ASEAN IVO Project
Final Report
Detailed Form

I. PROJECT TITLE

Cyber-Attack Detection and Information Security for Industry 4.0.

II. PROJECT LEADER

Full name:	Nguyen Linh Trung
Institution:	Advanced Institute of Engineering and Technology (AVITECH)
	VNU University of Engineering and Technology (VNU-UET)
	Vietnam National University, Hanoi (VNU)
Address:	E3, 144 Xuan Thuy, Cau Giay, Hanoi, Vietnam
Phone:	+84 24 3780 0150
E-mail:	linhtrung@vnu.edu.vn

III. PROJECT MEMBERS

Full Name (First, Middle, Family)	Position/ Degree	Department, Institution Country	Email Address
Nguyen Linh Trung (Project Leader)	Director/ Associate Professor	Advanced Institute of Engineering and Technology (AVITECH), VNU University of Engineering and Technology (VNU-UET), Vietnam National University, Hanoi (VNU).	linhtrung @vnu.edu.vn
Nguyen Viet Ha (Project Member)	Rector/ Associate Professor	VNU University of Engineering and Technology, Vietnam National University Hanoi, Vietnam.	hanv@vnu.edu.vn
Dusit Niyato (Project Member)	Professor	School of Computer Science and Engineering, Nanyang Technological University, Singapore.	dniyato@ntu.edu.sg
Eryk Dutkiewicz (Project Member)	Head of School/ Professor	School of Electrical and Data Engineering, University of Technology Sydney, Australia.	eryk.dutkiewicz @uts.edu.au
Diep Nguyen (Project Member)	Senior Lecturer/ Ph.D.	School of Electrical and Data Engineering, University of Technology Sydney, Australia.	diep.nguyen @uts.edu.au
Dinh Thai Hoang (Project Member)	Senior Lecturer/ Ph.D.	School of Electrical and Data Engineering, University of Technology Sydney, Australia.	hoang.dinh @uts.edu.au
Tran Thi Thuy Quynh	Ph.D.	Advanced Institute of Engineering and Technology (AVITECH), VNU University of Engineering and Technology (VNU-UET), Vietnam National University, Hanoi (VNU).	quynhttt@vnu.edu.vn
Ta Duc Tuyen	Ph.D.	Advanced Institute of Engineering and Technology (AVITECH), VNU University of Engineering and Technology (VNU-UET), Vietnam National University, Hanoi (VNU).	tuyentd@vnu.edu.vn

Bui Minh Tuan	Ph.D. student	School of Electrical and Data Engineering, University of Technology Sydney, Australia; Advanced Institute of Engineering and Technology (AVITECH), VNU University of Engineering and Technology (VNU-UET), Vietnam National University, Hanoi (VNU).	bui.m.tuan @student.uts.edu.au tuanbm.uet @vnu.edu.vn
Tran Viet Khoa	Ph.D. student	School of Electrical and Data Engineering, University of Technology Sydney, Australia; Advanced Institute of Engineering and Technology (AVITECH), VNU University of Engineering and Technology (VNU-UET), Vietnam National University, Hanoi (VNU).	khoa.v.tran @student.uts.edu.au khoatv.uet @vnu.edu.vn
Do Hai Son	M.Sc. student	Advanced Institute of Engineering and Technology (AVITECH), VNU University of Engineering and Technology (VNU-UET), Vietnam National University, Hanoi (VNU).	dohaison1998 @vnu.edu.vn

IV. PROJECT REPORT

1) Introduction

Information and communication technology (ICT) is expected to play an increasingly pivotal role in deepening economic integration and community building across the Association of Southeast Asian Nations (ASEAN), transitioning towards a digitally-enabled economy that is secure, sustainable, and transformative [1]. This Project considers the development of connected smart cities for a smart ASEAN society in general and in Vietnam in particular. A main driver for smart city development is Industry 4.0, in which ICT helps connect physical systems to the cyber-world, thereby enabling supply chain market more efficient, agile, and customer-focused. However, cyber-security risks become a key concern due to open systems with IP addresses, creating more avenues for cyber-attacks.

This Project aims to provide tools to enhance cyber-security in Industry 4.0, contributing to the enhancement of information reliability for smart society. In particular, it will develop (i) an innovative method to detect cyber-security threats in Industry 4.0 using advanced deep learning technology, (ii) an unprecedented framework to protect massive Industry 4.0 data from cyber-attacks using blockchain technology, and (iii) novel security solutions at the physical interface of information transmission using physical-layer security technology. To do this, we implement the following tasks to achieve the above objectives.

The project has 7 research tasks, as follows:

1. **Task 1:** Analyze and identify potential cyber-security risks in Industry 4.0.
2. **Task 2:** Develop an innovative risk assessment model to quantify the risks in Industry 4.0.
3. **Task 3:** Implement an online web reference service listing and ranking the risks in Industry 4.0.
4. **Task 4:** Develop and implement an innovative method to detect and isolate cyber-security attacks using deep learning.
5. **Task 5:** Develop an unprecedented data securing method using blockchain technology.
6. **Task 6:** Develop receiver-based friendly jamming and collaborative beamforming methods to safeguard sensors/actuators.
7. **Task 7:** Implement and evaluate performance of the proposed blockchain application on a real testbed.

To implement the above tasks, we have addressed 5 research problems, as follows:

1. ***Problem 1:*** *Framework for Cyber Risk Assessment for Industry 4.0 and Recommendations for Vietnam (for Tasks 1, 2, 3)*

   Industry 4.0 encompasses smart manufacturing and Internet of Things, which has brought huge benefits to a wide range of industries. This development, however, has raised more cyber security risks for both Information Technology (IT) and Operational Technology (OT) systems. In this work, potential cyber vulnerabilities and threats in manufacturing in Industry 4.0 are briefly reviewed based on the architecture and operating principle of the manufacturing system in Industry 4.0. Criteria for cyber risk assessment for both IT and OT are reviewed via different standards. We then provide recommendations for cyber risk assessment and discuss a new framework for IT/OT risk assessment in Vietnam.

2. ***Problem 2:*** *Collaborative Learning Model for Cyberattack Detection Systems in IoT Networks (for Task 4)*

   Federated Learning (FL) has recently become an effective approach for cyberattack detection systems, especially in Internet-of-Things (IoT) networks. By distributing the learning process across IoT gateways, FL can improve learning efficiency, reduce communication overheads and enhance privacy for cyberattack detection systems. However, one of the biggest challenges for deploying FL in IoT networks is the unavailability of labeled data and dissimilarity of data features for training. In this work, we propose a novel collaborative learning framework that leverages Transfer Learning (TL) to overcome these challenges. Particularly, we develop a novel collaborative learning approach that enables a target network with unlabeled data to effectively and quickly learn "knowledge" from a source network that possesses abundant labeled data. It is important that the state-of-the-art studies require the participated datasets of networks to have the same features, thus limiting the efficiency, flexibility as well as scalability of intrusion detection systems. However, our proposed framework can address these problems by exchanging the learning "knowledge" among various deep learning models, even when their datasets have different features. Extensive experiments on recent real-world cybersecurity datasets show that the proposed framework can improve more than 40% as compared to the state-of-the-art deep learning based approaches.

3. ***Problem 3:*** *Framework of Private Ethereum Blockchain Networks for Smart Grid (for Task 5)*

   A smart grid is an important application in Industry 4.0 with a lot of new technologies and equipment working together. Hence, sensitive data stored in the smart grid is vulnerable to malicious modification and theft. This work proposes a framework to build a smart grid based on a highly effective private Ethereum network.

Our framework provides a real smart grid that includes modern hardware and a smart contract to secure data in the blockchain network. To obtain high throughput but a low uncle rate, the difficult calculation method used in the mining process of the Ethereum consensus mechanism is modified to adapt to the practical smart grid setup. The performance in terms of throughput and latency are evaluated by simulation and verified by the real smart grid setup. The enhanced private Ethereum-based smart grid has significantly better performance than the public one. Moreover, this framework can be applied to any system used to store data in the Ethereum network.

4. ***Problem 4:*** *Learning-based Friendly Jamming with Imperfect CSI for Security in MIMO Wiretap Channel (for Task 6)*

   Using deep learning in communication has been a topic of interest recently. This work proposes a learning-based friendly jamming (FJ) method to secure communication in MIMO wiretap channels. This method is applicable for IoT security due to its low computational complexity at the receivers. Firstly, we introduce the autoencoder-based communication with FJ to guarantee secrecy. Unlike the previous works that require perfect channel state information (CSI) of legitimate channels at the transmitter, we can rely on the generalisation characteristic of neural networks to construct a robust FJ method in case of imperfect CSI. Secondly, we leverage FJ based on mutual information neural estimation (MINE) to demonstrate that it is possible to achieve a comparable security performance with the conventional FJ method without CSI. The proposed security schemes can combine MIMO security and detection tasks into a single end-to-end learning process, maximizing throughput and minimizing block error rate.

5. ***Problem 5:*** *Collaborative Learning for Cyberattack Detection in Blockchain Networks (for Task 7)*

   This work aims to study intrusion attacks and then develop a novel cyberattack detection framework for blockchain networks. Specifically, we first design and implement a blockchain network in our laboratory. This blockchain network will serve two purposes, i.e., to generate the real traffic data (including both normal data and attack data) for our learning models and implement real-time experiments to evaluate the performance of our proposed intrusion detection framework. To the best of our knowledge, this is the first dataset that is synthesized in a laboratory for cyberattacks in a blockchain network. We then propose a novel collaborative learning model that allows efficient deployment in the blockchain network to detect attacks. The main idea of the proposed learning model is to enable blockchain nodes to actively collect data, share the knowledge learned from its data, and then exchange the knowledge with other blockchain nodes in the network. In this way, we can not only leverage the knowledge from all the nodes in the network but also do

not need to gather all raw data for training at a centralized node like conventional centralized learning solutions. Such a framework can also avoid the risk of exposing local data's privacy as well as the excessive network overhead/congestion. Both intensive simulations and real-time experiments clearly show that our proposed collaborative learning-based intrusion detection framework can achieve an accuracy of up to 97.7% in detecting attacks.

