Deep Reinforcement Learning-based Automated Penetration Testing for Active Distribution Networks

Yuanliang Li

A Thesis
In
The Department
of
Concordia Institute for Information Systems Engineering

Presented in Partial Fulfillment of the Requirements
For the Degree of
Doctor of Philosophy (Information and Systems Engineering)
Concordia University
Montréal, Québec, Canada

September 2024

Concordia University
School of Graduate Studies

This is to certify that the thesis prepared

By: **Mr. Yuanliang Li**

Entitled: **Deep Reinforcement Learning-based Automated Penetration Testing for Active Distribution Networks**

and submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy (Information and Systems Engineering)

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the final examining commitee:

Dr. Wen-Fang Xie _____ Chair

Dr. Yufei Tang _____ External Examiner

Dr. Yiming Xiao _____ External to Program

Dr. Mohsen Ghafouri _____ Examiner

Dr. Suryadipta Majumdar _____ Examiner

Dr. Jun Yan _____ Supervisor

Approved by _____
Dr. Farnoosh Naderkhani, Graduate Program Director

_____
Dr. Mourad Debbabi, Dean
Gina Cody School of Engineering and Computer Science

ABSTRACT

Deep Reinforcement Learning-based Automated Penetration Testing for Active Distribution Networks

Yuanliang Li, Ph.D.
Concordia University, 2024

The smart grid is a highly complex cyber-physical system of heterogeneous components with sensory, control, computation, and communication. Due to its complexity, dimensionality, uncertainty, and strong cyber-physical coupling, manually identifying critical vulnerabilities against cyberattacks at infrastructure levels has proven to be challenging. In the information and communication technology (ICT) industry, penetration testing (PT) has demonstrated its efficacy in pinpointing vulnerabilities within information systems through authorized cyberattacks. Building upon the principles of PT, this study delves into exploring effective and efficient PT approaches to discover vulnerabilities for active distribution networks (ADNs) of smart grids based on deep reinforcement learning (DRL) methods.

To overcome the poor efficiency and non-comprehensiveness of common PT in identifying vulnerabilities for an ADN caused by its complex structure and strong cyber-physical coupling, we first propose a DRL-based PT framework and formulate the PT as a Markov decision process (MDP) specifically for the industrial control networks of the ADN. This framework comprehensively considers cyber-physical coupling, realistic cyberattacks, and the physical impacts of ADNs. The framework is applied to model a replay attack scheme on an ADN as the study case, which aims to identify critical attack paths that lead to system voltage violations. Additionally, a co-simulation platform named *GridBattleSim* was developed specifically for DRL-based PT on ADNs, integrating dedicated simulators for different parts of the ADN. The simulation results show the efficacy of DRL-based PT in learning optimal attack paths under varying system conditions and different levels of attack difficulty.

To overcome the limited observability in practical PT scenarios, a partially observable Markov decision process (POMDP) formulation is proposed, which allows the PT agent to learn PT policies under partially observable conditions. To solve the

POMDP and obtain the optimal PT policy, we apply the physical model of the ADN to estimate its full state based on the local observable data captured by the PT agent and then transform the POMDP to an MDP that can be solved by DRL.

Furthermore, to address the sparse reward issue, improve the generalization of reward functions, and improve the interpretability of DRL-based PT, a knowledge-informed AutoPT framework (RM-PT) is proposed, which incorporates cybersecurity domain knowledge based on Reward Machine (RM). We use the lateral movement of PT on ANDs as a case study, where two RMs are designed based on MITRE ATT&CK knowledge base as two PT guidelines. Finally, the deep Q-learning with RM (DQRM) algorithm is applied to train the PT policies. The proposed RM-PT is evaluated under the *CyberBattleSim* platform. The experimental results show that the knowledge-informed PT exhibits a higher training efficiency compared to the PT without knowledge embedding. Furthermore, RMs that incorporate more detailed domain knowledge exhibit superior PT performance compared to RMs with simpler knowledge.

Finally, we also discuss the future directions of this study in terms of domain knowledge integration for AI-powered PT. We anticipate that the methodologies and findings presented in this study can inspire efforts in securing critical infrastructure and closing research gaps for the cybersecurity of smart grids.

Acknowledgments

I would like to express my heartfelt gratitude to all those whose support is essential to the completion of my Ph.D. thesis.

First, I would like to express my sincere gratitude to my supervisor, Prof. Jun Yan, for his unwavering support throughout my doctoral program. Prof. Yan's exceptional patience and extensive knowledge have been invaluable in guiding me through my research endeavors to achieve my Ph.D. goal. I am truly privileged to have had such an outstanding advisor and mentor during my Ph.D. studies.

Besides my supervisor, I would like to thank my committee members, Dr. Yufei Tang, Dr. Ximing Xiao, Dr. Suryadipta Majumdar, and Dr. Mohsen Ghafouri for their insightful comments from various perspectives. My sincere appreciation also goes to Dr. Walter Lucia for his previous participation in my research evaluation.

I also extend my gratitude to my Ericsson mentor, Mohamed Naili, for providing me with valuable guidance and support throughout the Mitcas project during my internship at Ericsson.

In addition, I would like to thank all of my colleagues in the research lab at Concordia University for their insightful discussions, valuable guidance, and the time spent together. I am especially thankful to Dr. Luyang Hou, Dr. Moshfeka Rahman, Hang Du, Juanwei Chen, Pengyi Liao, Chengming Hu, Jeremy Frandon, William Lardier, and Hanzheng Dai.

I also want to thank my master's supervisor, all colleagues, and friends at Hohai University for their kind support. Special thanks to Prof. Kun Ding, Dr. Jinwei Zhang, Yongjie Liu, and Dr. Xiang Chen.

I also want to thank my roommate, Hanlin Chen, who shared a great experience with me for a long time in Montréal.

Last but not least, I want to thank my parents and my wife, who have always been encouraging and supporting me to pursue my Ph.D. dream.

To my mother, the strongest woman I have ever seen.
To my father, the man who is always proud of me.
To my wife, my love in every sense of the word.

Contents

List of Figures

List of Tables

Abbreviations

ADN	Active Distribution Network
AGC	Automatic Generation Control
AMI	Advanced Metering Infrastructure
AutoPT	Automated Penetration Testing
BDD	Bad Data Detector
CA	Contingency Analysis
CPS	Cyber-Physical System
CVR	Conservation Voltage Reduction
DA	Distribution Automation
DER	Distributed Energy Resource
DNN	Deep Neuron Network
DoS	Denial of Service
DQN	Deep Q-Network
DQRM	Deep Q-Learning with Reward Machine
DRL	Deep Reinforcement Learning
ESS	Energy Storage System
EV	Electric Vehicle
FDIA	False Data Injection Attack
GPS	Global Positioning System
ICS	Industrial Control System

ICT	Information and Communication Technology
IDS	Intrusion Detection System
IT	Information Technology
MDP	Markov Decision Process
MITM	Man-in-the-Middle
OLTC	On-Load Tap Changer
OPF	Optimal Power Flow
OSIT	Open-Source Intelligence Technique
OT	Operational Technology
PDC	Phasor Data Concentrator
PLL	Phase Locked Loop
PMU	Phasor Measurement Unit
POMDP	Partially Observable Markov Decision Process
POMDPRM	Partially Observable Markov Decision Process with Reward Machine
PT	Penetration Testing
PV	Photovoltaic
QoS	Quality of Service
QRM	Q-Learning with Reward Machine
RL	Reinforcement Learning
RM-PT	Knowledge-Informed AutoPT Framework based on Reward Machine
SAVMVI	System Average Voltage Magnitude Violation Index
SCADA	Supervisory Control and Data Acquisition
SE	State Estimation
WAMPAC	Wide Area Monitoring Protection and Control
δ	Action Duration
γ	Discount Factor
$\mathcal{A}, a_t$	Action Space, Action

$\mathcal{O}, o_t$	Observation Space, Observation
$\mathcal{S}, s_t$	State Space, State
ϕ_t	Feature Vector
b_t	Belief State
M	Number of sampling during one action duration
N	Number of bus nodes of the power grid
N_P	Number of PMUs deployed in the power grid
N_{VP}	Number of visible PMUs deployed in the power grid
$R(\cdot), \mathcal{R}, r_t$	Reward Function, Reward Machine, Reward

Chapter 1

Introduction

1.1 Background

1.1.1 Smart Grid as a Cyber-Physical System

With the growing pressure of the global energy crisis and climate change, over 120 countries worldwide have committed to achieving net-zero emissions by 2050. This ambitious target necessitates that countries' economies either cease emitting greenhouse gases entirely or offset their emissions through the utilization of carbon-neutral technologies [3]. Among all industries, the energy and power sector holds a pivotal role in a country's economy and bears significant responsibilities in striving towards the goal of net-zero emissions. A key objective of the energy and power sector is to build efficient, reliable, and sustainable power grids for the nation through enhancing energy conversion efficiency, embracing renewable energies, enforcing energy-saving regulations, and maintaining the stability and quality of service (QoS) of grids.

In this context, the rapid development of information and communication technologies (ICT) plays a critical role and contributes dramatically to real-time monitoring, optimal control, energy management, and timely emergency response of power

grids. The extensive integration of ICT combined with the development of distribution generation techniques has gradually transformed conventional power grids into smart grids that embrace many advanced features and applications, such as advanced metering infrastructure (AMI) [4], active distribution networks (ADNs) [5], distribution automation [6], demand-side response [7], virtual power plants (VPPs) [8], transactive energy systems [9], among others.

The physical infrastructure and ICT integration of the smart grid have evolved it into a large-scale cyber-physical system (CPS) [10]. A typical structure of the cyber-physical smart grid is shown in Figure 1.1, which consists of a physical part and a cyber part.

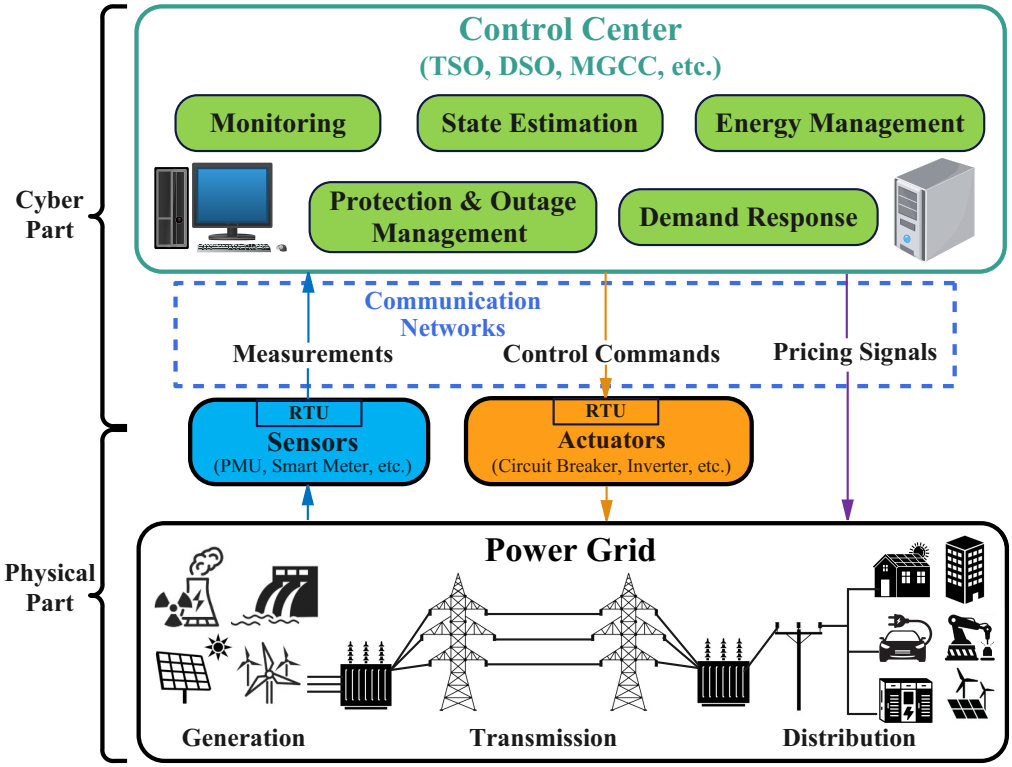


Figure 1.1: The typical structure of the smart grid: an integrative system with cyber and physical parts.

The physical part of the smart grid involves generation systems, transmission systems, and distribution systems [11]. Generation systems produce electricity from

various types of power plants, e.g., hydropower stations, solar farms, wind farms, etc. Transmission systems transport high-voltage electricity from power plants to distribution systems over long distances. Distribution systems encompass the network of medium- and low-voltage power lines, transformers, and other components that deliver electricity from the transmission system to customers (end users). Customers include any entity that consumes energy, e.g., houses, buildings, electric vehicles (EVs), factories, among others.

Furthermore, the advancement of distributed generation techniques has led to an increasing integration of distributed energy resources (DERs), such as rooftop photovoltaic (PV) systems, wind turbines, and energy storage systems (ESSs), in distribution systems. This integration enables power generation on the customer side, transforming the traditional unidirectional power flow into a bidirectional one [12]. It also enhances the flexibility of control and energy management within the grid. Consequently, many advanced power grids, such as active distribution networks (ADNs), microgrids, and virtual power plants (VPPs), have emerged as key players in evolving distribution systems.

Sensors and actuators deployed throughout the grid play as bridges between the cyber and physical parts through remote terminal units (RTUs). Sensors, such as phasor measurement units (PMUs) [13] and smart meters, monitor various electrical parameters of the power grid, including voltage, current, frequency, and energy consumption of customers, among others. Actuators, such as circuit breakers, voltage regulators, etc., carry out control commands (like opening/closing breakers) to control the physical state of the grid.

The cyber part of the smart grid encompasses communication networks and the control center. Communication networks facilitate the transmission of sensor data to the control center and control commands to actuators distributed across the grid. In

addition, pricing signals of current electricity will also be delivered to customers to facilitate the demand response program.

Serving as the brains of smart grids, control centers, also called grid operators, deploy Supervisory Control and Data Acquisition (SCADA) systems [14] to balance supply and demand, regulate voltage and frequency, and support the emergence response of smart grids. Typical tasks of control centers include data monitoring, state estimation, energy management (automatic generation control(AGC), optimal power flow (OPF), voltage-var control (VVC), etc.), demand response, protection, and outage management.

Control centers have multiple control objectives for smart grids: 1) **stability**, which ensures there is just enough power to meet the load demand; 2) **quality**, which ensures the quality of electricity required by the load demand; 3) **economics**, which generates, transmits, and distributes electricity at lowest costs; 4) **reliability**, which ensures uninterruptible delivery of electricity; 5) **sustainability**, which reduces carbon emission and other pollution from smart grids. Based on their control scopes, control centers can be categorized into transmission system operators (TSOs), distribution system operators (DSOs), microgrid control centers (MGCCs), and VPP coordinators, among others.

1.1.2 Security Challenges in Smart Grids

With the pervasive fusion of ICT, the cybersecurity of smart grids has received a great deal of attention. The interconnection of millions of field devices creates an enormous attack surface that could be exploited by prominent threats to the confidentiality, integrity, and availability (CIA) of information systems, as well as the physical security of power systems [15]. Cyberattacks on smart grids could cause physical impacts due to the tight integration between the grid's cyber (control and information systems)

and physical (generation, transmission, and distribution infrastructure) components. Physical impacts of smart grids include interruptions in electricity supply, wide-area power outages, equipment damage, or even safety hazards and casualties.

The infamous 2003 power blackout in the U.S. and Canada is an example of alarming incidences: a software bug led to a series of misoperations in a regional control room in Ohio, which was aggravated by unforeseen cascading physical failures across interconnected power grids of eight states/ provinces, eventually cutting power to over 50 million people with an estimated loss of over $6 billion [16]. The Ukrainian blackout in 2015 caused by spear-phishing emails and BlackEnergy 3 malware targeting power grid SCADA systems finally resulted in power outages to over 225,000 customers for 1-6 hours [17]. In recent years, multiple U.S. critical infrastructure sectors, including energy, nuclear, and commercial facilities, have witnessed a spate of cyberattacks ranging from network reconnaissance to host-based exploitation and denial of service, among other tactics[18].

Emerging DERs and customer-owned smart inverters (ICT-integrated power electronics that convert the DC power produced by DERs to AC power for grid connections [36]) further aggravate the risk of cyberattacks on distribution systems due to many factors. A large number of smart inverters are installed on customer sites, extending the attack surface and making them easier to access, especially when connected to building automation and other public IT networks [37]. DER owners may not have sufficient awareness, expertise, or manpower of the cybersecurity of smart inverters [37]. In addition, third parties, such as smart inverter manufacturers and DER aggregators, often enable remote accesses to monitor, configure, and even directly control the operation of smart inverters. They offer online services like cloud storage, real-time performance evaluation [38], remote maintenance, and fault diagnosis [39], providing critical paths that can be targeted for remote code injection and

Table 1.1: Identified threats and potential impacts to smart grids.

Target	Threat	Impact	Ref.
Synchronous Generators	Aurora attack	Generator damage; power shortage	[19]
Generation control	AGC attack	Disturbance; power shortage	[20]
WAMPAC	GPS spoofing	Misinformation; line outage	[21]
Transmission State Estimation	DoS, FDIA on PMU data	Misinformation; BDD bypassing; OPF performance degradation	[22]
Outage Management	Line outage masking	Voltage violation; line overflow	[23]
ADN CVR	Measurements modification on smart meters	Voltage violations; active power increase	[24]
ADN dynamic reactive power control	Voltage measurements manipulation of transformer	Voltage violation; great power loss	[25]
ADN load shedding	Active/reactive power set-points manipulation	Voltage violation	[26]
ADN under frequency load shedding	Active power set-points manipulation	Frequency excursions; load loss	[27]
Microgrid secondary droop controller	DoS on frequency measurements	Large frequency, voltage, active power oscillations	[28]
Microgrid secondary droop controller	Supplementary active power set-point manipulation	Large frequency, voltage, active power oscillations	[29]
Smart inverter communication interface	Reconnaissance; Replay; DoS; MITM	Inverter shut down	[30, 31, 32]
Smart inverter	Firmware replacement	Inverter disconnection	[33]
Smart inverter	PLL attack through high voltage pulse generators	Output voltage waveform distortion	[34]
Smart inverter	Hall spoofing	Increase in output voltage and output active power	[35]

execution. Additionally, cyberattacks through third parties can influence a portion of smart inverters with the same brand or service, which will probably result in incalculable consequences, e.g., blackouts caused by disconnecting or curtailing a significant portion of the solar generation on a sunny day. Moreover, with the development of module-level power electronics (MLPE) in recent years, smart micro-inverters that control one or a small number of PV panels have started to prevail in distribution systems to improve the output performance of PV arrays [40]; this will significantly increase the number of inverters and communication nodes connected to networks. The small size and resource-constrained nature of micro-inverters will also challenge the security improvement in terms of cost [41].

According to surveys about cyber-physical attacks in smart grids [15] and our investigation on smart inverter security [1], many vulnerabilities and threats, as well as their potential impacts have been identified and investigated in smart grids, as summarized in Table 1.1.

1.1.3 Defense Against Cyberattacks on Smart Grids

Concerned about these cybersecurity challenges in smart grids, extensive efforts have been carried out in the power industry. Among them, the National Institute of Standards and Technology (NIST) proposed the *NISTIR 7628: Guidelines for Smart Grid Cybersecurity*, which presents an analytical framework that organizations can use to develop effective cybersecurity strategies tailored to their particular combinations of smart grid-related characteristics, risks, and vulnerabilities [42]. The National Electric Sector Cybersecurity Organization Resource (NESCOR) described the architecture and identified the cybersecurity requirements for DERs [43] based on the

reference model proposed in the *NISTIR 7628*. NESCOR also introduced cybersecurity failure scenarios and impact analysis for DERs [44]. Sandia National Laboratories (SNL) provided basic principles of cybersecurity, encryption, communication protocols, cybersecurity recommendations and requirements for DERs, and proposed best practices to ensure data confidentiality, integrity, and availability for DER vendors, aggregators, and grid operators [45]. SNL also proposed a five-year roadmap to improve cybersecurity for communication-enabled PV systems [46]. California's Rule 21 Smart Inverter Working Group (SIWG) has provided recommendations for technical requirements for smart inverters with cybersecurity requirements [47]. The National Renewable Energy Laboratory (NREL) proposed the certification process, which contains 11 test cases to verify the data and communication security for DER communication networks [48].

In addition, various defense strategies against cyberattacks on smart grids have been developed in recent years, including attack detection strategies, impact mitigation strategies, and prevention strategies, among others. Attack detection strategies, as illustrated in Figure 1.2, mainly deploy signature-based intrusion detection systems (IDSs) [32, 49], or behavior-based IDSs [41, 50, 51] to detect cyberattacks by analyzing captured data from the cyber part (IP addresses, source ports, event logs, etc.), or from the physical part (voltages, current, active power, reactive power, etc.), or from both parts (i.e., hybrid IDS [52, 53]) of the smart grid.

Impact mitigation strategies refer to strategies that, when cyberattacks occur, the system can actively respond to them and take appropriate actions to reduce adverse effects and protect the grid from system failure [1]. It includes attack-resistant control of field actuators, such as smart inverters [54, 55, 56], to regulate their outputs and keep the electrical parameters (e.g., voltage, current) within acceptable ranges.

Prevention strategies aim to prevent cyberattacks on smart grids. Common

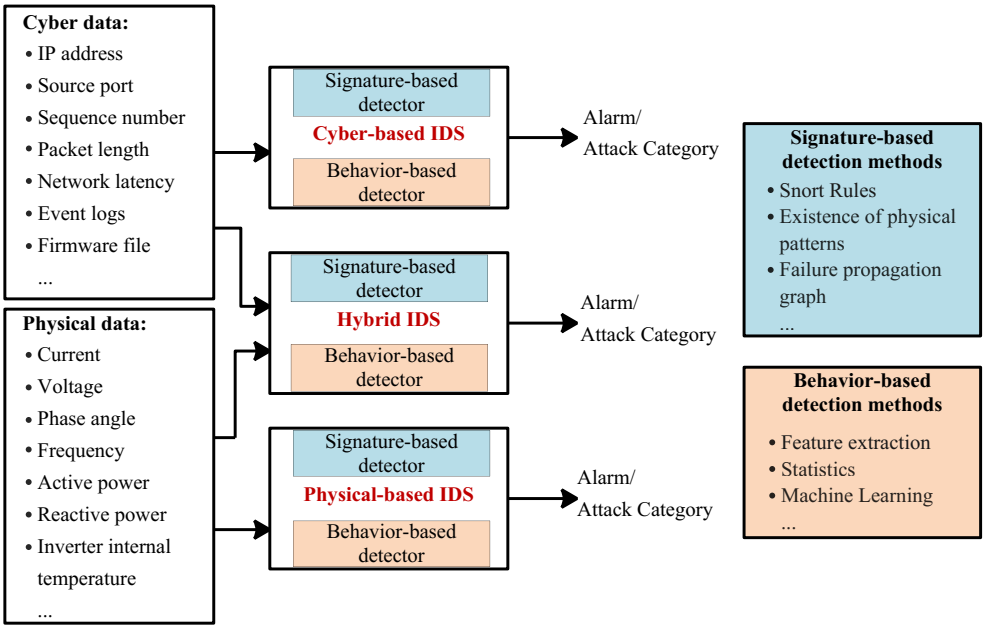


Figure 1.2: Different types of IDS against cyberattacks for smart grids [1].

prevention measures for smart grids include access control [57], implementation of whitelisting and anti-virus software on servers, data encryption [31], segmentation of communication networks [58], and vulnerability identification techniques, such as the penetration testing (PT), among others.

1.1.4 Penetration Testing and Its Challenges for Smart Grids

This study delves into preventive strategies for smart grids, with a particular focus on the theory and application of penetration testing (PT) to identify and analyze vulnerabilities of smart grids in an effective and efficient way. For a smart grid, the vulnerability is considered as potential attack paths that can be executed by adversaries to bring negative impacts on the smart grid, such as the power outage and voltage violations. Here, the PT is an offensive process that stands on adversaries' perspective to discover vulnerabilities at the system level. It serves as a preventive strategy as it is typically conducted prior to the deployment and operation of the

target smart grid, allowing for timely patching of vulnerabilities that are identified.

Vulnerability identification and analysis have been investigated in various domains. Table 1.2 lists common techniques for identifying and analyzing vulnerabilities utilized in IT security, physical safety, and OT security. In the IT industry, PT, widely recognized as ethical hacking, is one of the prevention strategies against cyberattacks. It is adapted to access the security of target digital assets by actively identifying their bugs, weaknesses, or vulnerabilities [59]. This is accomplished by simulating a series of cyberattacks from the viewpoint of adversaries executed by pen-testers, also known as ethical hackers. The scope of PT can encompass individual computers, websites, software, communication networks, or comprehensive information systems.

Table 1.2: Common techniques used for vulnerability identification and analysis in different domains.

Domain	Target	Technique
IT Security	Software, local area networks, etc.	PT, etc.
Physical Safety	Generation / transmission / distribution systems, power electronics, etc.	Contingency analysis, failure assessment, stability analysis, etc.
OT Security	ICS components, e.g., PLC, RTU, SCADA, etc.	PT, etc.

Presently, PT is increasingly becoming a necessary and even mandatory procedure for organizations and businesses in many nations. For instance, organizations in Europe are mandated to establish appropriate technical and organizational measures to ensure the continuous confidentiality, integrity, availability, and resilience of processing applications and services, according to European General Data Protection Regulation (GDPR) Article 32 [60]. Apart from legal obligations, the cybersecurity

community regards PT as the most effective approach to evaluate the effectiveness of security defenses against skilled adversaries and to assess compliance with security policies [61].

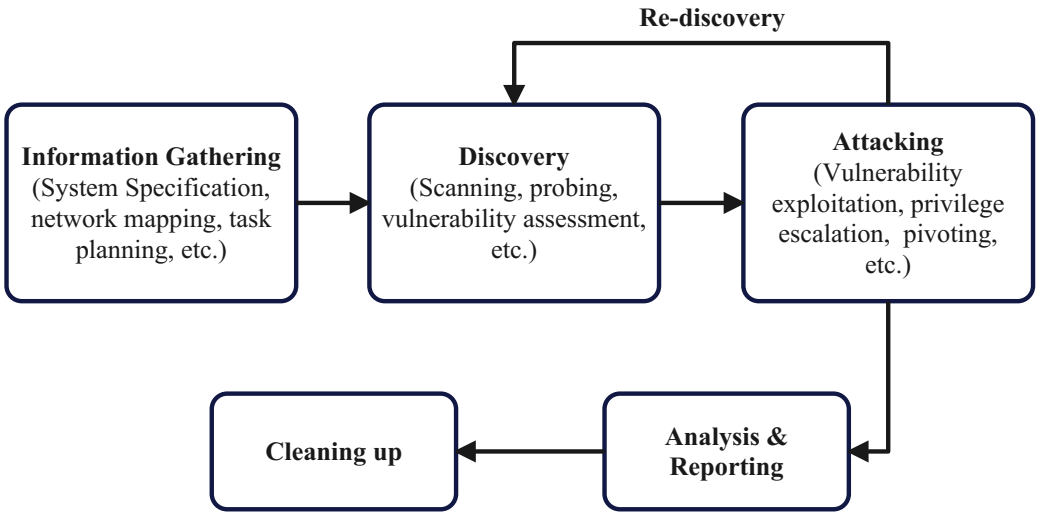


Figure 1.3: Work flow of common PT [2].

In practical terms, PT is a multi-phase process, as illustrated in Figure 1.3, comprising the following phases: information gathering, discovery, attacking, analysis and reporting, and cleaning up [62]. During the information-gathering phase, pen-testers conduct system specifications, identify the reachable systems, define the scope of the PT, and prepare the testing plan. In the discovery phase, pen-testers use vulnerability scanning tools to identify vulnerabilities and compile a list of specific vulnerabilities to focus on. Subsequently, in the attack phase, pen-testers attempt to exploit these vulnerabilities using various attack techniques. Following the attack phase, the analysis and reporting phase becomes necessary, providing an overview of the vulnerabilities explored and exploited and their potential risks to the system. Lastly, the clean-up phase of the PT process involves removing artifacts and restoring any modified configurations made during the PT to ensure the system can resume its regular operation.

PT often requires a high degree of expertise and competence due to the complex characteristics of digital assets such as medium and large networks. As a result, there has been a research emphasis on investigating the possible application of automated tools for different phases of PT, known as AutoPT, which will be reviewed in Chapter 2.1.

Similarly, for the physical safety domain, contingency analysis (CA) [63], failure assessment [64], and stability analysis [65] serve as common techniques to pinpoint potential physical vulnerabilities or failure events in the target physical system. These methods also help to quantitatively or qualitatively evaluate the impacts of failures or vulnerabilities on the target system. In power grids, failure events such as short circuit failure, voltage violation, frequency oscillation, cascading blackout, etc., under the designed physical parameters/configurations/structures and the designed control policies could be identified by adopting these techniques. Equivalent physical models, represented by a set of differential equations (for transient state analysis) [66] or static power flow equations (for static state analysis) [67], are commonly used and analyzed in these techniques for different parts of power grids.

However, existing techniques used in the physical safety domain fail to consider malicious events that can go beyond probable failures or inadvertent misoperations and lack the connection between the cyber and physical layers, rendering them helpful but insufficient for CPS security. Although some studies started to make associations with these two layers, most of these considerations are assumptions without detailed modeling for the cyber part.

Consequently, there is increasing attention to the field of operational technology (OT) security, which commonly considers the interdependence of safety and security issues in vulnerability analysis for CPSs. The testing objectives include components or devices in industrial control systems (ICSs), such as programmable logic

controllers (PLCs), SCADA systems in many CPSs, including building automation systems, transportation systems, and power systems, among others. Thus, OT security is the focal field of this study, and smart grids are our target systems. PT in ICS/OT environments has focused primarily on two main strategies: (1) building physical testbeds or digital twins for the target system to perform offline vulnerability analysis and (2) developing specialized tools to screen for vulnerabilities. Testbeds or digital twins in the first strategy are valuable for identifying device-specific vulnerabilities in CPSs. However, their limited scale compared to real-world environments makes them more suitable for host-level testing rather than comprehensive security assessments of entire CPSs. For the second strategy, specialized tools like PLCScan [68], and SimaticScan [69] can query various data from target PLCs but lack additional functionality, requiring manual assessment or other tools to perform further vulnerability analysis.

To identify critical vulnerabilities that can trigger negative physical impacts on smart grids, both physical and cyber parts should be thoroughly considered. However, due to the large scale, interconnection, and strong cyber-physical coupling of smart grids, the PT task has become highly complicated, which makes it impractical to conduct manual PT at the system level by human experts who rely on exhaustive specification and execution. Even for middle-sized networks, a PT requires extensive repetitive tasks that can take anywhere from a day to a week to complete, resulting in significant downtime that can be detrimental for smart grids with critical services such as electricity supply [2].

Furthermore, hidden or complex vulnerabilities due to cyber-physical coupling, complicated system dynamics, and uncertainties of smart grids should also be investigated under different cyberattack schemes with enriched setups like timing and

order, precisely targeting the most vulnerable targets in the system. Such investigations will have directive significance in the design, operation, and maintenance of the system, to enhance the security and safety of smart grids.

Therefore, automated, effective, and efficient PT approaches or frameworks for smart grids should be investigated considering their nature of complexity and strong cyber-physical coupling. Three key performance indicators for PT, i.e., automaticity, effectiveness, and efficiency, are defined as follows.

Automaticity: The PT is designed to work automatically without or with little human intervention. It is able to gather data from the target smart grid, process and analyze the data, make decisions or predictions, and execute actions automatically to identify vulnerabilities.

Effectiveness: Critical threats and vulnerabilities, including hidden vulnerabilities that can lead to a negative impact on smart grids, can be captured systematically and correctly during the PT.

Efficiency: The PT process can optimize the use of the resources by eliminating time-consuming and irrelevant directions [2].

It should be noted that satisfying the three requirements above perfectly at the same time is challenging, especially for a large-scale cyber-physical smart grid, including generation systems, transmission systems, and distribution systems. To deal with it, the PT can turn to focus on a small area/ sub-task/ aspect of the system. In other words, we can narrow down the space of attack vectors and perform the vulnerability identification in that space. Thus, based on this idea, this study focuses on the PT on ADNs of smart grids, in which the vulnerability identification method can meet the three requirements as much as possible.

1.2 Problem Statement

With the advancement and integration of ICT, smart grids are extensively exposed to the increasing frequency of cyberattacks, creating a significant threat to the stable and reliable operation of smart grids. To protect smart grids from cyberattack, PT is adopted as a preventive strategy. However, the complexity and large scale of smart grids due to the strong cyber-physical coupling challenge the performance of PT for identifying critical or hidden vulnerabilities that can be exploited by adversaries.

One promising solution to automate PT and improve its efficiency is the application of reinforcement learning (RL) or deep reinforcement learning (DRL). RL/DRL is used to handle complicated decision-making tasks by utilizing an intelligent agent that interacts with an unknown environment through trial and error, which has already shown remarkable achievements in various artificial intelligence (AI) applications, including DeepMind's AlphaGo, OpenAI's ChatGPT, autonomous driving, robotics, games, among others [70, 71]. One key advantage of RL/DRL is that the agent does not require prior knowledge of the task, such as the dynamic model or operational mechanism of the system. Instead, the decision-making policy is developed through interactions with the black-box environment, where the agent executes actions and observes feedback from the environment sequentially. Analytical modeling of the operation process for smart grids is challenging due to the need for comprehensive modeling of all components, from the physical to the cyber level, and their interconnection and interoperation. This requires extensive effort from multiple domains, such as control and communication. Meanwhile, according to many works, analytical models can assist in analyzing the vulnerability of a system through CA or stability analysis in a mathematical way; the accuracy of results strongly relies on the precision of the model. This poses a significant challenge, particularly for complex smart grids, not only because of their large-scale and time-varying characteristics but also

because, in most cases, equipment suppliers do not provide device parameters considering intellectual property protection. Therefore, this study explores the potential of applying RL/DRL methods to enhance PT for smart grids.

