this work, we consider a finite number of J SP categories that compose the set $\mathcal{Y} := \{y^j\}_{j \in \mathcal{J}}$, where $\mathcal{J} := \{1, \cdots, J\}$. The current SP can be assessed based on penetration tests, honeypots, and alert analysis [229]. Since an organization's SP changes probabilistically based on the dynamic behaviors of attackers, users, and defenders, we let $b_Y(y^j) \in [0,1]$ denote the probability of the organization to be in the state of SP $y^j \in \mathcal{Y}, \forall j \in \mathcal{J}$. With a slight abuse of notation, we define $b_Y \in \Delta\mathcal{Y}$ as the probability distribution over $\mathcal{Y}$.

9.1.2 Zero-Trust Audit Policy

The defender of an organization needs to follow prescribed security rules to improve its cyber hygiene. These rules can be set and audited by regulatory agencies, cyber insurance providers, or the organization itself. Aligning with the zero-trust security principle (e.g., see [181]), the audit is applied to all employees in an organization without a prior trust assignment. Consider a finite set of I Audit Schemes (ASs), denoted by $\mathcal{X}$. Each AS contains the entire audit procedure. For example, for a given AS $x \in \mathcal{X}$, the audit involves the steps of (1) monitoring and checking the employee's behaviors (2) assigning a compliance score to the employee, and (3) informing (the defender) of the compliance score and action. A different AS $x' \in \mathcal{X}, x' \neq x$, can vary in the monitoring or scoring scheme. The audit schemes are prescribed based on the security posture of the organization. Let $\psi \in \Psi : \mathcal{Y} \mapsto \Delta\mathcal{X}$ denote the audit policy, which probabilistically determines an AS $x \in \mathcal{X}$, where $|\mathcal{X}| = I$. The probability of choosing $x \in \mathcal{X}$ given the security posture $y \in \mathcal{Y}$ is thus given by $\psi(x|y) \in [0,1]$. The outcome of the audit scheme is used by the defender to create penalties or rewards for the employees to shape their incentives and elicit compliant behaviors. Hence, the incentives of the employees and the security objective of the defender are naturally dependent on the prescribed audit scheme. They will be further elaborated on in Section 9.1.3.

Example 3 (**Stochastic Audit of Critical Security Rules**). *Consider an organization that needs to comply with a finite set of H critical security rules, denoted by*

$\mathcal{H} := \{1, \cdots, H\}$, set by a U.S. regulatory agency. The rules entail proper behaviors for remote access, user accounts, and backups [187]. The compliance of an employee is monitored by checking each rule. Its outcome, denoted by o^h, also known as the compliance status concerning rule $h \in \mathcal{H}$, is either full, partial, or no compliance, denoted by ι_f, ι_p, and ι_n, respectively. By lumping the outcomes into a vector, we let $a = (o^1, \cdots, o^H) \in \mathcal{A} := \prod_{h \in \mathcal{H}} \mathcal{O}^h$, where $\mathcal{O}^h = \{\iota_f, \iota_p, \iota_n\}$, be the consolidated compliance status of an employee. An employee can choose his consolidated compliance status $a \in \mathcal{A}$ based on his incentives. Let $\mathcal{X} = \{x^1, \cdots, x^{H+1}\}$ be the set of $I = H + 1$ ASs. Each AS follows the same procedure of checking the compliance of the H rules to report an employee's compliance status but differs in assessing compliance scores. AS $x^h \in \mathcal{X}, h = 1, \cdots, H$, yields a compliance score $r^h \in \mathbb{R}$ solely based on the outcome $o^h \in \mathcal{O}^h$, i.e., $r^h = g^h(o^h)$, where $g^h : \mathcal{O}^h \to \mathbb{R}$ is the scoring function associated with AS x^h. AS $x^{H+1} \in \mathcal{X}$ uses the outcomes associated with all the rules for the assessment, i.e., $r^{H+1} = g^{H+1}(a)$, where $g^{H+1} : \mathcal{A} \to \mathbb{R}$ is the scoring function associated with AS x^{H+1}. It is clear that x^{H+1} is the most stringent AS among all. The score is used as the criterion to penalize employees and thus affects their incentives that will be formally defined in Section 9.1.3.

In Example 3, the audit policy $\psi \in \Psi$ is chosen based on a predetermined level of tolerance. A proper level of tolerance trades off between the organization's security and the compliance cost resulting from the overhead and the lack of flexibility [146]. An appropriate choice of tolerance depends on the SP; e.g., an audit policy can prescribe the stringent audit $x^{H+1} \in \mathcal{X}$ at a higher rate under high-risk SP than low-risk SP. We assume that the audit policy ψ set by the organization or regulatory agencies remains the same for a sufficiently long time, making the policy more implementable and agreeable to employees over the entire corporate network

[213].

Let $\mathcal{A}$ denote the set (with cardinality K) of an insider's actions. In Example 3, an action $a \in \mathcal{A}$ is referred to as the rule compliance profile, i.e., $a = (o^1, \cdots, o^H)$, which is a result of the insiders' behaviors, including keystrokes, full application contents (e.g., email, chat, data import, and data export), and screen captures [201]. In the case where there is one rule, *i.e.*, $H = 1$, $\mathcal{A}$ is reduced to an action set that comprises three actions: full, partial, and no compliance. The insider's actions are monitored by the AS, and the defender is informed of the insider's compliance to nudge compliant behaviors.

9.1.3 Utilities of the Defender and Employees

In the past five years, financially motivated insider threats have continued to be the most common motive of threat actors [15]. We define utility functions $v_p : \mathcal{Y} \times \mathcal{X} \times \mathcal{A} \mapsto \mathbb{R}$ for $p \in \{U, D\}$ to capture an employee's incentive and the defender's security objective, respectively. The defender's utility $v_D(y, x, a)$ assesses the impact of an employee's action $a \in \mathcal{A}$ on network security under the SP $y \in \mathcal{Y}$ and AS $x \in \mathcal{X}$. Since the impact is assessed subjectively by the defender, v_D represents the defender's security objective. For example, under life-critical scenarios with zero tolerance to non-compliance, the defender can assign $v_D(y, x, a^{ic}) = -\infty, \forall y \in \mathcal{Y}, x \in \mathcal{X}$.

An employee's utility $v_U(y, x, a)$ models his extrinsic and intrinsic incentives to take action $a \in \mathcal{A}$ under SP $y \in \mathcal{Y}$ and AS $x \in \mathcal{X}$. On the one hand, v_U can incorporate monetary incentives (through reward and recognition) and disincentives (through penalty and punishment) from the defender. On the other hand, v_U can represent an employee's proclivity for compliant behaviors. Readers can refer to

Section 9.5.1 and 9.5.1 for an example of v_D and v_U, respectively.

9.1.4 Strategic Recommendations for Customized Compliance

Following Section 9.1.2, the audit policy ψ remains unchanged once determined. Since it is challenging for a fixed audit policy to achieve optimal inspection outcomes for employees with different compliance requirements and incentives (as will be further elaborated in Section 9.1.4), the defender designs a customized recommendation mechanism, including a recommendation policy $\pi \in \Pi$ that results in a recommendation $s \in \mathcal{S}$.

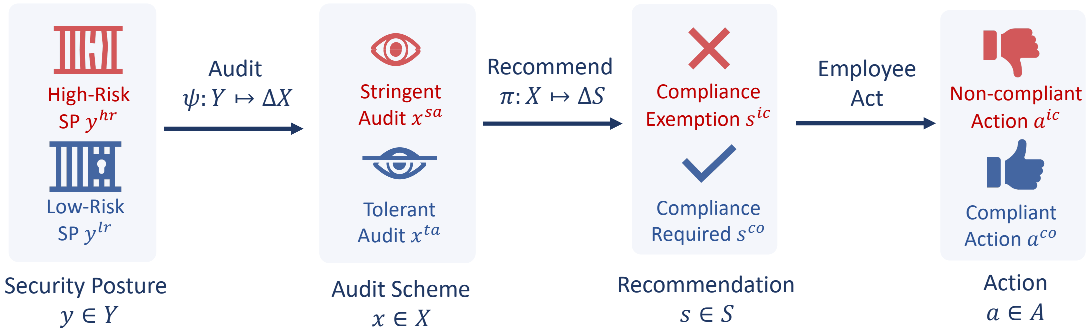

High-Risk SP y^{hr}
Low-Risk SP y^{lr}
Security Posture $y \in Y$
Audit $\psi: Y \mapsto \Delta X$
Stringent Audit x^{sa}
Tolerant Audit x^{ta}
Audit Scheme $x \in X$
Recommend $\pi: X \mapsto \Delta S$
Compliance Exemption s^{ic}
Compliance Required s^{co}
Recommendation $s \in S$
Employee Act
Non-compliant Action a^{ic}
Compliant Action a^{co}
Action $a \in A$

Figure 9.2: The timeline of ZETAR to enhance employees' compliance in corporate networks. The defender is informed of the security posture and the audit outcomes of all employees' behaviors. The defender designs a recommendation policy $\pi \in \Pi$ to improve compliance.

Information Structure and Timeline

As stated in [146], transparent criteria for organizational policies can create a culture of trust and consequently serve as a positive incentive to reduce non-compliance. Thus, we assume that the sets $\mathcal{Y}, \mathcal{X}, \mathcal{S}, \mathcal{A}$, the prior statistics $b_Y \in \Delta\mathcal{Y}$, the audit policy $\psi \in \Psi$, and the recommendation policy $\pi \in \Pi$ are common knowledge. Since the organization's SP changes randomly as shown in Section 9.1.1, the SP is assessed repeatedly on a weekly or monthly basis, and it is unknown to employees.

Fig. 9.2 illustrates the timeline of ZETAR as follows. Given the current SP $y \in \mathcal{Y}$ and the audit policy $\psi \in \Psi$, the chosen AS $x \in \mathcal{X}$ is known to the defender yet remains unknown to the employees. Before implementing the chosen AS x, the defender recommends an action based on x and the recommendation policy $\pi \in \Pi$. Then, the employee takes an action $a \in \mathcal{A}$ that is not necessarily the recommended one. Finally, the defender implements the chosen AS and penalizes employees based on the audit outcome. Employees observe the chosen AS after it is implemented. After the zero-trust audit, the employees' actions become known to the defender.

Employee's Initial Compliance

Without the recommendation mechanism, an employee takes an action $a_0 \in \mathcal{A}$ to maximize his expected utility concerning the prior statistics $b_Y \in \Delta\mathcal{Y}$ and $\psi \in \Psi$, i.e., $a_0 \in \arg\max_{a\in\mathcal{A}} \sum_{y\in\mathcal{Y}} b_Y(y) \sum_{x\in\mathcal{X}} \psi(x|y) v_U(y,x,a)$. Due to the misalignment between an employee's incentive v_U and the defender's security objective v_D, the employee's initial compliance status represented by $a_0 \in \mathcal{A}$ may negatively affect corporate security. For example, a self-interested employee tends to break the security rules for convenience if the audit policy ψ chooses a stringent audit (e.g., $x^{H+1} \in \mathcal{X}$ in Example 3) less frequently.

Recommendation Mechanism

To align employees' incentives with the defender's security objective, the defender can recommend an action to an employee. Thus, set $\mathcal{S} := \{s^k\}_{k \in \mathcal{K}}$ has the same cardinality with $\mathcal{A}$ and represents the finite set of K recommendations where $s^k \in \mathcal{S}$ recommends the employee to take action $a^k \in \mathcal{A}$. The defender recommends the action according to a stochastic recommendation policy $\pi \in \Pi : \mathcal{X} \mapsto \Delta\mathcal{S}$; i.e., given the chosen AS $x \in \mathcal{X}$, the defender chooses recommendation $s \in \mathcal{S}$ with probability $\pi(s|x)$. As will be shown in Section 9.1.4, by strategically choosing the recommendation policy, the defender can manipulate an employee's belief of the current SP and the chosen AS, thus affecting his perception of the expected utility and enhancing compliance.

Employee's Belief Update and Best-Response Action

The received recommendation reveals the defender's knowledge of the SP and the chosen AS. An employee can form and update a belief of the unknowns by observing the recommendations. Denote $b_{Y,X} \in \mathcal{B}_{Y,X} \subseteq \Delta(\mathcal{X} \times \mathcal{Y})$ as the joint prior distribution of the current SP and the chosen AS, i.e., $b_{Y,X}(y, x) := b_Y(y)\psi(x|y), \forall x \in \mathcal{X}, y \in \mathcal{Y}$. Analogously, we define $b_X(x) := \sum_{y' \in \mathcal{Y}} b_{Y,X}(y', x)$ as the marginal prior probability of AS $x \in \mathcal{X}$, $b_{Y|X}(y|x) := b_{Y,X}(y, x)/b_X(x)$ as the conditional prior probability of SP $y \in \mathcal{Y}$ under AS $x \in \mathcal{X}$, and $b^{\pi}_S(s) := \sum_{x' \in \mathcal{X}} b_X(x')\pi(s|x')$ as the probability of recommendation $s \in \mathcal{S}$ under $\pi \in \Pi$, where $b_X \in \mathcal{B}_X \subseteq \Delta\mathcal{X}$, $b_{Y|X} \in \mathcal{B}_{Y|X}$, and $b^{\pi}_S \in \Delta\mathcal{S}$.

Following the requirement of transparent criteria in Section 9.1.4, the recommendation policy $\pi \in \Pi$ is assumed to be common knowledge. The assumption can be justified by the fact that an employee can learn the recommendation policy

$\pi \in \Pi$ based on the repeated observations of the recommendation policy input (i.e., AS $x \in \mathcal{X}$) and the policy output (i.e., recommendation $s \in \mathcal{S}$) after they are implemented. Thus, for rational employees who adopt Bayesian rules to update their beliefs, each recommendation $s \in \mathcal{S}$ under recommendation policy $\pi \in \Pi$ results in posterior belief $b^{\pi}_{Y,X}(y,x|s) \in \mathcal{B}^{\pi}_{Y,X} \subseteq \Delta(\mathcal{X} \times \mathcal{Y})$, i.e.,

$$b^{\pi}_{Y,X}(y,x|s) = \frac{b_{Y,X}(y,x)\pi(s|x)}{\sum_{x' \in \mathcal{X}} b_X(x')\pi(s|x')}, \forall x \in \mathcal{X}, y \in \mathcal{Y}. \tag{9.1}$$

Then, we can obtain the employee's marginal posterior belief of AS $x \in \mathcal{X}$, his marginal posterior belief of SP $y \in \mathcal{Y}$, and the associated conditional posterior belief under recommendation $s \in \mathcal{S}$ as $b^{\pi}_X(x|s) := \sum_{y \in \mathcal{Y}} b^{\pi}_{Y,X}(y,x|s) = \frac{b_X(x)\pi(s|x)}{b^{\pi}_S(s)} \in \mathcal{B}^{\pi}_X \subseteq \Delta\mathcal{X}$, $b^{\pi}_Y(y|s) := \sum_{x \in \mathcal{X}} b^{\pi}_{Y,X}(y,x|s) \in \Delta\mathcal{Y}^{\pi} \subseteq \Delta\mathcal{Y}$, and $b^{\pi}_{Y|X}(y|x,s) := b^{\pi}_{Y,X}(y,x|s)/b^{\pi}_X(x|s)$, respectively. Since $b^{\pi}_{Y|X}(y|x,s) = b_{Y|X}(y|x), \forall s \in \mathcal{S}$, these recommendations under policy $\pi \in \Pi$ have no impact on the conditional probability $b^{\pi}_{Y|X}$. However, as it does not hold in general that $b^{\pi}_{Y,X} = b_{Y,X}, b^{\pi}_X = b_X, b^{\pi}_Y = b_Y, \forall s \in \mathcal{S}$, the recommendation mechanism (i.e., $\pi \in \Pi$ and $s \in \mathcal{S}$) can change the employees' marginal beliefs of the current SP and the implemented AS as well as their joint beliefs. We summarize the above observations in Lemma 3.

Lemma 3 (**Invariance of Conditional Belief**). *A recommendation policy $\pi \in \Pi$ has no impact on $b^{\pi}_{Y|X}$.*

With a recommendation policy $\pi \in \Pi$, the employee takes a best-response action denoted by $a^*_{\pi,s} \in \mathcal{A}$ to maximize his posterior utility under recommendation $s \in \mathcal{S}$, i.e.,

$$a^*_{\pi,s} \in \arg\max_{a \in \mathcal{A}} \mathbb{E}_{y,x \sim b^{\pi}_{Y,X}(\cdot|,s)}[v_U(y,x,a)]. \tag{9.2}$$

Letting $\bar{v}_p(x,a) := \sum_{y \in \mathcal{Y}} b_{Y|X}(y|x)v_p(y,x,a), \forall x \in \mathcal{X}$ for $p \in \{U, D\}$ be an employee's expected incentive and the defender's expected security objective, respectively, we obtain

$$\mathbb{E}_{y,x\sim b^{\pi}_{Y,X}(\cdot|,s)}[v_p(y,x,a)] = \sum_{x\in\mathcal{X}} b^{\pi}_{X}(x|s)\bar{v}_p(x,a), \forall s \in \mathcal{S}. \tag{9.3}$$

We refer to $a^*_{\pi,s} \in \mathcal{A}$ as an action induced by recommendation policy $\pi \in \Pi$ under recommendation $s \in \mathcal{S}$, which is in general different from the employee's initial compliance status $a_0 \in \mathcal{A}$ in Section 9.1.4. For a given π, not all recommendations induce a compliant action. However, by strategically choosing the recommendation policy, the defender can improve compliance on average. We formally quantify the improvement of compliance and its average impact on the corporate security in Section 9.1.4.

Trustworthiness of the Recommendation Scheme

Following Section 9.1.4, the defender's recommendation $s \in \mathcal{S}$ from a recommendation policy $\pi \in \Pi$ may not be trusted by an employee; i.e., the recommended action is not a best-response action. We formalize the definitions of trustworthy recommendations and trustworthy recommendation policies in Definitions 22 and 23, respectively.

Definition 22 (**Trustworthy Recommendations**). *A recommendation $s^k \in \mathcal{S}, k \in \mathcal{K}$, under a recommendation policy $\pi \in \Pi$ is trustworthy (resp. untrustworthy); i.e., the policy π is trusted by an employee with an incentive $\bar{v}_U$, if the induced action follows (resp. does not follow) the recommended action $a^k \in \mathcal{A}$, i.e., $a^k \in$ (resp. $\notin$) arg$\max_{a\in\mathcal{A}} \mathbb{E}_{x\sim b^{\pi}_{X}(\cdot|,s)}[\bar{v}_U(x,a)]$.*

Remark 13 (**Compliance and Trustworthiness**). *Following Definition 22, an employee complies with a recommendation (i.e., takes the recommended action)* *only if it is trustworthy.*

Definition 23 (**Trustworthy Recommendation Policies**). *Recommendation policies under which recommendation $s^k \in \mathcal{S}, k \in \mathcal{K}$, is trustworthy (resp. untrustworthy) formulate the k-th Partially Trustworthy (PT) (resp. Partially Untrustworthy (PU)) policy set $\Pi_{pt}^k \subseteq \Pi$ (resp. $\Pi_{pu}^k \subseteq \Pi$). A recommendation policy $\pi \in \Pi$ is Completely Trustworthy (CT) (resp. Completely Untrustworthy (CU)) if all recommendations under π are trustworthy (resp. untrustworthy). All CT (resp. CU) recommendation policies formulate the CT (resp. CU) policy set $\Pi_{ct} := \cap_{k=1}^K \Pi_{pt}^k$ (resp. $\Pi_{cu} := \cap_{k=1}^K \Pi_{pu}^k$).*

Different recommendation policies reveal varied amounts information about the AS and the SP, which consequently affect the employee's compliance status. Two extreme cases are defined in Definition 24. Let the optimal action of an employee U or the defender D at AS $x \in \mathcal{X}$ and SP $y \in \mathcal{Y}$ be given by $\tilde{a}_p^{max}(y,x) \in \arg\max_{a\in\mathcal{A}} v_p(y,x,a)$. Analogously, let $a_p^{max}(x) \in \arg\max_{a\in\mathcal{A}} \bar{v}_p(x,a)$ for all $x \in \mathcal{X}$ and $p \in \{U, D\}$. A zero-information recommendation policy, denoted by $\pi_z \in \Pi$, recommends the same actions as an employee's initial compliance status in Section 9.1.4 regardless of the chosen AS. Hence π_z does not change the employee's belief, i.e., $b_X^{\pi_z}(x|s) = b_X(x), \forall s \in \mathcal{S}, \forall x \in \mathcal{X}$, and does not bring new information to the employee. Meanwhile, a full-information recommendation policy denoted by $\pi_f \in \Pi$ recommends optimal action $a_U^{max}(x)$ under the chosen AS $x \in \mathcal{X}$. Remark 14 shows that it is feasible for the defender to implement CT recommendation policies regardless of ZETAR settings in Sections 9.1.1 to 9.1.3.

Definition 24 (**Zero- and Full-Information Recommendation Policy**). *A recommendation policy $\pi_z \in \Pi$ contains zero information if $\pi_z(s^k|x) = \mathbf{1}_{\{a^k=a_0\}}, \forall k \in$*

$\mathcal{K}, \forall x \in \mathcal{X}$. A recommendation policy $\pi_f \in \Pi$ contains full information if $\pi_f(s^k|x) = \mathbf{1}_{\{a^k = a_U^{max}(x)\}}, \forall k \in \mathcal{K}, \forall x \in \mathcal{X}$.

Remark 14 (**Feasibility**)**.** *Following Definition 24, zero- and full-information recommendation policies are CT, i.e., $\pi_z, \pi_f \in \Pi_{ct}$. Thus, Π_{ct} is nonempty regardless of ZETAR settings.*

Defender's Optimal Recommendation Policy

Following (9.2) and (9.3), an employee's expected utility defined in (9.4) represents the employee's Acquired Satisfaction Level (ASaL) under recommendation policy $\pi \in \Pi$.

$$J_U(\pi, b_X, \bar{v}_U) := \sum_{s \in \mathcal{S}} b_S^{\pi}(s) \mathbb{E}_{y,x \sim b_{Y,X}^{\pi}(\cdot|,s)}[v_U(y, x, a_{\pi,s}^*)]. \tag{9.4}$$

Since an employee's action induced by zero-information policy π_z is his initial-compliance action $a_0 \in \mathcal{A}$, $J_U(\pi, b_X, \bar{v}_U)$ represents the employee's Innate Satisfaction Level (ISaL). To capture the average impact of an employee's compliance status on corporate security under different recommendations, we define the defender's Acquired Security Level (ASeL) under recommendation policy $\pi \in \Pi$ as

$$\begin{aligned} \tilde{J}_D(\pi, b_{Y,X}, v_D, v_U) &:= \mathbb{E}_{y,x \sim b_{Y,X}(\cdot)} \mathbb{E}_{s \sim \pi(\cdot|x)}[v_D(y, x, a_{\pi,s}^*)] \\ &= \sum_{x \in \mathcal{X}} b_X(x) \sum_{s \in \mathcal{S}} \pi(s|x) \bar{v}_D(x, a_{\pi,s}^*) := J_D(\pi, b_X, \bar{v}_D, \bar{v}_U). \end{aligned} \tag{9.5}$$

Since an employee's best-response action $a_{\pi_z,s}^* \in \mathcal{A}$ remains the same as $a_0 \in \mathcal{A}$ in Section 9.1.4 under all recommendations $s \in \mathcal{S}$, a zero-information recommendation policy $\pi_z \in \Pi_{ct}$ has no impact on the employee's compliance. Hence $J_D(\pi_z, b_X, \bar{v}_D, \bar{v}_U)$ quantifies the impact of an employee's initial compliance status and represents the defender's Initial Security Level (ISeL). The difference in the defender's security level $J_D^{acel}(\pi, b_X, \bar{v}_D, \bar{v}_U) := J_D(\pi, b_X, \bar{v}_D, \bar{v}_U) - J_D(\pi_z, b_X, \bar{v}_D, \bar{v}_U)$ measures the average impact of the employee's compliance status changes (under recommendation policy $\pi \in \Pi$) on the corporate security, and we refer to J_D^{acel} as the Average Compliance Enhancement Level (ACEL) in Definition 25.

Definition 25 (**Average Compliance Enhancement Level**). *For an employee with incentive $\bar{v}_U$ and the defender with security objective $\bar{v}_D$, we define $J_D^{acel}(\pi, b_X, \bar{v}_D, \bar{v}_U) \in \mathbb{R}$ as the Average Compliance Enhancement Level (ACEL) under the prior statistic $b_X \in \mathcal{B}_X$ defined in Section 9.1.4 and recommendation policy $\pi \in \Pi$ defined in Section 9.1.4.*

The defender's goal is to design the optimal recommendation policy $\pi^* \in \Pi$ that maximizes ACEL, where $J_D^{acel,*}$ denotes the optimal ACEL, i.e., $J_D^{acel,*}(b_X, \bar{v}_D, \bar{v}_U) := J_D^{acel}(\pi^*, b_X, \bar{v}_D, \bar{v}_U) = \max_{\pi \in \Pi} J_D^{acel}(\pi, b_X, \bar{v}_D, \bar{v}_U) \geq 0$. The optimal ACEL measures the maximum improvement of an employee's compliance and thus shows how persuadable the employee is under the recommendation scheme.

Remark 15 (**Scoring Metrics**). *The ISeL $J_D(\pi_z, b_X, \bar{v}_D, \bar{v}_U)$ and the optimal ACEL $J_D^{acel,*}(b_X, \bar{v}_D, \bar{v}_U)$ provide scores to quantify how compliant and persuadable, respectively, an employee with incentive $\bar{v}_U$ is under security objective $\bar{v}_D$.*

9.2 Computational Framework of ZETAR

In this section, we formulate the design of ZETAR into mathematical programming problems, where the defender has complete information of an employee's incentive $\bar{v}_U$.

9.2.1 Level of Recommendation Customization

As illustrated in Section 9.1.2 and 9.1.4 and also in Fig. 7.1, the defender determines a unified audit policy to inspect all employees' behaviors yet designs customized recommendation policies. Since the difference in these recommendation policies can lead to the perceptions of unfairness and distrust [213], the defender needs to strike a balance between the optimal ACEL and the Level of Recommendation Customization (LoRC). We let $\eta \in \mathbb{R}^+$ be the defender's LoRC, $\pi_d \in \Pi$ be a default recommendation policy, and the KL divergence $KL(\pi||\pi_d) := \sum_{k\in\mathcal{K},x\in\mathcal{X}} \pi(s^k|x) \log \frac{\pi(s^k|x)}{\pi_d(s^k|x)}$ be the measure of policy difference, respectively. If $\pi_d(s^k|x) = 0$, then $\pi(s^k|x) = 0$ by default, and $\pi(s^k|x) \log \frac{\pi(s^k|x)}{\pi_d(s^k|x)} = 0$ as $\lim_{z\to 0^+} z \log z = 0$.

9.2.2 Primal Mathematical Programming

Without loss of generality, the defender can narrow the policy search space to $\Pi_{ct} \subseteq \Pi$ to achieve the optimal ACEL [109], i.e.,

$$J_D^{acel,*}(b_X, \bar{v}_D, \bar{v}_U) = \max_{\pi\in\Pi_{ct}} J_D^{acel}(\pi, b_X, \bar{v}_D, \bar{v}_U).$$

Under a CT recommendation policy, the employee complies to the recommendation and chooses $a^k \in \mathcal{A}$ when the recommendation is $s^k \in \mathcal{S}, \forall k \in \mathcal{K}$. Thus, $\max_{\pi\in\Pi_{ct}} J_D^{acel}(\pi, b_X, \bar{v}_D, \bar{v}_U) = \max_{\pi\in\Pi_{ct}} \sum_{x\in\mathcal{X}} b_X(x) \sum_{k\in\mathcal{K}} \pi(s^k|x)\bar{v}_D(x, a^k)$. For a

LoRC η, we formulate the following convex program denoted by P_η.

$$[P_\eta]: r_\eta = \max_{\pi\in\Pi} \quad \sum_{x\in\mathcal{X}} b_X(x) \sum_{k\in\mathcal{K}} \pi(s^k|x)\bar{v}_D(x,a^k) - \frac{KL(\pi||\pi_d)}{\eta}$$

$$(a).\ \pi(s^k|x) \geq 0, \forall k\in\mathcal{K}, \forall x\in\mathcal{X},$$

$$(b). \sum_{k\in\mathcal{K}} \pi(s^k|x) = 1, \forall x\in\mathcal{X},$$

$$(c). \sum_{x\in\mathcal{X}} b_X(x)\pi(s^k|x)[\bar{v}_U(x,a^k) - \bar{v}_U(x,a^l)] \geq 0, \forall k,l\in\mathcal{K}.$$

Let $\pi^*_\eta \in \Pi_{ct}$ and r_η be the maximizer and the optimal value of P_η, respectively, for all $\eta\in\mathbb{R}^+$. Constraints (a), (b) explicitly describe the set Π, and constraint (c) limits the recommendation policy to be CT defined in Definition 23. All recommendation policies that satisfy constraints (a), (b), (c) compose the set $\Pi_{ct} \subseteq \Pi$. Due to the feasibility of CT policies in Remark 14 and the boundedness of v_D, the program P_η is feasible and bounded for all $\eta\in\mathbb{R}^+$. When the defender aims to design CT recommendation policies closest to the default policy $\pi_d\in\Pi$ (i.e., $\eta\to 0^+$), then $\pi^*_0 = \pi_d$ if and only if $\pi_d\in\Pi_{ct}$. As the LoRC η increases, the defender focuses more on compliance enhancement, and the optimizer of P_∞ coincides with π^* that achieves the optimal ACEL $J_D^{acel,*}$, i.e., $\pi^*_\infty = \pi^*$. By specifying $a^l\in\mathcal{A}$ in constraint (c) of P_η as the initial-compliance action $a_0\in\mathcal{A}$, we prove that CT policies never decrease an employee's satisfaction level in Proposition 5.

Proposition 5 (**Trustworthiness Promotes Satisfaction**). *An employee's ASaL $J_U(\pi, b_X, \bar{v}_U)$ is not lower than his ISaL $J_U(\pi_z, b_X, \bar{v}_U)$ for all $\pi\in\Pi_{ct}$ and $b_X\in\mathcal{B}_X$.*

Proof. An employee's ASaL in (9.4) under a CT recommendation policy $\pi\in\Pi_{ct}$ can be represented as $J_U(\pi, b_X, \bar{v}_U) = \sum_{s\in\mathcal{S}} b^\pi_S(s)\cdot \max_{a\in\mathcal{A}} \sum_{x\in\mathcal{X}} b^\pi_X(x|s)\bar{v}_U(x,a) = \sum_{x\in\mathcal{X}} b_X(x) \sum_{k\in\mathcal{K}} \pi(s^k|x)\bar{v}_U(x,a^k)$. Based on constraint (c) of P_η, we arrive at the result $\sum_{x\in\mathcal{X}} b_X(x)\pi(s^k|x)[\bar{v}_U(x,a^k)-\bar{v}_U(x,a_0)] \geq 0$ for all $k \in \mathcal{K}$ and $\pi \in \Pi_{ct}$. Hence, $J_U(\pi, b_X, \bar{v}_U) \geq \sum_{x\in\mathcal{X}} b_X(x)\bar{v}_U(x,a_0) = \max_{a\in\mathcal{A}} \sum_{x\in\mathcal{X}} b(x)\bar{v}_U(x,a) = J_U(\pi_z, b_X, \bar{v}_U)$. □

Remark 16 (Win-Win Situation). *Proposition 5 shows that an employee's ASaL is not lower than his ISaL if a recommendation policy is CT. Based on Remark 14, the defender's ASeL is not lower than her ISEL under the optimal recommendation policy, i.e.,* $J_D^{acel,*}(b_X, \bar{v}_D, \bar{v}_U) \geq 0$. *Thus, the optimal policy achieves a win-win situation between the defender and employees by promoting cyber hygiene and employees' satisfaction.*

9.2.3 Dual Mathematical Programming

Let $\alpha_\eta(s^k,x) \geq 0$, $\beta_\eta(x) \in \mathbb{R}$, and $\lambda_\eta(s^k,a^l) \geq 0$ denote the dual variables of the constraints (a), (b), and (c) in P_η, respectively. Define shorthand notation $\bar{\beta}_\eta(s^k,x,\lambda_\eta) := \bar{v}_D(x,a^k) + \sum_{a^l\in\mathcal{A}} \lambda_\eta(s^k,a^l)[\bar{v}_U(x,a^k)-\bar{v}_U(x,a^l)]$, where $\lambda_\eta(s^l,a^l), \forall l \in \mathcal{K}$, can take any finite values. The dual problem is denoted as D_η. The strong duality proved in Proposition 6 yields the bounds for the optimal value r_η of the primal problem P_η in Proposition 7. Define $\alpha^*_\eta(s^k,x)$, $\beta^*_\eta(x)$, and $\lambda^*_\eta(s^k,a^l)$ as the optimal dual variables and the shorthand notation $\underline{r} := \sum_{x\in\mathcal{X}}[\max_{k\in\mathcal{K}} b_X(x)\bar{\beta}_\eta(s^k,x,\lambda_\eta) + \log(\pi_d(s^k|x))/\eta]$.

$$[D_\eta]: \quad \min_{[\beta_\eta(x)\in\mathbb{R}]_{x\in\mathcal{X}},[\lambda_\eta(s^k,a^l)\in\mathbb{R}^{0+}]_{k,l\in\mathcal{K}}} \quad \sum_{x\in\mathcal{X}}[\beta_\eta(x)+\frac{1}{\eta}]$$

$$(a). \quad \sum_{x\in\mathcal{X}}[\bar{v}_U(x,a^k)-\bar{v}_U(x,a^l)]b_X(x)\pi_d(s^k|x) \cdot e^{\eta[b_X(x)\bar{\beta}_\eta(s^k,x,\lambda_\eta)-\beta_\eta(x)]} \geq 0, \forall k,l \in \mathcal{K},$$

$$(b). \quad \sum_{k\in\mathcal{K}} \pi_d(s^k|x)e^{\eta[b_X(x)\bar{\beta}_\eta(s^k,x,\lambda_\eta)-\beta_\eta(x)]-1} = 1, \forall x \in \mathcal{X}.$$

Proposition 6 (**Strong Duality**)**.** *For all $\eta \in \mathbb{R}^+$, D_η is the dual problem of P_η, and the optimal value of D_η is r_η.*

Proof. Since all constraints in P_η are linear, Slater's condition reduces to the feasibility of D_η [20], and strong duality holds. Thus, P_η and D_η achieve the same optimal value. Setting the gradient of the Lagrangian function of P_η concerning π to 0 yields $\frac{1}{\eta}(\log\frac{\pi(s^k|x)}{\pi_d(s^k|x)} + 1) = b_X(x)\bar{v}_D(x, a^k) + \alpha_\eta(s^k, x) - \beta_\eta(x) + \sum_{l\in\mathcal{K}} \lambda_\eta(s^k, a^l) b_X(x)[\bar{v}_U(x, a^k) - \bar{v}_U(x, a^l)]$, for all $k \in \mathcal{K}, x \in \mathcal{X}$, which leads to

$$\pi^*_\eta(s^k|x) = \pi_d(s^k|x) \cdot e^{\eta[b_X(x)\bar{\beta}_\eta(s^k,x,\lambda_\eta)+\alpha_\eta(s^k,x)-\beta_\eta(x)]-1}. \tag{9.6}$$

Since $\pi^*_\eta(s^k|x)$ in (9.6) is non-negative for all $k \in \mathcal{K}, x \in \mathcal{X}$, constraint (a) of P_η holds. Moreover, the complementary slackness implies the optimal dual variables $\alpha^*_\eta(s^k, x) = 0, \forall k \in \mathcal{K}, x \in \mathcal{X}$. Plugging $\pi^*_\eta(s^k|x)$ in (9.6) into constraints (b) and (c) of P_η leads to constraints (a) and (b) of D_η, respectively. Then, by strong duality, D_η minimizes the Lagrangian function $L(\pi^*_\eta, \alpha^*_\eta, \beta_\eta, \lambda_\eta) = \sum_{x\in\mathcal{X}}[\beta_\eta(x) + 1/\eta]$ over dual variables $\beta_\eta(x) \in \mathbb{R}, \lambda_\eta(s^k, a^l) \in \mathbb{R}^{0+}, \forall k, l \in \mathcal{K}, x \in \mathcal{X}$. □

Proposition 7 (**Bounds of the Optimal Value**)**.** *The lower and upper bounds of r_η are $\underline{r}$ and $\underline{r} + \log(K)/\eta$, respectively.*

Proof. Constraint (b) in D_η is equivalent to the log-sum-exp expression: $\beta_\eta(x) = \frac{1}{\eta}\log(\sum_{k\in\mathcal{K}} \pi_d(s^k|x)e^{\eta b_X(x)\bar{\beta}_\eta(s^k,x,\lambda_\eta)-1})$. Therefore, $\max_{k\in\mathcal{K}} \eta b_X(x)\bar{\beta}_\eta(s^k, x, \lambda_\eta) - 1 + \log(\pi_d(s^k|x)) \leq \eta\beta_\eta(x) \leq \max_{k\in\mathcal{K}} \eta b_X(x)\bar{\beta}_\eta(s^k, x, \lambda_\eta) - 1 + \log(\pi_d(s^k|x)) + \log(K)$ for all $x \in \mathcal{X}$. Since strong duality holds, we obtain the bounds for r_η in P_η. □

Following (9.6) and Proposition 7, the optimal policy π^*_η has the closed-form expression in (9.7) concerning the optimal dual variables $\lambda^*_\eta(s^k, a^l) \in \mathbb{R}^{0+}, l, k \in \mathcal{K}$, and the default recommendation policy $\pi_d \in \Pi$; i.e., for all $x \in \mathcal{X}, s^k \in \mathcal{S}, k \in \mathcal{K}$,

$$\pi^*_\eta(s^k|x) = \frac{\pi_d(s^k|x) \cdot e^{\eta b_X(x)\bar{\beta}_\eta(s^k,x,\lambda^*_\eta)}}{\sum_{k\in\mathcal{K}} \pi_d(s^k|x) \cdot e^{\eta b_X(x)\bar{\beta}_\eta(s^k,x,\lambda^*_\eta)}}. \tag{9.7}$$

9.2.4 Interpretation of ZETAR from Employees' Perspectives

When ZETAR designs a fully customized recommendation policy (i.e., LoRC η goes to infinity), then the dual problem D_∞ is a Linear Program (LP) as shown in Proposition 8. Define $\hat{\beta}_\infty(x) := \beta_\infty(x)/b_X(x)$ where $\hat{\beta}_\infty(x) = 0$ if $b_X(x) = 0$.

Proposition 8. *When LoRC η goes to infinity, the dual problem D_∞ degenerates to the following Linear Program (LP):*

$$\begin{aligned} [D_\infty]: \quad & \min_{[\hat{\beta}_\infty(x)\in\mathbb{R}]_{x\in\mathcal{X}},[\lambda_\infty(s^k,a^l)\in\mathbb{R}^{0+}]_{k,l\in\mathcal{K}}} \quad \sum_{x\in\mathcal{X}} b_X(x)\hat{\beta}_\infty(x) \\ s.t. \quad & \hat{\beta}_\infty(x) \geq \bar{\beta}_\infty(s^k, x, \lambda_\infty), \forall k \in \mathcal{K}, \forall x \in \mathcal{X}. \end{aligned}$$

Proof. When $\eta \to \infty$, the upper and lower bounds for $\beta_\eta(x)$ in Proposition 7 attain the same value, which implies $\beta_\eta(x) = \max_{k\in\mathcal{K}} \eta b_X(x)\bar{\beta}_\eta(s^k, x, \lambda_\eta)$. Thus, constraint (a) of D_η is feasible. Constraint (b) and the objective function of the convex program D_η are equivalent to the constraint and the objective function of the LP D_∞, respectively. □

The dual problem D_∞ provides an interpretation of ZETAR with fully customized recommendation policies from an employee's perspective; i.e., each employee aims to minimize his effort to satisfy the security objective of the corporate network. Variable $\lambda_\infty(s^k, a^l)$ represents the employee's frequency to take action $a^l \in \mathcal{A}$ under recommendation $s^k \in \mathcal{S}$. The variable $\bar{\beta}_\infty(s^k, x, \lambda_\infty)$ represents the mixed security objective of the corporate network under AS $x \in \mathcal{X}$ and recommendation $s^k \in \mathcal{S}$, which involves the sum of the defender's utility $\bar{v}_D$ and the employee's expected utility, i.e., $\sum_{a^l \in \mathcal{A}} \lambda_\infty(s^k, a^l)[\bar{v}_U(x, a^k) - \bar{v}_U(x, a^l)]$. The variable $\hat{\beta}_\infty(x)$ represents the employee's effort at AS $x \in \mathcal{X}$, and the effort is required to satisfy the security objective at each AS for all recommendations. An employee who prioritizes convenience over security chooses the rate of actions to minimize his expected effort $\sum_{x \in \mathcal{X}} b_X(x) \hat{\beta}_\infty(x)$.

9.3 Characterization of Trust and Compliance

Section 9.2 provides a unified computational framework to design the optimal CT recommendation policy under any LoRC. In this section, we consider fully customized recommendation policies, i.e., $\eta = \infty$. We characterize the invariance of an employee's compliance status and the defender's optimal recommendation policy under linear utility transformations in Section 9.3.1. In Section 9.3.2, we provide a geometric characterization of the CT policy set based solely on an employee's incentive v_U. The characterizations are useful to develop efficient algorithms in Section 9.4 when employees' incentives are unknown. In Section 9.3.3, we characterize the optimal ACEL under different levels of misalignment between the defender's security objective v_D and an employee's incentive v_U.

9.3.1 Impact of Linear Utility Transformations

We define the linear utility transformation for the defender and an employee in Definition 26. Following Remark 13, if a recommendation $s \in \mathcal{S}$ is trustworthy (or untrustworthy) to both two employees, then they have the same compliant status under s. Lemma 4 illustrates the preservation of an employee's compliance status under linear transformations of v_U. The proof directly follows from Definition 22.

Definition 26 (Linear Utility Transformation). *Define the linear transformation of a player's utility with a scaling factor $\rho_p^{sa} \in \mathbb{R}$ and translation factors $[\rho_p^{tr}(y,x) \in \mathbb{R}]_{x\in\mathcal{X},y\in\mathcal{Y}}$ as $v_p^{lt}(y,x,a) := \rho_p^{sa} v_p(y,x,a) + \rho_p^{tr}(y,x)$ for all $x \in \mathcal{X}, y \in \mathcal{Y}, a \in \mathcal{A}, p \in \{D,U\}$.*

Lemma 4 (Preservation of Compliance Status). *Interacting with the same defender, two employees with incentives v_U and v_U^{lt}, respectively, have the same compliance status for all recommendation $s \in \mathcal{S}$ under any recommendation policy $\pi \in \Pi$. Moreover, the defender applies the same optimal recommendation policy to both employees.*

Lemma 5 characterizes ZETAR from the perspective of linear systems; i.e., a linear transformation of v_D results in a linear transformation of the defender's ASeL in (9.5) and the ACEL in Definition 25 for any recommendation policy $\pi \in \Pi$. The proof directly follows from (9.5) and the fact that $\max_{\pi\in\Pi} J_D(\pi, b_X, \bar{v}_D^{lt}, \bar{v}_U) = \max_{\pi\in\Pi} J_D(\pi, b_X, \bar{v}_D, \bar{v}_U)$.

Lemma 5 (Preservation of Linearity). *Interacting with the same employee, the ASeL of two defenders with security objectives v_D and v_D^{lt} are linearly dependent, i.e., $J_D(\pi, b_X, \bar{v}_D^{lt}, \bar{v}_U) = \rho_D^{sa} J_D(\pi, b_X, \bar{v}_D, \bar{v}_U) + \sum_{x\in\mathcal{X}} \sum_{y\in\mathcal{Y}} b_{Y,X}(y,x)\rho_D^{tr}(y,x)$, for all $\pi \in \Pi$. Moreover, the two defenders use the same optimal recommendation policy.*

Remark 17 (Policy Invariance). *Lemmas 4 and 5 show that linear utility transformation does not affect the optimal recommendation policy. The structures of an employee's incentive and the defender's security objective play more critical roles in compliance status and ASeL than their absolute values.*

9.3.2 Geometric Characterization of CT Sets, ASaL, and ASeL

We define the following notations for the matrix representations of recommendation policies and utilities in Sections 9.3.2 and 9.4. Define $\hat{v}_p^k := [b_X(x^1)\bar{v}_p(x^1, a^k), b_X(x^2)\bar{v}_p(x^2, a^k), \cdots, b_X(x^I)\bar{v}_p(x^I, a^k)]^T \in \mathbb{R}^{1\times I}, p \in \{D, U\}$, for all $k \in \mathcal{K}$. For each recommendation $s^k \in \mathcal{S}, k \in \mathcal{K}$, and the shorthand notation $\hat{\pi}^{k,i} := \pi(s^k|x^i)$, we can define an I-dimension vector $\hat{\pi}^k = [\hat{\pi}^{k,1}, \cdots, \hat{\pi}^{k,I}] \in \hat{\Pi}^k$. By definition, $\sum_{k=1}^K \hat{\pi}^k = [1, 1, \cdots, 1] \in \mathbb{R}^I$. Then, a recommendation policy $\pi \in \Pi$ has an equivalent matrix form as $\hat{\pi} := [\hat{\pi}^1, \cdots, \hat{\pi}^K]^T \in \hat{\Pi}$. Analogously, the k-th PT, k-th PU, and CT recommendation policies in matrix forms compose sets $\hat{\Pi}_{pt}^k$, $\hat{\Pi}_{pu}^k$, and $\hat{\Pi}_{ct}$, respectively. In Proposition 9, we identify $\hat{\pi}^k \in \hat{\Pi}^k$ as the sufficient component of $\hat{\pi} \in \hat{\Pi}$ to determine the trustworthiness of recommendation $s^k \in \mathcal{S}$.

Proposition 9 (**Minimal Sufficiency for Trustworthy Recommendations**). *Policy vector $\hat{\pi}^k \in \hat{\Pi}^k$ is the minimal sufficient component of the policy matrix $\hat{\pi} \in \hat{\Pi}$ to determine the trustworthiness of recommendation $s^k \in \mathcal{S}$.*

Proof. Based on (9.2) and Definition 22, a recommendation $s^k \in \mathcal{S}, k \in \mathcal{K}$, is trustworthy if and only if $\hat{\pi}^k[\hat{v}_U^k - \hat{v}_U^l] \geq 0, \forall l \in \mathcal{K}$ (i.e., the matrix representation of constraint (c) in P_η). □

Proposition 9 leads to the policy separability principle in Remark 18; i.e., the defender can design the k-th policy vector $\hat{\pi}^k \in \hat{\Pi}^k$ separately for all $k \in \mathcal{K}$ to learn the k-th PT policy set. The policy separability contributes to efficient CT policy set learning algorithms in Section 9.4. We characterize an employee's ASaL, the convexity of the CT policy set, and the defender's ASeL in Lemmas 6-8, respectively. Section 9.5.3 illustrates these characterizations when $I = J = K = 2$.

Remark 18 (**Policy Separability**)**.** *The defender can determine the k-th PT policy set, i.e., $\hat{\Pi}^k_{pt}$, independently from other PT policy sets $\hat{\Pi}^{k'}_{pt}, \forall k' \in \mathcal{K} \setminus \{k\}$, to determine CT policy set $\hat{\Pi}_{ct}$.*

Lemma 6 (**PWL and Convex of ASaL**)**.** *ASaL $J_U(\pi, b_X, \bar{v}_U)$ is PieceWise Linear (PWL) and convex in $\hat{\pi}^k \in \hat{\Pi}^k, \forall k \in \mathcal{K}$.*

Proof. Following (9.4), we can represent an employee's ASaL as $J_U(\pi, b_X, \bar{v}_U) = \sum_{k \in \mathcal{K}} \max_{a \in \mathcal{A}}[\sum_{x \in \mathcal{X}} b_X(x)\pi(s^k|x)\bar{v}_U(x,a)]$. Since $\sum_{x \in \mathcal{X}} b_X(x)\pi(s^k|x)\bar{v}_U(x,a)$ is an linear function in $\hat{\pi}^k \in \hat{\Pi}^k, \forall k \in \mathcal{K}$, and $a \in \mathcal{A}$, the point-wise maximum of a group of linear functions in (9.4) leads to a PieceWise Linear (PWL) and convex function concerning $\hat{\pi}^k \in \hat{\Pi}^k, \forall k \in \mathcal{K}$. Then, the sum of a group of PWL and convex functions remains PWL and convex. □

Denote $\mathcal{C}^k_l := \{\hat{\pi} \in \hat{\Pi} | \hat{\pi}^k[\hat{v}^l_U - \hat{v}^h_U] \geq 0, \forall h \in \mathcal{K}\}$ as the set of recommendation policies that induce action $a^l \in \mathcal{A}$ under recommendation $s^k \in \mathcal{S}$. Define $\mathcal{C}_{\{l_1,\cdots,l_K\}} := \cap_{k=1}^K \mathcal{C}^k_{l_k}$ as the set of recommendation policies that induce action $a^{l_k} \in \mathcal{A}$ under recommendation $s^k \in \mathcal{S}$ for all $k \in \mathcal{K}$. Within each (possibly empty) set $\mathcal{C}_{\{l_1,\cdots,l_K\}}, \forall l_1, \cdots, l_K \in \mathcal{K}$, we can represent the defender's ASeL in (9.5) equivalently as the matrix form $\hat{J}_D(\hat{\pi}, \hat{v}_D, v_U) := \sum_{k \in \mathcal{K}} \hat{\pi}^k \hat{v}^{l_k}_D, \ \forall \hat{\pi} \in \mathcal{C}_{\{l_1,\cdots,l_K\}}$.

Lemma 7. *The K^K sets $\mathcal{C}_{\{l_1,\cdots,l_K\}}, \forall l_1, \cdots, l_K \in \mathcal{K}$, are mutually exclusive and convex. The union of these sets composes the entire recommendation policy set, i.e., $\hat{\Pi} = \cup_{l_1,\cdots,l_K \in \mathcal{K}} \mathcal{C}_{\{l_1,\cdots,l_K\}}$. The k-th PT policy set $\mathcal{C}^k_k, \forall k \in \mathcal{K}$, and the CT policy set $\hat{\Pi}_{ct} = \mathcal{C}_{\{1,\cdots,K\}} = \cap_{k=1}^K \mathcal{C}^k_k$ are convex.*

Proof. The convexity of set $\mathcal{C}^k_l$ directly follows its definition. The properties of the mutual exclusiveness and the union directly come from the definition of $\mathcal{C}_{\{l_1,\cdots,l_K\}}$.