2) Project Activities

(1) Development and Implementation

In what follows, we will describe the development and implementation of the above-mentioned five research problems.

PROBLEM 1: New Framework for Cyber Risk Assessment for Industry 4.0

Introduction

Industry 4.0 has made advancements to manufacturing in optimizing supply chains and assets, analyzing and predicting maintenance problems by applying new technologies such as Internet-of-Things (IoT) and cloud computing [2].

Such developments, however, face an increase in cyber attacks (CBAs) since manufacturing systems are exposed to the Internet; also, new types of CBAs appear.

It was estimated that CBAs cost the global economy up to 400 billion USD a year [3]. For smart manufacturing, CBAs are even more severe since it would lead to system failure. Therefore, organizations should pay more attention to cyber risk management to mitigate the consequence of CBAs.

Cyber risks assessment (RAS) has an important role in cyber risk management, which helps a security system to make accurate decision on risk treatment. Outcomes of an RAS process are determined based on vulnerabilities, threats, and assets. According to ISO/IEC 27005:2008 [4], a vulnerability is a weakness or hole of a security program that threats can exploit to gain illegitimate access to the assets of an organization. A threat is identified as what impact each vulnerability will have on the assets. Threats may arise from objective or subjective reasons and be intentional or unintentional attacks. Risks refer to potential consequences when threats can cause damage to the assets of an organization.

Typically, a manufacturer consists of information technology (IT) and operational technology (OT) systems. The former consists of computers and telecommunication for storing, recovering, transmitting, manipulating, protecting data or information, and exchanging data among different organizations. The latter is defined as a system of software and hardware to manage/monitor physical devices, machines, as well as processes and product segments in the operation of an enterprise [5].

Numerous international standards and frameworks have been developed for RAS. For examples, NIST SP 800-30 and ISO/IEC 27001:2005 for IT are published by the National Institute of Standards and Technology (NIST) and the International Standards Organization (ISO), respectively. For OT, ISA/IEC 62443 series is jointly published by the International Society for Automation (ISA) and the International Electrotechnical Commission (IEC). It presents a process-based approach for deploying, implementing, operating, and maintaining security.

Previous studies propose practical approaches and models for RAS of IT [6, 7], of OT [8, 9], and of IoT [10]. However, they looked at RAS for IT, OT, and IoT separately and did not consider new vulnerabilities, threats, and assets (VTA) in Industry 4.0.

Further, existing RAS frameworks have been created in each organizational perimeter [11]. They mainly deal with information security [12] and are based on antivirus software, firewall, intrusion detection, and malware protection tools to detect vulnerabilities and corresponding threats at the entry of the perimeter. However, the boundaries between the physical world and the cyber world, and among companies are blurred when IoT is applied in Industry 4.0. Data and intellectual property is now shared on the Internet across partner companies. Also, IT and OT networks can be accessed from many points via smart sensors, IoT devices, and clouds. Hence, the existing RAS frameworks may not be sufficient and need to be upgraded.

The contributions of this work are as follows:

- We provide a brief review of methodologies and existing standards used for cyber RAS, primarily focusing on OT and recommendations for improving Cyber RAS for IT and OT systems in Industry 4.0 in Vietnam.
- We propose a possible framework for IIoT risk assessment in Vietnam. The proposed framework considers IT, OT, and IIoT system. We build an experiment that simulates an IIoT network and compare with several existing frameworks. The results show that our method gives the same severity level as OWASP.

Architecture, Operating Principle, Vulnerabilities and Threats of Manufacturing in Industry 4.0

This section will discuss the architecture and operating principles of manufacturing. It is the basis for determining vulnerabilities and threats in Industry 4.0.

Architecture and Operating Principle

A manufacturing system in Industry 4.0 combines a cyber-physical system (CPS) with IoT technology. A cybersystem represents entities and functions related to IT, while physical systems include production processes and applications. CPS combines IoT technology in production processes and other services [13].

The architecture of a CPS in Industry 4.0 is illustrated in Figure 1 [14]. In the architecture, sensor networks connect directly to the Internet. The interconnection of sensors and actuators, and computing provide the ability to collect raw data from a real-world environment via sensors and exchange the data across platforms for analysis and processing. Monitoring, planning, and controlling can be performed via the Internet. A large amount of collected data from sensor networks can be used to predict issues of maintenance and improve productivity and risk management by giving back control data

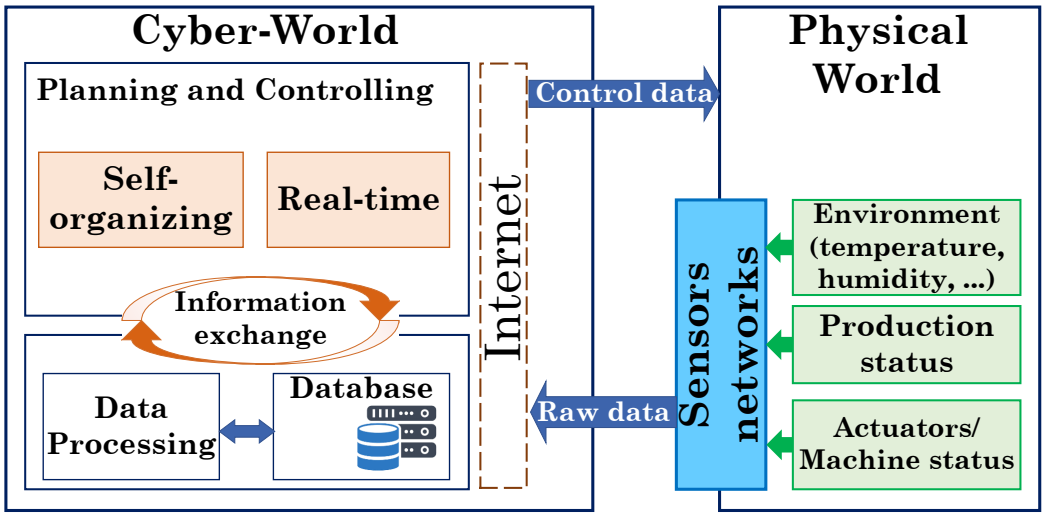


Figure 1: Cyber–Physical Systems leveled architecture.

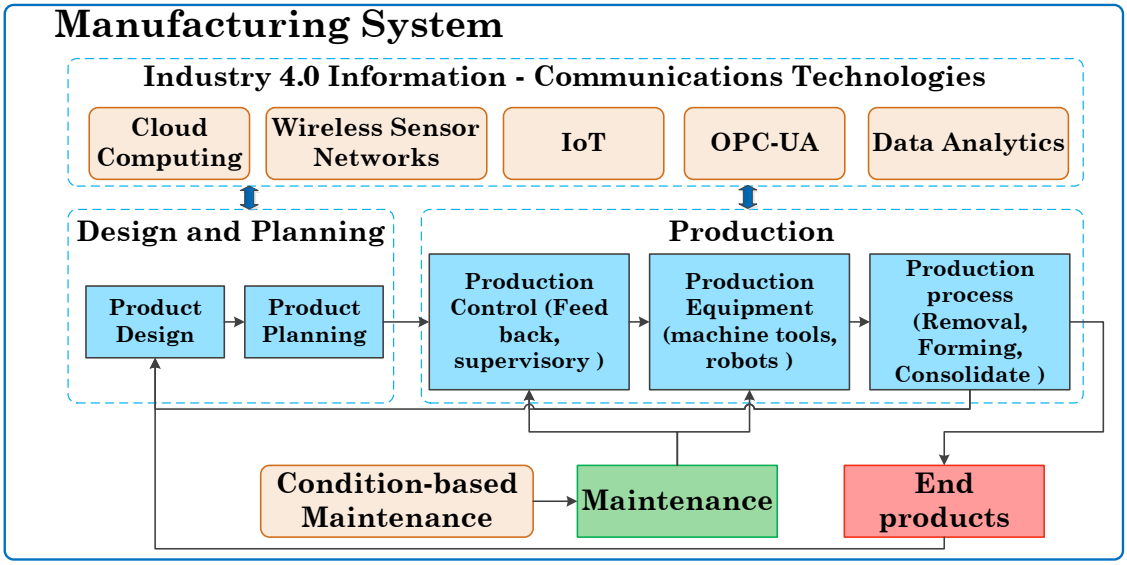


Figure 2: Manufacturing workflow in Industry 4.0 [17].

to the physical world [15]. That process will provide a self-optimizing capability of the system.