However, RL/DRL-based PT also encounters several challenges, including poor sampling efficiency, intricate reward specification, and limited interpretability, which are also handled in this study. An overview of the literature on PT based on RL/DRL for information systems and RL/DRL-based vulnerability analysis for smart grids are presented in Chapter 2.

Based on the information provided above, this section outlines the key challenges and problems of PT techniques for smart grids that will be investigated in this study, including the efficiency issue of common PT, the limited observability of PT, and the sparse reward issue of RL/DRL-based PT. Moreover, corresponding solutions will also be briefly summarized.

1.2.1 Problem 1: Poor Efficiency and Non-Comprehensiveness of Common PT

The complexity of today's smart grids makes it impractical to conduct exhaustive manual PT at a system level. The extensive and repetitive PT tasks performed by human experts are time-consuming and labor-costing. In addition, current PT studies focus more on the cyber part of the system. It mainly sets the goal of taking ownership of digital resources in the networks without considering the physical process and physical impact of the CPS. On the contrary, most CA studies for smart grids lack considerations of the cyber layer, where cyber events happening in the communication networks can affect the physical output of the smart grid. Although some studies for CA started to make associations with the cyber level, most of these considerations are assumptions without detailed modeling for the cyber part.

Proposed Solution for Problem 1: We proposed a DRL-based PT framework to efficiently and adaptively identify critical vulnerabilities in cyber-physical ADNs. Using replay attacks on an ADN with conservation voltage reduction (CVR) control as a study case, this work models the pen-tester as a DRL agent and formulates the PT as a Markov Decision Process (MDP) with three actions - stop, record, and replay - to learn the optimal timing and ordering of replay attacks in different operating scenarios. The deep Q-network (DQN) algorithm, serving as the MDP solver, was applied to train agents and optimize the PT policy. In addition, a cyber-physical co-simulation platform, called GridBattleSim, with dedicated simulators for the physical, cyber, control, and attacker aspects of ADNs, was developed as a sandbox environment to train the DRL agent. Scenarios with different levels of difficulty were tested to validate the learning capability and performance to find critical attack paths of the DRL-based PT. The detailed explanation of the proposed solution is presented in Chapter 3

1.2.2 Problem 2: Limited Observability of PT

Although the MDP formulation and DRL application provide a framework for automating PT and improving PT efficiency, its underlying assumption of full system state visibility often proves impractical in real-world scenarios. In practice, a pen-tester aiming to inflict system-wide impact typically only gains access to a subset of the system's digital assets. As a result, the pen-tester is forced to make strategic decisions based on what it can observe from a limited scope of locally available data. The inherent lack of a comprehensive view of the system's state poses a challenge to PT, as it makes the potential impact of outside agent's compromised digital assets invisible. This limitation restricts the agent's ability to make well-informed decisions, thereby impeding the effectiveness of PT strategies.

Proposed Solution for Problem 2: To address these limitations, a partially observable MDP (POMDP) formulation for the PT is proposed, which is an extension of the MDP formulation specifically to deal with the environment with partial observability. Using the same study case as Problem 1, the optimal replay attacks are investigated to trigger the grid voltage violation based on a subset of information collected by the agent. To solve the POMDP, a physical model of the power grid is established for the agent to estimate the full state of the system based on the obtained local data, then transforms the POMDP problem into an MDP problem that can be solved by standard RL/DRL algorithms. The detailed explanation of the proposed solution is presented in Chapter 4.

1.2.3 Problem 3: Sparse-Reward Issues in RL/DRL-based PT

To identify critical vulnerabilities that can be exploited to create a negative impact on smart grids, PT usually takes a long trajectory of searching over a large action space and a large state space of the environment driven by the agent's reward. The sparse-reward issue of RL/DRL-based PT appears when the reward given to the agent only reflects the achievement of the ultimate goal of PT. In other words, the agent will receive a reward of zero value most of the time during PT until the PT goal is achieved. Under such circumstances, most agent-environment interactions with zero reward will have limited contributions to its learning progress toward the optimal solution.

To handle the sparse-reward issue of the PT, one common approach is to improve the reward function by integrating human knowledge about the target system. This can help decompose the PT goal into many subgoals. By adding additional rewards to the agent when some subgoals of PT are achieved instead of rewarding the agent

only when the ultimate goal is achieved. The sparsity could be improved since subgoals will be easier to achieve than the ultimate goal. Based on this idea, our first attempt, presented in Chapter 4, is to reshape the reward function by decomposing the main goal into subgoals based on the agent's knowledge about the control policy of the power grid.

However, the improved reward function mentioned above is designed only for the power grid with a certain control function. If the grid control function changes, the reward function could be ineffective. Therefore, the reward function introducing application-specific human knowledge about the target system could lose generality since if the system changes, the reward function should be redesigned. Currently, there is no standard principle for designing reward functions for PT across various systems. Therefore, it is essential to explore the general task decomposition in terms of reward function design based on human knowledge of PT.

Moreover, encoding intricate human knowledge into a single reward function could increase the complexity of the reward function, making it difficult for the agent to differentiate which aspects of contributions or losses result from its PT actions. Therefore, the agent may need many more interactions to figure it out. In addition, the reshaped reward function cannot strictly guarantee the optimal PT policy from a mathematical point of view because it is designed empirically without considering policy convergence. This is also a limitation for most RL/DRL-based PT or vulnerability analysis studies.

In addition, interpretability is commonly absent in RL/DRL-based PT. The PT policy, once trained, may not be able to explicitly determine the current phase or situation of the PT agent and the direction it should take next. While this perception or awareness could potentially be embedded into the neural networks of the DRL policy during training, extracting such information by decoding the neural networks

remains a challenging task.

Proposed Solution for Problem 3:

To address the challenges related to sampling efficiency, reward specification, and interpretability in RL/DRL-based PT, we proposed to integrate cybersecurity domain knowledge as guidelines into the agent's learning process in an interpretable manner.

We introduce a knowledge-informed AutoPT framework named RM-PT, which leverages a reward machine (RM) to incorporate domain expertise from cybersecurity knowledge bases, such as the MITRE ATT&CK [72]. The RM defines a series of events within PT and decomposes the task into various subtasks based on established PT practices. Furthermore, the RM enables the specification of diverse reward functions for different phases of PT, offering greater flexibility compared to conventional reward functions in RL/DRL-based PT. Within the RM-PT framework, PT is modeled as a partially observable Markov decision process (POMDP) guided by RMs.

In this study, we focus on the lateral movement within the cyber part of the smart grid as a case study, assuming that the pen-tester has already gained initial access to the control center networks and aims to progress further to obtain high-value assets. We explore two different RMs as guidelines. Finally, we employ the deep Q-learning algorithm with RM (DQRM) to optimize the PT policy.

1.3 Thesis Contributions

This study proposes a set of AutoPT methodologies used to identify vulnerabilities for cyber-physical ADNs effectively and efficiently based on DRL methods.

The scope of ADNs in this work falls into distribution systems of smart grids with voltage regulation functions actively performed by a control center. Two types of communication networks form the AND. One is the industrial control network consisting of sensors, grid control devices, and communication infrastructure. Sensors

(e.g., PMUs and PDCs) are deployed remotely at different locations of the grid to obtain the voltage magnitude of the grid. Grid control devices (e.g., voltage regulators) are used to regulate system voltage based on voltage measurements. The other one is the local area network (LAN) deployed in the control center to facilitate its operation.

The proposed AutoPT techniques identify vulnerabilities for different types of networks in an ADN, respectively. For industrial control networks of the ADN, the vulnerability is defined as cyberattack paths that could be utilized by adversaries to compromise the grid control functions and create negative physical impacts. For the LAN of the ADN, the vulnerability is defined as attack paths that could be exploited to gain privilege for operating critical hosts in the control center.

The opposed AutoPT techniques require a simulator of the target AND to create a digital sandbox. An agent, a computer program playing as the pen-tester, will interact with the simulator by launching cyberattacks, aiming to train PT policies that can identify vulnerabilities of the target ADN through trial and error.

Different groups of people can refer to or benefit from our opposed methods, including grid operators, cybersecurity experts, and cybersecurity researchers.

The key contributions are summarized in terms of different scopes of ADNs, listed as follows.

- First, to enhance the efficiency and reduce manual intervention of vulnerability identification for industrial control networks of ADNs, we proposed a DRL-based PT framework and formulated the PT as an MDP problem, where the cyber-physical coupling, realistic cyberattack means, and physical impacts of ADNs are comprehensively considered. Based on this framework, we formulated the replay attack on ADNs with CVR control as an MDP and applied DRL to identify the critical replay attack path toward the system voltage violation. We

also analyzed the PT difficulty by performing PT in different scenarios with different system load levels, solar power generations, and weather variations. This contribution was published in the paper "Deep reinforcement learning for penetration testing of cyber-physical attacks in the smart grid" in 2022 IEEE World Congress on Computational Intelligence [73].

- Second, to address the limited observability in realistic PT scenarios, we introduced the POMDP formulation of PT for ADNs. To solve the POMDP and obtain the optimal PT policy, we used a physical model of the power grid to estimate its full state based on the local observable data captured by the agent and then transformed the POMDP into an MDP that can be solved by DRL. This solution was submitted as a paper "Penetration Testing of Cyber-Physical Attacks in Smart Grids Based on Partially Observable Markov Decision Process" to IEEE Transactions on Dependable and Secure Computing.

- In addition, to conveniently create a digital sandbox for the industrial control networks of ANDs and train PT policies using DRL, we developed a cyber-physical co-simulation platform called GridBattleSim, which integrates dedicated simulators and programs to simulate different parts of ADNs as well as PT agents powered by DRL.

- Finally, to address the sparse reward issue, reduce the difficulty of reward function specification, and improve the interpretability of RL/DRL-based PT for ANDs, we proposed a knowledge-informed AutoPT framework (RM-PT), which can encode human knowledge using RMs to break down the PT goal into many attainable subgoals. The framework was validated through case studies on lateral movement for the LAN of ADNs, where MITRE ATT&CK framework is considered as the human knowledge representation to design RMs. Guided by

RMs, the DQRM algorithm was applied to train the agent and obtain the optimal PT policies efficiently. This contribution was presented in the paper "Knowledge-Informed Auto-Penetration Testing Based on Reinforcement Learning with Reward Machine", which has been published by the 2024 IEEE World Congress on Computational Intelligence [74].

In addition to the contributions outlined in this thesis, the author also published a survey paper titled "Cybersecurity of Smart Inverters in the Smart Grid: A Survey" [1] in IEEE Transactions on Power Electronics. The paper investigated cybersecurity issues related to smart inverters and conducted a literature review on attack and defense strategies for smart inverters and inverter-based power grids against cyberattacks. Some parts of this paper are referenced in this chapter as background. Moreover, the author has other publications in related areas, but not within the main scope of this thesis, as listed in Chapter 6.

1.4 Thesis Outline

The remainder of the thesis is organized as follows.

The literature review on vulnerability analysis approaches against cyberattacks for smart grids, and AutoPT for the ICT industry will be presented in Chapter 2. The DRL-based PT framework and the MDP formulation of PT will be introduced in Chapter 3. Our developed co-simulation platform GridBattleSim will also be introduced in Chapter 3. The POMDP formulation of PT will be detailed in Chapter 4. In Chapter 5, the knowledge-informed AutoPT based on RM will be presented. Chapter 6 concludes the thesis and depicts future research directions. The outline of this thesis is shown in Figure 1.4.

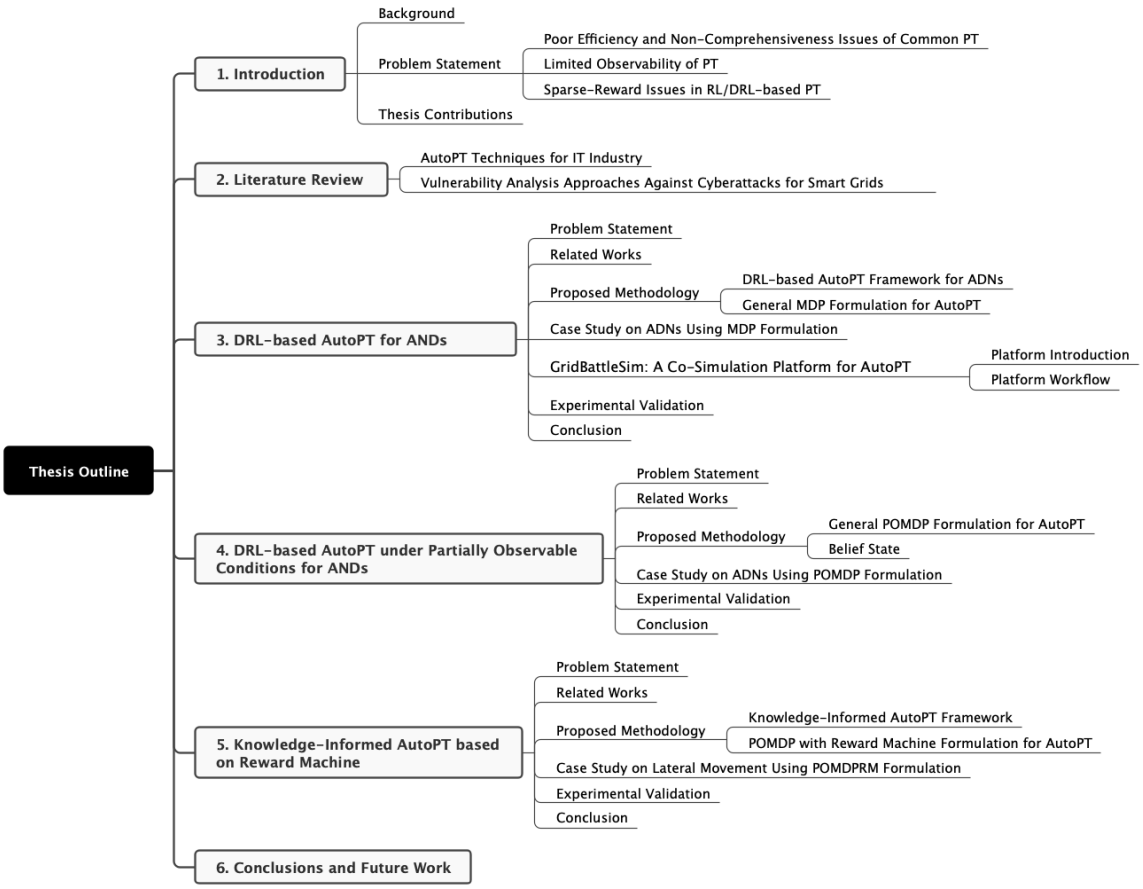


Figure 1.4: Thesis outline.

Chapter 2

Literature Review

This thesis presents an interdisciplinary study of IT and OT security. We partially investigate an RL/DRL-based AutoPT for smart grids to identify vulnerabilities that can be exploited to create negative physical impacts. Prior to this study, numerous research studies have been conducted in the fields of IT and smart grids, offering valuable insights. Hence, we conduct a comprehensive literature review on vulnerability analysis approaches used in IT and smart grids, respectively, as the basic references for this study.

In this chapter, we first present a literature review on the development of AutoPT techniques in the IT industry, including existing AutoPT tools, attack-graph-based AutoPT, and RL/DRL-based AutoPT. Then, we conduct a literature review on vulnerability analysis approaches dedicated to smart grids against cyberattacks, including approaches based on optimization methods and RL/DRL methods, respectively.

2.1 AutoPT Techniques for IT Industry

PT has been performed primarily by skilled human experts. However, this process relies on pen-testers with a high level of expertise and implicit knowledge about

cybersecurity, making it time-consuming, labor-costing, and susceptible to human mistakes. In addition, driven by the expansion of ICT markets and security policies, there is an increasing demand for PT services. Therefore, the security industry is focusing on developing automated PT (AutoPT) techniques to improve efficiency and effectiveness (accuracy) in identifying vulnerabilities for various types of digital assets and information systems.

2.1.1 AutoPT Tools

Currently, many AutoPT tools and frameworks have been developed in the IT industry to enhance PT automation and efficiency in different scopes or levels of impact. PT tools can commonly be classified into or a mix of the following categories [75]:

- Port scanners: detect open ports on a system, aiding in identifying the operating system and applications active on a network they aim to access. Port scanners are utilized in reconnaissance and can reveal information for possible attack vectors.

- Vulnerability scanners: identify known vulnerabilities in servers, operating systems, and applications, as well as configuration errors that can be exploited. Reports generated by vulnerability scanners assist pen-testers in choosing a vulnerability that can be exploited to gain initial access to the system.

- Network sniffer: capture and analyze data packets in network traffic, revealing details such as the origin, destination, communicating devices, and protocols and ports in use. This tool is valuable for verifying data encryption and pinpointing communication routes that may be vulnerable during PT.

- Web proxy: capture and alter traffic between a web browser and the organization's web servers. This allows for identifying hidden form fields and other

HTML features that could facilitate attacks such as cross-site scripting (XSS) or cross-site request forgery (CSRF).

- Password cracker: attackers often target password hashes to escalate privileges within a system or network. Password crackers enable pen-testers to check if organizations use weak passwords that could be exploited.

Table 2.1 lists many widely used PT tools. Among them, Wireshark [76] is a basic packet analyzer. It can record live or saved network traffic, enabling troubleshooting of network traffic and security analysis. It can parse network packets for numerous protocols, allowing pen-testers to identify security vulnerabilities. Metasploit [79] is a Ruby programming language-based PT framework that enables pen-testers to write, test, and execute exploit code against a remote target machine. The Metasploit Framework contains a suite of tools that can be used to find vulnerabilities, execute attacks, and avoid detection. At its core, the Metasploit framework is a collection of commonly used tools that provide a complete environment for PT. Nmap (Network Mapper) is a network scanner used to discover hosts and services on a computer network by sending packets and analyzing responses [80]. John the Ripper is a password cracking tool with many options for password testing, including auto-detection of password hash types and the ability to crack password encryption based on DES, MD5, Blowfish, and MD4, among others [78]. Kali Linux [81] is a Debian-based Linux distribution, which was delicately developed for digital forensics and penetration testing. Kali Linux has integrated approximately 600 PT tools, including Nmap, Wireshark, Metasploit, John the Ripper, etc.

However, automating the entire testing process usually includes the diverse tasks and subtasks for each PT phase, presenting a significant challenge and often failing to achieve the desired objective when executed inappropriately. The utilization of automated tools that perform all possible tests without optimization or pre-processing can

Table 2.1: Common PT tools for IT security.

Name	Category	Functionality	Ref.
Wireshark	Network sniffer	Packet capture, protocol analysis, statistics and reporting, etc.	[76]
Burp Suite	Web proxy	Web application scanning, proxy server, spidering and crawling, statistics and reporting, etc.	[77]
John the Ripper	Password cracker	Brute force cracking, dictionary cracking, rule-based cracking password auditing, etc.	[78]
Metasploit	Port scanner, vulnerability scanner, network sniffer	Exploit development, vulnerability analysis, payload generation, post-exploitation, social engineering, statistics and reporting, etc.	[79]
Nmap	Port scanner, vulnerability scanner, network sniffer	Host discovery, port scanning, service version detection, vulnerability analysis, network mapping statistics and reporting, etc.	[80]
Kali Linux	Port scanner, vulnerability scanner, network sniffer, Web proxy, Password cracker	a Linux platform integrated with various PT tools	[81]

lead to suboptimal results [82, 83, 84]. Most of these automated tools require continuous oversight from a human pen-tester and usually fail to deliver satisfactory results, particularly in medium and large-scale network testing, due to the substantial number of operations needed to cover entire networks [85, 86]. Furthermore, automation introduces additional problems, such as long testing durations that exceed practical limits, increased network congestion from generated traffic, and a high volume of false positive alerts triggered by the system's defense solutions, such as intrusion detection and prevention systems (IDPSs) and firewalls [2]. Blind AutoPT is also restricted to small networks and some medium-sized networks, relying on customized scripts that are cumbersome and resource-intensive [61].

2.1.2 Attack-Graph-based AutoPT

To address the limitations of existing AutoPT tools, AutoPT methods have been further explored to reduce the degree of manual intervention and improve efficiency. Many AutoPT methods have been proposed. Among them, numerous works define PT as a path planning problem represented by attack graphs. Phillips et al. [87] first proposed a method based on the attack graph to analyze the system's vulnerability, which can pinpoint attack paths with a high likelihood of success or low effort for the attacker. The method requires a database of common attacks, network configuration details, and an attacker profile as input to create a comprehensive attack graph. By assigning probabilities or costs to the attack paths, the system can identify the most probable attack paths using graph algorithms, contributing to enhancing network security measures. Based on [87], Kyle et al. [88] developed the NetSPA attack graph, which leverages firewall rules and vulnerability scans to evaluate multiple targets in minutes, significantly reducing the time required to construct attack graphs. To improve the reliability of attack paths of attack graphs, Qiu et al. [85] introduced

a method for the automatic creation of attack graphs utilizing the Common Vulnerability Scoring System (CVSS). However, most of the methods mainly generate action paths for static environments, offering guidelines for PT, but cannot perform PT dynamically and interactively in changing environments. When the environment changes, these methods usually need to rebuild the attack graphs. Therefore, the attack graphs used before are not easily reused again, leading to a waste of knowledge. In addition, these methods require a comprehensive understanding of the target system and detailed machine information, which sometimes are not easy to satisfy, especially for large-scale and complex information systems.

2.1.3 RL/DRL-based AutoPT

Therefore, the security industry is currently investigating more intelligent and autonomous methods for PT. Early research efforts focused on enhancing PT by optimizing the planning phase, which was conceptualized using attack trees or attack graphs [89, 90]. These models were designed to map the sequential decision-making process of PT to graph representations, aiming to identify critical attack paths in a logical way. Nevertheless, these methods encounter several issues. First, they require a clear and accurate understanding and specification of the target systems, thereby limiting their applicability. Second, the constructed attack trees and attack graphs are difficult to reuse when the target systems are changed in terms of structure and configurations [91], making it challenging to maintain the effectiveness of PT methods over time.

To overcome these challenges, there has been a rise in research on RL/DRL-powered PT on information systems. Schwartz et al. [92] proposed a DQN-based PT approach for enterprise networks. The PT was formulated as a Markov decision process (MDP) with the known configuration of the network as states and available

scans and exploits as actions. To validate the DQN agent, a lightweight network simulator was utilized to play as the environment. Hu et al. [93] proposed a two-stage deep reinforcement learning (DRL)-based PT approach for communication networks. In the first stage, the approach utilizes scanning tools to collect network information and build an attack tree. In the second stage, the DQN algorithm is applied to identify the most impactful and shortest attack path based on the attack tree. This approach aims to save labor costs and improve PT performance. To further enhance the PT performance and reduce manual labor costs, Ghanem et al. [94, 2] developed the Intelligent Automated Penetration Testing System (IAPTS). The IAPTS combines the reinforcement learning (RL) module with industrial IT PT frameworks. Tran et al. [95] introduced a hierarchical DRL method for PT to address its large discrete action space, where a decomposition strategy for the action space is utilized. Qianyu et al. [96] proposed the INNES model for DRL-based PT to characterize its observation space and action space of PT, which makes the MDP formulation of PT more accurate. The Microsoft Defender Research Team developed an open-source RL-based PT research platform called CyberBattleSim [97]. This platform builds a high-level parameterizable model of enterprise networks with pre-defined vulnerabilities to play as the environment. The agent launches RL-powered cyberattacks based on the lateral movement mechanism that exploits network vulnerabilities.

The reviewed advanced AutoPT methods (attack graph-based and RL/DRL-based) for the IT industry are summarized in Table 2.2.

While studies employing RL/DRL techniques offered valuable insights, they are still in the early stages and lack established practical standards, protocols, or frameworks. Moreover, most of the models used in these studies are highly abstract and simplified, which cannot fully capture real-world systems. In addition, the goal of

Table 2.2: Summary of advanced AutoPT methods for IT Industry.

Authors	Year	Approach	Contributions	Limitations	Ref.
Phillips et al.	1998	Attack graph-based vulnerability analysis and path planning.	First introduced attack graphs to pinpoint attack paths.	Requires database of common attacks and network details; Limited to static environments.	[87]
Kyle et al.	2009	NetSPA attack graph utilizing firewall rules and vulnerability scans.	Developed NetSPA attack graph to reduce time in constructing attack graphs.	Limited to static environments.	[88]
Qiu et al.	2014	Automatic creation of attack graphs using Common CVSS for improved reliability.	Introduced methods for the automatic creation of attack graphs using CVSS that enhances the reliability of attack graphs.	Limited to static environments; Requires detailed information about the target system.	[85]
Ghanem et al.	2018	Developed IAPTS, which combines RL with existing PT frameworks.	Reduce manual labor costs; Integration of RL with industrial IT frameworks.	Not be validated in diverse IT environments.	[94, 2]
Schwartz et al.	2019	DQN-based PT for enterprise networks.	MDP formulation; DQN as the MDP solver.	Validation using a lightweight network simulator.	[92]
Hu et al.	2020	Two-stage DQN-based PT for communication networks.	Integration of existing PT tools with attack tree and DQN algorithm.	Identified attack paths are restricted by the attack tree, which cannot identify hidden vulnerabilities outside of attack trees.	[93]
Tran et al.	2021	Hierarchical RL (HRL) method for PT to address large discrete action spaces.	Propose action decomposition strategy for PT based on HRL.	Complicated implementation; Need for further validation in diverse PT scenarios.	[95]
Li et al.	2023	INNES model for DRL-based PT to enhance MDP formulation accuracy.	Proposed an observation and action space characterization method for MDP formulation.	Limited real-world validation.	[96]

these works is to take ownership of the digital assets of the target system. For cyber-physical smart grids, the goals of PT should also consider identifying vulnerabilities that can lead to negative physical impacts. Thus, there is a pressing need for the development of comprehensive and system-level PT solutions specifically for smart grids.

2.2 Vulnerability Analysis Approaches Against Cyberattacks for Smart Grids

According to the review of the literature conducted by this study, vulnerability analysis approaches for smart grids can generally be categorized into optimization-based methods and RL/DRL-based methods. Optimization-based vulnerability analysis is commonly used to identify critical attack vectors or the most vulnerable components of smart grids that can create an instant impact under the current grid condition without consideration of the subsequent attack sequences. On the contrary, RL/DRL-based vulnerability analysis methods try to identify a sequence of coordinated attack vectors with the goal of maximizing a long-term impact or achieving an impact that should go through a series of attack steps.

2.2.1 Optimization-based Vulnerability Analysis

We first give a review of optimization-based vulnerability analysis for smart grids as follows.

For identifying critical false data injection attacks (FDIAs) on smart grids, Ref. [98] first introduced FDIA against state estimation (SE) in electric power grids as a new class of attacks, which can introduce undetected errors into certain state variables by manipulating meter measurements at certain nodes of the grid. The injected error is

produced based on the topological matrix (also known as the Jacobian matrix) of the power grid, which can bypass the bad data detector (BDD) of the SCADA system.

Ref. [99] considered resource constraints in FDIA, which decides the minimum number of sensors of the power grid that could be attacked by injecting Gaussian noises. This constrained FDIA is formulated into a convex optimization problem that is solved based on matrix theory, where the trace of the remote estimation error covariance is maximized.

Similarly, [100] applied a multi-objective optimization approach to identify critical FDIA on SE of smart grids, aiming to minimize attack cost and maximize attack impact stealthily. The optimization involves an objective for reducing the number of compromised measurements and an objective for maximizing errors in estimated states or branch power flows. The SPEA2 algorithm was utilized to solve the optimization problem, which can find a set of optimal attack vectors on the Pareto front while considering physical and operational constraints to ensure stealthiness.

Ref. [101] investigates the vulnerability of the AC-based State Estimation (SE) function in power systems against FDIA. To identify stealthy, low-cost, and impactful FDIAs, this study proposed a convexification framework based on semidefinite programming (SDP). By solving the SDP, it found the optimal FDIAs that can delineate the "attackable region" for any given set of measurement types and grid topology, where the spurious state can be falsified by FDIA.

Ref. [102] introduced the concept of optimal FDIAs (OFDIAs) in the power system frequency control loop, utilizing a linearized formulation of power systems' dynamics in an optimization framework. It investigated the impact of continuous and time-limited FDIAs on power grid frequency behavior, showing that continuous FDIAs can lead to severe consequences like frequency instability, while time-limited FDIAs can cause fluctuations triggering protection relays.

Ref. [103] investigated the impact of FDIAs on power system frequency-based protection relays, which can lead to false relay operations and threaten power system security. An optimization-based formal model is proposed to identify the optimal FDIA with the shortest time required to trigger a false relay operation. The model considered power system dynamics to determine the optimal attack size over multiple generator dispatching cycles, showcasing that systems with higher inertia and certain governor settings are more resilient against such attacks.

Ref. [104] analyzed the risk of GPS spoofing attacks on smart grids. It introduced a novel measurement model to assess the impact of GPS spoofing attacks and formulated an optimization problem to identify the most vulnerable PMUs. This optimization problem was subsequently addressed using a greedy algorithm.

To identify vulnerable components in smart grids, Ref. [105] explores the vulnerability of power grids to cascading failures induced by physical sabotages and cyberattacks on substations and transmission lines. Unlike prior research focusing on individual components, this study introduces a joint perspective considering simultaneous attacks on both substations and transmission lines. Through the introduction of the component interdependency graph (CIG) metric and analysis of joint attack strategies, the research illustrates the efficacy of the CIG-based approach in identifying vulnerabilities and improving attack performance in diverse power grid systems.

Similarly, Ref. [106] analyzed the vulnerability of the power grid to simultaneous attacks, emphasizing the potential of critical system failures. By utilizing a modified cascading failure simulator with enhanced efficiency, the study assesses the impact of combined attacks on generation power loss and blackout duration. A new damage measurement matrix is proposed to identify the most effective attack combinations that lead to maximum damage, offering valuable insights for enhancing system resilience against coordinated threats.

2.2.2 RL/DRL-based Vulnerability Analysis

For RL/DRL-based vulnerability analysis on smart grids against cyberattacks, Yan et al. in [107] proposed a sequential topology attack scheme on the transmission system of smart grids based on the Q-learning algorithm [107]. This approach enables the identification of the optimal ordering for sequentially triggering power lines to create cascading blackouts. Subsequently, Ref. [108, 109] expanded on Yan's work by considering the involvement of defenders and formulating the attack as a Markov Game, thereby increasing the attack difficulty. In [110], the authors assumed that attackers could not only cut power lines but also attack the communication topology, which could mask the line outage signal sent to the control center and cause improper energy management actions. To address this issue, they applied the Deep Q-learning (DQN) algorithm to identify the critical attack path that could result in a cascading failure. Ref. [111] proposed a cybersecurity assessment approach that uses DRL and CVSS to evaluate power grids' vulnerabilities against cascading failures, considering intermittent DER generation and IT/OT device vulnerabilities. The simulation results validated the effectiveness of the approach, showing that the DQN algorithm closely matches a graph-search approach in identifying optimal attack policies with fewer transitions needed. Ref. [112] focuses on identifying critical transmission lines in smart grids considering load uncertainty to prevent cascading failures and enhance system robustness. The study introduced an optimal virtual attacking problem to maximize expected generation loss under given resources and proposed an online algorithm based on prioritized multi-agent-attention-actor-critic (PMA3C) to efficiently identify critical lines instantly under load uncertainty.

Another area of investigation is RL/DRL-based FDIAs on the SE function of the transmission system. In [113], researchers utilized the Q-learning algorithm to determine the optimal injected error on phasor measurement unit (PMU) measurements.

Furthermore, the trained FDIA could be launched stealthily and bypass the bad data detection (BDD) of the supervisory control and data acquisition (SCADA) system via the design of the reward function. Ref. [114] proposed a DRL-based vulnerability analysis scheme that allows the control center to identify vulnerable meters by constructing attack vectors based on power system states, meter measurements, previous analyses, and injected errors without knowledge of the power system topology. The proposed actor-critic architecture of the DRL agent effectively handles continuous and high-dimensional vulnerability analysis policies. The simulations demonstrated improved vulnerability detection rates, reduced number of analyzed meters, and enhanced utility while considering computational complexity. Ref. [115] applied DRL to identify underlying systemic vulnerabilities of the reactive power market of the smart grid. The researchers demonstrated how attackers could exploit the market by inducing constraint violations and profiting from the flexibility market, revealing previously unknown attack strategies. Recent research has also explored the influence of RL-based FDIA on microgrids, where intelligent FDIA modifies the control parameters of distribution energy resources (DER) [116].

However, these studies reviewed above rely heavily on the assumption that the power grid has been deeply penetrated by attackers who can directly modify control commands without considering real-world scenarios such as communication protocols. Furthermore, established models of these studied or experimental testbeds in these works often neglect the cyber aspect of the power grid, resulting in oversimplification and lack of comprehensiveness.

The summary of existing advanced vulnerability analysis methods (optimization-based and RL/DRL-based) for smart grids is listed in Table 2.3.

Table 2.3: Summary of advanced vulnerability analysis against Cyberattacks for smart grids.