Definition 23 leads to $\hat{\Pi}_{ct} = \cap_{k=1}^{K} \mathcal{C}_k^k$. Since the intersection of any collection of convex sets is convex, sets $\mathcal{C}_{\{l_1,\cdots,l_K\}}$ and $\hat{\Pi}_{ct}$ are convex. □

Lemma 8 (PWL of ASeL). *ASeL $\hat{J}_D(\hat{\pi}, \hat{v}_D, v_U)$ is (possibly discontinuous) piecewise linear concerning $\hat{\pi}^k \in \hat{\Pi}^k, \forall k \in \mathcal{K}$.*

Proof. Based on Lemma 7, the entire recommendation policy set $\hat{\Pi}$ is divided into K^K mutually exclusive (possibly empty) sets determined by an employee's incentive v_U. Within each set $\mathcal{C}_{\{l_1,\cdots,l_K\}}$, the defender's ASeL $\hat{J}_D(\hat{\pi}, \hat{v}_D, v_U)$ in matrix form is linear in $\hat{\pi}^k \in \hat{\Pi}^k, \forall k \in \mathcal{K}$. □

9.3.3 Optimal ACEL under Incentive Misalignment

We first classify the insiders into three incentive categories in Definition 27 based on the alignment of their incentives with the defender's security objective. Denote $\chi(\mathcal{K}) := [\chi(1), \chi(2), \cdots, \chi(K)]$ as a permutation of set $\mathcal{K}$, i.e., $\chi(k) \in \mathcal{K}, \forall k \in \mathcal{K}$, and $\chi(k) \neq \chi(k')$ if $k \neq k', \forall k, k' \in \mathcal{K}$.

Definition 27 (Incentive Categories). *Consider the defender with security objective v_D. An insider is categorized as amenable (resp. malicious) if he shares the same (resp. opposite) preference ranking with the defender concerning actions for each SP and AS; i.e., for any given $x \in \mathcal{X}, y \in \mathcal{Y}$, if $v_U(y, x, a^{\chi(1)}) \geq v_U(y, x, a^{\chi(2)}) \geq \cdots \geq v_U(y, x, a^{\chi(K)})$, then $v_D(y, x, a^{\chi(1)}) \geq$ (resp. $\leq$)$v_D(y, x, a^{\chi(2)}) \geq$ (resp. $\leq$)$\cdots \geq$ (resp. $\leq$)$v_D(y, x, a^{\chi(K)})$. An insider is self-interested if he is neither amenable nor malicious.*

An amenable insider has a strong sense of responsibility to enhance security and prioritizes security over convenience. A malicious insider, e.g., a disgruntled employee or an employee whose credentials have been stolen, can misbehave or sabotage corporate security on purpose. Self-interested insiders represent the majority of employees who are willing to follow security rules when there is no conflict of interests. Following Definition 26 and Remark 17, linear transformations of a malicious, self-interested, or amenable employee's incentive do not change his incentive category. Lemma 9 characterizes the optimal recommendation policy and the ACEL when an employee's incentive and the defender's security objective are linearly dependent.

Lemma 9. *Consider linearly dependent incentives of an employee and the defender with a scaling factor $\rho^{sa}_{D,U} \in \mathbb{R}$ and translation factors $[\rho^{tr}_{D,U}(y,x) \in \mathbb{R}]_{y\in\mathcal{Y},x\in\mathcal{X}}$, i.e., $v_D(y,x,a) = \rho^{sa}_{D,U} v_U(y,x,a) + \rho^{tr}_{D,U}(y,x)$ for all $y \in \mathcal{Y}, x \in \mathcal{X}, a \in \mathcal{A}$. Then, the following two statements hold.*

1. *$J^{acel,*}_D(b_X, \bar{v}_D, \bar{v}_U) = 0, \forall b_X \in \mathcal{B}_X$, if and only if $\rho^{sa}_{D,U} \leq 0$. Zero-information recommendation policy $\pi_z \in \Pi_{ct}$ achieves the optimal ACEL.*

2. *$J^{acel}_D(\pi, b_X, \bar{v}_D, \bar{v}_U) \geq 0, \forall \pi \in \Pi, \forall b_X \in \mathcal{B}_X$, if and only if $\rho^{sa}_{D,U} > 0$. Full-information recommendation policy $\pi_f \in \Pi_{ct}$ achieves the optimal ACEL, and the following holds: $J^{acel,*}_D(b_X, \bar{v}_D, \bar{v}_U) = \rho^{sa}_{D,U} \sum_{x\in\mathcal{X}} b_X(x) \max_{a\in\mathcal{A}} \bar{v}_U(x,a) - \rho^{sa}_{D,U} \max_{a\in\mathcal{A}} \sum_{x\in\mathcal{X}} b_X(x)\bar{v}_U(x,a)$.*

Proof. Under $\pi_z \in \Pi_{ct}$ and the linear dependency condition, the ASeL in (9.5) becomes $\tilde{J}_D(\pi_z, b_{Y,X}, v_D, v_U) = \sum_{y\in\mathcal{Y},x\in\mathcal{X}} b_{Y,X}(y,x)[\rho^{sa}_{D,U} v_U(y,x,a_0) + \rho^{tr}_{D,U}(y,x)] = \rho^{sa}_{D,U} \max_{a\in\mathcal{A}} \sum_{y\in\mathcal{Y},x\in\mathcal{X}} b_{Y,X}(y,x) v_U(y,x,a) + \sum_{y\in\mathcal{Y},x\in\mathcal{X}} b_{Y,X}(y,x)\rho^{tr}_{D,U}(y,x)$. Based on the concavification technique in [109], $\tilde{J}_D(\pi^*, b_{Y,X}, v_D, v_U)$ is the concave closure of $\tilde{J}_D(\pi_z, b_{Y,X}, v_D, v_U)$ over $b_{Y,X} \in \mathcal{B}_{Y,X}$. As $\max_{a\in\mathcal{A}} \sum_{y\in\mathcal{Y}} \sum_{x\in\mathcal{X}} b_{Y,X}(y,x) v_U(y,x,a)$ is PWL and convex concerning $b_{Y,X}(y,x), \forall y \in \mathcal{Y}, x \in \mathcal{X}$, $\tilde{J}_D(\pi_z, b_{Y,X}, v_D, v_U)$ is PWL and convex (resp. concave) in $b_{Y,X} \in \mathcal{B}_{Y,X}$ if and only if $\rho^{sa}_{D,U} > 0$ (resp.

$\rho^{sa}_{D,U} \leq 0$). Then, the convex hull of a concave function is itself, and the optimal ACEL equals 0. Meanwhile, the convex hull of a convex function depends only on the vertices of the convex set $\mathcal{B}_{Y,X}$, i.e., $v_D(y, x, \tilde{a}^{max}(y, x)), \forall x \in \mathcal{X}, y \in \mathcal{Y}$. We arrive at the result: $\tilde{J}_D(\pi^*, b_{Y,X}, v_D, v_U) = \sum_{x\in\mathcal{X}} b_X(x)\bar{v}_D(x, a^{max}_U(x)) = \rho^{sa}_{D,U} \sum_{x\in\mathcal{X}} b_X(x) \max_{a\in\mathcal{A}} \bar{v}_U(x, a) + \sum_{y\in\mathcal{Y},x\in\mathcal{X}} b_{Y,X}(y, x)\rho^{tr}_{D,U}(y, x)$, under the optimal recommendation policy $\pi_f \in \Pi_{ct}$. □

Remark 19. *Since a recommendation policy $\pi \in \Pi$ has impact on $b^{\pi}_{Y,X}$ as shown in Lemma 3, the incentive $\bar{v}_D$ is not a constant as b_Y changes. Thus, $\tilde{J}_D(\pi^*, b_{Y,X}, v_D, v_U)$ is linear in $b_{Y,X} \in \mathcal{B}_{Y,X}$ but not $b_X \in \mathcal{B}_X$ (or $b_Y \in \Delta\mathcal{Y}$) in general. If mapping $\psi \in \Psi$ is non-stochastic, then ZETAR degenerates to the Bayesian persuasion model in [109], and $J_D(\pi^*, b_X, \bar{v}_D, \bar{v}_U)$ is linear in $b_X \in \mathcal{B}_X$ (or $b_Y \in \Delta\mathcal{Y}$).*

According to Definition 27, an insider is amenable if $\rho^{sa}_{D,U} > 0$ and malicious if $\rho^{sa}_{D,U} \leq 0$. Therefore, Lemma 9 provides a closed-form solution of the optimal ACEL for malicious and amendable insiders. We extend the discussion on the optimal ACEL concerning amenable and malicious insiders in Proposition 10 and 11, respectively. For an amendable employee, Proposition 10 shows that within the entire action preference, the optimal action is the decisive factor.

Proposition 10. *If the incentives of an employee and the defender share the same optimal action $\tilde{a}^{max}(y, x) \in \mathcal{A}$ for each AS $x \in \mathcal{X}$ and SP $y \in \mathcal{Y}$, then for all $b_{Y,X} \in \mathcal{B}_{Y,X}$, full-information recommendation policy $\pi_f \in \Pi_{ct}$ achieves the optimal ACEL, and*

$$J^*_D(b_X, \bar{v}_D, \bar{v}_U) = J^*_D(b_X, \bar{v}_U, \bar{v}_U) - \sum_{x\in\mathcal{X}} b_X(x)\delta(x), \tag{9.8}$$

where $J_D^*(b_X, \bar{v}_U, \bar{v}_U) = \sum_{x\in\mathcal{X}} b_X(x) \max_{a\in\mathcal{A}} \bar{v}_U(x,a)$ *and* $\delta(x) := \bar{v}_U(x, a^{max}(x)) - \bar{v}_D(x, a^{max}(x)), \forall x \in \mathcal{X}$*. Moreover,* $J_D^{acel}(\pi, b_X, \bar{v}_D, \bar{v}_U) \geq 0, \forall \pi \in \Pi, b_X \in \mathcal{B}_X$*.*

Proof. Based on Lemma 9, if we construct $v_D^0 := v_U$, then $\tilde{J}_D(\pi_z, b_{Y,X}, v_D^0, v_U)$ is PWL and convex in $b_{Y,X} \in \mathcal{B}_{Y,X}$, and $\tilde{J}_D(\pi^*, b_{Y,X}, v_D^0, v_U)$ only depends on $v_D^0(y, x, \tilde{a}^{max}(y,x)), \forall x \in \mathcal{X}, y \in \mathcal{Y}$, for all $b_{Y,X} \in \mathcal{B}_{Y,X}$. Thus, we can construct v_D^1 from v_D^0 to make $\tilde{J}_D(\pi^*, b_{Y,X}, v_D^1, v_U) = \tilde{J}_D(\pi^*, b_{Y,X}, v_D^0, v_U)$ as long as $v_D^1(y, x, \tilde{a}^{max}(y,x)) = v_D^0(y, x, \tilde{a}^{max}(y,x)), \forall y \in \mathcal{Y}, x \in \mathcal{X}$.

If an employee with incentive v_U and the defender with security objective v_D prefer the same optimal action for each AS and SP, we can construct $\bar{v}_D^{eq}(x,a) := \bar{v}_D(x,a) + \delta(x)$ such that $\bar{v}_D^{eq}(x, a^{max}(x)) = \bar{v}_U(x, a^{max}(x)), \forall x \in \mathcal{X}$. Then, we have $J_D^*(b_X, \bar{v}_D^{eq}, \bar{v}_U) = J_D^*(b_X, \bar{v}_U, \bar{v}_U)$. Based on Lemma 5, $J_D^*(b_X, \bar{v}_D^{eq}, \bar{v}_U) = J_D^*(b_X, \bar{v}_D, \bar{v}_U) + \sum_{x\in\mathcal{X}} \delta(x)$, which leads to (9.8). Since $J_D(\pi, b_X, \bar{v}_D^{eq}, \bar{v}_U) \geq J_D(\pi_z, b_X, \bar{v}_D^{eq}, \bar{v}_U), \forall \pi \in \Pi, b_X \in \mathcal{B}_X$, based on Lemma 9, Lemma 5 leads to $J_D(\pi, b_X, \bar{v}_D, \bar{v}_U) \geq J_D(\pi_z, b_X, \bar{v}_D, \bar{v}_U)$. □

Remark 20 (**Sufficiency under Aligned Action Preference**)**.** *Proposition 10 shows that if an employee and the defender share the same optimal action* $\tilde{a}^{max}(y,x) \in \mathcal{A}$ *for each AS* $x \in \mathcal{X}$ *and SP* $y \in \mathcal{Y}$*, then* $v_D(y, x, \tilde{a}^{max}(y,x)), \forall x \in \mathcal{X}, y \in \mathcal{Y}$*, are the minimal sufficient component of the defender's security objective* v_D *to determine the defender's optimal ASeL.*

Remark 21 (**Simplified ZETAR Problem**)**.** *When* $v_D = v_U$*, the principal-agent problem* $\tilde{J}_D(\pi^*, b_{Y,X}, v_U, v_U)$ *is equivalent to a single-agent decision problem that directly solves* $\sum_{x\in\mathcal{X}} b_X(x) \max_{a\in\mathcal{A}} \bar{v}_U(x,a)$*. Thus, Proposition 10 contributes to an efficient computation of the defender's optimal ASeL when she shares the same* $\tilde{a}^{max}$ *with an employee.*

Remark 22 (**Full Information Disclosure to Amendable Employees**). *The defender should share full information (i.e., adopt $\pi_f \in \Pi_{ct}$) with amendable employees. By synchronizing information with compliant employees, the defender encourages amendable employees to contribute to corporate security.*

For malicious employees, we first introduce a class of invariant perturbations of the defender's security objective that achieve the same optimal ACEL of $J_D^{acel,*}(b_X, \bar{v}_D, \bar{v}_U) = 0$ in Proposition 11. Define shorthand notation $a^{min}(y,x) \in \text{arg}min_{a \in \mathcal{A}} v_D(y,x,a)$ for all $x \in \mathcal{X}, y \in \mathcal{Y}$.

Proposition 11 (**Compliance Equivalency under Security Objective Perturbations**). *We construct v_D^{ip} as a copy of $\bar{v}_D$ with the following revision: for each $x^i \in \mathcal{X}, i \in \mathcal{I}, y \in \mathcal{Y}$, if $a^{min}(y,x^j) \neq a^{min}(y,x^i)$, then $v_D^{ip}(y,x^j,a^{min}(y,x^i)) \leq v_D(y,x^j,a^{min}(y,x^i)), \forall j \in \mathcal{I} \setminus \{i\}$. If there exist $\rho_{D,U}^{sa} < 0$ and $\rho_{D,U}^{tr}(y,x) \in \mathbb{R}$ such that $v_D(y,x,a) = \rho_{D,U}^{sa} v_U(y,x,a) + \rho_{D,U}^{tr}(y,x)$ for all $y \in \mathcal{Y}, x \in \mathcal{X}, a \in \mathcal{A}$, then $J_D^*(b_X, \bar{v}_D^{ip}, \bar{v}_U) = J_D^*(b_X, \bar{v}_D, \bar{v}_U)$.*

Proof. Lemma 9 shows that function $\tilde{J}_D(\pi^*, b_{Y,X}, v_D, v_U)$ as the concave closure of $\tilde{J}_D(\pi_z, b_{Y,X}, v_D, v_U)$ is PWL and concave in $b_{Y,X} \in \mathcal{B}_{Y,X}$ if $\rho_{D,U}^{sa} < 0$. Based on the geometry, changing v_D to v_D^{ip} does not affect the concave closure (yet $\pi^* \in \Pi$ can change and does not contain zero information), i.e., $J_D^*(b_X, \bar{v}_D^{ip}, \bar{v}_U) = J_D^*(b_X, \bar{v}_D, \bar{v}_U)$. □

Remark 22 provides the defender with the guidance of full-information disclosure to amendable employees. Based on Lemma 9 and Proposition 10, it is natural to conjecture that the defender should disclose zero information to malicious employees. However, it does not hold, and we present a counterexample in Proposition 11; i.e., although an employee with incentive v_U is malicious to both the defender with security objective v_D and the one with v_D^{ip}, zero information recommendation policy is not the optimal policy for the defender with v_D^{ip}. Thus, Proposition 11 leads to the strategic information disclosure guideline in Remark 23. It further shows that ZETAR can improve an insider's compliance even if he is malicious based on the incentive categorization in Definition 27.

Remark 23 (**Strategic Information Disclosure to Malicious Insiders**). *The defender should disclose information strategically rather than hide information (i.e., adopting $\pi_z \in \Pi$) even when the employee is malicious and tends to take an action that results in the least utility to the defender.*

9.4 Feedback Design for Unknown Incentives

When the defender knows an employee's incentive v_U, we can use primal and dual convex programs in Section 9.2 to compute the optimal recommendation policy π_η^* for a given LoRC $\eta \in \mathbb{R}^{0+}$. In practice, however, employees' incentives usually remain unknown to the defender, and the defender may not be able to formulate constraint (c) in P_η to determine the CT policy set. To this end, we provide a feedback design approach in Section 9.4 for the defender to learn the optimal recommendation policy based on the employees' responses to recommendations, as shown in Fig. 7.1. In the proposed learning algorithms, the defender needs no prior knowledge nor trust in an employee's incentives. The zero-trust audit of all employees provides the defender with their behaviors to learn incentives.

A straightforward feedback learning paradigm optimizes the defender's ASeL $J_D(\pi, b_X, \bar{v}_D, \bar{v}_U)$ over all recommendation policies in set $\hat{\Pi}$ directly. For a new employee with an unknown incentive, the defender at stage $m \in \{1, 2, \cdots\}$ recommends actions according to a recommendation policy $\hat{\pi}_m \in \hat{\Pi}$. Then, the defender observes the employee's responses to these recommendations and evaluates her ASeL. At stage $m+1$, the defender uses her ASeL evaluation to update the recommendation policy from $\hat{\pi}_m \in \hat{\Pi}$ to $\hat{\pi}_{m+1} \in \hat{\Pi}$. The update rule depends on bespoke optimization methods (e.g., simulated annealing, Bayesian optimization, and reinforcement learning). The above learning paradigm is universal yet inefficient and does not guarantee global optimality. In Algorithms 8 and 9, we design efficient feedback learning algorithms by exploiting the ZETAR features characterized in Section 9.3. In particular, the defender only learns the CT policy set $\hat{\Pi}_{ct}$ and then uses the primal convex program P_η in Section 9.2 to compute the optimal recommendation policy $\hat{\pi}^*_\eta$ and the optimal ACEL $J_D^{acel,*}$. The defender can achieve it as she knows her security objective v_D to compute the objective function in P_η. Based on Definition 23, we only need to learn all the PT policy sets $\hat{\Pi}^k_{pt}, \forall k \in \mathcal{K}$, to determine the CT policy set.

Following the matrix representation in Section 9.3.2, the k-th row vector $\hat{\pi}^k \in \hat{\Pi}^k$ of a recommendation policy $\hat{\pi} \in \hat{\Pi}$ can be equivalently represented a point, denoted by $(p_k^1, \cdots, p_k^I)$, in the unit hypercube of dimension I, where the coordinate $p_k^i = \hat{\pi}^{k,i} \in [0,1], \forall k \in \mathcal{K}, i \in \mathcal{I}$. We refer to a point in the k-th hypercube as a k-th PT point if it represents the k-th row vector $\hat{\pi}^k$ of a k-th PT recommendation policy $\hat{\pi} \in \hat{\Pi}^k_{pt}$. Since the k-th row $\hat{\pi}^k$ of $\hat{\pi}$ is sufficient to determine whether $\hat{\pi}$ is PT based on Proposition 9, learning the k-th PT set $\hat{\Pi}^k_{pt}$ is equivalent to determining the region formulated by the k-th PT points. We refer to the region as the k-th PT region, which is a convex polytope in the k-th hypercube of dimension I based on Lemma 7. Since a convex polytope can be uniquely represented by its vertices, we develop the following two algorithms to obtain the vertex representation (V-representation) of these regions. Due to the policy separability principle in Remark 18, we can determine the V-representation of the k-th PT region independently for each $k \in \mathcal{K}$. For any point $(p_k^1, \cdots, p_k^I)$ of the k-th hypercube, we define $\Omega(p_k^1, \cdots, p_k^I) \subseteq \hat{\Pi}$ as the set of recommendation policies whose k-th row vectors satisfy $\hat{\pi}^k = [p_k^1, \cdots, p_k^I]$. In Algorithm 8, we determine the whether the 2^I vertices of the k-th hypercube are k-th PT points. Let $V^k := \{(p_k^1, \cdots, p_k^I) | p_k^i \in \{0, 1\}, \forall i \in \mathcal{I}\}$ be the set of these 2^I vertices. Among these 2^I vertices, the k-th PT ones compose the k-th PT cube-vertex set denoted as $V_{pt}^k \subseteq V^k$.

Algorithm 8: Algorithm to learn the k-th PT cube-vertex set V_{pt}^k for a given employee.

```
Initialize the $k$-th PT cube-vertex set $V_{pt}^k \leftarrow \emptyset$;
foreach vertex $(p_k^1, \cdots, p_k^I) \in V^k$ do
    while $s^k$ has not been recommended, i.e., $k' \neq k$ do
        Recommend $s^{k'} \in \mathcal{S}$ randomly based on a recommendation policy
          $\hat{\pi} \in \Omega(p_k^1, \cdots, p_k^I)$;
    if recommendation $s^k$ induces $a^k \in \mathcal{A}$ then $V_{pt}^k \leftarrow V_{pt}^k \cup \{(p_k^1, \cdots, p_k^I)\}$;
Return the $k$-th PT cube-vertex set $V_{pt}^k$;
```

In Algorithm 9, we determine the vertex coordinates of the k-th PT region. As a convex polytope, the region contains a finite set of polytope-vertices defined as $\bar{V}_{pt}^k$. Since the k-th PT region is determined by a hyperplane in the k-th hypercube, its vertices are on the edges, and it contains all the elements in the k-th PT cube-vertex set V_{pt}^k as shown in the initialization step in line 7. Each cube-vertex $(p_k^1, \cdots, p_k^I) \in V_{pt}^k$ has I neighboring cube-vertices, and the coordinate of its i-th neighboring cube-vertex is $(p_k^1, \cdots, p_k^{i-1}, 1 - p_k^i, p_k^{i+1}, \cdots, p_k^I)$. After we select a k-th PT cube-vertex in line 8, we search over its I neighboring cube-vertices in line 9. If the neighboring vertex is also k-th PT, then the points on the edge of these two cube-vertices are both k-th PT. If the neighboring vertex is not k-th PT as shown in line 10, then there is an additional polytope-vertex on the edge of these two cube-vertices. As shown in lines $11-17$, we use the binary search to learn the coordinate of this additional polytope-vertex and add it to the k-th polytope-vertex set $\bar{V}_{pt}^{k}$ in line 18. In particular, the binary search adopts an accuracy $\epsilon > 0$ used in the stopping criteria shown in line 12. For the worst case where a polytope-vertex is close to a cube-vertex, we need $N \in \mathbb{Z}^{+}$ iterations to reach the stop criteria, i.e., $(1/2)^{N} \leq \epsilon$, which leads to $N \geq \log_2(1/\epsilon)$. Since an I-dimensional hypercube has $2^{n-1}n$ edges, Algorithm 9 is guaranteed to stop within $2^{n-1}n\log_2(1/\epsilon)$ steps.

Algorithm 9: Algorithm to learn the polytope-vertex set $\bar{V}_{pt}^{k}$ for a given employee.

```
97  Initialize $\bar{V}_{pt}^{k} \leftarrow V_{pt}^{k}$, and accuracy $\epsilon > 0$;
98  foreach $k$-th PT vertex $(p_k^1, \cdots, p_k^I) \in V_{pt}^{k}$ do
99      for $i \leftarrow 1$ to $I$ do
100         if $(p_k^1, \cdots, p_k^{i-1}, 1-p_k^i, p_k^{i+1}, \cdots, p_k^I) \notin V_{pt}^{k}$ then
101             $lb \leftarrow 0$ and $ub \leftarrow 1$;
102             while $ub - lb > \epsilon$ do
103                 Recommend $s \in \mathcal{S}$ randomly based on
                    $\hat{\pi} \in \Omega(p_k^1, \cdots, p_k^{i-1}, \frac{lb+ub}{2}, p_k^{i+1}, \cdots, p_k^I)$;
104                 if $s = s^k$ and $p_k^i = 0$ then
105                     if Employee takes action $a^k \in \mathcal{A}$ then $lb = \frac{lb+ub}{2}$ else
                          $ub = \frac{lb+ub}{2}$;
106                 else $s = s^k$ and $p_k^i = 1$
107                     if Employee takes action $a^k \in \mathcal{A}$ then $ub = \frac{lb+ub}{2}$ else
                          $lb = \frac{lb+ub}{2}$;
108             $\bar{V}_{pt}^{k} \leftarrow \bar{V}_{pt}^{k} \cup \{(p_k^1, \cdots, p_k^{i-1}, \frac{lb+ub}{2}, p_k^{i+1}, \cdots, p_k^I)\}$;
109 Return the $k$-th PT polytope-vertex set $\bar{V}_{pt}^{k}$;
```

After we obtain the V-representation of the k-th PT policy set, i.e., $\bar{V}_{pt}^{k}$, for each $k \in \mathcal{K}$, we can use facet enumeration methods (e.g., [11]) to obtain the half-space representation (H-representation) that can be directly used to construct the constraints in the primal convex program $P_\eta, \forall \eta \in \mathbb{R}^+$, in $\hat{\pi} \in \hat{\Pi}$. We provide a graphical illustration in Section 9.5.2 when $I = J = K = 2$.

9.5 Case Study

In this section, we illustrate the design of ZETAR in Fig. 7.1 under fully customized recommendation policies (i.e., $\eta = \infty$) to improve compliance for employees with different incentives.

9.5.1 Model Description

Following Fig. 9.2, we consider the binary security posture, i.e., $\mathcal{Y} = \{y^{hr}, y^{lr}\}$, where y^{hr} and y^{lr} represent the high-risk SP and the low-risk SP, respectively. For illustration purposes, we consider binary audit schemes, i.e., $\mathcal{X} = \{x^{sa}, x^{ta}\}$, where x^{sa} and x^{ta} represent stringent audit and tolerant audit, respectively. The employee's behaviors are categorized into binary actions, i.e., $\mathcal{A} = \{a^{ic}, a^{co}\}$, where a^{ic} and a^{co} represent non-compliant and compliant behaviors, respectively. Since employees have different risk attitudes toward gains and losses, we introduce a risk perception function κ^γ with parameter $\gamma := [\gamma_d, \gamma_s]$, where $\kappa^\gamma(v) := v^{\gamma_d}, v \geq 0$, and $\kappa^\gamma(v) := -\gamma_s(-v)^{\gamma_d}, v < 0$.

Employee's Intrinsic and Extrinsic Incentives

As shown in Table 9.1, we separate an employee's incentive v_U^γ under κ^γ into intrinsic incentive $v_{U,I}$ and extrinsic incentive $v_{U,E}^\gamma$. The extrinsic incentive in Table I(a) is independent of SP and captures the impact of AS on compliance. Compliant action $a^{co} \in \mathcal{A}$ introduces a compliance cost $c_U^{co} \in \mathbb{R}^+$ to an employee

$v_{U,E}^{\gamma}$	x^{sa}	x^{ta}
a^{ic}	$\kappa^{\gamma}(-c_D^{ic})$	0
a^{co}	$\kappa^{\gamma}(r_D^{co}) - c_U^{co}$	$-c_U^{co}$

(a) Extrinsic incentive.

$v_{U,I}$	y^{hr}	y^{lr}
a^{ic}	c_U^{hr}	c_U^{lr}
a^{co}	r_U^{hr}	r_U^{lr}

(b) Intrinsic incentive.

Table 9.1: Employee's utility $v_U^{\gamma} = v_{U,I} + v_{U,E}^{\gamma}$.

regardless of the AS. For example, an employee compliant with the air-gap rule has to spend additional time and effort to transfer data using a CD rather than a USB. Under AS $x^{sa} \in \mathcal{X}$, the defender introduces a reward $r_D^{co} \in \mathbb{R}^+$ and a penalty $c_D^{ic} \in \mathbb{R}^+$ to compliant action a^{co} and non-compliant action a^{ic}, respectively. We assume that the tolerant audit scheme x^{ta} introduces a reward and penalty of 0 to a^{co} and a^{ic}, respectively. The intrinsic incentive in Table I(b) is independent of AS and captures an employee's internal inclination to comply under different SP realizations. Under high-risk SP y^{hr} (resp. low-risk SP y^{lr}), an employee receives an intrinsic penalty (e.g., the guilty of misconduct) denoted by c_U^{hr} (resp. c_U^{lr}) to take non-compliant action a^{ic} and an intrinsic reward (e.g., the gratification of being compliance-seeking) denoted by r_U^{hr} (resp. r_U^{lr}) to take compliant action a^{co}. Based on Lemma 4, we can choose $\rho_U^{tr}(y^{hr}, x) = -r_U^{hr}$ and $\rho_U^{tr}(y^{lr}, x) = -r_U^{lr}$ for all $x \in \mathcal{X}$ without affecting the employee's compliance and the optimal recommendation policy $\pi^* \in \Pi$. Thus, without loss of generality, we calibrate $r_U^{hr} = r_U^{lr} = 0$ and $c_U^{hr}, c_U^{lr} \in \mathbb{R}$ and characterize the following three compliance attitudes of employees in Definition 28.

Definition 28 (**Compliance Attitudes**). *An employee is said to be compliance-seeking, compliance-averse, and compliance-neutral if both c_U^{hr} and c_U^{lr} are positive, negative, and zero, respectively.*

Defender's Security Objective

Table 9.2 illustrates the defender's security objective v_D. Following Section 9.1.2, stringent audit increases employees' pressures and reduces their working efficiency. We capture the efficiency reduction with a cost $c_D^{ca} \in \mathbb{R}^+$. When an employee takes a non-compliant action a^{ic}, the stringent audit x^{sa} requires an immediate correction from the employee to patch the induced vulnerability, which yields a reward of $r_D^{ca} \in \mathbb{R}^+$ in Table 9.2 regardless of the SP realizations. Meanwhile, the tolerant audit x^{ta} introduces no cost of efficiency reduction but additional risks of insider threats captured by the cost $c_D^{hr} \in \mathbb{R}^+$ in Table II(a) and $c_D^{lr} \in \mathbb{R}^+$ in Table II(b) under high-risk SP y^{hr} and low-risk SP y^{lr}, respectively. When an employee takes a compliant action a^{co}, the risk of insider threats is reduced to a minimum and is represented as the defender's payoff $r_D^{sa} \in \mathbb{R}$.

v_D	x^{sa}	x^{ta}
a^{ic}	$r_D^{ca} - c_D^{ca}$	$-c_D^{hr}$
a^{co}	$-c_D^{ca}$	r_D^{sa}

(a) v_D at high-risk SP.

v_D	x^{sa}	x^{ta}
a^{ic}	$r_D^{ca} - c_D^{ca}$	$-c_D^{lr}$
a^{co}	$-c_D^{ca}$	r_D^{sa}

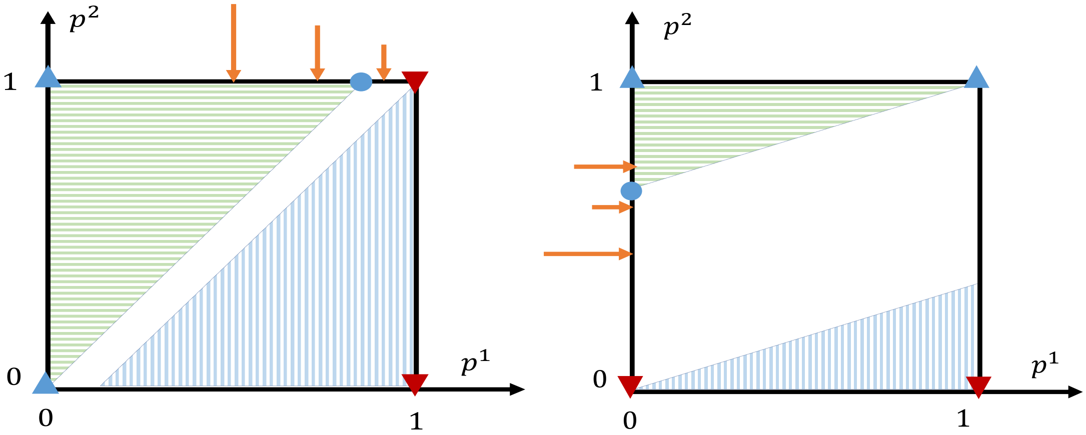
(b) v_D at low-risk SP.

Table 9.2: Defender's utility v_D at two SP states.

9.5.2 Graphical Illustration of Learning Algorithms

In this case study, the defender has no prior knowledge of an employee's risk and compliance attitudes. Moreover, the defender assigns no prior trust to the employees and applies the algorithms in Section 9.4 to learn their incentives. Fig. 9.3a and Fig. 9.3b illustrate the Algorithms under compliance-seeking and compliance-averse employees, respectively. Since $K = 2$, the recommendation policy $\hat{\pi} \in \hat{\Pi}$ can be equivalently represented as a point (p^1, p^2) in the unit hypercube of dimension $I = 2$

(i.e., a unit square), where $p^1 = \pi(s^{ic}|x^{sa}), p^2 = \pi(s^{ic}|x^{ta})$ as shown in Section 9.4. In the hypercube space illustrated by Fig. 9.3, the green and blue regions represent the CT and CU policy sets, respectively.



(a) Compliance-seeking insiders.

(b) Compliance-averse insiders.

Figure 9.3: The blue upward and the red downward triangles represent the CT and CU recommendations policies, respectively. The green (resp. blue) region with horizontal (resp. vertical) lines represents the CT (resp. CU) policy sets, respectively. The orange lines show the steps of the binary search in Algorithm 9.

Algorithm 8 determines whether the four vertices of the hypercube are the 1-st PT or not, which are illustrated by the blue upward and red downward triangles, respectively, in Fig. 9.3. Algorithm 9 further determines the additional polytope-vertex (represented by the blue circle in Fig. 9.3) of the 1-st PT polytope in green. From each blue triangle (line 8), if a neighboring cube-vertex (line 9) is also a blue triangle, then no additional polytope-vertices are needed to determine the green region (line 10). If the neighboring cube-vertex is red, then binary search is applied to determine the additional polytope-vertex. In Fig. 9.3a (resp. Fig. 9.3b), from the blue cube-vertex $(0, 1)$, the red neighboring cube-vertex is $(1, 0)$ (resp. $(0, 0)$), and the binary search adopts line 15 (resp. line 17). We use the orange lines in Fig. 9.3b to illustrate the binary search process, i.e., line 11 to 17 in Algorithm 9. The first step of the binary search (represented by the longest orange line) evaluates the recommendation policy represented by the point $(0, 1/2)$, and the policy is not the 1-st PT. Thus, we update the lower bound lb based on the *else* condition in line 16 of Algorithm 9. The second step (represented by the second-longest orange line) evaluates the recommendation policy represented by the point $(0, 3/4)$, and the policy is the 1-st PT. Thus, we update the upper bound ub based on the *then* condition in line 14. The third step (represented by the third-longest orange line) evaluates the recommendation policy represented by the point $(0, 5/8)$, and the policy is not the 1-st PT. Thus, we update the lower bound again. We repeat the above process of binary search until $ub - lb \leq \epsilon$ as shown in line 12, and we find the additional polytope vertex represented by the blue circle in Fig. 9.3b. After we obtain all the vertices of the polytope that represent the CT policy set, we can use facet enumeration methods to obtain the H-representation and construct the constraints of P_η concerning $p^1, p^2 \in [0, 1]$. For example, if the coordinate of the blue circle in Fig. 9.3b is $(0, w), w \in [0, 1]$, then the constraint is $p^2 \geq (1-w)p^1 + w$.

9.5.3 Numerical Results

We choose $\psi(x^{sa}|y^{hr}) = 0.8$ and $\psi(x^{sa}|y^{lr}) = 0.3$; i.e., the audit firm chooses a stringent audit with probability 0.8 and 0.3 under high-risk SP y^{hr} and low-risk SP y^{lr}, respectively.

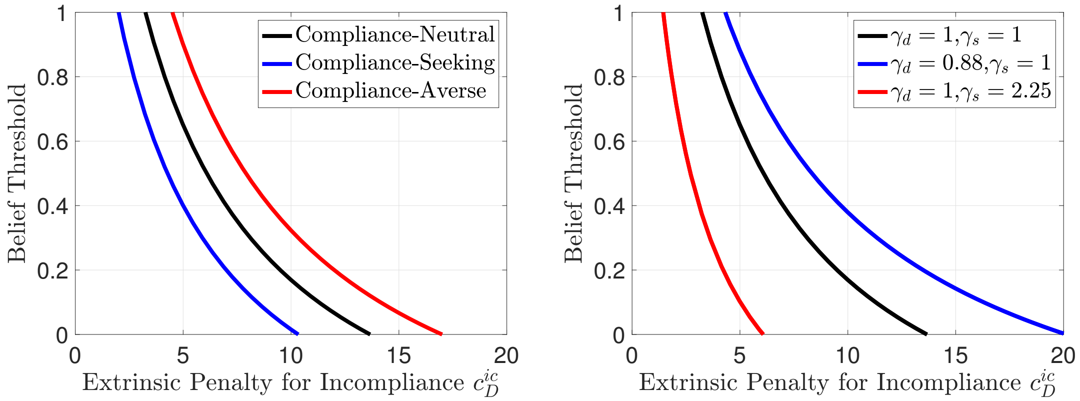
Compliance Threshold

Following Section 9.1.4, we investigate the initial compliance of an employee with three compliance attitudes in Definition 28 and different risk perception parameters

γ. Define $t_{ze} \in \mathbb{R}$ as the zero of the function

$$f(b_Y(y^{hr})) := \sum_{y \in \mathcal{Y}} b_Y(y) \sum_{x \in \mathcal{Y}} \psi(x|y)[v_U(y,x,a_1) - v_U(y,x,a_2)].$$

Let $t_{bt} := \max\{\min\{t_{ze}, 1\}, 0\}$ be the belief threshold of an employee. For binary actions, an insider adopts a *threshold policy* where $a_0 = a^{co}$ if $b_Y(y^{hr}) \geq t_{bt}$ and $a_0 = a^{ic}$ if $b_Y(y^{hr}) < t_{bt}$. Fig. 9.4 illustrates the belief threshold versus the non-compliance penalty $c_D^{ic} \in \mathbb{R}^+$. The plots show that increasing penalty c_D^{ic} can make insiders more likely to take compliant action a^{co} (i.e., a smaller belief threshold). Fixing the penalty value, compliance-averse (resp. compliance-seeking) insiders are the least (resp. most) likely to comply, i.e., the largest (resp. smallest) belief thresholds, among insiders with three compliance attitudes, as shown in Fig. 9.4a. In Fig. 9.4b, a larger γ_s in red represents a higher degree of loss aversion, which makes an insider more likely to comply. A small γ_d in blue enhances the effect of diminishing sensitivity, which makes a large penalty less effective to induce



(a) Insiders with three compliance attitudes under $\gamma_d = \gamma_s = 1$.

(b) Compliance-neutral insiders under distorted risk perceptions.

Figure 9.4: Insiders' belief thresholds $t_{bt} \in [0,1]$ to compliant actions versus the value of non-compliance penalty $c_D^{ic} \in \mathbb{R}^+$.

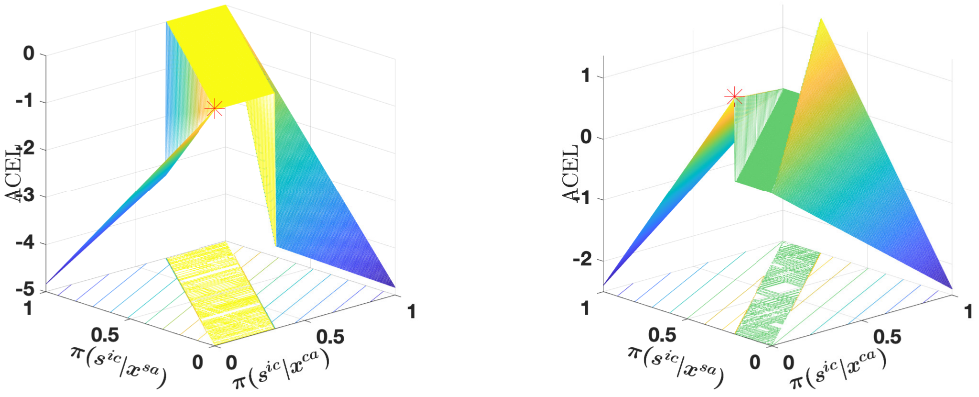
compliant behaviors.

Impacts of Recommendation Policies

Here, we specify $b(y^{hr}) = 0.2$ and $c_D^{ic} = 10$ to inspect the impact of recommendation policies on an employee's behaviors. Fig. 9.5 illustrates the impact of different recommendation policies $\hat{\pi} \in \hat{\Pi}$ on the ACEL under insiders with two compliance attitudes, which corroborate the PWL property in Lemma 8. Different compliance attitudes only affect the policy set partition denoted by $\mathcal{C}_l^k, \forall l, k \in \{ic, co\}$, following Section 9.3.2. In Fig. 9.5a, the policy sets (also illustrated in Fig. 9.3a) illustrated by the contour plots on the xy-plane are sets $\mathcal{C}_{ic,co}$, $\mathcal{C}_{co,co}$, and $\mathcal{C}_{co,ic}$, respectively, from left to right. In Fig. 9.5b, the policy sets (also illustrated in Fig. 9.3b) illustrated by the contour plots on the xy-plane are sets $\mathcal{C}_{ic,co}$, $\mathcal{C}_{ic,ic}$, and $\mathcal{C}_{co,ic}$, respectively, from left to right. These policy sets are convex as shown in Lemma 7. The sets $\mathcal{C}_{ic,co}$ and $\mathcal{C}_{co,ic}$ are CT and CU, respectively.

Fig. 9.5 illustrates that an improper recommendation policy may lead to a negative ACEL, but the optimal ACEL represented by the red star is always non-negative, as shown in Section 9.1.4. For compliance-seeking insiders, the defender's ISeL $J_D(\pi_z, b_X, \bar{v}_D, \bar{v}_U)$ and the optimal ASeL $J_D(\pi^*, b_X, \bar{v}_D, \bar{v}_U)$ are both 1.8. For compliance-averse insiders, the defender's ISeL $J_D(\pi_z, b_X, \bar{v}_D, \bar{v}_U)$ and the optimal ASeL $J_D(\pi^*, b_X, \bar{v}_D, \bar{v}_U)$ are -0.64 and 0.73, respectively.

Remark 24 (Adaptivity and Structural Improvement). *The above results show that ZETAR can well adapt to insiders with different compliance attitudes and achieve a structural improvement of compliance (from a negative ISeL to a positive ASeL) for compliance-averse insiders.*



(a) Compliance-seeking insiders.

(b) Compliance-averse insiders.

Figure 9.5: ACEL $J_D^{acel}(\pi, b_X, \bar{v}_D, \bar{v}_U)$ versus $\pi(s^{ic}|x^{sa}) \in [0,1]$ in x-axis and $\pi(s^{ic}|x^{ta}) \in [0,1]$ in y-axis when $\gamma_d = \gamma_s = 1$.

The Optimal ACEL

We illustrate the impacts of the optimal recommendation policy π^* on the defender's and an employee's utilities under different likelihoods of the high-risk SP In Fig. 9.6. Following Section 9.5.3, the belief threshold $t_{bt} \in [0,1]$, represented by the vertical dashed black lines, divides the entire prior belief region into the compliant region $b_Y(y^{hr}) \in [t_{bt}, 1]$ on the right and non-compliant region $b_Y(y^{hr}) \in [0, t_{bt})$ on the left, where an employee takes a^{co} and a^{ic}, respectively. Under the compliant regions, an employee tends to take compliant actions, resulting in zero ACEL and zero-information recommendation policy $\pi^*(s^{ic}|x^{sa}) = \pi^*(s^{ic}|x^{ta}) = 0$. Under the non-compliant regions where an employee tends not to comply, the optimal recommendation policy induces positive ACEL. The defender's ISeL in compliant regions is larger than the one in non-compliant regions as shown by the blue lines in the two regions. As an insider changes from being compliance-averse to compliance-seeking, his ASaL in black decreases in the non-compliant region, the belief threshold reduces (also illustrated in Fig. 9.4), and the peak of the optimal

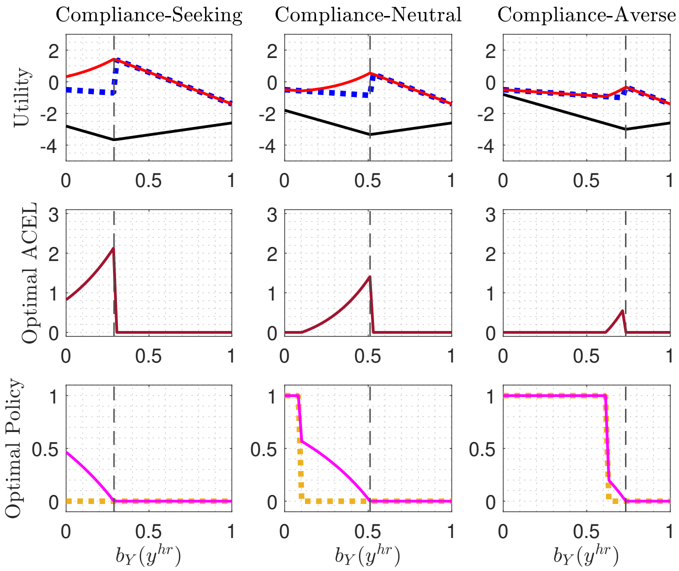


Figure 9.6: Utilities, the optimal ACEl, and the optimal recommendation policies in the first, second, and third rows, respectively, versus prior statistic $b_Y(y^{hr}) \in [0,1]$ concerning insiders with three compliance attitudes under $\gamma_s = \gamma_d = 1$. The defender's ISeL $J_D(\pi_z, b_X, \bar{v}_D, \bar{v}_U)$, her optimal ASeL $J_D(\pi^*, b_X, \bar{v}_D, \bar{v}_U)$, and an employee's optimal ASaL $J_U(\pi^*, b_X, \bar{v}_U)$ are in blue, red, and black, respectively. Two elements of the optimal recommendation policy, $\pi^*(s^{ic}|x^{sa})$ and $\pi^*(s^{ic}|x^{ta})$, are illustrated in orange and pink, respectively. The vertical dashed black lines represent the belief threshold $t_{bt} \in [0,1]$.

ACEL increases. The orange and pink lines illustrate that a large ACEL results from a more distinguished recommendation policy, i.e., a larger difference between $\pi^*(s^{ic}|x^{sa})$ and $\pi^*(s^{ic}|x^{ta})$. Moreover, the defender can recommend compliant actions, i.e., s^{co}, with a high probability as an insider changes from compliance-averse to compliance-seeking. Despite the linearity of the defender's ISeL in blue, her optimal ASeL in red and the optimal ACEL in brown are nonlinear in b_Y, as shown in Remark 19. In Fig. 9.6, we further observe that an employee's ISaL coincides with his optimal ASaL, both represented by the black solid lines for all $b_Y(y^{hr}) \in [0,1]$, which corroborates Proposition 5.

Chapter 10

Duplicity Game: Integrated Mechanism Design for Insider Threat Mitigation

Cyber deception technologies, e.g., honeypots, can be used to mitigate insider threats. The design of successful defensive deception relies on a formal approach that quantifies the strategic interactions of the three classes of players, including a defender, users, and adversaries. A useful framework to design cyber deception mechanisms needs to capture three main features. First, the defender, the users, and the adversaries are strategic players with clear but imperfectly aligned objectives or incentives. Second, the defender cannot distinguish adversaries from the normal users. For example, the defender does not know who is an adversarial insider when designing a security policy for the network. Apart from this, the defender cannot distinguish the type of users in the network concerning their objectives, resources, and trust values. Third, a sophisticated adversary behaves stealthily and intelligently, e.g., by conducting successful reconnaissance or acting like a normal user to gain access or trust.

In this work, we propose *Duplicity Games* (DG) as a mechanism design framework for defensive deception to elicit desirable security outcomes when a defender, normal users, and adversaries interact to attain their individual objectives. A DG is a two-stage game between a defender and a normal/adversarial user with two-sided asymmetric information. The defender, or the defensive deceiver, has private information of the system state. The user has a private type, which characterizes the user's objectives, trustworthiness, and attributes, e.g., normal or adversarial. At the first stage of the game, the defender designs three composable components of the mechanism, i.e., a *generator*, an *incentive modulator*, and a *trust manipulator*. The generator is a mechanism that stochastically generates signals or security policies based on the system's private information and system constraints. The modulator reshapes the user's incentive by creating constrained utility transfers between two players. The manipulator distorts the user's prior belief over the unknowns. These three components are together referred to as the GMM mechanism. After the mechanism is designed and implemented, the user observes the security policies, updates his trust through the Bayesian rule, and then responds to the GMM mechanism by taking an action that serves his objective. The optimal design of the GMM mechanisms anticipates the behaviors of different types of users under a given set of security policies and elicits desirable security behaviors. The GMM mechanisms we introduce here represent a class of multi-dimensional security mechanisms that control the security policies, the (dis)incentives, and the digital footprints (e.g., feature patterns and configurations of honeypots and normal servers).

10.1 Duplicity Game Model

We present a motivating example of insider threat mitigation in Section 10.1.1. Then, we present the structure of DG in Section 10.1.2 and the timeline of the GMM mechanism design in Section 10.1.3, respectively. Finally, we illustrate the relation of the DG-GMM mechanism to the Bayesian persuasion framework in Section 10.1.4.

10.1.1 Motivating Example of Insider Threat Mitigation

Insider threats have been a long-standing problem in cybersecurity. Due to their information, privilege, and resource advantages over external attackers, insider threats can circumvent classical defense techniques such as intrusion prevention and detection systems. As a result, defensive deception methods, such as honeypots, have been used for insider threat detection and mitigation (see e.g., [200, 224]). Theoretically, honeypots are assumed to achieve a zero false-positive rate and low false-negative rate by generating decoys accessed only by attackers. This assumption may not hold for insider threats. On the one hand, non-adversarial insiders who are curious or error-prone can access honeypots, which intensifies alert fatigue. On the other hand, adversarial insiders can access the internal information and fingerprint honeypots [35, 148] using features such as open ports, protocols, and error responses. To address these two challenges, we need to configure the honeypot and the normal servers strategically. The configuration needs to elicit desirable behaviors from both adversarial and non-adversarial insiders even though they have the same insider information. This work introduces three configuration methods that can be used independently or jointly; i.e., configure the feature pattern adaptively (see Example 4 for details), prolong or shorten the authentication time to change insiders' incentives, and misreport the percentage of honeypots to make use of the insiders' trust.

Categorization of Insiders' Motives

An insider's motive can be roughly classified into seven subcategories based on the VERIS Community Database (VCDB) [217]. We divide these subcategories of motives into three classes of motives: selfish, adversarial, and unintentional. They make up 12%, 26%, and 62%, respectively. The class of selfish motives includes fun, convenience, fear, or ideology. The adversarial motives include espionage, financial gain, or grudge. The category of unintentional motives refers to the negligent insiders who take no notice of the deceptive configuration and make habitual decisions. The incentives of unintentional insiders are often uncontrollable through incentives. Our incentive design mechanism here focuses on the class of the selfish insiders, who seek self-interest, and the adversarial ones, who seek to sabotage the organization.

Corporate Network with Insiders and Honeypots

Fig. 10.1 illustrates a corporate network with honeypots (denoted by x^H) and normal servers (denoted by x^N) as nodes. The Security Operation Center (SOC), or the defender, can privately determine the percentage, the location, and the configuration of honeypots in the corporate network. The goal of the defender is to elicit desirable behaviors from the selfish insiders (denoted by θ^g) and the adversarial insiders (denoted by θ^b). Both types of insiders can take harmful actions intentionally yet for different reasons or motives. For example, selfish insiders may

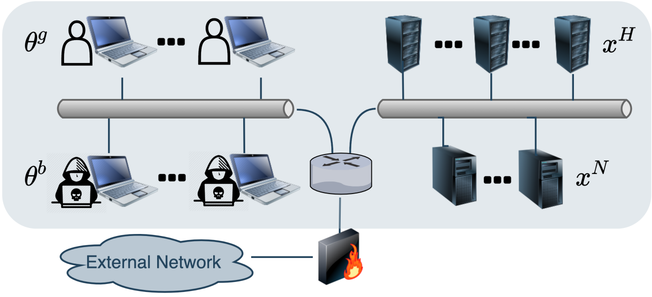


Figure 10.1: An example corporate network consists of normal servers and honeypots. The light blue background shows the region of the internal network.

violate security rules and abuse their privileges to save time and effort in finishing their tasks. They do not seek to sabotage the organization as the adversarial ones do. For each node in the corporate network, an insider can either access it (denoted by action a_{AC}) or not (denoted by action a_{DO}).