The operation principle of a manufacturing system in Industry 4.0 is illustrated in the workflow in Figure 2 [16]. The figure illustrates the whole process of production with supportive technologies such as robotics, cloud computing, data analytics, IoT, and smart sensors. Communication technologies for exchanging data from sensors to clouds, such as OPC-UA [18], wireless sensor networks, and Web services, maintain the communica-

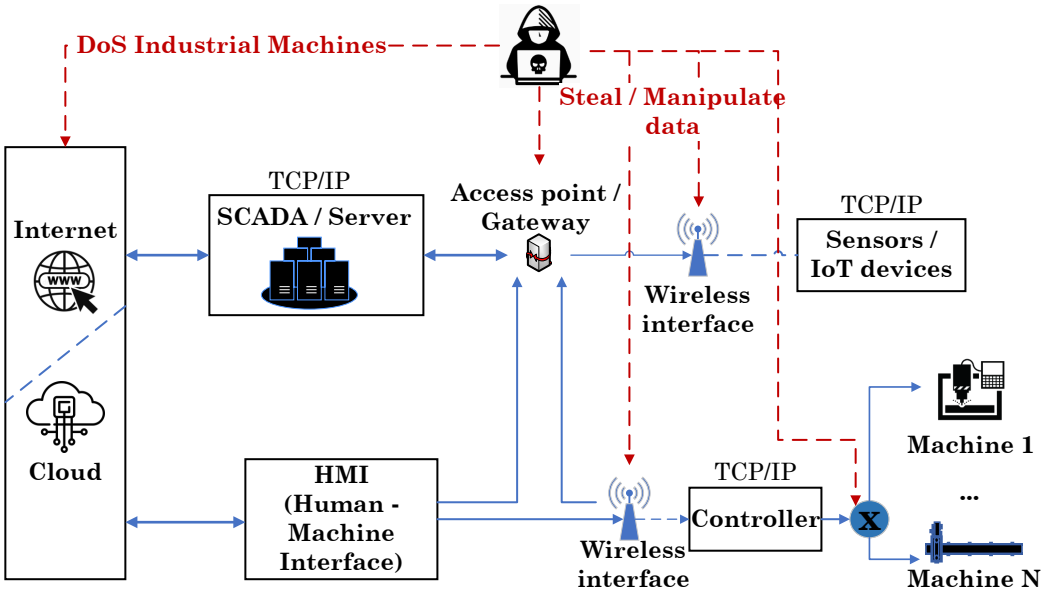


Figure 3: Example of threats in OT system.

tion between humans and machines. The production process is operated by IT and OT systems. From designing to the production stage, data are stored and exchanged via the Internet for processing by cloud computing.

Vulnerabilities and Threats of IT Systems

The advancements of the Internet and the IoT Internet have made an impact on IT systems in Industry 4.0 in terms of cyber threats. Some main types of threats are grouped in Table 2. Firstly, when most of the devices can connect to the Internet, IT systems have to face cyber attacks more frequently. However, operating systems and software used to operate hardware may no longer be supported because factories rarely stop operations to upgrade IT systems. Hence, the system cannot be maintained and supported and could be exploited by old variants of network malware. Secondly, autorun.inf related cyber risks have been detected significantly in manufacturing as compared to in other industries. The common practice of using USB drives to copy and transfer information between computers and networks could be an ideal way of malware propagation, e.g., Stunex. Thirdly, companies and technical teams now tend to use clouds to share their work. Hackers can perform targeted campaigns that aim to steal intellectual property or critical information. Potential threats come from insecure clouds [19].

Vulnerabilities and Threats of OT Systems

The main vulnerabilities and threats of OT systems are presented in Table 3. Human-machine interfaces (HMIs) allow operators and engineers to monitor and control the equipment while programmable logic controllers are used to program logic into several pieces of equipment. However, industrial malware can access HMIs when exchanging data via the Internet or the USB. Figure 3 illustrates an example of the OT system in Industry 4.0 with threats from an attacker. OT devices and industrial control systems (ICS) can connect to the Internet using Web-based protocols [20], increasing the possibility of cyber attacks. Most attacks have exploited insecure communications between hardware and software of OT systems. A hacker can access the network and perform an interception to steal data and manipulate the system.

Vulnerabilities and Threats of IoT Systems

While IIoT and Industry 4.0 are separate concepts, they should not be viewed that way when introducing greater efficiency and automation in manufacturing [21]. From the reference model of the open systems interconnection (OSI), an IoT system contains perception/physical, network, service, and application layers [14]. Thereby, cyber threats will be examined from each layer. In the perception layer, unauthorized access is the most concern because hackers can use malicious sensors or unauthorized IoT devices to gain information exchanging among the entities of the system. Attackers can manipulate the system from the received data, makes it stop working, or damage them. In the network layer, threats include Denial of Services (DoS), routing, man-in-the-middle attacks, and data breaches. In the service layer, attackers mainly use malicious information to manipulate users, which can be seen in spoofing or phishing attacks. They try to get user or system information by pretending to be legitimate businesses or partners. The application layer is the layer closest to users and faces numerous attack interfaces. Attackers can exploit poor security applications through the HMI connecting internet interface. They can perform malicious code injection, illegitimate configuration, and phishing attacks.

From the IIoT perspective, previous works propose different layers for IIoT [22–24]. In [24], the authors classified the IoT network into five layers including business layer, application layer, middleware layer, network layer and perception layer. These layers can

Table 2: Common threats in IT systems

Vulnerabilities	**IT Threats**
Long replacement cycle in operating system	Old variant malware
Software is no longer be supported	Pervasiveness of network worms
Using USB drives to copy and transfer information	Auto-run
The importance of industry	Targeted campaigns

Table 3: Common threats in OT systems

Vulnerabilities	OT Threats
Modern Human-Machine Interfaces (HMIs) expose to the Internet	Unauthorized tampering
Manufacturing equipment is not designed with security	Industrial malware
Insecure communications from sub-systems to higher-level systems	Illegitimate reconfigure sub-systems
Data breach from IT systems to ICS devices	Malware targeting ICS

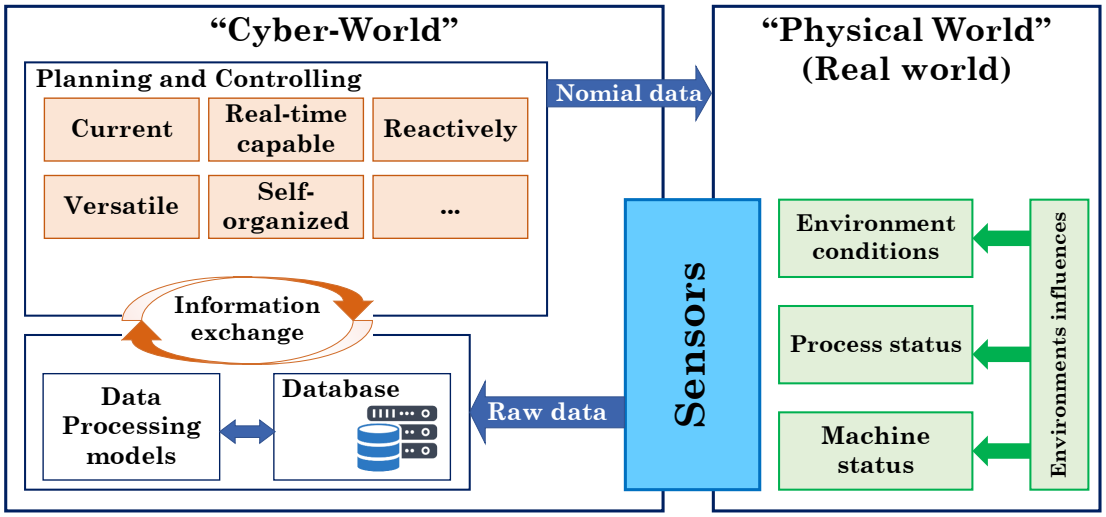


Figure 4: Architecture of the IIoT network.

cover nearly all purposes and technical architecture of IoT as well as IIoT networks. In this work, we focus on the following three layers of an IIoT network: perception, edge, and cloud/data, as described in Figure 4.

- **Perception layer:** This layer includes an enormous number of sensors, cameras, controllers, and smart devices. These devices play a crucial role in an IIoT system by collecting information from the system and the environment such as temperature, humidity, power, and state, as well as executing the decision from other layers. Through these devices, the other layers can have information about the working system and perform analysis to find out the abnormal behavior of the system and make decisions. The decisions are then sent back to these layers to be implemented.
- **Edge layer:** This layer, as the edge gateway, receives and collects data from the perception layer. It also can perform some slightly additional functions such as edge analysis, storage, data aggregation, or making quick decisions for some authorized tasks.
- **Cloud and data layer:** This layer is the heart of an IIoT system. The functions

of this layer are management, processing, and monitoring of a large amount of data collected from the perception layer. In addition, this layer applies novel technologies, e.g., machine learning to learn the historical data and then predict or identify abnormal behaviors in the network. After analysis, the data are also stored in the storage system of this layer.

Cyber Risk Assessment

RAS Methodologies

An RAS process contains two steps: risks analysis and risks evaluation [25]. The former uses information systems to identify sources and estimate risks. The latter compares the estimated risks with an acceptable risk level to determine the severity of the risks (ISO/IEC 27001:2013). Hence, RAS is a non-trivial task since it involves all kinds of platforms, operating systems, application programs, networks, people, processes, and inter-dependencies. Based on the analysis of the scenario and the target of the organization in assessing risks, cyber risks can be evaluated in the main approaches as seen in Table 4. Among them, appraisement is widely used in organizations. It is divided into three categories (quantitative, qualitative, and hybrid) as presented in Table 5.

RAS for IT

There is a wide range of laws and regulations worldwide to manage and assess cyber risks, including NIST and ISO/IEC 27001. NIST Cyber Security Framework (CSF) and NIST Risk Management Framework (RMF) were created to acknowledge and standardize specific controls and processes. ISO/IEC 27001 is widely used for managing information security. It outlines a method for performing an information security management system (ISMS) of an organization and then certifies the method. It also introduces general security techniques that help governments and organizations solve problems of information security. Both ISO and NIST standards are created for ISMS and RAS from different aspects and involve different scopes, as demonstrated in Table 6 [26], where ISO/IEC 27001:2005 and NIST SP 800-30r1 are the two frameworks that provide guidance for RAS.