Authors	Year	Approach	Contributions	Ref.
Liu et al.	2011	FDIA modeling against SE that can bypass BDD.	First work that identified stealthy FDIA on SE of power grids	[98]
Li et al.	2018	FDIA modeling against SE based on convex optimization with resource constraints.	Formulated constrained FDIA with Gaussian noises	[99]
Jin et al.	2018	FDIA modeling against AC-based SE based on SDP	Found out the optimal FDIAs that can delineate the "attackable region"	[101]
Risbud et al.	2018	GPS spoofing vulnerability analysis on smart grids	Introduced novel measurement model to identify most vulnerable PMUs.	[104]
Rahman et al.	2020	FDIA modeling based on multi-objective optimization	The proposed FDIA minimizes attack cost and maximize attack impact	[100]
Jafari et al.	2022	FDIA modeling against frequency control loop	Studied impact of continuous and time-limited FDIAs on power grid frequency behavior	[102]

Zhu et al.	2015	Cascading failures analysis for cyberattacks on substations and transmission lines.	Introduced a joint perspective considering simultaneous attacks on both substations and transmission lines	[105]
Paul et al.	2017	Simultaneous topology attack analysis on smart grids.	Assessed and identified simultaneous topology attack that leads to generation power loss and blackout duration.	[106]
Yan et al.	2016	Sequential topology attack analysis based on Q-learning.	Analyzed and identified the most impactful sequential topology attacks.	[107]
Chen et al.	2018	Optimal FDIA identification on SE function based on POMDP	Formulated FDIA on SE as a POMDP	[113]
Ni et al.	2019	Sequential topology attack analysis with defender involved based on Q-learning.	Identified critical attack path under defence.	[108]
Liu et al.	2020	Cascading failure analysis for power grids based on DRL and CVSS.	CVSS integration into DRL-based vulnerability analysis.	[111]
Wang et al.	2020	Critical attack path analysis involving communication topology attacks	Applied DQN algorithm to identify critical attack paths considering communication topology attacks	[110]

Yu et al.	2022	PMA3C algorithm for critical line identification under load uncertainty	Introduced optimal virtual attacking problem for critical line identification under load uncertainty.	[112]
Neal et al.	2021	RL-based FDIA impact analysis on microgrids	Analyzed impact of RL-based FDIA that can modify DER control parameters of microgrids	[116]

Chapter 3

DRL-based AutoPT for ADNs [1]

3.1 Problem Statements

The increasing complexity of smart grids makes manual penetration testing at a system level impractical for identifying critical vulnerabilities to cyberattacks. The extensive and repetitive PT tasks carried out by human experts are time-consuming and labor-consuming. Current PT studies on information systems, as reviewed in Chapter 2.1, concentrated primarily on the cyber aspects of the system, with the aim of securing digital resources within networks. They commonly neglected to consider the impact of cyberattacks on the physical processes of the target system, especially for a CPS. On the contrary, most CA studies against cyberattacks for smart grids, reviewed in Chapter 2.2, overlooked the cyber aspects, neglecting the practical way of cyberattack intrusion. While some CA studies have begun to explore connections with the cyber layer, these considerations are often based on assumptions rather than conducting detailed modeling of the cyber component and implementing realistic cyberattacks.

[1] This chapter is published in 2022 IEEE World Congress on Computational Intelligence [73].

3.2 Related Works

RL/DRL-based cyberattacks against power grids have been investigated. Ref. [107] studied on sequential topology attack schemes on transmission systems with Q-learning, where the learned policy is able to identify critical transmission lines that can cause cascading failures of the power grid if tripped sequentially. Ref. [108, 109] extended the work in [107], considering the participation of the defenders and forming the problem as a Markov Game to be solved by RL. Ref. [110] extended the attacker's capability to target not only the power grid topology but also the communication topology, so the outage signal would be masked, and the control center misled into improper dispatch with negative impacts. Ref. [113] investigated the RL-based false data injection attack (FDIA) on the state estimation in the transmission system, where the optimal injected attack vector on phase measurement units (PMU) can be learned online by the Q-learning algorithm. Similarly, researchers in [116] analyzed the impact of the RL-based FDIA on microgrids where the control input of distribution energy resource (DER) controllers can be strategically manipulated by FDIA. However, the above-mentioned studies are based on assumptions that the power grid has been compromised so that the attackers are able to launch cyberattacks and inflict physical actions directly as the starting point. The simulators also only model the physical side of the power grid, whereas the cyber part was neglected.

Meanwhile, RL/DRL-based PT on information systems for the ICT industry has recently drawn attention. Ref. [92] proposed an RL-based PT for communication networks, where a lightweight network attack simulator was developed to perform the PT. The study then proposed the PT as a Markov decision process (MDP) with the known network configuration as states and the available scans and exploits as actions; the final PT actions are determined by applying Q-learning and DQN, respectively. However, the built communication networks are highly abstract and simplified - not

simulated - that cannot fully capture real-world systems. Ref. [93] proposed a two-stage DRL-based PT on communication networks, where the first stage uses search engines to collect network information to build an attack tree; the second stage applies the DQN algorithm to discover the easiest-to-exploit attack path based on the attack tree. Ref. [94, 2] proposed the Intelligent Automated Penetration Testing System (IAPTS), which integrates the RL module with industrial IT PT frameworks with the aim to save human resources while producing much-enhanced PT results. These studies are valuable but still at the early stage without practical standards, protocols, or devices considered. Moreover, these works are limited to network infrastructures PT planning and not the entire practice. These studies also only focused on vulnerabilities of the cyber part, while the cyber-physical coupling and the physical impact were not considered.

To address this gap, we proposed a DRL-based PT framework aimed at efficiently and adaptively identifying critical vulnerabilities in cyber-physical ADNs. Based on this framework, the pen-tester is modeled as a DRL agent, and the PT is formulated as a Markov decision process (MDP). To illustrate the effectiveness of our framework, we conducted a case study on replay attacks targeting an ADN with Conservation Voltage Reduction (CVR) control. This approach enables the agent to learn the optimal timing and ordering of replay attacks in various operating scenarios. To solve the MDP, we employed the Deep Q-Network (DQN) algorithm to train the agent and optimize the PT policy. Furthermore, we developed a cyber-physical co-simulation platform named GridBattleSim, which includes specialized simulators for the physical, cyber, control, and attacker components of ADNs. This platform serves as a sandbox environment for training the DRL agent. We conducted tests on scenarios of varying difficulty levels to assess the learning capability and performance of our DRL-based PT framework in identifying critical attack paths.

3.3 Proposed Methodology

3.3.1 DRL-based AutoPT Framework for ADNs

In this work, the PT is a sequence of cyberattacks on the target cyber-physical ADN launched by a pen-tester or ethical hacker. The objective of PT is to identify critical attack paths that can lead to negative physical impacts on the system. Thus, PT is considered as a sequential decision-making problem. A typical mathematical formulation of PT is the Markov decision process (MDP), which can be solved by applying RL/DRL algorithms to obtain the optimal decision-making policy.

Fig. 3.1 shows the proposed DRL-based PT framework, which involves an agent and an environment. The agent, acting as a pen-tester, takes PT actions to interact with an environment that is a target cyber-physical smart grid. The environment is coupled with various physical components and cyber components. The physical components include grid feeders, transformers, DERs (e.g., PV arrays, wind turbines, batteries), power electronic devices (e.g., DC-AC inverters, DC-DC converters), loads (e.g., houses, factories), voltage regulation devices (e.g., voltage regulators, capacitor banks), among others. The cyber components include endpoints for many above-mentioned power devices using communication interfaces (e.g., RTUs), sensors (e.g., PMUs, smart meters), as well as the communication networks that facilitate the message delivery between endpoints and the control center (e.g., routers, communication channels). In addition, grid control applications (e.g., voltage control and energy management) in the control center are also covered in the cyber part, which gathers measurements from the power grid through sensors and dispatch control commands to actuators to achieve the control objectives.

The actions taken by the agent can be atomic actions of various cyberattack techniques used in PT, such as deny of service (DoS), packet drop, packet replay, and

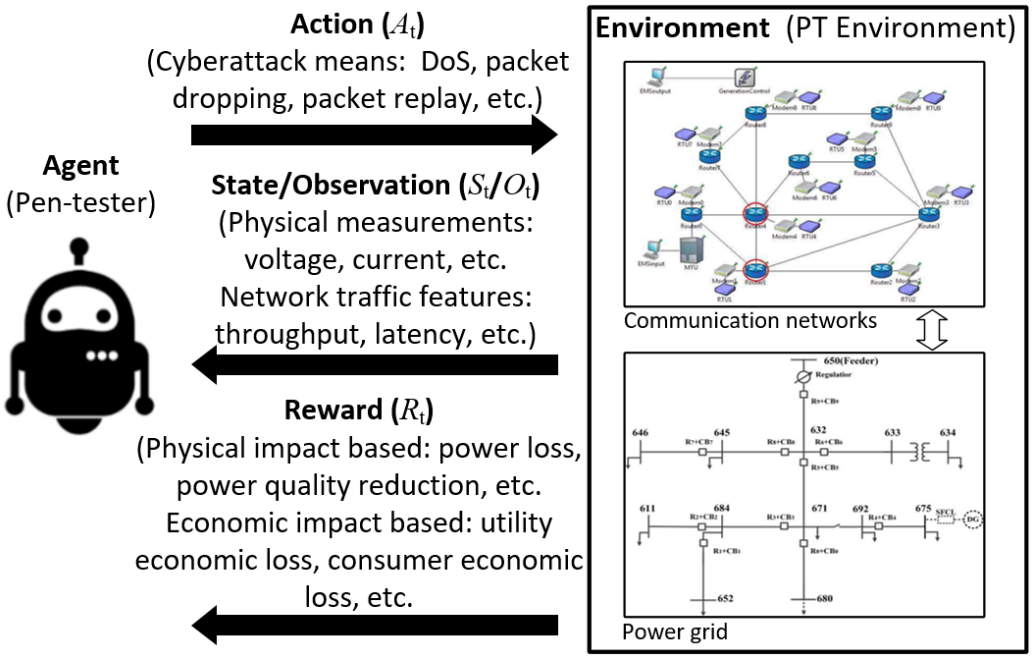


Figure 3.1: DRL-based PT framework.

man-in-the-middle (MITM), among others. The environment will transit its state to a new state when the action works and outputs its state or observations to the agent, including electrical measurements from the physical part of the power grid (e.g., voltage, current, frequency) and measurements that describe the cyber part of the power grid (e.g., throughput, latency). At the same time, the agent will also receive an immediate reward signal from the environment obtained by a reward function to judge the agent's action performance. Since the goal of PT is to identify attack paths that can create a negative impact on the power grid, the reward function can be designed to reflect the existence or extent of a physical impact, such as power loss and reduction in power quality, or an economic impact, such as the economic loss, or the combination of the physical impact and the economic impact. In this work, we are mainly concerned with the physical impact caused by cyberattacks. Therefore, the reward function is designed based on the physical impact of the power grid. Under the RL/DRL paradigm, the agent's objective is to find an optimal PT policy

that can maximize the accumulated rewards during PT through learning experiences from agent-environment interactions [117]. The PT policy determines the action to take in a given state or an observation from the environment. In this framework, to address the curse of dimensionality challenge resulting from the large state space or observation space in the complicated cyber-physical system [118], deep artificial neural networks are applied to represent the policy in terms of an approximation of a value function or a policy function [119].

In this study, the terms "pen-tester" and "agent" can be interchangeably since they refer to the same entity.

Under the proposed DRL-based PT framework, PT is formulated as an MDP problem, which will be introduced in the next subsection.

3.3.2 General MDP Formulation for AutoPT

An MDP is a discrete-time stochastic control process that uses a mathematical framework to describe the sequential decision-making process of an agent in a dynamic system. It is utilized in situations where the internal state of the environment is uncertain and influenced by the agent who makes sequential decisions over time [120].

We formulate the proposed PT as an MDP, which is defined by a tuple with five components $\langle \mathcal{S}, \mathcal{A}, P(s_{t+1}|s_t, a_t), R(s_t, a_t, s_{t+1}), \gamma \rangle$ [121], where $\mathcal{S}$ is a set of environment states; $s_t \in \mathcal{S}$ is a state at time t; $\mathcal{A}$ is a set of actions; $a_t \in \mathcal{A}$ is an action taken by the agent at time t. $P(s_{t+1}|s_t, a_t)$ is the state transition probability that describes the transition of the environment state when the agent performs an action a_t in a current state s_t. $R(s_t, a_t, s_{t+1})$ is the reward function that calculates the immediate reward r_{t+1} received by the agent after performing the action a_t and the environment transitions to the state s_{t+1}. $\gamma \in [0, 1)$ is the discount factor that determines the trade-off between the immediate and long-term rewards that the agent prefers to

achieve. In addition to the MDP tuple, we utilize the feature vector ϕ_t to represent the extracted feature information from the environmental state at time t. The design of feature extraction functions is tailored to specific study cases.

Due to the complex nature of the target smart grid, the determination of $P(s_{t+1}|s_t, a_t)$ poses challenges for PT. However, model-free RL/DRL methods allow the agent to take the environment as a black box and learn the PT policy through pure trial and error.

A general description of MDP components for the proposed DRL-based PT framework is presented below.

State: In the field of power systems, the steady operational state of a power grid is commonly characterized by the magnitude and angle of the three-phase voltages of every node bus. These state parameters can be used to calculate the power flow of each bus node and each branch of the power grid. These state parameters are also the outputs of the state estimation (SE) function performed by the control center based on the obtained measurements from field sensors, such as PMUs. Accordingly, in this study, the environment state of the target smart grid is defined as a vector of the three-phase voltage magnitude and three-phase voltage angle of every node bus in the power grid. Let N denote the number of node buses; $V_{i,p}$ and $\theta_{i,p}$ denote the voltage magnitude and angle, respectively, of the i-th node in phase p. Then, the state vector is expressed as follows:

$$\begin{aligned} s_t = [&V_{1,A}, V_{1,B}, V_{1,C}, ..., V_{N,A}, V_{N,B}, V_{N,C}, \\ &\theta_{1,A}, \theta_{1,B}, \theta_{1,C}, ..., \theta_{N,A}, \theta_{N,B}, \theta_{N,C}] \end{aligned} \tag{3.1}$$

Action: The action a_t taken by the agent at time t is selected from a set $\mathcal{A}$ (also called the action space). The agent needs to select an action from this set in each time step to form a cyberattack. Usually, a pen-tester can perform various cyberattack

techniques, including denial of service (DoS), packet drop, packet replay, and packet delay, among others. Each cyberattack technique may have several atomic actions, and each action will also have many action parameters set in the background. The aggregation of these atomic actions forms the action space $\mathcal{A}$ for PT. Table 3.1 lists four typical cyberattack techniques on smart grids as well as their action spaces and action parameters.

Table 3.1: Action space of PT on smart grid

Cyberattack Techniques	**Action Spaces**	**Action Parameters**
DoS	{*DoS Start*, *DoS End*}	IP address, port number, packet size, time interval
Packet Drop	{*Drop Start*, *Drop End*}	Type of packets
Packet Delay	{*Delay Start*, *Delay End*}	Type of packets, delay duration
Packet Replay	{*Packet Record*, *Packet Replay*, *Replay End*}	Type of packets, replay order

For a DoS attack, two atomic actions could be defined: *DoS Start* and *DoS End*. For *DoS Start* action, the pen-tester should determine some parameters to perform a flooding attack, including the IP address of the target device, the port number of the target device, the size of the sending packet, and the time interval between sending each message. For *DoS End* action, since it is a stop of DoS, no parameters need to be determined.

For a packet drop attack, two atomic actions could be defined: *Drop Start* and *Drop End*. For *Drop Start* action, the pen-tester should determine what types of packets captured by the pen-tester will be dropped, which could be the PMU packets, dynamic pricing packets, or both. For *Drop End*, since it is a stop action, no

parameters need to be determined.

For packet delay attack, two atomic actions could be defined: *Delay Start* and*Delay End*. For *Delay Start*, the pen-tester should determine what types of packets captured by the pen-tester will be delayed and a fixed delay duration that will be conducted on the received packets. For *Delay End*, since it is a stop action, no parameters need to be determined.

For replay attack, three atomic actions could be defined: *Packet Record*, *Packet Replay*, and *Replay End*. For *Packet Record*, the pen-tester should determine what type of packets will be recorded. For *Packet Replay* action, the pen-tester should determine the replay order of recorded packets saved in its memory. The order can start from the latest recorded packet (packets are recorded in a first-in-last-out memory, FILO) or starting from the oldest recorded packet (packets are recorded in a first-in-first-out memory, FIFO). For the *Replay End* action, since it is a stop action, no parameters need to be determined, but it will end either *Packet Record* action or *Packet Replay* action.

Moreover, we define the action duration, denoted as δ (unit: second), as the time duration used by each atomic action.

Reward Function: In this work, the reward function encodes the tactical goal of PT. More specifically, the reward function is designed to reflect the existence or extent of negative physical impacts on the power grid. A negative physical impact is defined as the violation of the compliance limit (safety range) for the power quality parameters, including voltage violation, frequency variation, voltage unbalance, and voltage harmonics, among others [122]. A general form of the reward function is expressed in Eq. 3.2:

$$R(s_t, a_t, s_{t+1}) = \sum_{i=1}^{N_Q} (U_i \times R_i^Q(s_t, a_t, s_{t+1})), \tag{3.2}$$

Where N_Q is the number of possible power quality parameters that we focus on for the smart grid; U_i is a binary variable that indicates whether the i^{th} power quality parameter is considered or not ($U_i = 1$: considered, $U_i = 0$: not considered); $R_i^Q(\cdot)$ is the impact function for the i^{th} power quality parameter. If the action a_t performed by the agent brings some violations on the safety range for the i^{th} power quality parameter, $R_i^Q(\cdot)$ will output a continuous positive value to quantify the extent of the impact or output the integer 1 to indicate the occurrence of this violation. A case-specific reward function design will be presented by the case study in Section 3.4.

Objective Function: The PT can be considered as a kind of worst-case identification procedure on the target system, where the pen-tester tries to create as many negative physical impacts on the smart grid through cyberattacks. Therefore, the discounted accumulated rewards during the PT can serve as the objective function for the agent to maximize, denoted as G and represented in Eq. 3.3:

$$G = \sum_{t=1}^{\mathcal{T}} \gamma^{t-1} r_t, \tag{3.3}$$

where $\mathcal{T}$ is the maximum number of actions taken in the PT. r_t is the immediate reward obtained by the reward function $R(s_t, a_t, s_{t+1})$. $\gamma \in [0, 1)$ is the discount factor that controls the trade-off between the immediate and long-term rewards that the agent tends to achieve. By optimizing Eq. 3.3, we hope to find out an optimal PT policy $a_t^* = \pi^*(s_t)$ that maps the current state s_t to the an optimal action a_t^*, as expressed in Eq. 3.4

$$\pi^* = \arg\max_{\pi} G. \tag{3.4}$$

Thus, solving the MDP and obtaining the optimal policy π^* that maximizes G is an optimization problem that can be solved by RL/DRL algorithms.

3.4 Case Study on ADNs Using MDP Formulation

In this chapter, we focus on PT on an ADN with conservation voltage reduction (CVR) as the control application deployed by the control center. We investigate the replay attack on PMU packets as the main cyberattack technique used in the PT and hope to identify the optimal timing and ordering of replay attacks toward the voltage instability of the ADN.

3.4.1 Threat Modeling of Replay Attacks on ADNs

In this study, a centralized CVR control application is applied as a distribution automation (DA) function performed by the control center. The replay attack used in the PT is a category of network attack techniques in which an attacker detects a data transmission and fraudulently has it delayed or repeated [50]. In this work, the replay attack replays the latest recorded PMU packets sent from the endpoints to the control center server to mislead the CVR to dispatch wrong control signals to the voltage regulator of the power grid, as shown in Figure 3.2. The replay attack has two atomic actions: packet record and packet replay. The record action will continuously capture the packets and save them into the agent's buffer, and the replay action will replay the last packet from the buffer. This work focuses on investigating the timing and ordering of launching these two atomic actions sequentially during the PT.

The CVR function is widely adopted by utilities for peak demand reduction and energy savings through reducing the voltage of distribution feeders for the ADN [123]. The deployed voltage regulator, known as the on-load tap changer (OLTC), changes its three-phase tap positions according to system voltage and keeps the minimal voltage within a low-level voltage band (CVR control band). For the centralized CVR function, the server will send control commands to increase or decrease the tap position by one tap, as shown in Figure 3.2.

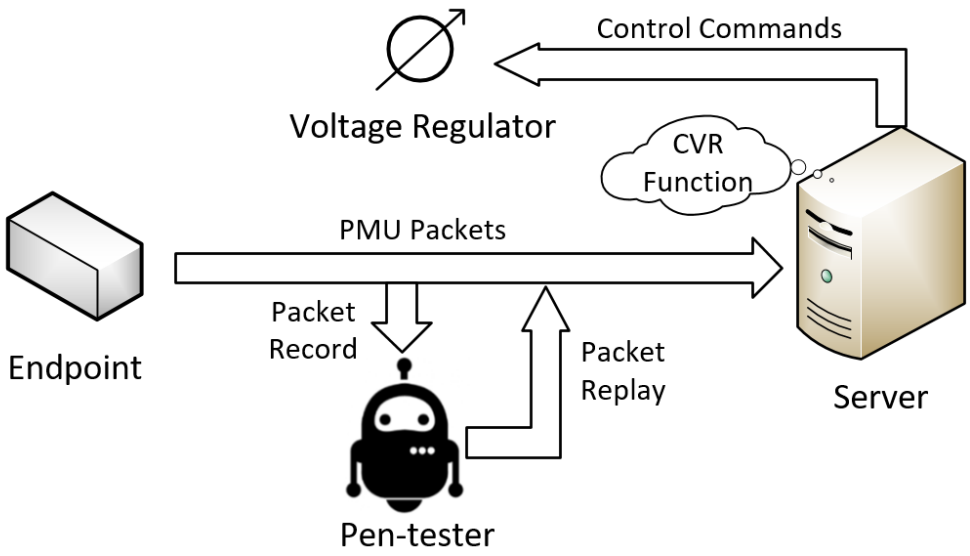


Figure 3.2: The replay attack scheme on PMU packets in the ADN.

3.4.2 MDP Design

Based on the general MDP formulation proposed in Section 3.3.2, we give a more specific MDP design for the case study.

State: The state vector s_t of the ADN at time t is defined as a vector of the three-phase voltage magnitude and three-phase voltage angle of every node bus in the ADN, as expressed in Eq. 3.1.

Action: The a_t taken by the agent at time t is selected from the action space $\mathcal{A}$ for replay attacks. According to Table 3.1, $\mathcal{A}$ can be defined as:

$$\mathcal{A} = \{a^{(1)}, a^{(2)}, a^{(3)}\}, \tag{3.5}$$

where $a^{(1)}$ is the *Packet Record* action, $a^{(2)}$ is the *Packet Replay* action, $a^{(3)}$ is the *Replay Stop* action. The action duration is set by the parameter δ. The *Packet Record* action will continuously capture PMU packets sent from the RTU of the PMUs and save them into the agent's buffer. The *Packet Replay* action will replay PMU packets starting from the latest one of the buffer. Therefore, this buffer can be considered as

a FILO memory.

Feature Vector: We assume that the agent is able to parse PMU packets based on the knowledge of protocols and obtain the node measurements of the target ADN. Let N_P be the number of PMUs deployed in the ADN. For each PMU, the pen-tester can obtain the voltage magnitude for each phase. If M samplings are performed during one action duration by the agent, we can directly extract features from these measurements into a vector f_t, as expressed as follows:

$$f_t = [V_a^-, V_{a-}, \tilde{V_a}, V_b^-, V_{b-}, \tilde{V_b}, V_c^-, V_{c-}, \tilde{V_c}], \tag{3.6}$$

where V_a^- is the maximal phase-A voltage magnitude among N PMUs averaged by M samples. V_{a-} is the minimal voltage magnitude of phase-A among N PMUs averaged by M samples; similarly, $\tilde{V_a}$ is the average voltage magnitude of phase-A among N PMUs averaged by M samples. The feature extractions for phase-B and phase-C are the same as that of phase-A.

The final feature vector ϕ_t is formulated as:

$$\phi_t = [f_t^c, f_t^m], \tag{3.7}$$

where f_t^c represents the extracted features from current measurements, f_t^m represents the extracted features from measurements obtained during the latest *Packet Record* action. f_t^m is an important part of the ϕ_t since the *Packet Replay* action may replay different packets from its buffer under the same system state, which may transfer the system to an unsure state, ignoring the state transition probability $P(s_{t+1}|s_t, a_t)$. However, if the f_t^m is included, the current state has the information of the *Packet Replay* action, making the next state deterministic.

Reward Function: Based on the general form of reward function defined by

Eq. 3.2, in this case study, we select the voltage magnitude as the power quality parameter to form the reward function, which is designed to reflect the voltage violation extent of the power grid. We apply the System Average Voltage Magnitude Violation Index (SAVMVI) proposed in [124] to shape this function as:

$$R(s_t, a_t, s_{t+1}) = \frac{1}{N_P}\frac{1}{M}\frac{1}{3}\sum_{i=1}^{N_v}\sum_{k=1}^{M}\sum_{p=1}^{3} VIO^{i,p,k}, \tag{3.8}$$

where the $VIO^{i,p,k}$ denotes the voltage magnitude violation of phase p obtained from the i^{th} PMU of k^{th} sample within the one action duration. More specifically, $VIO^{i,p,k}$ is defined as:

$$VIO^{i,p,k} = \begin{cases} V_{mag} - V_{max}, & \text{if } V_{mag} > V_{max} \\ 0, & \text{if } V_{min} \leq V_{mag} \leq V_{max} \\ V_{min} - V_{mag}, & \text{if } V_{mag} < V_{min}, \end{cases} \tag{3.9}$$

where V_{min} and V_{max} are the minimum and maximum acceptable values for steady-state voltage magnitudes, respectively.

3.4.3 PT Policy Optimization based on DQN

The objective function of PT, as defined in Eq. 3.3, aims to maximize the discounted accumulated rewards during the PT process. The solution to Eq. 3.3 yields the optimal PT policy π^*, which dictates the best action to take given a specific state. In order to determine the optimal PT policy, RL algorithms are employed to solve the MDP.

While traditional RL algorithms, such as Q-learning and SARSA [117], are effective for solving decision-making problems with finite and discrete state spaces, they

encounter the curse of dimensionality challenges when confronted with the PT problem, characterized by large and continuous state space due to the complexity nature of the cyber-physical smart grid. To overcome this challenge in the context of PT, DRL algorithms are applied, which integrate deep neural networks (DNNs) into RL algorithms. More specifically, we apply deep Q-network (DQN), as a typical DRL algorithm, to obtain the optimal PT. The framework of DQN is shown in Figure 3.3.

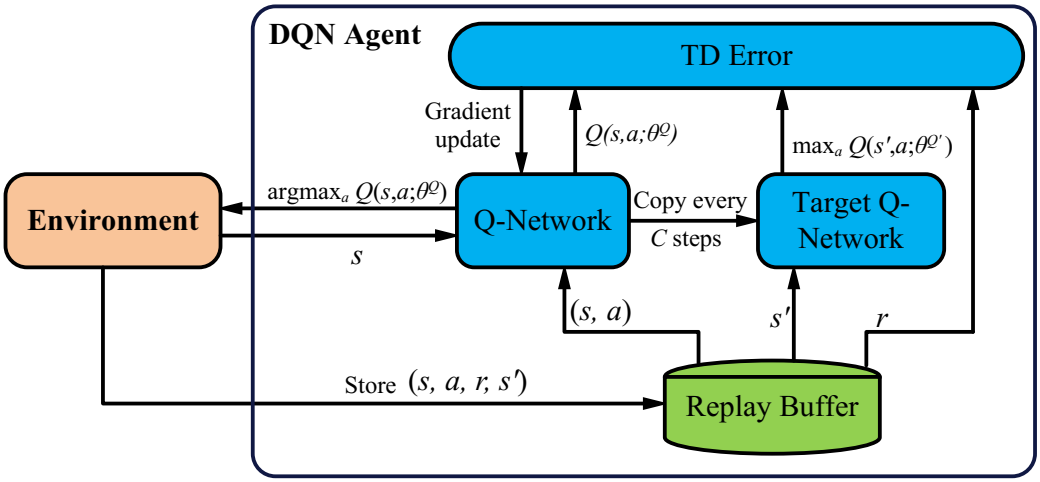


Figure 3.3: The framework of DQN algorithm.

The action value function $Q(s,a)$, which estimates the future accumulated rewards when taking action a under state s, is approximated by a neural network (Q-network) parameterized by θ^Q [125]. The update rule for the Q-networks is to minimize the mean square Temporal-Difference (TD) error as follows:

$$\mathcal{L}(\theta^Q) = \mathbb{E}[(Q(s_t, a_t; \theta^Q) - y)^2], \tag{3.10}$$

where $y = r_t + \gamma \max_a Q'(s_{t+1}, a; \theta^{Q'})$, $Q'(\cdot)$ is the target Q-networks parameterized by $\theta^{Q'}$.

The $Q(s,a)$ will be iteratively updated through the interactions between the agent

and the environment. Finally, the optimal action a_t^* at s_t can be obtained by:

$$a_t^* = \arg\max_{a_j \in \mathcal{A}} Q(s_t, a_j). \tag{3.11}$$

In this study, the input state of the Q function is replaced by the feature vector ϕ_t.

The pseudo-code of the DQN-based PT optimization is shown in Algorithm 3.1.

Algorithm 3.1 PT policy optimization based on DQN

Initialize the $Q(\cdot)$ and $Q'(\cdot)$ with parameter θ^Q and $\theta^{Q'}$;
Initialize the experience replay buffer $\mathcal{D}$;
for *episode* = 1 to N_{ep} **do**
    Get the initial feature vector ϕ_0 extracted from PMU packets;
    for $t = 1$ to $\mathcal{T}$ **do**
        The agent selects an action a_t from $\mathcal{A}$ based on the current feature vector ϕ_t following the ϵ-greedy strategy;
        The agent executes a_t on the AND and gets the new feature vector ϕ_{t+1};
        Calculate the reward r_t based on Eq. 3.8;
        Store transition $(\phi_t, a_t, r_t, \phi_{t+1})$ in D;
        Sample random minibatch of transitions from D, and update the θ^Q for $Q(\cdot)$ by minimizing the Eq. 3.10;
        Every C steps rest $\theta^{Q'} = \theta^Q$;
    end for
end for

The algorithm begins by initializing the main Q-network $Q(\cdot)$ and the target Q-network $Q'(\cdot)$ with respective parameters θ^Q and $\theta^{Q'}$. In addition, an experience replay buffer $\mathcal{D}$ is initialized to store and sample experiences from agent-environment interactions for policy training. During each episode, the algorithm progresses through a series of time steps, denoted by t, within a predefined time horizon $\mathcal{T}$. At each time step, the agent selects an action a_t from the action space $\mathcal{A}$ based on the current feature vector ϕ_t, employing an ϵ-greedy strategy to control exploration and exploitation of the action space. Subsequently, the agent executes the selected action on the

system and observes the resulting new feature vector ϕ_{t+1}. A reward r_t is then calculated according to the system's responses based on Eq. 3.8. The algorithm stores the transition tuple $(\phi_t, a_t, r_t, \phi_{t+1})$ in the experience replay buffer $\mathcal{D}$. Periodically, a random minibatch of transitions is sampled from $\mathcal{D}$, and the parameters θ^Q of the Q-network $Q(\cdot)$ are updated by minimizing the loss function specified in Eq. 3.10. Moreover, to stabilize the learning process, the target network parameters $\theta^{Q'}$ are synchronized with the main network parameters θ^Q every C steps. Finally, we will get the trained $Q(\cdot)$ as the final PT policy.

3.5 GridBattleSim: A Co-Simulation Platform for AutoPT on ADNs

To facilitate the study on the DRL-based PT for cyber-physical smart grids, a software-based co-simulation platform, called *GridBattleSim*, was developed, where the proposed solutions in Chapter 3 and 4 are validated. *GridBattleSim* is developed based on the research platform ASGARDS-H (Enabling Advanced Smart Grid Cyber-Physical Attacks, Risk and Data Studies with HELICS) [126]. *GridBattleSim* and ASGARDS-H are all based on the Hierarchical Engine for Large-scale Infrastructure Co-Simulation (HELICS) developed by the Pacific Northwest National Laboratory [127].

3.5.1 Platform Introduction

HELICS and ASGARDS-H

Before introducing *GridBattleSim*, we first briefly introduce HELICS and ASGARDS-H:

HELICS is an open-source cyber-physical co-simulation framework for energy

systems, with a strong tie to power systems [127]. It provides users with a high-performance way to integrate multiple simulators across a variety of computation platforms (Linux, Windows, etc.) and languages (C++, Python, etc.) from various domains. HELICS can co-simulate diverse simulators for power systems, including GridLAB-D, GridDyn, Matpower, Pypower, OpenDSS, PSLF, InterPSS, etc., but is also general enough to support many simulators in other domains by using HELICS API. Under the HELICS framework, a simulator or a program running the simulation model is called a "federate". These dedicated federates are time-synchronized by a HELICS core and a HELICS broker. HELICS core is the software that will be embedded inside the simulator to allow it to become a HELICS federate. HELICS broker is a special executable distributed with HELICS; it is responsible for performing two key tasks for co-simulation: (1) maintaining synchronization for federates and (2) facilitating message exchange between federates based on the ZeroMQ protocol [128].

Based on the HELICS framework, ASGARDS-H [126] was developed, which is a flexible co-simulation platform that can simulate cyberattacks on cyber-physical smart grids. It integrates four simulators and programs as federates, including the GridLAB-D for power grid simulation, OmNet++ for communication networks simulation, one Python program for launching the grid control functions, one Python program for launching cyberattacks through man-machine interaction, and one Python program for data monitoring and visualization. This platform allows users to test different cyberattack means on a distributed power grid, including eavesdropping, DoS/DDoS, replay attack, packet dropping, packet delay, and false data injection attack (FDIA), etc. However, it does not support users to perform AutoPT using specific strategies. It also does not implement a specific grid control program but leaves such development to users.

GridBattleSim

GridBattleSim is a secondary development based on ASGARDS-H. It inherits the basic structure of ASGARDS-H and develops dedicated programs and API for creating a co-simulation project, implementing the DRL-based AutoPT on specific power grids with specific grid control applications, making it a dedicated research platform for DRL-based AutoPT on ADNs.