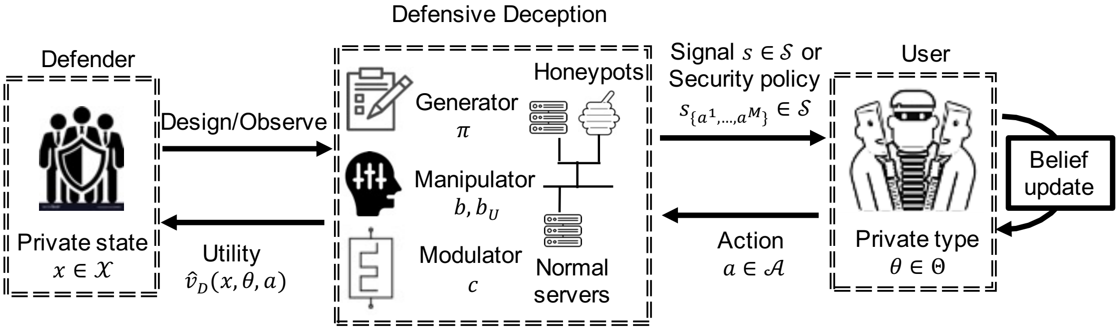


Figure 10.2: Timeline for the GMM mechanism design.

10.1.2 Game Elements

The DG consists of four elements; i.e., the *basic game* $(\mathcal{X}, \Theta, \mathcal{A}, v_D, v_U, b \in \Delta\mathcal{X})$, the *belief statistics* $(b_D(\cdot|x) \in \Delta\Theta, b_U(\cdot|\theta) \in \Delta\mathcal{X})$, the *information structure* $(\mathcal{S}, \pi \in \Pi)$, and the *utility transfer* $(\gamma, c \in \mathcal{C})$.

Basic Game

The DG consists of two players $i \in \{D, U\}$, a defender $i = D$ (hereafter she) and a user $i = U$ (hereafter he). Define the finite sets of N states, M types, and K actions as $\mathcal{X} := \{x_1, \cdots, x_N\}$, $\Theta := \{\theta_1, \cdots, \theta_M\}$, and $\mathcal{A} := \{a_{DO}, a_1, \cdots, a_{K-1}\}$, respectively. Action $a_{DO} \in \mathcal{A}$ is the drop-out action. It indicates that the user chooses not to participate in the game and takes no action.

The game has two-sided asymmetric information. The defender can privately observe or know the realization of the state $x \in \mathcal{X}$ from a probability distribution $b \in \Delta\mathcal{X}$. For example, in the corporate network in Fig. 10.1, $b(x^H)$ and $b(x^N)$ represent the percentages of honeypots and normal servers, respectively. The user does not know each node's state, i.e., whether a honeypot or a normal server. The user has a private type $\theta \in \Theta$ that represents his motive, capacity, rationality, or risk perception. The user's behaviors are abstracted as an action $a \in \mathcal{A}$. The defender can observe the user's action by monitoring and logging but she cannot observe the user's type; e.g., whether the user accesses the confidential data by accident (i.e., the unintentional type), out of self-interest (i.e., the selfish type), or for adversarial purposes (i.e., the adversarial type). The utility functions of the defender and the user, denoted by $v_i : \mathcal{X} \times \Theta \times \mathcal{A} \mapsto \mathbb{R}, i \in \{D, U\}$, depend on the state, type, and action.

Belief Statistics

The user's initial belief of the state under type $\theta \in \Theta$ is $b_U(\cdot|\theta) \in \Delta\mathcal{X}$. Since the user does not know the true state distribution $b(\cdot)$, his perceived state distribution b_U can be different from the true one. The defender's belief of the user's type at state $x \in \mathcal{X}$ is $b_D(\cdot|x) \in \Delta\Theta$. In the game, the defender can design b and b_U

through a virtual *trust manipulator*. For example, the defender can determine the percentage of honeypots to be $b(x^H)$ but report the percentage as $b_U(x^H|\theta)$ to the type-θ users who determine the percentage of honeypots based on the report without additional information. The trust manipulator is *overt* if the user's perceived state distribution equals the true one for all types, i.e., $b_U(x|\theta) = b(x), \forall x \in \mathcal{X}, \forall \theta \in \Theta$. Otherwise, the trust manipulator is said to be *covert* as the defender stealthily manipulates users' initial beliefs.

Information Structure

The *information structure* consists of a finite set of signals $\mathcal{S}$ and a generator $\pi \in \Pi : \mathcal{X} \mapsto \Delta\mathcal{S}$. With a slight abuse of notation, we use $\pi(s|x)$ to represent the probability of signal $s \in \mathcal{S}$ at state $x \in \mathcal{X}$. In Example 4 below, the signal can be interpreted as the *feature patterns*, including protocols, ports, the response time, and the error response.

Example 4 (**Dynamic Feature Pattern Configurations**). *To defend against honeypot fingerprinting, dynamic [195] and adaptive [88, 218] configurations have been adopted in honeypots. The SOC can also configure normal servers and disguise them as honeypots by generating honeypot-related features [183].*

Suppose that there are J features that both honeypots and normal servers can generate. Denote the value of feature $j \in \{1, \cdots, J\}$ by $e^j \in \mathcal{E}^j$, where $\mathcal{E}^j$ is a finite set. For example, the error response feature can take a binary value $e^j \in \mathcal{E}^j = \{0, 1\}$ based on whether an abnormal error message appears under intentionally erroneous requests [148]. We refer to the tuple of J features as the feature pattern denoted by $s = (e^1, \cdots, e^J) \in \mathcal{S} := \prod_{j=1}^{J} \mathcal{E}^j$. Then, the feature pattern of each node changes dynamically accordingly to the generator $\pi \in \Pi$; i.e., a honeypot and a normal server *generate feature pattern $s \in \mathcal{S}$ with frequency $\pi(s|x^H)$ and $\pi(s|x^N)$, respectively. Insiders can use these feature patterns as the digital footprint to fingerprint a node's state, either a honeypot or a normal server. The DG still applies to the case when the SOC cannot configure normal server. In that case, the decision variable $\pi(\cdot|x^N)$ will be taken as fixed.*

Utility Transfer

The *utility transfer* consists of a scaling factor $\gamma \in [0, \infty)$ and an incentive modulator $c \in \mathcal{C} : \mathcal{A} \mapsto \mathbb{R}$ which modifies the utilities of the defender and the user to be $\hat{v}_D(x, \theta, a) = v_D(x, \theta, a) + \gamma c(a)$ and $\hat{v}_U(x, \theta, a) = v_U(x, \theta, a) - c(a)$, respectively, for all $x \in \mathcal{X}, \theta \in \Theta, a \in \mathcal{A}$. Besides monetary (dis)incentives, $c(a)$ can also represent the additional cost or benefit of taking action $a \in \mathcal{A}$. For example, it captures the authentication time to access a normal server or a honeypot. The defender can determine the authentication time to incentivize the user (i.e., $c(a) < 0$) or disincentivize him (i.e., $c(a) > 0$) to take the action $a \in \mathcal{A}$. Although the modulator c is type-independent, its influence on users is type-dependent. For example, a curiosity-driven insider may lose interest and give up accessing confidential data under a long authentication delay or a convoluted *multi-factor authentication* process. However, an adversarial insider can be persistent if the data access leads to a comparably high financial return. Definition 29 defines a special utility structure where one action $a_k \in \mathcal{A}$ yields the highest benefit for the user of type $\theta \in \Theta$ regardless of the state values. For a user with a dominant action, a generator does not influence the user's belief and action.

Definition 29. *An action $a_k \in \mathcal{A}$ dominates (resp. is dominated) under type $\theta \in \Theta$ if $\hat{v}_U(x, \theta, a_k) \geq$ (resp. $\leq$)$\hat{v}_U(x, \theta, a), \forall a \in \mathcal{A}, \forall x \in \mathcal{X}$.*

10.1.3 Timeline for the GMM Mechanism Design

As shown in Fig. 10.2, the GMM mechanism design in DGs has two stages to achieve the intended outcomes of the defensive deception. At stage one, the defender designs (resp. observes) the generator $\pi \in \Pi$, the manipulator $b \in \Delta\mathcal{X}, b_U(\cdot|\theta) \in \Delta\mathcal{X}, \forall\theta \in \Theta$, and the modulator $c \in \mathcal{C}$ if these components can (resp. cannot) be designed. Based on the realized state value x, the generator generates a signal $s \in \mathcal{S}$ with probability $\pi(s|x)$. In the insider threat example, the defender configures the feature pattern s with probability $\pi(s|x^H)$ (resp. $\pi(s|x^N)$) when the node is a honeypot (resp. normal server). At stage two, the user of type $\theta \in \Theta$ receives the signal $s \in \mathcal{S}$ and obtains his posterior belief b_U^π of the state using the Bayesian rule, i.e.,

$$b_U^\pi(x|\theta, s) := \frac{b_U(x|\theta)\pi(s|x)}{\sum_{x' \in \mathcal{X}} b_U(x'|\theta)\pi(s|x')}, \forall x \in \mathcal{X}. \tag{10.1}$$

Then, the user of type $\theta \in \Theta$ takes a best-response action denoted by $a_\theta^*(b_U^\pi) \in \mathcal{A}$ to maximize his expected posterior utility under the posterior belief b_U^π, i.e.,

$$a_\theta^*(b_U^\pi) \in \arg\max_{a \in \mathcal{A}} \mathbb{E}_{x \sim b_U^\pi(\cdot|\theta,s)}[\hat{v}_U(x, \theta, a)]. \tag{10.2}$$

The utility of the users is a way to capture the user behavior a_θ^*. For example, a_θ^* can represent how an insider routinely follows the security rules or abuses his privilege for personal gain. The defender's goal is to determine the optimal GMM mechanism to proactively prevent undesirable user behaviors and improve the security posture. This objective is achieved by maximizing her *expected posterior utility* $\bar{v}_D$ that captures the outcomes of the user's behaviors, i.e., $\bar{v}_D(\pi, b, b_U, c) := \mathbb{E}_{x \sim b(\cdot)}\mathbb{E}_{s \sim \pi(\cdot|x)}\mathbb{E}_{\theta \sim b_D(\cdot|x)}[\hat{v}_D(x, \theta, a_\theta^*(b_U^\pi))]$. Different generators provide the user with different amounts of information about the state. Two extreme cases are defined in Definition 30. A signal from a zero-information generator denoted by $\pi^0 \in \Pi$ does not change the user's belief, i.e., $b_U^{\pi^0}(x|\theta, s) = b_U(x|\theta), \forall s \in \mathcal{S}, \forall x \in \mathcal{X}, \forall \theta \in \Theta$. Meanwhile, a signal from a full-information generator deterministically reveals the state to the user.

Definition 30 (**Zero- and Full-Information Generators**). *A generator $\pi \in \Pi$ contains zero information if $\pi(s|x) = \pi(s|x'), \forall s \in \mathcal{S}, \forall x, x' \in \mathcal{X}$. It contains full information if the mapping $\pi : \mathcal{X} \mapsto \mathcal{S}$ is injective.*

Readers can refer to Section 10.4 for a case study of insider threat that illustrates the two-stage GMM design.

10.1.4 Relation to Bayesian Persuasion

DG-GMM mechanism design is a generalized class of the Bayesian persuasion framework [109] with heterogeneous receivers, two-sided asymmetric information, and a joint design of information, incentive, and trust. If the user's type set Θ is a singleton and the defender cannot design the modulator and the manipulator, then DG-GMM degenerates to the Bayesian persuasion framework. The consolidation of the modulator and the manipulator into the mechanism gives the defender a higher degree of freedom to improve the performance in the deception design. It yet increases the computation complexity as illustrated in Section 10.2 and causes the violation of Bayesian plausibility in Section 10.1.4.

Violation of Bayesian Plausibility

The concept of Bayesian plausibility has been defined in [109], which states that the expected posterior belief should equal the prior belief for all $\pi \in \Pi$. However, we show in Lemma 10 that the trust manipulator can violate Bayesian plausibility when the user of type $\theta \in \Theta$ holds a different initial belief as the defender, i.e., $\exists x \in \mathcal{X} : b(x) \neq b_U(x|\theta)$.

Lemma 10 (**Bayesian Plausibility**)**.** *For all $\pi \in \Pi$ and $\theta \in \Theta$, the user's expected posterior probability $b^e_U(x|\theta) := \sum_{s\in\mathcal{S}} \sum_{x'\in\mathcal{X}} b(x')\pi(s|x')b^\pi_U(x|\theta, s)$ is always a valid probability measure yet is Bayesian plausible if and only if the defender and the user have the same initial belief $b(x) = b_U(x|\theta), \forall x \in \mathcal{X}$.*

Proof. A generator $\pi \in \Pi$ generates s with probability $\sum_{x'\in\mathcal{X}} b(x')\pi(s|x')$. After receiving s, the user of type θ obtains his posterior belief $b^\pi_U(x|\theta, s)$ according to (10.1). Thus, the expected posterior probability $\sum_{s\in\mathcal{S}} \sum_{x'\in\mathcal{X}} b(x')\pi(s|x')b^\pi_U(x|\theta, s)$ is a valid probability measure over x. The Bayesian plausibility requires $b^e_U(x|\theta) = \sum_{s\in\mathcal{S}} \frac{\sum_{x'\in\mathcal{X}} b(x')\pi(s|x')}{\sum_{x'\in\mathcal{X}} b_U(x'|\theta)\pi(s|x')}\pi(s|x)b_U(x|\theta) = b_U(x|\theta), \forall x \in \mathcal{X}$, under all $\pi \in \Pi$, which is equivalent to the condition $b(x) = b_U(x|\theta), \forall x \in \mathcal{X}$. □

10.2 GMM Designs by Mathematical Programming

In Section 10.2, we provide an integrated design of the GMM mechanism by mathematical programming. We first elaborate on the relationship between signals and the user's best-response action to introduce the notion of security policies. Each signal s from generator $\pi \in \Pi$ updates the user's belief via (10.1) and consequently induces the user of type $\theta \in \Theta$ to take the best-response action $a^*_\theta(b^\pi_U) \in \mathcal{A}$. Regardless of the signal set $\mathcal{S}$ and the generator π, these signals can elicit at most $|\mathcal{A}|^{|\Theta|} = K^M$ distinct outcomes; i.e., the user's best-response action $a^*_\theta(b^\pi_U)$ is a^l if his type is θ_l for all permutations of $\theta_l \in \Theta, a^l \in \mathcal{A}$. We can aggregate signals in $\mathcal{S}$ based on their elicited actions and divide the entire signal set $\mathcal{S}$ into K^M mutually exclusive subsets denoted as $\mathcal{S}_{\{a^1,a^2,\cdots,a^M\}}, a^l \in \mathcal{A}, l \in \{1, 2, ..., M\}$. Then, the signals in subset $\mathcal{S}_{\{a^1,a^2,\cdots,a^M\}}$ can be interpreted as the *security policy* that requires the user of type θ_l to take action a^l for all $l \in \{1, 2, \cdots, M\}$. Without loss of generality, we use one aggregated signal $s_{\{a^1,a^2,\cdots,a^M\}}$ to represent the signals in the set $\mathcal{S}_{\{a^1,a^2,\cdots,a^M\}}$. Then, the total number of signals are $|\mathcal{S}| = K^M$, and $\pi(\cdot|x) \in \Delta\mathcal{S}$ is a probability distribution over K^M security policies for each state $x \in \mathcal{X}$. The set Π naturally contains two feasibility constraints, i.e., $\pi(s_{\{a^1,\cdots,a^M\}}|x) \geq 0$, $\forall s_{\{a^1,\cdots,a^M\}} \in \mathcal{S}, \forall x \in \mathcal{X}$, and $\sum_{s_{\{a^1,\cdots,a^M\}} \in \mathcal{S}} \pi(s_{\{a^1,\cdots,a^M\}}|x) = 1, \forall x \in \mathcal{X}$. In Example 5 below, we continue to use the insider threat scenario in Section 10.1.1 to illustrate how we obtain security policies based on the feature patterns.

Example 5 (**Security Policies based on Feature Patterns**). *For binary action set $\mathcal{A} = \{a_{DO}, a_{AC}\}$ and binary type set $\Theta = \{\theta^g, \theta^b\}$, the feature patterns in Example 4 can be aggregated into $K^M = 4$ categories of security policies. They are $s_{\{a_{DO},a_{DO}\}}$ (i.e., both types of insiders choose a_{DO}), $s_{\{a_{DO},a_{AC}\}}$ (i.e., selfish insiders choose a_{AC} while adversarial insiders choose a_{DO}), $s_{\{a_{AC},a_{DO}\}}$ (i.e., adversarial insiders choose a_{AC} while selfish insiders choose a_{DO}), and $s_{\{a_{AC},a_{AC}\}}$ (i.e., both types of insiders choose a_{AC}).*

We can rewrite (10.2) as $\sum_{x \in \mathcal{X}} b^{\pi}_U(x|\theta_l, s_{\{a^1,\cdots,a^M\}})[\hat{v}_U(x, \theta_l, a^l) - \hat{v}_U(x, \theta_l, a^h)] \geq 0, \forall s_{\{a^1,\cdots,a^M\}} \in \mathcal{S}, \forall a^h \in \mathcal{A}, \forall \theta_l \in \Theta$, concerning security policies. The defender's expected posterior utility $\bar{v}_D(\pi, b, b_U, c)$ can be equivalently represented as $\sum_{x \in \mathcal{X}} b(x)$ $\sum_{s_{\{a^1,\cdots,a^M\}} \in \mathcal{S}} \pi(s_{\{a^1,\cdots,a^M\}}|x) \sum_{\theta_l \in \Theta} b_D(\theta_l|x)\hat{v}_D(x, \theta_l, a^l)$. Replacing b^{π}_U with (10.1), we formulate the GMM mechanism design as the following constrained optimization

COP.

$$\begin{aligned}
&\text{(COP):} \quad r := \sup_{\pi\in\Pi, b, b_U, c\in\mathcal{C}} \quad \bar{v}_D(\pi, b, b_U, c)\\
&\text{(IC)} \sum_{x\in\mathcal{X}} [\hat{v}_U(x,\theta_l,a^l) - \hat{v}_U(x,\theta_l,a^h)]\pi(s_{\{a^1,\cdots,a^M\}}|x)\\
&\qquad b_U(x|\theta_l) \geq 0, \forall s_{\{a^1,\cdots,a^M\}} \in \mathcal{S}, \forall a^h \in \mathcal{A}, \forall \theta_l \in \Theta.\\
&\text{(MF) } c(a_{DO}) = 0.
\end{aligned}$$

The decision variables π, b, b_U, and c are vectors of dimension $N \times K^M$, N, $N \times M$, and K, respectively. The feasibility constraint contained in Π and the Incentive-Compatible (IC) constraint induce $N \times K^M + 1$ and $K^M \times K \times M$ constraints, respectively.

Denote b^*, b_U^*, π^*, c^* as the maximizers of COP and r as the value of the objective function under the maximizers. The (IC) constraint requires all security policies from the generator to be compatible with the user's incentives; i.e., the user receives the maximum benefit on average when taking the action required by the security policy. A security policy cannot be generated if it is not incentive-compatible. Based on the (IC) constraint, we define the credible and the optimal generators in Definition 31 and enforceable security policies in Definition 32.

Definition 31 (**Credible and Optimal Generators**). *A generator $\pi \in \Pi$ is called credible if it satisfies (IC). A credible generator is called optimal if it maximizes COP.*

Definition 32 (**Enforceable Security Policies**). *For a given generator $\pi \in \Pi$, a security policy $s_{\{a^1,\cdots,a^M\}} \in \mathcal{S}$ is enforceable (resp. unenforceable) if $\exists x \in \mathcal{X}$ such that $\pi(s_{\{a^1,\cdots,a^M\}}|x) \neq 0$ (resp. $\pi(s_{\{a^1,\cdots,a^M\}}|x) = 0, \forall x \in \mathcal{X}$).*

The Modulation-Feasible (MF) constraint results from the fact that the defender cannot modulate the user's incentive if the user does not participate in the game. Although the co-domain of c is $\mathbb{R}$, Theorem 6 shows that the optimal utility transfer $c^* \in \mathcal{C}$ has to remain bounded due to the user's potential threat of taking the drop-out action a_{DO}. We define the following shorthand notations for Theorem 6, i.e., $\underline{c}(\theta, a) := \max_{x\in\mathcal{X}} v_U(x, \theta, a) - v_U(x, \theta, a_{DO})$, $\bar{r} = \max_{x\in\mathcal{X}} \mathbb{E}_{\theta\sim b_D}[\max_{a\in\mathcal{A}} v_D(\theta, x, a)]$ and $\underline{r} = \min_{x\in\mathcal{X}} \mathbb{E}_{\theta\sim b_D}[\min_{a\in\mathcal{A}} v_D(\theta, x, a)]$.

Theorem 6 (**Feasibility and Design Capacity**). *COP is feasible and bounded. The upper bound of r is $\max\{\max_{x\in\mathcal{X}} \mathbb{E}_{\theta\sim b_D}[v_D(x, \theta, a_{DO})], \bar{r} + \gamma \max_{a\in\mathcal{A},\theta\in\Theta} \underline{c}(\theta, a)\}$ and the lower bound is $\underline{r}$.*

Proof. We first prove the feasibility. Define shorthand notation

$$a^{*,l} := arg \max_{a\in\mathcal{A}} \mathbb{E}_{x\sim b_U(x|\theta_l)}[v_U(x, \theta_l, a) - c(a)], \forall l \in \{1, \cdots, M\}$$

as the optimal action of the user of type $\theta_l \in \Theta$ under any feasible prior belief $b_U(x|\theta_l)$ and modulator $c \in \mathcal{C}$. Then, the zero-information generator, denoted as $\pi^0(s_{(a^{*,1},\cdots,a^{*,M})}|x) = 1, \forall x \in \mathcal{X}$, is a feasible solution to COP.

We prove the boundedness in two steps. We first consider $c(a) = 0, \forall a \in \mathcal{A}$. Since all decision variables b, π, b_D are probability measures, we obtain the upper bound $\bar{r}$ and the low bound $\underline{r}$ of r. In the second step, we turn the modulator c into a free decision variable with the (MF) constraint. Since $c(a) = 0, \forall a \in \mathcal{A}$, is a feasible solution, the maximum value of COP does not increase. Thus, the value of $\underline{r}$ is bounded. To show that the value of $\bar{r}$ is bounded in step two, we focus on action $a_j \in \mathcal{A}$, if it exists, that results in a non-negative maximizer $c^*(a_j)$. On the one hand, if $\underline{c}(\theta, a_j) \leq 0, \forall\theta \in \Theta$, then the drop-out action a_{DO} dominates for all types and $r = \max_{b \in \Delta\mathcal{X}} \mathbb{E}_{x \sim b} \mathbb{E}_{\theta \sim b_D}[v_D(x, \theta, a_{DO})] \leq \max_{x \in \mathcal{X}} \mathbb{E}_{\theta \sim b_D}[v_D(x, \theta, a_{DO})]$. On the other hand, if there exists a type $\theta \in \Theta$ where $\underline{c}(\theta, a_j) > 0$ and $c^*(a_j) \geq \underline{c}(\theta, a_j)$, then the user of type θ will choose the drop-out action a_{DO}. Thus, $r \leq \gamma \max_{a \in \mathcal{A}, \theta \in \Theta} \underline{c}(\theta, a)$. □

The upper and lower bounds provide the design capacity of the GMM mechanism. COP is unbounded without the (MF) constraint as the defender can arbitrarily increase (resp. decrease) the value of r by letting $c(a)$ be an arbitrarily large (resp. small) constant. If $c(a) = 0, \forall a \in \mathcal{A}$, we can transform COP into a Linear Program (LP) by introducing the following variables, i.e., $\eta(s_{\{a^1, \cdots, a^M\}}, x) := b(x)\pi(s_{\{a^1, \cdots, a^M\}}|x)$ and $\eta_U(\theta, s_{\{a^1, \cdots, a^M\}}, x) := b_U(x|\theta)\pi(s_{\{a^1, \cdots, a^M\}}|x)$.

These new variables take non-negative values and satisfy the following constraints, i.e., $\sum_{x \in \mathcal{X}, s_{\{a^1, \cdots, a^M\}} \in \mathcal{S}} \eta = 1$ and $\sum_{x \in \mathcal{X}, s_{\{a^1, \cdots, a^M\}} \in \mathcal{S}} \eta_U = 1, \forall \theta \in \Theta$. After we have solved the LP, we obtain $b(x) = \sum_{s_{\{a^1, \cdots, a^M\}} \in \mathcal{S}} \eta(s_{\{a^1, \cdots, a^M\}}, x)$ and $b_U(x|\theta) = \sum_{s_{\{a^1, \cdots, a^M\}} \in \mathcal{S}} \eta_U(\theta, s_{\{a^1, \cdots, a^M\}}, x)$ for all $x \in \mathcal{X}, \theta \in \Theta$.

10.3 Graphical Analysis of GMM Designs

In Section 10.2, we aggregate signals into K^M equivalent security policies to relate them with the user's best-response action. In Section 10.3, we directly analyze the posterior belief and the action as each signal uniquely determines a posterior belief. Throughout Section 10.3, we focus on the *overt* trust manipulator defined in Section 10.1.2, i.e., $b_U(x|\theta) = b(x), \forall x \in \mathcal{X}, \theta \in \Theta$. Define $p_j^0 := b(x_j), \forall j \in \{1, \cdots, N\}$, and the common prior belief in the vector form as $\mathbf{p}^0 := [p_1^0, \cdots, p_N^0]$. Since different types of users have the same initial beliefs, the posterior beliefs are also the same. Denote $p_j \in [0, 1]$ as the user's posterior belief under state

$x_j \in \mathcal{X}, \forall j \in \{1, \cdots, N\}$. Define the belief vector $\mathbf{p} := [p_1, \cdots, p_N]$ and the utility vector $\hat{\mathbf{v}}_U(\theta, a) := [\hat{v}_U(x_1, \theta, a), \cdots, \hat{v}_U(x_N, \theta, a)]'$ where notation $'$ denotes the matrix transpose. For both the prior and the posterior belief vectors, the total probability is one, i.e., $\sum_{n=1}^{N} p_n^0 = 1$ and $\sum_{n=1}^{N} p_n = 1$.

Section 10.3.1 provides the optimal generator design under the benchmark case where the defender can neither modify the user's incentive, i.e., $c(a) = 0, \forall a \in \mathcal{A}$, nor manipulate their initial beliefs. Section 10.3.2 incorporates the modulator and the manipulator into the GMM mechanism design.

10.3.1 Generator Design under the Benchmark Case

We rewrite (10.2) in its matrix form as $a_\theta^*(\mathbf{p}) \in arg \max_{a \in \mathcal{A}} \mathbf{p}\hat{\mathbf{v}}_U(\theta, a)$. Since $\mathbf{p}\hat{\mathbf{v}}_U(\theta, a)$ is an affine function of $\mathbf{p}$ for any action $a \in \mathcal{A}$, maximizing $\mathbf{p}\hat{\mathbf{v}}_U(\theta, a)$ over a in the convex domain $\mathbf{p} \in \Delta\mathcal{X}$ results in a Piece-Wise Linear and Convex (PWLC) function as summarized in Proposition 12. The proof of convexity follows directly from the fact that $a_\theta^*(\mathbf{p})$ is the point-wise maximum of a group of affine functions over $\mathbf{p}$.

Proposition 12. *The user's expected posterior utility under a give type $\theta \in \Theta$, i.e., $\max_{a \in \mathcal{A}} \mathbf{p}\hat{\mathbf{v}}_U(\theta, a)$, is continuously PWLC with respect to vector $\mathbf{p} \in \Delta\mathcal{X}$.*

We visualize $\max_{a \in \mathcal{A}} \mathbf{p}\hat{\mathbf{v}}_U(\theta, a)$ under a binary state set in Fig. 10.3. When $N = 2$, we can use the first element p_1 as the x-axis to uniquely represent the posterior belief $\mathbf{p} \in \Delta\mathcal{X}$. The four belief thresholds, i.e., $0, t_1^\theta, t_2^\theta$, and 1, divide the entire belief region of $p_1 \in [0, 1]$ into three sub-regions. The user of type θ takes action a_{K-1} if his posterior belief belongs to the sub-region $p_1 \in [0, t_1^\theta]$, action a_1 if $p_1 \in [t_1^\theta, t_2^\theta]$, and action a_{DO} if $p_1 \in [t_2^\theta, 1]$. Although action a_2 is not dominated under type θ based on Definition 29, it is inactive over $p_1 \in [0, 1]$.

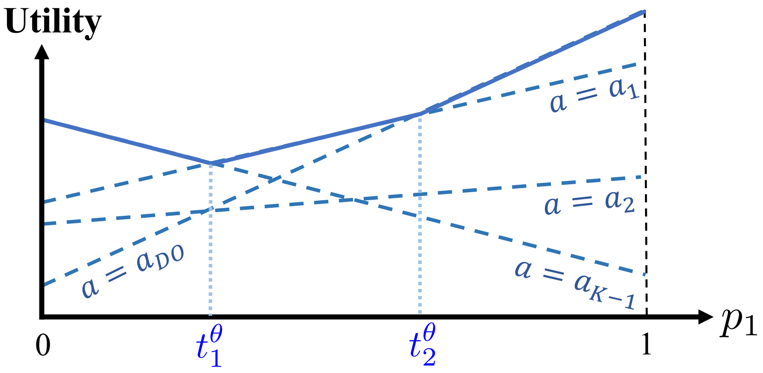


Figure 10.3: The expected posterior utility of the user of type $\theta \in \Theta$ versus posterior belief $p_1 \in [0, 1]$. The solid lines represent the utility $\max_{a\in\mathcal{A}} \sum_{n=1}^{N} p_n \hat{v}_U(x_n, \theta, a)$ as a PWLC function of p_1.

For a high-dimensional state space $N \geq 3$, the user's entire belief region $\Delta\mathcal{X}$ is an $N - 2$ simplex. For each type θ, we can divide the entire belief region into at most K sub-regions $\mathcal{C}_{a_i}^{\theta} := \{\mathbf{p} \geq \mathbf{0}|\mathbf{p}'[\hat{\mathbf{v}}_U(\theta, a_i) - \hat{\mathbf{v}}_U(\theta, a_j)] \geq 0, \forall a_j \in \mathcal{A}$. Then, $\Delta\mathcal{X} = \cup_{i\in\{DO,1,\cdots,K-1\}}\mathcal{C}_{a_i}^{\theta}$. If the posterior belief falls into the sub-region $\mathcal{C}_{a_i}^{\theta}$, the user of type θ takes a_i as his best-response action. Take Fig. 10.3 as an example, $\mathcal{C}_{a_{DO}}^{\theta}$ is the interval $[t_2^\theta, 1]$ and $\mathcal{C}_{a_2}^{\theta}$ is the empty set. As a direct result of the definition of convexity, sets $\mathcal{C}_{a_i}^{\theta}, \forall i \in \{DO, 1, \cdots, K - 1\}$, are convex and connected.

We have illustrated the belief region partition under any given type $\theta \in \Theta$. Since the user has M possible types, we further divide the belief region into finer sub-regions. Let $\mathcal{C}_{\{a^1,\cdots,a^M\}} := \mathcal{C}_{a^1}^{\theta_1} \cap \cdots \cap \mathcal{C}_{a^M}^{\theta_M}$ be the sub-region of the posterior belief under which the best-response action of the user of type $\theta_l, \forall l \in \{1, \cdots, M\}$, is action $a^l \in \mathcal{A}$. In particular, define $\mathcal{C}_{i,j}^{l,h} := \mathcal{C}_{a_i}^{\theta_l} \cap \mathcal{C}_{a_j}^{\theta_h}$ as the belief region where the user takes action a_i when his type is θ_l and a_j when his type is θ_h for all $i, j \in \{DO, 1, \cdots, K - 1\}$ and $l \neq h, \forall l, h \in \{1, \cdots, M\}$. Based on the definition, $\mathcal{C}_{i,j}^{l,h} \equiv \mathcal{C}_{j,i}^{h,l}$. Since the intersection of any collection of convex sets is convex,

$\mathcal{C}_{\{a^1,\cdots,a^M\}}$ and $\mathcal{C}_{i,j}^{l,h}$ are all convex and connected sets, i.e., convex polytopes. We visualize these convex polytopes in Fig. 10.4 when there are two types $M = 2$, two actions $K = 2$, and three states $N = 3$. The belief region $\Delta\mathcal{X}$ is an $N - 2$ simplex, i.e., an equilateral triangle. Under type θ_1, the belief region is divided into $\mathcal{C}_{a_{DO}}^{\theta_1} = \mathcal{C}_{\{a_{DO},a_1\}} \cup \mathcal{C}_{\{a_{DO},a_{DO}\}}$ and $\mathcal{C}_{a_1}^{\theta_1} = \mathcal{C}_{\{a_1,a_1\}} \cup \mathcal{C}_{\{a_1,a_{DO}\}}$. Under type θ_2, the belief region is divided into $\mathcal{C}_{a_{DO}}^{\theta_2} = \mathcal{C}_{\{a_{DO},a_{DO}\}} \cup \mathcal{C}_{\{a_1,a_{DO}\}}$ and $\mathcal{C}_{a_1}^{\theta_2} = \mathcal{C}_{\{a_1,a_1\}} \cup \mathcal{C}_{\{a_{DO},a_1\}}$. Since there are only two types, we have $\mathcal{C}_{1,DO}^{1,2} = \mathcal{C}_{\{a_1,a_{DO}\}}$.

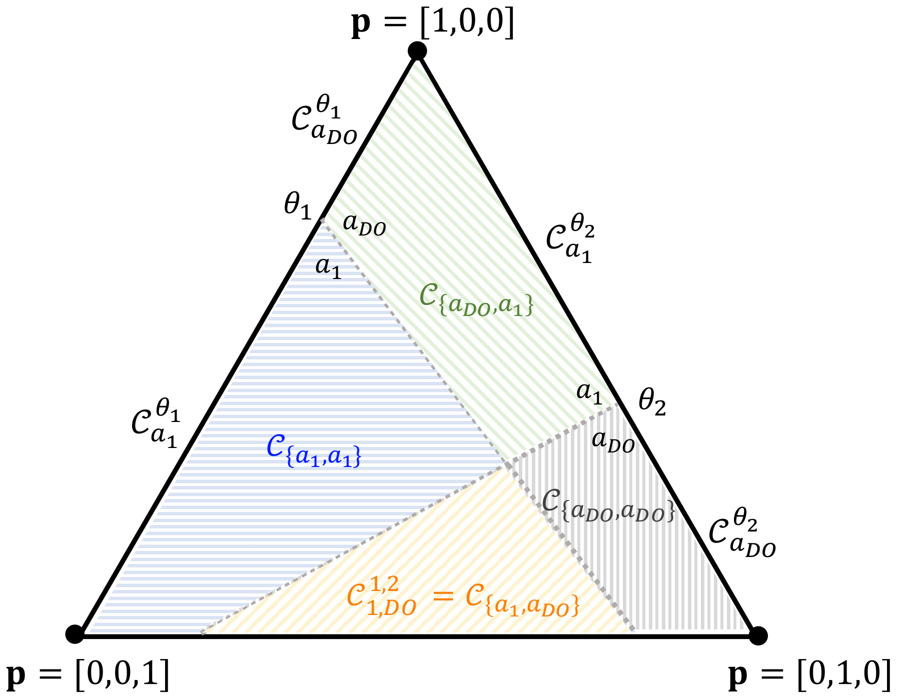

$\mathbf{p} = [1,0,0]$
$\mathcal{C}_{a_{DO}}^{\theta_1}$
θ_1
a_{DO}
a_1
$\mathcal{C}_{a_1}^{\theta_2}$
$\mathcal{C}_{\{a_{DO},a_1\}}$
a_1
θ_2
a_{DO}
$\mathcal{C}_{a_1}^{\theta_1}$
$\mathcal{C}_{\{a_1,a_1\}}$
$\mathcal{C}_{\{a_{DO},a_{DO}\}}$
$\mathcal{C}_{a_{DO}}^{\theta_2}$
$\mathcal{C}_{1,DO}^{1,2} = \mathcal{C}_{\{a_1,a_{DO}\}}$
$\mathbf{p} = [0,0,1]$
$\mathbf{p} = [0,1,0]$

Figure 10.4: Illustration of $K^M = 4$ convex polytopes $\mathcal{C}_{\{a_1,a_1\}}$, $\mathcal{C}_{\{a_{DO},a_1\}}$, $\mathcal{C}_{\{a_{DO},a_{DO}\}}$, and $\mathcal{C}_{\{a_1,a_{DO}\}}$ in blue (horizontal stripes), green (downward diagonal stripes), grey (vertical stripes), and orange (upward diagonal stripes), respectively. Each point in the equilateral triangle represents a belief $\mathbf{p} = [p_1, p_2, p_3] \in \Delta\mathcal{X}$.

Among K^M possible sets $\mathcal{C}_{\{a^1,\cdots,a^M\}}, \forall a^l \in \mathcal{A}, l \in \{1, \cdots, M\}$, most of them are empty. Take $N = 2$ as an example, K actions can generate at most $K(K-1)/2$ belief thresholds over $p_1 \in (0, 1)$ for each type as shown in Fig. 10.3. Thus, the whole belief region $p_1 \in [0,1]$ can be divided into at most $MK(K-1)/2+1$ regions under M types. When $N=3$, the belief region is an equilateral triangle as shown in Fig. 10.4. For each given type, K actions represent K planes. Projecting these planes vertically onto the equilateral triangle, we obtain at most $K(K-1)/2$ lines. Thus, these lines under M types can divide the equilateral triangle into at most $\frac{MK(K-1)}{2}(\frac{MK(K-1)}{2}+1)/2$ belief regions. The results can be extended to $N>3$ as a variant of the hyperplane arrangement problem [162]. We summarize the above result in Proposition 13; i.e., the number of belief region partitions grows in a polynomial rate denoted by $\chi(K,M,N)$ rather than the exponential rate of K^M, where $\chi(K,M,N)$ is a polynomial function of K, M for each N.

Proposition 13 (**Upper Limit of Enforceable Policies**). *For any credible generator, at most $\chi(K,M,N)$ security policies are enforceable.*

Remark 25. *Solely dependent on the user's utility vector $\hat{\mathbf{v}}_U$, the belief partition $\Delta\mathcal{X} = \cup_{a^1\in\mathcal{A},\cdots,a^M\in\mathcal{A}}\mathcal{C}_{\{a^1,\cdots,a^M\}}$ characterizes the user's incentive under different types. If $\mathcal{C}_{\{a^1,\cdots,a^M\}} = \emptyset$, then the security policies that require the user of type θ_l to take action a^l for any $l \in \{1,\cdots,M\}$ are unenforceable as they violate the user's incentive. Proposition 13 illustrates that the number of enforceable security policies cannot exceed a threshold determined by K, M, N; i.e., among all $|\mathcal{S}| = K^M$ potential security policies, the defender can choose at most $\chi(K,M,N)$ ones to be compatible with the user's incentive.*

Cyber Attribution and Type Identification

The honeypot example motivates us to investigate the condition under which public security policies elicit different actions from different types of users. The condition is useful for cyber attribution, i.e., tracing observable actions back to the user's private types. Since each security policy uniquely determines a posterior belief for a given generator, we define type identifiability concerning the posterior belief in Definition 33.

Definition 33 (**Identifiable Types**)**.** *Two different types* $l, h \in \{1, \cdots, M\}$ *are identifiable under a posterior belief* $\mathbf{p} \in \Delta\mathcal{X}$ *if* $\exists i, j \in \{DO, 1, \cdots, K-1\}$ *and* $i \neq j$ *such that* $\mathbf{p} \in \mathcal{C}_{i,j}^{l,h}$*.*

The posterior beliefs under which two different types $l, h \in \{1, \cdots, M\}$ are identifiable constitute a belief region that may not be connected. This belief region solely depends on the user's utility vector $\hat{\mathbf{v}}_U$ as the finest belief partition $\Delta\mathcal{X} = \cup_{a^1 \in \mathcal{A}, \cdots, a^M \in \mathcal{A}} \mathcal{C}_{\{a^1, \cdots, a^M\}}$ solely depends on $\hat{\mathbf{v}}_U$. Intuitively, the size of the region is reduced as the utilities of the users of type θ_l and θ_h become better aligned. Definition 34 defines two extremes of utility alignment.

Definition 34 (**Completely (Mis)aligned Utilities**)**.** *Two different types of users have completely aligned (resp. misaligned) utilities, or equivalently zero (resp. full) utility misalignment, if they are unidentifiable (resp. identifiable) under all posterior belief* $\mathbf{p} \in \Delta\mathcal{X}$*.*

If two utilities have the same (resp. opposite) values, then they are completely aligned (resp. misaligned). If two types of users' utilities are completely aligned (resp. misaligned), then the security policies that procure them to take different actions (resp. the same action) are not enforceable under any credible generators. Proposition 14 shows that the results are translation- and scale-invariant.

Proposition 14 (**Alignment under Linear Dependence**)**.** *Consider linearly dependent utilities of two types* $l, h \in \{1, \cdots, M\}$ *of users; i.e., there exist a scaling* *factor* $\rho_U^s(\theta_l, \theta_h) \in \mathbb{R}$ *and translation factors* $\rho_U^t(x, \theta_l, \theta_h) \in \mathbb{R}, \forall x \in \mathcal{X}$*, such that* $\hat{v}_U(x, \theta_l, a) = \rho_U^s(\theta_l, \theta_h)\hat{v}_U(x, \theta_h, a) + \rho_U^t(x, \theta_l, \theta_h), \forall x \in \mathcal{X}, a \in \mathcal{A}$*. Two utilities are completely aligned (resp. misaligned) if and only if* $\rho_U^s(\theta_l, \theta_h) \geq 0$ *(resp.* < 0*).*

Proof. For any given $\mathbf{p} \in \Delta\mathcal{X}$ and $\theta_l \in \Theta$, there exists an action $a_i^* \in \mathcal{A}$ such that $\sum_{n=1}^N p_n[\hat{v}_U(x_n, \theta_l, a_i^*) - \hat{v}_U(x_n, \theta_l, a_k)] \geq 0, \forall a_k \in \mathcal{A}$.

Then, $\rho_U^s(\theta_l, \theta_h) \sum_{n=1}^N p_n[\hat{v}_U(x_n, \theta_h, a_i^*) - \hat{v}_U(x_n, \theta_h, a_k)] \geq 0, \forall a_k \in \mathcal{A}$, and the user of type $\theta_h \in \Theta$ at any posterior belief $\mathbf{p}$ has the same best-response action a_i^* if and only if $\rho_U^s(\theta_l, \theta_h) \geq 0$. □

Characterization of the Optimal Generator

Under a zero-information generator $\pi^0 \in \Pi$, the user's posterior belief equals the prior belief $\mathbf{p}^0$ and we can rewrite the user' best-response action $a_\theta^*(b_U^{\pi^0})$ in (10.2) as $a_\theta^*(\mathbf{p}^0)$. Since variables b_U, c are not designable in the benchmark case, we omit them in function $\bar{v}_D$ and rewrite the defender's expected posterior utility as $\bar{v}_D(\pi, \mathbf{p}^0)$. Since the users make decisions based on their prior beliefs, we refer to the expected posterior utility $\bar{v}_i$ of player $i \in \{D, U\}$ as his *prior utility* $\tilde{v}_i$ when the generator contains zero information. In particular, the defender's prior utility $\tilde{v}_D$ is a function of the prior belief $\mathbf{p}^0$, i.e.,

$$\tilde{v}_D(\mathbf{p}^0) := \bar{v}_D(\pi^0, \mathbf{p}^0) = \mathbb{E}_{x\sim\mathbf{p}^0}\mathbb{E}_{\theta\sim b_D(\cdot|x)}[\hat{v}_D(x, \theta, a_\theta^*(\mathbf{p}^0)].$$

We obtain the piece-wise linear structure of the defender's prior utility $\tilde{v}_D$ in Proposition 15. The solid lines in Fig. 10.5 illustrate $\tilde{v}_D$.

Proposition 15. *The defender's prior utility* $\tilde{v}_D$ *is a (possibly discontinuous) piece-wise linear function of the common prior belief vector* $\mathbf{p}^0 \in \Delta\mathcal{X}$ *with at most*

$\chi(K, M, N)$ *pieces.*

Proof. The piece-wise linear structure follows from the fact that $\tilde{v}_D$ is linear with respect to $\mathbf{p}^0$ inside each convex polytope $\mathcal{C}_{\{a^1,\cdots,a^M\}}, \forall a^l \in \mathcal{A}, l \in \{1, \cdots, M\}$. As a result of Proposition 13, the upper bound of the number of different convex polytopes is $\chi(K, M, N)$. Since the polytopes are determined based on the user's prior utility rather than the defender's, $\tilde{v}_D$ is possibly discontinuous at the boundaries of these polytopes. □

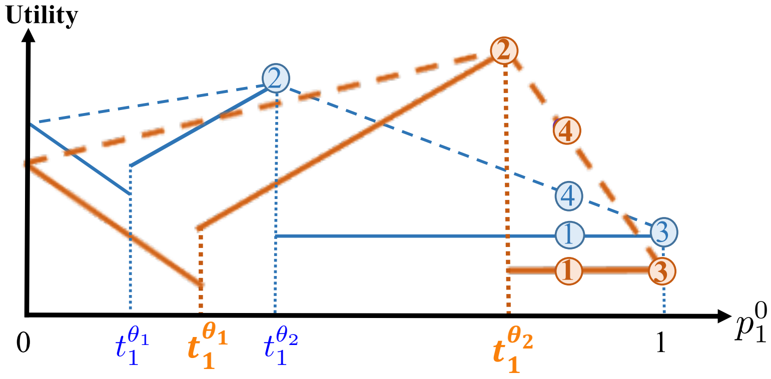


Figure 10.5: The defender's expected posterior utility versus prior belief p_1^0 with and without the modulator in orange and blue, respectively. We denote orange lines and notations in bold. The solid lines indicate that the defender's prior utility $\tilde{v}_D$ is discontinuous and piece-wise linear under three belief regions, i.e., $[0, t_1^{\theta_1}]$, $[t_1^{\theta_1}, t_1^{\theta_2}]$, and $[t_1^{\theta_2}, 1]$. The dashed lines represent the defender's optimal posterior utility V_D.

The defender's expected posterior utility $\bar{v}_D$ is a function of $\pi \in \Pi$ and $\mathbf{p}^0 \in \Delta\mathcal{X}$. Thus, the defender's optimal posterior utility $V_D(\mathbf{p}^0) := \sup_{\pi\in\Pi} \bar{v}_D(\pi, \mathbf{p}^0)$ is a function of $\mathbf{p}^0 \in \Delta\mathcal{X}$. Based on Theorem 6, there exists an optimal generator $\pi^* \in \Pi$ that achieves the optimal posterior utility, i.e., $V_D(\mathbf{p}^0) = \bar{v}_D(\pi^*, \mathbf{p}^0) = r$. Denote the convex hull of function $\tilde{v}_D$ as $co(\tilde{v}_D)$. Then, we can use the concavification technique introduced in [10, 109] to show that the defender's optimal posterior utility $V_D(\mathbf{p}^0)$ is the concave closure of her prior utility $\tilde{v}_D(\mathbf{p}^0)$ over the entire belief region $\mathbf{p}^0 \in \Delta\mathcal{X}$, i.e., $V_D(\mathbf{p}^0) = \sup\{z \in \mathbb{R} | (\mathbf{p}^0, z) \in co(\tilde{v}_D)\}$.

We visualize the concavification process under the binary state space $N = 2$ in Fig. 10.5. Suppose that there are two types of users and each type $\theta \in \{\theta_1, \theta_2\}$ has a single belief threshold denoted by t_1^{θ} where $0 < t_1^{\theta_1} < t_1^{\theta_2} < 1$. Consider a common prior belief $p_1^0 \in [t_1^{\theta_2}, 1]$ denoted by node 1's abscissa. Then, the defender's prior utility $\tilde{v}_D(p_1^0)$ is denoted by node 1's ordinate. The defender can improve the utility from node 1's ordinate to at most node 4's ordinate by adopting the optimal generator $\pi^* \in \Pi$ as follows. Generator π^* generates two signals $s_2 \in \mathcal{S}$ and $s_3 \in \mathcal{S}$ with proper probabilities under different states so that the user's posterior belief is node 2's abscissa when observing policy s_2 and node 3's abscissa when observing s_3. Based on the Bayesian plausibility condition in Section 10.1.4, the defender's optimal posterior utility $V_D(p_1^0)$ can be represented as the linear interpolation of the ordinates of nodes 2 and 3, i.e., node 4's ordinate. The same reasoning applies to all feasible common prior beliefs $p_1^0 \in [0, 1]$. Therefore, for all $[p_1^0, 1 - p_1^0] \in \Delta\mathcal{X}$, the defender's optimal posterior utility $V_D(\mathbf{p}^0)$ is the concave closure of her prior utility $\tilde{v}_D(\mathbf{p}^0)$ and $V_D(\mathbf{p}^0) \geq \tilde{v}_D(\mathbf{p}^0)$.

Although we need at least $|\mathcal{S}| = K^M$ security policies to represent all the permutations of actions under different types, Fig. 10.5 shows that the defender can achieve her optimal posterior utility by generating two different security policies with proper probabilities when $N = 2$. Proposition 16 generalizes the result to $N > 2$ and shows that the generator only needs to generate a small number of security policies to achieve her optimal posterior utility. If $\tilde{v}_D(\mathbf{p}^0) = V_D(\mathbf{p}^0)$ and $\mathbf{p}^0$ is further an interior point of any convex polytope $\mathcal{C}_{\{a^1,\cdots,a^M\}}, \forall a^l \in \mathcal{A}, l \in \{1, \cdots, M\}$, then there exist infinitely many credible generators that achieve $V_D(\mathbf{p}^0)$.

Proposition 16 (**Efficiency of the Optimal Generator**)**.** *For any DG with common prior belief $\mathbf{p}^0 \in \Delta\mathcal{X}$, there exist either one or infinitely many optimal generators to achieve the optimal posterior utility $V_D(\mathbf{p}^0)$. For each state $x \in \mathcal{X}$, there exists one optimal generator $\pi^*(\cdot|x) \in \Delta\mathcal{S}$ that generates at least $K^M - N$ security policies with zero probability.*

Proof. Since COP under the benchmark case is a linear program, the optimal solution is either unique or innumerable. If $N = 2$, the convex hull consists of pieces of line segments where each line segment can be determined uniquely by its two endpoints. If $N = 3$, the convex hull as a polygon consists of finite pieces of triangles where each triangle can be determined uniquely by its three endpoints. We can extend to any finite N where the convex hull consists of pieces of $(N-1)$-simplex where each piece can be determined uniquely by N endpoints. Thus, for any $\mathbf{p}^0 \in \Delta\mathcal{X}$, it requires at most N points to achieve $V_D(\mathbf{p}^0)$, which corresponds to N distinct security policies. □

Remark 26. *Proposition 16 shows that the defender does not need to apply all enforceable security policies to achieve the optimal posterior utility; i.e., the optimal generator is efficient and generates at most N security policies for each state $x \in \mathcal{X}$.*

We define the trust margin under a credible generator $\pi \in \Pi$ in Definition 35. The maximum trust margin is achieved when the optimal generator $\pi^* \in \Pi$ is applied. The trust margin can be negative if generator π is not well designed. However, the maximum trust margin is non-negative as it is the difference between the defender's optimal posterior utility and prior utilities, i.e., $V_D(\mathbf{p}^0) - \tilde{v}_D(\mathbf{p}^0)$. Based on whether the maximum trust margin is zero or positive, Definition 36 defines the user to be unmanageable or manageable.