Table 4: Risks evaluation approaches

Appraisement	**Perspective**	**Resource Valuation**	**Risk Measurement**
Quantitative	Asset-driven	Vertical View	Non-Propagated
Qualitative	Service-driven	Horizontal View	Propagated
Hybrid	Business-driven		

Table 5: Appraisement: quantitative, qualitative, and hybrid

Methods	Advantages	Disadvantages
Quantitative: (monetary value or percentages) Inputs and outputs can be monetary and non-monetary	(i) The levels of estimated risks can be identified in monetary terms; (ii) The levels of estimated risks can be illustrated in numerical results	(i) Difficult to estimate probabilities of threats; (ii) Expensive and time-consuming since the calculation of risks level will take much time to monitor and record the events
Qualitative: (non-numerical methods) Inputs and outputs are linguistic and range or rank variables respectively	(i) Threat likelihood information may not be required; (ii) Can perform quickly; (iii) Cost effective; (iv) Risk assessment can be performed by operators that not are experts on security or computers	(i) Monetary and probabilities are not achieved; (ii) Lack of sufficient measurable detail; (iii) Highly depend on the knowledge of operators and may not accurate
Hybrid (NIST SP 800-30r1): Using both quantitative and qualitative	Flexibility	

RAS for OT

The difference between RAS for OT systems compared to RAS for IT systems can be seen in information assets, in which RAS is evaluated in terms of confidentiality (C), integrity (I), and availability (A) measures. Though criteria of information security in OT and IT systems include confidentiality, integrity and availability altogether, these measures are not given the same priority, as illustrated in Figure 5 [26].

The IT system considers confidentiality the most important in the group of the three measures. In contrast, the OT system complements control and prioritizes control and availability. Their order in IT systems is CIA (confidentiality, integrity and availability) while in OT systems is CAIC (control, availability, integrity and confidentiality), respectively. That difference will significantly affect the cyber security frameworks.

The ISA/IEC 62443 standards, first created by ISA and then developed by IEC, deal with security issues unique to OT systems and specifically for ICS. It consists of four main areas: general basics, operators and service providers, requirements for automation systems, and automation component requirements. ISA/IEC 62443 and ISO 27001 standards have some similarities in contents related to policies such as management commitments and organization responsibilities.

Figure 6 illustrates the frameworks and standards that adapt to the IT and OT environments. The arrow from the left to right presents the decrease in the levels of

Table 6: NIST and ISO Comparison

NIST	ISO 27001
First intent built to help the United States of American organizations manage risks	Internationally recognized approach for ISMS
Three key components: core, implementation tiers, and profiles with categories	Not focus on details of technical methods, concentrate on ISMS and provide recommendations
Control catalogs, 5 functions, 21 categories, and 78 subcategories	Annex A has 14 Control Domains, with 114 total controls
Voluntary, self-certification mechanism, and not certifiable	Relies on independent audit and certification bodies

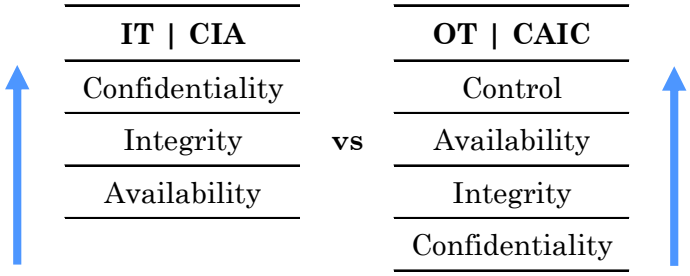


Figure 5: Different priorities of OT and IT in terms of cyber security.

confidentiality, where the ISA/IEC 62443 standards are considered as "made for OT", such as standard IEC 62443-3-2 suggests is the guidance of RAS for system design. There are frameworks used for risks managements for OT systems and ICS that are included in NIST, such as NIST Special Publication 800-82 R2.

Vietnamese Standards for Cyber RAS

Vietnamese agencies and organizations implement information security risk management that is based on the following principles (stated in standard TCVN 10295:2014):

1. Risk management must be conducted regularly and continuously in accordance with the regulations, policies, processes, and procedures to ensure the information security of agencies and organizations;

2. Risk treatment should be conduct feasibility and guarantee the balance between the cost and the efficiency.

3. Decentralize risk, and avoid transferring the risk to reduce the consequences.

Measures for ISMS and RAS are promulgated by the Authority of Information Security (AIS) based on the ISO/IEC 27005:2011 framework, the deployment of the ISO/IEC

Standards	ISO 27001	NIST CSF	NIST CSF + Special publications	ISA 62443	ISA 62443 + Technical References
Confidential level	Ultra high	High	Medium	Low	Low
Usage	Rework needed	Need work for OT	Can work for OT	Made for OT	Made for OT

IT ⟶ OT

Figure 6: Different standards for OT and IT in term of cyber security.

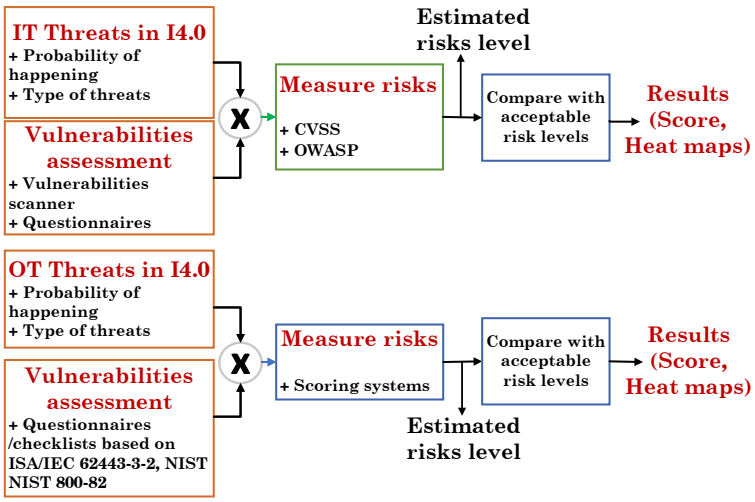


Figure 7: Recommendation of RAS for IT and OT systems.

27001. Vietnam's National Standards for Cybersecurity (VSC) and their corresponding ISO standards are presented in Table 7. Since VSC is based on ISO/IEC 27001, they are *mainly* applied to IT systems. Regarding RAS, there are several standards for IT systems but none for OT systems. RAS for OT needs to be consulted by the government or experts in that area. The information security for ICS is only mentioned in Circular 03/2017/TT-BTTTT of the Ministry of Information and Communications of Vietnam.

VSC standards for securing OT systems should be established in a systematical and synchronous way, among organizations, assurances, governance, and technical measures.

Recommendations for Cyber RAS for IT and OT systems in Vietnam

In this part, we provide recommendations for RAS for IT and OT systems that are illustrated in Figure 7. For IT systems, the system assets are evaluated based on the

Table 7: Vietnam National Standards for Cybersecurity and corresponding ISO standards

Vietnam National Standards	ISO
TCVN ISO/IEC 27001:2009	ISO/IEC 27001:2005 – IT - ISMS - Requirements
TCVN ISO/IEC	ISO/IEC 27002:2005 – IT - Security techniques - Code of practice for ISMS
TCVN 10295:2014	ISO/IEC 27005:2011 – IT - Security techniques - ISMS
TCVN 8709-1:2011	ISO/IEC 15408-1:2008 – IT - Security Techniques - Part 1
TCVN 8709-2:2011	ISO/IEC 15408-2:2008 – IT - Security Techniques - Part 2
TCVN 8709-3:2011	ISO/IEC 15408-1:2008 – IT - Security Techniques - Part 3
TCVN 11386: 2016	ISO/IEC 18045:2008 – IT - Security techniques - Methodology for IT security evaluation
TCVN 11930:2017	NIST SP 800-53r4 – IT - Basic requirements for security information system according to security levels
TCVN ISO/IEC 27002:2020	ISO/IEC 27002:2013 – IT - Security techniques - Code of practice
TCVN ISO/IEC 27002:2020	ISO/IEC 27002:2013 – IT - Security techniques - Code of practice
TCVN 10295:2014	ISO/IEC 27005:2011 – IT - Security techniques - Information security risk management
TCVN 11239:2015	ISO/IEC 27035:2011 – IT - Security techniques - Information security incident management

security measures of C, I and A mentioned above, representing the importance and the necessity of the information levels of each organization. For example, the values of C, I, and A are 4, 3, and 2, respectively. The total value of the asset is the sum of those values, which is 9. The organization should follow Decree 85/2016/ND-CP of the government to determine the level of the IT system and the corresponding values of C, I and A. The vulnerabilities of the system can be detected by a scanner or set of questionnaires. The questionnaires are established based on RAS for IT following TCVN 10295:2014 (ISO/IEC 27005:2011).

For OT systems, since the priority levels are CAIC or AIC, the measures of assets need to be re-evaluated as abovementioned. The organization should follow the guidance of RAS for OT, such as ISA/IEC 62443-3-2 or NIST 822-82, to build practical questionnaires or checklists for determining the vulnerabilities. Measuring risks can be conducted by a scoring system such as the Common Vulnerability Scoring System (CVSS) [27]. The estimated risk is compared with the acceptance risks, which are defined by the organization or by referring to scoring systems such as the Common Vulnerabilities and Exposures (CVE) database [28]. Consequently, the levels of risks, such as low or high, are given and demonstrated on a heat map table or table of the levels.