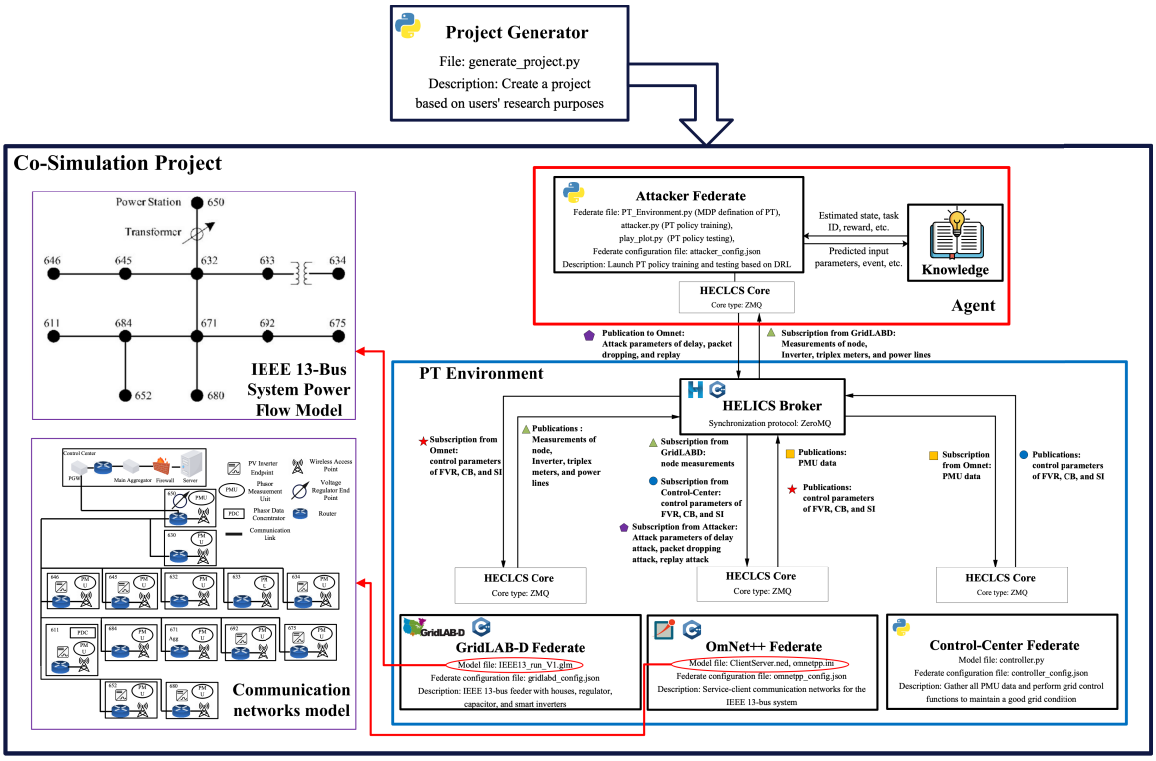


Figure 3.4: The framework of *GridBattleSim*.

Figure 3.4 shows the framework of *GridBattleSim*. In this framework, *GridBattleSim* adopts a **Project Generator** (a Python program used in ASGARDS-H but modified specifically for DRL-based AutoPT) to create a co-simulation project based on our research purposes. We can configure different cyber-physical settings of an ADN in terms of the physical part, cyber part, and cyberattack techniques, among

others. For the physical part of ADN, configurations include the load level, power generation level, PV penetration ratio, and weather-related parameters, among others. For the cyber part of ADN, configurations include PMU locations, PDC locations, and PDC-aggregated PMU locations, among others. For the cyberattack techniques, we will configure the hook layer for desired endpoints (assign which device can be attacked by which cyberattack techniques). In addition, many time-related settings should also be configured, including the co-simulation duration, PT starting time, and PT duration.

The generated co-simulation project consists of multiple dedicated simulators or programs for the physical power grid, communication networks, control center, and attacker, respectively. Under the HELICS co-simulation framework, simulators and programs are considered federates and are time-synchronized by a central HELICS broker. Data exchanges between federates are realized through publications/subscriptions to/from the HELICS broker. The applied federates of GridBattleSim are **GridLAB-D federate**, **OmNet++ federate**, **Control-Centre federate**, and **Attacker federate**.

In this platform, the physical process of the AND is simulated by the **GridLAB-D federate**; the endpoints, communication networks, and cyberattacks are simulated in the **OmNet++ federate**; the grid control functions, such as the CVR, are implemented in the **Control-Center federate**; the DRL-based AutoPT is implemented in the **Attacker federate**. In addition, a **knowledge module** is communicated with the Attacker federate and provides supporting information to the Attacker federate to assist its decision-making. The knowledge module will be used and explained in Chapter 4.

The **publications** and **subscriptions** for each federate are also illustrated in Figure 3.4. We use different colored shapes representing different publication-subscription

pairs to describe the information flow between federates. The GridLAB-D federate publishes node measurements subscribed by the RTU endpoints in OmNet++. These RTU endpoints will generate PMU packets and deliver them through communication networks to the control center server. When the server receives the PMU packets, it parses them and publishes the payload, which will be subscribed by the Control-Center federate.

The Control-Center federate performs the grid control function based on the received PMU data. The control commands for the voltage regulator will be published by the Control-Center federate and then subscribed by the server in the OmNet++ federate. The server will send the control commands through the network to the voltage regulators. The voltage regulator will publish the control parameters that will be subscribed by the GridLAB-D federate. The GridLAB-D federate then performs the power flow calculation to update measurements for the power grid. To make the co-simulation more realistic, in OmNet++, PMU packets are implemented with the C37-244-2013 synchrophasor protocol specifications. The control commands dispatched by the Control-Center server follow the Modbus-TCP protocol.

The GridLAB-D, OmNet++, and Control-Center federates constitute the PT environment. The Attacker federate plays as the RL agent: It launches cyberattacks (actions) by publishing parameters for attack actions subscribed by the OmNet++ federate; it also subscribes node measurements from the GridLAB-D federate to form its observation vectors for the DRL agent. The proposed testbed is flexible in configuring and testing different distribution systems, communication networks, centralized control strategies, and cyberattack strategies. The implemented cyberattacks include packet drop, packet delay, DoS, and replay attack, where we use replay attack as the main cyberattack technique to investigate DRL-based PT.

3.5.2 Platform Workflow

To study the DRL-based PT by using *GridBattleSim*, we need to follow three basic steps:

Step 1: Create a co-simulation project. In this step, we can generate a co-simulation project based on the research purpose by configuring parameters for the power grid and the attack capability of the pen-tester. The platform allows us to configure the load level, generation level, PMU location, and other parameters for the power grid. It also allows us to set the hook on endpoints that we want to attack. Here, the hook is an additional layer inserted between the network layer and the data link layer of a target endpoint. A hook can be configured to implement different cyberattacks, such as packet drop, packet delay, and packet replay, among others.

Step 2: Train the PT policy. Once we build a co-simulation project, we can train our agent to optimize the PT policy. Before launching the training, we need to configure the MDP or POMDP settings, including the action space, observation space, step function, reset function, and reward function. We also need to configure the hyperparameters of the RL/DRL algorithms. Then, we can run the training to optimize the PT policy. During the training, the Q-network will be saved.

Step 3: Test the PT policy. When the training is completed, we can test the trained PT policy, where we select the trained Q-network to generate the PT decisions.

3.6 Experimental Validation

3.6.1 Experiment Setup

AND Configurations

The standard IEEE 13-bus distribution feeder [129], as shown in Figure 3.5, is used as a skeleton to build the cyber-physical ADN, serving as the PT environment. The configurations include settings for the physical power grid, communication networks, and the control center. Based on this standard feeder, we added residential houses, PV stations, and a voltage regulator to the system. Configuration parameters and locations of these entities are listed in Table 3.2.

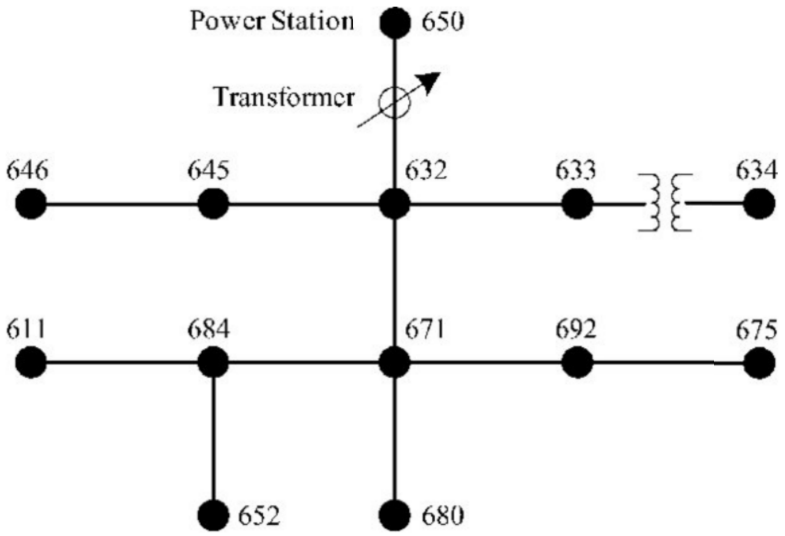


Figure 3.5: IEEE 13-bus distribution feeder.

For the communication network, we add one router in each bus node to create a star-type topology. Each bus node has one PMU endpoint. Some bus nodes also have endpoints for smart inverters or voltage regulators. All endpoints are wirelessly connected to their local access points. The communication network is visualized in Figure 3.6. Moreover, the sampling interval for PMU measurement is 0.3s. The data rate of the communication network is 50Mbps.

Table 3.2: Configurations of the ADN entities

Name	Node Locations	Configurations
Houses	646, 645, 634, 611, 692, 675, 652, 680	All houses equip with lights, HVACs, water heaters
PV stations	646, 645, 634, 611, 692, 675, 652, 680	Smart inverters operate on unity-power-factor mode
Voltage regulator	650	The tap position ranges from -16 to +16

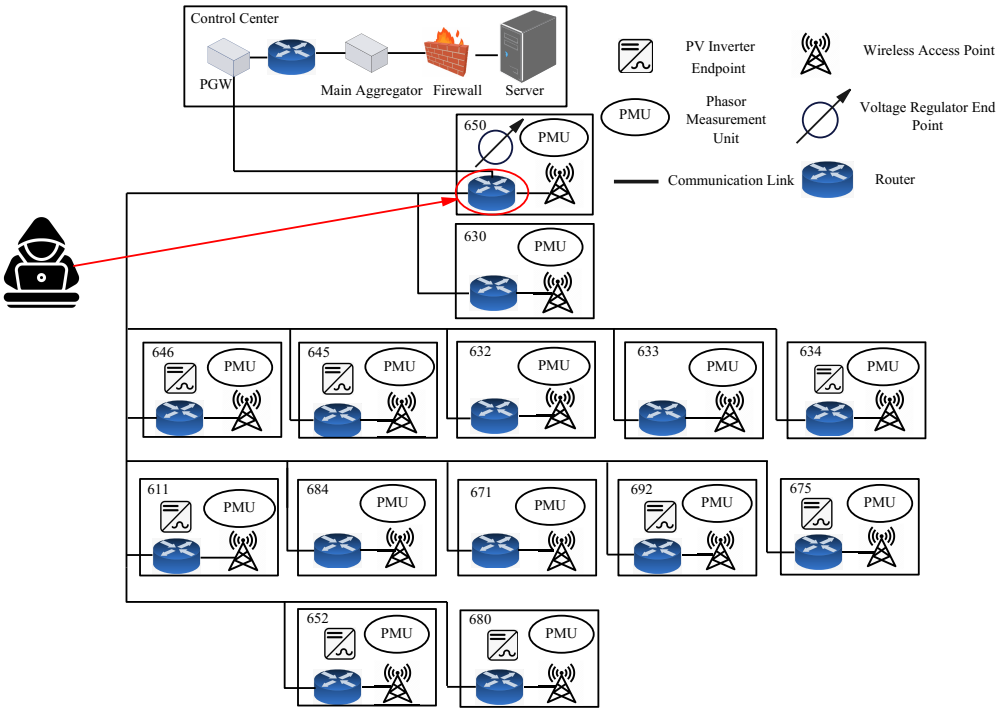


Figure 3.6: The communication network for the ADN.

PT Configurations

The replay attack will record and replay outgoing packets from compromised endpoints to external networks, implemented as a replay attack hook [126] inserted between the data link layer and the network layer of the router at Node 650 in OmNet++, as shown in Figure 3.7. This means the attacker is able to record and replay PMU packets from all PMU endpoints. The PT duration is 30s, and the duration of each attack is 3s.

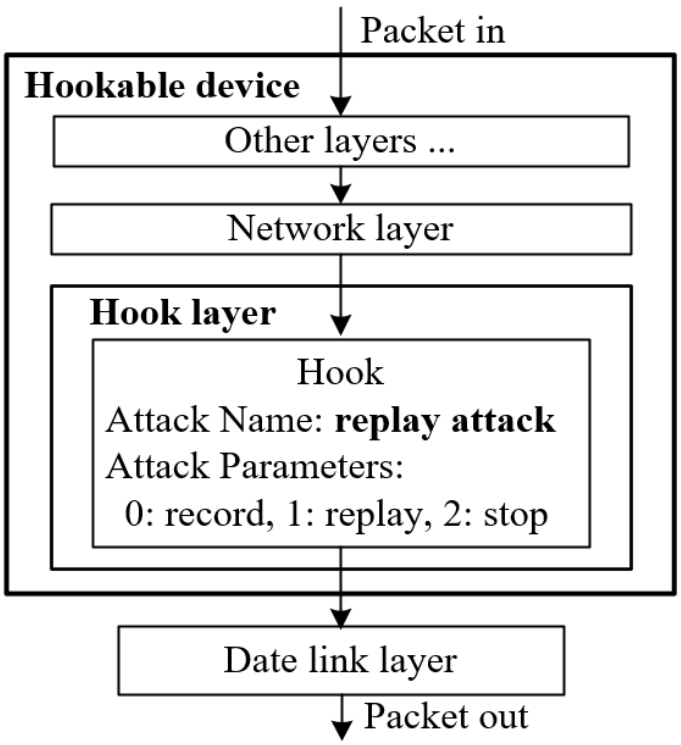


Figure 3.7: The replay attack hook implementation in OmNet++.

For the control center, the CVR control function will adjust the tap positions of the voltage regulator for each phase according to the minimal measured voltage from PMUs. The CVR control set-point is set to 0.962 p.u., and the width of the CVR control band is set to 0.012 p.u.

Hyperparameters of the DQN-based Agent

The DQN hyperparameters are selected empirically based on a number of experiments and evaluations. The entire training process takes 6,000 episodes, during which ϵ, a parameter that controls the action selection strategy, changes linearly from 0.5 to 0.1 to encourage exploration at the beginning and exploitation at the end. The learning rate for updating the Q-network is 0.001, with 30 neurons in each of the two hidden layers. The discount factor γ is fixed at 1.0.

3.6.2 Simulation Results

We investigated the training performance of PT in different scenarios with different attack difficulties. There are three main factors, namely load level, PV generation percentage, and irradiation drop ratio, that can affect the difficulty of PT. The load level is controlled by the number of houses in each phase. The PV generation percentage is the ratio of PV power to the load consumption, which can be controlled by the PV penetration ratio of the system. The irradiation drop ratio reflects the changing percentage of solar irradiation caused by changing weather.

Load Demand in PT Difficulty

We first created three scenarios with two different load levels (100 houses, 50 houses, 20 houses). The other two factors are fixed with a 100% PV penetration ratio and a 90% irradiation drop starting from 5s and lasting for 6s. The PTs are launched in the noon when the irradiation level is around 850 W/m^2.

In Scenario-1, the number of houses for each phase is 100. Figure 3.9a shows the lowest 3-phase voltages in the system when no attack is launched. When the irradiation drops, the voltages of each phase fluctuate for some time and trigger the CVR control simultaneously when the voltages exceed the CVR lower and upper

bounds. The CVR control will adjust the tap positions of the voltage regulator to maintain the lowest voltage of each phase within the CVR control band (shown as the black dotted lines in Figure 3.9a.

Figure 3.8 shows the training log of the PT agent based on DQN, where the solid blue curve represents the episode rewards; the solid orange and green curve shows the moving average of 10 and 20 episode rewards, respectively. From Figure 3.8, we can see that the episode reward increases with the number of episodes, suggesting that the DQN is improving the policy based on the designed MDP during the PT process.

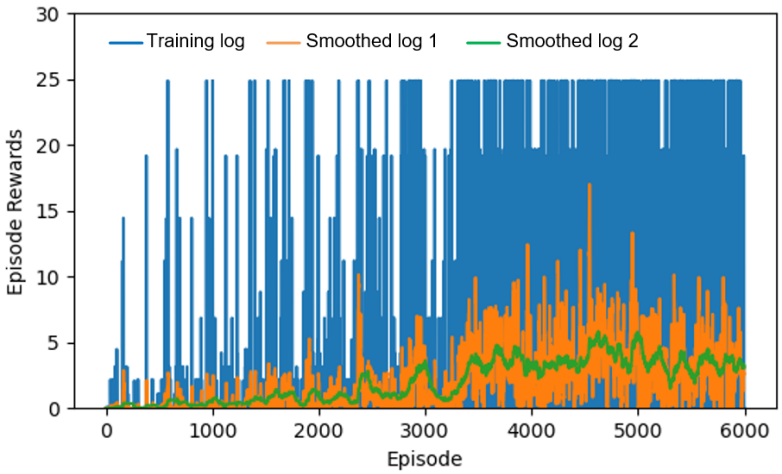
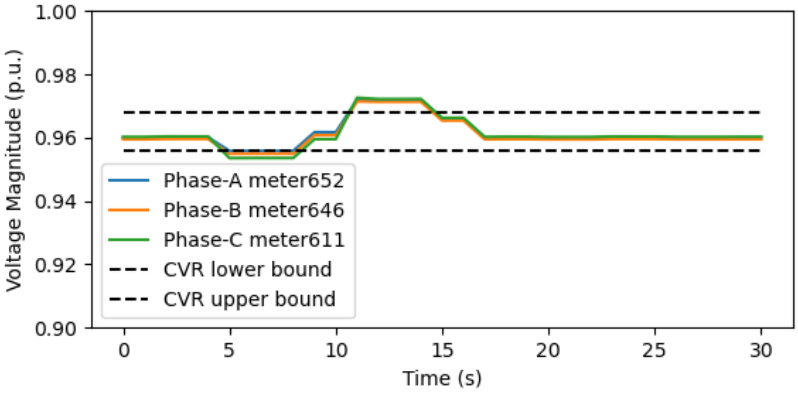


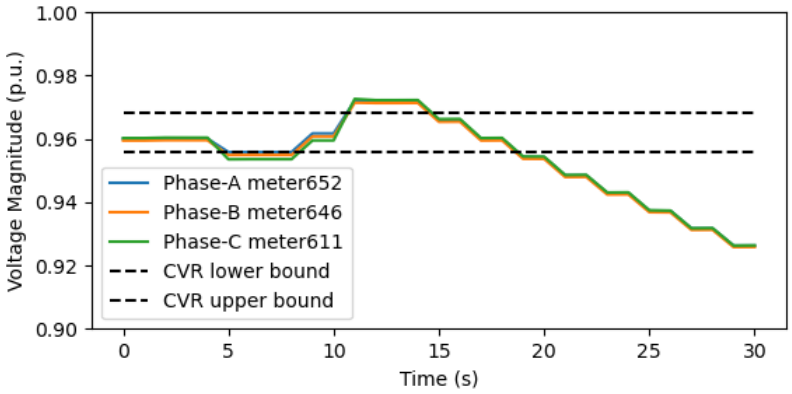
Figure 3.8: The training process for the DRL-based PT in Scenario-1.

We also captured 6 PT policies in different training stages to illustrate how the agent learns to improve the attach path. As shown in Figure 3.10, 6 attack paths are generated by Policies A to F captured at 1,000th to 6,000th episodes, respectively, with the corresponding episode rewards for each policy.

Policy A is a null policy that launches the Stop action from the start to the end, which cannot have any impact on the system. After 1,000 epochs, Policy B launches 2 consecutive Record actions between 6s and 12s, then launches 3 consecutive Replay actions starting from 21s. While the final episode reward is 1.11, which is relatively lower than that of subsequent policies, it is an important improvement over Policy A



(a)



(b)

Figure 3.9: The lowest voltage of each phase under (a) a non-attack condition and (b) the attack path of the trained PT policy in Scenario-1.

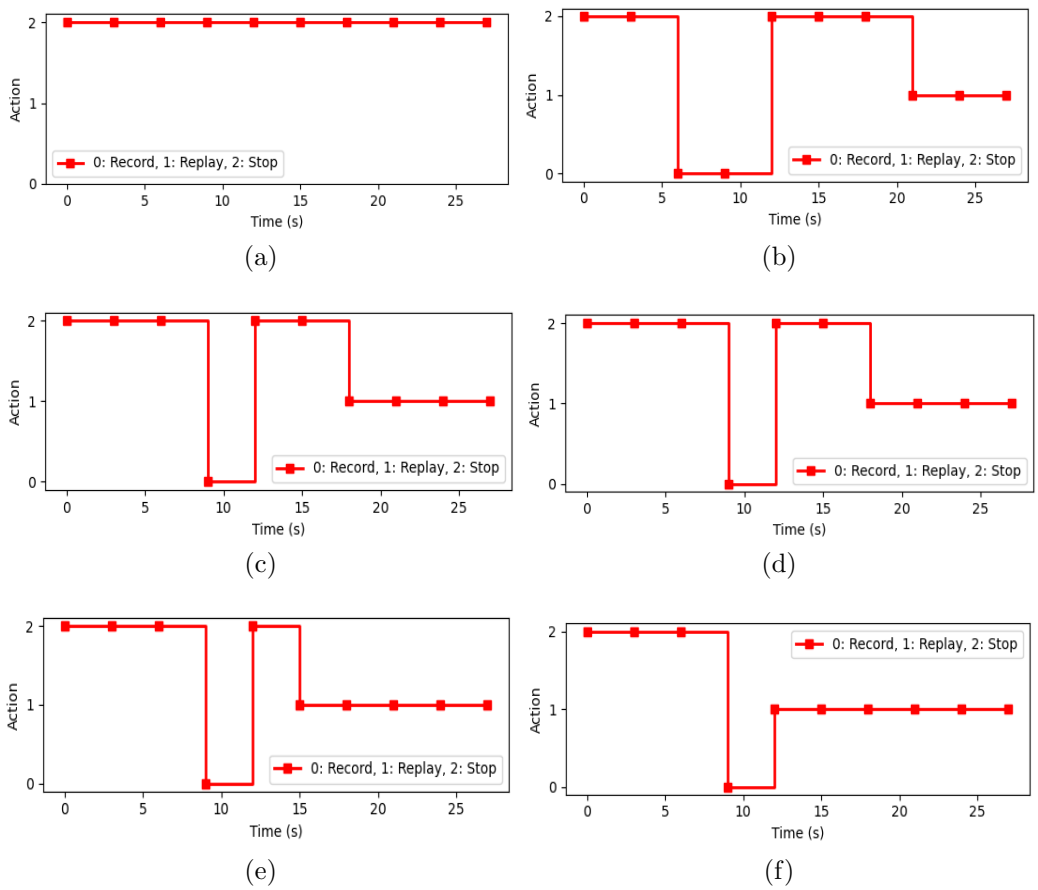


Figure 3.10: PT policies in different stages of the training process in Scenario-1: (a) Policy A (episode reward = 0), (b) Policy B (episode reward = 1.11), (c) Policy C (episode reward = 6.87), (d) Policy D (episode reward = 6.87), (e) Policy E (episode reward = 19.68), (f) Policy F (episode reward = 24.89).

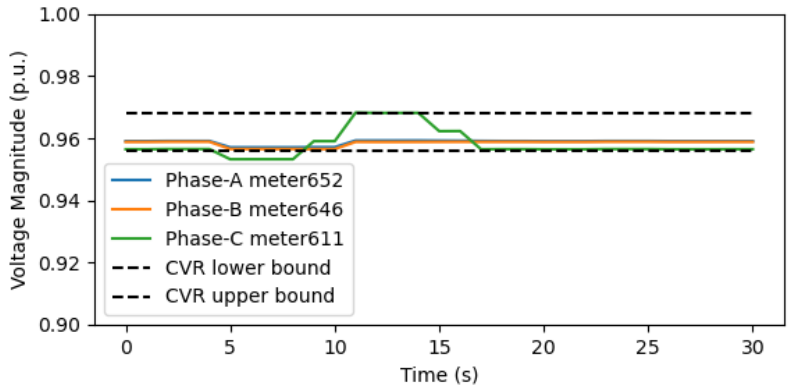
because Policy B is able to capture the packets when the lowest voltage is higher than the upper limit of the CVR control (as shown in Figure 3.9a). The second recorded packets were later replayed three times and triggered the movement of the tap position towards the lower side. The last replay action triggered the voltage violation on the lower limit (0.95 p.u.).

Then, after another 1,000 epochs, Policy C only launches 1 Record action between 9s and 12s and then launches 4 consecutive replay actions starting from 18s. Compared to Policy B, Policy C further improves the attack path by extending the duration of the Replay action, which can bring more negative impacts to the system. It also neglected the first Record action in Policy A, because the subsequent Replay action only replays the latest recorded packet from the attacker's replay memory, making it unnecessary to record the first packet that is not important in the attack path.

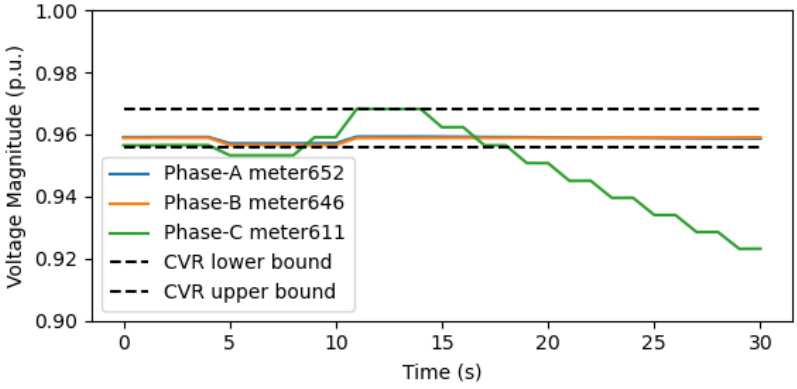
For the remaining policies, while Policy D at 4,000 episodes keeps the same policy as Policy C, Policy E at 5,000 episodes extends the repetition of Replay from 4 to 5, which further extends the attack impact. Eventually, Policy F learns to remove all Stop actions after the Record action; it replays packets immediately when it captures the voltage violation on the CVR control upper bound, making the voltage over-limit as quickly as possible before the CVR control can recover the voltage. Compared with the previous policies, this final policy brings more negative impacts within this PT period and gets a 24.89 episode reward finally. Figure 3.9b shows the lowest voltage of each phase when the agent applies the final Policy F. It can be seen that the voltage starts to violate the lower limit (0.95 p.u.) after 20s and continuously decreases until the end.

Scenario-2 has 50 houses in each phase to represent a lower load level. Figures 3.11a and 3.11b show the lowest 3-phase voltages under non-attack and the

trained attack path, respectively. In this scenario, only the lowest voltage in phase-C violates the CVR band and triggers the CVR control when there is no attack. When we launch the PT with the trained policy, which is the same as Figure 3.10f, only phase-C voltages violate the lower limit and finally obtain a 13.26 episode reward, which is lower than that of Scenario-1.



(a)



(b)

Figure 3.11: The lowest voltage of each phase under (a) a non-attack condition and (b) the attack path of the trained PT policy in Scenario-2.

Scenario-3 has 20 houses in each phase to represent the lowest load level. Figure 3.12 shows the lowest 3-phase voltages when there is no attack in the system. In this scenario, the voltages in each phase do not violate the CVR control band when the irradiation drops, so they cannot trigger the CVR control.

The PT training was also performed under Scenario-3 using DRL. However, the

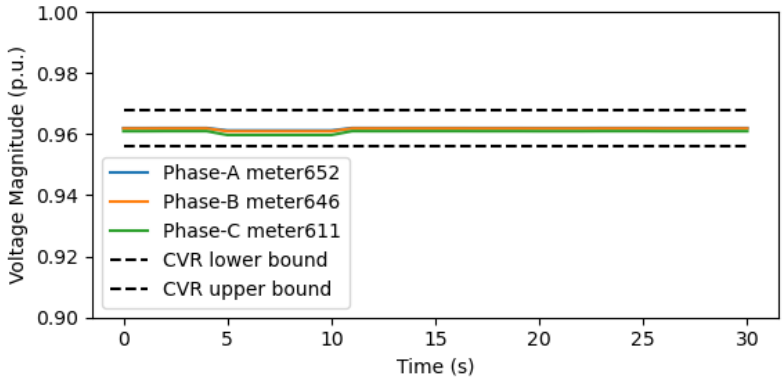


Figure 3.12: The lowest voltage of each phase under non-attack condition in Scenario-3.

episode rewards are all 0 during the training, which means that the agent cannot learn from the interaction with the PT environment. Since no actions can bring positive rewards to the agent, the agent simply keeps the Stop action from the beginning to the end.

From the experiments of Scenario-1 to 3, when the load increases to the level that can trigger the CVR control, the PT agent is able to learn from the environment. A higher load level will bring greater impacts on the system stability.

PV Generation Percentage in PT Difficulty

Scenario-4, which has a 50% PV penetration ratio, is compared with Scenario-2 to investigate the influence of the PV generation percentage. The load level is fixed with 50 houses in each phase, and the irradiation drop is set to 90%. From Figure 3.13, similar to Scenario-3, voltages in each phase do not violate the CVR control band under the non-attack condition, which means the agent cannot learn from the environment.

However, when we set the PV penetration ratio to 120% and set the number of houses to 20 in Scenario-5 to make a comparison with Scenario-3, we can find phase-A voltages violate the CVR control band, as shown in Figure 3.14a, which can be

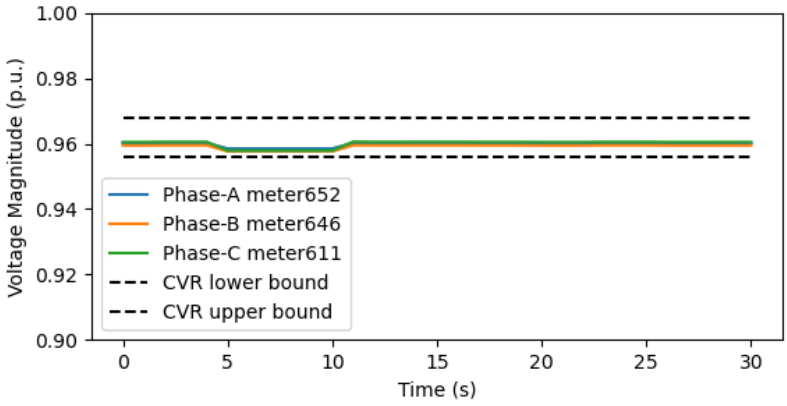


Figure 3.13: The lowest voltage of each phase under non-attack condition in Scenario-4.

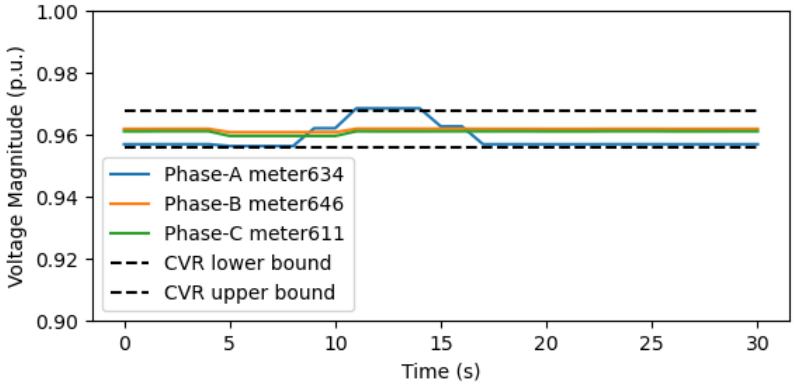
captured and learned by the agent. Figure 3.14b shows the impact of the trained PT policy launched in Scenario-5, in which the agent can receive an episode reward of 12.43.

Irradiation Drop Ratio in PT Difficulty

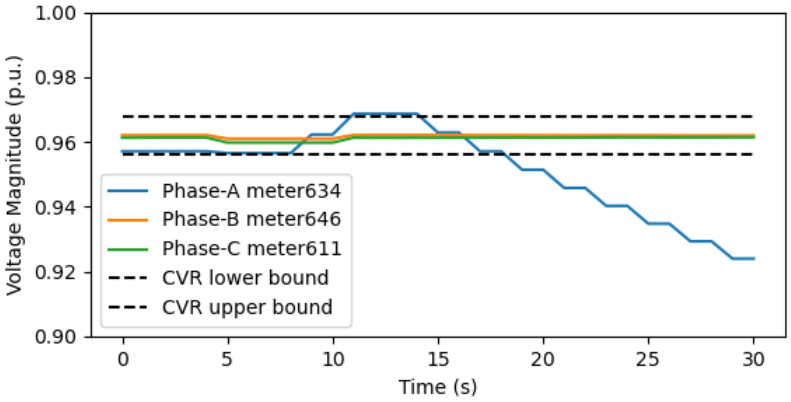
We also created Scenario-6 with a 20% irradiation drop ratio to represent a slight weather change. Compared with Scenario-2, as shown in Figure 3.15, since there is no violation on the CVR control band, the agent cannot capture packets that can trigger the movement of the tap position of the voltage regulator. Thus, the agent cannot also learn from this scenario.

Comparative studies

We also compared the DRL-based PT policy in Scenario-1 with the random policy, where actions are selected with equal probabilities, and a human policy that records the packets when the system experiences the voltage reduction and replays them afterward. For the random policy, since it has a lower probability to capture the CVR boundary violations and replay them continuously, in most of the cases, it



(a)



(b)

Figure 3.14: The lowest voltage of each phase under (a) non-attack condition and (b) the attack path of the trained PT policy in Scenario-5.

cannot bring negative impacts, and the voltage curves remain the same as the none-attack condition like Figure 3.9a. For the human policy, since the replayed packets are measurements lower than the CVR lower bound, it will trigger the tap position increase of the voltage regulator and lead to the voltage violation on the higher limit (l.05 p.u.) finally. However, compared with the DRL-based policy, this human policy will take more time to achieve the voltage violation (for DRL-based policy, less than 30s; for human policy, more than 35s). This is because, under the CVR control scheme, the system voltage is approaching 0.95 p.u., which is relatively far away from 1.05 p.u.