Definition 35 (**Trust Margin**). *We define $\bar{v}_D(\pi, \mathbf{p}^0) - \tilde{v}_D(\mathbf{p}^0)$ as the trust margin under the common prior belief $\mathbf{p}^0 \in \Delta\mathcal{X}$ and a credible generator $\pi \in \Pi$.*

Definition 36 (**Manageability**). *The user is manageable (resp. unmanageable) under prior belief $\mathbf{p}^0$ if the maximum trust margin is greater than (resp. equals) zero.*

Intuitively, a user is manageable if he shares the same utility with the defender but unmanageable if he has an opposite utility. We introduce $\rho^s_D \in \mathbb{R}$ to represent the user's level of maliciousness. Theorem 7 investigates how the user's level of maliciousness affects his manageability.

Theorem 7 (**Manageability and Level of maliciousness**). *Let the common prior belief be state-independent, i.e., $b_D(\theta|x) = \hat{b}_D(\theta), \forall \theta \in \Theta, \forall x \in \mathcal{X}$, and two players' utilities be linearly dependent, i.e., there exist a scaling factor $\rho^s_D \in \mathbb{R}$ and translation factors $\rho^t_D(x, \theta) \in \mathbb{R}$, such that $\hat{v}_D(x, \theta, a) = \rho^s_D \hat{v}_U(x, \theta, a) + \rho^t_D(x, \theta), \forall x \in \mathcal{X}, \theta \in \Theta, a \in \mathcal{A}$. Then, the following two statements hold.*

(a) The defender's trust margin is zero for all $\mathbf{p}^0 \in \Delta\mathcal{X}$ and credible generators if and only if $\rho^s_D \leq 0$. The optimal generator contains zero information.

(b) The defender's trust margin is non-negative for all $\mathbf{p}^0 \in \Delta\mathcal{X}$ and credible generators if and only if $\rho^s_D > 0$. Moreover, the optimal generator contains full information. If $\mathbf{p}^0$ is an interior point of the $(N-1)$-simplex and there exists at least one $\theta \in \Theta$ under which no actions dominate, then the defender's trust margin is positive.

Proof. The given conditions lead to $\tilde{v}_D(\mathbf{p}^0) = \mathbb{E}_{\theta\sim\hat{b}_D}\mathbb{E}_{x\sim\mathbf{p}^0}[\rho_D^s\hat{v}_U(x,\theta,a_\theta^*(\mathbf{p}^0)) + \rho_D^t(x,\theta)] = \rho_D^s\mathbb{E}_{\theta\sim\hat{b}_D}\mathbb{E}_{x\sim\mathbf{p}^0}[\hat{v}_U(x,\theta,a_\theta^*(\mathbf{p}^0))] + \mathbb{E}_{\theta\sim\hat{b}_D}\mathbb{E}_{x\sim\mathbf{p}^0}[\rho_D^t(x,\theta)]$. Proposition 12 has shown that $\mathbb{E}_{x\sim\mathbf{p}^0}[\hat{v}_U(x,\theta,a_\theta^*(\mathbf{p}^0))]$ is a PWLC function of $\mathbf{p}^0$ for each $\theta\in\Theta$. Since $\hat{b}_D(\theta)\geq 0, \forall\theta\in\Theta$, the linear combination $\mathbb{E}_{\theta\sim\hat{b}_D}\mathbb{E}_{x\sim\mathbf{p}^0}[\hat{v}_U(x,\theta,a_\theta^*(\mathbf{p}^0))]$ is also PWLC. The term $\mathbb{E}_{\theta\sim\hat{b}_D}\mathbb{E}_{x\sim\mathbf{p}^0}[\rho_D^t(x,\theta)]$ is a linear function of $\mathbf{p}^0$. Thus, $\tilde{v}_D$ is a piece-wise linear and concave (resp. linear) function of $\mathbf{p}^0$ if and only if $\rho_D^s<0$ (resp. $\rho_D^s=0$). If $\tilde{v}_D$ is concave or linear over the entire belief region $\Delta\mathcal{X}$, its convex hull is itself. Thus, $V_D(\mathbf{p}^0)=\tilde{v}_D(\mathbf{p}^0)$ for all $\mathbf{p}^0\in\Delta\mathcal{X}$ and any zero-information generator is optimal. Similarly, $\tilde{v}_D$ is PWLC if and only if $\rho_D^s>0$, and any full-information generator is optimal. If there exists at least one $\theta\in\Theta$ under which no actions dominate, then $\tilde{v}_D$ is strictly convex over the entire belief region. Thus, we have $V_D(\mathbf{p}^0)<\tilde{v}_D(\mathbf{p}^0)$ when $\mathbf{p}^0$ is an interior point of the $(N-1)$-simplex. □

Theorem 7 shows that when two players' utilities are linearly dependent, the user's manageability depends on the sign of the scaling factor ρ_D^s rather than its value. Thus, the user's level of maliciousness has a threshold impact on the manageability and the threshold is 0.

10.3.2 Incentive Modulator and Trust Manipulator

We illustrate the modulator design and the manipulator design in Section 10.3.2 and 10.3.2, respectively. The GMM mechanism design is presented in Section 10.3.2.

Joint Design of Generator and Modulator

The modulator incentivizes unmanageable users and increases the security and efficiency of the networks. Under the binary state $N=2$, Fig. 10.5 illustrates the defender's prior utility with the modulator in orange solid lines. The orange solid lines are different from the blue ones in two folds. From the user's perspective, the modulator changes the user's expected utility under different actions and thus results in translations of the dashed lines in Fig. 10.3. Those translations change the belief region partition, e.g., the right shifts of $t_1^{\theta_1}$ and $t_1^{\theta_2}$ in Fig. 10.5. From the defender's perspective, the modulator modifies her utility in each new belief regions, and the value of the modification is $\mathbb{E}_{x\sim\mathbf{p}^0}\mathbb{E}_{\theta\sim b_D(\cdot|x)}[\gamma c(a_\theta^*(\mathbf{p}^0))]$. If the defender's belief is independent of state, i.e., $b_D(\theta|x)=\hat{b}(\theta),\forall\theta\in\Theta,\forall x\in\mathcal{X}$, then the defender's utility change $\mathbb{E}_{x\sim\mathbf{p}^0}\mathbb{E}_{\theta\sim b_D(\cdot|x)}[\gamma c(a_\theta^*(\mathbf{p}^0))]=\gamma\mathbb{E}_{\theta\sim\hat{b}_D(\cdot)}[c(a_\theta^*(\mathbf{p}^0))]$ is a constant with respect to $\mathbf{p}^0$ in each new belief region. When the state space is binary as shown in Fig. 10.5, it means that designing c introduces translations but not rotations to each segment of the function $\tilde{v}_D$.

The joint design of the modulator and the generator results in the new convex hull denoted by the dashed blue lines in Fig. 10.5. Based on both players' perspectives, the optimal design needs to strike a balance between incentivizing users to change their belief region partitions and the costs to provide the incentives. Take Fig. 10.5 as an example, we observe that the modulator incurs costs to the defender for all actions, i.e., $c(a)\leq 0,\forall a\in\mathcal{A}$. Thus, in all three belief regions, the defender's prior utilities with the modulator, represented by the solid orange lines, are lower than the ones without the modulator, represented by the solid blue lines. However, the benefit of the user's incentive change outweighs the costs; i.e., the defender's optimal posterior utility $V_D(p_1^0)$ increases from node 4 in blue to node 4 in orange.

Joint Design of Generator and Manipulator

The manipulator directly distorts the user's prior belief to elicit desirable behaviors. When the generator cannot be designed, the manipulator design is equivalent to the process of finding the initial belief $\mathbf{p}_g^0 := arg\max_{\mathbf{p}^0 \in \Delta\mathcal{X}} \tilde{v}_D(\mathbf{p}^0)$ that achieves the global maximum of the prior utility $\tilde{v}_D$. Proposition 17 proves the existence of the optimal distorted belief $\mathbf{p}_g^0$.

Proposition 17. *For any given $\hat{v}_D, \hat{v}_U$ of two players, there exists an initial belief $\mathbf{p}_g^0 \in \Delta\mathcal{X}$ at the boundary of the convex polytopes $\mathcal{C}_{\{a^1,\cdots,a^M\}}, \forall a^l \in \mathcal{A}, l \in \{1, \cdots, M\}$, such that $\mathbf{p}_g^0 = arg\max_{\mathbf{p}^0 \in \Delta\mathcal{X}} \tilde{v}_D(\mathbf{p}^0)$.*

Proof. For each $\hat{v}_D, \hat{v}_U$, the global maximum $\tilde{v}_D(\mathbf{p}_g^0) = \max_{\mathbf{p}^0 \in \Delta\mathcal{X}} \tilde{v}_D(\mathbf{p}^0)$ exists and has a finite value due to Theorem 6. Proposition 16 shows that the global maximum is either unique or infinite. In either case, at least one global maximum is at the boundary of the convex polytopes due to the piece-wise linear property stated in Proposition 15. □

When the optimal generator is applied, the joint design of the manipulator and the generator is equivalent to the process of finding the initial belief $\bar{\mathbf{p}}_g^0 := arg\max_{\mathbf{p}^0 \in \Delta\mathcal{X}} V_D(\mathbf{p}^0)$ that achieves the global maximum of V_D. Based on the piece-wise linear property of $\tilde{v}_D$ in Proposition 15, the prior utility $\tilde{v}_D$ and its concave closure V_D share the same global maximum. Thus, $\mathbf{p}_g^0 = \bar{\mathbf{p}}_g^0$ and the optimal generator contains zero information. Take Fig. 10.5 as an example, $\mathbf{p}_g^0 = [t_1^{\theta_2}, 1 - t_1^{\theta_2}]$ achieves the global maximum denoted by node 2's ordinate, and node 2 is on both the solid and the dashed lines. These results are summarized in Theorem 8.

Theorem 8. *The design of optimal overt manipulator changes the common initial belief* $\mathbf{p}^0$ *into* $\mathbf{p}^0_g = \bar{\mathbf{p}}^0_g$*. The defender's optimal posterior utility has the value of* $\tilde{v}_D(\mathbf{p}^0_g) = V_D(\mathbf{p}^0_g)$ *and is independent of the initial belief* $\mathbf{p}^0 \in \Delta\mathcal{X}$*. In the joint design of the overt manipulator and the generator, the optimal generator contains zero information.*

Design of the GMM Mechanism

We incorporate the modulator design into the joint design of the generator and the manipulator to complete the GMM mechanism design. Based on the analysis in Section 10.3.2, the first step of the GMM design is to determine the optimal modulator $c^* \in \mathcal{C}$ that results in the prior utility function with the largest value of the global maximum, i.e., $c^* = arg \max_c [\max_{\mathbf{p}^0 \in \Delta\mathcal{X}} \tilde{v}_D(\mathbf{p}^0)]$. With the given modulator c^*, the second step of the design is to reduce the problem to the joint design of modulator and manipulator presented in Section 10.3.2.

Remark 27 (**Separation Principle**)**.** *The two-step design of the GMM mechanism shows that the defender can design the optimal modulator* $c^* \in \mathcal{C}$ *independently.*

We identify the *equivalence principle* in Remark 28 based on the results in Theorem 8. If the overt manipulator allows the defender to manipulate the initial belief arbitrarily, then the optimal generator contains zero information; i.e., the defender no longer needs the optimal generator to achieve her optimal posterior utility. Note that the equivalence principle does not mean that the generator is redundant. When the belief manipulation is not arbitrary and under practical constraints (e.g., the belief changes within a limited range), the joint design of the two components can yield better performance than the single design of the manipulator.

Remark 28 (Equivalence Principle). *For any given modulator $c \in \mathcal{C}$, the joint design of the generator and the overt manipulator results in the same outcomes as the single design of the overt manipulator does.*

10.4 Case Study

In Section 10.4, we illustrate how the defender can use the DG to mitigate insider threats where honeypots are configured adaptively to detect and deter misbehavior.

10.4.1 Model Description

We have $\Theta = \{\theta^b, \theta^g\}$, $\mathcal{X} = \{x^H, x^N\}$, and $\mathcal{A} = \{a_{DO}, a_{AC}\}$ based on the running example introduced in Section 10.1.1, Example 4, and Example 5. The true percentage of honeypots $p_D^{0,H} := b(x^H) \in [0, 1]$, is only known to the SOC. Thus, the insiders' perceived honeypot percentage $p_U^{0,H} := b_U(x^H|\theta) \in [0, 1], \forall\theta \in \Theta$, can be different from the true percentage.

Table 10.1 lists the utilities of the SOC and the insiders. The column represents the binary state of a node, and the row represents the insiders' actions. In each matrix entry, we list the payoffs resulting from the selfish (resp. adversarial) insiders on the left (resp. right) of the semicolon. When the insider chooses not to access a node, we calibrate the payoffs to be 0 for both the SOC and the insiders. The other four possible scenarios are listed as follows. First, a selfish insider's access to a normal server maintains the organization's normal operation and results in a positive reward $r_D > 0$ (resp. $r_U > 0$) on average to the organization (resp. the selfish insider). Second, when an adversarial insider accesses a normal server, he disrupts the normal operation and compromises confidential data, which brings him a reward of $\phi_U^N r_U > 0$ and incurs a security loss of $\phi_D^N r_D < 0$ to the organization. Third, if an adversarial insider accesses a honeypot, he is detected and prohibited from data theft. Meanwhile, the SOC obtains valuable threat intelligence. We use $\phi_D^H > 0$ and $\phi_U^H < 0$ to represent the degrees of the SOC's gain and the adversarial insider's loss, respectively. Finally, once a selfish insider accesses the honeypot, the SOC has to quarantine the insider and investigate the incident, which incurs a suspension of normal services as well as an investigation cost. Meanwhile, the selfish insider also receives penalties and additional security training sessions. We use $\phi_D^g r_D < 0$ and $\phi_U^g r_U < 0$ to represent the cost for the SOC and the selfish insider, respectively.

Selfish θ^g; Adversarial θ^b	Honeypot x^H	Normal Server x^N
No Access a_{DO}	0 ; 0	0 ; 0
Access a_{AC}	$r_i\phi_i^g$; $r_i\phi_i^H$	r_i ; $r_i\phi_i^N$

Table 10.1: Two players' utilities $v_i(x, \theta, a), i \in \{D, U\}$.

Compared to a computing system that precisely follows its instructions, human insiders alter their behaviors in response to (dis)incentives. In this case study, the (dis)incentives refer to the insider's authentication cost $c(a_{AC}) := r_U\phi^0$ to access a node, where the ratio $\phi^0 \in \mathbb{R}$ takes the value of 0 in the default setting. We assume that the SOC can increase (i.e., $\phi^0 < 0$) or decrease (i.e., $\phi^0 > 0$) an insider's authentication cost at no additional cost, i.e., $\gamma = 0$. The revenues, losses, and costs can be quantified in dollars and their values vary for different security scenarios.

Threshold Policy Analysis

In this case study, both selfish and adversarial insiders share the same prior belief $p_U^{0,H} \in [0,1]$. Hence they share the same posterior belief denoted by $p_U^H \in [0,1]$ and adopt the following threshold policies. Define the decision thresholds of the selfish and the adversarial insiders as $t^g(\phi^0) := \max\{\min\{(1-\phi^0)/(1-\phi_U^g), 1\}, 0\}$ and $t^b(\phi^0) := \max\{\min\{(\phi_U^N - \phi^0)/(\phi_U^N - \phi_U^H), 1\}, 0\}$, respectively. Since both denominators are positive, i.e., $1 - \phi_U^g > 1$ and $\phi_U^N - \phi_U^H > 0$, the selfish insider (resp. the adversarial insider) chooses to access a node if and only if the node is unlikely to be a honeypot, i.e., $p_U^H < t^g(\phi^0)$ (resp. $p_U^H < t^b(\phi^0)$). If a selfish (resp. adversarial) insider accesses a node, his expected utility $r_U(1 - \phi^0 + p_D^{0,H}(\phi_U^g - 1))$ (resp. $r_U(\phi_U^N - \phi^0 + p_D^{0,H}(\phi_U^H - \phi_U^N)))$ decreases linearly in $p_D^{0,H}$, i.e., the true percentage of honeypots.

Since the selfish and adversarial insiders share the same insider information, the difference in their decision thresholds results purely from their incentive misalignment. Given the insiders' utility matrices, the SOC can change their incentives and elicit desirable behaviors by a proper design of the authentication cost determined by the ratio ϕ^0. If $\phi^0 \leq \phi_U^g < 0$ (resp. $\phi^0 \leq \phi_U^H < 0$), then the selfish (resp. adversarial) insider chooses a_{AC} for all security scenarios. If $\phi^0 \geq 1$ (resp. $\phi^0 \geq \phi_U^N > 0$), then the selfish (resp. adversarial) insider chooses a_{DO} for all security scenarios. Since the deceptive honeypot configuration can possibly change insiders' behaviors only if ϕ^0 is in the region $[\min(\phi_U^g, \phi_U^H), \max(1, \phi_U^N)]$, we refer to the region as the *incentivized region* of ϕ^0. As a special case of Proposition 13, Corollary 2 shows that security policies $s_{\{a_{DO}, a_{AC}\}}$ and $s_{\{a_{AC}, a_{DO}\}}$ cannot be both enforceable for any node in the corporate network.

Corollary 2. *If $\phi_U^g < 0, \phi_U^N > 0, \phi_U^H < 0$, then for all $\phi^0 \in \mathbb{R}$ and credible con-*

figuration $\pi \in \Pi$, *either* $\pi(s_{\{a_{DO},a_{AC}\}}|x) = 0, \forall x \in \{x^H, x^N\}$, *or* $\pi(s_{\{a_{AC},a_{DO}\}}|x) = 0, \forall x \in \{x^H, x^N\}$.

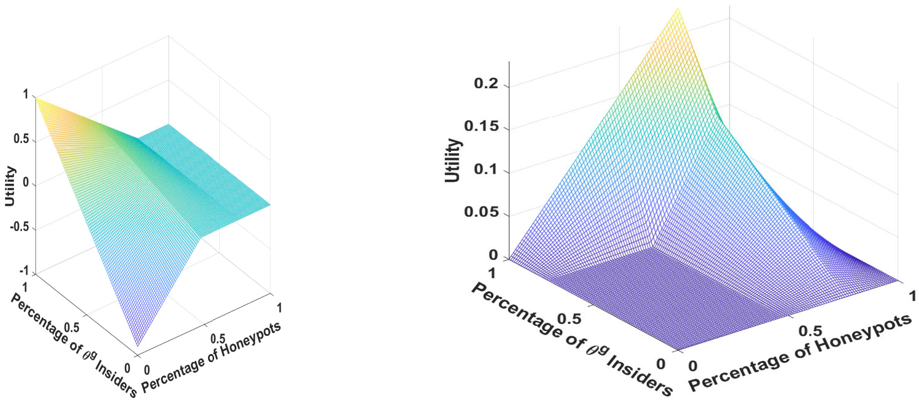
10.4.2 Numerical Results

Following the insider categorization in Section 10.1, we re-weight the percentage from the VCDB and adopt $q^g := b_D(\theta^g|x) = 0.32$ and $q^b := b_D(\theta^b|x) = 0.68$ for all $x \in \{x^N, x^H\}$ as the benchmark value of the insiders' type statistics. Based on the analysis in Section 10.4.1, the values of r_U do not affect the insiders' actions, and the value of r_D only scales the SOC's utility by a constant. Thus, we normalize $r_U = r_D = 1$. We consider $\phi_U^g = \phi_D^g = -0.3$, $\phi_U^H = -\phi_D^H = -1$, and $\phi_U^N = -\phi_D^N = 0.9$ as the benchmark values. Then, the selfish insider has the same utility as the SOC, i.e., $v_D(x, \theta^g, a) = v_U(x, \theta^g, a), \forall x \in \{x^H, x^N\}, \forall a \in \{a_{AC}, a_{DO}\}$, while the adversarial insider has an exactly opposite utility to the one of the SOC, i.e., $v_D(x, \theta^b, a) = -v_U(x, \theta^b, a), \forall x \in \{x^H, x^N\}, \forall a \in \{a_{AC}, a_{DO}\}$. In Section 10.4.2, the SOC cannot change the authentication cost, i.e., $c(a_{AC}) = 0$. In Sections 10.4.2 and 10.4.2, the insider has the correct prior belief of the honeypot percentage, i.e., $p_U^{0,H} = p_D^{0,H}$.

Security Posture under the Optimal Generator

Fig. 10.6a shows how the SOC's normalized revenue $\tilde{v}_D$ without the optimal generator is affected by the percentages of honeypots and the selfish insiders, respectively. The maximum (resp. minimum) value of $\tilde{v}_D$ is achieved when insiders are all selfish (resp. adversarial) and no honeypots are applied. The two decision thresholds $t^b(0)$ and $t^g(0)$ divide the percentage of honeypots into three regions, i.e., high, medium, and low, in which the insiders' behaviors and the SOC's normalized revenue $\tilde{v}_D$ have different characteristics.

If the intended security outcomes are not achieved due to the insiders' misbehavior, the SOC can apply the optimal generator to elicit desirable behaviors and reduce the cyber risks of the organization. To illustrate the effectiveness of the optimal generator, we plot the maximum trust margin in Fig. 10.6b. Fig. 10.6b corroborates Theorem 7; i.e., when all insiders are adversarial (resp. selfish), no (resp. all) credible generators, including the optimal one, can improve the SOC's normalized revenue for any percentage of honeypots $p_D^{0,H} \in [0,1]$. The flat region represented by $q^g \in [0, (\phi_D^N - \phi_D^H)/(\phi_D^g - 1 + \phi_D^N - \phi_D^H)]$ and $p_D^{0,H} \in [0, \min(t^b(0), t^g(0))]$ identifies two critical thresholds. On the one hand, we refer to $(\phi_D^N - \phi_D^H)/(\phi_D^g - 1 + \phi_D^N - \phi_D^H)$ as the insider's *motive threshold* that is used to quantify the average motive of the entire insider population. If the percentage of adversarial insiders exceeds the *motive threshold*, then insiders' behaviors are on average destructive to the organization. On the other hand, we refer to $\min(t^b(0), t^g(0))$ as the *deterrence threshold* that measures the adequacy of the honeypots. If the percentage of honeypots is below the *deterrence threshold*, then



(a) Prior utility $\tilde{v}_D$.

(b) Maximum trust margin.

Figure 10.6: SOC's utilities vs. $p_D^{0,H} \in [0,1]$ and $q^g \in [0,1]$.

the SOC does not have a sufficient number of honeypots to create a credible threat for the insiders not to access nodes in the corporate network. Based on Definition 36, the insiders are unmanageable in the flat region.

For the other regions, the insiders are manageable, and the optimal generator can effectively reduce the cyber risk of the organization. The increase depends on the percentage of selfish insiders and honeypots. When the percentage of honeypots is $t^g(0)$ and insiders are all selfish, the organization's revenue with the optimal generator is 114 times higher than the one without the optimal generator. Averaged over the entire region of $q^g \in [0,1]$ and $p_D^{0,H} \in [0,1]$, the organization's revenue with the optimal generator is 35.6% higher than the one without the optimal generator. The results in Fig. 10.6 demonstrate that the optimal generator design provides a constructive way to quantify the accuracy of the information that the SOC should reveal to the insiders to establish trust with them, while in the meantime, retain her information advantage to elicit desirable insider behaviors and maximize the organization's well-being. These results provide a guideline to address the challenges identified in 2c and 2d of Table 2 in [147].

Security Posture under Various Modulators

In Section 10.4.2, we investigate how the (dis)incentives affect the insiders' behaviors and the security posture of the insider network. In Fig. 10.7, we plot the decision thresholds of selfish and adversarial insiders in blue and red, respectively. Since the blue line has a steeper slope than the red line, Fig. 10.7 demonstrates that the same authentication cost affects the selfish insiders more significantly than the adversarial ones. As defined in Definition 33, two types of insiders are identifiable under posterior belief p_U^H if $p_U^H \in [t^b(\phi^0), t^g(\phi^0)]$. Furthermore, a larger difference in the two thresholds, i.e., $t^g(\phi^0) - t^b(\phi^0)$, indicates a higher incentive misalignment between selfish and adversarial insiders.

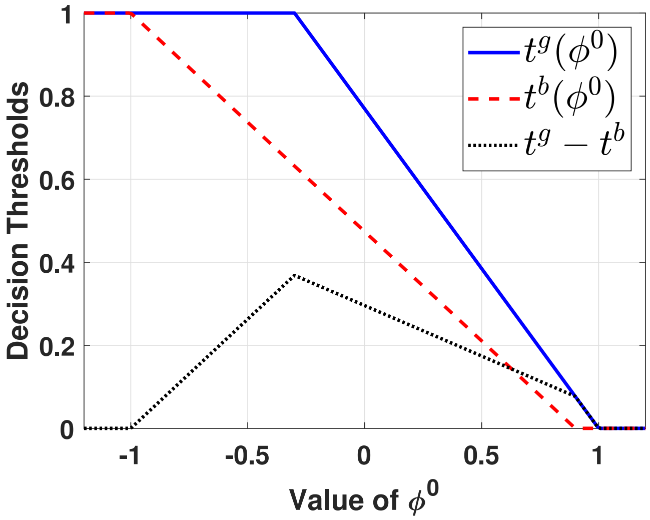
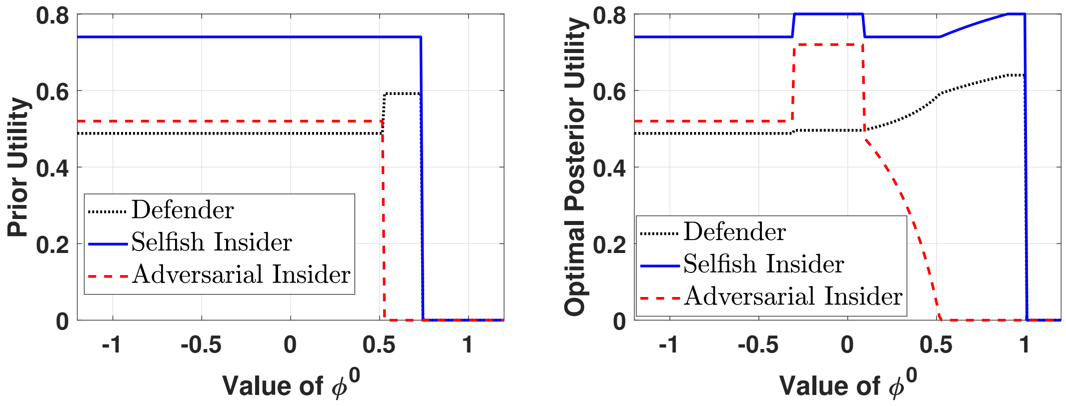


Figure 10.7: The adversarial and the selfish insiders' decision thresholds $t^b(\phi^0)$ and $t^g(\phi^0)$ in the red dashed line and the blue solid line, respectively. The difference $t^g(\phi^0)-t^b(\phi^0)$ denoted in the black dotted line represents their utility misalignment.

Fig. 10.8a illustrates the organization's original payoff $\tilde{v}_D$ without a generator. The selfish insider and the SOC achieve a win-win situation at the region $\phi^0 \in [0.5, 0.74]$ as they both achieve their maximum payoffs at that region. The adversarial insider and the SOC cannot achieve a win-win situation for all $\phi^0 \in \mathbb{R}$ as adversarial insiders seeking to compromise sensitive data and sabotage the organization have a completely misaligned payoff structure. Fig. 10.8b illustrates the organization's improved payoff V_D when the optimal generator is applied. The results show that the optimal generator can always increase the payoffs of the selfish insiders and the organization regardless of the (dis)incentives represented by $\phi^0 \in \mathbb{R}$. Win-win situations still exist (resp. do not exist) for the SOC and the selfish (resp. adversarial) insider.



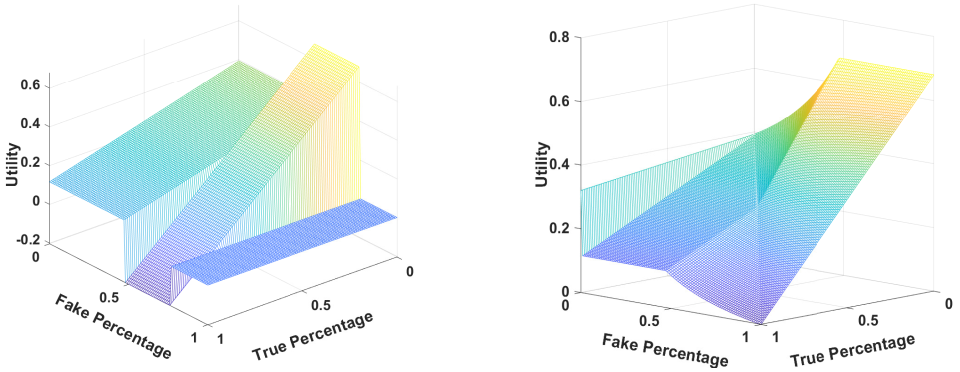
(a) Players' prior utilities.

(b) Optimal posterior utilities.

Figure 10.8: Utilities of the SOC, selfish insiders, and adversarial insiders in the dotted black, the solid blue, and the dashed red lines, respectively.

Security Posture under Covert and Overt Trust Manipulators

In Section 10.4.2, the SOC can generate ambiguous or fake reports of the honeypot percentage so that the insiders' initial beliefs of the honeypot percentage deviate from the truth, i.e., $p_U^{0,H} \neq p_D^{0,H}$. Figs. 10.9a and 10.9b illustrate the SOC's payoffs with and without the optimal generator, respectively, under different values of $p_U^{0,H}$ and $p_D^{0,H}$. In Fig. 10.9a, the insiders' initial beliefs fall into the following three regions. If $p_U^{0,H} \in [t^g(0), 1]$, both types of insiders choose not to access the node. Then, the SOC's normalized payoff $\tilde{v}_D$ is zero regardless of the true percentage of honeypots $p_D^{0,H}$. If $p_U^{0,H} \in [t^b(0), t^g(0)]$, selfish insiders choose a_{AC} and adversarial insiders choose a_{DO}. Then, reducing the percentage of honeypots increases the SOC's normalized payoff $\tilde{v}_D$ as it reduces the false alarm rate when selfish insiders access the honeypots. If $p_U^{0,H} \in [0, t^b(0)]$, both types of insiders choose to access the node. Then, reducing the percentage of honeypots also increases the SOC's normalized payoff $\tilde{v}_D$. However, the increase rate is lower than the one in the second region as the two types of insiders take the same action and are not identifiable.



(a) Prior utility $\tilde{v}_D$. (b) Optimal posterior utility.

Figure 10.9: SOC's utilities vs. $p_D^{0,H} \in [0,1]$ and $p_U^{0,H} \in [0,1]$.

These results illustrate that without a deceptive generator, the SOC may not always benefit from faking the percentage of honeypots. On the contrary, when the optimal generator is applied in Fig. 10.9b, the SOC can benefit from a fake percentage of honeypots for all $p_D^{0,H}, p_U^{0,H} \in [0,1]$. Moreover, the benefit of faking honeypot percentage is a non-decreasing function of $|p_D^{0,H} - p_U^{0,H}|$. Thus, the SOC obtains a higher payoff V_D with the optimal generator when there is a larger mismatch between the true and the fake percentages of honeypots. The maximum value of V_D is achieved when the true percentage of honeypots is zero and the SOC makes the insiders believe that the percentage of honeypots exceeds $t^b(0)$. Averaged over the true percentage $p_D^{0,H} \in [0,1]$ and the fake one $p_U^{0,H} \in [0,1]$, the SOC's payoff with the optimal generator, i.e., V_D is 59.3% higher than her original payoff $\tilde{v}_D$.

Part VI

Hodatology for Cognitive Security

Chapter 11

ADVERT: An Attention Enhancement Mechanism for Phishing Prevention

Following Section 1.3.2, attacks exploiting the *innate* and the *acquired* vulnerabilities of human users have posed severe threats to cybersecurity. In Chapter 11, we focus on inattention, one type of innate human vulnerability, and use phishing email as a prototypical scenario to explore the users' visual behaviors when they determine whether a received email is secure or not. Based on the users' eye-tracking data and phishing recognition results, we develop ADVERT[1] to provide a human-centric data-driven attention enhancement mechanism for phishing prevention. In particular, ADVERT enables an adaptive visual-aid generation to guide and sustain the users' attention to the right content of an email and consequently makes users less likely to fall victim to phishing. The design of the ADVERT contains two

[1] ADVERT is an acronym for ADaptive Visual aids for Efficient Real-time security-assistive Technology.

feedback loops of attention enhancement and phishing prevention at short and long time scales, respectively, as shown in Fig. 11.1.

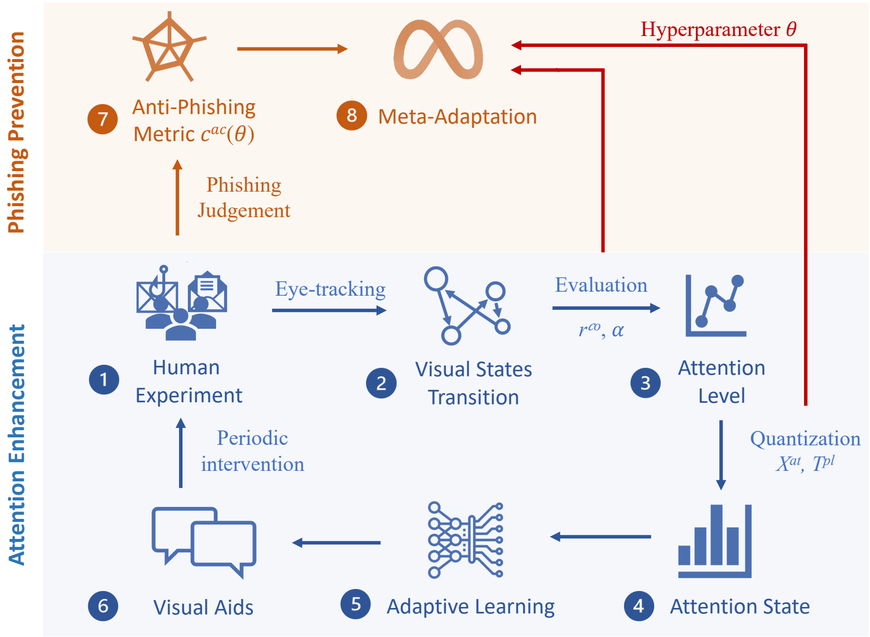


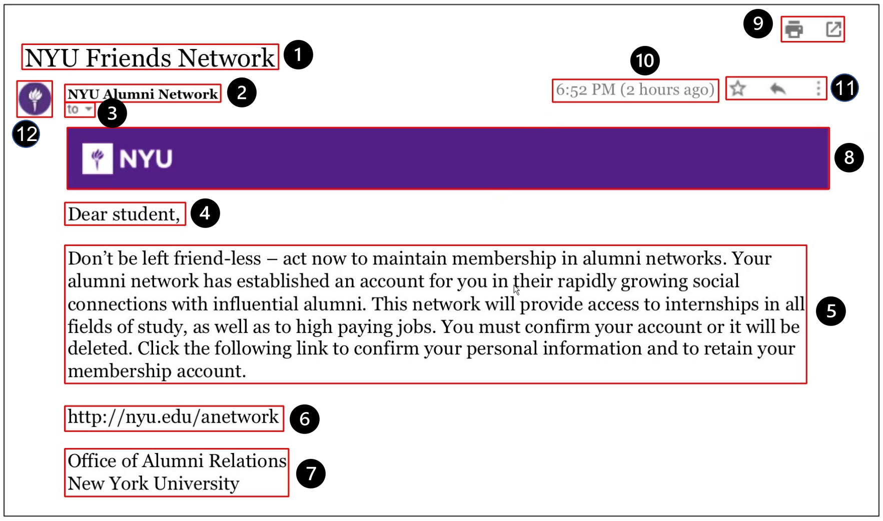
Figure 11.1: The design diagram of ADVERT. The adaptive learning loops of the attention enhancement mechanism and the phishing prevention mechanism are highlighted using juxtaposed blue and orange backgrounds, respectively. Since a user needs to persistently pay attention to an email to make a phishing judgment, the meta-adaptation feedback in orange updates less frequently than the feedback of attention enhancement in blue.

The bottom part of Fig. 11.1 in blue illustrates the design of adaptive visual aids (e.g., highlighting, warnings, and educational messages) to engage human users in email processing. First, as a human user reads emails and judges whether they are phishing or legitimate, a covert eye-tracking system can record the user's eye-gaze locations and pupil sizes in real-time. Second, based on the eye-tracking data, we abstract the email's Areas of Interest (AoIs), e.g., title, hyperlinks, attachments, etc, and develop a Visual State (VS) transition model to characterize the eye-gaze dynamics. Third, we develop system-level attention metrics to evaluate the user's attention level based on the VS transition trajectory. Then, we quantize the attention level to obtain the Attention State (AS) and develop adaptive learning algorithms to generate visual aids as feedback of the AS. The visual aids change the user's hidden cognitive states and lead to the set of eye-tracking data with different patterns of VS transition and AS, which then updates the design of visual aids and enhances attention iteratively.

The attention enhancement loop serves as a stepping-stone to achieving the ultimate goal of phishing prevention. The orange background in the top part of Fig. 11.1 illustrates how we tune the hyperparameters in the attention enhancement loop to safeguard users from phishing emails. First, we create a metric to evaluate the user's accuracy in phishing recognition under the current attention enhancement mechanism. Then, we iteratively revise the hyperparameters to achieve the highest accuracy. Since the accuracy evaluation depends on the implementation of the entire attention enhancement loop, the evaluation is costly and time-consuming. Thus, we leverage Bayesian optimization to propose an efficient meta-level tuning algorithm to improve the accuracy.

11.1 Attention Enhancement Mechanism

As illustrated by step 1 of Fig. 11.1, we consider a group of M human users who vet a list of N emails and classify them as phishing or legitimate. As a user $m \in \mathcal{M} := \{1, \cdots, M\}$ reads an email $n \in \mathcal{N} := \{1, \cdots, N\}$ on the screen for a duration of T_m^n, the eye-tracking device records the vertical and the horizontal coordinates of his eye gaze point in real-time. To compress the sensory outcomes and facilitate RL-driven attention enhancement solutions, we aggregate potential gaze locations (i.e., pixels on the screen) into a finite number of I non-overlapping Areas of Interest (AoIs) as shown in Fig. 11.2. We index each potential AoI by



NYU Friends Network

NYU Alumni Network

to

6:52 PM (2 hours ago)

NYU

Dear student,

Don't be left friend-less – act now to maintain membership in alumni networks. Your alumni network has established an account for you in their rapidly growing social connections with influential alumni. This network will provide access to internships in all fields of study, as well as to high paying jobs. You must confirm your account or it will be deleted. Click the following link to confirm your personal information and to retain your membership account.

http://nyu.edu/anetwork

Office of Alumni Relations
New York University

Figure 11.2: A sample email with 12 AoIs. In sequence, they are the email's title, the sender's information, the receiver's information, the salutation, the email body, the URL, the sender's signature, the organization logo, the 'print' and 'share' buttons, the timestamp, the 'bookmark' and 'forward' buttons, and the sender's profile picture. The AoI partition in red boxes and their index numbers in black circles are invisible to users.

$i \in \mathcal{I} := \{1, 2, ..., I\}$.

Each email does not need to contain all the AoIs and the AoI partition remains unknown to the users. Previous works [138, 144, 175] have identified the role of AoIs in helping human users recognize phishing while different research goals can lead to different AoI partitions. For example, the email body AoI (i.e., area 5 in Fig. 11.2) can be divided into finer AoIs based on the phishing indicators such as misspellings, grammar mistakes, and threatening sentences. We refer to all other areas in the email (e.g., blank areas) as the *uninformative area*. When the user's eyes move off the screen during the email vetting process, no coordinates of the gaze location are available. We refer to these off-screen areas as the *distraction* *area.*

11.1.1 Visual State Transition Model

As illustrated by step 2 of Fig. 11.1, we establish the following transition model based on the AoI to which the user's gaze location belongs at different times. We define $\mathcal{S} := \{s^i\}_{i \in \mathcal{I}} \cup \{s^{ua}, s^{da}\}$ as the set of $I+2$ *Visual States (VSs)*, where s^i represents the i-th AoI; s^{ua} represents the *uninformative area*; and s^{da} represents the *distraction area*. We provide an example transition map of these VSs in Fig. 11.3. The links represent the potential shifts of the gaze locations during the email reading process; e.g., the users can shift their focus from the title to the main content or the distraction area. We omit most links for illustration purposes; e.g., it is also possible for a user to regain attention to the AoIs from distraction or inadvertence.

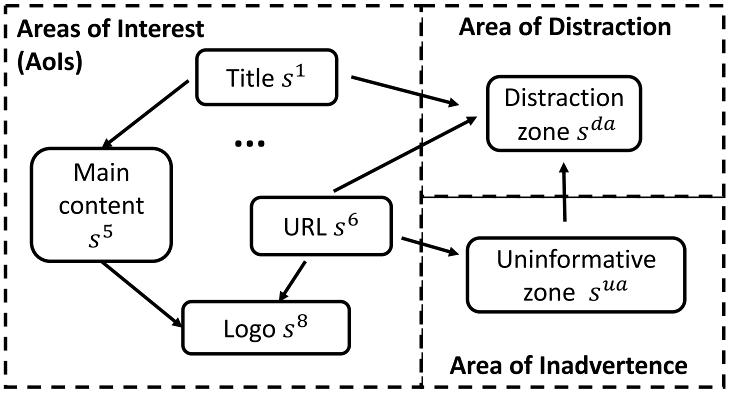


Figure 11.3: Transitions among visual states in $\mathcal{S}$. The VS indexes are consistent with Fig. 11.2.

We denote $s_t \in \mathcal{S}$ as the VS of user $m \in \mathcal{M}$ vetting email $n \in \mathcal{N}$ at time $t \in [0, T_m^n]$. In this work, we do not distinguish among human users concerning their attention processes while they read different emails. Then, each user's gaze path during the interval $[0, T_m^n]$ can be characterized as the same stochastic process $[s_t]_{t\in[0,T_m^n]}$. The stochastic transition of the VSs divides the entire time interval $[0, T_m^n]$ into different *transition stages*. We visualize an exemplary VS transition trajectory $[s_t]_{t\in[0,T_m^n]}$ in Fig. 11.4 under $I = 4$ AoIs and $T_m^n = 50$ seconds. As denoted by the colored squares, 40 VSs arrive in sequence, which results in 40 discrete transition stages.

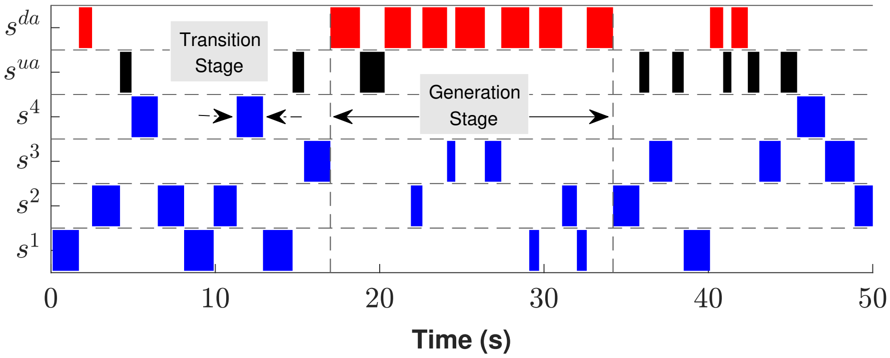


Figure 11.4: An exemplary visual state transition trajectory $[s_t]_{t\in[0,T_m^n]}$. The x-axis and the y-axis represent $T_m^n = 50$ seconds and $I + 2 = 6$ visual states, respectively. We denote visual states s^{da}, s^{ua}, and $\{s^i\}_{i\in\mathcal{I}}$ in red, black, and blue, respectively. Each generation stage contains different numbers of transition stages.

11.1.2 Feedback Visual-Aid Design

Propel visual aids can help guide and sustain the users' attention. Previous works have proposed different classes of visual aids to enhance phishing recognition, including highlights of contents [128, 223], warnings of suspicious hyperlinks and attachments [4, 44], and anti-phishing educational messages [194]. These potential classes of visual aids construct the visual-aid library denoted as a finite set $\mathcal{A}$.

As illustrated by step 6 of Fig. 11.1, different visual aids can affect the users' visual behaviors. The influence, however, can be beneficial (e.g., timely highlights prevent users from mind-wandering) or detrimental (e.g., extensive highlights make humans weary and less attentive to the AoIs). Due to the unpredictability and heterogeneity of human behaviors and their mental processes, there lacks mature theories or design rules to generate the most beneficial visual aids directly under different conditions. Moreover, the visual aid should adapt to the human visual attention that changes during the email vetting. Therefore, we apply reinforcement learning techniques to learn the dynamic design of visual aids based on the real-time evaluation of the user's attention status detailed in Section 11.1.3.

The sequence of adaptive visual aids is generated with a period of length T^{pl} and we refer to the time interval between every two visual aids as the *generation stage* indexed by $k \in \mathcal{K}_m^n := \{1, 2, \cdots, K_m^n\}$, where K_m^n is the maximum generation stage during $[0, T_m^n]$; i.e., $K_m^n T^{pl} \leq T_m^n$ and $(K_m^n + 1)T^{pl} \geq T_m^n$. Then, we denote $a_k \in \mathcal{A}$ as the visual aid at the k-th generation stage. Fig. 11.4 illustrates how visual aids affect the transition of visual states in $K_m^n = 3$ generation stages divided by the two vertical dashed lines. During the second generation stage, an improper visual aid leads to more frequent transitions to the distraction area and also a longer sojourn time at the visual state s^{da}. On the contrary, the proper visual aids during the first and the third generation stages engage the users and extend their attention spans, i.e., the amount of time spent on AoIs before a transition to s^{da} or s^{ua}.

11.1.3 Evaluation of Attention Status

From the VS transition trajectory (e.g., Fig. 11.4), we aim to construct the *Attention State (AS)* used as the feedback value for the adaptive visual-aid design. We define $\mathcal{X}$ as the set of all possible attention states. Previous works, e.g., [138, 169], have defined attention metrics based on the AoIs, e.g., the proportion of time spent on each AOI, gaze duration means, fixation count, and average duration. Compared to these detailed-level metrics extracted directly from raw eye-gaze data, we propose the following system-level metric of attention level based on the VS transition history as shown in Section 11.1.3. Such system-level metric serves as sufficient statistics to effectively characterize the attention status. Moreover, it preserves the users' privacy as the raw data of gaze locations can reveal sensitive information about their biometric identities, including gender, age, and ethnicity [119, 127].

To this end, we assign scores to each visual state in Section 11.1.3 to evaluate the user's attention (e.g., gaze at AoIs) and inattention (e.g., gaze at uninformative and distraction areas). The scores can be determined manually based on the expert recommendation and empirical studies (e.g., [169]), or based on other biometric data (e.g., the pupil sizes in Fig. 11.7). Moreover, we can apply Bayesian optimization for further fine-tuning of these scores as shown in Section 11.2.2.

Concentration Scores and Decay Rates

Both the gaze location and the gaze duration matter in the identification of phishing attacks. For example, at the first glance, users cannot distinguish the spoofed email address '`paypa1@mail.paypaI.com`' from the authentic one '`paypal@mail.paypal.com`' while a guided close look reveals that the lower case letter '`l`' is replaced by the number '`1`' and the capital letter '`I`'. Therefore, we assign a *concentration score* $r^{co}(s) \in \mathbb{R}$ to characterize the transient and the sustained attention associated with visual state $s \in \mathcal{S}$. Since the amount of information that a user can extract from a VS $s \in \mathcal{S}$ is limited, we use an exponential decay rate of

$\alpha(s) \in \mathbb{R}^+$ to penalize the effect of concentration score as time elapses. Different visual states can have different concentration scores and decay rates. For example, the email body AoI usually contains more information than other AoIs, and an extended attention span extracts more information, e.g., the substitution of letter ‘l’ into ‘I’, to identify the phishing email. Thus, the email body AoI turns to have a high concentration score and a low decay rate, which is corroborated in Table 11.1 based on the data set collected from human experiments [32] as shown in Section 11.3.

Cumulative Attention Level

We construct the metric for attention level illustrated in step 3 of Fig. 11.1 as follows. Let $W_k \in \mathbb{Z}^+$ be the total number of transition stages contained in generation stage $k \in \mathcal{K}_m^n$. Then, we define $t_k^{w_k}, w_k \in \{1, 2, \cdots, W_k\}$, as the duration of the w_k-th transition stage in the k-th generation stage. Take the gaze path in Fig. 11.4 as an example, the first generation stage contains $w_1 = 12$ transition stages and the first 7 transition stages last for a total of $\sum_{w_1=1}^{7} t_1^{w_1} = 10$ seconds. Based on the sets of scores associated with $s \in \mathcal{S}$, we compute the cumulative reward $u_k^{w_k}(s,t)$ at time t of the w_k-th transition stage in the k-th generation stage as $u_k^{w_k}(s,t) = \int_0^t r^{co}(s) e^{-\alpha(s)\tau} \cdot \mathbf{1}_{\{s=s^\tau\}} d\tau, 0 \leq t \leq t_k^{w_k}$. At generation stage k, we define $\bar{w}_k^t$ as the latest transition stage before time t, i.e., $\sum_{w_k=1}^{\bar{w}_k^t} t_k^{w_k} \leq t$ and $\sum_{w_k=1}^{\bar{w}_k^t+1} t_k^{w_k} > t$. Then, we define the user's *Cumulative Attention Level (CAL)* $v_k(t)$ over time interval $[(k-1)T^{pl}, t]$ at generation stage $k \in \mathcal{K}_m^n$ as the following cumulative reward

$$v_k(t) := \sum_{s \in \mathcal{S}} \sum_{w_k=1}^{\bar{w}_k^t} u_k^{w_k}(s,t), 0 \leq t \leq T^{pl}, \tag{11.1}$$

We visualize the CAL of $K_m^n = 3$ generation stages in Fig. 11.5 based on the exemplary gaze path in Fig. 11.4.