Proposed RAS framework in Vietnam

In this section, we propose a new RAS framework for IT, OT and IIoT systems. For simplicity of scanning cybersecurity for the end-user IT devices, we build an open-source

Table 8: OWASP scoring system

Overall Risk Severity				
Impact	High	Medium	High	Critical
	Medium	Low	Medium	High
	Low	Note	Low	Medium
		Low	Medium	High
	Likelihood			

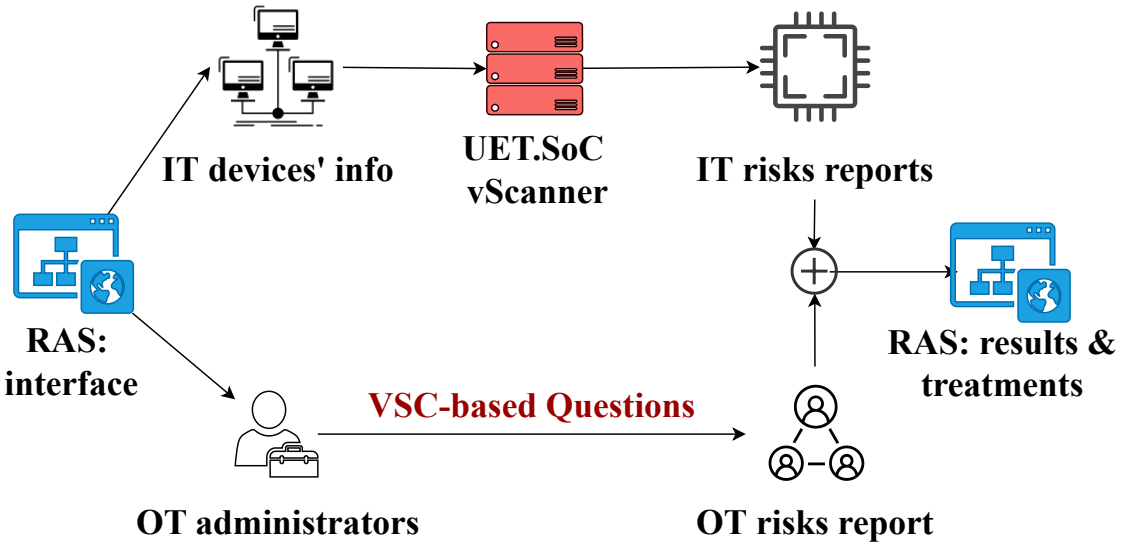


Figure 8: Proposed framework for RAS of IIoT.

Web application namely RASVN. Our application provides a high-end framework to scan IT risks and a set of TCVN-based questions about OT risks. The architecture is shown in Figure 8.

In detail, each device in the IoT system is scanned for IT risks by the proposed system namely UET.SoC in [7] to get a single report. Our proposed system uses vScanner software to scan the vulnerability of IT devices. This software uses the data of the list of vulnerabilities and exposures from CVE security vulnerability database [28], which uses CVSS [27] to mark the scoring system. CVSS is a well-known scoring system which classifies the vulnerabilities based on the severity levels of the heat map ranging from 0 as lowest to 10 as highest, as given in Table 9. To quantify the severity levels of the heat map, CVSS uses exploitability and impact measures including confidentiality (C), integrity (I), and availability (A) to identify the base score as follows [29]:

$$Severity\_level = 1 - [1 - A] * [1 - I] * [1 - C]. \quad (1)$$

As described in Figure 4, the IIoT systems include a large number of IT devices

Table 9: CVSS Scoring system

Score ranges	Level
0.0 - 3.9	Low
3.9 - 6.9	Medium
7.0 - 10.0	High

which perform different tasks in the network. The result from scanning these IT devices provides weights for individual IT devices. The weight of an IT device is indicated by the number of other devices which are affected when this IT device is under attacks. For example, a device at the edge layer which is a gateway for four sensors at the perception layer will have the weight of 4. A device is responsible for more devices will have higher points because they have more data and affect the IIoT system more seriously. In the proposed method, the severity level is accompanied to the weight of the devices:

$$Device\_severity = Severity\_level * Device\_weight. \tag{2}$$

The total risk of the IT system can then be calculated as

$$IT\_severity = \frac{\sum Device\_severity}{\sum weight}. \tag{3}$$

Table 10 summarizes our proposed set of question to evaluate OT systems. These questions provide guidance on how to secure ICS. It follows NIST SP-800-82-r2, including Supervisory Control and Data (SCADA) systems, distributed control systems, and other control system configurations such as PLC, while addressing their unique performance, reliability, and safety requirements. The standard provides an overview of ICS and typical system topologies, identifies typical threats and vulnerabilities to these systems, and provides recommended security countermeasures to mitigate the associated risks.

The level of risks of an OT system after having the answers of all questions are shown in Table 10. Each question is represented for a vulnerability of OT systems. Thus, the number of unanswered questions or questions with "No" as answer are described as flaws of the system. From these flaws, the risk of the OT system is determined as

$$OT\_severity = \frac{Q_{total} - Q_{yes}}{Q_{total}}, \tag{4}$$

where Q_{total} is the total number of questions and Q_{yes} is the number of "Yes" answers. After that, we map the result to the CVSS scoring system to find the severity level of the OT system.

IT and OT systems can play an equal role for the security of IIoT systems [30]. Having obtained RAS for the IT and OT systems above, we can now calculate the total

Table 10: Proposed set of questions used for determining vulnerabilities of OT systems.

Group	Number of questions
Access Control	8
Account Management	2
Communication Protection	35
Continuity	6
Environmental Security	10
Incident Response	1
Personnel	4
Physical Security	36
Portable/Mobile/Wireless	3
Remote Access Control	2
Software	1
System Integrity	4
System Protection	4
Training	1

severity of the complete IIoT systems by

$$Total\_severity = \frac{IT\_severity + OT\_severity}{2}. \quad (5)$$

Results

Proposed method

Next, we present our results in evaluating the cybersecurity risks of IT, OT, and IIoT systems. To evaluate the results of our proposed method, we build an experiment that simulates a real IIoT system as in Figure 9. In this experiment, there are 8 sensors at the perception layer, 2 IoT gateways at the edge layer, and 2 servers at the cloud and data layer. Each IoT gateway connects with 4 sensors by different industrial communication standards. The servers can connect with each other by a network, e.g., blockchain or local area network. This system is set up to work as a real IIoT system with real-time monitoring. After setting up the system, we assess the risk of this system by our proposed method and compare it with other state-of-the-art methods, i.e., CVSS and OWASP.

To assess the IT system, we first need to scan the vulnerabilities of all configurable devices in our network. However, most of the sensors are purely physical devices, they can not be scanned to find vulnerabilities. Thus, we scan the following devices in different layers to find the vulnerabilities: a BeagleBone wireless module, two IoT gateways and two servers. The risk scoring and the number of slave devices for each IT device are described in Table 11. From Equations 2 and 3, we found a the risk level of 5.9, which is medium, according to CVSS.

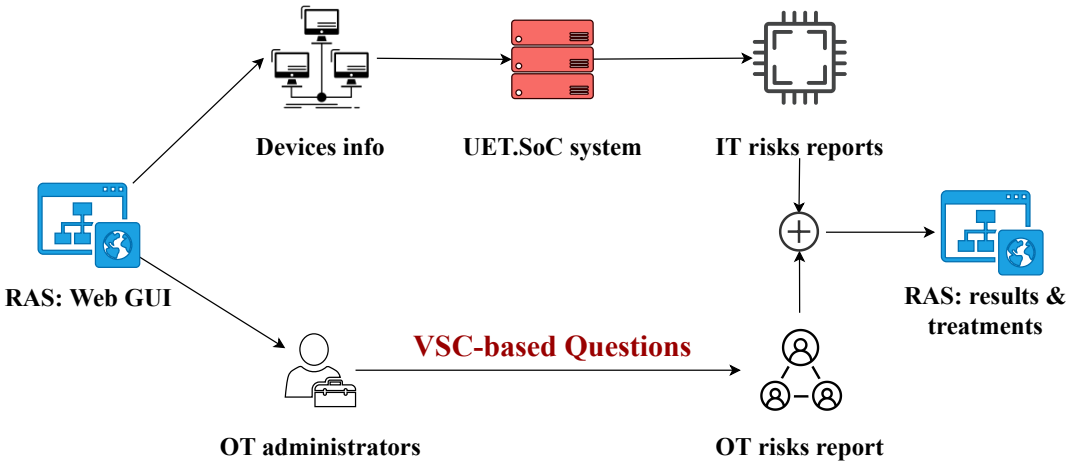


Figure 9: Our experimental model to simulate a real IIoT system.

Table 11: Risk scoring for IT devices

IT Devices	Risk scoring	Number of slave devices
Server 1	7.5	5
Server 2	0	5
IoT Gateway CPS200 RE	4.9	4
IoT Gateway Adlink 212	9	4
BeagleBone Wireless	9.3	1

To assess the OT system, we first need to answer a list of OT security questions in detail. Each question identifies a risk of an OT system. In our experimental system, after carefully checking each question, we identified a risk of 7.4, which is high according to CVSS.

Finally, the total risk can be identified by Equation (5). In our experimental system, we obtained a total risk of 6.65, which is medium, according to CVSS.

RAS by CVSS

The CVSS scoring system assesses the risks of a system based on three parts: base metrics, temporal metrics, and environmental metrics. The base metrics evaluate the internal system quality facing vulnerability, the temporal metrics evaluate the evolution characteristics over the lifetime of vulnerability, and the environmental metrics evaluate the vulnerability based on the environment or implementation. An example of using CVSS to assess the risk of a system is described in Figure 10.