From the experiments above, we can see that increasing the load level, power

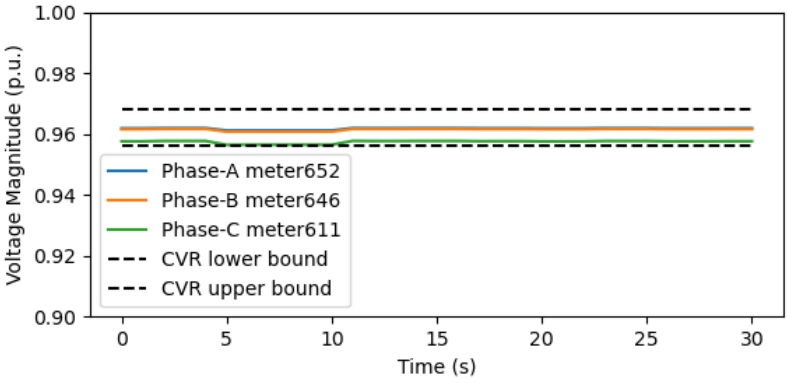


Figure 3.15: The lowest voltage of each phase under non-attack condition in Scenario-6.

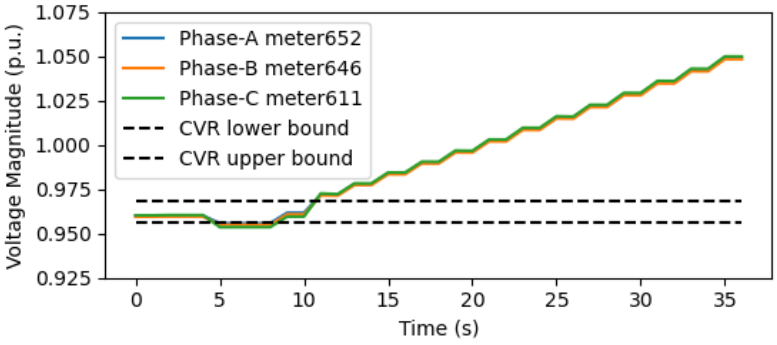


Figure 3.16: The lowest voltage of each phase under human policy in Scenario-1.

generation ratio, and irradiation drop ratio can all decrease the difficulty of the PT since they have the chance to make the voltages violate the CVR control band that can be captured by the agent. Moreover, compared with high load level conditions, to make more impact through PT, the lower load level scenario requires greater power generation.

3.7 Conclusions

To improve the efficiency and non-comprehensiveness of common PT for cyber-physical smart grids, this chapter proposed a DRL-based PT framework to efficiently and adaptively identify critical vulnerabilities for ADNs. Using replay attacks on ADNs with CVR control as a study case, this chapter modeled the attack as an MDP, where the state space, action space, feature vectors, and reward function are designed specifically. The DQN algorithm was adopted to solve the MDP and learn the optimal timing and ordering of replay attacks toward maximal voltage instability in different operating scenarios. A cyber-physical co-simulation platform, named *GridBattleSim*, was developed as a sandbox environment to train the DRL agent. This platform integrates dedicated simulators for the physical part, cyber part, control part, and attacker part of ADNs. Using *GridBattleSim*, scenarios with different levels of difficulty were tested to validate the learning capability and performance in finding critical attack paths of the DRL-based PT. The simulation results showed that DRL-based PT can learn to find the optimal attack path against system stability when the grid is under high load demand, solar power generation, and weather variation. These results are promising first steps toward a highly customizable framework for testing complex cyber-physical power systems with automated DRL agents and various attack schemes.

Chapter 4

DRL-based AutoPT under Partially Observable Conditions for ADNs[1]

4.1 Problem Statement

While the MDP formulation and DRL methods proposed in Chapter 3 provide a valuable framework for automating PT and improving its efficiency, the assumption of complete system state visibility may not be feasible in real-world scenarios. In practice, a pen-tester aiming to infiltrate a system and cause critical impacts typically only has access to a subset of the system's digital resources. As a result, the tester must carefully select their actions based on the limited information available locally. The lack of a holistic view of the system's state poses challenges to PT, as it obscures the potential impact beyond the compromised digital assets. This limitation restricts the agent's ability to make well-informed decisions, ultimately reducing the efficacy of PT strategies.

[1]This chapter was submitted to IEEE Transactions on Dependable and Secure Computing.

Moreover, the incomplete visibility of the system state introduces uncertainties in PT that can impede the effectiveness of PT efforts. Without a comprehensive understanding of the system's overall state, the penetration tester may overlook critical vulnerabilities or fail to anticipate the full extent of the potential damage that could be inflicted. This limited perspective undermines the accuracy of the vulnerability identification. Therefore, addressing the challenge of incomplete system state visibility is crucial for enhancing the robustness and reliability of AutoPT.

Solution Overview:

To address the aforementioned limitations in the context of AutoPT for ADNs, an improved approach utilizing a partial observable Markov Decision Process (POMDP) formulation is proposed. This framework extends the MDP formulation to accommodate scenarios characterized by partial observability. Building upon the case study detailed in Chapter 3, the study is also investigating the identification of optimal replay attacks capable of triggering grid voltage violations by leveraging a subset of information gathered by the agent during PT.

In order to tackle the uncertainty of PT introduced by partial observability, a physical model of the power grid was established to enable the agent to infer the system's complete state from the local data it can access to. Subsequently, the POMDP problem can be reformulated into an MDP problem, thereby enabling the application of conventional RL/DRL algorithms to obtain the optimal PT policy.

4.2 Related Works

In recent years, there are many works applying POMDP to formulate the process of PT or vulnerability analysis. Authors in [86] and [130] proposed the use of POMDP to model PT in network security, providing a more accurate representation of attack planning in PT compared to traditional planning methods, allowing intelligent

decision-making, integrating scanning actions with actual exploits to enhance the effectiveness of automated PT tools.

Ref. [131] highlighted the shortcomings of PT methods, such as classical planning and previous POMDP-based approaches, which fail to accurately simulate real-world scenarios and scale effectively. To address these issues, the authors modeled the PT as a partially observable contingent problem, accommodating partial observability and non-deterministic action effects. The experimental findings suggest that the contingent planning approach provides a more sophisticated and scalable solution for PT, surpassing the capabilities of conventional methods.

Ref. [2] pinpointed the inefficiencies in current PT techniques, which are non-standardized, intricate, and resource-heavy. To mitigate these issues, the authors introduced the Intelligent Automated Penetration Testing System (IAPTS), which operates by conceptualizing PT environments and tasks as a POMDP problem that was solved by RL. The experimental validations suggested that RL can significantly enhance PT, outperforming human capabilities in terms of time efficiency, coverage of attack vectors, and PT accuracy and reliability.

Ref. [132] identified a gap in AutoPT where existing methods do not fully account for uncertainties caused by partial network observability and network changes made by the defender. To address this, the authors proposed a new framework based on the POMDP that incorporates the defenders' behaviour as an information decay factor. Two models are proposed: D-PenTesting, which assumes the decay factor is known beforehand, and LD-PenTesting, which learns the decay factor during the process. Both models, when tested on two benchmark scenarios, outperformed existing POMDP-based PT methods and demonstrated greater robustness.

Ref. [133] modeled the PT as a black-box POMDP and proposed an algorithm,

called ND^{3}RQN, to optimize the PT. The algorithm utilizes a Long Short-Term Memory (LSTM) structure to make decisions based on historical memory and enhances performance through neural network structure adjustments. Experimental results indicate that ND^{3}RQN outperforms other state-of-the-art algorithms to find greater attack paths and exhibit superior generalization and robustness in various simulation scenarios.

To overcome the challenges in AutoPT due to the incomplete knowledge of target network systems and the challenges of managing uncertainties, Ref. [134] introduced the EPPTA (efficient POMDP-driven PT agent), which utilizes an asynchronous RL framework equipped with an implicit belief module to estimate the current environment state. EPPTA significantly reduces the convergence time in training the PT policies, achieving about a 20-fold acceleration compared to existing methods.

4.3 Proposed Methodology

4.3.1 General POMDP Formulation for AutoPT

A partially observable MDP (POMDP) serves as a generalization of the MDP, aiming to model the decision-making process of an agent under the assumption that the system dynamics adhere to an MDP, yet the agent lacks direct access to the complete underlying state of the system [135].

Based on the proposed DRL-based PT framework illustrated in Figure 3.1, the proposed PT is formulated as a POMDP in this chapter, which is defined by a tuple consisting of seven components $\langle \mathcal{S}, \mathcal{A}, \mathcal{O}, T(s_{t+1}|s_t, a_t), O(o_t|s_t), R(o_t, a_t, o_{t+1}), \gamma \rangle$ [121]. $\mathcal{S}$ denotes the set of system states with $s_t \in \mathcal{S}$ representing the state at time t. $\mathcal{O}$ denotes the set of observations the agent can observe, and $o_t \in \mathcal{O}$ represents the observation at time t; $\mathcal{A}$ represents the set of actions with $a_t \in \mathcal{A}$ denoting the action

taken by the agent at time t. $T(s_{t+1}|s_t, a_t)$ denotes the state transition probability when the agent performs an action a_t in the current state s_t. $O(o_t|s_t)$ denotes the emission function that maps the current state to the observation the agent receives. The reward function $R(o_t, a_t, o_{t+1})$ calculates the immediate reward received by the agent after the action is performed. The constant $\gamma \in [0, 1)$ represents the reward discount factor across time. Due to the intricate nature of cyber-physical smart grids, the pen-tester faces challenges in determining $T(s_{t+1}|s_t, a_t)$ and $O(o_t|s_t)$. Therefore, model-free RL/DRL are still the primary methods to solve the POMDP and obtain the optimal PT policies.

The state space, action space, reward function, and objective function in this context align with the design provided in Section 3.3.2 of Chapter 3. The state of the environment is represented by the vector comprising voltage magnitude and voltage angle values for each bus node, as outlined in Eq. 3.1. The action space encompasses a range of cyberattack techniques commonly employed in PT, detailed in Table 3.1. A generic reward function for PT is structured to capture deviations in various power quality parameters, as depicted in Eq. 3.2. The objective function for PT aims at maximizing the discounted cumulative rewards, as expressed in Eq. 3.3.

4.3.2 Belief State

Since the pen-tester lacks the global perception of the system state, it cannot directly observe the attack impact beyond the observable area. To overcome this limitation and transfer the POMDP to MDP that can be solved by standard RL/DRL algorithms, the belief of the state (belief state) is utilized, which is the estimation of the system's full state based on the agent's observation from the environment. The belief state of the smart grid is generated by performing state estimation in the agent side, which combines the pen-tester's prior knowledge of the power grid and its local

observations.

The open-source intelligence techniques (OSINTs) used by attackers have already been shown to be an effective means of gaining knowledge about the target power grid through publicly available sources. The knowledge about the power grid includes the topology, feeder parameters, number of residential houses, and distribution of distributed energy resources (DERs) per node, among others [111]. Therefore, by utilizing OSINTs, the pen-tester is able to establish a physical model, such as the power flow model, of the target power grid. This physical model is capable of simulating the operation process of the power grid, from which the estimated state can be obtained if the input of the physical model can be well-configured. Additionally, the power consumption and generation of each node usually follow a similar pattern or profile, which allows the pen-tester to make a general estimation of the state of the power grid based on gathered measurements from certain locations.

Figure 4.1 illustrates the proposed belief state estimation process for PT on smart grids.

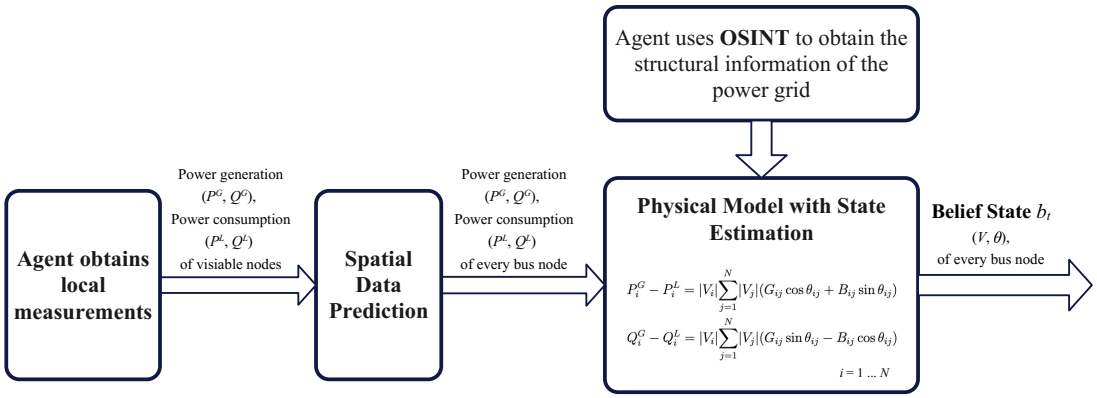


Figure 4.1: The belief state estimation based on OSINT and local observations.

The agent begins by using OSINT to gather enough structural information about the target power grid, including the topology, the profile of physical components, the impedance of each component, the profile of residential houses and DERs of the grid, etc. The gathered information is to establish a power flow model of the power

grid. The power flow model, also known as the load flow model, is a mathematical representation used to analyze the flow of electrical power through a transmission or distribution network. The model helps determine the voltage magnitude and voltage angle at each bus node, the real and reactive power flowing in each line, and each node under steady-state conditions [136].

Based on the established power flow model, the agent can use it to generate the belief state. During the PT, the agent will obtain some local measurements from compromised digital assets or captured packets. These local measurements can include the active/reactive power generation (P^G/Q^G) and active/reactive power consumption (P^L/Q^L) of visible bus nodes. Using these measurements, the agent can perform a spatial data prediction for other invisible nodes based on the assumption that the power consumption and generation of each node for a power grid in a region usually follow a similar pattern. Since the agent already knows the profile of residential houses and DERs by using OSINT, the agent is able to estimate the power consumption and generation of invisible nodes by scaling on the measurements from visible nodes according to their ratio of the rated power.

The output of the spatial data prediction is the power generation and power consumption for every bus node, which will be the input of the established power flow model. By running the state estimation over the power flow model, the agent can obtain the voltage magnitude and voltage angle of every bus node, which will form the belief state vector b_t.

After getting the belief state, the POMDP can be transferred to an MDP that can be solved by traditional RL/DRL methods.

4.4 Case Study on ADNs Using POMDP Formulation

In this case study, our focus remains on examining the replay attack on PMU packets as the primary cyberattack technique within an AND with Conservation CVR as the grid control application. Our objective is to determine the most effective timing and sequence for executing replay attacks to induce voltage instability within the ADN. This investigation is conducted under the constraint of partial observability, where the agent's ability is restricted to capturing only a limited number of PMU packets.

4.4.1 POMDP Design

Based on the general POMDP formulation proposed in Section 4.3.1, we give a more specific POMDP design for the case study.

State: The state vector s_t of the ADN at time t is still defined as a vector of the three-phase voltage magnitude and three-phase voltage angle of every node bus in the ADN, as expressed in Eq. 3.1.

Action: The a_t taken by the agent at time t is selected from the action space $\mathcal{A} = \{a^{(1)}, a^{(2)}, a^{(3)}\}$, for replay attacks, where $a^{(1)}$ is the *Packet Record* action, $a^{(2)}$ is the *Packet Replay* action, $a^{(3)}$ is the *Replay Stop* action. The action duration is set by the parameter δ. The *Packet Record* action will continuously capture PMU packets sent from the RTU of the PMUs and save them into the agent's buffer. The *Packet Replay* action will replay PMU packets starting from the latest one of the buffer. Therefore, this buffer can be considered as a FILO memory.

Belief State: Based on the proposed belief state generation process illustrated in Figure 4.1, the pen-tester establishes a distribution system power flow model of the

target ADN based on OSINTs. During the PT, the pen-tester obtains the measurements from captured PMU packets for visible nodes, including the voltage magnitude, voltage angle, active/reactive power consumption, and active/reactive power generation. The agent then performs the spatial data prediction on the power consumption and generation for other invisible nodes based on the obtained local measurements, which allows the agent to obtain the global view of the power consumption and generation of all nodes of the ADN. Finally, the agent combined the measured and predicted power consumption and generation as input to the power flow model and ran the state estimation to obtain the belief state b_t (voltage magnitude and angle of all nodes).

Belief Feature Vector: When the agent obtains the belief state b_t of the power grid, it is able to extract feature information from it to form the belief feature vector ϕ_t^b. If a sequence of M samples of PMU packets is obtained during a single action duration, the relevant belief feature can be extracted from b_t and represented as:

$$f_t^b = [V_a^-, V_{a-}, \tilde{V_a}, V_b^-, V_{b-}, \tilde{V_b}, V_c^-, V_{c-}, \tilde{V_c}], \tag{4.1}$$

where V_a^- represents the maximum phase-A voltage magnitude among all bus nodes averaged over M samples. Similarly, V_{a-} is the minimum phase-A voltage magnitude among all nodes averaged over M samples, and $\tilde{V_a}$ is the average phase-A voltage magnitude among all nodes averaged over M samples. The feature extraction for phase-B and phase-C follows the same procedure as phase-A.

We can then define the final belief feature vector as:

$$\phi_t^b = [f_t^{bc}, f_t^{bm}], \tag{4.2}$$

where f_t^{bc} represents belief features extracted from current b_t, and f_t^{bm} represents

belief features extracted from belief state obtained during the previous *Packet Record* action. The inclusion of f_t^{bm} is still crucial as the *Packet Replay* action may replay different packets from its buffer under the same system state, which may transfer the system to an uncertain state, ignoring the state transition probability $T(s_{t+1}|s_t, a_t)$. However, with the inclusion of f_t^{bm}, the current observation contains information about the *Packet Replay* action, making the next state deterministic.

Reward Function: Based on the general form of the reward function defined by Eq. 3.2, in this case study, we still select the voltage magnitude as the target power quality parameter to form the reward function. The agent hopes to trigger voltage violations through cyberattacks. To achieve this goal, two reward functions are designed. We adopt Eq. 3.8 designed in Section 3.4 as the default reward function (we call it Reward Function I), which employs SAVMVI to measure the extent of voltage violation in the system.

The results of the experimental validation section reveal that the episode rewards during the training phase are consistently 0. This is because the Reward Function I only rewards the agent when the voltage violation is observed, which usually takes a relatively long exploration time for the agent. Therefore, most of the time, the agent lacks feedback in the form of rewards, which hinders the agent's learning efficiency. To address the issue of sparse rewards and improve learning efficiency, Reward Function II is proposed, as presented in Eq. 4.3. Reward Function II incentivizes the agent by breaking down the overall objective into some smaller achievable goals. It is designed based on the assumption that the pen-tester is able to obtain knowledge of the CVR control program that aims to maintain the system's lowest voltage within the CVR control band. As such, the reward function gives +0.01 when the voltage falls outside the CVR control band, +1 when the voltage violates the system constraints, and -0.01 when the voltage stays within the CVR control band.

$$R(s_t, a_t, s_{t+1}) = \frac{1}{M}\frac{1}{3}\sum_{k=1}^{M}\sum_{p=1}^{3}(\frac{1}{N_P}\sum_{i=1}^{N_P} r_v^{i,p,k} + r_c^{p,k}) \tag{4.3}$$

where $r_v^{i,p,k}$ and $r_c^{p,k}$ are respectively defined as:

$$r_v^{i,p,k} = \begin{cases} 1, & \text{if } V_{mag} \text{ violates constraints} \\ 0, & \text{others} \end{cases} \tag{4.4}$$

$$r_c^{p,k} = \begin{cases} 0.01, & \text{if } V_{min} \text{ violates CVR} \\ -0.01, & \text{others} \end{cases} \tag{4.5}$$

4.4.2 PT Policy Optimization based on DQN with Belief State

This section introduces the PT policy optimization. Specifically, the DQN with belief state algorithm will be applied to solve the POMDP, as presented in Algorithm 4.1.

This algorithm begins by initializing the action-value functions $Q(\cdot)$ and $Q'(\cdot)$ with parameters θ^Q and $\theta^{Q'}$, respectively, alongside creating the experience replay buffer $\mathcal{D}$. Throughout each episode, the agent captures PMU packets from PDCs to acquire local measurements regarding power generation and load consumption from visible nodes. Subsequently, the agent conducts spatial data prediction to estimate power generation and load consumption for every bus node based on the gathered information. The initial belief state b_0 is derived using the established power flow model, followed by the generation of the initial belief feature vector ϕ_0^b based on b_0.

Within each time step t, the agent selects an action a_t from the action space $\mathcal{A}$ according to the current belief feature vector ϕ_t^b, employing an ϵ-greedy strategy to explore the action space. The selected action a_t is executed on the ADN, and

Algorithm 4.1 PT policy optimization based on DQN with belief state

Initialize the $Q(\cdot)$ and $Q'(\cdot)$ with parameter θ^Q and $\theta^{Q'}$;
Initialize the experience replay buffer $\mathcal{D}$;
for *episode* $= 1$ to N_{ep} **do**
    Agent captures PMU packets from visible nodes and obtains local measurements for the power generation and power consumption;
    Agent performs spatial data prediction to estimate the power generation and consumption for every bus node;
    Agent gets the initial belief state b_0 by using the established power flow model;
    Agent gets the initial belief feature vector ϕ_0^b based on b_0;
    for $t = 1$ to $\mathcal{T}$ **do**
        Agent selects an action a_t from $\mathcal{A}$ based on the current belief feature vector ϕ_t^b following the ϵ-greedy strategy;
        Agent executes a_t on the AND and captures new PMU packets from visible nodes;
        Agent performs spatial data prediction to estimate the power generation and consumption for every bus node;
        Agent gets the next belief state b_{t+1} by using the established power flow model;
        Agent gets the next belief feature vector ϕ_{t+1}^b based on b_{t+1};
        Calculate the reward r_t based on Eq. 3.8 or Eq. 4.3;
        Store transition $(\phi_t^b, a_t, r_t, \phi_{t+1}^b)$ in D;
        Sample random minibatch of transitions from D, and update the θ^Q for $Q(\cdot)$ by minimizing the Eq. 3.10;
        Every C steps rest $\theta^{Q'} = \theta^Q$;
    end for
end for

new PMU packets are captured from visible nodes. Subsequently, the agent conducts spatial data prediction to estimate power metrics for all bus nodes, leading to the determination of the next belief state b_{t+1} through the power flow model. The corresponding next belief feature vector ϕ^b_{t+1} is generated based on b_{t+1}.

The algorithm calculates the reward r_t according to specified equations, either Eq. 3.8 or Eq. 4.3. The transitions $(\phi^b_t, a_t, r_t, \phi^b_{t+1})$ are stored in the experience replay buffer D, from which a random minibatch is sampled for updating the parameter θ^Q of $Q(\cdot)$ by minimizing the loss function defined in Eq. 3.10. Additionally, the parameter $\theta^{Q'}$ is updated to match θ^Q every C steps to ensure the stability of the learning process.

4.5 Experimental Validation

The *GridBattleSim* co-simulation platform introduced in Chapter 3 continues to be adopted to validate the proposed POMDP formulation for PT on smart grids. Illustrated in Figure 3.4, the Attacker federate communicates with a **Knowledge module** that models the power flow dynamics of the ADN based on the topology information obtained by the pen-tester in advance. The Knowledge module inputs are observable and predicted load consumption and power generation for visible and invisible nodes, and its output is the belief state of the entire power grid, forming the belief feature vectors for the Attacker federate.

In this work, the Knowledge module is implemented by another GridLAB-D simulator that runs power flow model of the ADN for belief state calculation. This Knowledge module is also co-simulated under the HELICS framework through publications and subscriptions.

4.5.1 Experiment Setup

AND Configurations

The standard IEEE 13-bus system is used to build the cyber-physical ADN as the PT environment. Based on this, we add residential houses, PV stations, and a voltage regulator into the system.

Figure 4.2 is the corresponding communication network of the target ADN, which is designed as a star-type topology with a router placed in each bus node. The central router is placed in Node-650 and connects to the control center. Endpoints for PMUs, smart inverters, and voltage regulators are located in certain nodes. We also add a Phasor data concentrator (PDC) in Node 611 as a PMU aggregator to aggregate PMU packets from Node 611, 692, 675, 652, and 680. These endpoints are connected wirelessly to their local access points.

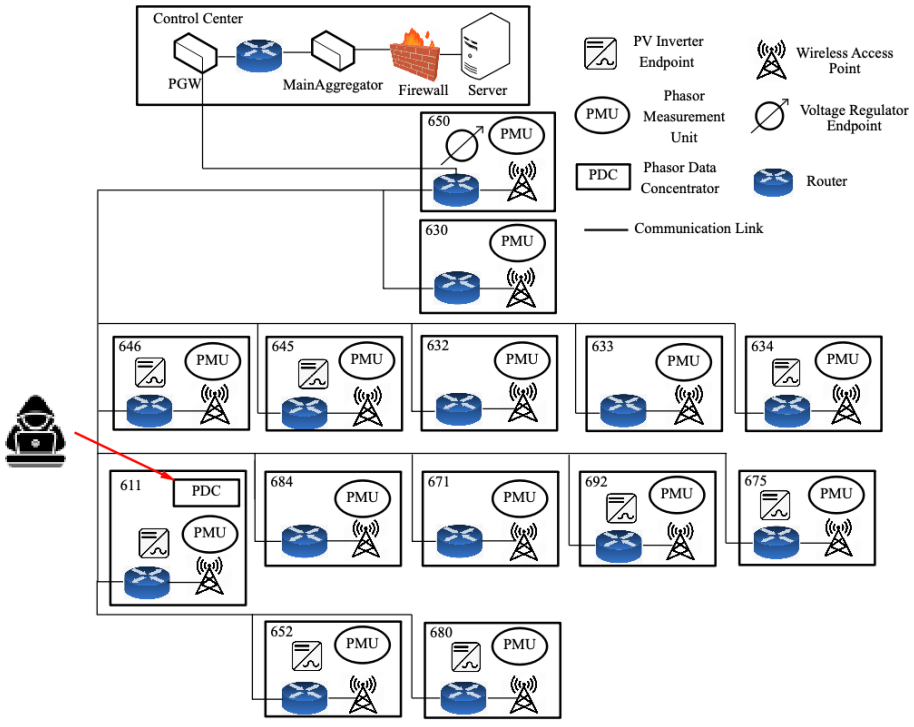


Figure 4.2: Communication networks for the target ADN.

The detailed system configuration, including device locations, default control parameters, network parameters, simulation time duration, etc., are listed in Table 4.1.

Table 4.1: Configurations of the target ADN

Parameter	Value
House location	646, 645, 634, 611, 692, 675, 652, 680
Number of houses in each phase	100
PV station location	646, 645, 634, 611, 692, 675, 652, 680
PV penetration ratio	100%
Smart inverter working mode	Unity-power-factor mode
Voltage regulator location	650
PDC location	611
PMUs aggregated by PDC	611, 692, 675, 652, 680
PMU data sampling interval	0.3s
Tap position ranges of voltage regulator	from -16 to +16
CVR control set-point	0.962 p.u.
CVR control band width	0.012 p.u.
Network bandwidth	50Mbp
Simulation duration	40s
Irradiation drop duration	4s to 10s

PT Configurations

The replay attack will record and replay outgoing packets from compromised endpoints to external networks, implemented as a replay attack hook inserted between the data link layer and the network layer of the endpoint in OmNet++ [126], as shown in Figure 3.7. In this work, the pen-tester attacks on the PDC at Node-611, which aggregates PMUs from Node-611, 692, 675, 652, and 680. That allows the pen-tester to record or replay packets from these aggregated PMUs simultaneously. The PT will

last for the 40s, with each action lasting 1s.

Hyperparameters of the DQN-based Agent

The hyperparameters for the DQN algorithm were determined through a series of experiments and evaluations. The training process consists of 3,000 episodes, during which the ϵ parameter that regulates the action selection strategy gradually decreases from 0.5 to 0.1 to encourage exploration at the beginning and exploitation at the end. The Q-network is a neural network that has two hidden layers with a neuron size of 30 in each layer. The learning rate for updating the Q-network is set at 0.001. The discount factor γ is fixed at 0.95.

4.5.2 Simulation Results

In this study, we conducted an in-depth investigation on the training performance of the PT by considering different reward functions and the use of belief states. Additionally, we also compared the performance of DQN with the Q-learning algorithm, which is a popular traditional RL algorithm that adopts the Q table to approximate the action-value function (Q function) instead of using a neural network.

PT with Belief State and Reward Function I

The training log of the DQN-based PT using the belief state under Reward Function I is depicted in Figure 4.3, where the episodic reward remains at zero (i.e., sparse reward issue) throughout the entire training process, indicating that the agent is unable to trigger any voltage violation during the PT and the agent is unable to improve its policy from its learning experiences. Thus, the agent repeatedly chooses the Stop action, as shown in Figure 4.4. The reason is that Reward Function I only rewards the agent upon successfully achieving the final goal (system voltage violation), which

typically takes a long time to achieve. As a result, the agent has a low probability of obtaining positive episodic rewards when the number of training episodes is limited, such as 3000 episodes in this case.

To address this issue, a new Reward Function II was proposed in Section 4.4.1, which can handle the sparse reward issues by rewarding the agent when subgoals of PT are achieved. The proposed Reward Function II will be validated in the following sections.

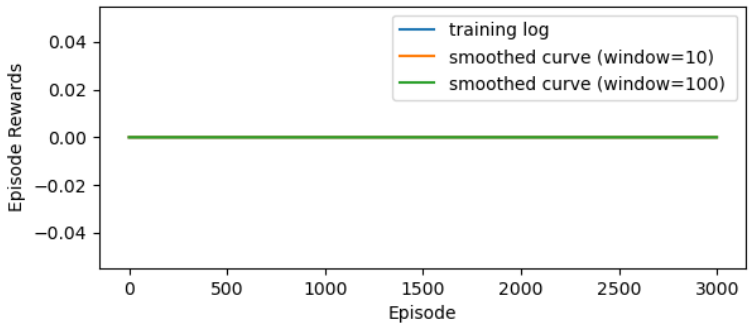


Figure 4.3: The training process for the DQN-based PT using Reward Function I and belief state.

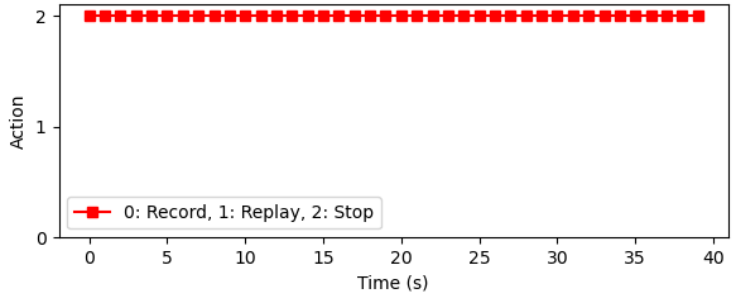


Figure 4.4: Attack path based on DQN using Reward Function I and belief state.

PT with Belief State and Reward Function II

Figures 4.5 and 4.6 illustrate the training logs of the PT utilizing the DQN and Q-learning algorithms, respectively. Driven by Reward Function II, the agent can

achieve non-zero episodic rewards in both DQN-based and Q-learning-based training processes. Notably, the performance of the DQN algorithm surpasses that of Q-learning in terms of learning efficiency within 3000 episodes. An obvious trend from the smoothed curves indicates that the episodic rewards of the DQN agent increase at a faster pace compared to the Q-learning agent after 2000 episodes. This trend underscores the superior learning capabilities of the DQN algorithm for solving POMDP in the context of PT on smart grids.

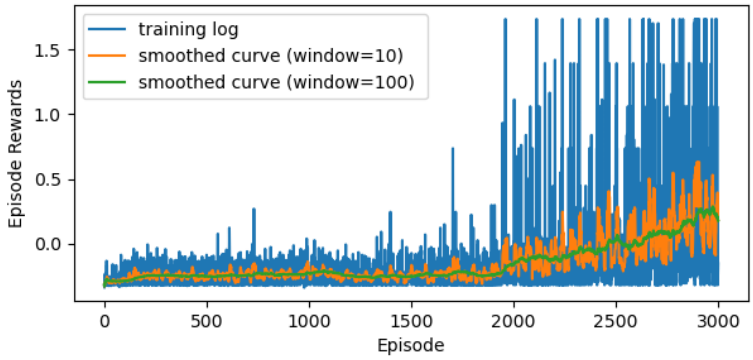


Figure 4.5: The training process for the DQN-based PT using Reward Function II and belief state.

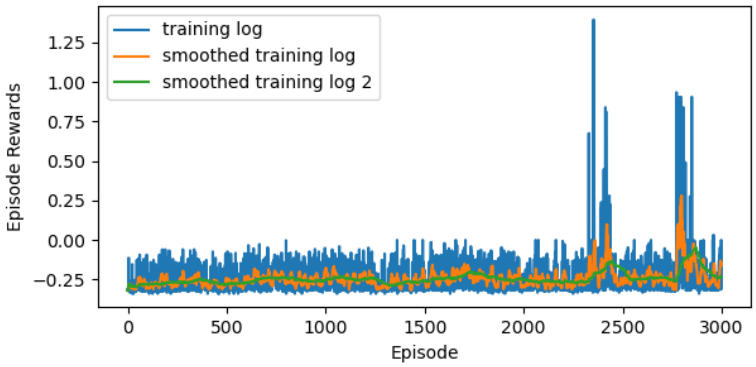


Figure 4.6: The training process for the Q-learning-based PT using Reward Function II and belief state.

Figure 4.7 shows the attack paths based on DQN using Reward Function II and belief state in different stages of the training process. Four policies (Policy A, B, C and D) are captured at 0th, 1,000th, 2,000th, and 3,000th episodes, respectively.

Policy A is the initial policy (non-trained policy) that launches the Stop action from the start to the end, which cannot have any impact on the system. After 1,000 epochs, Policy B launches 6 consecutive Record actions before 6s, then launches 13 consecutive Replay actions starting from 6. It is an important improvement over Policy A because Policy B is able to capture PMU packets when the grid's lowest voltage is lower than the CVR control band lower bound (as illustrated in Figure 3.9a that shows the non-attack condition). The recorded packets were later replayed many times and triggered the movement of the tap position towards the higher side. Then, after another 1,000 epochs, Policy C launches 7 Record actions before 7s and then launches 20 consecutive replay actions. Compared to Policy B, Policy C further improves the attack path by extending the duration of the Replay action, which can bring more negative impacts to the system. After the training, the final policy (Policy D) further extends the Replay action duration to the end of the episode, leading to the continuous increment of the tap position and a maximized negative impact.