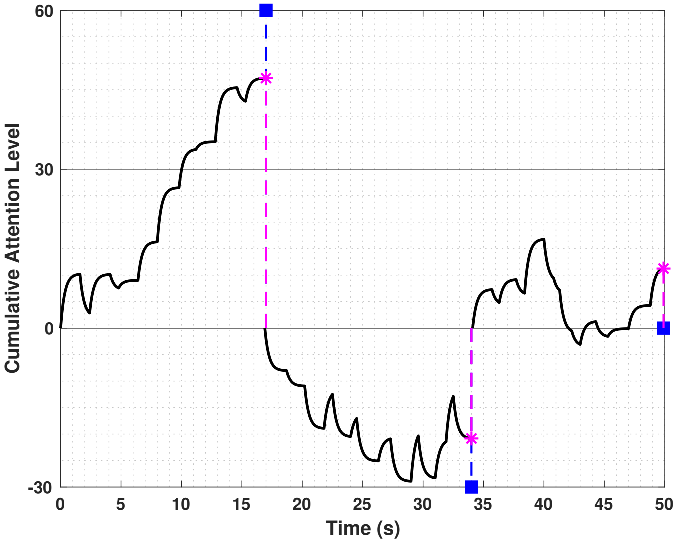


Figure 11.5: The user's cumulative attention level $v_k(t-(k-1)T^{pl}), k \in \mathcal{K}_m^n, t \in [(k-1)T^{pl}, kT^{pl}]$, over $K_m^n = 3$ generation stages in $T_m^n = 50$ seconds. The horizontal lines quantize $v_k(t)$ into $X = 4$ values that form the finite set $\mathcal{X} = \{-30, 0, 30, 60\}$. The purple star and the blue square denote the values of $\bar{v}_k \cdot T^{pl}$ and $\bar{v}_k^{qu} \cdot T^{pl}$, respectively, at each generation stage $k \in \mathcal{K}_m^n$.

Since $v_k(t)$ is bounded for all $k \in \mathcal{K}_m^n, t \in [0, T^{pl}]$, we can quantize it into X finite values to construct the set $\mathcal{X}$ of the attention states illustrated by step 4 of Fig. 11.1. We represent the quantized value of $v_k(t) \in \mathbb{R}$ as $v_k^{qu}(t) \in \mathcal{X}$ for all $k \in \mathcal{K}_m^n, t \in [0, T^{pl}]$, and define the average attention level and quantized average attention level for each generation stage in Definition 37.

Definition 37. *Let $\bar{v}_k \in \mathbb{R}$ and $\bar{v}_k^{qu} \in \mathcal{X}$ denote the user's Average Attention Level (AAL) and Quantized Average Attention Level (QAAL) over generation stage $k \in \mathcal{K}_m^n$, respectively. They are measured by the improvement of CAL and the quantized value of the CAL improvement per unit time, i.e., $\bar{v}_k := v_k(T^{pl})/T^{pl}$ and*

$\bar{v}_k^{qu} := v_k^{qu}(T^{pl})/T^{pl}$, *respectively.*

11.1.4 Q-Learning via Consolidated Data

We elaborate on the adaptive learning block in step 5 of Fig. 11.1 in Section 11.1.4. Since the inspection time of a user reading one email is not sufficiently long, we consolidate a group of email inspection data to learn the optimal visual-aid generation policy over a population.

The QAAL $\bar{v}_k^{qu} \in \mathcal{X}$ represents the attention state at the generation stage $k \in \mathcal{K}_m^n$. Since the goal is to enhance the user's attention represented by the CAL, the reward function $R : \mathcal{X} \times \mathcal{A} \mapsto \mathbb{R}$ should be monotone concerning the value of $\bar{v}_k^{qu}$, e.g., $R(\bar{v}_k^{qu}, a_k) := \bar{v}_k^{qu}, \forall a_k \in \mathcal{A}$. In this work, we assume that each visual aid $a_k \in \mathcal{A}$ exerts the same statistical effect on the attention process regardless of different users and emails. Thus, we can consolidate the data set of $\bar{M} \in \{1, \cdots, M\}$ users and $\bar{N} \in \{1, \cdots, N\}$ emails[2] to learn the optimal visual-aid generation policy $\sigma \in \Sigma : \mathcal{X} \mapsto \mathcal{A}$ in a total of $\bar{K} := \sum_{m=1}^{\bar{M}} \sum_{n=1}^{\bar{N}} K_m^n$ stages. With a given discounted factor $\beta \in (0, 1)$, the expected long-term objective can be represented as $\max_{\sigma \in \Sigma} \mathbb{E}[\sum_{k=1}^{\bar{K}} (\beta)^k \cdot R(\bar{v}_k^{qu}, \sigma(\bar{v}_k^{qu}))]$.

The Q-table $[Q_k(\bar{v}_k^{qu}, a_k)]_{\bar{v}_k^{qu} \in \mathcal{X}, a_k \in \mathcal{A}}$ represents the user's attention pattern at generation stage $k \in \bar{\mathcal{K}} := \{1, \cdots, \bar{K}\}$, i.e., the estimated payoff of applying visual aid $a_k \in \mathcal{A}$ when the attention state is $\bar{v}_k^{qu} \in \mathcal{X}$. Let the sequence of learning rate $\gamma_k(\bar{v}_k^{qu}, a_k)$ satisfy $\sum_{k=0}^{\infty} \gamma_k(\bar{v}_k^{qu}, a_k) = \infty$ and $\sum_{k=0}^{\infty} (\gamma_k(\bar{v}_k^{qu}, a_k))^2 < \infty$ for all $\bar{v}_k^{qu} \in \mathcal{X}, a_k \in \mathcal{A}$. Then, we can update the attention pattern at each generation

[2] When sufficiently large data sets are available, we can carefully choose these $\bar{M}$ users to share similar attributes (e.g., ages, sexes, races, etc) and these $\bar{N}$ emails to belongs to the same categories (e.g., business or personal emails).

stage $k \in \bar{\mathcal{K}}$ as follows, i.e.,

$$\begin{aligned} Q_{k+1}(\bar{v}_k^{qu}, \sigma_k(\bar{v}_k^{qu})) = {} & Q_k(\bar{v}_k^{qu}, \sigma_k(\bar{v}_k^{qu})) \\ & + \gamma_k(\bar{v}_k^{qu}, \sigma_k(\bar{v}_k^{qu})) \cdot [R(\bar{v}_k^{qu}, \sigma_k(\bar{v}_k^{qu})) \\ & + \beta \max_{a \in \mathcal{A}} Q_k(\bar{v}_{k+1}^{qu}, a) - Q_k(\bar{v}_k^{qu}, \sigma_k(\bar{v}_k^{qu}))], \end{aligned} \tag{11.2}$$

where the visual-aid generation policy $\sigma_k(\bar{v}_k^{qu})$ at generate stage $k \in \bar{\mathcal{K}}$ is an ϵ_k-greedy policy; i.e., with probability $\epsilon_k \in [0, 1]$, the visual aid a_k is selected randomly from $\mathcal{A}$; with probability $1 - \epsilon_k$, the optimal visual aid $a_k^* \in \arg\max_{a \in \mathcal{A}} Q_k(\bar{v}_k^{qu}, a)$ is implemented. To obtain a convergent attention policy and visual-aid policy, the value of ϵ_k gradually decreases from 1 to 0.

11.2 Phishing Prevention Mechanism

The attention enhancement mechanism in Section 11.1 tracks the attention process in real-time to enable the adaptive visual-aid generation. By properly modifying the user's attention and engaging him in vetting emails, the attention enhancement mechanism serves as a stepping-stone to achieving the ultimate goal of phishing prevention. Empirical evidence and observations such as the Yerkes–Dodson law [228] have shown that a high attention level, or mental arousal, does not necessarily yield good performance. Thus, besides attention metrics, e.g., the AAL, we need to design anti-phishing metrics to measure the users' performance of phishing recognition as shown in Section 11.2.1.

In Section 11.2.2, we develop an efficient meta-level algorithm to tune the hyperparameters in the attention enhancement mechanism, e.g., the period length T^{pl} of the visual-aid generation, the number of attention states X, the attention scores $r^{co}(s), \alpha(s), \forall s \in \mathcal{S}$, etc. We denote these hyperparameters as one d-dimensional variable $\theta = [T^{pl}, X, [r^{co}(s)]_{s\in\mathcal{S}}, [\alpha(s)]_{s\in\mathcal{S}}] \in \mathbb{R}^d$ where $d = 2 + 2|\mathcal{S}|$. Let the i-th element θ^i be upper and lower bounded by $\bar{\theta}^i$ and $\underline{\theta}^i$, respectively. Thus, $\theta \in \Theta^d := \{[\theta^i]_{i\in\{1,\cdots,d\}} \in \mathbb{R}^d | \underline{\theta}^i \leq \theta^i \leq \bar{\theta}^i\}$.

11.2.1 Metrics for Phishing Recognition

As illustrated by step 7 in Fig. 11.1, we provide a metric to evaluate the outcome of the users' phishing identification under a given hyperparameter $\theta \in \Theta^d$. After vetting email $n \in \{1, \cdots, \bar{N}\}$, the user $m \in \{1, \cdots, \bar{M}\}$ judges the email to be phishing or legitimate. The binary variable $z^n_m(\theta) \in \{z^{co}, z^{wr}\}$ represents whether the judgment is correct (denoted by z^{co}) or not (denoted by z^{wr}). We can reshape the two-dimension index (m, n) as a one-dimension index $\hat{n}$ and rewrite $z^n_m(\theta)$ as $z_{\hat{n}}(\theta)$. Once these users have judged in total of N^{bo} emails, we define the following metric $c^{ac} \in \mathcal{C} : \Theta^d \mapsto [0, 1]$ to evaluate the *accuracy* of phishing recognition, i.e.,

$$c^{ac}(\theta) := \frac{1}{N^{bo}} \sum_{\hat{n}=1}^{N^{bo}} |\mathbf{1}_{\{z_{\hat{n}}(\theta)=z^{co}\}}|, \forall \theta \in \Theta^d. \tag{11.3}$$

The goal is to find the optimal hyperparameter $\theta^* \in \Theta^d$ to maximize the accuracy of phishing identification; i.e., $\theta^* \in \arg\max_{\theta\in\Theta^d} c^{ac}(\theta)$. However, we cannot know the value of $c^{ac}(\theta)$ for a $\theta \in \Theta^d$ a priori until we implement this hyperparameter θ in the attention enhancement mechanism. The implemented hyperparameter affects the adaptive visual-aid generation that changes the user's attention and the anti-phishing performance metric $c^{ac}(\theta)$. Since the experimental evaluation at a given $\theta \in \Theta^d$ is time-consuming, we present an algorithm in Section 11.2.2 to determine how to choose and update the hyperparameter to maximize the detection accuracy.

11.2.2 Efficient Hyperparameter Tuning

We illustrate the meta-adaptation (i.e., step 8 in Fig. 11.1) in Section 11.2.2. As illustrated in Fig. 11.6, we refer to the duration of every N^{bo} security decisions as a *tuning stage*. Consider a time and budget limit that restricts us to conduct L tuning stages in total. We denote θ_l as the hyperparameter at the l-th tuning stage where $l \in \mathcal{L} := \{1, 2, \cdots, L\}$. Since each user's email inspection time is different, each tuning stage can contain different numbers of generation stages.

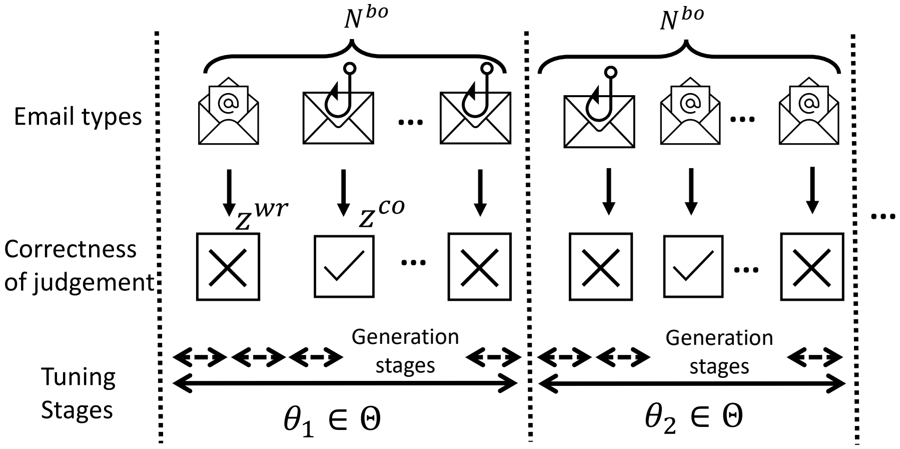


Figure 11.6: Hyperparameter tuning based on the user's phishing recognition. Each tuning stage consists of N^{bo} emails and contains several generation stages.

To find the optimal hyperparameter $\theta^* \in \Theta^d$ within L trials is challenging. The empirical methods such as a naive grid search and random search over $\Theta^d \subset \mathbb{R}^d$ become inefficient when $d > 1$. Bayesian Optimization (BO) [53] provides a systematic way to update the hyperparameter and balance between exploration and exploitation. BO consists of a Bayesian statistical model of the objective function $c^{ac} \in \mathcal{C}$ and an acquisition function for deciding the hyperparameter to implement at the next tuning stage. The statistical model of $c^{ac} \in \mathcal{C}$ is a Gaussian process $\mathcal{N}(\mu^0, \Sigma^0)$ with a mean function $\mu^0(\theta) = \bar{\mu}^0$ and covariance function or kernel $\Sigma^0(\theta, \bar{\theta}) = \lambda^0 \cdot \exp(\sum_{i=1}^{d} \lambda^i(\theta^i - \bar{\theta}^i)^2)$ for all $\theta, \bar{\theta} \in \Theta^d$, where $\bar{\mu}^0$, λ^0 and $\lambda^i, i \in \{1, 2, \cdots, d\}$, are parameters of the kernel. The kernel Σ^0 is required to be positive semi-definite and has the property that the points closer in the input space are more strongly correlated. For any $l \in \mathcal{L}$, we define three shorthand notations $\mu^0(\theta_{1:l}) := [\mu^0(\theta_1), \cdots, \mu^0(\theta_l)]$, $c^{ac}(\theta_{1:l}) := [c^{ac}(\theta_1), \cdots, c^{ac}(\theta_l)]$, and

$$\Sigma^0(\theta_{1:l}, \theta_{1:l}) := \begin{bmatrix} \Sigma^0(\theta_1, \theta_1) & \cdots & \Sigma^0(\theta_1, \theta_l) \\ \vdots & \ddots & \vdots \\ \Sigma^0(\theta_l, \theta_1) & \cdots & \Sigma^0(\theta_l, \theta_l) \end{bmatrix}.$$

Then, the evaluation vector of $l \in \mathcal{L}$ elements is assumed to be multivariate Gaussian distributed, i.e., $c^{ac}(\theta_{1:l}) \sim \mathcal{N}(\mu^0(\theta_{1:l}), \Sigma^0(\theta_{1:l}, \theta_{1:l}))$. Conditioned on the values of $\theta_{1:l}$, we can infer the value of $c^{ac}(\theta)$ at any other $\theta \in \Theta \setminus \{\theta_{l'}\}_{l' \in \{1, \cdots, l\}}$ by Bayesian rule, i.e.,

$$c^{ac}(\theta) | c^{ac}(\theta_{1:l}) \sim \mathcal{N}(\mu^n(\theta), (\Sigma^n(\theta))^2), \tag{11.4}$$

where $\mu^n(\theta) = \Sigma^0(\theta, \theta_{1:l}) \cdot \Sigma^0(\theta_{1:l}, \theta_{1:l})^{-1} \cdot (c^{ac}(\theta_{1:l}) - \mu^0(\theta_{1:l})) + \mu^0(\theta)$ and $(\Sigma^n(\theta))^2 = \Sigma^0(\theta, \theta) - \Sigma^0(\theta, \theta_{1:l}) \cdot \Sigma^0(\theta, \theta_{1:l})^{-1} \cdot \Sigma^0(\theta_{1:l}, \theta)$.

We adopt *expected improvement* as the acquisition function. Define $c_l^* := \max_{l' \in \{1, \cdots, l\}} c^{ac}(\theta_{l'})$ as the optimal evaluation among the first l evaluations and a shorthand notation $(c^{ac}(\theta) - c_l^*)^+ := \max\{c^{ac}(\theta) - c_l^*, 0\}$. For any $l \in \mathcal{L}$, we define $\mathbb{E}_l[\cdot] := \mathbb{E}[\cdot | c^{ac}(\theta_{1:l})]$ as the expectation taken under the posterior distribution of $c^{ac}(\theta)$ conditioned on the values of l evaluations $c^{ac}(\theta_{1:l})$. Then, the expected improvement is $\mathrm{EI}_l(\theta) := \mathbb{E}_l[(c^{ac}(\theta) - c_l^*)^+]$. The hyperparameter at the next tuning stage is chosen to maximize the expected improvement at the current stage, i.e,

$$\theta_{l+1} \in \arg\max_{\theta \in \Theta^d} \mathrm{EI}_l(\theta). \tag{11.5}$$

The expected improvement can be evaluated in a closed form, and (11.5) can be computed inexpensively by gradient methods [53].

At the first $L^0 \in \{1, 2, \cdots, L\}$ tuning stages, we choose the hyperparameter $\theta_l, l \in \{1, 2, \cdots, L^0\}$ uniformly from Θ^d. We can use the evaluation results $c^{ac}(\theta_l), l \in \{1, 2, \cdots, L^0\}$, to determine the parameters $\bar{\mu}^0, \lambda^0$, and $\lambda^i, i \in \{1, 2, \cdots, d\}$, by Maximum Likelihood Estimation (MLE); i.e., we determine the values of these parameters so that they maximize the likelihood of observing the vector $[c^{ac}(\theta_{1:L^0})]$. For the remaining $L - L^0$ tuning stages, we choose $\theta_l, l \in \{L^0, L^0 + 1, \cdots, L\}$, in sequence as summarized in Algorithm 10.

Algorithm 10: Hyperparameter tuning via BO.

```
Implement the initial $L^0$ evaluations $c^{ac}(\theta_l), l \in \{1, 2, \cdots, L^0\}$;
Place a Gaussian process prior on $c^{ac} \in \mathcal{C}$, i.e.,
  $c^{ac}(\theta_{1:L^0}) \sim \mathcal{N}(\mu^0(\theta_{1:L^0}), \Sigma^0(\theta_{1:L^0}, \theta_{1:L^0}))$;
for $l \leftarrow L^0$ to $L$ do
    Obtain the posterior distribution of $c^{ac}(\theta)$ in (11.4) based on the existing $l$
      evaluations;
    Compute $\mathrm{EI}_l(\theta), \forall \theta \in \Theta^d$, based on the posterior distribution;
    Determine $\theta_{l+1}$ via (11.5);
    Implement $\theta_{l+1}$ at the next tuning stage $l + 1$ to evaluate $c^{ac}(\theta_{l+1})$;
end
Return the maximized value of all observed samples
  $\theta^* \in \arg\max_{\theta_l \in \{\theta_1, \cdots, \theta_L\}} c^{ac}(\theta_l)$;
```

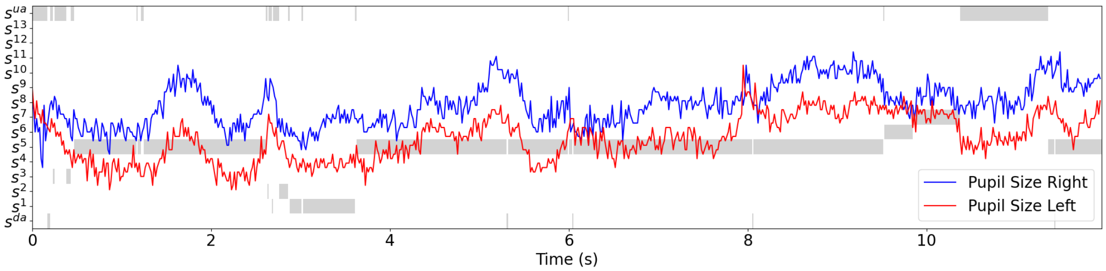


Figure 11.7: Gaze locations and pupil sizes collected in one trail of the data set. The grey squares illustrate the transition of 15 visual states. The red and blue lines represent the variations of the participant's left and right pupil sizes, respectively, as he reads the email. The x-axis represents the time of the email inspection.

11.3 Case Study

In this case study, we verify the effectiveness of ADVERT via a data set collected from human subject experiments conducted at New York University [32]. We elaborate on the experiment setup and the data processing procedure in Section 11.3.1. Based on the features obtained from the data set, we generate synthetic data under adaptive visual aids to demonstrate the proposed attention enhancement mechanism and the phishing prevention mechanism in Section 11.3.2 and 11.3.3, respectively.

11.3.1 Experiment Setting and Data Processing

The data set involves $M = 160$ undergraduate students ($n_{\text{White}} = 27$, $n_{\text{Black}} = 19$, $n_{\text{Asian}} = 64$, $n_{\text{Hispanic/Latinx}} = 17$, $n_{\text{other}} = 33$) who are asked to vet $N = 12$ different emails (e.g., the email of NYU friends network in Fig. 11.2) separately and then give a rating of how likely they would take actions solicited in the emails (e.g., maintain membership in Fig. 11.2). When presented to different participants, each email is described as either posing a cyber threat or risk-free legitimate opportunities to investigate how the above description affects the participants' phishing recognition.

While the participants vet the emails, the Tobii Pro T60XL eye-tracking monitor records their eye locations on a 1920×1200 resolution screen and the current pupil diameters of both eyes with a sampling rate of 60Hz. Fig. 11.7 illustrates the pupil sizes of left and right eyes in red and blue, respectively. At different times, the average of the pupil diameters (resp. gaze locations) of the right and left eyes represent the pupil size (resp. gaze location). Since the covert eye-tracking system does not require head-mounted equipment or chinrests, the tracking can occur without the participants' awareness. We refer the reader to the supplement materials of [32] for the survey data and the details of the experimental procedure.

Estimate Concentration Scores and Decay Rates based on Pupil Sizes

Empirical works in [107, 110] have demonstrated that pupils dilate as a consequence of attentional efforts. Building on the findings, we assume that the average pupil diameters of both eyes at time t of the generation stage $k \in \mathcal{K}_m^n$ is approximately proportional to the participant's attention level $\frac{dv_k}{dt}(t)$ at time t. We obtain the benchmark values of $r^{co}(s), \alpha(s), \forall s \in \mathcal{S}$, in Table 11.1 by minimizing the mean square error between the CAL in Section 11.1.3 and the cumulative pupil size through global optimization methods such as Simulated Annealing (SA) [216]. The results in Table 11.1 corroborate that the main content AoI $s^5 \in \mathcal{S}$ has the highest concentration score and the lowest decay rate.

Synthetic VS Trajectory Generation under Visual Aids

In the case study, we consider $I = 13$ AoIs. The sample email in Fig. 11.2 illustrates the first 12 AoIs and the 13-th AoI is on the email attachment. For visual

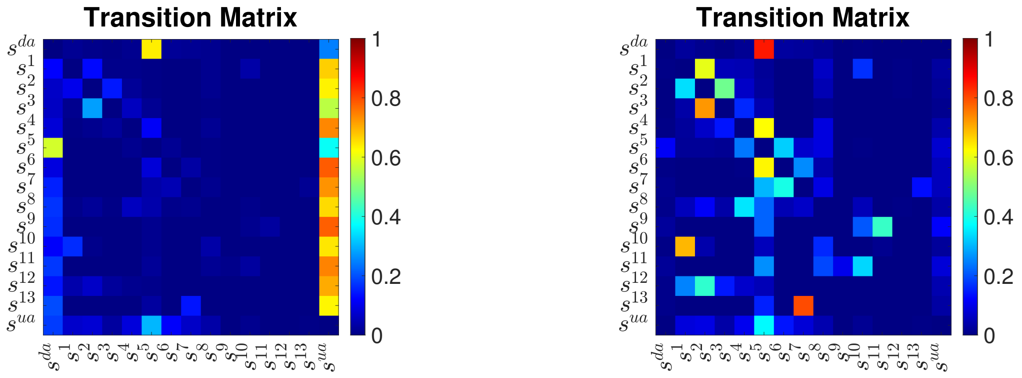
AoIs	Meaning	$r^{co}(s^i)$	$\alpha(s^i)$
s^1	Title	9.48	2.17
s^2	Sender	3.55	4.04
s^3	Receiver	7.62	0.22
s^4	Salutation	13.76	0.57
s^5	Main Content	21.05	0.16
s^6	URL	7.84	10.90
s^7	Signature	6.47	5.46
s^8	Logo	6.44	5.16
s^9	Print& Share	4.86	13.91
s^{10}	Time	3.81	6.68
s^{11}	Bookmark& Forward	7.34	2.19
s^{12}	Profile	7.26	2.02
s^{13}	Attachment	4.74	3.46

Table 11.1: The concentration score $r^{co}(s^i)$ and decay rate $\alpha(s^i)$ for $I = 13$ AoIs.

aid $a \in \mathcal{A}$, we denote $P^{i,j}(a)$ as the probability of attention arriving at visual state $j \in \mathcal{S}$ from visual state $i \in \mathcal{S}$ and $\phi^i(a)$ as the average sojourn time at visual state $i \in \mathcal{S}$. We specify the participants' VS transition trajectory $[s_t]_{t\in[0,T_m^n]}, \forall m \in \mathcal{M}, n \in \mathcal{N}$, under visual-aid generation policy $\sigma \in \Sigma$ as a semi-Markov transition process with probability transition matrix $P(\sigma(i)) := [P^{i,j}(\sigma(i))]_{i,j\in\mathcal{S}}$ and exponential sojourn distribution of the scale parameter $\phi(\sigma(i)) := [\phi^i(\sigma(i))]_{i\in\mathcal{S}}$.

In particular, we consider a binary set of visual aid $\mathcal{A} = \{a^N, a^Y\}$, where a^N represents the benchmark case without visual aids and a^Y represents the visual aid of highlighting the entire email contents. Based on the VS transition trajectory from the data set, we obtain the probability transition matrix $P(a^N)$ and the sojourn distribution parameter $\phi(a^N)$ under the benchmark case a^N. The transition matrix $P(a^Y)$ and sojourn distribution $\phi(a^Y)$ under visual aid a^Y modify $P(a^N)$ and $\phi(a^N)$ based on the following observations. On the one hand, the visual aid a^Y decreases $P^{i,s^{ua}}(a^Y), P^{i,s^{da}}(a^Y), \forall i \in \mathcal{S}$; i.e., the participants will be guided by the visual aid to pay more frequent attention to the AoIs than the uninformative and distraction areas. On the other hand, the visual aid a^Y decreases $\phi_{s^5}(a^Y)$; i.e., the persistent highlighting makes participants weary and reduces their attention spans on the email's main content.

We illustrate $P(a^N)$ and $P(a^Y)$ using heat maps in Fig. 11.8a and Fig. 11.8b, respectively. In Fig. 11.9, we illustrate an exemplary transition trajectory of $I + 2$ visual states under a^N and a^Y in blue and red, respectively. The trajectory corroborates that participants under visual aid a^Y incline to pay attention to AoIs yet have less sustained attention. To accurately quantify the impact of the visual aid on the VS transition depends on many factors [77], including the graphic design, the human subject, and the cognitive task. Here, we provide one potential estimation of the impact based on the human experiments to illustrate the implementation procedure and the effectiveness of the ADVERT framework.



(a) Under visual aid a^N.

(b) Under visual aid a^Y.

Figure 11.8: Heat maps of the transition matrices $P(a), a \in \mathcal{A}$. The row and the column represent the source and the destination of the $I + 2$ visual states, respectively. Under a^Y, the participants tend to pay attention to AoIs rather than the uninformative and distraction areas.

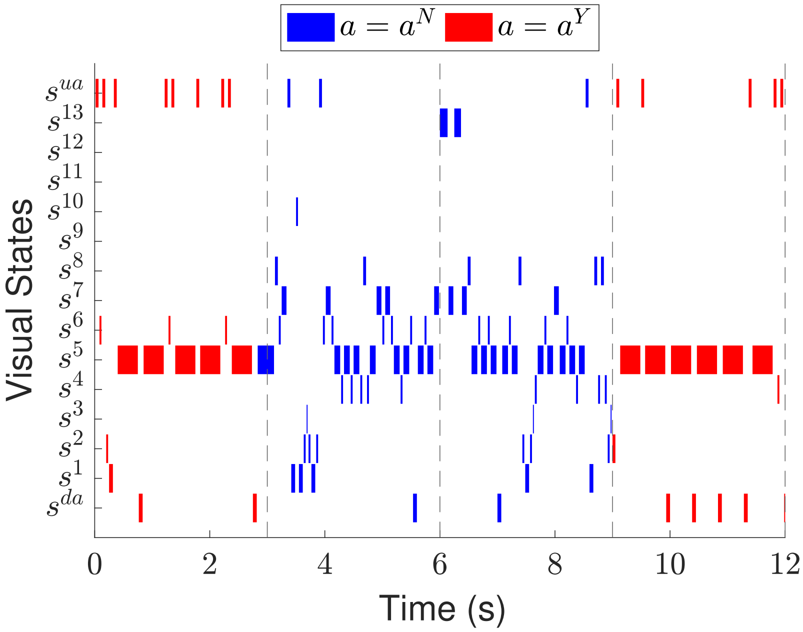


Figure 11.9: The VS transition trajectory when the visual aids in four generation stages are a^Y, a^N, a^N, and a^Y, respectively. The inspection lasts for 12 seconds and the period length T^{pl} is 3 seconds.

11.3.2 Validation of Attention Enhancement Mechanism

Based on the benchmark attention score in Section 11.3.1, Fig. 11.10 illustrates the CAL of the exemplary VS transition trajectory shown in Fig. 11.9. Here, we consider $X = 2$ attention states $\mathcal{X} = \{x^H, x^L\}$ with *attentive state* x^H and *inattentive state* x^L. Define $X^{at} \in \mathbb{R}$ as the *attention threshold*. If the AAL at generation stage $k \in \mathcal{K}_m^n$ is higher (resp. lower) than the attention threshold, i.e., $\bar{v}_k \geq X^{at}$ (resp. $\bar{v}_k \leq X^{at}$), then the attention state $x_k \in \mathcal{X}$ at generation k is the attentive state x^H (resp. inattentive state x^L). Fig. 11.11 further shows the impact of visual aids a^N and a^Y on the AAL in red and blue, respectively. The figure demonstrates that a^Y can increase the mean of AAL yet increase its variance.

In Algorithm 11, we present the Q-learning process for participant $m \in \mathcal{M}$ who

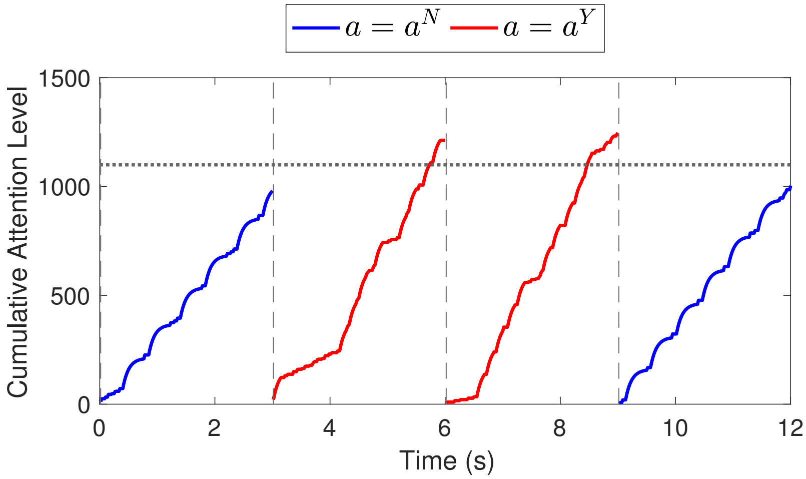


Figure 11.10: The CAL of the exemplary VS transition trajectory shown in Fig. 11.9. The horizontal dotted line represents the attention threshold X^{at}. The visual aids in four generation stages are a^Y, a^N, a^N, and a^Y, respectively, and the resulting attention states are x^L, x^H, x^H, and x^L, respectively.

read email $n \in \mathcal{N}$ for T_m^n seconds. Define $\eta_k(x,a)$ as the total number of visits to attention state $x \in \mathcal{X}$ and visual aid $a \in \mathcal{A}$ up to generation stage k. Then, we choose the learning rate $\gamma_k(x_k,a_k) = \frac{\eta^0}{\eta_k(x,a)-1+\eta^0}$ for all $x_k \in \mathcal{X}, a_k \in \mathcal{A}$ to guarantee the asymptotic convergence, where $\eta^0 \in (0,\infty)$ is a constant parameter.

Based on the benchmark data set of $M = 160$ participants who inspect $N = 12$ emails in Section 11.3.1, the inspection time $T_m^n, \forall m \in \mathbf{M}, n \in \mathcal{N}$, follows a *Burr distribution*; i.e., its cumulative distribution function is described by $F^{Burr}(t \mid \rho_1, \rho_2, \rho_3) = 1 - \frac{1}{(1+(t/\rho_1)^{\rho_2})^{\rho_3}}$ with the scale parameter $\rho_1 = 11.7$, and the shape parameters $\rho_2 = 62.5, \rho_3 = 0.04$. The average inspection time of $M \times N$ samples is 18.7 seconds. During T_m^n seconds of the email vetting process, the eye-tracking device records the participant's gaze locations, which leads to the VS transition trajectory. In Algorithm 11, we simulate the human email-reading process through the synthetic VS transition trajectory generated by the sufficient statistics $P(a_t)$

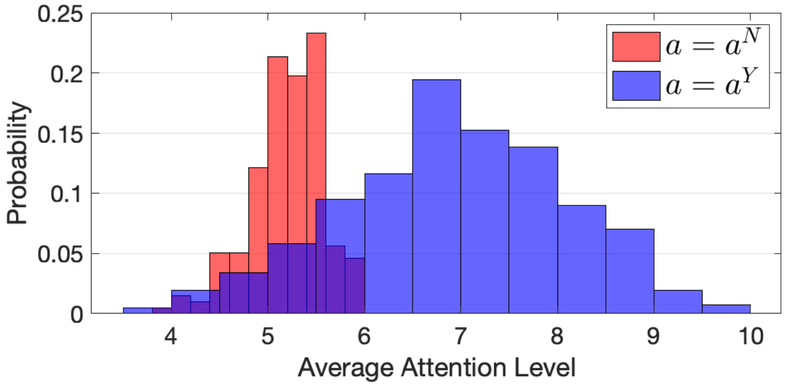


Figure 11.11: The normalized histogram of average attention level under visual aids a^N and a^Y in red and blue, respectively.

and $\phi(a_t)$. Every T^{pl} seconds, ADVERT updates the Q-matrix and the visual aid based on (11.2).

Following Section 11.1.4, we develop Algorithm 12 to illustrate the entire attention enhancement loop that involves the consolidation of the data set from $\bar{M} \in \{1, \cdots, M\}$ participants and $\bar{N} \in \{1, \cdots, N\}$ emails. After the participant $m \in \{1, \cdots, \bar{M}\}$ finishes reading the email $n \in \{1, \cdots, \bar{N}\}$, Algorithm 11 returns the Q-matrix and the attention state at the final generation stage K_m^n. These results then serve as the inputs for the next email inspection until N^{bo} emails have been inspected.

Based on Algorithm 12, we plot the entire Q-learning updates with $N^{bo} = 100$ emails in Fig. 11.12 that contains a total of 609 generations stages. The learning results show that the visual aid a^Y outweighs a^N for both attention states and should be persistently applied under the current setting.

Algorithm 11: [**Individual Adaptation**] Optimal visual-aid learning and attention enhancement for participant $m \in \mathcal{M}$ vetting email $n \in \mathcal{N}$.

```
119 Input: Initial Q-matrix $[Q_0(x,a)]_{x \in \mathcal{X}, a \in \mathcal{A}}$, initial attention state $x_0 \in \mathcal{X}$, the
      number of visits $\eta_k(x,a)$, and the hyperparameter $\theta = [X^{at}, T^{pl}]$;
120 Initialize time $t = 0$ and the inspection length $T_m^n$ based on the Burr distribution
      $F^{Burr}$;
121 Set the initial visual aid $a_0 \in \mathcal{A}$ based on the initial Q-matrix $Q_0$, the initial
      attention state $x_0$ and the $\epsilon_k$-greedy policy in Section 11.1.4;
122 while $t < T_m^n$ do
123     Obtain VS transition $s_t \in \mathcal{S}$ based on $P(a_t)$ and $\phi(a_t)$ (i.e., use synthetic
          visual data to achieve step 2 of Fig. 11.1);
124     Evaluate the CAL $v_k(t)$ based on $r^{co}, \alpha$ as shown in step 3 of Fig. 11.1;
125     if $t = kT^{pl}, k \in \mathbb{Z}^+$ then
126         if $\bar{v}_k \geq X^{at}$ (shown in step 4 of Fig. 11.1) then attentive attention state
              $x_k = x^H$ else inattentive attention state $x_k = x^L$;
127         Update Q-matrix $Q_k$ based on (11.2) as shown in step 5 of Fig. 11.1;
128         Implement the visual aid $a_k \in \mathcal{A}$ based on the current Q-matrix $Q_k$ and
              the $\epsilon_k$-greedy policy (i.e., step 6 of Fig. 11.1);
129         if $x_k = x, a_k = a$ then update the number of visits
              $\eta_{k+1}(x,a) \leftarrow \eta_k(x,a) + 1$;
130         Output the number of updates $K_m^n \leftarrow k$;
131     end
132 end
133 Implement the pre-trained neural network in Section 11.3.3 to estimate whether
      participant $m$ has made the correct judgment concerning email $n$, i.e.,
      $z_m^n(\theta) \in \{z^{co}, z^{wr}\}$ (i.e., use synthetic decision data to achieve step 7 of Fig.
      11.1);
134 Return: Q-matrix $[Q_{K_m^n}(x,a)]_{x \in \mathcal{X}, a \in \mathcal{A}}$, final attention state $x_{K_m^n} \in \mathcal{X}$, number of
      visits $\eta_{K_m^n}(x,a)$, and $z_m^n(\theta)$;
```

11.3.3 Validation of Phishing Prevention Mechanism

After we obtain a participant's synthetic response (characterized by his VS transition trajectory) under the adaptive visual aids, we apply a pre-trained neural network to estimate whether the participant has made a correct judgment as shown in line 24 of Algorithm 11. In Section 11.3.3, we elaborate on the training process of the neural network based on the data set used in Section 11.3.1. We apply the Bayesian optimization in Algorithm 10 to evaluate the accuracy metric $c^{ac} \in \mathcal{C}$ as

Algorithm 12: [**Population Adaptation**] Optimal visual-aid learning through a consolidated data set of $\bar{M} \in \{1, \cdots, M\}$ participants vetting $\bar{N} \in \{1, \cdots, N\}$ emails.

```
Input: Hyperparameter $\theta = [X^{at}, T^{pl}]$;
Initialize Q-matrix $[Q_0(x, a)]_{x \in \mathcal{X}, a \in \mathcal{A}}$ as a zero matrix,
  $\eta_0(x, a) = 0, \forall x \in \mathcal{X}, a \in \mathcal{A}$, and initial attention state $x_0 \in \mathcal{X}$;
for participant $m \in \{1, \cdots, \bar{M}\}$ vetting email $n \in \{1, \cdots, \bar{N}\}$ do
    Implement Algorithm 11 with the inputs of $[Q_0(x, a)]_{x \in \mathcal{X}, a \in \mathcal{A}}$, $x_0 \in \mathcal{X}$, and
      $\eta_0(x, a)$;
    Save the outputs of $[Q_{K_m^n}(x, a)]_{x \in \mathcal{X}, a \in \mathcal{A}}$, $x_{K_m^n} \in \mathcal{X}$, $\eta_{K_m^n}(x, a)$, and $z_m^n(\theta)$;
    Cascade the outputs to the inputs of the next loop: $Q_0 \leftarrow Q_{K_m^n}$, $x_0 \leftarrow x_{K_m^n}$,
      and $\eta_0 \leftarrow \eta_{K_m^n}$;
end
Return: the accuracy metric $c^{ac}(\theta)$ based on (11.3);
```

illustrated in step 8 of Fig. 11.1. In Section 11.3.3, we show the results.

Neural Network

In this case study, we regard the majority choice of the $M = 160$ participants as the email's true label. Without visual aids, these participants achieve an accuracy of 74.6% on average. Under the assumption that the hyperparameters affect the participants' phishing recognition only through their VS transitions, we construct a neural network with an LSTM layer, a dropout layer, and a fully-connected layer to establish the relationship from the sequence of VS transition trajectory $[s_t]_{t \in T_m^n}$ to the label of judgment correctness $z_m^n \in \{z^{co}, z^{wr}\}$. We split the entire trials of the data set into 1113 training data and 128 test data. The trained neural network achieves a sensitivity of 0.89, a specificity of 0.21, an f1-score of 0.73, and an accuracy of 0.61.

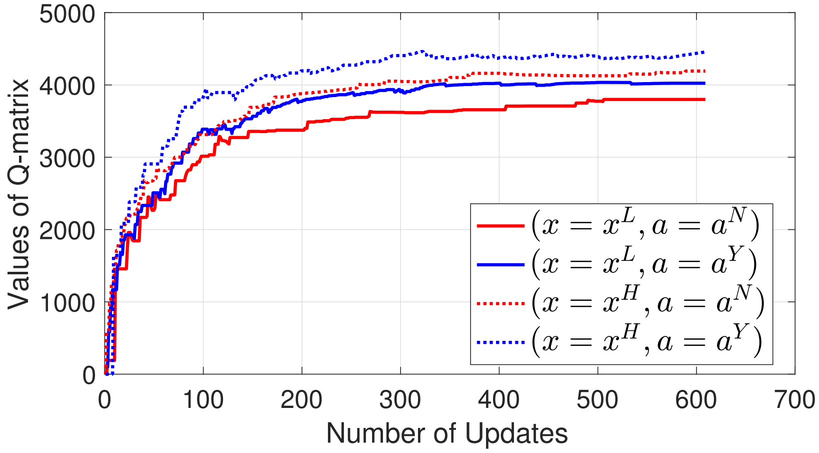


Figure 11.12: The Q-learning updates under hyperparameters $X^{at} = 5.56$ and $T^{pl} = 3$ seconds. The red and blue lines represent the Q-matrix values under visual aids a^N and a^Y, respectively. The solid and dashed lines represent the Q-matrix values under attention states x^L and x^H, respectively.

Bayesian Optimization Results

As explained in Section 11.2, for each different application scenario, a meta optimization of the accuracy metric $c^{ac}(X^{at}, T^{pl})$ is required to find the optimal attention threshold X^{at} and the period length T^{pl} for visual-aid generation. To obtain the value of $c^{ac}(X^{at}, T^{pl})$ under different values of the hyperparameter $\theta = [X^{at}, T^{pl}]$, we need to implement the hyperparameter in Algorithm 12 and repeat for n^{rp} times to reduce the noise. Thus, the evaluation is costly and Bayesian optimization in Algorithm 10 is a favorable method to achieve the meta optimization. We illustrate the Bayesian optimization for $L = 60$ tuning stages in Fig. 11.13. Each blue point represents the average value of $c^{ac}(X^{at}, T^{pl})$ over $n^{rp} = 20$ repeated samples under the hyperparameter $\theta = [X^{at}, T^{pl}]$. Based on the estimated Gaussian model in red, we find that the attention threshold $X^{at} \in [1, 33]$ has a small impact on phishing recognition while the period length $T^{pl} \in [60, 600]$ has a periodic impact on phishing recognition. The optimal hyperparameters for phishing prevention are

$X^{at,*} = 8.8347$ and $T^{pl,*} = 6.63$ seconds.

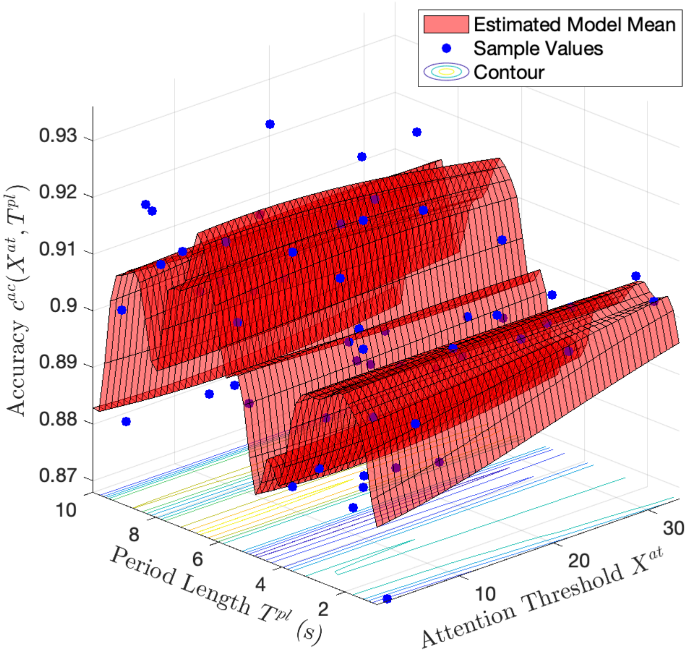


Figure 11.13: The estimated Gaussian model of the objective function $c^{ac}(\theta)$ concerning the hyperparameter $\theta = [X^{at}, T^{pl}]$ in red with its contour on the bottom. The blue points represent the sample values of 60 trials.

We illustrate the temporal procedure of Bayesian optimization of $L = 60$ tuning stages in Fig. 11.14. As we increase the number of tuning stages to conduct more trials and obtain more samples, the maximized value of the accuracy metric $c^{ac} \in \mathcal{C}$ monotonously increases as shown in red. The blue line and its error bar represent the mean and variances of the sample values at each tuning stage, respectively. Throughout the $L = 60$ tuning stages, the variance remains small, which indicates that ADVERT is *robust* to the noise of human attention and decision processes.

Compared to the benchmark accuracy of 74.6% without visual aids, participants with visual aid achieve the accuracy of a minimum of 86% under all 60 trials of

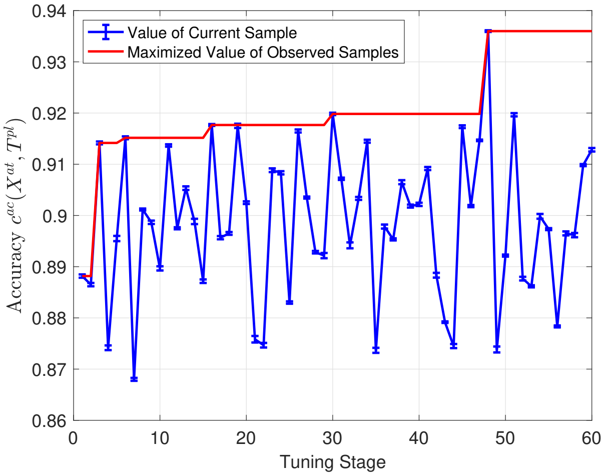


Figure 11.14: Accuracy metric $c^{ac}(X^{at}, T^{pl})$ at $L = 60$ tuning stages. The blue line and its error bar represent the mean value of the samples and their variances, respectively. The red line represents the maximized value of the observed samples up to the current tuning stage.

different hyperparameters. The above accuracy improvement corroborates that the ADVERT's attention enhancement mechanism illustrated in blue of Fig. 11.1 effectively serves as a stepping stone to facilitate phishing recognition. The results shown in the blue line further corroborate the efficiency of the ADVERT's phishing prevention mechanism illustrated in orange of Fig. 11.1; i.e., in less than 50 tuning stages, we manage to improve the accuracy of phishing recognition from 86.8% to 93.7%. Besides, the largest accuracy improvement (from 88.7% to 91.4%) happens within the first 3 tuning stages. Thus, if we have to reduce the number of tuning stages due to budget limits, ADVERT can still achieve a sufficient improvement of phishing recognition accuracy.

Chapter 12

RADAMS: Alert and Attention Management Strategies against Information-DoS Attacks

In Chapter 11, we have addressed the challenge of *reactive attentional attacks*, where stealthy attackers exploit inattention to evade attention. In Chapter 12, we address the challenge of *proactive attentional attacks* that aim to overload human attention. We refer to this new class of attacks as the Informational Denial-of-Service (IDoS) attacks.

IDoS is no stranger to us in this age of information explosion. We are commonly overloaded with terabytes of unprocessed data or manipulated information on online media. However, the targeted IDoS attacks on specific groups of people, e.g., security guards, operators at the nuclear power plant, and network administrators, can pose serious threats to lifeline infrastructures and systems. The attacker customizes attack strategies to targeted individuals or organizations to quickly and maximally deplete their human cognitive resources. As a result, common methods (e.g., set tiered alert priorities) to mitigate alert fatigue are insufficient under these targeted and intelligent attacks that generate massive feints strategically. There is a need to understand this phenomenon, quantify its consequence and risks, and develop new mitigation methods. In this work, we establish a probabilistic model to formalize the definition of IDoS attacks, evaluate their severity levels, and assess the induced cyber risks. The model captures the interaction among attackers, human operators, and assistive technologies as highlighted by the orange, green, and blue backgrounds, respectively, in Fig. 12.1.

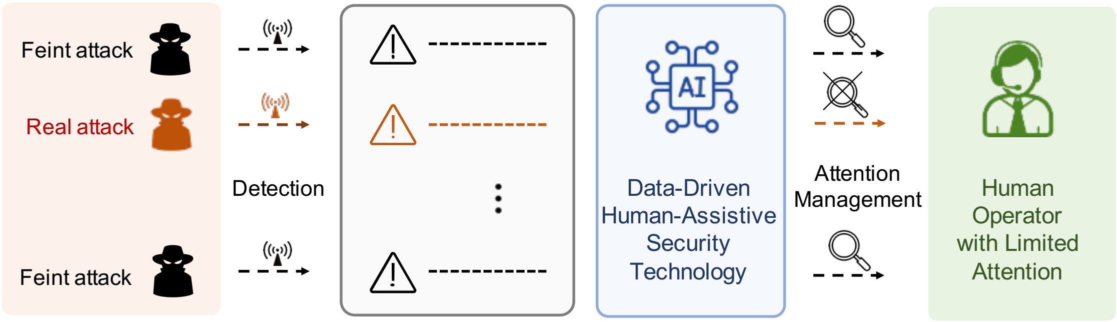

Feint attack
Real attack
Feint attack
Detection
Data-Driven Human-Assistive Security Technology
Attention Management
Human Operator with Limited Attention

Figure 12.1: Interaction among IDoS attacks, human operators, and assistive technologies.

Attackers generate feints and real attacks that trigger alerts of Intrusion Detection System (IDS). Due to the detection imperfectness, human operators need to inspect these alerts in detail to determine the attacks' types, i.e., feint or real, and take responsive security decisions. The accuracy of the security decisions depends on the inspection time and the operator's sustained attention without distractions. The large volume of feints exerts an additional cognitive load on each human operator and makes it hard to focus on each alert, which can significantly decrease the accuracy of his security decisions and increase cyber risks. Accepting the innate human vulnerability, we aim to develop assistive technologies to compensate for the human attention limitation. Evidence from the cognitive load theory [221] has shown that divided attention to multiple stimuli can degrade the performance and cost more time than responding to these stimuli in sequence. Hence, we design the *Attention Management* (AM) strategies to intentionally make some alerts inconspicuous so that the human operator can focus on the other alerts and finish the inspection with less time and higher accuracy. We further define risk measures to evaluate the inspection results, which serves as the stepping stone to designing adaptive AM strategies to mitigate attacks induced by human vulnerabilities.

12.1 High-Level Abstraction and Motivating Example

As shown in Fig. 12.2, there is an analogy between the DoS attacks in communication networks and the IDoS attacks in the human-in-the-loop systems. Both of them achieve their attack goals by exhausting the limited resources. DoS attacks happen when the attacker generates a large number of superfluous requests to deplete the computing resource of the targeted machine and prevent the fulfillment of legitimate services. Analogously, IDoS attacks create a large amount of unprocessed information to deplete cognitive resources of human operators and prevent them from acquiring the knowledge contained in the information. We list several assailable cognitive resources under IDoS attacks as follows.