We use CVSS as a baseline to compare with our RAS framework. The results of CVSS scoring are described in Table 12. With these results, our experimental IIoT system

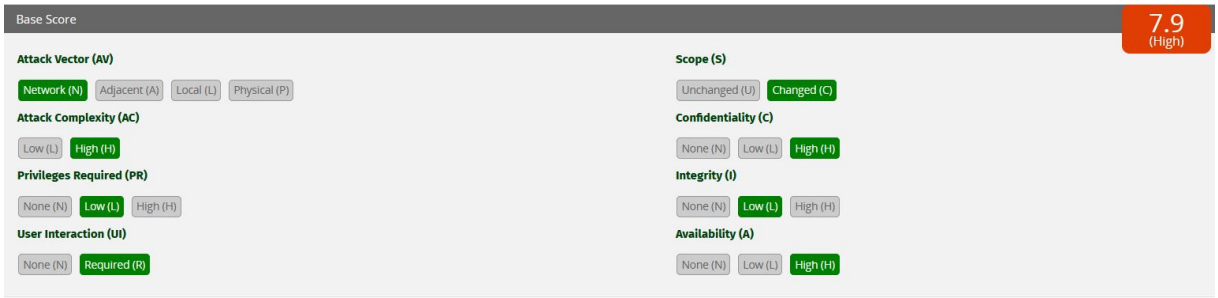


Figure 10: An example of CVSS risk evaluation.

Table 12: CVSS results from our system

	Base	Temporal	Environmental
Score	8.3	7.7	7.7
Level	High	High	High

identified the risk as high. It can be seen that CVSS rates our experimental system with a higher level of risk, in comparison with our proposed method. However, as described in the previous section, CVSS includes some common questions, and thus it is difficult for CVSS to understand the specified IT, and OT systems of the overall IIoT system, unlike our proposed method.

RAS OWASP

Unlike CVSS, OWASP [31] uses likelihood and impact scores to assess the risks of the system. While the likelihood score gives an estimate of a successful attack from a group of attackers, the impact score gives the impact of an attack on technical and business factors. The likelihood and impact scores are categorized into three levels corresponding to the severity of a system, namely low, medium, and high. The answers to the corresponding questions and the results of OWASP are presented in Figure 11. The risk levels are determined by combining the levels of impact and likelihood, as seen in Table 8. Figure 11 also provides the results of the likelihood score of 5.25 (medium) and impact score of 4.875 (medium) of our system. The risk level of our system was assessed as medium, according to Table 8. Thus, it can be seen that our proposed method provides a similar risk severity as OWASP, even if our proposed method focuses on the IT and OT systems for an IIoT network.

Risk assessment web-service for Vietnam

A risk assessment web-service[1] for Vietnam (RASVN) is built based on the proposed framework as abovementioned. Figure 12 shows the sequence diagram of RASVN, which

[1] `http://rasvn.systems`

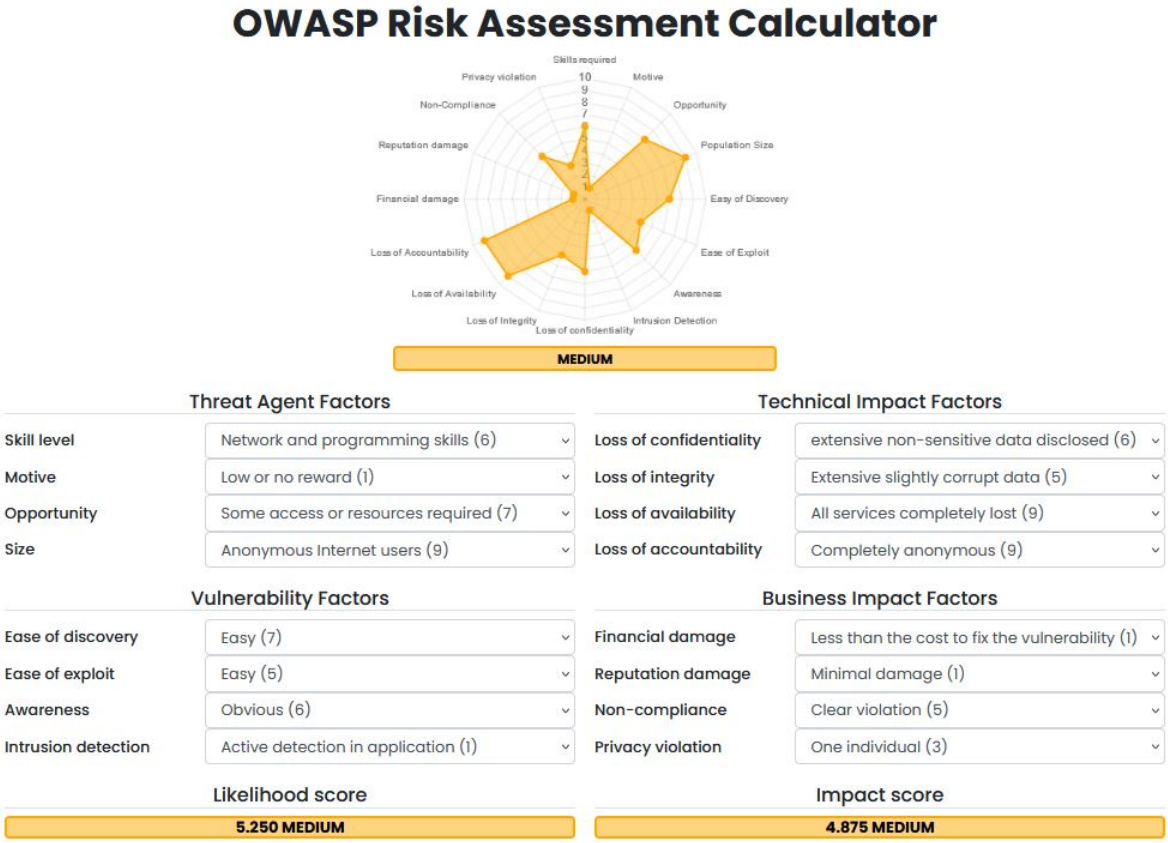


Figure 11: The evaluation results obtained by OWASP.

includes three objects: the website interface (front-end), the UET.SoC vScanner, and the website back-end.

In detail, the users need to provide RASVN with some identifying information (e.g., name, email), information on IT devices in the user system (e.g., public IP address, number of slave devices), and answer a list of OT questions. The rest of the processes are all automated, i.e., RASVN will create new scanning tasks on UET.SoC system based on IT devices' information. After the UET.SoC system has scanned all devices and reported IT risks of the devices, the RASVN back-end gets and sends them together with OT answers to the user interface. At the RASVN web interface, the IT and OT severity levels are used to calculate the total risk of the overall IIoT system. Finally, RASVN visualizes RAS results of the user system. Moreover, the full report is in raw type, which includes all detailed results of the RAS session, being available for download.

An example of results of RASVN is shown in Figure 13. Since we do not have enough public IPv4 addresses, we only assessed risks in the edge and perception layers of our experimental model as shown in Figure 9 (with an IoT Gateway CPS200RE, an IoT Gateway Adlink 212, a BeagleBone Wireless, and the other unscannable IoT devices/sensors). The first three devices were scanned to produce IT risk report. 117 OT

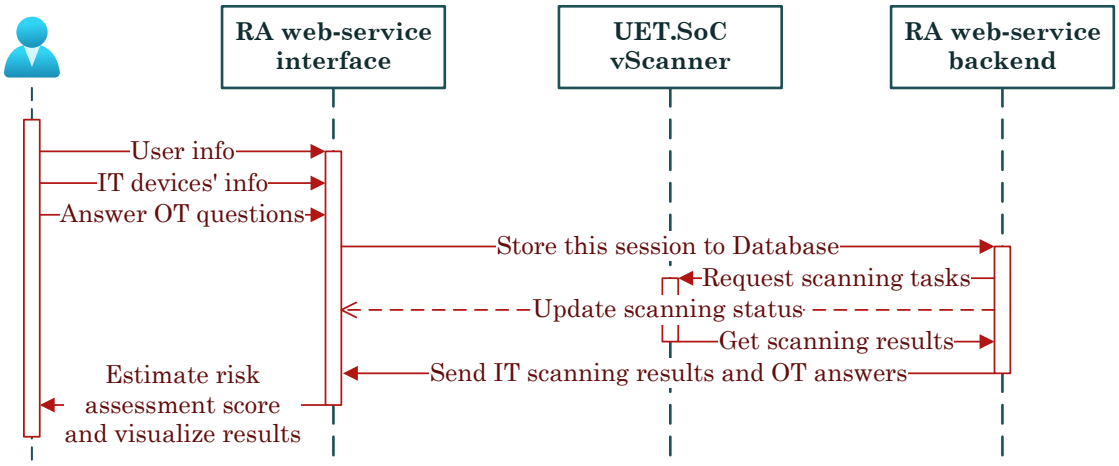


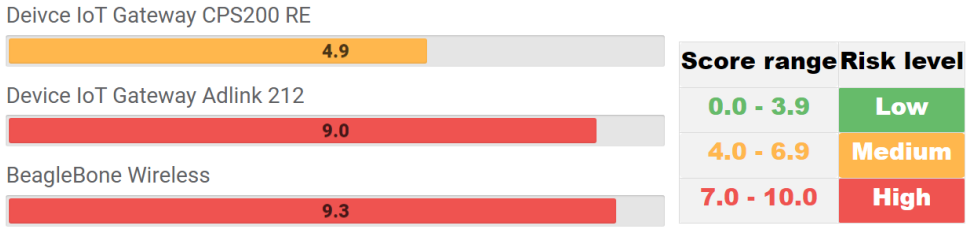
Figure 12: Sequence diagram of RA web-service for Vietnam.

questions are then answered, as shown in Table 13, for the OT risk report. In Figure 13(a) and 13(b), the IT score of each device and OT score for each device are visualized side by side; this helps users compare and find out the risky device/OT area. The final score results of the system are given in Figure 13(c), in which the IT, OT, and Overall risk scores are 7.2, 7.3, and 7.3, respectively. All of them are at the high level of risk. Users may then download our detailed report to check and treat system vulnerabilities.