Figure 4.8 displays the lowest three-phase voltage magnitude of the system and the corresponding predicted voltages (belief state) under Policy D. As seen in the figure, the pen-tester records the packets when the voltage is lower than the CVR lower bound due to the weather changes and replays them subsequently. Since the CVR control strives to keep the lowest voltage of the system within the CVR control band, the tap position of the voltage regulator will continuously increase when it receives the replayed PMU packets during the previous time. As a result, the actual voltages increase until they violate the system constraints.

PT Using Reward Function II without Belief State

Figure 4.9 and 4.10 are training logs for DQN-based PT and Q-learning-based PT without belief state, respectively. The DQN also outperforms the Q-learning for the

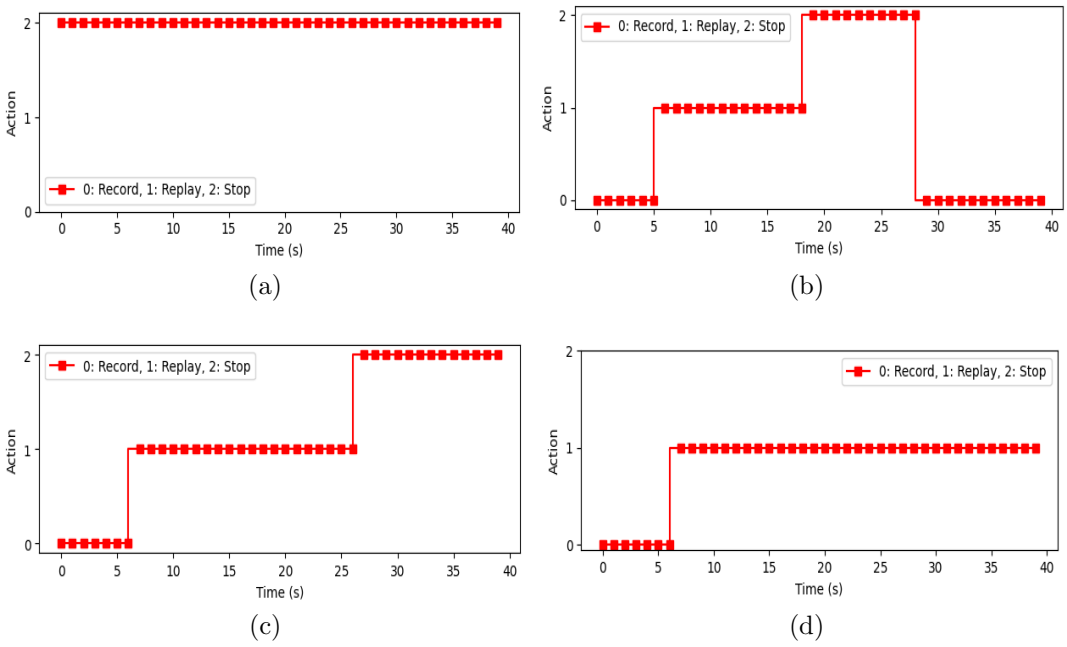


Figure 4.7: Attack paths based on DQN using Reward Function II and belief state in different stages of the training process: (a) Policy A (initial policy), (b) Policy B (after 1000 episodes), (c) Policy C (after 2000 episodes), (d) Policy D (final policy).

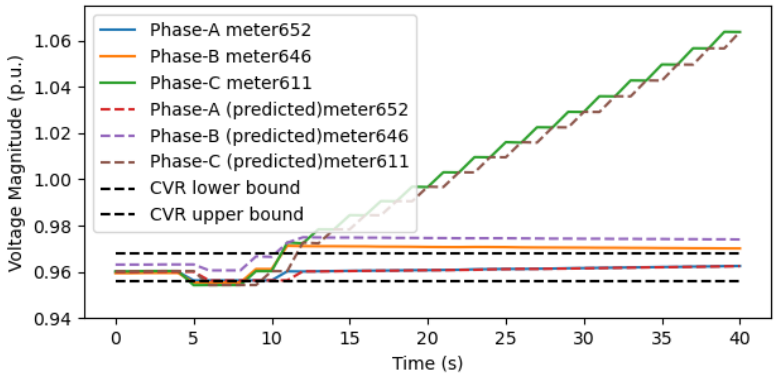


Figure 4.8: The lowest voltage of the system under the DQN-based PT with Reward Function II and belief state.

PT training in terms of learning efficiency. By the end of the training, the average episodic reward of DQN-based PT can exceed zero, while the Q-learning-based PT is still under zero.

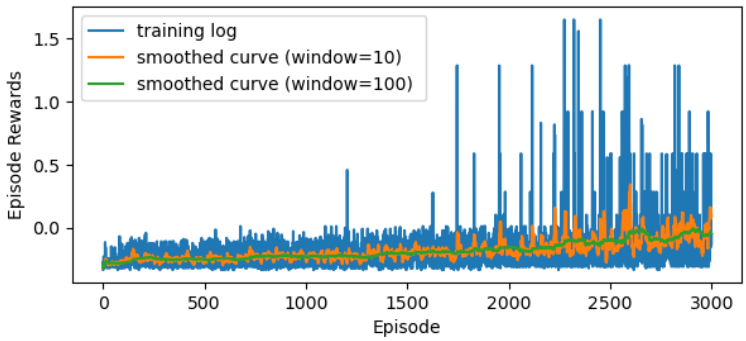


Figure 4.9: The training process for the DQN-based PT using Reward Function II without belief state.

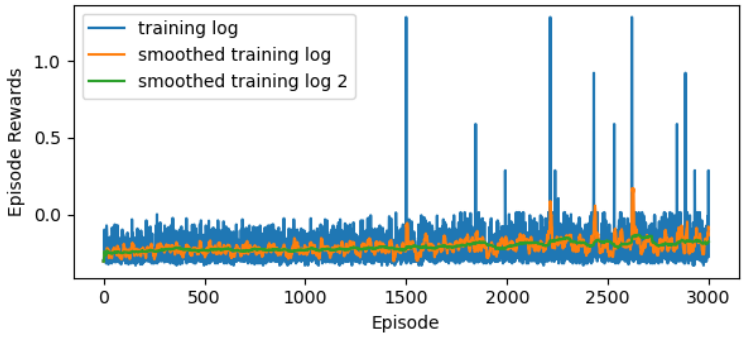


Figure 4.10: The training process for the Q-learning-based PT using Reward Function II without belief state.

However, without the global perception of the full state of the system, the learning efficiency is slowing down in this case, as shown in the comparison between Figure 4.5 and Figure 4.9, where the average episodic reward of DQN-based PT using the belief state can approach 0.5.

4.6 Conclusions

To overcome the limited observability of PT in practical scenarios, this chapter proposed to apply POMDP to formulate the PT process for ADNs, where the replay

attacks on PMU/PDC endpoints were investigated in a developed GridBattleSim. The agent established a physical model of the power grid based on its knowledge of the system to estimate its full state (belief state) by using the obtained local data. An improved reward function was proposed to decompose the ultimate goal into some sub-goals to improve learning efficiency. Then, the POMDP of the PT was transformed into an MDP problem, which was solved by the DQN algorithm to obtain the optimal timing and ordering of replay attacks. The experimental results showed that the belief state generated by the physical model could help the pen-tester to perceive the conditions or cyberattack impact of the invisible area and improve the learning efficiency of the PT. Compared with the reward function without considering the sub-goal, the proposed improved reward function can effectively address the sparse reward issue and boost learning efficiency.

Chapter 5

Knowledge-Informed AutoPT based on Reward Machine[1]

5.1 Problem Statement

The primary objective of PT is to identify critical vulnerabilities that could potentially be exploited, resulting in critical negative impacts on the system. This process typically involves an extensive search of action sequences from action spaces over a large state space driven by the agent's reward. A challenge arises in RL/DRL-based PT due to the sparse-reward issue. This issue emerges when the reward given to the agent solely reflects the achievement of the ultimate PT goal. Consequently, the agent receives a reward of zero value for the majority of the PT duration until the goal is achieved. Under these conditions, most agent-environment interactions yielding zero reward contribute minimally to the agent's learning progress toward the optimal solution. This lack of substantial contribution can potentially impede the process of finding the optimal solution.

[1]This chapter was published on the paper "Knowledge-Informed Auto-Penetration Testing Based on Reinforcement Learning with Reward Machine" [74], which was published by IEEE WCCI2024.

To address the sparse-reward issue of the PT, a prevalent strategy is to reshape the reward function by integrating human knowledge about the target system, which facilitates the decomposition of the PT goal into many subgoals or subtasks. By providing the agent with additional rewards upon achieving PT subgoals or subtasks rather than solely upon reaching the ultimate goal, the sparsity issue can be mitigated. This is because the subgoals or subtasks are usually more attainable compared to the ultimate objective. Based on this idea, our initial effort, detailed in Chapter 4, involves restructuring the reward function by decomposing the main goal into subgoals according to the assumption that the agent knows the control rule of the CVR performed in the control center, as expressed in Eq. 4.3.

However, the reshaped reward function Eq. 4.3 is specifically designed for ADNs with CVR control applications. This reward function introduces application-specific human knowledge about the target system, which loses generality since if the system changes, the reward function should be redesigned. Currently, there is no standard principle on the design of reward functions for PT in a variety of systems. Therefore, general task decomposition in terms of the reward function design based on human knowledge of PT, should be investigated.

Furthermore, encoding complex human knowledge into a single reward function could increase the complexity of the reward function itself. For example, the reward function could be a linear combination of rewards for a set of subgoals. Such a complex reward function will make it difficult for the agent to differentiate which aspects of contributions or losses result from its PT actions since the agent can only receive a single reward to guide its learning process. As a result, the agent may require a substantially larger number of interactions with the environment to comprehend this. Additionally, from a mathematical standpoint, the reshaped reward function cannot definitively assure the optimal PT policy, as it is empirically designed without taking

policy convergence into account.

In addition, interpretability is often lacking in RL/DRL-based PT. The trained PT policy cannot explicitly identify the current phase or situation of the PT agent and the subsequent direction it shall be heading. This type of perception or awareness could be encoded into the agent's neural networks representing the DRL policy through training, but it remains challenging to extract such information by decoding the neural networks.

Solution Overview: In response to the aforementioned challenges with respect to sampling efficiency, reward specification, and interpretability for RL/DRL-based PT, we propose to embed knowledge from the cybersecurity domain as guidelines into the agent's learning process in an explainable way. This approach can automatically break down the complex PT task into multiple subtasks, allowing the agent to learn more efficiently.

More specifically, we propose a knowledge-informed AutoPT framework called RM-PT, which utilizes a reward machine (RM) to encode domain knowledge based on cybersecurity knowledge bases, such as MITRE ATT&CK [72] and Cyber Kill Chain [137], etc. The RM specifies a set of events that occurred in PT and decomposes PT into multiple subtasks based on existing PT practices. Moreover, RM can specify different reward functions for PT in different phases, expanding the flexibility of traditional reward functions in RL/DRL-based PT. Under the RM-PT framework, PT is formulated as a POMDP guided by RMs. We focus specifically on the lateral movement of the cyber part of the smart grid as a case study, which assumes the pen-tester has gained initial access to the network of the control center and will move deeper into the network to own high-value assets. We investigate two different RMs as two guidelines. Finally, we employ the deep Q-learning algorithm with RM (DQRM) to solve the POMDP and optimize the PT policy.

5.2 Related Works

Human knowledge integration in RL/DRL-based PT and vulnerability analysis has already been investigated.

In [138], the authors explored the model-free RL-based PT specifically for the Capture-the-Flag (CTF) game. The research demonstrated through simulations that while RL can effectively solve simple CTF instances with basic structures, changes in the problem's structure can increase the PT complexity and require a significant number of samples for the RL agent to learn the PT policy. To address these challenges, the paper proposed to use techniques such as lazy loading, state aggregation, and imitation learning to equip the RL agent with essential prior knowledge about the problem structure, enabling it to handle more intricate problems efficiently. The authors concluded that RL agents that combine model-free algorithms with rich prior knowledge can be effective and useful in enhancing PT performance.

To overcome challenges involving vast state and action spaces and ineffective exploration in existing RL/DRL-based PT. Ref. [139] applied the hierarchical DRL (HDRL) model with experts' prior knowledge to improve the training performance of PT agents. The applied HDRL model, designed with two layers of agents and deep neural networks, aims to decompose PT tasks and reduce complexity, while expert prior knowledge provides rules and knowledge graphs to guide decision-making and reduce invalid exploration. Experimental results based on actual network environments demonstrated that the proposed method significantly improves sample efficiency, reduces learning time, and performs well in large-scale network scenarios.

Ref. [140] proposed the DQfD-AIPT PT framework, which combines RL with imitation learning to improve PT efficiency. The framework defines expert knowledge structure and utilizes the deep Q-learning from demonstrations (DQfD) algorithm

to enhance the training performance of the agent while avoiding overfitting. Experimental results in a simulated network scenario with honeypots demonstrated the effectiveness of incorporating expert knowledge, showing that the DQfD algorithm outperforms classical RL/DRL methods in terms of efficiency and cumulative reward, especially in reducing interactions with honeypots.

Similar to [140], Ref. [141] proposed a novel framework called Generative Adversarial Imitation Learning based intelligent PT (GAIL-PT), which utilizes expert knowledge bases and GAIL network to guide policy generation of RL/DRL agents with lower costs. Experimental results showed that GAIL-PT outperforms the state-of-the-art method DeepExploit in exploiting Metasploitable2 and Q-learning in different scale networks, demonstrating its effectiveness and suitability for enhancing RL/DRL-based PT methods.

5.3 Proposed Methodology

5.3.1 Knowledge-Informed AutoPT Framework

In this chapter, the investigated AutoPT method employs a computer program acting as an agent to launch a series of cyberattacks on network systems. The objective is to discover possible attack paths to take ownership of critical resources, similar to the capture-the-flag game. Thus, PT is a sequential decision-making problem, which can be formulated as a partially observable Markov decision process (POMDP) [2], and the optimal PT policy with respect to the POMDP can be obtained by applying RL/DRL algorithms.

To address the challenges of sampling efficiency, reward specification, and interpretability in RL/DRL-based PT, we propose a knowledge-informed AutoPT framework based on RL with reward machine (RM-PT), as depicted in Figure 5.1, which

uses a reward machine (RM) to encode domain knowledge of PT based on existing cybersecurity knowledge bases.

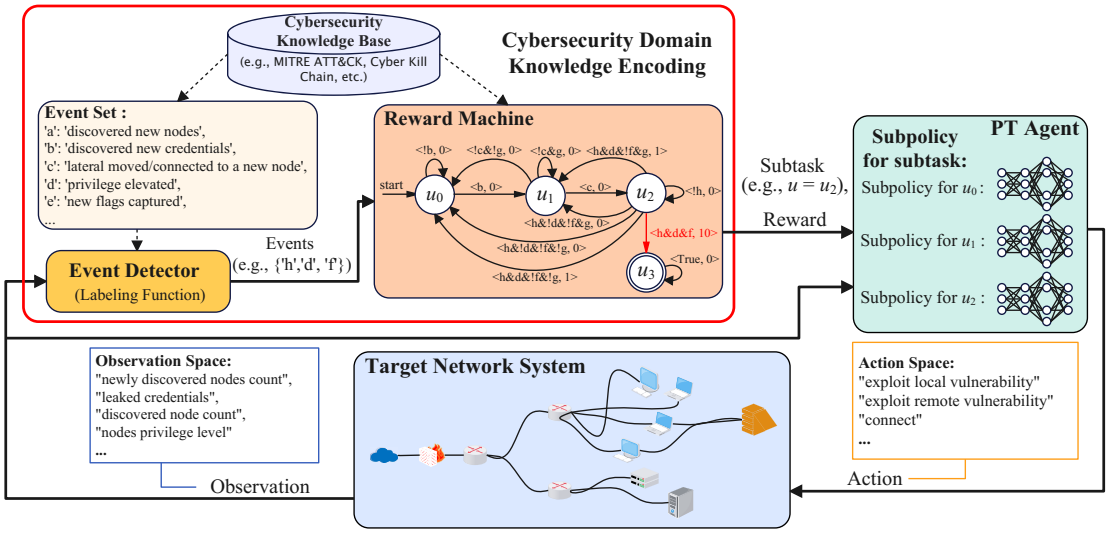


Figure 5.1: The proposed knowledge-informed AutoPT framework (RM-PT).

This framework involves an agent that acts as a pen-tester and interacts with the target network system that makes up the environment. This environment typically consists of hosts, firewalls, routers, and communication channels, among other components. The agent's action is selected from a range of PT activities (action space), such as network scanning, exploiting vulnerabilities, lateral movement, and privilege escalation. Additionally, the agent can gather observations from the environment through the information from scanning operations, defined by an observation space containing newly discovered vulnerabilities, leaked credentials, etc. The immediate reward reflects the evaluation of the agent's action. The aim of PT typically involves taking ownership of critical resources within the networks, such as customer data. Thus, the agent can assign a positive reward to itself when the action is beneficial for accomplishing this goal or a negative reward when it is not. The agent wants to find an optimal PT policy to maximize accumulated rewards through learning experiences

from agent-environment interactions [117]. To tackle the challenge of high dimensionality of the target system arising from the large observation or action spaces[118], the PT policy is represented by deep neural networks to determine the best action based on input observation. During the training phase, after each step, the agent will use the experience (observation, action, reward, next observation) to update the weights of the neural networks (iteratively improving the PT policy).

In this framework, the agent is informed and guided by a cybersecurity domain knowledge encoding module, which uses an RM to encode the domain knowledge of PT based on the cybersecurity knowledge bases, such as MITRE ATT&CK, Cyber Kill Chain, etc.

The RM is a state machine with two essential functions: 1) decomposing the PT task into a series of subtasks, such as "discover credentials" and "privilege escalation", and organizing them using a state machine structure; 2) specifying a reward function for each state transition within the RM.

The RM receives a set of events detected during PT as input, leading to a transition of its internal state from one state to another according to its transition rules (logic formula over the event set). The output of the RM includes the RM state and a reward function. The RM state indicates the completion of the previous subtask and the initiation of a new subtask; thus, it can be used to denote the subtask ID.

The event detector (also called the labeling function under the RM theory [142]) can identify occurring events, denoted by symbols, e.g., 'a' for "discovered new nodes" and 'b' for "discovered new credentials", from the environment by analyzing the latest agent-environment interaction. These events are designed to represent successful executions of adversary tactics as defined by cybersecurity knowledge bases. Because these tactics serve as the adversary's tactical goals, providing the motivation behind

their actions [72]. For instance, privilege escalation is one tactic for attacking enterprise networks defined by ATT&CK. Therefore, the corresponding event can be defined as its successful outcome, i.e., "privilege escalated". All events are predefined in an event set and can be detected using the event detector. Table 5.1 shows an example of an event set designed for PT based on ATT&CK tactics.

Table 5.1: Event set of PT ($\mathcal{P}$)

Event Symbol	**Event Description**
'a'	Discovered new nodes
'b'	Discovered new credentials
'c'	Lateral moved (connected to a new node)
'd'	Privilege elevated
'e'	New flags captured (new target nodes owned)
'f'	Achieved the PT goal
'g'	Has unused credentials
'h'	Taken action to elevate the privilege

As an example shown in Figure 5.1, when events 'h', 'd', and 'f' are detected, the RM will transit its state from u_2 to u_3 and output u_3 (subtask ID) and a constant reward function (reward = 10) according to its defined transition rule $<$ h&d&f, 10 $>$.

In addition, RM outputs a reward function instead of a reward value. Thus, RM allows the agent to specify task-specific reward functions to enhance the flexibility of the single reward function used in traditional RL algorithms.

Furthermore, RM-PT utilizes deep Q-learning with RM algorithm (DQRM) to train a PT policy, which decomposes the policy into a set of sub-policies for every subtask and can train all sub-policies simultaneously. Therefore, the final PT policy selects the sub-policy to determine the PT action.

5.3.2 POMDP with Reward Machine Formulation for AutoPT

The proposed PT under the RM-PT framework is formulated as a POMDP with RM (POMDPRM). A POMDPRM is characterized by a tuple consisting of seven components $\langle \mathcal{S}, \mathcal{A}, \mathcal{O}, T(s_{t+1}|s_t, a_t), O(o_t|s_t), \mathcal{R}, \gamma \rangle$ [143].

$\mathcal{S}$ represents the set of environment states, where $s_t \in \mathcal{S}$ denotes the state at time t. $\mathcal{O}$ represents the set of observations, and $o_t \in \mathcal{O}$ denotes the observation at time t. $\mathcal{A}$ represents the set of actions, and $a_t \in \mathcal{A}$ denotes the action taken by the agent at time t. $T(s_{t+1}|s_t, a_t)$ is the probability of transition from the current state s_t to the next state s_{t+1} when the agent performs an action a_t. $O(o_t|s_t)$ is the emission function that maps the current state to the observation received by the agent.

$\mathcal{R}$ is the RM, which is a tuple with six components $\mathcal{R} = \langle \mathcal{P}, L, U, u_0, \delta^u, \delta^r \rangle$. $\mathcal{P}$ is an event set of PT. L is the labeling function (event detector), $L : \mathcal{O} \times \mathcal{A} \times \mathcal{O} \to 2^{\mathcal{P}}$, which assigns truth values to events (True: event occurred, False: event not occurred), by analyzing the input experience (o_t, a_t, o_{t+1}). U is a finite set of RM states. $u_0 \in U$ is an initial state. δ^u is the state-transition function, $\delta^u : U \times 2^{|P|} \to U$, which determines the next RM state based on its current state and the captured events (i.e., $u_{t+1} \leftarrow \delta^u(u_t, L(o_t, a_t, o_{t+1}))$). δ^r is the reward-transition function, $\delta^r : U \times 2^{|P|} \to [\mathcal{O} \times \mathcal{A} \times \mathcal{O} \to \mathbb{R}]$, which outputs a reward function based on its current state and the captured events (i.e., $R(o_t, a_t, o_{t+1}) \leftarrow \delta^r(u_t, L(o_t, a_t, o_{t+1}))$). The agent can use the output reward function to obtain the reward.

$\gamma \in [0, 1)$ is the discount factor that determines the trade-off between immediate and long-term rewards that the agent prefers to achieve.

Due to the complex nature of the target network system, the determination of $T(s_{t+1}|s_t, a_t)$ and $O(o_t|s_t)$ poses challenges for PT. However, the agent can take the environment as a black box and learn the policy through pure trial and error.

For a specific scope of PT, the action space, observation space, and RM can be customized accordingly.

5.4 Case Study on Lateral Movement Using POMDPRM Formulation

In this work, the lateral movement on enterprise networks is considered as the study case of PT, which is under the assumption that the agent has already entered the target network (post-exploitation assumption). The POMDPRM formulation and two RMs are designed in the following subsections.

5.4.1 POMDPRM Design

Action Space

We consider three types of actions that the agent can execute during lateral movement. The first is scanning, which involves collecting network information by discovering new machines (nodes), determining the connections between these nodes, acquiring machine configuration data, and gathering vulnerability information for discovered nodes.

The second type of action is vulnerability exploitation, which can be classified into local vulnerability exploitation and remote vulnerability exploitation. Local vulnerability exploitation can only be performed on a connected node (the node where the agent is operating), and the agent seeks to steal local information, increase host privileges, or discover credentials for connecting to other nodes. Remote vulnerabilities come from nodes that are currently discovered but are not owned by the agent. By exploiting remote vulnerabilities, the agent can gather more information about the remote nodes.

The third type of action is connection, which enables the agent to connect a node using specific credentials and ports.

Due to the agent's lack of direct access to action outcomes, scanning is considered a mandatory action that must be performed after each action rather than being optional in action space. Additionally, the scanning operation can also contribute to forming an observation.

The action space is listed in Table 5.2, which includes three subspaces for local vulnerability exploitation, remote vulnerability exploitation, and connection, respectively. i, j, l, r, p, and c denote the ID of the source node, the target node, the local vulnerability, the remote vulnerability, the port, and the credentials, respectively. $\hat{n}$, $\hat{n}_l$, $\hat{n}_r$, $\hat{n}_p$, and $\hat{n}_c$ are agent's estimations of the maximum number of nodes, local vulnerabilities, remote vulnerabilities, ports, and credentials, respectively. Table 5.2 implies that for local vulnerability exploitation, the agent should choose the node ID and local vulnerability ID. For remote vulnerability exploitation, the agent should choose the source node ID, target node ID, and remote vulnerability ID. The source node ID, target node ID, port ID, and credential ID should be set for the connection. The PT action is a vector selected from one of the subspaces.

Table 5.2: Action space for lateral movement

Action	**Notation**	**Subspace Size**
Local vulnerability exploit	$[i, l]$	$\hat{n} \times \hat{n}_l$
Remote vulnerability exploit	$[i, j, r]$	$\hat{n} \times (\hat{n} - 1) \times \hat{n}_r$
Connection	$[i, j, p, c]$	$\hat{n} \times (\hat{n} - 1) \times \hat{n}_p \times \hat{n}_c$

We consider each action will last a constant unit period, after which it will be considered completed and terminated.

Observation Space

The observation of the agent in PT is obtained by scanning operation after each action taken using scanning tools, such as Nmap [144]. The observation space is designed and shown in Table 5.3, which consists of many subspaces, including the *discovered nodes count*, *nodes privilege level, discovered nodes properties*, *leaked credentials*, and *lateral move*.

Table 5.3: Observation space of PT

Observation	**Notation**	**Subspace Size**
Discovered nodes count	n_d	$\hat{n}$
Nodes privilege level	$[a_i]_{\hat{n}}$	$2^{\hat{n}}$
Discovered nodes properties	$[a_{i,p}]_{\hat{n}\times\hat{n}_{pr}}$	$3^{\hat{n}\times\hat{n}_{pr}}$
Leaked credentials	$[a_{i,p,c}]_{\hat{n}\times\hat{n}_p\times\hat{n}_c}$	$3^{\hat{n}\times\hat{n}_p\times\hat{n}_c}$
Lateral move	b_l	2

Among them, *nodes privilege level* is a vector describing the privilege level of every node, where two values can be assigned to each entry: 0 = "not owned", 1 = "Admin".

Discovered nodes properties tells what properties each node has. Properties include different types of operating systems (e.g., Windows, Linux), different types of databases (e.g., SQLServer, MySQL), etc. It is represented by a $\hat{n}\times\hat{n}_{pr}$ matrix, where $\hat{n}_{pr}$ is the estimated maximum number of properties. Each entry has three values: 0 = "No", 1 = "Yes", 2 = "Unknown".

Leaked credentials is a $\hat{n}\times\hat{n}_p\times\hat{n}_c$ tensor with the first dimension indicating the target node ID, the second dimension indicating the port ID, the third dimension indicating the credential ID. Each entry has three values: 0 = "Not discovered", 1 = "Used", 2 = "Unused".

Lateral move indicates whether the agent successfully moves from one node to another in a new interaction with two values: 0 = "No", 1 = "Yes".

By flattening and concatenating all vectors from these five subspaces, we can get an observation vector.

Reward Machine I ($\mathcal{R}_1$)

In this work, the PT agent is guided by the domain knowledge of PT encoded in an RM. In the field of PT, one useful guideline is that the pen-tester attempts to discover as many login credentials as possible to gain access and control over as many nodes as possible. Therefore, the PT can be divided into three subtasks: 1) discover new credentials, 2) gain access (connect) to a new node by using the discovered credentials, and 3) elevate the privilege of the connected node to own its properties. This process will be repeated to own more and more nodes until the PT goal is met, such as discovering critical data or owning a specific number of nodes.

According to this guideline, our first designed RM ($\mathcal{R}_1$) is shown in Figure 5.2, which has four states ($U_1 = \{u_0, u_1, u_2, u_3\}$) starting from u_0 and terminates at u_3. Its event set $\mathcal{P}_1 = \{$'b', 'c', 'd', 'f', 'g', 'h'$\}$ is a subset of the set defined in Table 5.1.

$\mathcal{R}_1$ tells that the agent will stay on u_0 until new credentials are found ('b'), then transition to u_1. The agent will remain in u_1 if no successful connection is made ('c') and the previously discovered credentials do not run out ('g'). If all credentials are used ('!g'), but still cannot connect to a new node ('!c'), it will go back to u_0 to find new credentials. If the agent can connect to a new node from u_1 ('c'), it will move to u_2. Then, the agent tries to take actions to elevate the privilege level of the connected node ('h'). If the privilege level is elevated ('d') and the final goal is reached ('f'), $\mathcal{R}_1$ ends in u_3. If the final goal is not reached ('!f'), the agent checks for unused credentials. If it is ('g'), the agent returns to u_1 to try other credentials. If not, the

agent returns to $u0$ to find new credentials.

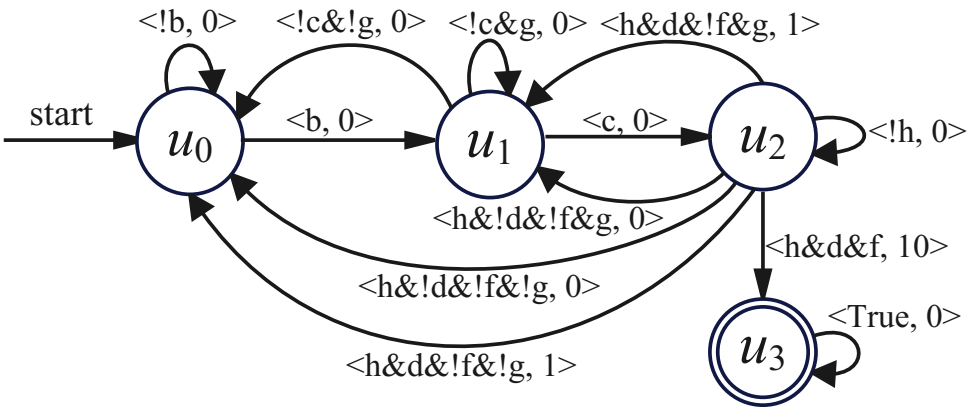


Figure 5.2: The diagram of Reward Machine I ($\mathcal{R}_1$).

Each transition in $\mathcal{R}_1$ also outputs a reward value. According to Figure 5.2, when the privilege level of a connected node is elevated ('d'), the agent gets a reward = 1. When the final goal is reached ('f'), the agent gets a reward = 10.

Reward Machine II ($\mathcal{R}_2$)

We also investigate a more detailed RM $\mathcal{R}_2$ where the knowledge guides the agent to discover new nodes first, then discover new credentials. Next, connect to a new node, and finally, elevate the privilege of the connected node. Therefore, its event set is $\mathcal{P}_2$ = {'a', 'b', 'c', 'd', 'f', 'g', 'h'}, where 'a' is added. The diagram of $\mathcal{R}_2$ is shown in Figure 5.3. $\mathcal{R}_2$ has five states since it has one more task compared to $\mathcal{R}_1$.

According to Figure 5.3, the agent will stay on u_0 until new nodes are discovered ('a'), then transition to u_1. Sometimes, the agent also finds credentials simultaneously ('a&b') because discovering a new credential usually comes with a new target node; thus, it can directly transition to u_2. The transitions in $\mathcal{R}_2$ from u_1 are similar to those in $\mathcal{R}_2$ from u_0. One difference is that after a successful connection ('c'), if all

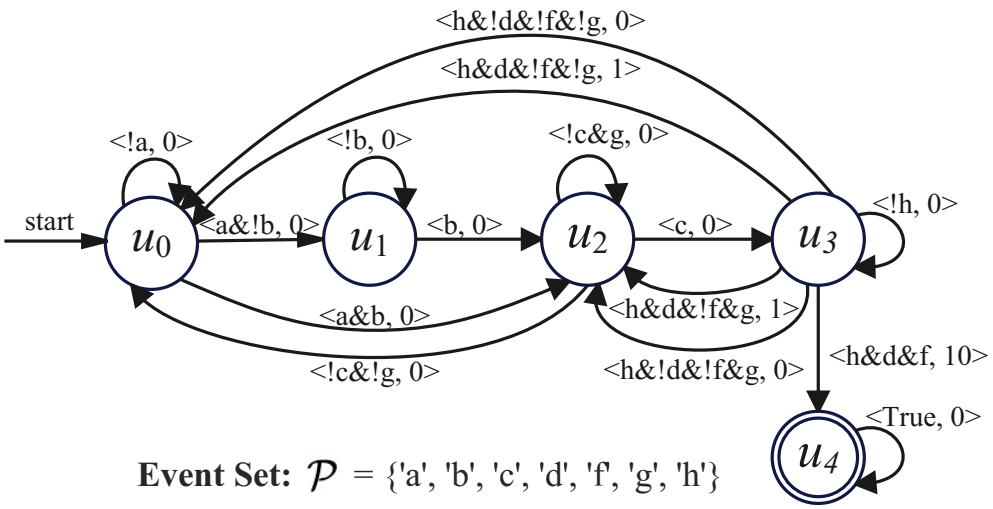


Figure 5.3: The diagram of Reward Machine II ($\mathcal{R}_2$).

discovered credentials are run out ('!g') but still cannot achieve the PT goal ('!f'), it will transition to u_0 to find new nodes instead of to find new credentials.

The reward transition function of $\mathcal{R}_2$ is similar to that of $\mathcal{R}_1$. When the connected node's privilege level is elevated ('d'), the agent receives a reward of 1. When the agent reaches the final goal ('f'), the reward is 10.

PT Objective

The goal of lateral movement is to take ownership (elevate privilege) of as many nodes as possible to own their properties. Therefore, discounted accumulated rewards with respect to $\mathcal{R}$ (denoted by $G_{\mathcal{R}}$) during PT can be used as the objective function for the agent to maximize, as expressed in Eq. 5.1.

$$G_{\mathcal{R}} = \sum_{t=1}^{\mathcal{T}} \gamma^{t-1} r_t, \tag{5.1}$$

where $\mathcal{T}$ is the maximum number of actions taken in the PT. r_t is the immediate reward obtained by $\mathcal{R}$. γ is the discount factor. By optimizing Eq. 5.1, we hope to

find out an optimal PT policy with respect to $\mathcal{R}$, denoted as $\pi^*_{\mathcal{R}} = \arg\max_\pi G_{\mathcal{R}}$.