- **Attention**: Paying sustained attention to acquire proper information is costly. From an economic perspective, inattention occurs when the cost of information acquisition is lower than the attention cost measured by the information entropy [198]. IDoS attacks generate feints to distract the human

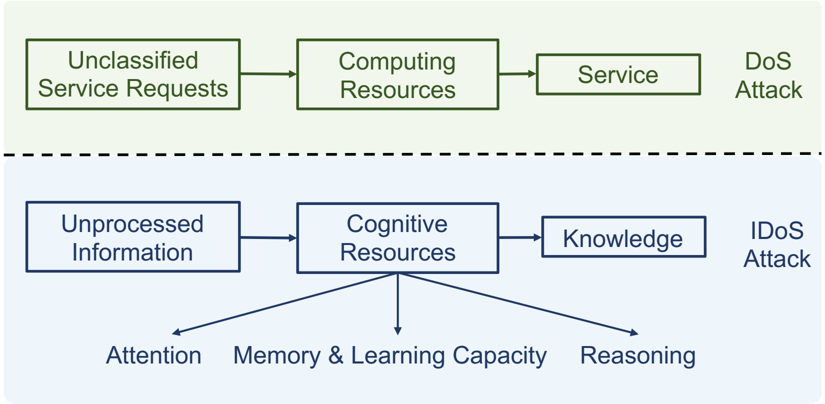


Figure 12.2: The service request fulfillment process under DoS attacks and the information processing flows under IDoS attacks in green and blue backgrounds, respectively.

from the right information. An excessive number of feints prohibit the human from process any information.

- **Memory and Learning Capacity**: Humans have limited memory and learning capacity. Humans cannot remember the details or learn new things if there is an information overload [221].

- **Reasoning**: Human decision-making consumes a large amount of energy, which is one of the reasons why we have two modes of thought [108] ('system 1' thinking is fast, instinctive, and emotional; while 'system 2' thinking is slower and more logical). IDoS attacks can exert a heavy cognitive load to prevent humans from deliberative decisions that use the 'system 2' thinking. Moreover, evidence shows the *paradox of choice* [190]; i.e., rich choices can bring anxiety and prevent humans from making any decisions.

When these cognitive resources are exhausted, the information cannot be processed correctly and timely and serves as noise that leads to *alert fatigue* [12]. We use operators in the control room of nuclear power plants as a stylized example to illustrate the consequences of IDoS attacks and motivate the need for the security technology to assist human operators against IDoS attacks. In Fig. 12.3, a monitor screen contains meters that show the real-time readings of the temperature, pressure, and flow rate in a nuclear power plant. Based on the pre-defined generation rules, warnings and messages pop up at different locations. Due to the complexity of the nuclear control system, the inspection of these alerts consumes the operator's time and cognitive resources. The attempt to inspect all alerts and the constant switching among them can lead to missed detection and erroneous behaviors. If the alerts are generated strategically by attacks, they may further mislead humans to take actions in the attacker's favor; e.g., focusing on feints and ignoring the real attacks that hide among feints.

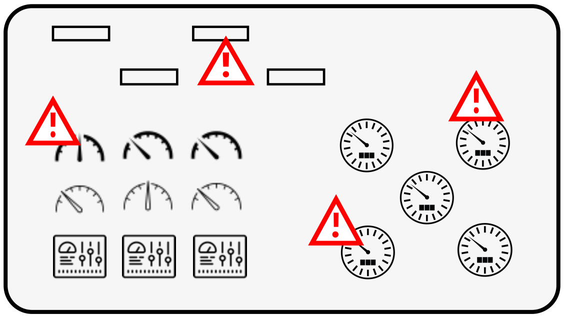


Figure 12.3: A stylized example of the monitor screen for operators in the control room of nuclear power plants. The red triangles represent warnings and security messages.

One way to mitigate IDoS attacks is to train the operators or human users to deal with the information overload and remain vigilant and productive under a heavy cognitive load. However, attentional training can be time-consuming and the effectiveness is not guaranteed. The second method is to recruit more human operators to share the information load. It would require the coordination of the operator team and can incur additional costs of human resources. The third method is to develop assistive technologies to rank and filter the information to alleviate the cognitive load of human operators. It would leverage past experiences and data analytics to pinpoint and prioritize critical alerts for human operators to process. The first two methods aim to increase the capacity or the volume of the cognitive resources in Fig. 12.2. The third method pre-processes the information so that it adapts to the capacity and characteristics of cognitive resources.

In this work, we adopt the thrid method and develop a Resilient and Adaptive Data-driven alert and Attention Management Strategy (RADAMS) to protect Industrial Control Systems (ICSs) from IDoS attacks. We illustrate the overview diagram of Resilient and Adaptive Alert and Attention Management Strategy (RADAMS) in Fig. 12.4. RADAMS enriches the existing alert selection frameworks with the IDoS attack model, the human attention model, and the human-assistive security technology highlighted in red, green, and blue, respectively.

12.2 IDoS Attacks and Sequential Alert Arrivals

As illustrated in the first column of Fig. 12.4, after the IDoS attacker has generated feint and real attacks, the IDS monitors the readings from physical layers and log files from cyber layers and generates alerts according to the *generation rules*. Then, the alerts are sent to the SOC and a triage system automatically generates their category labels (e.g., the alerts' criticality) based on the *mapping rules*. The rules for alert generation and triage mapping are pre-defined and their designs are not the focus of this work.

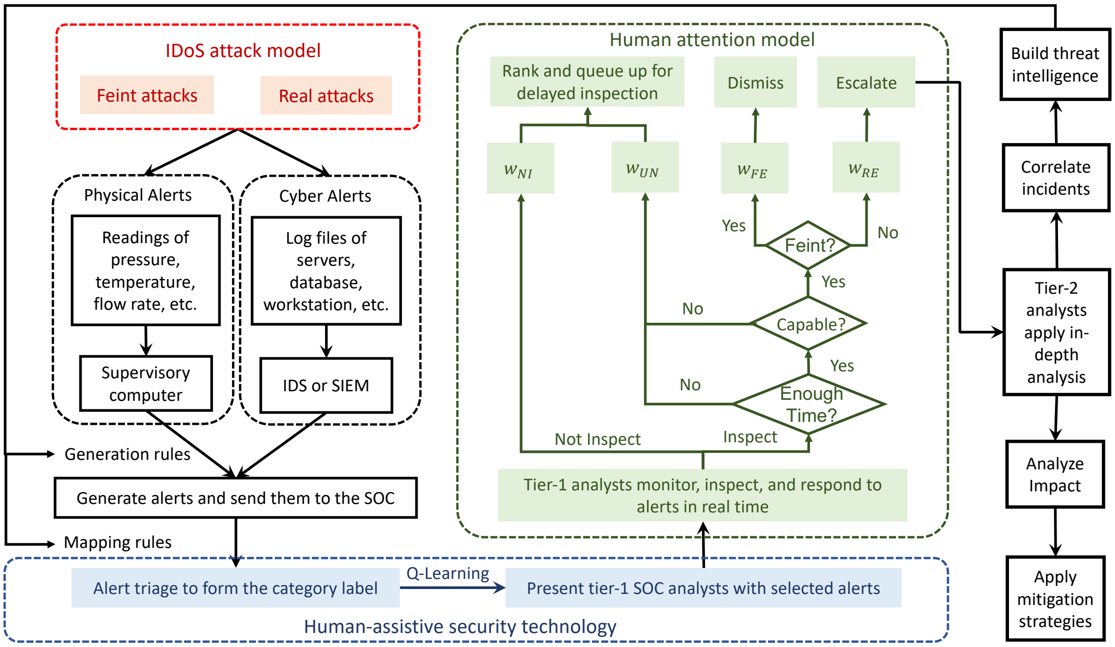

IDoS attack model
Feint attacks
Real attacks
Physical Alerts
Readings of pressure, temperature, flow rate, etc.
Supervisory computer
Cyber Alerts
Log files of servers, database, workstation, etc.
IDS or SIEM
Generation rules
Generate alerts and send them to the SOC
Mapping rules
Human attention model
Rank and queue up for delayed inspection
Dismiss
Escalate
w_{NI}
w_{UN}
w_{FE}
w_{RE}
Yes
Feint?
No
Yes
No
Capable?
Yes
No
Enough Time?
Not Inspect
Inspect
Tier-1 analysts monitor, inspect, and respond to alerts in real time
Alert triage to form the category label
Q-Learning
Present tier-1 SOC analysts with selected alerts
Human-assistive security technology
Build threat intelligence
Correlate incidents
Tier-2 analysts apply in-depth analysis
Analyze Impact
Apply mitigation strategies

Figure 12.4: The overview diagram of RADAMS against IDoS in ICS, which incorporates the IDoS attack model, human attention model, and the human-assistive security technology in the red, green, and blue boxes, respectively. The processes in black are not the focus of this work. The modern SOC adopts a hierarchical alert analysis process. The tier-1 SOC analysts, also referred to as the operators, are in charge of real-time alert monitoring and inspections. The tier-2 SOC analysts are in charge of the in-depth analysis.

12.2.1 Feint and Real Attacks of Heterogeneous Targets

After the initial intrusion, privilege escalation, and lateral movement, IDoS attackers can launch feint and real attacks sequentially as illustrated by the solid red arrows in Fig. 12.5. With a deliberate goal of triggering alerts, feint attacks require fewer resources to craft. Although feints have limited impacts on the target system, they aggravate the alert fatigue by depleting human attention resources and preventing human operators from a timely response to real attacks. For example, the attacker can attempt to access a database with wrong credentials intentionally, and in the meantime, gradually changes the temperature of the reactor of a nuclear power plant. The repeated log-in attempts trigger an excessive number of alerts so that the overloaded human operators fail to pay sustained attention and respond timely to the sensor alerts of the temperature deviation.

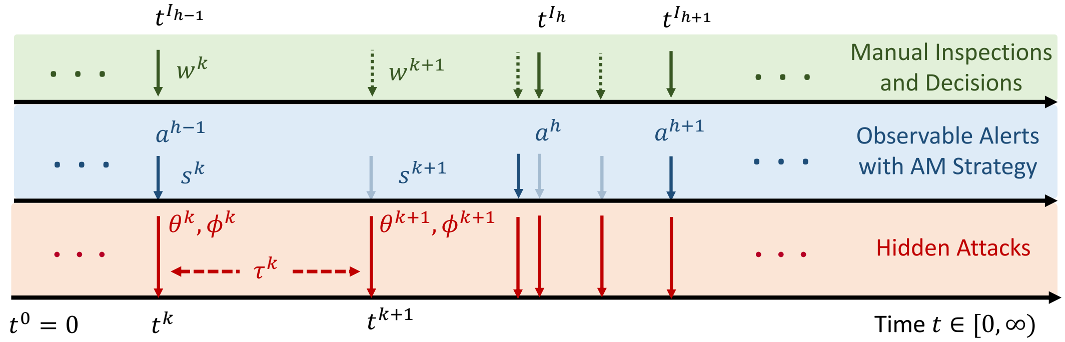


Figure 12.5: The timelines of an IDoS attack, alerts under AM strategies, and manual inspections are depicted in red, blue, and green, respectively. The inspection stage $h \in \mathbb{Z}^{0+}$ is equivalent to the attack stage $I_h \in \mathbb{Z}^{0+}$. The red arrows represent the sequential arrivals of feints and real attacks. The semi-transparent blue the dashed green arrows represent the de-emphasized alerts and the alerts without inspections, respectively.

We denote feint and real attacks as θ_{FE} and θ_{RE}, respectively, where $\Theta := \{\theta_{FE}, \theta_{RE}\}$ is the set of attacks' types. Each feint or real attack can target cyber components (e.g., servers, databases, and workstations) or physical components (e.g., sensors of pressure, temperature, and flow rate) in the ICS. We define Φ as the set of the potential attack targets. The stochastic arrival of these attacks is modeled as a Markov renewal process where $t^k, k \in \mathbb{Z}^{0+}$, is the time of the k-th arrival. We refer to the k-th attack equivalently as the attack at *attack stage* $k \in \mathbb{Z}^{0+}$ and let $\theta^k \in \Theta$ and $\phi^k \in \Phi$ be the attack's type and target at attack stage $k \in \mathbb{Z}^{0+}$, respectively. Define $\kappa_{AT} \in \mathcal{K}_{AT} : \Theta \times \Phi \times \Theta \times \Phi \mapsto [0, 1]$ as the transition kernel where $\kappa_{AT}(\theta^{k+1}, \phi^{k+1} | \theta^k, \phi^k)$ denotes the probability that the $(k + 1)$-th attack has type $\theta^{k+1} \in \Theta$ and target $\phi^{k+1} \in \Phi$ when the k-th attack has type $\theta^k \in \Theta$ and target $\phi^k \in \Phi$. The inter-arrival time $\tau^k := t^{k+1} - t^k$ is a continuous random variable with support $[0, \infty)$ and Probability Density Function (PDF) $z \in \mathcal{Z} : \Theta \times \Phi \times \Theta \times \Phi \mapsto \mathbb{R}^{0+}$ where $z(t|\theta^{k+1}, \phi^{k+1}, \theta^k, \phi^k)$ is the probability that the inter-arrival time is t when the attacks' types and targets at attack stage k and $k+1$ are θ^k, ϕ^k and θ^{k+1}, ϕ^{k+1}, respectively. The values of $\kappa_{AT} \in \mathcal{K}_{AT}$ and $z \in \mathcal{Z}$ are unknown to human operators and the designer of RADAMS. Attackers can adapt κ_{AT} and z to different ICS and alert inspection schemes to achieve the attack goals. We formally define IDoS attacks in Definition 38.

Definition 38. *An IDoS attack is a sequence of feint and real attacks of heterogeneous targets, which can be characterized by the* 4*-tuple* $(\Theta, \Phi, \mathcal{K}_{AT}, \mathcal{Z})$.

12.2.2 Alert Triage Process and System-Level Metrics

The alerts triggered by IDoS attacks contain *device-level* contextual information, including the software version, hardware parameters, existing vulnerabilities, and security patches. The alert triage process consists of rules that map the device-level information to the *system-level* metrics, which helps human operators make timely responses. Some essential metrics are listed as follows.

- **Source** $s_{SO} \in \mathcal{S}_{SO}$: The ICS sensors or the cyber components that the alerts are associated with.
- **Time Sensitivity** $s_{TS} \in \mathcal{S}_{TS}$: The length of time that the potential attack needs to achieve its attack goals.
- **Complexity** $s_{CO} \in \mathcal{S}_{CO}$: The degree of effort that a human operator takes to inspect the alert.

- **Susceptibility** $s_{SU} \in \mathcal{S}_{SU}$: The likelihood that the attack succeeds and inflicts damage on the protected system.
- **Criticality** $s_{CR} \in \mathcal{S}_{CR}$: The consequence or the impact of the attack's damage.

These alert metrics are observable to the human operator and the RADAMS designer, and they form the *category label* of an alert. We define the category label associated with the k-th alert as $s^k := (s^k_{SO}, s^k_{TS}, s^k_{CO}, s^k_{SU}, s^k_{CR}) \in \mathcal{S}$ where $\mathcal{S} := \mathcal{S}_{SO} \times \mathcal{S}_{TS} \times \mathcal{S}_{CO} \times \mathcal{S}_{SU} \times \mathcal{S}_{CR}$. The joint set $\mathcal{S}$ can be adapted to suit the organization's security practice. For example, we have $\mathcal{S}_{TS} = \emptyset$ if time sensitivity is unavailable or unimportant.

The alert triage process establishes a stochastic connection between the hidden types and targets of IDoS attacks and the observable category labels of the associated alerts. Let $o(s^k|\theta^k, \phi^k)$ be the probability of obtaining category label $s^k \in \mathcal{S}$ when the associated attack has type $\theta^k \in \Theta$ and target $\phi^k \in \Phi$. The revelation kernel o reflects the quality of the alert triage. For example, feints with lightweight resource consumption usually have a limited impact. Thus, a high-quality triage process should classify the associated alert as low criticality with a high probability. Letting $b(\theta^k, \phi^k)$ denote the probability that the k-th attack has type θ^k and target ϕ^k at the steady-state, we can compute the steady-state distribution b in closed form based on κ_{AT}. Then, the transition of category labels at different attack stages is also Markov and is represented by $\kappa_{CL} \in \mathcal{K}_{CL} : \mathcal{S} \times \mathcal{S} \mapsto [0, 1]$. We can compute $\kappa_{CL} = \frac{\Pr(s^{k+1}, s^k)}{\sum_{s^{k+1} \in \mathcal{S}} \Pr(s^{k+1}, s^k)}$ based on κ_{AT}, o, b, where $\Pr(s^{k+1}, s^k) = \sum_{\theta^k, \theta^{k+1} \in \Theta} \sum_{\phi^k, \phi^{k+1} \in \Phi} \kappa_{AT}(\theta^{k+1}, \phi^{k+1}|\theta^k, \phi^k) o(s^k|\theta^k, \phi^k) o(s^{k+1}|\theta^{k+1}, \phi^{k+1}) b(\theta^k, \phi^k)$.

In this work, we focus on the case where the IDS introduces the same delay between attacks and their triggered alerts. Since the sequences of attacks and alerts have a one-to-one mapping, we can consider zero delay time without loss of generality. Hence, the sequence of alerts associated with an IDoS attack $(\Theta, \Phi, \mathcal{K}_{AT}, \mathcal{Z})$ is also a Markov renewal process characterized by the 3-tuple $(\mathcal{S}, \mathcal{K}_{CL}, \mathcal{Z})$.

12.3 Attention Model under IDoS Attacks

An SOC typically adopts a hierarchical alert analysis [236]. The attention model in this section applies to the tier-1 SOC analysts, or the operators, who are in charge of monitoring, inspecting, and responding to alerts in real time. As illustrated by the green box in Fig. 12.4, the operators choose to inspect certain alerts, dismiss the feints, and escalate the real attacks to tier-2 SOC analysts for in-depth analysis. The in-depth analysis can last hours to months, during which the tier-2 analysts correlate incidents from different components in the ICS over long periods to build threat intelligence and analyze the impact. The threat intelligence is then incorporated to form and update the generation rules of the IDS and mapping rules of the triage process.

12.3.1 Alert Responses

Due to the high volume of alerts and the potential short-term surge arrivals, human operators cannot inspect all alerts in real time. The uninspected alerts receive an alert response w_{NI}. Whether the operator chooses to inspect an alert depends on the switching probability in Section 12.3.2.

When the operator inspects an alert, he can be distracted by the arrival of new alerts and switch to newly-arrived alerts without completing the current inspection.

We elaborate on the attention dynamics in Section 12.3.3. The alert with incomplete inspection is labeled by w_{UN}. Besides the insufficient inspection time, the operator's cognitive capacity constraint can also prevent him from determining whether the alert is triggered by a feint or a real attack. In this work, we consider prudent operators. When they cannot determine the attack's type after a full inspection, the associated alert is labeled as w_{UN}. We elaborate on how the insufficient inspection time and the operator's cognitive capacity constraint lead to w_{UN}, i.e., referred to as the *inadequate alert response*, in Section 12.3.4. The alerts labeled as w_{NI} and w_{UN} are ranked and queued up for delayed inspections at later stages.

When the operator successfully completes the alert inspection with a deterministic decision, he either dismisses the alert (denoted by w_{FE}) or escalates the alert to tier-2 SOC analysts for in-depth analysis (denoted by w_{RE}) as shown in Fig. 12.4. We use $w^k \in \mathcal{W} := \{w_{FE}, w_{RE}, w_{UN}, w_{NI}\}$ to denote the operator's response to the alert at attack stage $k \in \mathbb{Z}^{0+}$. We can extend the set $\mathcal{W}$ to suit the organization's security practice. For example, some organizations let the operators report their estimations and confidence levels concerning incomplete alert inspection, i.e., divide the label w_{UN} into finer subcategories. At later stages, the delayed inspection prioritizes the alerts estimated to be associated with real attacks of high confidence levels.

12.3.2 Probabilistic Switches within Allowable Delay

Alerts are monitored in real time when they arrive. When the category label of the new alert indicates higher time sensitivity, susceptibility, or criticality, the operator can delay the current inspection (i.e., label the alert under inspection as w_{UN}) and switch to inspect the new alert. We denote $\kappa_{SW}^{\Delta k}(s^{k+\Delta k}|s^k)$ as the operator's switching probability when the previous alert at attack stage k and the new alert at stage $k+\Delta k, \Delta k \in \mathbb{Z}^{+}$, have category label $s^k \in \mathcal{S}$ and $s^{k+\Delta k} \in \mathcal{S}$, respectively. As a probability measure,

$$\sum_{\Delta k=1}^{\infty} \sum_{s^{k+\Delta k} \in \mathcal{S}} \kappa_{SW}^{\Delta k}(s^{k+\Delta k}|s^k) \equiv 1, \forall k \in \mathbb{Z}^{0+}, \forall s^k \in \mathcal{S}. \tag{12.1}$$

Since the operator cannot observe the attack's hidden type and hidden target, the switching probability $\kappa_{SW}^{\Delta k}$ is independent of θ^k, ϕ^k and θ^{k+1}, ϕ^{k+1}. The switching probability depends on the time that the operator has already spent on the current inspection. For example, an operator becomes less likely to switch after spending a long time inspecting an alert of low criticality or beyond his capacity, which can lead to the Sunk Cost Fallacy (SCF).

We denote $D_{max} \in \mathbb{R}^{+}$ as the Maximum Allowable Delay (MAD). At time $t \geq t^k$, the k-th alert's Age of Information (AoI) [227] is defined as $t_{AoI}^k := t - t^k$. This work focuses on time-critical ICSs where a defensive response for the k-th alert is only effective when the alert's AoI is within the MAD, i.e., $t_{AoI}^k \leq D_{max}$. Therefore, the operator will be reminded when an alert's AoI exceeds the MAD so that he can switch to monitor and inspect new alerts. The MAD and the reminder scheme help mitigate the SCF when the operators are occupied with old alerts and miss the chance to monitor and inspect new alerts in real time.

12.3.3 Attentional Factors

We identify the following human and environmental factors affecting operators' alert inspection and response processes.

- The operator's expertise level denoted by $y_{EL} \in \mathcal{Y}_{EL}$.

- The k-th alert's category label $s^k \in \mathcal{S}$.
- The k-th attack's type θ^k and target ϕ^k.
- The operator's stress level $y_{SL}^t \in \mathbb{R}^+$, which changes with time t as new alerts arrive.

The first three factors are the static attributes of the analyst, the alert, and the IDoS attack, respectively. They determine the average inspection time, denoted by $\bar{d}(y_{EL}, s^k, \theta^k, \phi^k) \in \mathbb{R}^+$, to reach a *complete response* w_{FE} or w_{RE}. For example, if the inspected alert is of low complexity, the operator can reach a complete response in a shorter time. Also, it takes a senior operator less time on average to reach a complete alert response than a junior one does. We use $d(y_{EL}, s^k, \theta^k, \phi^k)$ to represent the Actual Inspection Time Needed (AITN) when the operator is of expertise level y_{EL}, the alert is of category label s^k, and the attack has type θ^k and target ϕ^k. AITN $d(y_{EL}, s^k, \theta^k, \phi^k)$ is a random variable with mean $\bar{d}(y_{EL}, s^k, \theta^k, \phi^k)$.

The fourth factor reflects the temporal aspect of human attention during the inspection process. Evidence has shown that the continuous arrival of the alerts can increase the stress level of human operators [8] and 52% of employees attributes their mistakes to stress [211]. We denote $n^t \in \mathbb{Z}^{0+}$ as the number of alerts that arrive during the current inspection up to time $t \in [0, \infty)$ and model the operator's stress level y_{SL}^t as an increasing function f_{SL} of n^t, i.e., $y_{SL}^t = f_{SL}(n^t)$. At time $t \in [0, \infty)$, the human operator's Level of Operational Efficiency (LOE), denoted by $\omega^t \in [0, 1]$, is a function f_{LOE} of the stress level y_{SL}^t, i.e.,

$$\omega^t = f_{LOE}(y_{SL}^t) = (f_{LOE} \circ f_{SL})(n^t), \forall t \in [0, \infty). \tag{12.2}$$

Based on the Yerkes–Dodson law, the function f_{LOE} follows an inverse U-shape that contains the following two regions. In region one, a small number of alerts result in a moderate stress level and allow human operators to inspect the alert efficiently. In region two, the LOE starts to decrease when the number of alerts to inspect is beyond some threshold $\bar{n}(y_{EL}, s^k)$ and the human operator is overloaded. The value of the *attention threshold* $\bar{n}(y_{EL}, s^k)$ depends on the operator's expertise level $y_{EL} \in \mathcal{Y}_{EL}$ and the alert's category label $s^k \in \mathcal{S}$. For example, it requires more (resp. fewer) alerts (i.e., higher (resp. lower) attention threshold) to overload a senior (resp. an inexperienced) operator. We can also adapt the value of $\bar{n}(y_{EL}, s^k)$ to different scenarios. In the extreme case where all alerts are of high complexity and create a heavy cognitive load, we let $\bar{n}(y_{EL}, s^k) = 0, \forall y_{EL} \in \mathcal{Y}_{EL}, s^k \in \mathcal{S}$, and the LOE decreases monotonously with the number of alert arrivals during an inspection.

12.3.4 Alert Responses under Time and Capacity Limitations

After we identify attentioinal factors in Section 12.3.3, we illustrate their impacts on the operators' alert responses as follows. We define the Effective Inspection Time (EIT) during inspection time $[t_1, t_2]$ as the integration $\tilde{\omega}^{t_1,t_2} := \int_{t_1}^{t_2} \omega^t dt$. When the operator is overloaded and has a low LOE during $[t_1, t_2]$, the EIT $\tilde{\omega}^{t_1,t_2}$ is much shorter than the actual inspection time $t_2 - t_1$.

Suppose that the operator of expertise level y_{EL} inspects the k-th alert for a duration of $[t_1, t_2]$. If the EIT has exceed the AITN $d(y_{EL}, s^k, \theta^k, \phi^k)$, then the operator can reach a complete response w_{FE} or w_{RE} with a high success probability denoted by $p_{SP}(y_{EL}, s^k, \theta^k, \phi^k) \in [0, 1]$. However, when $\tilde{\omega}^{t_1,t_2} < d(y_{EL}, s^k, \theta^k, \phi^k)$, it indicates that the operator has not completed the inspection and the alert response concerning the k-th alert is $w^k = w_{UN}$. The success probability p_{SP} depends on the operator's capacity to identify attacks' types, which leads to the definition of the capacity gap below.

Definition 39 (**Capacity Gap**). *For an operator of expertise level $y_{EL} \in \mathcal{Y}_{EL}$, we define $p_{CG}(y_{EL}, s^k, \theta^k, \phi^k) := 1 - p_{SP}(y_{EL}, s^k, \theta^k, \phi^k)$ as his capacity gap to inspect an alert with category label $s^k \in \mathcal{S}$, type $\theta^k \in \Theta$, and target $\phi^k \in \Phi$ defined in Section 12.2.*

12.4 Human-Assistive Security Technology

As illustrated in Section 12.3, the frequent arrival of alerts triggered by IDoS attacks can overload the human operator and reduce the LOE and the EIT. To compensate for the human's attentional limitation, we can intentionally make some alerts less noticeable, e.g., without sounds or in a light color, based on their category labels. As illustrated by the blue box in Fig. 12.4, based on the category labels from the triage process, RADAMS automatically emphasizes and de-emphasizes alerts, and then presents them to the tier 1 SOC analysts.

12.4.1 Adaptive AM Strategy

In this work, we focus on the class of Attention Management (AM) strategies, denoted by $\mathcal{A} := \{a_m\}_{m \in \{0,1,\cdots,M\}}$, that de-emphasize consecutive alerts. As explained in Section 12.3.1, the operator can only inspect some alerts in real time. Thus, we use $I_h \in \mathbb{Z}^{0+}$ and $t^{I_h} \in [0, \infty)$ to denote the index and the time of the alert under the h-th inspection; i.e., the inspection stage $h \in \mathbb{Z}^{0+}$ is equivalent to the attack stage $I_h \in \mathbb{Z}^{0+}$. Whenever the operator starts a new inspection at inspection stage $h \in \mathbb{Z}^{0+}$, RADAMS determines the AM action $a^h \in \mathcal{A}$ for the h-th inspection based on the stationary strategy $\sigma \in \Sigma : \mathcal{S} \mapsto \mathcal{A}$ that is adaptive to the category label of the h-th alert. We illustrate the timeline of the manual inspections and the AM strategies in green and blue, respectively, in Fig. 12.5. The solid and dashed green arrows indicate the inspected and uninspected alerts, respectively. The non-transparent and semi-transparent blue arrows indicate the emphasized and de-emphasized alerts, respectively. At inspection stage h, if $a^h = a_m$, RADAMS will make the next m alerts less noticeable; i.e., the alerts at attack stages $I_h + 1, \cdots, I_h + m$ are de-emphasized. Denote $\bar{\kappa}_{SW}^{I_{h+1}-I_h,a^h}(s^{I_{h+1}}|s^{I_h})$ as the operator's switching probability to these de-emphasized alerts under the AM action $a^h \in \mathcal{A}$. Analogously to (12.1), the following holds for all $h \in \mathbb{Z}^{0+}$ and $a^h \in \mathcal{A}$, i.e.,

$$\sum_{I_{h+1}=I_h+1}^{\infty} \sum_{s^{I_{h+1}} \in \mathcal{S}} \bar{\kappa}_{SW}^{I_{h+1}-I_h,a^h}(s^{I_{h+1}}|s^{I_h}) \equiv 1, \forall s^{I_h} \in \mathcal{S}. \tag{12.3}$$

The deliberate de-emphasis on selective alerts brings the following tradeoff. On the one hand, these alerts do not increase the operator's stress level and the operator can pay sustained attention to the alert under inspection with high LOE and EIT. On the other hand, these alerts do not draw the operator's attention and the operator is less likely to switch to them during the real-time monitoring and inspections.

Since the operator may switch to inspect a de-emphasized alert with switching probability $\bar{\kappa}_{SW}^{I_{h+1}-I_h,a^h}$ (e.g., the h-inspection in Fig. 12.5), RADAMS recomputes the AM strategy and implements the new strategy whenever the operator has started to inspect a new alert. Although the operator can switch unpredictably, Proposition 18 shows that the transition of the inspected alerts' category labels is Markov.

Proposition 18. *For a stationary AM strategy $\sigma \in \Sigma$, the set of random variables $(\mathbf{S}^{\mathbf{I}_h}, \mathbf{T}^{\mathbf{I}_h})_{h \in \mathbb{Z}^{0+}}$ is a Markov renewal process.*

Proof. The sketch of the proof includes two steps. First, we prove that the state transition from s^{I_h} to $s^{I_{h+1}}$ is Markov for all $h \in \mathbb{Z}^{0+}$. Due to the uncertainty of switching in inspection, the transition stage $\mathbf{I}_{h+1}$ is also a random variable for all $h \in \mathbb{Z}^{0+}$ and we can represent the transition probability as

$$\Pr(\mathbf{S}^{\mathbf{I}_{h+1}} = s^{I_{h+1}} | s^{I_h}) = \sum_{l=1}^{\infty} \Pr(\mathbf{I}_{h+1} = I_h + l) \cdot \Pr(\mathbf{S}^{\mathbf{I}_{h+1}} = s^{I_{h+1}} | s^{I_h}),$$

where $\Pr(\mathbf{I}_{h+1} = I_h + l)$ is the probability that the $(h+1)$-th inspection happens at attack stage $I_h + l$. The term $\Pr(\mathbf{S}^{\mathbf{I}_{h+1}} = s^{I_{h+1}} | s^{I_h})$ is Markov and can be computed based on κ_{CL}. The term $\Pr(\mathbf{I}_{h+1} = I_h + l)$ depends on $d(y_{EL}, s^{I_h+l'}, \theta^{I_h+l'}, \phi^{I_h+l'})$, $\kappa_{SW}^{l'}$, $\bar{\kappa}_{SW}^{l'}$, $\tau^{l'}$, for all $l' \in \{1, \cdots, l\}$. Since $s^{I_h+l'}, \theta^{I_h+l'}, \phi^{I_h+l'}, l' \in \{1, \cdots, l\}$, are all stochastically related to s^{I_h} and $s^{I_{h+1}}$ based on o, κ_{AT} and κ_{CL}, the term $\Pr(\mathbf{I}_{h+1} = I_h + l)$ depends on s^{I_h} and $s^{I_{h+1}}$ for all $l \in \mathbb{Z}^+$.

Then, we show that the distribution of the inter-arrival time $\mathbf{T}_{IN}^{\mathbf{I}_h,m} := \mathbf{T}^{\mathbf{I}_{h+1}} - \mathbf{T}^{I_h}$ only depends on s^{I_h} and $s^{I_{h+1}}$. Analogously, the cumulative distribution function of $\mathbf{T}_{IN}^{\mathbf{I}_h,m}$ is

$$\Pr(\mathbf{T}_{IN}^{\mathbf{I}_h,m} \leq t) = \sum_{l=1}^{\infty} \Pr(\mathbf{I}_{h+1} = I_h + l) \cdot \Pr(\mathbf{T}_{IN}^{I_h,m} \leq t),$$

and hence we arrive at the Markov property. □

12.4.2 Stage Cost and Expected Cumulative Cost

For each alert at attack stage $k \in \mathbb{Z}^{0+}$, RADAMS assigns a stage cost $\bar{c}(w^k, s^k)$ based on the alert response $w^k \in \mathcal{W}$ and the category label $s^k \in \mathcal{S}$. The value of the cost can be estimated by the salary of SOC analysts and the estimated loss of the associated attack. For example, $\bar{c}(w_{UN}, s^{I_h})$ and $\bar{c}(w_{NI}, s^{I_h})$ are positive costs as those alerts without a complete response incur additional workloads. The delayed inspections also expose the organization to the threats of time-sensitive attacks. On the other hand, $\bar{c}(w_{FE}, s^{I_h})$ and $\bar{c}(w_{RE}, s^{I_h})$ are negative costs as the alerts with complete alert response w_{FE} and w_{RE} reduce the workload of tier 2 SOC analysts and enable them to obtain threat intelligence.

When the operator starts a new inspection at inspection stage $h+1$, RADAMS will evaluate the effectiveness of the AM strategy for the h-th inspection. The performance evaluation is reflected by the Expected Consolidated Cost (ECoC) $c : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ at each inspection stage $h \in \mathbb{Z}^{0+}$. We denote the realization of $c(s^{I_h}, a^h)$ as the Consolidated Cost (CoC) $\tilde{c}^{I_h}(s^{I_h}, a^h)$. Since the AM strategy σ at each inspection stage can affect the future human inspection process and the alert responses, we define the Expected Cumulative Cost (ECuC) $u(s^{I_h}, \sigma) := \sum_{h=0}^{\infty} \gamma^h c(s^{I_h}, \sigma(s^{I_h}))$ under adaptive strategy $\sigma \in \Sigma$ as the long-term performance measure. The goal of the assistive technology is to design the optimal adaptive strategy $\sigma^* \in \Sigma$ that minimizes the ECuC u under the presented IDoS attack based on the category label $s^{I_h} \in \mathcal{S}$ at each inspection stage h. We define $v^*(s^{I_h}) := \min_{\sigma \in \Sigma} u(s^{I_h}, \sigma)$ as the optimal ECuC when the category label is $s^{I_h} \in \mathcal{S}$. We refer to the *default AM strategy* $\sigma^0 \in \Sigma$ as the one when no AM action is applied under all category labels, i.e., $\sigma^0(s^{I_h}) = a_0, \forall s^{I_h} \in \mathcal{S}$.

12.4.3 Reinforcement Learning

Due to the absence of the following exact model parameters, RADAMS has to learn the optimal AM strategy $\sigma^* \in \Sigma$ based on the operator's alert responses in real time.

- Parameters of the IDoS attack model (e.g., κ_{AT} and z) and the alert generation model (e.g., o) in Section 12.2.
- Parameters of the human attention model (e.g., f_{LOE} and f_{Sl}), inspection model (e.g., $\kappa_{SW}^{\Delta k}$, $\bar{\kappa}_{SW}^{I_{h+1}-I_h,a^h}$, and d), and alert response model (e.g., y_{EL} and p_{SP}) in Section 12.3.

Define $Q^h(s^{I_h}, a^h)$ as the estimated ECuC during the h-th inspection when the category label is $s^{I_h} \in \mathcal{S}$ and the AM action is a^h. Based on Proposition 18, the state transition is Markov, which enables Q-learning as follows.

$$\begin{aligned} Q^{h+1}(s^{I_h}, a^h) &:= (1-\alpha^h(s^{I_h}, a^h))Q^h(s^{I_h}, a^h) \\ &\quad + \alpha^h(s^{I_h}, a^h)[\tilde{c}^{I_h}(s^{I_h}, a^h) + \gamma \min_{a' \in \mathcal{A}} Q^h(s^{I_{h+1}}, a')], \end{aligned} \tag{12.4}$$

where s^{I_h} and $s^{I_{h+1}}$ are the observed category labels of the alerts at the attack stage I_h and I_{h+1}, respectively. When the learning rate $\alpha^h(s^{I_h}, a^h) \in (0,1)$ satisfies $\sum_{h=0}^{\infty} \alpha^h(s^{I_h}, a^h) = \infty, \sum_{h=0}^{\infty}(\alpha^h(s^{I_h}, a^h))^2 < \infty, \forall s^{I_h} \in \mathcal{S}, \forall a^h \in \mathcal{A}$, and all state-action pairs are explored infinitely, $\min_{a' \in \mathcal{A}} Q^h(s^{I_h}, a')$ converges to the optimal ECuC $v^*(s^{I_h})$ with probability 1 as $h \to \infty$. At each inspection stage $h \in \mathbb{Z}^{0+}$, RADAMS selects AM strategy $a^h \in \mathcal{A}$ based on the ϵ-greedy policy; i.e., RADAMS chooses a random action with a small probability $\epsilon \in [0,1]$, and the optimal action $arg \min_{a' \in \mathcal{A}} Q^h(s^{I_h}, a')$ with probability $1-\epsilon$.

Algorithm 13: Algorithm to Learn the Adaptive AM strategy

```
Input $K$: The total number of attack stages;
Initialize The operator starts the $h$-th inspection under AM action $a^h \in \mathcal{A}$; $I_h = k_0$;
    $\tilde{c}^{I_h}(s^{I_h}, a^h) = 0$;
for $k \leftarrow k_0 + 1$ to $K$ do
    if The operator has finished the $I_h$-th alert (i.e., EIT $>$ AITN), then
        if Capable (i.e., $rand \leq p_{SP}(y_{EL}, s^k, \theta^k, \phi^k)$) then
            Dismiss (i.e., $w^{I_h} = w_{FE}$) or escalate (i.e., $w^{I_h} = w_{RE}$) the $I_h$-th alert;
        else
            Queue up the $I_h$-th alert, i.e., $w^{I_h} = w_{UN}$;
        $\tilde{c}^{I_h}(s^{I_h}, a^h) = \tilde{c}^{I_h}(s^{I_h}, a^h) + \bar{c}(w^{I_h}, s^{I_h})$;
        $I_{h+1} \leftarrow k$; The operator starts to inspect the $k$-th alert with category label $s^{I_{h+1}}$;
        Update $Q^{h+1}(s^{I_h}, a^h)$ via (12.4) and obtain the AM action $a^{h+1}$ by $\epsilon$-greedy
          policy;
        $\tilde{c}^{h+1}(s^{I_{h+1}}, a^{h+1}) = 0$; $h \leftarrow h + 1$;
    else
        if The operator chooses to switch or The MAD is reached, i.e., $t^k - t^{I_h} \geq D_{max}$
          then
            Queue up the $I_h$-th alert (i.e., $w^{I_h} = w_{UN}$);
            $\tilde{c}^{I_h}(s^{I_h}, a^h) = \tilde{c}^{I_h}(s^{I_h}, a^h) + \bar{c}(w_{UN}, s^{I_h})$;
            $I_{h+1} \leftarrow k$; The operator starts to inspect the $k$-th alert with category label
              $s^{I_{h+1}}$;
            Update $Q^{h+1}(s^{I_h}, a^h)$ via (12.4) and obtain the AM action $a^{h+1}$ by $\epsilon$-greedy
              policy;
            $\tilde{c}^{h+1}(s^{I_{h+1}}, a^{h+1}) = 0$; $h \leftarrow h + 1$;
        else
            The operator continues the inspection of the $I_h$-th alert with decreased LOE;
            The $k$-th alert is queued up for delayed inspection (i.e., $w^k = w_{NI}$);
            $\tilde{c}^{I_h}(s^{I_h}, a^h) = \tilde{c}^{I_h}(s^{I_h}, a^h) + \bar{c}(w_{NI}, s^k)$;
Return $Q^h(s, a), \forall s \in \mathcal{S}, a \in \mathcal{A}$;
```

We present the algorithm to learn the adaptive AM strategy based on the operator's real-time alert monitoring and inspection process in Algorithm 13. Each simulation run corresponds to the operator's work shift of 24 hours at the SOC. Since the SOC can receive over 10 thousand of alerts in each work shift, we can use infinite horizon to approximate the total number of attack stages $K > 10,000$. Whenever the operator starts to inspect a new alert at inspection stage I_{h+1}, RADAMS applies Q-learning in (12.4) based on the category label $s^{I_{h+1}}$ of the newly arrived alert and determines the AM action a^{h+1} for the $h+1$ inspection based on the ϵ-greedy policy as shown in lines 12 and 19 of Algorithm 13. The CoC $\tilde{c}^{I_h}(s^{I_h}, a^h)$ of the h-th inspection under the AM action $a^h \in \mathcal{A}$ and the category label s^{I_h} of the inspected alert can be computed iteratively based on the stage cost $\bar{c}(w^k, s^k)$ of the alerts during the attack stage $k \in \{I_h, \cdots, I_{h+1}-1\}$ as shown in lines 13, 20, and 24 of Algorithm 13.

12.5 Theoretical Analysis

In Section 12.5, we focus on the class of ambitious operators who attempt to inspect all alerts, i.e., $\kappa_{SW}(s^{k+\Delta k}|s^k) = \mathbf{1}_{\{\Delta k=1\}}, \forall s^k, s^{k+\Delta k} \in \mathcal{S}, \forall \Delta k \in \mathbb{Z}^+$. To assist this class of operators, the implemented AM action $a_m, m \in \{0, 1, \cdots, M\}$, chooses to make the selected alerts fully unnoticeable. Then, under $a_m \in \mathcal{A}$, the operator at inspection stage h can pay sustained attention to inspect the alert of category label $s^{I_h} \in \mathcal{S}$ for $m+1$ attack stages. Moreover, the operator switches to the new alert at attack stage I_{h+1}, i.e., $\sum_{s^{I_h+m+1} \in \mathcal{S}} \bar{\kappa}_{SW}^{I_{h+1}-I_h, a_m}(s^{I_h+m+1}|s^{I_h}) = \mathbf{1}_{\{I_{h+1}-I_h=m+1\}}$. Throughout the section, we omit the variable of the expertise level y_{EL} in functions $d, \bar{d}, p_{SP}$, and p_{CG} as y_{EL} is a constant for all attack stages.

12.5.1 Security Metrics

We propose two security metrics in Definition 40 to evaluate the performance of ambitious operators under IDoS attacks and different AM strategies. The first metric, denoted as $p_{UN}(s^{I_h}, a^h)$, is the probability that the operator chooses w_{UN} during the h-th inspection under the category label $s^{I_h} \in \mathcal{S}$ and AM action $a^h \in \mathcal{A}$. This metric reflects the Attentional Deficiency Level (ADL) of the IDoS attack. For example, as the attackers generate more feints at a higher frequency, the operator is persistently distracted by the new alerts, and it becomes unlikely for him to fully respond to an alert. The ADL $p_{UN}(s^{I_h}, a^h)$ is high in this scenario. We use the ECuC $u(s^{I_h}, \sigma)$ as the second metric that evaluates the *IDoS risk* under the category label $s^{I_h} \in \mathcal{S}$ and the AM strategy $\sigma \in \Sigma$. For both metrics, smaller values are preferred.

Definition 40 (Attentional Deficiency Level and Risk). *Under category label $s^{I_h} \in \mathcal{S}$ and the stationary AM strategy $\sigma \in \Sigma$, we define $p_{UN}(s^{I_h}, \sigma(s^{I_h}))$ and $u(s^{I_h}, \sigma)$ as the Attentional Deficiency Level (ADL) and the risk of the IDoS attacks defined in Section 12.2, respectively.*

12.5.2 Closed-Form Computations

The Markov renewal process that characterizes the IDoS attack or the associated alert sequence follows a Poisson process when Condition 1 holds.

Condition 1 (Poisson Arrival). *The inter-arrival time τ^k for all attack stage $k \in \mathbb{Z}^{0+}$ is exponentially distributed with the same arrival rate denoted by $\beta > 0$, i.e., $z(\tau|\theta^{k+1}, \phi^{k+1}, \theta^k, \phi^k) = \beta e^{-\beta\tau}, \tau \in [0, \infty)$ for all $\theta^{k+1}, \theta^k \in \Theta$ and $\phi^{k+1}, \phi^k \in \Phi$.*

Recall that random variable $\mathbf{T}_{IN}^{I_h,m}$ represents the inspection time of the I_h-th alert under the AM action $a^h = a_m \in \mathcal{A}$. For the ambitious operators under AM action $a_m \in \mathcal{A}$ at inspection stage h, the next inspection happens at attack stage $I_{h+1} = I_h + m + 1$. Thus, I_{h+1} is no longer a random variable. As a summation of $m + 1$ i.i.d. exponential distributed random variables of rate β, $\mathbf{T}_{IN}^{I_h,m}$ follows an *Erlang distribution* denoted by $\bar{z}$ with shape $m + 1$ and and rate $\beta > 0$ when condition 1 holds, i.e., $\bar{z}(\tau) = \frac{\beta^{m+1}\tau^m e^{-\beta\tau}}{m!}, \tau \in [0, \infty)$.

Denote $p^h_{SD}(w^{I_h}|s^{I_h}, a^h; \theta^{I_h}, \phi^{I_h})$ as the probability that the operator makes alert response w^{I_h} at inspection stage h. To obtain a theoretical underpinning, we consider the case where the AITN equals the average inspection time, i.e., $d(s^k, \theta^k, \phi^k) = \bar{d}(s^k, \theta^k, \phi^k)$. Then, the operator under AM action a_m makes a complete alert response (i.e., $w^{I_h} \in \{w_{FE}, w_{RE}\}$) at inspection stage h for category label s^{I_h} if the inspection time $\tau^{I_h,m}_{IN}$ is greater than the AITN. The probability of the above event can be represented as $\int_{d(s^{I_h},\theta^{I_h},\phi^{I_h})}^{\infty} p_{SP}(s^{I_h}, \theta^{I_h}, \phi^{I_h})\bar{z}(\tau)d\tau = p_{SP}(s^{I_h}, \theta^{I_h}, \phi^{I_h}) \cdot \sum_{n=0}^{m} \frac{1}{n!} e^{-\beta d(s^{I_h},\theta^{I_h},\phi^{I_h})}(\beta d(s^{I_h}, \theta^{I_h}, \phi^{I_h}))^n$, which leads to

$$\begin{aligned} p^h_{SD}(w_{UN}|s^{I_h}, a_m; \theta^{I_h}, \phi^{I_h}) = 1 - p_{SP}(s^{I_h}, \theta^{I_h}, \phi^{I_h}) \\ \cdot \sum_{n=0}^{m} \frac{1}{n!} e^{-\beta d(s^{I_h},\theta^{I_h},\phi^{I_h})}(\beta d(s^{I_h}, \theta^{I_h}, \phi^{I_h}))^n. \end{aligned} \tag{12.5}$$

Then, the ADL $p_{UN}(s^{I_h}, a^h)$ can be computed as

$$\sum_{\theta^{I_h}\in\Theta,\phi^{I_h}\in\Phi} \Pr(\theta^{I_h}, \phi^{I_h}|s^{I_h}) \cdot p^h_{SD}(w_{UN}|s^{I_h}, a^h; \theta^{I_h}, \phi^{I_h}), \tag{12.6}$$

where the conditional probability $\Pr(\theta^{I_h}, \phi^{I_h}|s^{I_h})$ can be computed via the Bayesian rule, i.e., $\Pr(\theta^{I_h}, \phi^{I_h}|s^{I_h}) = \frac{o(s^{I_h}|\theta^{I_h},\phi^{I_h})b(\theta^{I_h},\phi^{I_h})}{\sum_{\theta^{I_h}\in\Theta,\phi^{I_h}\in\Phi} o(s^{I_h}|\theta^{I_h},\phi^{I_h})b(\theta^{I_h},\phi^{I_h})}$.

We can compute the ECoC $c(s^{I_h}, a_m)$ explicitly as

$$\begin{aligned} c(s^{I_h}, a_m) = m\bar{c}(w_{NI}, s^{I_h}) + \sum_{\theta^{I_h}\in\Theta,\phi^{I_h}\in\Phi} \Pr(\theta^{I_h}, \phi^{I_h}|s^{I_h}) \\ \cdot \sum_{w^{I_h}\in\mathcal{W}} p^h_{SD}(w^{I_h}|s^{I_h}, a_m; \theta^{I_h}, \phi^{I_h})\bar{c}(w^{I_h}, s^{I_h}). \end{aligned} \tag{12.7}$$

For prudent operators in Section 12.3.1, we have

$$p^h_{SD}(w_i|s^{I_h}, a^h; \theta_i, \phi^{I_h}) = 1 - p^h_{SD}(w_{UN}|s^{I_h}, a^h; \theta_i, \phi^{I_h}), \tag{12.8}$$

for all $i \in \{FE, RE\}, s^{I_h} \in \mathcal{S}, a^h \in \mathcal{A}, \phi^{I_h} \in \Phi, h \in \mathbb{Z}^{0+}$. Plugging (12.8) into (12.7), we can simplify the ECoC $c(s^{I_h}, a_m)$ as

$$\begin{aligned} c(s^{I_h}, a_m) = &\sum_{\phi^{I_h} \in \Phi} \sum_{i \in \{FE, RE\}} \Pr(\theta_i, \phi^{I_h}|s^{I_h}) \cdot p^h_{SD}(w_i|s^{I_h}, a_m; \theta_i, \phi^{I_h}) \\ &\cdot [\bar{c}(w_i, s^{I_h}) - \bar{c}(w_{UN}, s^{I_h})] + m\bar{c}(w_{NI}, s^{I_h}) + \bar{c}(w_{UN}, s^{I_h}). \end{aligned} \tag{12.9}$$

As shown in Proposition 19, the ADL and the risk are monotone function of $\beta d(s^{I_h}, \theta^{I_h}, \phi^{I_h})$ for each AM strategy.