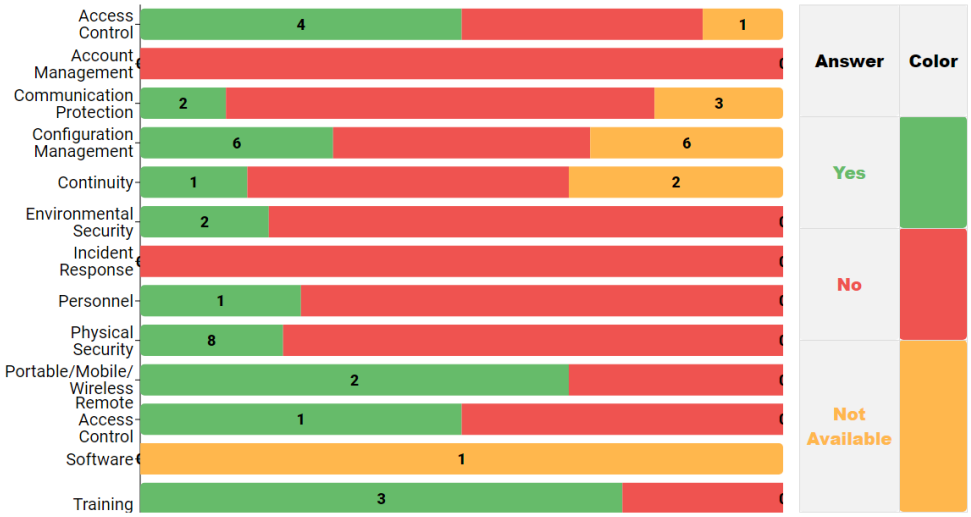
Conclusion

This work has provided an overview of potential cyber risks in Industry 4.0. We also review some standards for RAS in IT and OT systems. The cyber security standards used in Vietnam and their corresponding ISO and NIST standards are also showed in this work. Further, we recommend an approach for RAS and give a possible framework of RAS for OT systems. Finally, we propose a new framework for RAS in an IIoT system that can provide the total risk level of the system.

For future study, we realize that some issues need to be investigated related to RAS in Industry 4.0. Firstly, although vulnerabilities can be detected online, most of the existing RAS approaches or frameworks are conducted offline. It is difficult to deal with real-time processing that requires a dynamic risk assessment. Secondly, probabilities of given threats only hold with frequent attacks but may not hold with new types of threats or threats that rarely happen. Thus, the results of RAS will not be accurate. Lastly, acceptance risk levels in each organization are different, and they highly depend on their scale and function. Finding the scheme to determine the suitable acceptance risk levels is essential.



(a) IT score of each device



(b) OT score for each question group

IT Score	OT Score	Overall Score	Score range	Risk level
7.2	7.3	7.3	0.0 - 3.9	Low
			4.0 - 6.9	Medium
			7.0 - 10.0	High

(c) Overall scoring risk assessment

Figure 13: An example of risk assessment result using RA web-service for Vietnam.

Table 13: Risk assessment web-service for Vietnam: OT questions

No.	ID in CSET	Group	Question
1	239	Access Control	Are periodic reviews conducted of existing authorized physical and electronic access permissions to ensure they are current?
2	245	Access Control	Do electronic monitoring mechanisms alert system personnel when unauthorized access or an emergency occurs?
3	248	Access Control	Does the system enforce assigned authorizations for controlling electronic access to the system?
4	249	Access Control	Are access control policies and associated access mechanisms to control access to the system?

5	258	Access Control	Is a device verified against a pre-defined list of authorized devices before a connection is established? (e.g., Active Directory policy or firewall rules.)
6	259	Access Control	Does the system authenticate devices before establishing remote network connections using bi-directional authentication between devices that are cryptographically based?
7	263	Access Control	If your authentication encryption module fails can you still authenticate without creating a denial of service that impacts operational performance of system?
8	279	Access Control	Does the system prevent further access to the system by initiating a session lock after a defined time period of inactivity or a user initiated session lock?
9	226	Account Management	Are users required to take, and devices implement, specific measures to safeguard authenticators?
10	230	Account Management	Are unique authenticators required to be provided by vendors and manufacturers of system components?
11	384	Communication Protection	Do the system components separate telemetry/data acquisition services from management port functionality?
12	391	Communication Protection	Is the unauthorized release of information outside the system boundary or any unauthorized communication through the system boundary prevented when an operational failure occurs of the boundary protection mechanisms?
13	395	Communication Protection	Does the system prevent remote devices that have established connections (e.g., PLC, remote laptops) with the system from communicating outside that communications path with resources on uncontrolled/unauthorized networks?
14	398	Communication Protection	Have you evaluated the latency issues introduced by the use of cryptographic mechanisms to ensure that they do not impact operational performance?
15	406	Communication Protection	Are collaborative computing devices (e.g., video and audio conferencing) restricted on your control system network?
16	407	Communication Protection	Are collaborative computing devices disconnected and powered down when not in use?
17	409	Communication Protection	Are collaborative computing devices disabled or removed from systems in secure work areas?
18	410	Communication Protection	Does the system reliably associate security labels and markings with information exchanged between the enterprise systems and the control system?
19	413	Communication Protection	Is the use of VoIP authorized, monitored, and controlled?
20	416	Communication Protection	Are the system devices that collectively provide name/address resolution services for an organization fault tolerant?
21	417	Communication Protection	Does the use of secure name/address resolution services avoid adverse impacts to the operational performance of the system?
22	427	Communication Protection	Does the system enforce dynamic information flow control based on changing conditions or operational considerations?

23	431	Communication Protection	Does the system enforce defined one-way flows using hardware mechanisms (i.e., data diode)?
24	437	Communication Protection	Are automated or manual mechanisms (e.g., roles and responsibilities as defined by Active Directory) used as required to assist authorizing users in making the correct information sharing/collaboration decisions?
25	439	Communication Protection	Are communications limited to only the devices that need to communicate?
26	524	Configuration Management	Is the delivery and removal of system components limited, authorized, and recorded?
27	529	Configuration Management	Is there an inventory of systems and critical components and is it maintained?
28	534	Configuration Management	Is a baseline configuration for the development and test environments maintained and managed separately from the operational baseline?
29	540	Configuration Management	Are configuration changes tested, validated, and documented before installing them on the operational system, and has testing been ensured to not interfere with system operations?
30	546	Configuration Management	Is there physical security to restrict data devices, serial ports, network ports, USB, and secure digital memory card?
31	548	Configuration Management	Are the security settings configured to the most restrictive mode consistent with system operational requirements?
32	550	Configuration Management	Are exceptions from the mandatory configuration settings identified, documented, and approved based on explicit operational requirements?
33	551	Configuration Management	Are the configuration settings for all components of the system enforced?
34	552	Configuration Management	Are changes to the configuration settings monitored and controlled in accordance with policies and procedures?
35	557	Configuration Management	Has an inventory of the components of the system been developed, documented and maintained that accurately reflects the current system?
36	558	Configuration Management	Has an inventory list of the components of the system been developed, documented, and maintained that is consistent with the system boundary?
37	559	Configuration Management	Has an inventory list of the components of the system been developed, documented, and maintained that is at the level of granularity deemed necessary for tracking and reporting?
38	560	Configuration Management	Has an inventory of the components of the system been developed, documented, and maintained that includes defined information deemed necessary to achieve effective property accountability?
39	561	Configuration Management	Is the inventory of system components and programming updated as an integral part of component installation, replacement, and system updates?

40	562	Configuration Management	Are automated mechanisms used to help maintain an up-to-date, complete, accurate, and readily available inventory of system components, configuration files and set points, alarm settings and other required operational settings?
41	563	Configuration Management	Are automated mechanisms used to detect the addition of unauthorized components/devices/component settings into the system?
42	568	Configuration Management	Are critical digital assets (CDA) in security areas destroyed on removal from operations, or are they inspected and subject to an approved documented sanitization procedure on being removed from service (e.g., lifecycle plan)?
43	569	Configuration Management	Are all factory default authentication credentials changed on system components and applications upon installation?
44	570	Configuration Management	Does legacy equipment with known authentication deficiencies have compensatory access restrictions?
45	573	Configuration Management	Are the legacy components identified, tested, and documented to verify that the compensatory measures are effective?
46	642	Continuity	Is normal operation of the system resumed in accordance with its policies and procedures after a security event?
47	646	Continuity	Is the alternate storage site configured to facilitate timely and effective recovery operations?
48	647	Continuity	Are alternate command/control methods identified, and are agreements in place to permit the resumption of operations within a defined time period when the primary system capabilities are unavailable?
49	652	Continuity	Are necessary communications for the alternate control center identified, and are agreements in place to permit the resumption of system operations for critical functions within a defined time period when the primary control center is unavailable?
50	656	Continuity	Is the alternate control center fully configured to be used as the operational site supporting a minimum required operational capability?
51	663	Continuity	Are backup copies of the operating system and other critical system software stored in a separate facility or in a fire-rated container that is not collocated with the operational software?
52	511	Environmental Security	Is the emergency power shutoff protected from unauthorized activation?
53	512	Environmental Security	Is the emergency power-off capability protected from accidental and intentional/unauthorized activation?
54	513	Environmental Security	Is there a short-term uninterruptible power supply to be used for orderly system shutdown?
55	514	Environmental Security	Is there a long-term alternate power supply that is capable of maintaining minimally required operational capability?
56	515	Environmental Security	Is there a long-term alternate power supply that is self-contained and not reliant on external power generation?
57	517	Environmental Security	Are there fire suppression and detection devices/systems?