5.4.2 PT Policy Optimization based on DQRM

The deep Q-learning with RM algorithm (DQRM) is used to train the agent and obtain the optimal PT policy $\pi^*_{\mathcal{R}}$. It is an updated version based on the Q-learning with RM (QRM).

Give an RM $\mathcal{R}$, QRM decomposes the training process by learning one Q-function ($Q(o, a)$, a function used to evaluate future rewards for a certain observation and action) per state in $\mathcal{R}$ [145]. These Q-functions can be considered as subpolicies and represented by Q-tables. We denote Q_u as the Q-function for state u of $\mathcal{R}$. Since the terminal state will bring no future rewards, the corresponding Q-function will always output 0. For example, in Figure 5.2, $\mathcal{R}_1$ has four states and will have four Q-functions.

The update rule for Q-tables in QRM is as follows:

$$Q_u(o, a) \xleftarrow{\alpha} r(o, a, o') + \gamma \max_{a'} Q_{u'}(o', a'), \tag{5.2}$$

where Q_u is the Q-table of state u; $Q_{u'}$ is the Q-table of the next state u'; α is the learning rate. According to Ref. [142], QRM can guarantee the convergence to the optimal policy with respect to $\mathcal{R}$.

As the complexity of the environment increases, the observation space of PT defined in Table 5.3 will grow exponentially, bringing a curse of dimensionality challenging to the QRM. Therefore, we adopt DQRM, which utilizes deep neural networks (called Q-networks and are parameterized by θ_u) instead of Q-tables to approximate Q-functions. It also adopts target Q-networks (parameterized by θ^-_u) and the experience replay mechanism to stabilize the training process.

The update rule for Q-networks is as follows:

$$Q_u(o, a; \theta_u) \xleftarrow{\alpha} r(o, a, o') + \gamma \max_{a'} Q^-_{u'}(o', a'; \theta^-_{u'}), \tag{5.3}$$

where Q_u is the Q-network of state u; $Q^-_{u'}$ is the target Q-network of the next state u'.

These Q-networks will be iteratively updated through the agent's interactions with the environment until the end of training. Finally, the optimal action a^*_t given o_t can be obtained by:

$$a^*_t = \arg\max_{a \in \mathcal{A}} Q_u(o_t, a; \theta_u). \tag{5.4}$$

Algorithm 5.1 PT policy optimization using DQRM

Require: $\mathcal{A}, \mathcal{O}, \mathcal{R} = \langle \mathcal{P}, L, U, u_0, \delta^u, \delta^r \rangle, \gamma, \alpha, C$

For every $u \in U - \{u_T\}$, initialize $Q_u(\cdot)$ and $Q^-_u(\cdot)$ with parameters θ_u and θ^-_u, respectively;
Initialize an experience replay buffer D;
for *episode* $= 1$ to N_{ep} **do**
    Agent takes scanning operation to get the initial observation o_0 from the target network system, $o_t \leftarrow o_0$;
    $u_t \leftarrow u_0$;
    for $t = 1$ to N_{it} **do**
        Agent selects an action a_t under o_t from $\mathcal{A}$ based on ϵ-greedy strategy;
        Agent executes a_t and takes scanning operation to get the observation o_{t+1} from the target network system;
        $u_{t+1} \leftarrow \delta^u(u_t, L(o_t, a_t, o_{t+1}))$;
        $R(\cdot) \leftarrow \delta^r(u_t, L(o_t, a_t, o_{t+1}))$;
        $r_{t+1} = R(o_t, a_t, o_{t+1})$;
        Save the experience $(o_t, a_t, o_{t+1}, u_t, u_{t+1}, r_{t+1})$ into D;
        Sample random mini-batch of experiences (o, a, o', u, u', r) from D, and update θ_u for $Q_u(\cdot)$ based on Eq. 5.3;
        if t mode $C = 0$? **then**
            For every $u \in U - \{u_T\}$, $\theta^-_u \leftarrow \theta_u$;
        end if
        $o_t \leftarrow o_{t+1}$
        $u_t \leftarrow u_{t+1}$
    end for
end for

The pseudo-code of the PT optimization using DQRM is shown in Algorithm 5.1. The algorithm requires the specification for the action space $\mathcal{A}$, observation space $\mathcal{O}$, RM $\mathcal{R} = \langle \mathcal{P}, L, U, u_0, \delta^u, \delta^r \rangle$, and the configurations of three hyperparameters, i.e., γ, α, and C. Subsequently, for each RM state u within the set U excluding the terminal state u_T, the algorithm initializes $Q_u(\cdot)$ and $Q_u^-(\cdot)$ parameterized by θ_u and θ_u^-, respectively. The algorithm also initializes an experience replay buffer D to store the agent's experiences during the training process. At the beginning of each episode, the agent performs a scanning operation to obtain the initial observation o_0 from the target network system, setting o_t to o_0, and initializes the current RM state u_t to the initial tactic u_0. The agent then interacts with the environment iteratively to update its policy, where the agent will select an action based on the ϵ-greedy strategy under the observation o_t and execute the action to receive the next observation o_{t+1}. After that, the agent gets its next RM state u_{t+1} based on RM's state-transition function δ^u given the previous RM state u_t and the detected events from the labeling function $L(o_t, a_t, o_{t+1})$. The RM also outputs the reward function $R(\cdot)$ based on the reward-transition function δ^r and calculates the reward r_{t+1}. The experience $(o_t, a_t, o_{t+1}, u_t, u_{t+1}, r_{t+1})$ will be stored in the replay buffer D. The agent will sample a random mini-batch of experiences from D, where each sample (o, a, o', u, u', r) is used to update the Q-network parameters θ_u using the update rule specified in Eq. 5.3. Furthermore, as the same trick used in the DQN algorithm, the DQRM algorithm synchronizes the target Q-network parameters θ_u^- with the main Q-network parameters θ_u for every C steps to improve the stability of training.

5.5 Simulation Platform and Testing Environments

In this study, we use *CyberBattleSim* as our experimental platform [97]. This platform, developed by Microsoft, is an open-source network simulator designed for lateral

movement research. It enables users to examine PT strategies they establish on simulated enterprise networks. Simulation networks are modeled by an abstract graph with nodes (machines) parameterized by a set of pre-planted vulnerabilities. In addition, the OpenAI Gym interface allows automated agents to be trained by utilizing RL/DRL algorithms.

In *CyberBattleSim*, two typical networks are set as optional environments to test PT strategies, called *CyberBattleChain* (denoted as *env*-1) and *CyberBattleToyCtf* (denoted as *env*-2), as shown in Figure 5.4 and 5.5, respectively. Each node in both environments is a host that possesses a set of properties and is pre-planted with a set of local and remote vulnerabilities (vulnerabilities are listed on each node in the figure). These vulnerabilities could reveal the credentials of adjacent nodes or lead to the privilege escalation of target nodes. Moreover, both environments are pre-configured with 'flag' nodes, which contain important or sensitive resources, such as customer data. The agent's goal is to capture as many flags as possible during the PT while minimizing the number of actions.

env-1, as shown in Fig. 5.4, has a sequential network topology constructed by a number of "Linux-Windows" links (i.e., a Windows host connected with a Linux host). The first node is a Windows host and is assumed to be infected by the pen-tester. The flag is located at the terminal node, which means that to capture the flag, the agent needs to elevate the privileges of all nodes one by one. We use N to denote the number of "Linux-Windows" links. In this case, we set N to 8.

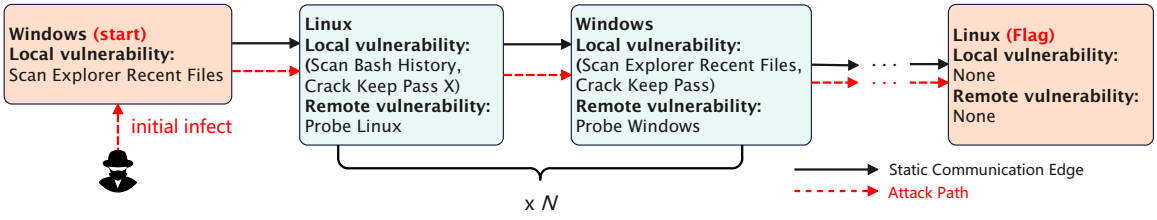


Figure 5.4: *CyberBattleChain* environment (*env*-1).

Compared to *env*-1, *env*-2 is more complex. *env*-2 is made up of a mesh network architecture, as shown in Fig. 5.5. *env*-2 contains four flags, of which two are unattainable since they did not leak credentials and the attacker cannot gain higher privileges, but the attacker can discover other nodes from it by performing remote attacks, such as navigating the web directory. The agent's goal is to capture two attainable flags.

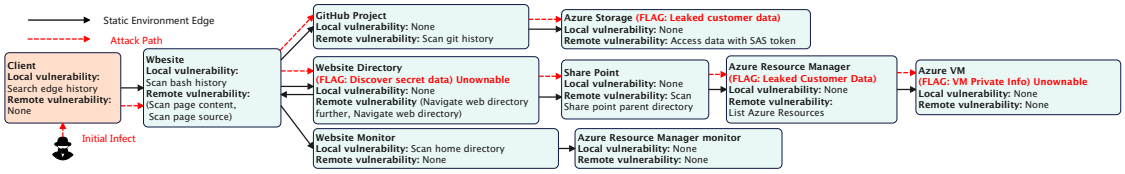


Figure 5.5: *CyberBattleToyCtf* environment (*env*-2).

5.6 Experimental Validation

We design the experiments to validate our proposed method and attempt to answer two research questions below:

- **RQ1**: Can the agent guided by RM improve the learning efficiency of PT compared to the agent without RM?
- **RQ2**: How will different RM designs affect the PT performance?

5.6.1 Agent Configurations

Table 5.4 lists four different agents, where DQRM-RM1 and DQRM-RM2 are agents that use the DQRM algorithm guided by $\mathcal{R}_1$ (shown in Figure 5.2) and $\mathcal{R}_2$ (shown in Figure 5.3), respectively. DQN-RM1 and DQN-RM2 are agents that use the DQN algorithm where only one policy will be trained, but the reward is determined by $\mathcal{R}_1$ and $\mathcal{R}_2$, respectively.

Table 5.4: Learning agents for comparison

Agent Name	Description
DQRM-RM1	DQRM guided and rewarded by $\mathcal{R}_1$
DQN-RM1	DQN rewarded by $\mathcal{R}_1$
DQRM-RM2	DQRM guided and rewarded by $\mathcal{R}_2$
DQN-RM2	DQN rewarded by $\mathcal{R}_2$

The agents' training parameters are empirically determined, as listed in Table 5.5. We run 100 episodes to train each agent and 50 episodes to evaluate their trained policies. Each episode is an execution of one PT, which is terminated upon the achievement of the predefined PT goal (capture all flags) or the maximum number of actions performed. The Q-functions are fully connected neural networks with two hidden layers, and each hidden layer has 150 neurons. After each step, the agents will randomly sample 100 experiences (batch size) to update their policies. For every 10 steps, the target neural networks will copy the weights from the training neural networks. The learning rate of neural networks is set to 0.001. γ is set to 0.9. ϵ is set to 0.3.

These four agents in Table 5.4 are comparable in the same environment since agents always receive the same reward given the same agent-environment interaction, which also means that their objective functions are the same. The performance indicators of an agent include training efficiency and PT efficiency, which will be compared in the training phase and in the evaluation phase, respectively.

Training efficiency indicates how quickly an agent learns to achieve a high accumulated reward within a certain number of training episodes, which can be visualized by two indicators: (1) the accumulated rewards with respect to the accumulated steps (actions), denoted by ACR, and (2) the average rewards per step, denoted by ARP. ACR is visualized by collecting the accumulated rewards for every time step from

1 to the maximum ($N_{it} = 1500$) among all training episodes. Specifically, for every time step t, we collect the accumulated rewards ending at t from all episodes and visualize them into a figure. ARP is obtained by calculating the ratio between the total rewards and the total number of steps used for every training episode.

PT efficiency indicates how quickly an agent achieves its goal during PT, which can also be observed using ACR and ARP, but can be directly measured by the number of steps per episode, denoted by TS.

Table 5.5: Training parameters of DQRM and DQN

Parameter	**Description**	**Value**
N_{it}	Maximum number of actions per episode	1500
N_{ep}	Number of episodes (train)	100
N_{evl}	Number of episodes (evaluation)	50
γ	Discount factor	0.9
α	Learning rate	0.001
C	Target networks update frequency	10
N_{bch}	Batch size	100
N_{hl}	Number of hidden layers	2
N_{hs}	Hidden size	150
ϵ	ϵ-greedy parameter	0.3

5.6.2 Comparative Studies

We train these four agents using the configurations listed in Table 5.4 in *env*-1 and *env*-2. The experimental results are shown in figures from 5.6 to 5.12.

Figures 5.6 and 5.7 illustrate the comparison of training efficiency between four agents in *env*-1. Figure 5.6 shows the comparisons of ACR. Figure 5.7 shows the comparisons of ARP. Figures 5.8 and 5.9 illustrate the comparison of PT efficiency between four agents in *env*-1. Figure 5.8 shows the comparisons of ACR. Figure 5.9

shows the comparisons of TS.

Similarly, Figures 5.10 and 5.11 illustrate the comparison of training efficiency between four agents in *env*-2. Figure 5.10 shows the comparisons of ACR. Figure 5.11 shows the comparisons of ARP. Figure 5.12 shows the comparisons of TS, which illustrates the comparison of PT efficiency between four agents in *env*-2.

Comparison Between DQRM Agents and DQN Agents in *env*-1

To answer **RQ1**, we compare the training efficiency between DQRM agents and the DQN agents guided by $\mathcal{R}_1$ and $\mathcal{R}_2$, respectively, in different environments.

By looking at the average value of ACR (smoothed curves) from Figure 5.6, we can see that both DQRM-RM1 and DQRM-RM2 outperform DQN-RM1 and DQN-RM2 in terms of the training efficiency in *env*-1, respectively. This is because by using the same number of actions, DQRM agents can obtain higher average accumulated rewards than DQN agents. As a result, the curve that exhibits superior performance will tend to approach the top left corner more closely. For instance, by 200 steps, the average accumulated rewards of DQRM-RM1 and DQRM-RM2 are about 15 and 16, respectively, whereas the average accumulated rewards of DQN-RM1 and DQN-RM2 are about 10 and 13, respectively.

The ARP of four agents is depicted in Figure 5.7 using box plots that display the data distribution of ARP for all training episodes, showcasing key statistical measures such as median, first quartile, third quartile, minimum, maximum values, and outliers. We can also see that the ARP median of DQRM-RM1 (0.154) is greater than DQN-RM1 (0.078), the ARP median of DQRM-RM2 (0.60) is greater than DQN-RM2 (0.26). Therefore, both DQRM-RM1 and DQRM-RM2 outperform DQN-RM1 and DQN-RM2 in terms of the training efficiency in *env*-1, respectively.

We also compare the PT efficiency using trained policies between four agents. In

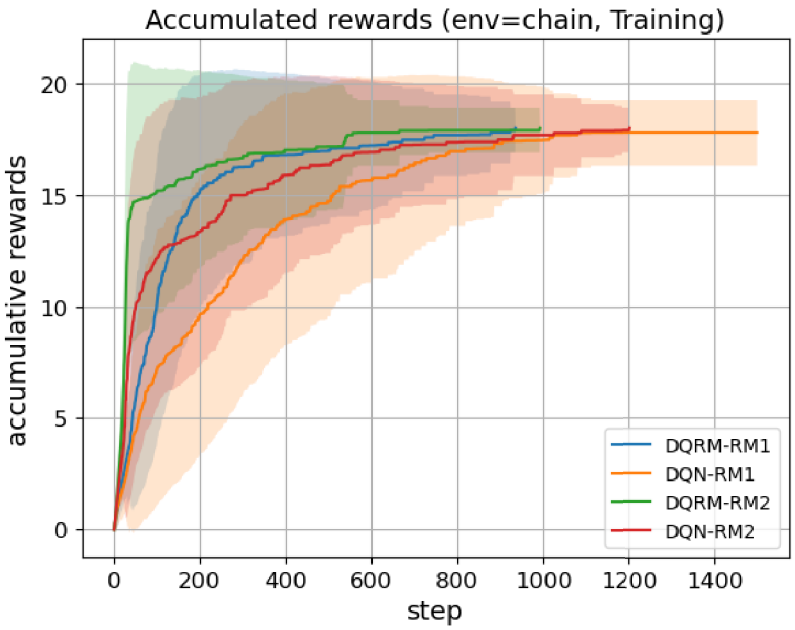


Figure 5.6: The training performance of four agents in *env*-1: accumulated rewards with respect to the accumulated steps (ACR).

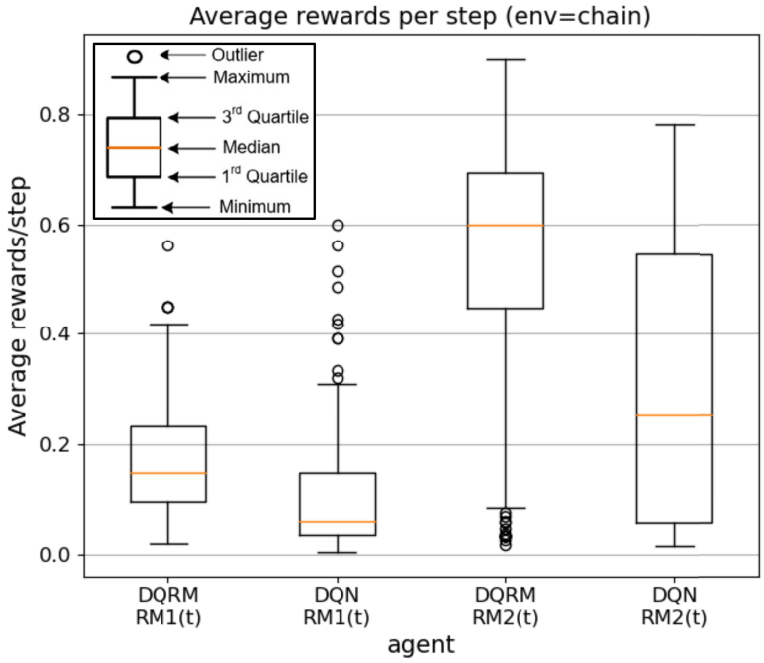


Figure 5.7: The training performance of four agents in *env*-1: average rewards per step (ARP).

Figure 5.8, the x-axis is the logarithmic scale of the step. The trained PT policies of DQRM-RM1, DQRM-RM2, and DQN-RM1 in *env*-1 can achieve higher accumulative rewards in around 50 steps compared to DQN-RM2. DQRM-RM1 and DQRM-RM2 exhibit similar levels of performance, both of which outperform DQN-RM1.

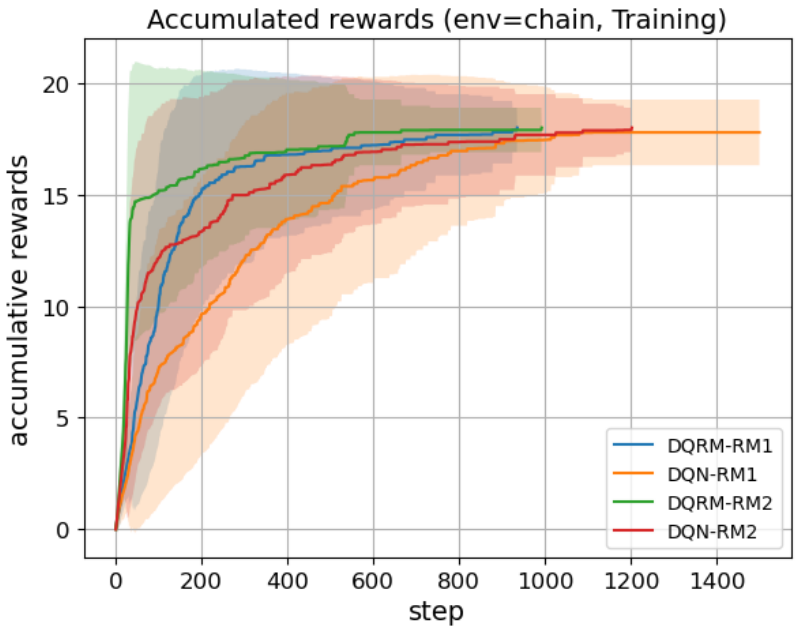


Figure 5.8: PT performance of four agents in *env*-1: accumulated rewards with respect to the accumulated steps (ACR).

Figure 5.9 uses box plots to display the data distribution of TS for all evaluation episodes, where we can observe that DQRM-RM1, DQRM-RM2, and DQN-RM1 can capture the flag using a limited number of steps, which means that their policies are well trained. However, the number of steps in DQN-RM2 has a great variance. Finally, the average number of steps for DQRM-RM1, DQRM-RM2, and DQN-RM1 is 23.48, 21.32, and 29.76, respectively. The average number of steps for DQN-RM2 is 767.46, which reflects poor PT performance.

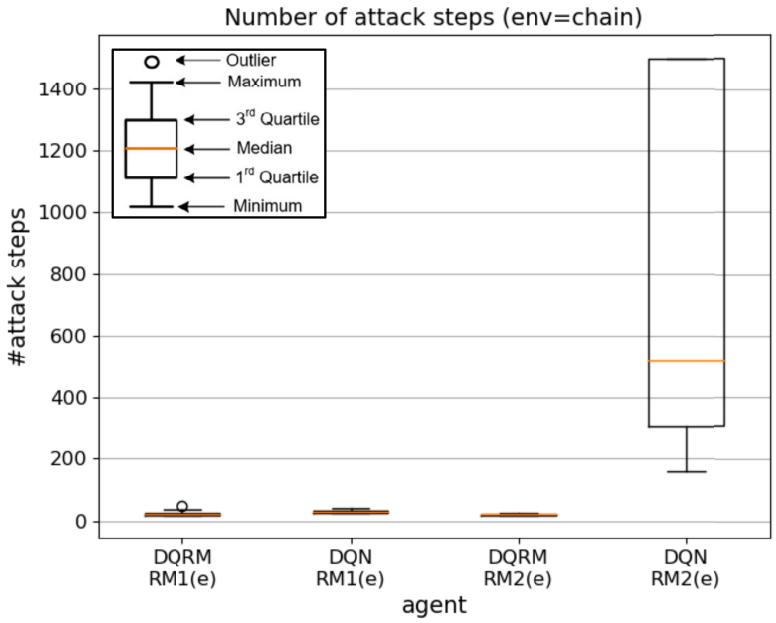


Figure 5.9: PT performance of four agents in *env*-1: number of steps per episode (TS).

Comparison Between DQRM Agents and DQN Agents in *env*-2

In *env*-2, DQRM agents also show improved learning efficiency according to Figures 5.10 and 5.11. By looking at the average value of ACR (smoothed curves) from Figure 5.10, we can see that both DQRM-RM1 and DQRM-RM2 can learn faster than DQN-RM1 and DQN-RM2 in *env*-2, respectively. For instance, by 200 steps, the average accumulated rewards of DQRM-RM1 and DQRM-RM2 are about 11 and 7, respectively, whereas the average accumulated rewards of DQN-RM1 and DQN-RM2 are all about 6.

Figure 5.11 shows box plots of ARP for four agents. We can also see that the ARP median of DQRM-RM1 (0.06) is greater than DQN-RM1 (0.05), the ARP median of DQRM-RM2 (0.077) is greater than DQN-RM2 (0.05). Therefore, both DQRM-RM1 and DQRM-RM2 outperform DQN-RM1 and DQN-RM2 in terms of the training efficiency in *env*-2, respectively.

We also compare the PT efficiency between four agents using trained policies

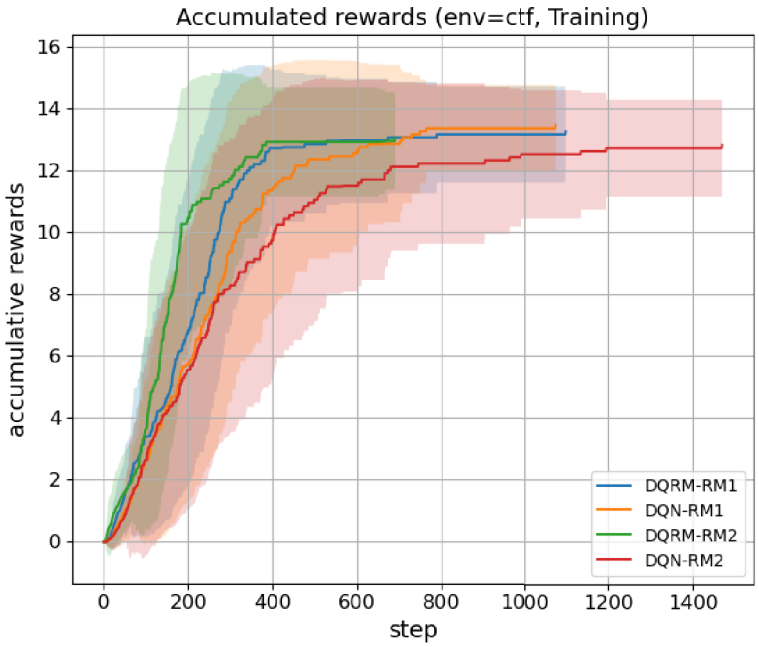


Figure 5.10: The training performance of four agents in *env*-2: accumulated rewards with respect to the accumulated steps (ACR).

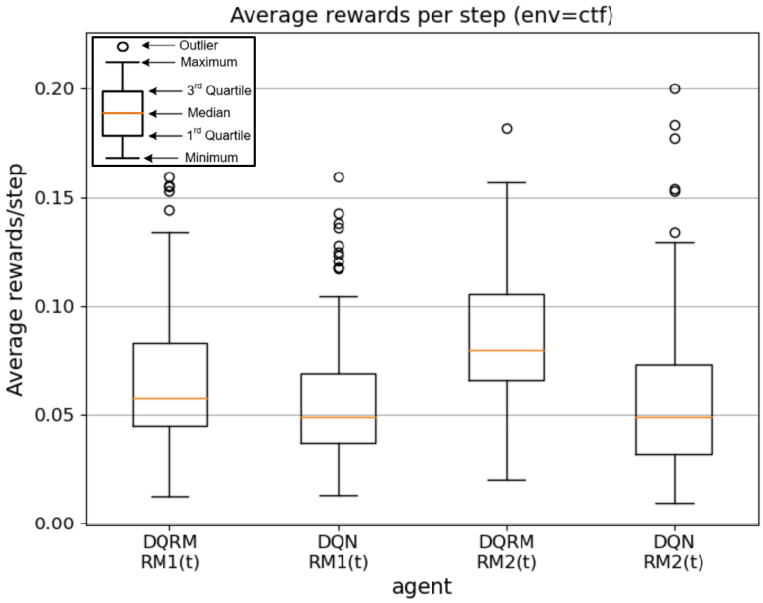


Figure 5.11: The training performance of four agents in *env*-2: average rewards per step (ARP).

in *env*-2. Figure 5.12 uses box plots to display the data distribution of TS for all evaluation episodes, where we can observe that both DQRM-RM1 and DQRM-RM2 have fewer attack steps than DQN-RM1 and DQN-RM2, respectively. The median values of TS are 202, 320, 163, and 311, respectively, for four agents.

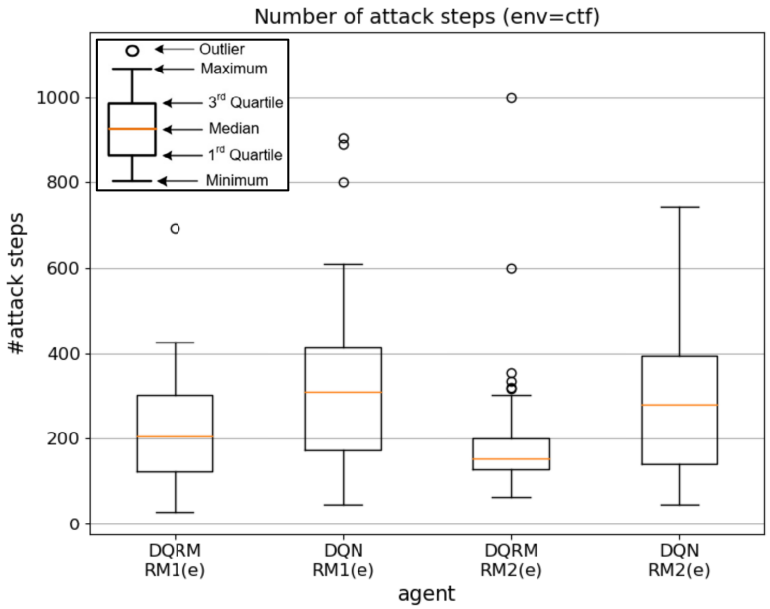


Figure 5.12: PT performance of four agents in *env*-2: number of steps per episode (TS).

From the previous analysis, we can answer **RQ1** that RMs can help the agent learn PT policies faster in different environments.

Comparison Between Different RMs

To answer **RQ2**, we compare the PT performance between DQRM-RM1 and DQRM-RM2 in *env*-1 and *env*-2, respectively.

From Figure 5.6, DQRM-RM2 demonstrates better training efficiency compared to DQRM-RM1 in *env*-1. We can see that after approximately 80 steps, DQRM-RM2 achieves an average accumulated reward of approximately 15, whereas DQRM-RM1 achieves around 10. The superior training efficiency of RM2 can also be observed from

Figure 5.7 where the ARP median of DQRM-RM2 (0.6) is greater than DQRM-RM1 (0.154).

From Figure 5.9, the DQRM-RM2 agent requires fewer steps to capture the flag in most episodes compared to the DQRM-RM1 agent in *env*-1. The median TS of DQRM-RM1 and DQRM-RM2 are 23 and 21, respectively, indicating that DQRM-RM2's PT efficiency is better than DQRM-RM1 in *env*-1. From Figure 5.12, DQRM-RM2 also shows lower median TS (163) compared to DQRM-RM1 (202) in *env*-2.

Based on the previous analysis, we can conclude that the PT performance of agents guided by $\mathcal{R}_2$ is more effective than $\mathcal{R}_1$ in different environments since it involves an additional subtask, i.e., discovering new nodes, which is more detailed than $\mathcal{R}_1$.

5.7 Conclusions

To handle the sparse reward issue, reduce the difficulty of reward function specification, and improve the interpretability of RL/DRL-bade PT, we proposed a knowledge-informed AutoPT framework called RM-PT in this chapter. This framework utilizes RMs to embed domain knowledge from the field of cybersecurity, which serves as guidelines for training PT policies. We took lateral movement in ADNs as a case study of PT, and we formulated it as a POMDP guided by RMs. We also designed two RMs based on the MITRE ATT&CK knowledge base. To train the agent and derive the PT policy, we adopted the DQRM algorithm. The effectiveness of our solution was evaluated using the *CyberBattleSim* platform, from which the experimental results demonstrate that the DQRM agent exhibits a higher training efficiency in PT compared to agents without knowledge embedding. Furthermore, RMs that incorporate more detailed domain knowledge exhibit superior PT performance compared to RMs with simpler knowledge.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

This work proposed a set of DRL-based PT techniques to identify critical vulnerabilities for cyber-physical ADNs.

Chapter 1 provided an overview of cyber-physical smart grids, covering cybersecurity issues, defense strategies, PT concepts, and associated challenges. It outlined the problem statements of PT for smart grids, including efficiency issues, non-comprehensiveness, limited observability, and sparse-reward issues of RL/DRL-based PT, setting the foundation and motivation for the study. Chapter 2 provided a literature review and identified research gaps in existing literature, including vulnerability analysis against cyberattacks for smart grids and AutoPT techniques used in the ICT industry.

To handle the poor efficiency and non-comprehensiveness issues of command PT for cyber-physical ANDs, Chapter 3 first proposed a DRL-based PT framework for ANDs. The framework formulates PT as an MDP considering cyber-physical coupling and realistic cyberattacks demonstrated through a replay attack model on an ADN with CVR control. By applying the DQN algorithm, critical replay attack paths

leading to system voltage violations were identified, showcasing the framework's effectiveness in addressing PT challenges under varying system conditions. The simulation results also showed that DRL-based PT can learn to find the optimal attack path against system stability when the grid is under high load demand, solar power generation, and weather variation.

To facilitate the study, a co-simulation platform named *GridBattleSim* was developed specifically for DRL-based PT on ADNs. This platform integrates and co-simulates various simulators and programs to model different parts of the ADN.

To handle the limited observability of PT, Chapter 4 introduced a POMDP formulation for PT on ADNs and applied a physical model to estimate the power grid's full state (belief state) from locally obtained data, which transfers the POMDP to an MDP solvable by DRL. In addition, the proposed reshaped reward function considering the goal decomposition can mitigate the sparse reward issue of DRL and accelerate learning progress compared to the baseline reward function used in the MDP formulation in Chapter 3.

To address the sparse reward issue, difficult reward function specification, and poor interoperability of RL/DRL-based PT, Chapter 5 introduced the RM-PT framework. This framework applies RMs to embed domain knowledge of cybersecurity into RL/DRL, which decomposes PT into subtasks and allows learning separate policies for each. The efficacy of the framework was demonstrated through case studies on lateral movement, where the MITRE ATT&CK framework serves as the human knowledge encoded by two different RMs. The DQRM algorithm was employed to train the PT agent and derive optimal PT policies. Experiments performed on the *CyberBattleSim* platform showed that the DQRM agent has better training efficiency in PT compared to agents without knowledge embedding, and the RM containing more detailed domain knowledge can perform better in PT than those with simpler

knowledge.

6.2 Future Work

In the future, we will explore further in the experimental sections, including a sensitivity analysis on the weights used in the updated reward function to trade-off between the optimal goal and the subgoals. Additionally, we will create more diverse operational scenarios of ANDs to examine the training performance of PT using different DRL algorithms.