Proposition 19. *If condition 1 holds, then the ADL $p_{UN}(s^{I_h}, \sigma(s^{I_h}))$ and the risk $u(s^{I_h}, \sigma)$ of an IDoS attack under category label $s^{I_h} \in \mathcal{S}$ and AM strategy $\sigma \in \Sigma$ increase in the value of the product $\beta d(s^{I_h}, \theta^{I_h}, \phi^{I_h})$.*

Proof. First, since $p^h_{SD}(w_{UN})$ in (12.5) increases monotonously with respect to the product $\beta d(s^{I_h}, \theta^{I_h}, \phi^{I_h})$, the values of $p^h_{SD}(w_{FE})$ and $p^h_{SD}(w_{RE})$ in (12.8) decrease monotonously with respect to the product. Plugging (12.5) into (12.6), we obtain that $p_{UN}(s^{I_h}, a_m)$ in (12.10) under any $a_m \in \mathcal{A}$ and $s^{I_h} \in \mathcal{S}$ is a summation of functions increasing in $\beta d(s^{I_h}, \theta^{I_h}, \phi^{I_h})$.

$$\begin{aligned} p_{UN}(s^{I_h}, a_m) = &\sum_{\phi^{I_h} \in \Phi} \sum_{i \in \{FE, RE\}} \Pr(\theta_i, \phi^{I_h}|s^{I_h})[1- \\ &p_{SP}(s^{I_h}, \theta_i, \phi^{I_h}) \cdot \sum_{n=0}^{m} \frac{1}{n!} e^{-\beta d(s^{I_h}, \theta_i, \phi^{I_h})} (\beta d(s^{I_h}, \theta_i, \phi^{I_h}))^n]. \end{aligned} \tag{12.10}$$

Second, since $\bar{c}(w_{FE}, s^{I_h})$ and $\bar{c}(w_{RE}, s^{I_h})$ are negative and $\bar{c}(w_{UN}, s^{I_h})$ is positive, the ECoC in (12.9) decreases with $\beta d(s^{I_h}, \theta^{I_h}, \phi^{I_h})$ under any $a_m \in \mathcal{A}$ and $s^{I_h} \in \mathcal{S}$. Then, the risk also decreases with the product due to the monotonicity of the Bellman operator [19]. □

Remark 29 (**Product Principle of Attention (PPoA)**). *On the one hand, as β increases, the feint and real attacks arrive at a higher frequency on average, resulting in a higher demand of attention resources from the human operator. On the other hand, as $d(s^{I_h}, \theta^{I_h}, \phi^{I_h})$ increases, the human operator requires a longer inspection time to determine the attack's type, leading to a lower supply of attention resources. Proposition 19 characterizes the PPoA that the ADL and the risk of IDoS attacks depend on the product of the supply and demand of attention resources for any stationary AM strategy $\sigma \in \Sigma$.*

12.5.3 Fundamental Limits under AM strategies

Section 12.5.3 aims to show the fundamental limits of the IDoS attack's ADL, the ECoC, and the risk under different AM strategies. Define the shorthand notation: $\underline{p}(s^{I_h}) := \sum_{\phi^{I_h} \in \Phi} \sum_{i \in \{FE,RE\}} \Pr(\theta_i, \phi^{I_h} | s^{I_h}) p_{CG}(s^{I_h}, \theta_i, \phi^{I_h})$.

Lemma 11. *If Condition 1 holds and $M \to \infty$, then for each $s^{I_h} \in \mathcal{S}$, the ADL $p_{UN}(s^{I_h}, a_m)$ decreases strictly to $\underline{p}(s^{I_h})$ as m increases.*

Proof. Since $\frac{1}{n!} e^{-\beta d(s^{I_h}, \theta^{I_h}, \phi^{I_h})})(\beta d(s^{I_h}, \theta^{I_h}, \phi^{I_h}))^n > 0$ for all $m \in \{0, \cdots, M\}$, the value of $p_{UN}(s^{I_h}, a_m)$ in (12.10) strictly decreases as m increases. Moreover, $\lim_{m \to \infty} \sum_{n=0}^{m} \frac{1}{n!} e^{-\beta d(s^{I_h}, \theta^{I_h}, \phi^{I_h})})(\beta d(s^{I_h}, \theta^{I_h}, \phi^{I_h}))^n = 1$ leads to the following equation $\min_{m \in \{0, \cdots, M\}} p_{UN}(s^{I_h}, a_m) = \underline{p}(s^{I_h})$ for all $s^{I_h} \in \mathcal{S}$. □

Remark 30 (**Fundamental Limit of ADL**). *Lemma 11 characterizes that the minimum ADL under all AM strategies $a_m \in \mathcal{A}$ is $\underline{p}(s^{I_h})$. The value of $\underline{p}(s^{I_h})$*

depends on both the operator's capacity gap $p_{CG}(s^{I_h}, \theta_{FE}, \phi^{I_h})$ *and the frequency of feint and real attacks with different targets, i.e.,* $\Pr(\theta^{I_h}, \phi^{I_h}|s^{I_h}), \forall \theta^{I_h} \in \Theta, \phi^{I_h} \in \Phi$.

Denote the expected reward of making a complete alert response (i.e., the rewards to dismiss feints and escalate real attacks) as

$$\lambda(s^{I_h}, m, \phi^{I_h}) = \sum_{i\in\{FE,RE\}} \bar{c}(w_i, s^{I_h}) \cdot \Pr(\theta_i, \phi^{I_h}|s^{I_h})$$
$$\cdot\, p^h_{SP}(s^{I_h}, \theta_i, \phi^{I_h}) \cdot [\sum_{n=0}^{m} \frac{1}{n!} e^{-\beta d(s^{I_h}, \theta_i, \phi^{I_h})} (\beta d(s^{I_h}, \theta_i, \phi^{I_h}))^n].$$

Combining (12.10) and (12.9), we can rewrite ECoC as a combination of the following three terms in (12.11).

$$c(s^{I_h}, a_m) = p_{UN}(s^{I_h}, a_m)\bar{c}(w_{UN}, s^{I_h}) + m\bar{c}(w_{NI}, s^{I_h}) + \sum_{\phi^{I_h}\in\Phi} \lambda(s^{I_h}, m, \phi^{I_h}). \tag{12.11}$$

Based on Lemma 11, the first term $p_{UN}(s^{I_h}, a_m)\bar{c}(w_{UN}, s^{I_h})$ and the third term $\sum_{\phi^{I_h}\in\Phi} \lambda(s^{I_h}, m, \phi^{I_h})$ decrease in m while the second term $m\bar{c}(w_{NI}, s^{I_h})$ in (12.11) increases in m linearly at the rate of $\bar{c}(w_{NI}, s^{I_h})$. The tradeoff among the three terms is summarized below.

Remark 31 (**Tradeoff among ADL, Reward of Alert Attention, and Impact for Alert Inattention**)**.** *Based on Lemma 11 and* (12.11)*, increasing* m *reduces the ADL, and achieves a higher reward of completing the alert response. However, the increase of* m *also linearly increases the impact for alert inattention represented by* $m\bar{c}(w_{NI}, s^{I_h})$*, the cost of uninspected alerts. Thus, we need to strike a balance among these terms to reduce the IDoS risk.*

We define $\lambda_{min}(s^{I_h}, \phi^{I_h}) := \sum_{i \in \{FE,RE\}} \bar{c}(w_i, s^{I_h}) \Pr(\theta_i, \phi^{I_h}|s^{I_h}) p^h_{SP}(s^{I_h}, \theta_i, \phi^{I_h})$, $\lambda^{\epsilon_0}_{max}(s^{I_h}, \phi^{I_h}) := (1-\epsilon_0)\lambda_{min}(s^{I_h}, \phi^{I_h})$, $c_{min}(s^{I_h}) := \sum_{\phi^{I_h} \in \Phi} \lambda_{min}(s^{I_h}, \phi^{I_h}) + \underline{p}(s^{I_h})$ $\bar{c}(w_{UN}, s^{I_h}) + m\bar{c}(w_{NI}, s^{I_h})$, and $c^{\epsilon_0}_{max}(s^{I_h}) := \sum_{\phi^{I_h} \in \Phi} \lambda^{\epsilon_0}_{max}(s^{I_h}, \phi^{I_h}) + [\underline{p}(s^{I_h}) + \epsilon_0(1 - \underline{p}(s^{I_h}))]\bar{c}(w_{UN}, s^{I_h}) + m\bar{c}(w_{NI}, s^{I_h})$.

Proposition 20. *Consider the scenario where Condition 1 holds and $M > \underline{m}(s^{I_h})$. For any $\epsilon_0 \in (0,1]$ and $s^{I_h} \in \mathcal{S}$, there exists $\underline{m}(s^{I_h}) \in \mathbb{Z}^+$ such that $c(s^{I_h}, a_m) \in [c_{min}(s^{I_h}), c^{\epsilon_0}_{max}(s^{I_h})], \forall a_m \in \mathcal{A}$, when $m \geq \underline{m}(s^{I_h})$. Moreover, the lower bound $c_{min}(s^{I_h})$ and the upper bound $c^{\epsilon_0}_{max}(s^{I_h})$ increase in m linearly at the same rate $\bar{c}(w_{NI}, s^{I_h})$.*

Proof. For any $\epsilon_0 \in (0,1]$, there exists $\underline{m}(s^{I_h}) \in \mathbb{Z}^+$ such that

$$\sum_{n=0}^{m} \frac{1}{n!} e^{-\beta d(s^{I_h}, \theta^{I_h}, \phi^{I_h})} (\beta d(s^{I_h}, \theta^{I_h}, \phi^{I_h}))^n \in [1-\epsilon_0, 1]$$

when $m \geq \underline{m}(s^{I_h})$. Based on Lemma 11, if $m > \underline{m}(s^{I_h})$, then $p_{UN}(s^{I_h}, a_m) \in [\underline{p}(s^{I_h}), \underline{p}(s^{I_h}) + \epsilon_0(1 - \underline{p}(s^{I_h}))]$. Plugging it into (12.11), we obtain the results. □

Let $\sigma^{\underline{m}} \in \Sigma$ denote the AM strategy that chooses to de-emphasize the next $m \geq \underline{m}(s^{I_h})$ alerts for all category label $s^{I_h} \in \mathcal{S}$. The monotonicity of the Bellman operator [19] leads to the following corollary.

Corollary 3. *Consider the scenario where Condition 1 holds and $M > \underline{m}(s^{I_h})$. For any $\epsilon_0 \in (0,1]$ and $s^{I_h} \in \mathcal{S}$, the upper and lower bounds of the risk $u(s^{I_h}, \sigma^{\underline{m}})$ increase in m linearly at the same rate of $\bar{c}(w_{NI}, s^{I_h})$.*

Remark 32 (Fundamental Limit of ECoC and Risk). *Proposition 20 and Corollary 3 show that the maximum length of the de-emphasized alerts for any $s^{I_h} \in \mathcal{S}$ should not exceed $\underline{m}(s^{hm})$ to reduce the ECoC and the risk of IDoS attacks.*

12.6 Case Study

The following section presents case studies to demonstrate the impact of IDoS attacks on human operators' alert inspections and alert responses, and further illustrate the effectiveness of RADAMS. Throughout the section, we adopt the attention model in Section 12.3.

12.6.1 Experiment Setup

We consider an IDoS attack targeting either the Programmable Logic Controllers (PLCs) in the physical layer or the data centers in the cyber layer of an ICS. We denoted these two targets as ϕ_P and ϕ_C, respectively. They constitute the binary set of attack targets $\Phi = \{\phi_P, \phi_C\}$ defined in Section 12.2.1. The SOC of the ICS is in charge of monitoring, inspecting, and responding to both the cyber and the physical alerts. We consider two system-level metrics defined in Section 12.2.2, the source $\mathcal{S}_{SO} = \{s_{SO,P}, s_{SO,C}\}$ and the criticality $\mathcal{S}_{CR} = \{s_{CR,L}, s_{CR,H}\}$, i.e., $\mathcal{S} = \mathcal{S}_{SO} \times \mathcal{S}_{CR}$. Let $s_{SO,P}$ and $s_{SO,C}$ represent the source of physical and cyber layers, respectively. We assume that the alert triage process can accurately identify the source of attacks, i.e., $\Pr(s_{SO,i}|\phi_j) = \mathbf{1}_{\{i=j\}}, \forall i, j \in \{P, C\}$. Let $s_{CR,L}$ and $s_{CR,H}$ represent low and high criticality, respectively. We assume that the triage process cannot accurately identify feints as low criticality and real attacks as high criticality. The revelation kernel is separable and takes the form of $o(s_{SO}, s_{CR}|\theta_i, \phi_j) = \Pr(s_{SO}|\phi_j) \cdot \Pr(s_{CR}|\theta_i), s_{SO} \in \mathcal{S}_{SO}, s_{CR} \in \mathcal{S}_{CR}, i \in \{FE, RE\}, j \in \{P, C\}$. We choose the values of o so that the attack is more likely to be feint (resp. real) when the criticality level is low (resp. high).

The inter-arrival time at attack stage $k \in \mathbb{Z}^{0+}$ follows an exponential distribution with rate $\beta(\theta^k, \theta^{k+1})$ parameterized by the attack's type θ^k, θ^{k+1}. Thus, the average inter-arrival time $\mu(\theta^k, \theta^{k+1}) := 1/\beta(\theta^k, \theta^{k+1})$ also depends on the attack's type at the current and the next attack stages as shown in Table 12.1. We choose the benchmark values based on the literature (e.g., [191, 192] and the references within) and attacks can change these values in different IDoS attacks.

Table 12.1: Benchmark values of the average inter-arrival time $\mu(\theta^k, \theta^{k+1}) = 1/\beta(\theta^k, \theta^{k+1}), \forall \theta^k, \theta^{k+1} \in \Theta$.

Average inter-arrival time from feints to real attacks	6s
Average inter-arrival time from real attacks to feints	10s
Average inter-arrival time between feints	15s
Average inter-arrival time between real attacks	8s

The average inspection time $\bar{d}$ in Section 12.3.3 depends on the criticality s^k_{CR} and attack's type θ^k at attack stage $k \in \mathbb{Z}^{0+}$ as shown in Table 12.2. We choose the benchmark values of $\bar{d}(s^k_{CR}, \theta^k)$ based on [191], and these values can change for different human operators and IDoS attacks. We add a random noise uniformly distributed in $[-5, 5]$ to the average inspection time to simulate the AITN.

Table 12.2: Benchmark values of the average inspection time $\bar{d}(s^k_{CR}, \theta^k), \forall \theta^k \in \Theta, s^k_{CR} \in \mathcal{S}_{CR}$.

Average time to inspect feints of low criticality	6s
Average time to inspect feints of high criticality	8s
Average time to inspect real attacks of low criticality	15s
Average time to inspect real attacks of high criticality	20s

The stage cost $\bar{c}(w^k, s^k_{SO})$ at attack stage $k \in \mathbb{Z}^{0+}$ in Section 12.4.2 depends on the alert response $w^k \in \mathcal{W}$ and the source $s^k_{SO} \in \mathcal{S}_{SO}$. We determine the benchmark values of $\bar{c}(w^k, s^k_{SO})$ per alert in Table 12.3 based on the salary of the SOC analysts and the estimated loss of the associated attacks.

Table 12.3: The benchmark values of the stage cost $\bar{c}(w^k, s_{SO}^k), \forall w^k \in \mathcal{W}, s_{SO}^k \in \mathcal{S}_{SO}$.

Reward of dismissing feints w_{FE}	\$80
Reward of identifying real attacks w_{RE} in physical layer	\$500
Reward of identifying real attacks w_{RE} in cyber layer	\$100
Cost of incomplete alert response w_{UN} or w_{NI}	\$300

12.6.2 Analysis of Numerical Results

We plot the dynamics of the operator's alert responses in Fig. 12.6 under the benchmark experiment setup in Section 12.6.1. We use green, purple, orange, and yellow to represent w_{UN}, w_{NI}, w_{FE}, and w_{RE}, respectively. The heights of squares are also used to distinguish the four categories.

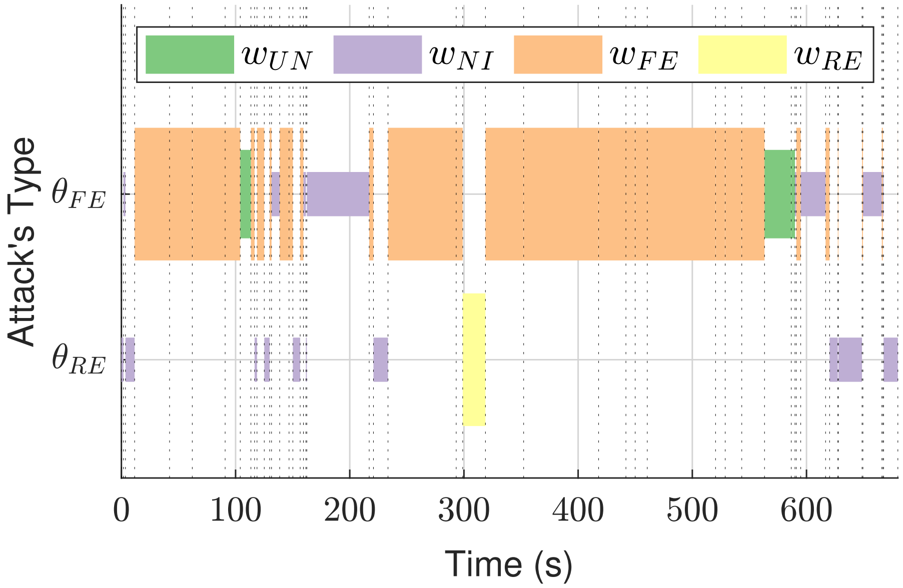


Figure 12.6: Alert response $w^k \in \mathcal{W}$ for the k-th attack whose type is shown in the y-axis. The k-th vertical dash line represents the k-th alert's arrival time t^k.

Learning during the Real-Time Monitoring and Inspection

Based on Algorithm 13, we illustrate the learning process of the estimated ECuC $Q^h(s^{I_h}, a^h)$ for all $s^{I_h} \in \mathcal{S}$ and $a^h \in \mathcal{A}$ at each inspection stage $h \in \mathbb{Z}^{0+}$ in Fig. 12.7. We choose $\alpha^h(s^{I_h}, a^h) = \frac{k_c}{k_{TI}(s^{I_h})-1+k_c}$ as the learning rate where $k_c \in (0, \infty)$ is a constant parameter and $k_{TI}(s^{I_h}) \in \mathbb{Z}^{0+}$ is the number of visits to $s^{I_h} \in \mathcal{S}$ up to stage $h \in \mathbb{Z}^{0+}$. Here, the AM action a^h is implemented randomly at each inspection stage h, i.e., $\epsilon = 1$. Thus, all four AM actions $(M = 3)$ are explored equally on average for each $s^{I_h} \in \mathcal{S}$ as shown in Fig. 12.7. Since the number of visits to different category labels depends on the transition probability κ_{AT}, the learning stages for four category labels are of different lengths.

We denote category labels $(s_{SO,P}, s_{CR,L})$, $(s_{SO,P}, s_{CR,H})$, $(s_{SO,C}, s_{CR,L})$, and $(s_{SO,C}, s_{CR,H})$ in blue, red, green, and black, respectively. To distinguish four AM actions, a deeper color represents a larger $m \in \{0, 1, 2, 3\}$ for each category label $s_{SO,i}, s_{CR,j}, i \in \{P, C\}, j \in \{H, L\}$. The inset black box magnifies the selected area. The optimal strategy $\sigma^* \in \Sigma$ is to take a_3 for all category labels. The risk $v^*(s^{I_h}) = u(s^{I_h}, \sigma^*)$ under the optimal strategy has the approximated values of \$1153, \$1221, \$1154, and \$1358 for the above category labels in blue, red, green, and black, respectively. We also simulate the operator's real-time monitoring and inspection under IDoS attacks when AM strategy is not applied based on Algorithm 13. The risks $v^0(s^{I_h}) := u(s^{I_h}, \sigma^0)$ under the default AM strategy $\sigma^0 \in \Sigma$ have the approximated values of \$1377, \$1527, \$1378, and \$1620 for the category label $(s_{SO,P}, s_{CR,L})$, $(s_{SO,P}, s_{CR,H})$, $(s_{SO,C}, s_{CR,L})$, and $(s_{SO,C}, s_{CR,H})$, respectively. These results illustrate that the optimal AM strategy $\sigma^* \in \Sigma$ can significantly reduce the risk under IDoS attacks for all category labels and the reduction percentage can be as high as 20%.

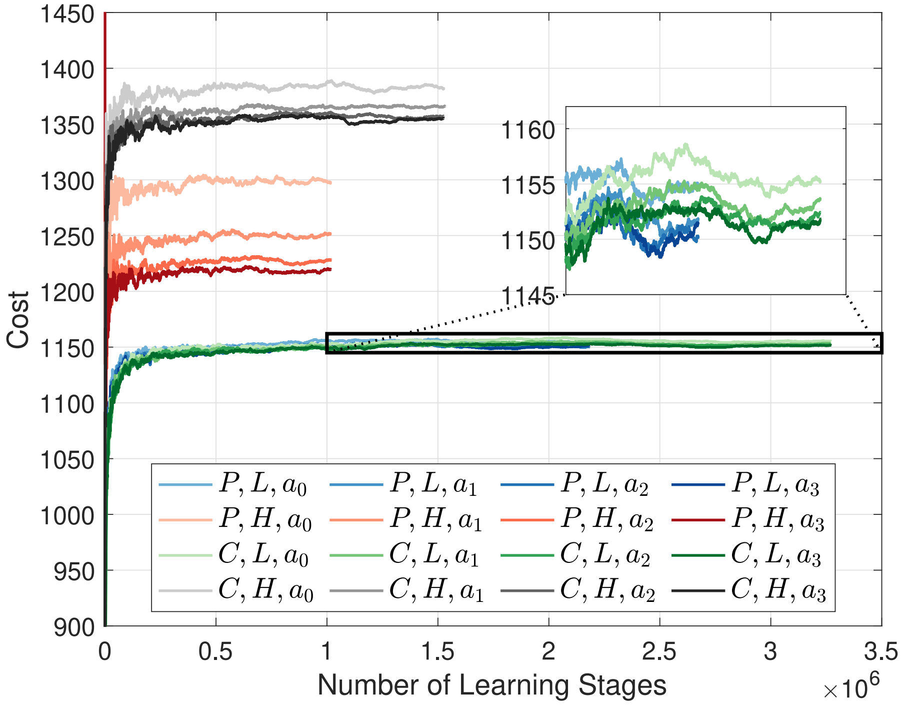


Figure 12.7: The convergence of the estimated ECuC $Q^h(s^{I_h}, a^h)$ vs. the number of inspection stages.

We further investigate the IDoS risk under the optimal AM strategy σ^* as follows. As illustrated in Fig. 12.7, when the criticality level is high (i.e., the attack is more likely to be real), the attacks targeting cyber layers (denoted in black) result in a higher risk than the one targeting physical layers (denoted in red). This asymmetry results from the different rewards of identifying real attacks in physical or cyber layers denoted in Table 12.3. Since dismissing feints bring the same reward in physical and cyber layers, the attacks targeting physical or cyber layers result in similar IDoS risks when the criticality level is low. Within physical or cyber layers, high-criticality alerts result in a higher risk than low-criticality alerts do.

The value of $Q^h(s^{I_h}, a_m), m \in \{0, 1, 2\}$, represents the risk when RADAMS deviates to sub-optimal AM action a_m for a single category label $s^{I_h} \in \mathcal{S}$. As illustrated by the red and black lines in Fig. 12.7, this single deviation can increase the risk under alerts of high criticality. However, it hardly increases the risk under alerts of low criticality as illustrated by the green and blue lines in the inset black box of Fig. 12.7. These results illustrate that we can deviate from the optimal AM strategy to sub-optimal ones for some category labels with approximately equivalent risk, which we refer to as the *attentional risk equivalency* in Remark 33.

Remark 33 (**Attentional Risk Equivalency**)**.** *The above results illustrate that we can contain the IDoS risk by selecting proper sub-optimal strategies. If applying the optimal AM strategy* σ^* *is costly, then RADAMS can choose not to apply AM strategy for* $(s_{SO,C}, s_{CR,L})$ *or* $(s_{SO,P}, s_{CR,L})$ *without significantly increasing the IDoS risks.*

Optimal AM Strategy and Resilience Margin under Different Stage Costs

We define *resilience margin* as the difference of the risks under the optimal and the default AM strategies. We investigate how the cost of incomplete alert response in Table 12.3 affects the optimal AM strategy and the resilience margin in Fig. 12.8.

As shown in the upper figure, the optimal strategy remains to choose AM action a_3 when the alert is of high criticality. When the alert is of low criticality, then as the cost increases, the optimal AM strategy changes sequentially from a_3, a_2, and a_1 to a_0; i.e., RADAMS gradually decreases $m \in \{0, 1, 2, 3\}$, the number of de-emphasized alerts. As shown in the lower figure, the resilience margin increases monotonously with the cost. The optimal strategy for alerts of high criticality yields a larger resilience margin than the one for low criticality.

Remark 34 (**Tradeoff of Monitoring and Inspection**)**.** *The results show that*

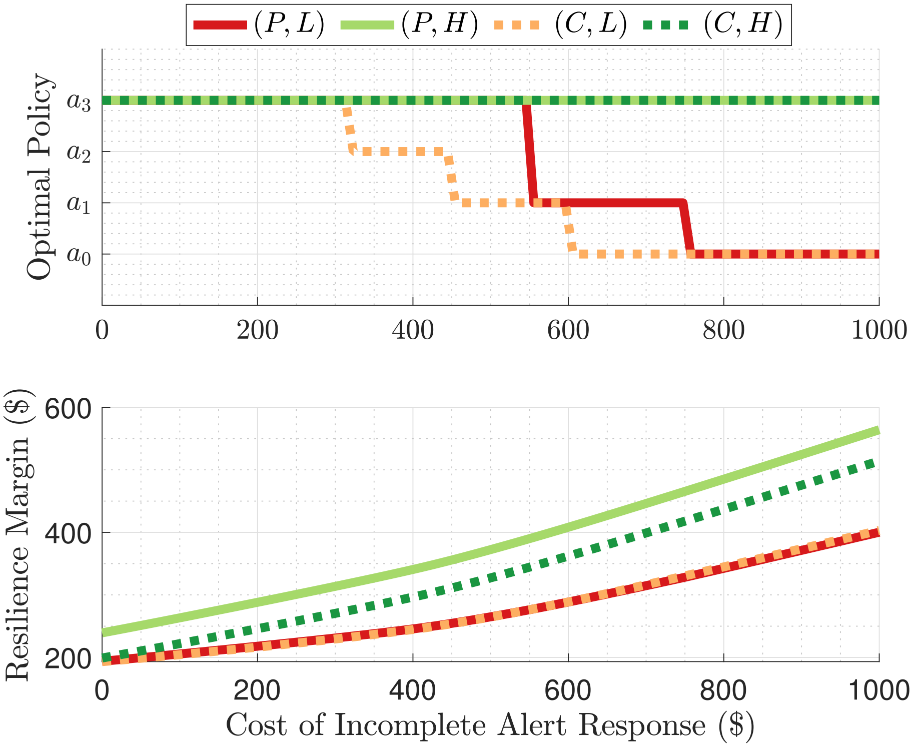


Figure 12.8: The optimal AM strategy and the risk vs. the cost of an incomplete alert response under category label $(s_{SO,P}, s_{CR,L})$, $(s_{SO,P}, s_{CR,H})$, $(s_{SO,C}, s_{CR,L})$, and $(s_{SO,C}, s_{CR,H})$ in solid red, solid green, dashed yellow, and dashed green, respectively.

the optimal strategy strikes a balance between real-time monitoring a large number of alerts and inspecting selected alerts with high quality. Moreover, the optimal strategy is resilient for a large range of cost values ($[\$0, \$1000]$*). If the cost is high and the alert is of low (resp. high) criticality, then the optimal strategy encourages monitoring (resp. inspecting) by choosing a small (resp. large) m. However, when the cost of an incomplete alert response is relatively low, the optimal strategy is* a_4 *for all alerts as the high-quality inspection outweighs the high-quantity monitoring.*

Arrival Frequency of IDoS Attacks

As stated in Section 12.2.1, feint attacks with the goal of triggering alerts require fewer resources to craft. Thus, we let $\hat{c}_{RE} = \$0.04$ and $\hat{c}_{FE} \in (0, \hat{c}_{RE})$ denote the cost to generate a real attack and a feint, respectively. With $\hat{c}_{RE}$ and $\hat{c}_{FE}$, we can compute the attack cost of feint and real attacks per work shift of 24 hours. Let ρ be the scaling factor for the arrival frequency and in Section 12.6.2, the average inter-arrival time is $\hat{\mu}(\theta^k, \theta^{k+1}) = \rho\mu(\theta^k, \theta^{k+1}), \forall \theta^k, \theta^{k+1} \in \Theta$. We investigate how the scale factor $\rho \in (0, 2.5]$ affects the IDoS risk and the attack cost in Fig. 12.9. As ρ decreases, the attacker generates feint and real attacks at a higher frequency. Then, the risks under both the optimal and the default strategies increase. However, the optimal AM strategy can reduce the increase rate for a large range of $\rho \in [0.5, 2]$.

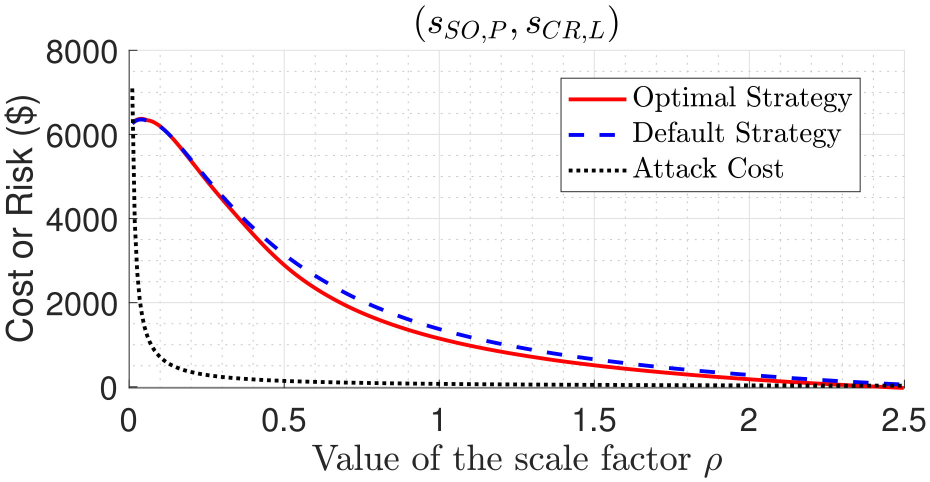


Figure 12.9: IDoS risk vs. ρ under the optimal and the default AM strategies in solid red and dashed blue, respectively. The black line represents the attack cost per work shift of 24 hours.

Remark 35 (**Attacker's Dilemma**). *From the attacker's perspective, although increasing the attack frequency can induce a high risk to the organization and the attacker can gain from it, the frequency increase also increases the attack cost* *exponentially as shown by the dotted black line in Fig. 12.9. Thus, the attacker has to strike a balance between the attack cost and the attack gain (represented by the IDoS risk). Moreover, attackers with a limited budget are not capable to choose small values of ρ (i.e., high attack frequencies).*

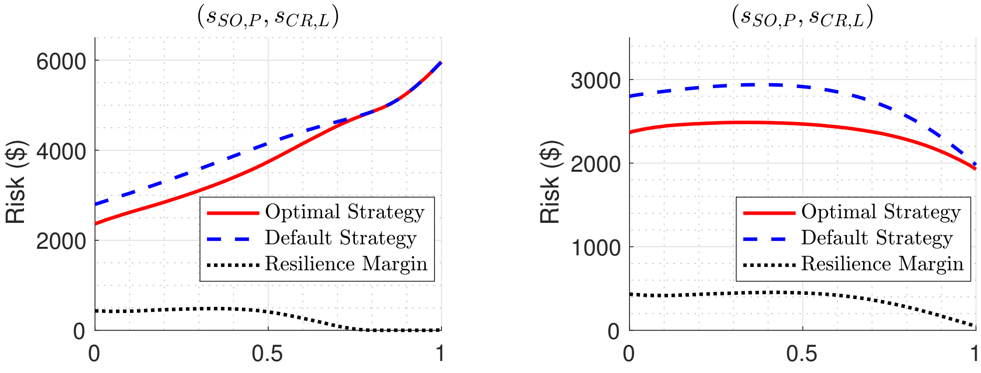
Percentage of Feint and Real Attacks

Consider the case where κ_{AT} independently generates feints and real attacks with probability η_{FE} and $\eta_{RE} = 1 - \eta_{FE}$, respectively. We consider the case where the attacker has a limited budge $\hat{c}_{max} = \$270$ per work shift (i.e., $86400s$) and generates feint and real attacks at the same rate $\hat{\beta}$, i.e., $\beta(\theta^k, \theta^{k+1}) = \hat{\beta}, \forall \theta^k, \theta^{k+1} \in \Theta$. Consider the attack cost in Section 12.6.2, the attacker has the following budget constraint, i.e.,

$$86400 \cdot \hat{\beta} \cdot (\eta_{FE}\hat{c}_{FE} + \eta_{RE}\hat{c}_{RE}) \leq \hat{c}_{max}. \tag{12.12}$$

The budget constraint results in the following tradeoff. If the attacker chooses to increase the probability of real attack η_{RE}, then he has to reduce the arrival frequency $\hat{\beta}$ of feint and real attacks. We investigate how the probability of feints affects the IDoS risk in Fig. 12.10 under the optimal and the default AM strategies in red and blue, respectively. The feints are of low and high costs in Fig. 12.10a and 12.10b, respectively.

As shown in Fig. 12.10a, when the feints are of low cost, i.e., $\hat{c}_{FE} = \hat{c}_{RE}/10$, generating feints with a higher probability monotonously increases the IDoS risks for both AM strategies. When the probability of feints is higher than 80%, the resilience margin is zero; i.e., the optimal and the default AM strategies both induce high risks. However, as the probability of feint decreases, the resilience margin increases to around \$500; i.e., the default strategy can moderately reduce the risk



(a) Low-cost feints $\hat{c}_{FE} = 1$.

(b) High-cost feints $\hat{c}_{FE} = 5$.

Figure 12.10: IDoS risk vs. $\eta_{FE} \in [0, 1]$ under the optimal and the default AM strategies in red and blue, respectively. The black line represents the resilience margin.

but the optimal strategy can excessively reduce the risk.

Remark 36 (**Half-Truth Attack for High-Cost Feints**). *As shown in Fig. 12.10b, when the feints are of high cost, i.e., $\hat{c}_{FE} = \hat{c}_{RE}/2$, then the optimal attack strategy is to deceive with half-truth, i.e., generating feint and real attacks with approximately equal probability to induce the maximum IDoS risk. As the probability of feints decreases from $\eta_{FE} = 1$, the risk increases significantly under the default AM strategy but moderately under the optimal one.*

The figures in Fig. 12.10 show that the optimal attack strategy under the budget constraint (12.12) needs to adapt to the cost of feint generation. Regardless of the attack strategy, the optimal AM strategy can reduce the risk and achieve a positive resilient margin for all category labels $(s_{SO,i}, s_{CR,j}), i \in \{P, C\}, j \in \{L, H\}$. Moreover, higher feint generation cost reduces the arrival frequency of IDoS attacks due to (12.12). Thus, comparing to Fig. 12.10a, the risk in Fig. 12.10b is lower for the same η_{FE} under the optimal or the default AM strategies, especially when η_{FE} is close to 1.

The Operator's Attention Capacity

We consider the following attention function $f_{LOE} \circ f_{SL}$ with a constant attention threshold, i.e., $\bar{n}(y_{EL}, s^k) = \bar{n}_0, \forall y_{EL}, s^k \in \mathcal{S}$. Consider the following trapezoid attention function. If $n^t \leq \bar{n}_0$, the LOE $\omega^t = 1$; i.e., the operator can retain the high LOE when the number of distractions is less than the attention threshold $\bar{n}_0$. If $n^t > \bar{n}_0$, the LOE ω^t gradually decreases as n^t increases. Then, a larger value of $\bar{n}_0$ indicates a high attention capacity. We investigate how the value of $\bar{n}_0$ affects the risk in Fig. 12.11.

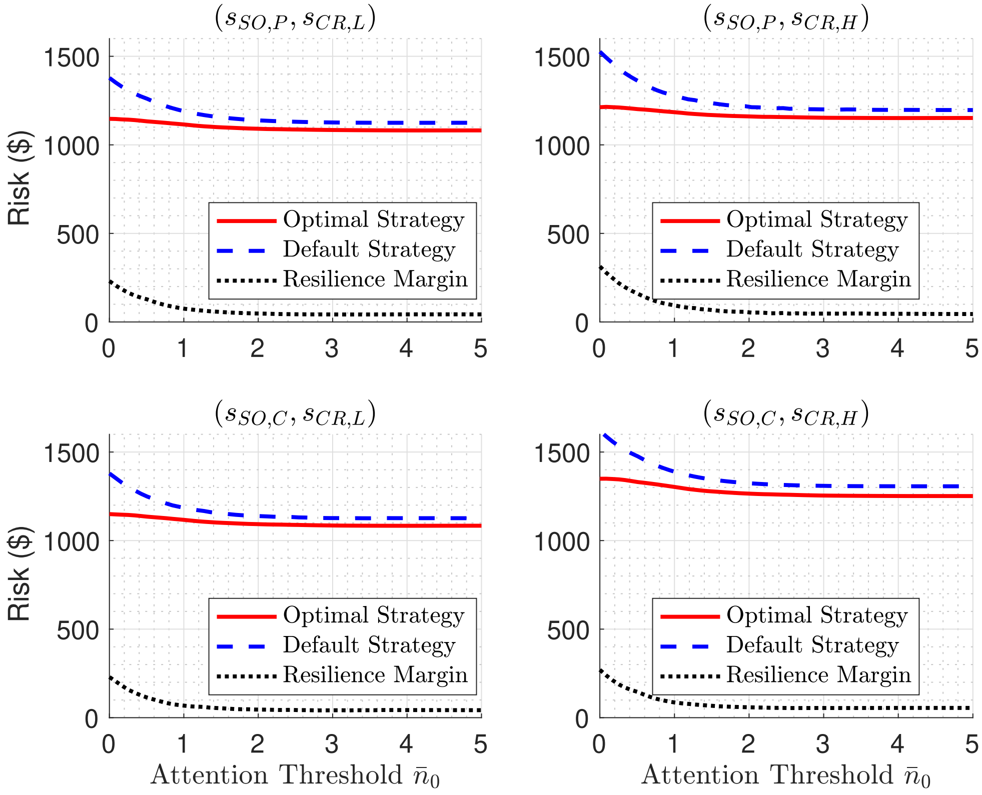


Figure 12.11: Risk vs. attention threshold under the optimal and the default AM strategies in red and blue, respectively. The black dotted line represents the resilience margin.

As the operator's attention capacity increases, the risks under the optimal and the default AM strategies decrease for all category labels. The resilience margin decreases from around \$200 to \$50 as $\bar{n}_0$ increases from 0 to 2 and then maintains the value of around \$50. Thus, the optimal strategy suits operators with a large range of attention capacity, especially for the ones with limited attention capacity.

Part VII

Discussions

Chapter 13

Insights and Future Directions

In this dissertation, we have explored the methodologies of Defense through AI-powered SYstem-scientific methods (DAISY) to mitigate three classes of vulnerabilities related to posture (Part II), information (Part III and IV), and human (Part V and VI) in CPSs. They have laid a solid foundation of 5G-SP established in Section 1.4.1 for high-confidence CPSs. In Section 13.1, we summarize the challenges addressed by this dissertation, its contributions to the six transformations of DAISY in Section 1.4.2, and the future works. In Section 13.2, we go beyond these works to discuss our visions and broader insights.

13.1 Conclusions and Future Works

We have learned from [101] that posture-related defense technologies have been extensively studied, while the mitigation solutions for information-related and human-induced vulnerabilities are underdeveloped. Due to the necessity and urgency to mitigate these three classes of vulnerabilities, it is a promising direction to explore further and pay more attention to the information-related and human-related vulnerabilities. In the following five subsections, we summarize Chapters 3 to 12 and their related research opportunities.

13.1.1 Posture-Related Vulnerability

In Chapter 3, we have focused on addressing the challenges of large-scale interdependence and the SoS protection under resource constraints. For the first challenge, we have proposed factored MDP and factored Markov games to exploit the networks' link sparsity and reduce the growth rate from exponential to polynomial. To address the second challenge, we have developed dynamic strategies for the protection-recovery tradeoff, which have enabled a resilient and cost-effective design. Future work would incorporate incomplete monitoring and resource constraints explicitly to capture more practical factors. It would be of interest to understand the impact of the network topology and investigate resilient strategies on structured networks, e.g., chordal networks.

In Chapter 4, the main challenge is that the protection of nuclear power systems needs to be *time-sensitive* and *risk-sensitive*. Thus, we have formulated a finite-horizon Semi-Markov Game (SMG) and developed non-stationary response policies. Probabilistic Risk Assessment (PRA) approaches have been used to identify model parameters under complex scenarios. We expect research in the following directions to improve the proposed method further. First, available data sets and reliable expert judgments can be incorporated to determine the model parameters. Sensitivity analysis can also be performed to assess the effect of parameter uncertainties on the results presented in the case studies. Second, the defender-attacker interaction in the real world is more complex, and many game components (e.g., the system state, and both players' rewards) are of incomplete information, which motivates possible extensions to more sophisticated forms of games. Third, it would be worth investigating how system states related to hardware failure, maintenance, and test affect the players' strategies and the consequence.

13.1.2 Information-Related Vulnerability: Adversarial Deception

In Chapter 5 and Chapter 6, we have focused on addressing cyber and physical deception, respectively, by formulating dynamic Bayesian games and solving Perfect Bayesian Nash Equilibrium (PBNE) for behavior predictions. Both chapters have considered *rational* and *multi-stage* deception and counter-deception; i.e., players aim to achieve their deception goals at minimum cost, and their private types remain unknown to others for all stages. Besides, we have proposed dynamic beliefs to quantify the uncertainty resulting from players' private types. The beliefs are continuously updated to reduce uncertainties and provide a probabilistic detection system. Our models can be broadly applied to scenarios in AI, economy, and social science, where multi-stage interactions occur between multiple agents with incomplete information. Multi-sided non-binary types can be defined based on the scenario, and our iteration algorithm of the forward belief update and the backward policy computation could be further extended for efficient PBNE computations.

The main challenge of game-theoretic approaches is the complexity of identifying utilities and feasible actions of defenders and users at each stage. One future direction to address the challenge would be to develop mechanisms that can automate the synthesis of verifiably correct game-theoretic models. It would alleviate the workload of the system defender and operator. Nevertheless, game theory provides a quantitative and explainable framework to design the proactive defensive response under uncertainties, compared to rule-based and machine-learning-based defense methods, respectively. On the one hand, the rule-based defense is static, and an attack can circumvent it through sufficient effort. On the other hand, machine learning methods require a huge amount of labeled data sets, which can be hard to obtain.

13.1.3 Information-Related Vulnerability: Defensive Deception

In Chapter 7 and Chapter 8, we have developed honeypot strategies to deter lateral movement and obtain threat intelligence, respectively. In Chapter 7, the key insight is that static security does not necessarily ensure long-term security due to the existence of temporal-spatial attack paths when attacks can persistently stay in the system. The defender cannot discover a complete attack path if focusing only on the attack-defense interactions at local stages or for a short period. The stealthiness of the attack is also challenging to deal with as the defender does not know when the attacker has entered the system and where the lateral movement has occurred. The traditional attack graph models show the causal relationship between preconditions and consequences to identify critical attack paths and assess the vulnerability of the assets. We have seen that a dynamic view of attack-defense plays an important role in discovering hidden and unknown attacks. For example, the exploitation of the same vulnerability at different times can sequentially lead to new attack paths. We have used time-expanded graphical models to capture the temporal-spatial relationships to reveal stealthy attack paths and address the reachability question of whether the asset can be compromised within a given time frame.

In Chapter 8, we have studied the use of honeypots for threat intelligence gathering, shifting from the applications of detection and deterrence. The need for understanding *zero-day threats* motivates the development of honeypots with active engagement strategies that go beyond the functionalities of passive information gathering or low-level interactions. As the defender increases the interaction level of the honeypot by emulating more services, performing more functions, and allowing outbound traffic, we have observed the benefits of the proactive honeypots in terms of threat intelligence, longer engagement time, and reduced risk of detection by adversaries. However, we have noted the overhead of resources and the increased risk of the attacker evading the honeypot and compromising the production system. We have seen that legitimate users can inadvertently access the honeypot when they do not have sufficient knowledge to distinguish it. We briefly discuss the challenges and related future directions about Reinforcement Learning (RL) in the honeypot engagement problem in Chapter 8 as follows.

Non-cooperative and Adversarial Learning Environment

The major challenge of learning under the security scenario is that the defender lacks full control of the learning environment, which limits the scope of feasible RL algorithms. In the classical RL task, the learner can choose to start at any state at any time, and repeatedly simulate the path from the target state. In the adaptive honeypot engagement problem, however, the defender can remove attackers but cannot willfully drive them to the target honeypot and oblige them to unveil their attacking behaviors because the true threat information is revealed only when attackers are unaware of the honeypot engagements. Future work could generalize the current framework to an adversarial learning environment where a savvy attacker can detect the honeypot and adopt deceptive behaviors to disrupt the learning process.

Risk Reduction during the Learning Period

Since the learning process is based on samples from real-time interactions, the defender needs to take into account the system's safety and security during the learning period. For example, if the visit and sojourn in the normal zone result in significant losses, we can use the State–Action–Reward–State–Action (SARSA) algorithm [184] to achieve a more conservative learning process in lieu of Q-learning. Other safe RL methods are stated in the survey [58], which are possible directions for future work.

Asymptotic versus Finite-Step Convergence

Since an attacker can choose to terminate the interaction and exit the system, the engagement time with the attacker can be short-lived. Thus, it is critical for the defender to achieve an acceptable outcome of the learning within finite steps, and meanwhile, maintain a good engagement performance in these steps.

Previous works have studied the convergence rate [47] and the non-asymptotic convergence [112, 113] in the MDP setting. For example, [47] has shown a relationship between the convergence rate and the learning rate of Q-learning; [113] has provided the performance bound of the finite-sample convergence rate; [112] has proposed E^3 algorithm which achieves near-optimal solutions with high probability in polynomial time. However, in the honeypot engagement problem, the defender does not know the remaining steps that she can interact with the attacker because the attacker can terminate on his own. Thus, we cannot directly apply the E^3

algorithm which depends on the horizon time. Moreover, since attackers may change their behaviors during the long learning period, the learning algorithm needs to adapt to the changes of SMDP model quickly.

In this preliminary work, we use the ϵ-greedy policy for the trade-off of exploitation and exploration for a finite learning time. The parameter ϵ can be set at a relatively large value so that the learning algorithm persistently adapts to the changes in the environment. On the other hand, the defender can keep a larger discounted factor γ to focus on the immediate investigation reward. If the defender expects a short interaction time; i.e., the attacker is likely to terminate shortly, she can increase the discounted factor in the learning process to adapt to her expectations.

Transfer Learning

In general, the learning algorithm on SMDP converges more slowly than the one on MDP because the sojourn distribution introduces extra randomness. Thus, instead of learning *ab initio*, the defender can reuse the past experience with attackers of similar behaviors to expedite the learning process. This observation motivates the investigation of transfer learning in RL [208], where side-channel information may also be incorporated.

13.1.4 Human-Related Vulnerability: Incentive Design

In Chapter 9 and Chapter 10, we have focused on the information design (i.e., strategic recommendation) and the holistic design (i.e., a policy generator, an incentive modulator, and a trust manipulator), respectively, to affect insiders' incentives and redress their misbehavior.

These two works have contributed to the six transformations of SPs elaborated in Section 1.4.2. First, incentive design is a proactive, affordable, and non-invasive method to mitigate insider threats as it motivates rather than commands an employee to act in the organization's interests. Second, the ZETAR and the duplicity game frameworks have explicitly captured the multi-agent interactions between defenders and attackers, and have provided two sets of socio-technical solutions; i.e., ZETAR integrates audit and recommendation, while duplicity games consolidate honeypots and security policies. Third, both frameworks have contributed to the transformation of SP from empirical to quantitative, automated, and transferable. On the one hand, they have provided formal design paradigms to mitigate insider threats with quantitative performance metrics and provable guarantees. On the other hand, they have yielded key measures for insiders' compliance and the organization's security level.

13.1.5 Human-Related Vulnerability: Bounded Attention

Most of the existing works have taken humans as an independent component in CPSs and aimed to compensate indirectly for the human vulnerability through additional mechanisms. An alternative way is to directly affect the human component and consider an integrated system. Developing security-assistive technologies that directly affect humans is still in its infancy, and it is a promising direction for integrative research. In Chapter 11 and Chapter 12, we have developed two adaptive human-assistive technologies (i.e., ADVERT and RADAMS) to protect humans from reactive and proactive attentional attacks, respectively.

ADVERT in Chapter 11 has addressed the challenge of phishing recognition improvement through enhancing users' attention. On the one hand, ADVERT has laid theoretical foundations to analyze, evaluate, and improve the human attention process in phishing. On the other hand, its effectiveness has been corroborated using a human-subject data set that contains the eye-tracking and survey data of 150 undergraduates reading phishing emails. The results have shown that RL adaption improves the accuracy of phishing recognition from 74.6% to 86%, while the meta-adaptation has further improved the accuracy to 91.5% and 93.7% in less than 3 and 50 tuning stages. The future work would focus on designing a more sophisticated visual support system that can determine when and how to generate visual aids in lieu of a periodic generation. We may also embed ADVERT into VR/AR technologies to mitigate human vulnerabilities under simulated deception scenarios, where the simulated environment can be easily repeated or changed. We would allow the participants to report their confidence levels while they make the security decisions. Finally, there would be an opportunity to incorporate factors (e.g., pressure and incentives) into the design by limiting the participant's vetting time and rewarding correct identification of phishing emails, respectively.

RADAMS in Chapter 12 has addressed the challenge of mitigating a new class of cognitive attacks called IDoS attacks that intentionally generate feint attacks to overload human operators. On the one hand, RADAMS has provided an automated, resilient, and socio-technical technology to manage alerts and human attention against IDoS attacks. On the other hand, it has incorporated Yerkes–Dodson law and the sunk cost fallacy to provide theoretical underpinnings to establish various fundamental limits and understand tradeoffs among crucial human and economic factors. Future works could be conducted in the following three directions. First, we could incorporate the spatial information of attention (e.g., the locations on the monitor screen) to create a temporal-spatial attention model. Second, we could consider evolving human model, in which both an IDoS attacker and the assistive technology learn using different samples/observations and different knowledge/information/decision-making schemes. Third, we could develop coordination technologies to reduce the cognition load by sharing it among multiple human operators. Based on the literature of cognitive science and existing results of human experiments, we could develop detailed models of human attention, reasoning, and risk-perceiving to better characterize human factors in CPS security.

13.2 Visions and Broader Insights

At the end of this dissertation, we present broader insights supported by the results of DAISY and our visions that would be promising in pushing the boundaries of this research further to encompass more impactful real-world applications.

13.2.1 Insights Related to Humans

Human is at the center of many main challenges in developing 5G-SP. A rich literature has focused on threats from cyber and physical domains but oversimplified the impact of the indispensable human factors in CPS security. This dissertation has presented five ways to influence human behaviors to fill the gap of *social cybersecurity*. They are providing incentives (诱之以利), providing disincentives (胁之以威), disclosing information (晓之以理), designing assistive technology (辅之以技), and manipulating emotions, beliefs, perceptions (动之以情).

Mitigating human-related vulnerability through these five ways is a promising research direction worthy of further exploration not just in developing 5G-SP but also in a broader research field such as human factor engineering. The final goal is to lay a theoretical foundation for the *theory of security mind* by characterizing the human mental processes in CPSs to make the above five mitigation methods **R**eliable, **E**xplicable, **A**pproachable, and **L**egible (referred as the **REAL** design principle).

13.2.2 Insights Related to Security Games

Game theory is an appropriate framework for cybersecurity for the following reasons. On the one hand, there are sufficient evidence and incidents of AI-powered cyber attacks, which have shown that the adversaries are increasingly capable of making quick and rational decisions. On the other hand, there is an urgent need for automated and proactive defense mechanisms that would rely on AI and machine learning techniques. The goal of the defense mechanism is to deter attacks or further create difficulties for the adversaries. Hence, in the cyber-physical battlefield, the interaction between attackers and defenders shifts from being *human-versus-human* to *algorithm-versus-algorithm*. The outcomes of the interactions between algorithms ultimately rely on their strategic reasoning through game-theoretic modeling and analysis.

The economic perspective of game theory yields the following two insights for the attacker and the defender, respectively. In particular, the attacker (resp. defender) tends not to attack (resp. defense) if the attack (resp. defense) cost outweighs the attack (resp. defense) gain.

Security by Design vs. Security by Defense

Security is a much broader concept than detection and prevention. The following ancient philosophy from Sunzi's the art of war [214] purports alternative deterrence strategies that can be applied to cyber-physical security.

- "*Hence to fight and conquer in all your battles is not supreme excellence; supreme excellence consists in breaking the enemy's resistance without fighting.*" (是故百战百胜，非善之善也；不战而屈人之兵，善之善者也)
- "*Thus the highest form of generalship is to balk the enemy's plans; the next best is to prevent the junction of the enemy's forces; the next in order is to attack the enemy's army in the field; the worst policy of all is to besiege walled cities.*" (故上兵伐谋，其次伐交，其次伐兵，其下攻城)

When attacks are intelligent and strategically respond to the defense methods, the defender can proactively deter attacks by designing attackers' cost, information, and epistemology.

- Designing attackers' cost: The defender has the advantage to design the system structure proactively to make it costly for the attacker to succeed. Examples include cyber DMZ to reduce the attack surface, layered defense and DiD to delay the penetration, and MTD to increase the attacker's cost in identifying the valuable assets. An attack is disincentivized or deterred, or at least is not sustainable, if its cost outweighs its gain in the long run.
- Designing attackers' information: The defender can gain an information advantage by introducing defensive deception techniques, including honeypots and honeyfiles. Such information advantage can increase the uncertainties of the attacks and reduce the attack success rate.
- Manipulating attackers' epistemology: The defender may have the advantage to exploit the human weakness of the adversaries. For example, human

attackers exhibit cognition biases, such as the *framing effect*, *confirmation bias*, and *inattentional blindness*. They become unaware of or misled to ignore signs of defensive deception and fall into honeypots that gather threat intelligence.

Absolute Security vs. Best-Effort Security

Absolute security (i.e., defending all potential attacks at all costs) becomes increasingly challenging to achieve as the CPSs and the attacks become more complex and advanced, respectively. As opposed to absolute security, the defender could pursue best-effort security by designing cost-effective security policies. Since accepting to co-exist and interact with an adversary is inevitable in many circumstances, the defender needs to strategically develop security policies and constantly evaluate the trust. This approach has been in accordance with the zero-trust security philosophy [181] that is strongly advocated for 5G-SP.

In light of this view, the outcome of a security game is no longer binary, i.e., winning or losing, which is zero-sum in game-theory terminology. However, the outcome of the attacker-defender interaction can be Nash Equilibrium (NE). Moreover, as we expand the scope from attack deterrence to utility maximization, we can introduce new mechanisms such as cyber insurance to transfer risk besides mitigate risk.

13.2.3 Insights Related to System-Scientific Approaches

A system is an entity that is made of parts that interact to achieve its design objective. System-thinking or system-level modeling is a way to identify the key interacting parts that contribute to the result within proper boundaries determined by their influence and budget. System-scientific approaches further incorporate scientific tools (e.g., optimization, game theory, AI, and learning) to pinpoint the leverage points of controllable variables to do the most efficient tuning/optimization. We briefly discuss four aspects of system-scientific approaches as follows.

Holistic Modeling and Modular Design

System-level thinking enables holistic modeling to understand the relationship among components of the system besides the components themselves. For example, in Part VI, we have abstracted complex human processes as a human system capable of sensing, decision-making, and acting. We focus on the input (e.g., human attention represented by their eye-gaze behaviors) and output (e.g., human actions and task performances) of the human system to measure its impact on the CPS and develop human-assistive technologies. Such system-level perspectives enable us to incorporate existing psychological findings (e.g., the Yerkes–Dodson law) and avoid the intricate details of human mental processes. The holistic modeling helps us identify *sufficient statistics* to simplify the system design and enables us to attain the optimality of the entire system through decentralized decision-making.

A system-level understanding of the components and their relationships also yields modular design and multi-scale solutions. Modular design is effective as we can use the *divide-and-conquer* approach (e.g., the joint design of the generator, modulator, and manipulator in Chapter 10). They are also customizable as we can replace or redesign each module based on different scenarios (e.g., the joint design of the IDoS attack model, human attention model, and human-assistive technologies in Chapter 12). Multi-scale solutions enable approximations to different granularity levels based on time and budget constraints.

Model-Guided AI

In the era of AI and big data, one may be dubious about the need for models. The answer is affirmative due to the following reasons. First, it is costly to collect high-quality data (e.g., with accurate labels), especially in complex CPS scenarios. Second, it is unclear what data to collect and how to use them, especially when humans are involved. Third, despite their successes and wide applications, many existing machine learning methods (e.g., neural networks) are inadequate in their explainability and theoretical underpinning.

Therefore, there is a need to develop model-guided AI to transition from the *era of big data* to the *era of deep intelligence* with incisive laws and principles. On the one hand, the models and these principles enable explainable and transferable solutions to incorporate practical constraints. On the other hand, they also guide us to collect, use, and analyze data more efficiently. For example, we can incorporate the knowledge of security experts to formulate explainable models. Then, we only need a small dataset to tune the unknown parameters in the model rather than learn the entire model; e.g., we use Bayesian optimization to tune the parameters of RL for attention enhancement and phishing recognition in Chapter 11.

System Science-Supported Technology

Our research bridges between science and technology to create both provable and implementable solutions. We aim to establish foundations, categorizations, and fundamental solutions rather than case-by-case investigations. The proposed methodologies are hence universal for a broad class of problems, and insights from one problem can be transferable to another. For example, we have established a theoretical foundation and formed a quantitative design of insiders' incentives in Chapter 9 instead of taking the path of creating empirical solutions (e.g. awareness training and relieving workload pressure) that are specific for an organization or a particular subpopulation.

The theoretical underpinning entails the creation of the baseline feedback-driven multi-agent framework, the analysis of the fundamental tradeoffs, and the insight-driven learning mechanisms. The essential ideas of feedback architecture, fundamental limits, and distinctive learning mechanisms are broadly applicable to many socio-technical systems. They share the similar features of unanticipated uncertainties of human behaviors, usability-performance tradeoffs, and the absence of exact human models. Nevertheless, they can benefit from the proposed methodologies, which have not only achieved a clean-slate design for insider threat mitigation but most importantly also created a metaphysical view of the socio-technical systems as a result of the system-thinking.

System-Thinking beyond CPS

This dissertation is the epitome of system-thinking, which integrates feedback-thinking, tradeoff-thinking, equilibrium-thinking, and data-thinking. It canvasses perspectives that rise above the traditional realm of engineering and create several concomitant impacts in related fields. One is the system-scientific approach to psychology. In contrast to examining the mind and human behaviors by understanding mechanisms in the white box and the connections within, this dissertation takes a system-level approach to create input-output behaviors using empirical findings and data. The second one is cybersecurity. This dissertation moves away from the laborious and perpetual examinations of the evolving vulnerabilities and security solutions. Instead, it metaphysically abstracts cybersecurity into the fundamental problems of adversarial interactions and examines the dimensions of information, dynamics, and tradeoffs. The equilibrium-thinking metamorphoses cybersecurity solutions from the tactical view of winning and losing into the strategic outlook of long-term planning and design of security mechanisms and policies. The third one is the meta-system theory. CPS is the cynosure of this dissertation, and many theories have been developed for its application domains. It is also a quintessential example of meta-systems or SoS. This dissertation contributes to the meta-system theory by bonding systems of multiple types using diverse system-science concepts and tools. The footprint of this meta-system theory can be found in many emerging applications that go beyond CPSs.

Bibliography

[1] Y. Abbasi, D. Kar, N. D. Sintov, M. Tambe, N. Ben-Asher, D. Morrison, and C. Gonzalez. Know your adversary: Insights for a better adversarial behavioral model. In *CogSci*, 2016.

[2] N. H. Abd Rahim, S. Hamid, M. L. M. Kiah, S. Shamshirband, and S. Furnell. A systematic review of approaches to assessing cybersecurity awareness. *Kybernetes*, 2015.

[3] M. Abdallah, P. Naghizadeh, A. R. Hota, T. Cason, S. Bagchi, and S. Sundaram. Behavioral and game-theoretic security investments in interdependent systems modeled by attack graphs. *IEEE Transactions on Control of Network Systems*, 7(4):1585–1596, 2020.

[4] D. Akhawe and A. P. Felt. Alice in warningland: A large-scale field study of browser security warning effectiveness. In *22nd USENIX Security Symposium (USENIX Security 13)*, pages 257–272, 2013.

[5] J. N. Al-Karaki, A. Gawanmeh, and S. El-Yassami. Gosafe: on the practical characterization of the overall security posture of an organization information system using smart auditing and ranking. *Journal of King Saud University-Computer and Information Sciences*, 2020.

[6] E. Al-Shaer, J. Wei, W. Kevin, and C. Wang. *Autonomous Cyber Deception*. Springer, 2019.

[7] T. Aldemir, M. Stovsky, J. Kirschenbaum, D. Mandelli, P. Bucci, L. Mangan, D. Miller, X. Sun, E. Ekici, S. Guarro, et al. Dynamic reliability modeling of digital instrumentation and control systems for nuclear reactor probabilistic risk assessments. Technical report, U.S. Nuclear Regulatory Commission, 2007.

[8] J. S. Ancker, A. Edwards, S. Nosal, D. Hauser, E. Mauer, and R. Kaushal. Effects of workload, work complexity, and repeated alerts on alert fatigue in a clinical decision support system. *BMC medical informatics and decision making*, 17(1):1–9, 2017.

[9] K. J. Åström. Theory and applications of adaptive control—a survey. *Automatica*, 19(5):471–486, 1983.

[10] R. J. Aumann, M. Maschler, and R. E. Stearns. *Repeated games with incomplete information*. MIT press, 1995.

[11] D. Avis, D. Bremner, and R. Seidel. How good are convex hull algorithms? *Comput Geom*, 7(5-6):265–301, 1997.

[12] T. Ban, N. Samuel, T. Takahashi, and D. Inoue. Combat security alert fatigue with ai-assisted techniques. In *Cyber Security Experimentation and Test Workshop*, pages 9–16, 2021.

[13] V. Barbu, M. Boussemart, and N. Limnios. Discrete-time semi-markov model for reliability and survival analysis. *Communications in Statistics-Theory and Methods*, 33(11):2833–2868, 2004.

[14] T. Basar and G. J. Olsder. *Dynamic noncooperative game theory*, volume 23. Siam, 1999.

[15] G. Bassett, D. Hylender, P. Langlois, A. Pinto, and S. Widup. Data breach investigations report. Technical report, Verizon DBIR Team, 2021.

[16] A. Bathelt, N. L. Ricker, and M. Jelali. Revision of the Tennessee Eastman process model. *IFAC-PapersOnLine*, 48(8):309 – 314, 2015. 9th IFAC Symposium on Advanced Control of Chemical Processes ADCHEM 2015.

[17] I. Baxter. Fake login attack evades logo detection, 2020. https://ironscales.com/blog/fake-login-attack-evades-logo-detection.

[18] D. Bergemann and S. Morris. Information design: A unified perspective. *Journal of Economic Literature*, 57(1):44–95, 2019.

[19] D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-dynamic programming*. Athena Scientific, 1996.

[20] S. Boyd, S. P. Boyd, and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[21] S. J. Bradtke and M. O. Duff. Reinforcement learning methods for continuous-time markov decision problems. In *Advances in neural information processing systems*, pages 393–400, 1995.

[22] A. A. Cárdenas, S. Amin, Z.-S. Lin, Y.-L. Huang, C.-Y. Huang, and S. Sastry. Attacks against process control systems: risk assessment, detection, and response. In *Proceedings of the 6th ACM symposium on information, computer and communications security*, pages 355–366, 2011.

[23] W. A. Casey, Q. Zhu, J. A. Morales, and B. Mishra. Compliance control: Managed vulnerability surface in social-technological systems via signaling games. In *Proceedings of the 7th ACM CCS International Workshop on Managing Insider Security Threats*, pages 53–62, 2015.

[24] M. Chatterjee and A.-S. Namin. Detecting phishing websites through deep reinforcement learning. In *2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC)*, volume 2, pages 227–232. IEEE, 2019.

[25] C. Chen and Y. Xu. The curse of rationality in sequential scheduling games. In *International Conference on Web and Internet Economics*, pages 295–308. Springer, 2020.

[26] D. Chen and K. S. Trivedi. Optimization for condition-based maintenance with semi-markov decision process. *Reliability engineering & system safety*, 90(1):25–29, 2005.

[27] P.-Y. Chen, S. Choudhury, L. Rodriguez, A. Hero, and I. Ray. Enterprise cyber resiliency against lateral movement: A graph theoretic approach. *arXiv preprint arXiv:1905.01002*, 2019.

[28] P. Cichonski, T. Millar, T. Grance, and K. Scarfone. Computer security incident handling guide: recommendations of the national institute of standards and technology. Technical report, National Institute of Standards and Technology, 2012.

[29] F. Cohen. Simulating cyber attacks, defences, and consequences. *Computers & Security*, 18(6):479–518, 1999.

[30] V. Combs. 3 ways criminals use artificial intelligence in cybersecurity attacks, Oct 2020. TechRepublic.

[31] T. M. Corporation. Enterprise matrix, 2019.

[32] E. B. Cox, Q. Zhu, and E. Balcetis. Stuck on a phishing lure: differential use of base rates in self and social judgments of susceptibility to cyber risk. *Comprehensive Results in Social Psychology*, 4(1):25–52, 2020.

[33] L. A. T. Cox Jr et al. Game theory and risk analysis. *Risk Analysis*, 29(8):1062–1068, 2009.

[34] V. P. Crawford and J. Sobel. Strategic information transmission. *Econometrica: Journal of the Econometric Society*, pages 1431–1451, 1982.

[35] R. Dahbul, C. Lim, and J. Purnama. Enhancing honeypot deception capability through network service fingerprinting. In *Journal of Physics: Conference Series*, volume 801, page 012057. IOP Publishing, 2017.

[36] R. Denning and V. Mubayi. Insights into the societal risk of nuclear power plant accidents. *Risk analysis*, 37(1):160–172, 2017.

[37] D. Department of Homeland Security. Nsa/css technical cyber threat framework v2 a report from: Cybersecurity operations the cybersecurity products and sharing division. Technical report, 2018.

[38] R. Dhamija, J. D. Tygar, and M. Hearst. Why phishing works. In *Proc. of the SIGCHI conference on Human Factors in computing systems*, pages 581–590, 2006.

[39] DHS. Roadmap to enhance cyber systems security in the nuclear sector, 2012.

[40] L. Duenas-Osorio and S. M. Vemuru. Cascading failures in complex infrastructure systems. *Structural safety*, 31(2):157–167, 2009.

[41] M. Dufresne. Putting the MITRE ATT&CK evaluation into context, 2018.

[42] C. Dukes. Committee on national security systems (cnss) glossary. *CNSSI, Fort 1322 Meade, MD, USA, Tech. Rep*, 1323, 2015.

[43] B. Edwards, A. Furnas, S. Forrest, and R. Axelrod. Strategic aspects of cyberattack, attribution, and blame. *Proceedings of the National Academy of Sciences*, 114(11):2825–2830, 2017.

[44] S. Egelman, L. F. Cranor, and J. Hong. You've been warned: an empirical study of the effectiveness of web browser phishing warnings. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1065–1074, 2008.

[45] D. Evans, A. Nguyen-Tuong, and J. Knight. Effectiveness of moving target defenses. In *Moving target defense*, pages 29–48. Springer, 2011.

[46] K. Evans, A. Abuadbba, M. Ahmed, T. Wu, M. Johnstone, and S. Nepal. Raider: Reinforcement-aided spear phishing detector. *arXiv preprint arXiv:2105.07582*, 2021.

[47] E. Even-Dar and Y. Mansour. Learning rates for q-learning. *Journal of Machine Learning Research*, 5(Dec):1–25, 2003.

[48] N. Falliere, L. O. Murchu, and E. Chien. W32. stuxnet dossier. Technical report, Symantec Corp., 2011.

[49] X. Feng, Z. Zheng, P. Hu, D. Cansever, and P. Mohapatra. Stealthy attacks meets insider threats: a three-player game model. In *MILCOM 2015-2015 IEEE Military Communications Conference*, pages 25–30. IEEE, 2015.

[50] J. Filar and K. Vrieze. *Competitive Markov decision processes*. Springer Science & Business Media, 2012.

[51] FireEye. Advanced Persistent Threat Groups — FireEye, 2017.

[52] L. Franklin, M. Pirrung, L. Blaha, M. Dowling, and M. Feng. Toward a visualization-supported workflow for cyber alert management using threat models and human-centered design. In *2017 IEEE Symposium on Visualization for Cyber Security (VizSec)*, pages 1–8. IEEE, 2017.

[53] P. I. Frazier. Bayesian optimization. In *Recent Advances in Optimization and Modeling of Contemporary Problems*, pages 255–278. INFORMS, 2018.

[54] D. Fridovich-Keil, V. Rubies-Royo, and C. J. Tomlin. An iterative quadratic method for general-sum differential games with feedback linearizable dynamics. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2216–2222, 2020.

[55] I. Friedberg, F. Skopik, G. Settanni, and R. Fiedler. Combating advanced persistent threats: From network event correlation to incident detection. *Computers & Security*, 48:35–57, 2015.

[56] A. Friedman. *Differential games*. Courier Corporation, 2013.

[57] X. Fu and Y. Yang. Modeling and analyzing cascading failures for internet of things. *Information Sciences*, 545:753–770, 2021.

[58] J. Garcıa and F. Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(1):1437–1480, 2015.

[59] D. Gertman, H. Blackman, and C. Marble, Julie Smith. The spar-h human reliability analysis method. Technical report, U.S. Nuclear Regulatory Commission, 2005.

[60] I. Ghafir, M. Hammoudeh, V. Prenosil, L. Han, R. Hegarty, K. Rabie, and F. J. Aparicio-Navarro. Detection of advanced persistent threat using machine-learning correlation analysis. *Future Generation Computer Systems*, 89:349–359, 2018.

[61] I. Ghafir, K. G. Kyriakopoulos, S. Lambotharan, F. J. Aparicio-Navarro, B. AsSadhan, H. BinSalleeh, and D. M. Diab. Hidden markov models and alert correlations for the prediction of advanced persistent threats. *IEEE Access*, 7:99508–99520, 2019.

[62] I. Ghafir, V. Prenosil, M. Hammoudeh, L. Han, and U. Raza. Malicious ssl certificate detection: A step towards advanced persistent threat defence. In *Proceedings of the International Conference on Future Networks and Distributed Systems*, page 27. ACM, 2017.

[63] A. Ghosh, D. Pendarakis, and W. Sanders. Moving target defense co-chair's report-national cyber leap year summit 2009. *Tech. Rep., Federal Networking and Information Technology Research and Development (NITRD) Program*, 2009.

[64] M. Gil, M. Albert, J. Fons, and V. Pelechano. Engineering human-in-the-loop

interactions in cyber-physical systems. *Information and software technology*, 126:106349, 2020.

[65] F. Greitzer and D. Frincke. Combining traditional cyber security audit data with psychosocial data: towards predictive modeling for insider threat mitigation. In *Insider threats in cyber security*. Springer, 2010.

[66] F. L. Greitzer, J. Strozer, S. Cohen, J. Bergey, J. Cowley, A. Moore, and D. Mundie. Unintentional insider threat: contributing factors, observables, and mitigation strategies. In *2014 47th Hawaii International Conference on System Sciences*, pages 2025–2034. IEEE, 2014.

[67] E. R. Griffor, C. Greer, D. A. Wollman, M. J. Burns, et al. Framework for cyber-physical systems: Volume 1, overview. 2017.

[68] B. Guembe, A. Azeta, S. Misra, V. Osamor, L. Fernandez-Sanz, and V. Pospelova. The emerging threat of ai-driven cyber attacks: A review. *Applied Artificial Intelligence*, pages 1–34, 2022.

[69] C. Guestrin, D. Koller, R. Parr, and S. Venkataraman. Efficient solution algorithms for factored MDPs. *Journal of Artificial Intelligence Research*, 19:399–468, 2003.

[70] H. Guo, C. Zheng, H. H.-C. Iu, and T. Fernando. A critical review of cascading failure analysis and modeling of power system. *Renewable and Sustainable Energy Reviews*, 80:9–22, 2017.

[71] A. Hagberg, A. Kent, N. Lemons, and J. Neil. Credential hopping in authentication graphs. In *2014 International Conference on Signal-Image Technology Internet-Based Systems (SITIS)*. IEEE Computer Society, Nov. 2014.

[72] H. Hajieghrary, D. Kularatne, and M. A. Hsieh. Cooperative transport of a buoyant load: A differential geometric approach. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2158–2163, 2017.

[73] M. A. Haque, G. K. De Teyou, S. Shetty, and B. Krishnappa. Cyber resilience framework for industrial control systems: concepts, metrics, and insights. In *2018 IEEE international conference on intelligence and security informatics (ISI)*, pages 25–30. IEEE, 2018.

[74] S. Harris. Insider threat mitigation guide. Technical report, Cybersecurity and Infrastructure Security Agency.

[75] J. C. Harsanyi. Games with incomplete information played by "Bayesian" players, i–iii part i. the basic model. *Management science*, 14(3):159–182, 1967.

[76] J. Hofbauer, K. Sigmund, et al. *Evolutionary games and population dynamics.* Cambridge university press, 1998.

[77] C. D. Holland and O. V. Komogortsev. Complex eye movement pattern biometrics: the effects of environment and stimulus. *IEEE Transactions on Information Forensics and Security*, 8(12):2115–2126, 2013.

[78] K. Horák, B. Bošanskỳ, P. Tomášek, C. Kiekintveld, and C. Kamhoua. Optimizing honeypot strategies against dynamic lateral movement using partially observable stochastic games. *Computers & Security*, 87:101579, 2019.

[79] H. Hu, Y. Liu, C. Chen, H. Zhang, and Y. Liu. Optimal decision making approach for cyber security defense using evolutionary game. *IEEE Transactions on Network and Service Management*, 17(3):1683–1700, 2020.

[80] Q. Hu and W. Yue. *Markov decision processes with their applications*, volume 14. Springer Science & Business Media, 2007.

[81] K. Huang, C. Zhou, Y.-C. Tian, S. Yang, and Y. Qin. Assessing the physical impact of cyberattacks on industrial cyber-physical systems. *IEEE Transactions on Industrial Electronics*, 65(10):8153–8162, 2018.

[82] L. Huang, J. Chen, and Q. Zhu. A factored MDP approach to optimal mechanism design for resilient large-scale interdependent critical infrastructures. In *2017 Workshop on Modeling and Simulation of Cyber-Physical Energy Systems (MSCPES)*, pages 1–6. IEEE, 2017.

[83] L. Huang, J. Chen, and Q. Zhu. A large-scale markov game approach to dynamic protection of interdependent infrastructure networks. In *International Conference on Decision and Game Theory for Security*, pages 357–376. Springer, Cham, 2017.

[84] L. Huang, J. Chen, and Q. Zhu. Distributed and optimal resilient planning of large-scale interdependent critical infrastructures. In *2018 Winter Simulation Conference (WSC)*, pages 1096–1107. IEEE, 2018.

[85] L. Huang, J. Chen, and Q. Zhu. Factored markov game theory for secure interdependent infrastructure networks. In *Game Theory for Security and Risk Management*, pages 99–126. Birkhäuser, Cham, 2018.

[86] L. Huang, S. Jia, E. Balcetis, and Q. Zhu. Advert: An adaptive and data-driven attention enhancement mechanism for phishing prevention. *arXiv preprint arXiv:2106.06907*, 2021.

[87] L. Huang and Q. Zhu. Analysis and computation of adaptive defense strategies against advanced persistent threats for cyber-physical systems. In *International Conference on Decision and Game Theory for Security*, pages 205–226. Springer, Cham, 2018.

[88] L. Huang and Q. Zhu. Adaptive honeypot engagement through reinforcement learning of semi-markov decision processes. In *International Conference on Decision and Game Theory for Security*, pages 196–216. Springer, 2019.

[89] L. Huang and Q. Zhu. Adaptive strategic cyber defense for advanced persistent threats in critical infrastructure networks. In *ACM SIGMETRICS Performance Evaluation Review*, volume 46, pages 52–56. ACM, 2019.

[90] L. Huang and Q. Zhu. Dynamic bayesian games for adversarial and defensive cyber deception. In E. Al-Shaer, J. Wei, K. W. Hamlen, and C. Wang, editors, *Autonomous Cyber Deception: Reasoning, Adaptive Planning, and Evaluation of HoneyThings*, pages 75–97. Springer International Publishing, Cham, 2019.

[91] L. Huang and Q. Zhu. A dynamic games approach to proactive defense strategies against advanced persistent threats in cyber-physical systems. *Computers & Security*, 89:101660, 2020.

[92] L. Huang and Q. Zhu. Farsighted risk mitigation of lateral movement using dynamic cognitive honeypots. In *International Conference on Decision and Game Theory for Security*, pages 125–146. Springer, Cham, 2020.

[93] L. Huang and Q. Zhu. Strategic learning for active, adaptive, and autonomous cyber defense. In *Adaptive Autonomous Secure Cyber Systems*, pages 205–230. Springer, Cham, 2020.

[94] L. Huang and Q. Zhu. Combating informational denial-of-service (idos) attacks: Modeling and mitigation of attentional human vulnerability. In *International Conference on Decision and Game Theory for Security*, pages 314–333. Springer, Cham, 2021.

[95] L. Huang and Q. Zhu. Convergence of bayesian nash equilibrium in infinite bayesian games under discretization. *arXiv preprint arXiv:2102.12059*, 2021.

[96] L. Huang and Q. Zhu. Duplicity games for deception design with an application to insider threat mitigation. *IEEE Transactions on Information Forensics and Security*, 16:4843–4856, 2021.

[97] L. Huang and Q. Zhu. A dynamic game framework for rational and persistent robot deception with an application to deceptive pursuit-evasion. *IEEE Transactions on Automation Science and Engineering*, 2021.

[98] L. Huang and Q. Zhu. Radams: Resilient and adaptive alert and attention management strategy against informational denial-of-service (idos) attacks. *arXiv preprint arXiv:2111.03463*, 2021.

[99] L. Huang and Q. Zhu. Zetar: Modeling and computational design of strategic and adaptive compliance policies. *arXiv preprint arXiv:2204.02294*, 2022.

[100] Y. Huang, J. Chen, L. Huang, and Q. Zhu. Dynamic games for secure and resilient control system design. *National Science Review*, 7(7):1125–1141, 2020.

[101] Y. Huang, L. Huang, and Q. Zhu. Reinforcement learning for feedback-enabled cyber resilience. *Annual Reviews in Control*, 2022.

[102] A. Humayed, J. Lin, F. Li, and B. Luo. Cyber-physical systems security—a survey. *IEEE Internet of Things Journal*, 4(6):1802–1831, 2017.

[103] E. Humphreys. *Implementing the ISO/IEC 27001: 2013 ISMS Standard*. Artech House, 2016.

[104] J. Hunker and C. W. Probst. Insiders and insider threats-an overview of definitions and mitigation techniques. *J. Wirel. Mob. Networks Ubiquitous Comput. Dependable Appl.*, 2(1):4–27, 2011.

[105] E. M. Hutchins, M. J. Cloppert, and R. M. Amin. Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. *Leading Issues in Information Warfare & Security Research*, 1(1):80, 2011.

[106] S. Jajodia, A. K. Ghosh, V. Swarup, C. Wang, and X. S. Wang. *Moving target defense: creating asymmetric uncertainty for cyber threats*, volume 54. Springer Science & Business Media, 2011.

[107] M. P. Janisse. Pupil size and affect: A critical review of the literature since 1960. *Canadian Psychologist/Psychologie canadienne*, 14(4):311, 1973.

[108] D. Kahneman. *Thinking, fast and slow*. Macmillan, 2011.

[109] E. Kamenica and M. Gentzkow. Bayesian persuasion. *American Economic Review*, 101(6):2590–2615, 2011.

[110] O. E. Kang, K. E. Huffer, and T. P. Wheatley. Pupil dilation dynamics track attention to high-level information. *PloS one*, 9(8):e102463, 2014.

[111] C. Katsini, Y. Abdrabou, G. E. Raptis, M. Khamis, and F. Alt. The role of eye gaze in security and privacy applications: survey and future hci research directions. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–21, 2020.

[112] M. Kearns and S. Singh. Near-optimal reinforcement learning in polynomial time. *Machine learning*, 49(2-3):209–232, 2002.

[113] M. J. Kearns and S. P. Singh. Finite-sample convergence rates for q-learning and indirect algorithms. In *Advances in neural information processing systems*, pages 996–1002, 1999.

[114] W. Keller and M. Modarres. A historical overview of probabilistic risk assessment development and its use in the nuclear power industry: a tribute to the late professor norman carl rasmussen. *Reliability Engineering & System Safety*, 89(3):271–285, 2005.

[115] S. M. Khattab, C. Sangpachatanaruk, D. Mossé, R. Melhem, and T. Znati. Roaming honeypots for mitigating service-level denial-of-service attacks. In *24th International Conference on Distributed Computing Systems, 2004. Proceedings.*, pages 328–337. IEEE, 2004.

[116] D. E. Kirk. *Optimal control theory: an introduction*. Courier Corporation, 2004.

[117] M. Korkali, J. G. Veneman, B. F. Tivnan, and P. D. Hines. Reducing

cascading failure risk by increasing infrastructure network interdependency. *arXiv preprint arXiv:1410.6836*, 2014.

[118] N. Krawetz. Anti-honeypot technology. *IEEE Security & Privacy*, 2(1):76–79, 2004.

[119] J. L. Kröger, O. H.-M. Lutz, and F. Müller. What does your gaze reveal about you? on the privacy implications of eye tracking. In *IFIP International Summer School on Privacy and Identity Management*, pages 226–241. Springer, 2019.

[120] M. Krotofil and A. A. Cárdenas. Resilience of process control systems to cyber-physical attacks. In *Nordic Conference on Secure IT Systems*, pages 166–182. Springer, 2013.

[121] K. Lalropuia and V. Gupta. Modeling cyber-physical attacks based on stochastic game and markov processes. *Reliability Engineering & System Safety*, 181:28–37, 2019.

[122] R. Langner. Stuxnet: Dissecting a cyberwarfare weapon. *IEEE Security & Privacy*, 9(3):49–51, 2011.

[123] E. E. Lee II, J. E. Mitchell, and W. A. Wallace. Restoration of services in interdependent infrastructure systems: A network flows approach. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 37(6):1303–1317, 2007.

[124] D. Li and J. B. Cruz. Defending an asset: A linear quadratic game approach. *IEEE Transactions on Aerospace and Electronic Systems*, 47(2):1026–1044, 2011.

[125] P. Li, X. Yang, Q. Xiong, J. Wen, and Y. Y. Tang. Defending against the advanced persistent threat: An optimal control approach. *Security and Communication Networks*, 2018, 2018.

[126] T. Li and J. Horkoff. Dealing with security requirements for socio-technical systems: A holistic approach. In *International Conference on Advanced Information Systems Engineering*, pages 285–300. Springer, 2014.

[127] D. J. Liebling and S. Preibusch. Privacy considerations for a pervasive eye tracking world. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication*, pages 1169–1177, 2014.

[128] E. Lin, S. Greenberg, E. Trotter, D. Ma, and J. Aycock. Does domain highlighting help people identify phishing sites? In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 2075–2084, 2011.

[129] G. Lingam, R. R. Rout, and D. V. Somayajulu. Deep q-learning and particle swarm optimization for bot detection in online social networks. In *2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, pages 1–6. IEEE, 2019.

[130] Y. Liu, P. Ning, and M. K. Reiter. False data injection attacks against state estimation in electric power grids. *ACM Transactions on Information and System Security (TISSEC)*, 14(1):1–33, 2011.

[131] P. I. LLC. 2018 cost of data breach study, 2018.

[132] J. Löfberg. Yalmip : A toolbox for modeling and optimization in MATLAB. In *In Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004.

[133] G. Loukas. *Cyber-physical attacks: A growing invisible threat.* Butterworth-Heinemann, 2015.

[134] K.-w. Lye and J. M. Wing. Game strategies in network security. *International Journal of Information Security*, 4(1-2):71–86, 2005.

[135] J. W. Mamer. Successive approximations for finite horizon, semi-markov decision processes with application to asset liquidation. *Operations Research*, 34(4):638–644, 1986.

[136] M. Marchetti, F. Pierazzi, M. Colajanni, and A. Guido. Analysis of high volumes of network traffic for advanced persistent threat detection. *Computer Networks*, 109:127–141, 2016.

[137] A. Marotta, F. Martinelli, S. Nanni, A. Orlando, and A. Yautsiukhin. Cyber-insurance survey. *Computer Science Review*, 24:35–61, 2017.

[138] J. McAlaney and P. J. Hills. Understanding phishing email processing and perceived trustworthiness through eye tracking. *Front. Psychol.*, 11:1756, 2020.

[139] P. Mell, K. Scarfone, and S. Romanosky. Common vulnerability scoring system. *IEEE Security & Privacy*, 4(6):85–89, 2006.

[140] B. I. Messaoud, K. Guennoun, M. Wahbi, and M. Sadik. Advanced persistent threat: New analysis driven by life cycle phases and their challenges. In *2016 International Conference on Advanced Communication Systems and Information Security (ACOSIS)*, pages 1–6. IEEE, 2016.

[141] S. M. Milajerdi and M. Kharrazi. A composite-metric based path selection technique for the tor anonymity network. *Journal of Systems and Software*, 103:53–61, 2015.

[142] S. Miserendino, C. Maynard, and J. Davis. Threatvectors: Contextual workflows and visualizations for rapid cyber event triage. In *2017 International Conference On Cyber Incident Response, Coordination, Containment & Control (Cyber Incident)*, pages 1–8. IEEE, 2017.

[143] K. D. Mitnick and W. L. Simon. *The art of deception: Controlling the human element of security.* John Wiley & Sons, 2011.

[144] D. Miyamoto, G. Blanc, and Y. Kadobayashi. Eye can tell: On the correlation between eye movement and phishing identification. In *Int. Conf. on Neural Information Processing*, pages 223–232. Springer, 2015.

[145] N. N. A. Molok, S. Chang, and A. Ahmad. Information leakage through online social networking: Opening the doorway for advanced persistence threats. 2010.

[146] A. Moore, J. Savinda, E. Monaco, J. Moyes, D. Rousseau, S. Perl, J. Cowley, M. Collins, T. Cassidy, N. VanHoudnos, P. Buttles, D. Bauer, and A. Parshall. The critical role of positive incentives for reducing insider threats. Technical Report CMU/SEI-2016-TR-014, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA, 2016.

[147] A. P. Moore, W. Novak, M. Collins, R. Trzeciak, and M. Theis. Effective insider threat programs: understanding and avoiding potential pitfalls. *Software Engineering Institute White Paper, Pittsburgh*, 2015.

[148] S. Morishita, T. Hoizumi, W. Ueno, R. Tanabe, C. Gañán, M. J. van Eeten, K. Yoshioka, and T. Matsumoto. Detect me if you… oh wait. an internet-wide view of self-revealing honeypots. In *2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, pages 134–143. IEEE, 2019.

[149] T. H. Morris and W. Gao. Industrial control system cyber attacks. In *Proceedings of the 1st International Symposium on ICS & SCADA Cyber Security Research*, pages 22–29, 2013.

[150] V. Mubayi, V. Sailor, and G. Anandalingam. Cost-benefit considerations in regulatory analysis. Technical report, Brookhaven National Lab., 1995.

[151] T. Nakagawa. *Stochastic processes: With applications to reliability theory.* Springer Science & Business Media, 2011.

[152] M. Nawrocki, M. Wählisch, T. C. Schmidt, C. Keil, and J. Schönfelder. A survey on honeypot software and data analysis. *arXiv preprint arXiv:1608.06249*, 2016.

[153] NEI. Cyber security plan for nuclear power reactors. Technical report, Nuclear Energy Institute, 2010.

[154] L. H. Newman. Ai wrote better phishing emails than humans in a recent test, Aug 2021. Wired.

[155] R. S. Nickerson. Confirmation bias: A ubiquitous phenomenon in many guises. *Review of general psychology*, 2(2):175–220, 1998.

[156] N. Nisan and A. Ronen. Algorithmic mechanism design. *Games and Economic behavior*, 35(1-2):166–196, 2001.

[157] N. Nissim, A. Cohen, C. Glezer, and Y. Elovici. Detection of malicious pdf files and directions for enhancements: A state-of-the art survey. *Computers & Security*, 48:246–266, 2015.

[158] NRC. Reactor safety study: an assessment of accident risks in u.s. commercial nuclear power plants. appendix i: accident definition and use of event trees. Technical report, U.S. Nuclear Regulatory Commission, 1975.

[159] NRC. Reactor safety study: an assessment of accident risks in u.s. commercial nuclear power plants. appendix v: quantitative results of accident sequences. Technical report, U.S. Nuclear Regulatory Commission, 1975.

[160] NRC. Protection of digital computer and communication systems and networks. https://www.nrc.gov/reading-rm/doc-collections/cfr/part073/part073-0054.html, 2017. accessed: March 21, 2019.

[161] B. Obama. Presidential policy directive 21: critical infrastructure security and resilience (ppd-21)[press release]. the white house, office of the press secretary, 2013.

[162] P. Orlik and H. Terao. *Arrangements of hyperplanes*, volume 300. Springer Science & Business Media, 2013.

[163] H. Orojloo and M. A. Azgomi. A game-theoretic approach to model and quantify the security of cyber-physical systems. *Computers in Industry*, 88:44–57, 2017.

[164] M. Ouyang. Review on modeling and simulation of interdependent critical infrastructure systems. *Reliability engineering & System safety*, 121:43–60, 2014.

[165] M.-E. Paté-Cornell, M. Kuypers, M. Smith, and P. Keller. Cyber risk management for critical infrastructure: a risk analysis model and three case studies. *Risk Analysis*, 38(2):226–241, 2018.

[166] J. Pawlick, J. Chen, and Q. Zhu. istrict: An interdependent strategic trust mechanism for the cloud-enabled internet of controlled things. *arXiv preprint arXiv:1805.00403*, 2018.

[167] J. Pawlick, E. Colbert, and Q. Zhu. Modeling and analysis of leaky deception using signaling games with evidence. *IEEE Transactions on Information Forensics and Security*, 14(7):1871–1886, 2018.

[168] J. Pawlick and Q. Zhu. *Game Theory for Cyber Deception: From Theory to Applications*. Springer Nature, 2021.

[169] K. Pfeffel, P. Ulsamer, and N. H. Müller. Where the user does look when reading phishing mails–an eye-tracking study. In *Int. Conf. on Human-Computer Interaction*, pages 277–287. Springer, 2019.

[170] N. Poolsappasit, R. Dewri, and I. Ray. Dynamic security risk management using bayesian attack graphs. *IEEE Transactions on Dependable and Secure Computing*, 9(1):61–74, 2011.

[171] J. Postel et al. Transmission control protocol. 1981.

[172] F. Pouget, M. Dacier, and H. Debar. White paper: honeypot, honeynet, honeytoken: terminological issues. *Rapport technique EURECOM*, 1275, 2003.

[173] E. Purvine, J. R. Johnson, and C. Lo. A graph-based impact metric for mitigating lateral movement cyber attacks. In *Proceedings of the 2016 ACM Workshop on Automated Decision Making for Active Cyber Defense*, pages 45–52, 2016.

[174] M. L. Puterman. *Markov decision processes: discrete stochastic dynamic programming*. Wiley-Interscience, 205.

[175] N. Ramkumar, V. Kothari, C. Mills, R. Koppel, J. Blythe, S. Smith, and A. L. Kun. Eyes on urls: Relating visual behavior to safety decisions. In *ACM Symposium on Eye Tracking Research and Applications*, pages 1–10, 2020.

[176] C. Reiger, I. Ray, Q. Zhu, and M. A. Haney. Industrial control systems security and resiliency. *Practice and Theory. Springer, Cham*, 2019.

[177] N. L. Ricker. Decentralized control of the tennessee eastman challenge process. *Journal of Process Control*, 6(4):205–221, 1996.

[178] T. Rid and B. Buchanan. Attributing cyber attacks. *Journal of Strategic Studies*, 38(1-2):4–37, 2015.

[179] S. M. Rinaldi, J. P. Peerenboom, and T. K. Kelly. Identifying, understanding, and analyzing critical infrastructure interdependencies. *IEEE control systems magazine*, 21(6):11–25, 2001.

[180] V. Rosato, L. Issacharoff, F. Tiriticco, S. Meloni, S. Porcellinis, and R. Setola. Modelling interdependent infrastructures using interacting dynamical models. *International Journal of Critical Infrastructures*, 4(1-2):63–79, 2008.

[181] S. Rose, O. Borchert, S. Mitchell, and S. Connelly. Zero trust architecture. Technical report, National Institute of Standards and Technology, 2020.

[182] C. Rosenzweig and W. Solecki. Hurricane sandy and adaptation pathways in new york: Lessons from a first-responder city. *Global Environmental Change*, 28:395–408, 2014.

[183] N. C. Rowe, E. J. Custy, and B. T. Duong. Defending cyberspace with fake honeypots. 2007.

[184] G. A. Rummery and M. Niranjan. *On-line Q-learning using connectionist systems*, volume 37. Citeseer, 1994.

[185] D. Sahoo, C. Liu, and S. C. Hoi. Malicious url detection using machine learning: a survey. *arXiv preprint arXiv:1701.07179*, 2017.

[186] F. Salahdine and N. Kaabouch. Social engineering attacks: a survey. *Future Internet*, 11(4):89, 2019.

[187] K. R. Sarkar. Assessing insider threats to information security using technical, behavioural and organisational measures. *information security technical report*, 15(3):112–133, 2010.

[188] N. Saxena, E. Hayes, E. Bertino, P. Ojo, K.-K. R. Choo, and P. Burnap. Impact and key challenges of insider threats on organizations and critical businesses. *Electronics*, 9(9):1460, 2020.

[189] B. Schäfer, D. Witthaut, M. Timme, and V. Latora. Dynamically induced cascading failures in power grids. *Nature communications*, 9(1):1–13, 2018.

[190] B. Schwartz. The paradox of choice: Why more is less. Ecco New York, 2004.

[191] A. Shah, R. Ganesan, S. Jajodia, and H. Cam. A two-step approach to optimal selection of alerts for investigation in a csoc. *IEEE Trans. Inf. Forensics Secur.*, 14(7):1857–1870, 2019.

[192] A. Shah, R. Ganesan, S. Jajodia, and H. Cam. Understanding tradeoffs between throughput, quality, and cost of alert analysis in a csoc. *IEEE Transactions on Information Forensics and Security*, 14(5):1155–1170, 2019.

[193] L. S. Shapley. Stochastic games. *Proceedings of the national academy of sciences*, 39(10):1095–1100, 1953.

[194] S. Sheng, M. Holbrook, P. Kumaraguru, L. F. Cranor, and J. Downs. Who falls for phish? a demographic analysis of phishing susceptibility and effectiveness of interventions. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 373–382, 2010.

[195] L. Shi, Y. Li, T. Liu, J. Liu, B. Shan, and H. Chen. Dynamic distributed honeypot based on blockchain. *IEEE Access*, 7:72234–72246, 2019.

[196] Y. Shoham and K. Leyton-Brown. *Multiagent systems: Algorithmic, game-theoretic, and logical foundations.* Cambridge University Press, 2008.

[197] J. Sigholm and M. Bang. Towards offensive cyber counterintelligence: Adopting a target-centric view on advanced persistent threats. In *2013 European Intelligence and Security Informatics Conference*, pages 166–171. IEEE, 2013.

[198] C. A. Sims. Implications of rational inattention. *Journal of monetary Economics*, 50(3):665–690, 2003.

[199] S. Smadi, N. Aslam, and L. Zhang. Detection of online phishing email using dynamic evolving neural network based on reinforcement learning. *Decision Support Systems*, 107:88–102, 2018.

[200] L. Spitzner. *Honeypots: tracking hackers*, volume 1. Addison-Wesley Reading, 2003.

[201] D. Spooner, G. Silowash, D. Costa, and M. Albrethsen. Navigating the insider threat tool landscape: low cost technical solutions to jump start an insider threat program. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 247–257. IEEE, 2018.

[202] K. Sreenath and V. Kumar. Dynamics, control and planning for cooperative manipulation of payloads suspended by cables from multiple quadrotor robots. In *Robotics: Science and Systems*, 2013.

[203] R. Steinberg and W. I. Zangwill. The prevalence of braess' paradox. *Transportation Science*, 17(3):301–318, 1983.

[204] K. Stouffer, J. Falco, K. Scarfone, et al. Guide to industrial control systems (ics) security. *NIST special publication*, 800(82):16–16, 2011.

[205] K. Stouffer, V. Pillitteri, S. Lightman, M. Abrams, and A. Hahn. Guide to industrial control systems (ics) security. Technical report, National Institute of Standards and Technology, 2015.

[206] S. C. Sundaramurthy, A. G. Bardas, J. Case, X. Ou, M. Wesch, J. McHugh, and S. R. Rajagopalan. A human capital model for mitigating security analyst burnout. In *Eleventh Symposium On Usable Privacy and Security (SOUPS 2015)*, pages 347–359, 2015.

[207] M. Tawarmalani and N. V. Sahinidis. A polyhedral branch-and-cut approach to global optimization. *Mathematical Programming*, 103:225–249, 2005.

[208] M. E. Taylor and P. Stone. Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, 10(Jul):1633–1685, 2009.

[209] C. I. T. Team. Unintentional insider threats: A foundational study. *cahier de recherche CMU/SEI-2013-TN-022, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA*, 18, 2013.

[210] M. D. A. R. Team. Detecting stealthier cross-process injection techniques with windows defender atp: Process hollowing and atom bombing, 2017.

[211] Tessian. The psychology of human error. Technical report, 2020.

[212] R. H. Thaler. Anomalies: The winner's curse. *Journal of economic perspectives*, 2(1):191–202, 1988.

[213] M. Theis, R. Trzeciak, D. Costa, A. Moore, S. Miller, T. Cassidy, and W. Clay. Common sense guide to mitigating insider threats. 2019.

[214] S. Tzu. The art of war. In *Strategic Studies*, pages 63–91. Routledge, 2008.

[215] M. Van Dijk, A. Juels, A. Oprea, and R. L. Rivest. Flipit: The game of "stealthy takeover". *Journal of Cryptology*, 26(4):655–713, 2013.

[216] P. J. Van Laarhoven and E. H. Aarts. Simulated annealing. In *Simulated annealing: Theory and applications*, pages 7–15. Springer, 1987.

[217] Verizon. Vocabulary for event recording and incident sharing (veris), 2017.

[218] G. Wagener, R. State, T. Engel, and A. Dulaunoy. Adaptive and self-configurable honeypots. In *12th IFIP/IEEE International Symposium on Integrated Network Management (IM 2011) and Workshops*, pages 345–352. IEEE, 2011.

[219] O. A. Wahab, J. Bentahar, H. Otrok, and A. Mourad. Resource-aware detection and defense system against multi-type attacks in the cloud: Repeated bayesian stackelberg game. *IEEE Transactions on Dependable and Secure Computing*, 18(2):605–622, 2019.

[220] Z. Wan, J.-H. Cho, M. Zhu, A. H. Anwar, C. Kamhoua, and M. P. Singh. Foureye: Defensive deception against advanced persistent threats via hypergame theory. *IEEE Transactions on Network and Service Management*, 2021.

[221] C. D. Wickens, J. G. Hollands, S. Banbury, and R. Parasuraman. *Engineering psychology and human performance*. Psychology Press, 2015.

[222] L. Xing. Cascading failures in internet of things: review and perspectives on reliability and resilience. *IEEE Internet of Things Journal*, 8(1):44–64, 2020.

[223] A. Xiong, R. W. Proctor, W. Yang, and N. Li. Is domain highlighting actually helpful in identifying phishing web pages? *Hum. Factors*, 59(4):640–660, 2017.

[224] M. M. Yamin, B. Katt, K. Sattar, and M. B. Ahmad. Implementation of insider threat detection system using honeypot based sensors and threat analytics. In *Future of Information and Communication Conference*, pages 801–829. Springer, 2019.

[225] M. M. Yamin, M. Ullah, H. Ullah, and B. Katt. Weaponized ai for cyber attacks. *Journal of Information Security and Applications*, 57:102722, 2021.

[226] L.-X. Yang, P. Li, Y. Zhang, X. Yang, Y. Xiang, and W. Zhou. Effective repair strategy against advanced persistent threat: A differential game approach. *IEEE Transactions on Information Forensics and Security*, 14(7):1713–1728, 2018.

[227] R. D. Yates, Y. Sun, D. R. Brown, S. K. Kaul, E. H. Modiano, and S. Ulukus. Age of information: An introduction and survey. *IEEE J. Sel. Areas Commun.*, 39:1183–1210, 2021.

[228] R. M. Yerkes, J. D. Dodson, et al. The relation of strength of stimulus to rapidity of habit-formation. *Punishment: Issues and experiments*, pages 27–41, 1908.

[229] M. Zhan, Y. Li, X. Yang, W. Cui, and Y. Fan. Nsaps: A novel scheme for network security state assessment and attack prediction. *Computers & Security*, 99:102031, 2020.

[230] J. Zhang and J. Zhuang. Modeling a multi-target attacker-defender game with multiple attack types. *Reliability Engineering & System Safety*, 185:465–475, 2019.

[231] M. Zhang, Z. Zheng, and N. B. Shroff. A game theoretic model for defending against stealthy attacks with limited resources. In *International Conference on Decision and Game Theory for Security*, pages 93–112. Springer, 2015.

[232] T. Zhang, L. Huang, J. Pawlick, and Q. Zhu. Game-theoretic analysis of cyber deception. *Modeling and Design of Secure Internet of Things*, 54:4649815, 2020.

[233] Y. Zhao, L. Huang, C. Smidts, and Q. Zhu. Finite-horizon semi-markov game for time-sensitive attack response and probabilistic risk assessment in nuclear power plants. *Reliability Engineering & System Safety*, 201:106878, 2020.

[234] Y. Zhao, L. Huang, C. Smidts, and Q. Zhu. A game theoretic approach for responding to cyber-attacks on nuclear power plants. *Nuclear Science and Engineering*, 2021.

[235] Q. Zhu and T. Başar. Dynamic policy-based ids configuration. In *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pages 8600–8605. IEEE, 2009.

[236] C. Zimmerman. Ten strategies of a world-class cybersecurity operations center. *The MITRE Corporation*, 2014.