Furthermore, we will continue to investigate the human knowledge integration for AI-powered PT on cyber-physical smart grids with the purpose of improving the environment adaptation ability, where the trained PT agents can handle PT tasks in various environments with minimal effort. We plan to extend our research from three aspects:

- **The forms of knowledge embedding**: The human knowledge will be embedded not only from the perspective of reward function design for RL/DRL-based PT, but also from the perspective of action space. For example, the action space of PT can be dynamically constrained during the PT based on the human knowledge and the current observation captured by the agent. This approach allows the agent to avoid selecting actions from a potentially large action space at all times, thereby enhancing learning efficiency and the robustness of the trained policy. More specifically, one direction for us is to improve the RM presented in Chapter 5 by constraining the action space for each subtask (tactic).

- **The quality of the embedded knowledge**: Since the quality of knowledge significantly influences the performance of RL/DRL-based PT, we will delve into enhancing the quality of the embedded knowledge by adopting reliable

knowledge bases and exploring advanced knowledge embedding techniques. For example, the application of the knowledge graph to encode knowledge from ATT&CK knowledge base.

- **LLM + Domain knowledge**: With the wide application of large language model (LLM) [146], we will also conduct research on LLM-based PT, where we use LLM-based agents to replace RL/DRL-based agents to perform PT. Since the LLM is a pre-trained language model, it already has the intelligence to perform contextual understanding and reasoning for a variety of tasks at the beginning. To optimize the performance of the LLM agent in PT tasks, the integration of domain knowledge into the LLM is crucial. This can be achieved through fine-tuning using domain-specific databases or prompt engineering techniques such as the retrieval-augmented generation (RAG) approach [147]. LLM fine-tuning for PT is the process of taking an LLM as the base model and further training it using query-answer pairs extracted from existing PT knowledge to improve LLMs' performance in PT tasks. RAG for PT is a non-training approach that combines LLMs with external knowledge bases storing PT knowledge, and then retrieving necessary knowledge from it as the context to form the queries used in PT. Following these two angles, we will investigate the fine-tuning techniques by designing an appropriate training set from the cybersecurity domain and designing an appropriate vector database for retrieval in RAG.

6.3 Author's Publications

To the end, the author has published or submitted the following papers that are related to the topic of the thesis:

- Yuanliang Li, Jun Yan, and Mohamed Naili. "Deep reinforcement learning for penetration testing of cyber-physical attacks in the smart grid." 2022 International Joint Conference on Neural Networks (IJCNN). IEEE, 2022 [73].

- Yuanliang Li, Jun Yan, and Mohamed Naili. "Penetration Testing of Cyber-Physical Attacks in Smart Grids Based on Partially Observable Markov Decision Process." Submitted to IEEE Transactions on Dependable and Secure Computing (under review).

- Yuanliang Li, Hanzheng Dai, and Jun Yan. "Knowledge-Informed Auto-Penetration Testing Based on Reinforcement Learning with Reward Machine." Accepted by the 2024 IEEE World Congress on Computational Intelligence [74].

- Yuanliang Li, and Jun Yan. "Cybersecurity of smart inverters in the smart grid: A survey." IEEE Transactions on Power Electronics 38.2 (2022): 2364-2383 [1].

In addition, the author has other publications in related areas, but not within the main scope of this thesis, including:

- Yuanliang Li, Luyang Hou, et al. "A Novel Iterative Double Auction Design and Simulation Platform for Packetized Energy Trading of Prosumers in A Residential Microgrid." Accepted by Energy Conversion and Economics.

- Yuanliang Li, Luyang Hou, et al. "A Two-Stage Packetized Energy Trading and Management Framework for Virtual Power Plants." 2023 IEEE Power & Energy Society General Meeting (PESGM) [148].

- Hou, Luyang, Yuanliang Li, et al. "Multi-agent reinforcement mechanism design for dynamic pricing-based demand response in charging network." International Journal of Electrical Power & Energy Systems 147 (2023): 108843 [149].

- Yuanliang Li, Luyang Hou, et al. "PEMT-CoSim: A Co-Simulation Platform for Packetized Energy Management and Trading in Distributed Energy Systems." 2022 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm) [150].

- Moshfeka Rahman, Yuanliang Li, et al. "Multi-objective evolutionary optimization for worst-case analysis of false data injection attacks in the smart grid." 2020 IEEE Congress on Evolutionary Computation (CEC) [100].

Bibliography

[1] Yuanliang Li and Jun Yan. Cybersecurity of smart inverters in the smart grid: A survey. *IEEE Transactions on Power Electronics*, 2022.

[2] Mohamed C Ghanem and Thomas M Chen. Reinforcement learning for efficient network penetration testing. *Information*, 11(1):6, 2019.

[3] Net-zero emissions by 2050. Available at `https://www.canada.ca/en/services/environment/weather/climatechange/climate-plan/net-zero-emissions-2050/`.

[4] Ramyar Rashed Mohassel, Alan Fung, Farah Mohammadi, and Kaamran Raahemifar. A survey on advanced metering infrastructure. *International Journal of Electrical Power & Energy Systems*, 63:473–484, 2014.

[5] Simon Gill, Ivana Kockar, and Graham W Ault. Dynamic optimal power flow for active distribution networks. *IEEE Transactions on Power Systems*, 29(1):121–131, 2013.

[6] James A Momoh. *Electric power distribution, automation, protection, and control*. CRC press, 2017.

[7] Peter Palensky and Dietmar Dietrich. Demand side management: Demand response, intelligent energy systems, and smart loads. *IEEE transactions on industrial informatics*, 7(3):381–388, 2011.

[8] Hedayat Saboori, M Mohammadi, and R Taghe. Virtual power plant (vpp), definition, concept, components and types. In *2011 Asia-Pacific power and energy engineering conference*, pages 1–4. IEEE, 2011.

[9] Qi Huang, Waqas Amin, Khalid Umer, Hoay Beng Gooi, Foo Yi Shyh Eddy, Muhammad Afzal, Mahnoor Shahzadi, Abdullah Aman Khan, and Syed Adrees Ahmad. A review of transactive energy systems: Concept and implementation. *Energy Reports*, 7:7804–7824, 2021.

[10] Nasser Jazdi. Cyber physical systems in the context of industry 4.0. In *2014 IEEE international conference on automation, quality and testing, robotics*, pages 1–4. IEEE, 2014.

[11] Xi Fang, Satyajayant Misra, Guoliang Xue, and Dejun Yang. Smart grid—the new and improved power grid: A survey. *IEEE communications surveys & tutorials*, 14(4):944–980, 2011.

[12] Junhui Zhao, Caisheng Wang, Bo Zhao, Feng Lin, Quan Zhou, and Yang Wang. A review of active management for distribution networks: current status and future development trends. *Electric Power Components and Systems*, 42(3-4):280–293, 2014.

[13] Antonello Monti, Carlo Muscas, and Ferdinanda Ponci. *Phasor measurement units and wide area monitoring systems*. Academic Press, 2016.

[14] Mini S Thomas and John Douglas McDonald. *Power system SCADA and smart grids*. CRC press, 2017.

[15] Haibo He and Jun Yan. Cyber-physical attacks and defences in the smart grid: a survey. *IET Cyber-Physical Systems: Theory & Applications*, 1(1):13–27, 2016.

[16] Göran Andersson, Peter Donalek, Richard Farmer, Nikos Hatziargyriou, Innocent Kamwa, Prabhashankar Kundur, Nelson Martins, John Paserba, Pouyan Pourbeik, Juan Sanchez-Gasca, et al. Causes of the 2003 major grid blackouts in north america and europe, and recommended means to improve system dynamic performance. *IEEE transactions on Power Systems*, 20(4):1922–1928, 2005.

[17] Gaoqi Liang, Steven R Weller, Junhua Zhao, Fengji Luo, and Zhao Yang Dong. The 2015 ukraine blackout: Implications for false data injection attacks. *IEEE Transactions on Power Systems*, 32(4):3317–3318, 2016.

[18] U Cert. Russian government cyber activity targeting energy and other critical infrastructure sectors. *Us Cert*, pages 1–19, 2018.

[19] Mark Zeller. Common questions and answers addressing the aurora vulnerability. In *DistribuTECH Conference*, 2011.

[20] Siddharth Sridhar and Manimaran Govindarasu. Model-based attack detection and mitigation for automatic generation control. *IEEE Transactions on Smart Grid*, 5(2):580–591, 2014.

[21] Muhammad Shoaib Almas, Luigi Vanfretti, Ravi S Singh, and Gudrun M Jonsdottir. Vulnerability of synchrophasor-based wampac applications' to time synchronization spoofing. *IEEE Transactions on Smart Grid*, 9(5):4601–4612, 2017.

[22] Gaoqi Liang, Junhua Zhao, Fengji Luo, Steven R Weller, and Zhao Yang Dong. A review of false data injection attacks against modern power systems. *IEEE Transactions on Smart Grid*, 8(4):1630–1638, 2016.

[23] Xuan Liu, Zhiyi Li, Xingdong Liu, and Zuyi Li. Masking transmission line outages via false data injection attacks. *IEEE Transactions on Information Forensics and Security*, 11(7):1592–1602, 2016.

[24] Daranith Choeum and Dae-Hyun Choi. Vulnerability assessment of conservation voltage reduction to load redistribution attack in unbalanced active distribution networks. *IEEE Transactions on Industrial Informatics*, 2020.

[25] Armin Teymouri, Ali Mehrizi-Sani, and Chen-Ching Liu. Cyber security risk assessment of solar pv units with reactive power capability. In *IECON 2018-44th Annual Conference of the IEEE Industrial Electronics Society*, pages 2872–2877. IEEE, 2018.

[26] Devendra Shelar and Saurabh Amin. Analyzing vulnerability of electricity distribution networks to der disruptions. In *2015 American Control Conference (ACC)*, pages 2461–2468. IEEE, 2015.

[27] Martine Chlela, Geza Joos, and Marthe Kassouf. Impact of cyber-attacks on islanded microgrid operation. In *Proceedings of the Workshop on Communications, Computation and Control for Resilient Smart Energy Systems*, pages 1–5, 2016.

[28] Shichao Liu, Zhijian Hu, Xiaoyu Wang, and Ligang Wu. Stochastic stability analysis and control of secondary frequency regulation for islanded microgrids under random denial of service attacks. *IEEE Transactions on Industrial Informatics*, 15(7):4066–4075, 2018.

[29] Shichao Liu, Peter X Liu, and Xiaoyu Wang. Effects of cyber attacks on islanded microgrid frequency control. In *2016 IEEE 20th International Conference on*

Computer Supported Cooperative Work in Design (CSCWD), pages 461–464. IEEE, 2016.

[30] Cedric Carter, Ifeoma Onunkwo, Patricia Cordeiro, and Jay Johnson. Cyber security assessment of distributed energy resources. In *2017 IEEE 44th Photovoltaic Specialist Conference (PVSC)*, pages 2135–2140. IEEE, 2017.

[31] Ifeoma Onunkwo, B Wright, P Cordeiro, Nicholas Jacobs, Christine Lai, Jay Johnson, Trevor Hutchins, William Stout, Adrian Chavez, Bryan T Richardson, et al. Cybersecurity assessments on emulated der communication networks. Technical report, Sandia Technical Report, 2018.

[32] Abhinav Singh. Distributed intrusion detection system for modbus protocol. 2020.

[33] D Jonathan Sebastian and Adam Hahn. Exploring emerging cybersecurity risks from network-connected der devices. In *2017 North American Power Symposium (NAPS)*, pages 1–6. IEEE, 2017.

[34] Hamdi Albunashee, Chris Farnell, Andrew Suchanek, Kelby Haulmark, Roy McCann, Jia Di, and Alan Mantooth. A testbed for detecting false data injection attacks in systems with distributed energy resources. *IEEE Journal of Emerging and Selected Topics in Power Electronics*, 2019.

[35] Anomadarshi Barua and Mohammad Abdullah Al Faruque. Hall spoofing: A non-invasive dos attack on grid-tied solar inverter. In *29th USENIX Security Symposium (USENIX Security 20)*, pages 1273–1290, 2020.

[36] Babak Arbab-Zavar, Emilio J Palacios-Garcia, Juan C Vasquez, and Josep M Guerrero. Smart inverters for microgrid applications: A review. *Energies*, 12(5):840, 2019.

[37] Junjian Qi, Adam Hahn, Xiaonan Lu, Jianhui Wang, and Chen-Ching Liu. Cybersecurity for distributed energy resources and smart inverters. *IET Cyber-Physical Systems: Theory & Applications*, 1(1):28–39, 2016.

[38] Sonali Goel and Renu Sharma. Performance evaluation of stand alone, grid connected and hybrid renewable energy systems for rural application: A comparative review. *Renewable and Sustainable Energy Reviews*, 78:1378–1389, 2017.

[39] Yuanliang Li, Kun Ding, Jingwei Zhang, Fudong Chen, Xiang Chen, and Jiabing Wu. A fault diagnosis method for photovoltaic arrays based on fault parameters identification. *Renewable Energy*, 143:52–63, 2019.

[40] David M Scholten, N Ertugrul, and WL Soong. Micro-inverters in small scale pv systems: A review and future directions. In *2013 Australasian Universities Power Engineering Conference (AUPEC)*, pages 1–6. IEEE, 2013.

[41] Abraham Peedikayil Kuruvila, Ioannis Zografopoulos, Kanad Basu, and Charalambos Konstantinou. Hardware-assisted detection of firmware attacks in inverter-based cyberphysical microgrids. *International Journal of Electrical Power & Energy Systems*, 132:107150, 2021.

[42] Victoria Y Pillitteri and Tanya L Brewer. Nistir 7628 revision 1, guidelines for smart grid cybersecurity. *Smart Grid Interoperability Panel (SGIP), Smart Grid Cybersecurity Committee*, page 668, 2014.

[43] Frances Cleveland and Annabelle Lee. Cyber security for der systems version 1.0. Technical report, Electric Power Research Institute, 2013.

[44] A Lee. Electric sector failure scenarios and impact analyses. *National Electric Sector Cybersecurity Organization Resource (NESCOR) Technical Working Group*, 1, 2013.

[45] Christine Lai, Nicholas Jacobs, Shamina Hossain-McKenzie, Cedric Carter, Patricia Cordeiro, Ifeoma Onunkwo, and Jay Johnson. Cyber security primer for der vendors, aggregators, and grid operators. *Tech. Rep.*, 12, 2017.

[46] Jay Johnson. Roadmap for photovoltaic cyber security. *Sandia National Laboratories*, 2017.

[47] California Public Utilities Commission. Recommendations for updating the technical requirements for inverters in distributed energy resources, smart inverter working group recommendations, 2014.

[48] Danish Saleem and Cedric Carter. Certification procedures for data and communications security of distributed energy resources. Technical report, National Renewable Energy Lab.(NREL), Golden, CO (United States), 2019.

[49] C Birk Jones, Adrian R Chavez, Rachid Darbali-Zamora, and Shamina Hossain-McKenzie. Implementation of intrusion detection methods for distributed photovoltaic inverters at the grid-edge. In *2020 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT)*, pages 1–5. IEEE, 2020.

[50] Fangyu Li, Qi Li, Jinan Zhang, Jiabao Kou, Jin Ye, WenZhan Song, and Homer Alan Mantooth. Detection and diagnosis of data integrity attacks in solar farms based on multilayer long short-term memory network. *IEEE Transactions on Power Electronics*, 36(3):2495–2498, 2020.

[51] Kin Gwn Lore, Devu Manikantan Shila, and Lingyu Ren. Detecting data integrity attacks on correlated solar farms using multi-layer data driven algorithm. In *2018 IEEE Conference on Communications and Network Security (CNS)*, pages 1–9. IEEE, 2018.

[52] A Chavez, C Lai, N Jacobs, S Hossain-McKenzie, CB Jones, J Johnson, and A Summers. Hybrid intrusion detection system design for distributed energy resource systems. In *2019 IEEE CyberPELS (CyberPELS)*, pages 1–6. IEEE, 2019.

[53] CC Sun, R Zhu, and CC Liu. Cyber attack and defense for smart inverters in a distribution system. In *CIGRE Study Committee D2 Colloquium, Helsinki, Finland*, 2019.

[54] Subham Sahoo, Yongheng Yang, and Frede Blaabjerg. Resilient synchronization strategy for ac microgrids under cyber attacks. *IEEE Transactions on Power Electronics*, 36(1):73–77, 2020.

[55] Martine Chlela, Diego Mascarella, Geza Joos, and Marthe Kassouf. Cyber-resilient control of inverter based microgrids. In *2016 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 841–845. IEEE, 2016.

[56] Ankur Majumdar, Yashodhan P Agalgaonkar, Bikash C Pal, and Ralph Gottschalg. Centralized volt–var optimization strategy considering malicious attack on distributed energy resources control. *IEEE Transactions on Sustainable Energy*, 9(1):148–156, 2017.

[57] Paolo V Chiantore and James Watson. Operation & maintenance best practices guidelines 3.0. Technical report, SolarPower Europe, London, 2018.

[58] Cynthia K Veitch, Jordan M Henry, Bryan T Richardson, and Derek H Hart. Microgrid cyber security reference architecture. *Sandia National Laboratories: Albuquerque, NM, USA*, 2013.

[59] Patrick Engebretson. *The basics of hacking and penetration testing: ethical hacking and penetration testing made easy*. Elsevier, 2013.

[60] Art.32: Security of processing. Available at `https://gdpr-info.eu/art-32-gdpr/`.

[61] Norah Ahmed Almubairik and Gary Wills. Automated penetration testing based on a threat model. In *2016 11th International Conference for Internet Technology and Secured Transactions (ICITST)*, pages 413–414. IEEE, 2016.

[62] Learn the art of ethical hacking. Available at `https://www.g2.com/articles/penetration-testing`.

[63] Veenavati Jagadishprasad Mishra and Manisha D Khardenvis. Contingency analysis of power system. In *2012 IEEE Students' Conference on Electrical, Electronics and Computer Science*, pages 1–4. IEEE, 2012.

[64] Saleh Soltan, Dorian Mazauric, and Gil Zussman. Analysis of failures in power grids. *IEEE Transactions on Control of Network Systems*, 4(2):288–300, 2015.

[65] Abhinav Kumar Singh, Ravindra Singh, and Bikash C Pal. Stability analysis of networked control in smart grids. *IEEE Transactions on Smart Grid*, 6(1):381–390, 2014.

[66] Richard K Miller and Anthony N Michel. *Ordinary differential equations*. Academic press, 2014.

[67] Hermann W Dommel and William F Tinney. Optimal power flow solutions. *IEEE Transactions on power apparatus and systems*, (10):1866–1876, 1968.

[68] Chuadhry Mujeeb Ahmed, Martin Ochoa, Jianying Zhou, and Aditya Mathur. Scanning the cycle: timing-based authentication on plcs. In *Proceedings of the 2021 ACM Asia Conference on Computer and Communications Security*, pages 886–900, 2021.

[69] Rob Antrobus, Sylvain Frey, Benjamin Green, and Awais Rashid. Simaticscan: Towards a specialised vulnerability scanner for industrial control systems. In *4Th international symposium for ICS & SCADA cyber security research 2016*. BCS Learning & Development, 2016.

[70] Yuxi Li. Deep reinforcement learning: An overview. *arXiv preprint arXiv:1701.07274*, 2017.

[71] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022.

[72] Mitre att&ck. Available at `https://attack.mitre.org/`.

[73] Yuanliang Li, Jun Yan, and Mohamed Naili. Deep reinforcement learning for penetration testing of cyber-physical attacks in the smart grid. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 01–09. IEEE, 2022.

[74] Yuanliang Li, Hanzheng Dai, and Jun Yan. Knowledge-informed auto-penetration testing based on reinforcement learning with reward machine, 2024.

[75] What are pentesting tools. Available at https://www.hackerone.com/knowledge-center/7-pentesting-tools-you-must-know-about.

[76] Wireshark. Available at https://en.wikipedia.org/wiki/Wireshark.

[77] Burp suite. Available at https://en.wikipedia.org/wiki/Burp_Suite.

[78] John the ripper. Available at https://en.wikipedia.org/wiki/John_the_Ripper.

[79] David Maynor. *Metasploit toolkit for penetration testing, exploit development, and vulnerability research*. Elsevier, 2011.

[80] Nmap. Available at https://en.wikipedia.org/wiki/Nmap.

[81] Kali linux. Available at https://en.wikipedia.org/wiki/Kali_Linux.

[82] Jörg Hoffmann. Simulated penetration testing: From" dijkstra" to" turing test++". In *Proceedings of the international conference on automated planning and scheduling*, volume 25, pages 364–372, 2015.

[83] Carlos Sarraute. Automated attack planning. *arXiv preprint arXiv:1307.7808*, 2013.

[84] Verina Saber, Dina ElSayad, Ayman M. Bahaa-Eldin, and Zt Fayed. Automated penetration testing, a systematic review. In *2023 International Mobile, Intelligent, and Ubiquitous Computing Conference (MIUCC)*, pages 373–380, 2023.

[85] X Qiu, Q Jia, S Wang, C Xia, and L Shuang. Automatic generation algorithm of penetration graph in 734 penetration testing, 19th int. In *Conf. on P2P, Parallel, Grid, Cloud and Internet Computing*, volume 735, 2014.

[86] Carlos Sarraute, Olivier Buffet, and Jörg Hoffmann. Pomdps make better hackers: Accounting for uncertainty in penetration testing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 26, pages 1816–1824, 2012.

[87] Cynthia Phillips and Laura Painton Swiler. A graph-based system for network-vulnerability analysis. In *Proceedings of the 1998 workshop on New security paradigms*, pages 71–79, 1998.

[88] Kyle Ingols, Matthew Chu, Richard Lippmann, Seth Webster, and Stephen Boyer. Modeling modern network attacks and countermeasures using attack graphs. In *2009 Annual Computer Security Applications Conference*, pages 117–126. IEEE, 2009.

[89] Azqa Nadeem, Sicco Verwer, Stephen Moskal, and Shanchieh Jay Yang. Alert-driven attack graph generation using s-pdfa. *IEEE transactions on dependable and secure computing*, 19(2):731–746, 2021.

[90] George Stergiopoulos, Panagiotis Dedousis, and Dimitris Gritzalis. Automatic analysis of attack graphs for risk mitigation and prioritization on large-scale and complex networks in industry 4.0. *International Journal of Information Security*, 21(1):37–59, 2022.

[91] Yichao ZHANG, Tianyang ZHOU, Junhu ZHU, and Qingxian WANG. Domain-independent intelligent planning technology and its application to automated penetration testing oriented attack path discovery. , 42(9):2095–2107, 2020.

[92] Jonathon Schwartz and Hanna Kurniawati. Autonomous penetration testing using reinforcement learning. *arXiv preprint arXiv:1905.05965*, 2019.

[93] Zhenguo Hu, Razvan Beuran, and Yasuo Tan. Automated penetration testing

using deep reinforcement learning. In *2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, pages 2–10. IEEE, 2020.

[94] Mohamed C Ghanem and Thomas M Chen. Reinforcement learning for intelligent penetration testing. In *2018 Second World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4)*, pages 185–192. IEEE, 2018.

[95] Khuong Tran, Ashlesha Akella, Maxwell Standen, Junae Kim, David Bowman, Toby Richer, and Chin-Teng Lin. Deep hierarchical reinforcement agents for automated penetration testing. *arXiv preprint arXiv:2109.06449*, 2021.

[96] Qianyu Li, Miao Hu, Hao Hao, Min Zhang, and Yang Li. Innes: An intelligent network penetration testing model based on deep reinforcement learning. *Applied Intelligence*, 53(22):27110–27127, 2023.

[97] Cyberbattlesim. Available at `https://github.com/microsoft/cyberbattlesim`, 2021.

[98] Yao Liu, Peng Ning, and Michael 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.

[99] Fangfei Li and Yang Tang. False data injection attack for cyber-physical systems with resource constraint. *IEEE transactions on cybernetics*, 50(2):729–738, 2018.

[100] Moshfeka Rahman, Yuanliang Li, and Jun Yan. Multi-objective evolutionary optimization for worst-case analysis of false data injection attacks in the smart grid. In *2020 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8. IEEE, 2020.

[101] Ming Jin, Javad Lavaei, and Karl Henrik Johansson. Power grid ac-based state estimation: Vulnerability analysis against cyber attacks. *IEEE Transactions on Automatic Control*, 64(5):1784–1799, 2018.

[102] Mohamadsaleh Jafari, Mohammad Ashiqur Rahman, and Sumit Paudyal. Optimal false data injection attacks against power system frequency stability. *IEEE Transactions on Smart Grid*, 14(2):1276–1288, 2022.

[103] Mohamadsaleh Jafari, Mohammad Ashiqur Rahman, and Sumit Paudyal. Optimal false data injection attack against load-frequency control in power systems. *IEEE Transactions on Information Forensics and Security*, 2023.

[104] Paresh Risbud, Nikolaos Gatsis, and Ahmad Taha. Vulnerability analysis of smart grids to gps spoofing. *IEEE Transactions on Smart Grid*, 10(4):3535–3548, 2018.

[105] Yihai Zhu, Jun Yan, Yufei Tang, Yan Lindsay Sun, and Haibo He. Joint substation-transmission line vulnerability assessment against the smart grid. *IEEE transactions on Information Forensics and Security*, 10(5):1010–1024, 2015.

[106] Shuva Paul and Zhen Ni. Vulnerability analysis for simultaneous attack in smart grid security. In *2017 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT)*, pages 1–5. IEEE, 2017.

[107] Jun Yan, Haibo He, Xiangnan Zhong, and Yufei Tang. Q-learning-based vulnerability analysis of smart grid against sequential topology attacks. *IEEE Transactions on Information Forensics and Security*, 12(1):200–210, 2016.

[108] Zhen Ni and Shuva Paul. A multistage game in smart grid security: A reinforcement learning solution. *IEEE transactions on neural networks and learning systems*, 30(9):2684–2695, 2019.

[109] Shuva Paul and Zhen Ni. A study of linear programming and reinforcement learning for one-shot game in smart grid security. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2018.

[110] Zhenhua Wang, Haibo He, Zhiqiang Wan, and Yan Sun. Coordinated topology attacks in smart grid using deep reinforcement learning. *IEEE Transactions on Industrial Informatics*, 17(2):1407–1415, 2020.

[111] Xiaorui Liu, Juan Ospina, and Charalambos Konstantinou. Deep reinforcement learning for cybersecurity assessment of wind integrated power systems. *IEEE Access*, 8:208378–208394, 2020.

[112] Liang Yu, Yangyang Tan, Shuqi Qin, and Dong Yue. Deep reinforcement learning for online vulnerability identification in smart grid under load uncertainty. In *2022 China Automation Congress (CAC)*, pages 4036–4041. IEEE, 2022.

[113] Ying Chen, Shaowei Huang, Feng Liu, Zhisheng Wang, and Xinwei Sun. Evaluation of reinforcement learning-based false data injection attack to automatic voltage control. *IEEE Transactions on Smart Grid*, 10(2):2158–2169, 2018.

[114] Shiyu Xu, Shi Yu, Liang Xiao, and Zefang Lv. Reinforcement learning based vulnerability analysis for smart grids against false data injection attacks. In *International Conference on Wireless Algorithms, Systems, and Applications*, pages 441–452. Springer, 2022.

[115] Thomas Wolgast, Eric MSP Veith, and Astrid Nieße. Towards reinforcement

learning for vulnerability analysis in power-economic systems. *Energy Informatics*, 4:1–20, 2021.

[116] Christopher Neal, Hanane Dagdougui, Andrea Lodi, and José M Fernandez. Reinforcement learning based penetration testing of a microgrid control algorithm. In *2021 IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 0038–0044. IEEE, 2021.

[117] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[118] Thanh Thi Nguyen and Vijay Janapa Reddi. Deep reinforcement learning for cyber security. *arXiv preprint arXiv:1906.05799*, 2019.

[119] Gregory Palmer, Chris Parry, Daniel JB Harrold, and Chris Willis. Deep reinforcement learning for autonomous cyber operations: A survey. *arXiv preprint arXiv:2310.07745*, 2023.

[120] Markov decision process. Available at `https://en.wikipedia.org/wiki/Markov_decision_process`.

[121] Abhijeet Sahu, Hao Huang, Katherine Davis, and Saman Zonouz. Score: A security-oriented cyber-physical optimal response engine. In *2019 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*, pages 1–6. IEEE, 2019.

[122] Power quality information. Available at `https://www.hydroone.com/business-services/commercial-industrial-generators-and-ldcs/power-quality/power-quality-definitions`.

[123] Zhaoyu Wang and Jianhui Wang. Review on implementation and assessment of conservation voltage reduction. *IEEE Transactions on Power Systems*, 29(3):1306–1315, 2013.

[124] Fei Ding, Andu Nguyen, Sarah Walinga, Adarsh Nagarajan, Murali Baggu, Sudipta Chakraborty, Michael McCarty, and Frances Bell. Application of autonomous smart inverter volt-var function for voltage reduction energy savings and power quality in electric distribution systems. In *2017 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT)*, pages 1–5. IEEE, 2017.

[125] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[126] William Lardier. *ASGARDS-H: Enabling Advanced Smart Grid cyber-physical Attacks, Risk and Data Studies with HELICS*. PhD thesis, Concordia University, 2020.

[127] Bryan Palmintier, Dheepak Krishnamurthy, Philip Top, Steve Smith, Jeff Daily, and Jason Fuller. Design of the helics high-performance transmission-distribution-communication-market co-simulation framework. In *2017 Workshop on Modeling and Simulation of Cyber-Physical Energy Systems (MSCPES)*, pages 1–6. IEEE, 2017.

[128] Pieter Hintjens. *ZeroMQ: messaging for many applications*. " O'Reilly Media, Inc.", 2013.

[129] Bjorn Vaagensmith, Jacob Ulrich, Justin Welch, Timothy McJunkin, and Craig Rieger. Ieee 13 bus benchmark model for real-time cyber-physical control and

power systems studies. In *2019 Resilience Week (RWS)*, volume 1, pages 112–120. IEEE, 2019.

[130] Carlos Sarraute, Olivier Buffet, and Jörg Hoffmann. Penetration testing== pomdp solving? *arXiv preprint arXiv:1306.4714*, 2013.

[131] Dorin Shmaryahu, Guy Shani, Joerg Hoffmann, and Marcel Steinmetz. Partially observable contingent planning for penetration testing. In *Iwaise: First international workshop on artificial intelligence in security*, volume 33, 2017.

[132] Jonathon Schwartz, Hanna Kurniawati, and Edwin El-Mahassni. Pomdp+ information-decay: Incorporating defender's behaviour in autonomous penetration testing. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 30, pages 235–243, 2020.

[133] Yue Zhang, Jingju Liu, Shicheng Zhou, Dongdong Hou, Xiaofeng Zhong, and Canju Lu. Improved deep recurrent q-network of pomdps for automated penetration testing. *Applied Sciences*, 12(20):10339, 2022.

[134] Zegang Li, Qian Zhang, and Guangwen Yang. Eppta: Efficient partially observable reinforcement learning agent for penetration testing applications. *Engineering Reports*, page e12818, 2023.

[135] Partially observable markov decision process. Available at `https://en.wikipedia.org/wiki/Partially_observable_Markov_decision_process`.

[136] Power-flow study. Available at `https://en.wikipedia.org/wiki/Power-flow_study`.

[137] Cyber kill chain.

[138] Fabio Massimo Zennaro and László Erdődi. Modelling penetration testing with reinforcement learning using capture-the-flag challenges: Trade-offs between model-free learning and a priori knowledge. *IET Information Security*, 17(3):441–457, 2023.

[139] Qianyu Li, Min Zhang, Yi Shen, Ruipeng Wang, Miao Hu, Yang Li, and Hao Hao. A hierarchical deep reinforcement learning model with expert prior knowledge for intelligent penetration testing. *Computers & Security*, 132:103358, 2023.

[140] Yongjie Wang, Yang Li, Xinli Xiong, Jingye Zhang, Qian Yao, Chuanxin Shen, et al. Dqfd-aipt: An intelligent penetration testing framework incorporating expert demonstration data. *Security and Communication Networks*, 2023, 2023.

[141] Jinyin Chen, Shulong Hu, Haibin Zheng, Changyou Xing, and Guomin Zhang. Gail-pt: An intelligent penetration testing framework with generative adversarial imitation learning. *Computers & Security*, 126:103055, 2023.

[142] Rodrigo Toro Icarte, Toryn Klassen, Richard Valenzano, and Sheila McIlraith. Using reward machines for high-level task specification and decomposition in reinforcement learning. In *International Conference on Machine Learning*, pages 2107–2116. PMLR, 2018.

[143] Rodrigo Toro Icarte, Ethan Waldie, Toryn Q Klassen, Richard Valenzano, Margarita P Castro, and Sheila A McIlraith. Learning reward machines: A study in partially observable reinforcement learning. *arXiv e-prints*, pages arXiv–2112, 2021.

[144] Angela Orebaugh and Becky Pinkard. *Nmap in the enterprise: your guide to network scanning*. Elsevier, 2011.

[145] Rodrigo Toro Icarte, Toryn Q Klassen, Richard Valenzano, and Sheila A McIlraith. Reward machines: Exploiting reward function structure in reinforcement learning. *Journal of Artificial Intelligence Research*, 73:173–208, 2022.

[146] Yifan Yao, Jinhao Duan, Kaidi Xu, Yuanfang Cai, Zhibo Sun, and Yue Zhang. A survey on large language model (llm) security and privacy: The good, the bad, and the ugly. *High-Confidence Computing*, page 100211, 2024.

[147] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474, 2020.

[148] Yuanliang Li, Luyang Hou, Jun Yan, Yuhong Liu, Mohsen Ghafouri, and Peng Zhang. A two-stage packetized energy trading and management framework for virtual power plants. In *2023 IEEE Power & Energy Society General Meeting (PESGM)*, pages 1–5. IEEE, 2023.

[149] Luyang Hou, Yuanliang Li, Jun Yan, Chun Wang, Li Wang, and Biao Wang. Multi-agent reinforcement mechanism design for dynamic pricing-based demand response in charging network. *International Journal of Electrical Power & Energy Systems*, 147:108843, 2023.

[150] Yuanliang Li, Luyang Hou, Hang Du, Jun Yan, Yuhong Liu, Mohsen Ghafouri, and Peng Zhang. Pemt-cosim: A co-simulation platform for packetized energy management and trading in distributed energy systems. In *2022 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*, pages 96–102. IEEE, 2022.