58	518	Environmental Security	Do fire detection devices/systems activate automatically and notify the organization and emergency responders in the event of a fire?
59	519	Environmental Security	Do fire suppression devices/systems provide automatic notification to the organization and emergency responders?
60	527	Environmental Security	Is the system power equipment and power cabling protected from damage and destruction?
61	528	Environmental Security	Are redundant power equipment and parallel power cabling paths provided for the system?
62	582	Incident Response	Are cyber and control system security incident information promptly reported to authorities?
63	177	Personnel	Are all required controls for employees terminated for cause completed within 24 hours?
64	179	Personnel	Are electronic and physical access permissions reviewed when individuals are reassigned or transferred?
65	180	Personnel	Are electronic and physical access permissions reviewed within 7 days when individuals are reassigned or transferred?
66	185	Personnel	Are periodic reviews of physical and electronic access conducted to validate terminated account access was removed?
67	470	Physical Security	Are lists of personnel with authorized access developed and maintained, and are appropriate authorization credentials issued?
68	471	Physical Security	Are the access list and authorization credentials reviewed and approved at least annually and those no longer requiring access removed?
69	472	Physical Security	Is physical access to the facility authorized based on position or role?
70	473	Physical Security	Are two forms of identification required to gain access to the facility?
71	474	Physical Security	Are physical access authorizations enforced for all physical access points to the facility?
72	475	Physical Security	Are individual access authorizations verified before granting access to the facility?
73	476	Physical Security	Is entry to the facility controlled by physical access devices and/or guards?
74	477	Physical Security	Are the areas officially designated as publicly accessible controlled in accordance with the organization's assessment of risk?
75	478	Physical Security	Are keys, combinations, and other physical access devices secured?
76	479	Physical Security	Are physical access devices inventoried on a periodic basis?
77	480	Physical Security	Are combinations and keys changed on a defined frequency, and when keys are lost, combinations compromised, or individuals are transferred or terminated?
78	481	Physical Security	Is physical access to distribution and communication lines controlled and verified?

79	482	Physical Security	Is physical access to output devices controlled?
80	483	Physical Security	Is physical access to the system controlled independently of the facility access controls?
81	484	Physical Security	Are security checks at physical boundaries performed for unauthorized removal of system components?
82	485	Physical Security	Is every physical access point to the facility guarded or alarmed and monitored 24 hours per day, 7 days per week?
83	486	Physical Security	Are lockable physical casings used to protect internal components of the system from unauthorized physical access?
84	487	Physical Security	Is physical access monitored to detect and respond to physical security incidents?
85	489	Physical Security	Are results of reviews and investigations coordinated with the organization's incident response capability?
86	490	Physical Security	Are real-time physical intrusion alarms and surveillance equipment monitored?
87	491	Physical Security	Are automated mechanisms used to recognize potential intrusions and initiate designated response actions?
88	492	Physical Security	Is physical access controlled by authenticating visitors before authorizing access?
89	493	Physical Security	Are visitors escorted and monitored as required in the security policies and procedures?
90	494	Physical Security	Are two forms of identification required for access?
91	495	Physical Security	Are visitor access records maintained, and are all physical access logs retained for as long as required by regulations or per approved policy?
92	496	Physical Security	Do visitor records include name and organization of the person visiting?
93	497	Physical Security	Do visitor records include the signature of the visitor?
94	498	Physical Security	Do visitor records include a form of identification?
95	503	Physical Security	Are automated mechanisms employed to facilitate the maintenance and review of access records?
96	504	Physical Security	Is cryptographic hardware protected from physical tampering and uncontrolled electronic connections?
97	505	Physical Security	Are all external system and communication connections identified and protected from tampering or damage?
98	506	Physical Security	Are asset location technologies used to track and monitor the movements of personnel and vehicles to ensure they stay in authorized areas?
99	507	Physical Security	Are asset location technologies used to identify personnel needing assistance?
100	508	Physical Security	Are asset location technologies used to support emergency response?

101	509	Physical Security	Is hardware (cages, locks, cases, etc.) used to detect and deter unauthorized physical access to system devices?
102	510	Physical Security	Is the ability to respond to an emergency not hindered by using tamper-evident hardware?
103	300	Portable / Mobile / Wireless	Are usage restrictions and implementation guidance established for organization-controlled mobile devices?
104	315	Portable / Mobile / Wireless	Is authentication and encryption used to protect wireless access to the system and the latency induced does NOT degrade the operational performance of the system?
105	324	Portable / Mobile / Wireless	Is peer-to-peer wireless networking capability disabled except for explicitly identified components in support of specific operational requirements?
106	293	Remote Access Control	Is remote access for privileged commands and security-relevant information authorized only for compelling operational needs and is the rationale for such access documented?
107	294	Remote Access Control	Is Bluetooth wireless networking capability disabled except for explicitly identified components in support of specific operational requirements?
108	379	Software	Are system components used that have no writable storage that is persistent across component restart or power on/off cycles?
109	449	System Integrity	Does the use of automated flaw remediation processes NOT degrade the operational performance of the system?
110	455	System Integrity	Is the correct operation of security functions verified upon system startup and restart, upon command by user with appropriate privilege, periodically, and at defined time periods?
111	463	System Integrity	Is tamper-evident packaging used during transportation from vendor to operational site, during operation, or both?
112	469	System Integrity	Is the output from the system handled and retained in accordance with applicable regulations, standards, and organizational policy as well as operational requirements?
113	332	System Protection	Are the operational system boundary, the strength required of the boundary, and the respective barriers to unauthorized access and control of system assets and components defined?
114	336	System Protection	Does the system design and implementation protect the integrity of electronically communicated information?
115	339	System Protection	Does the use of public key certificates avoid degrading (i.e., latency) the operational performance of the system?
116	351	System Protection	Has legacy equipment been updated with current or custom developed system components?
117	197	Training	Are simulated events incorporated into continuity of operations training to facilitate effective response by personnel in crisis situations?

PROBLEM 2: Novel Collaborative Learning Model for Cyberattack Detection Systems in IoT Networks

Introduction

In recent years, the rapid development of various technologies, such as 5G/6G, Industry 4.0, and Internet-of-Things (IoT), has enabled numerous applications to become an integral part in many aspects of our daily lives. However, such ever-fast growth has also led to an unprecedented massive amount of data and the proliferation of interconnected devices, e.g., sensors, smart cars, and cameras, which raises serious security and privacy concerns. Particularly, the increasing number of emerging applications has also brought forth many new types of cyberattacks. For example, the number of new (zero-day) cyberattacks has increased by 60% from 2018 to 2019 [32]. Besides the dire consequences to the economy, e.g., ransomware alone cost more than $5 billion globally in 2017 [33], cyberattacks pose serious threats to other areas with highly sensitive information such as healthcare and public security. As a result, cyberattack detection methods play a key role in detecting and promptly preventing consequences of cyberattacks in future IoT networks.

Recently, with outstanding classification ability, Machine Learning (ML) techniques, especially deep learning (DL), have been widely applied for cyberattack detection problems. Particularly, DL models can effectively learn the signatures of various cyberattack types. Moreover, DL models even can detect new types of attacks that have never been learned/trained before [34]. Nevertheless, DL-based cyberattack detection systems are also facing some practical challenges. Particularly, conventional DL approaches usually require a huge amount of data to achieve a high performance. However, in many applications, data are very difficult to collect because they are often stored locally on user devices such as IoT devices, smartphones, and wearable devices. This poses a threat to user privacy because sensitive data (e.g., location and private information) have to be sent over the network and stored at the centralized server for processing. Besides the privacy concerns, transmitting such a collectively large amount of data also imposes an extra communication burden over the network. Consequently, these limitations have been hindering the effectiveness of DL techniques in cyberattack detection systems.

To address these problems, Federated Learning (FL) has emerged to be a highly effective solution. Unlike conventional DL techniques that collect data and train the global model at a central server, FL enables the learning process to be distributed across all devices. Particularly, instead of sending data to a central server, the local data can be used to train a global model locally on each user device. Then, the obtained model weights of each device are periodically sent to a central server for aggregation. Afterward, the aggregated weights are sent back to all devices to update their local models' weights. Since only the weights are transmitted in FL, both the privacy and communication overhead issues can be mitigated [35].

Despite its effectiveness, FL is still facing some challenges. Particularly, FL only performs well if the training data and the predicting data are independent and identically distributed (i.i.d). Consequently, they are not robust to the changes in the system, e.g., changes in network traffic due to the mobility of users, new types of devices participating in the network, and so on. Moreover, the performance of FL largely relies on the availability of labeled data. However, acquiring sufficient labeled data might be costly and time-consuming. Even if the data are available, the participated user data usually have different structures such as features. This leads to difficulty or even mistakes when FL aggregates the global model. Consequently, they may not be suitable for the intensive training process of FL [35] [36].

To address these limitations, transfer learning (TL) has been emerging as a promising solution, especially for problems related to heterogeneous training data [37–39]. Unlike DL and FL techniques that are trained only for a specific problem, TL can utilize "knowledge" from rich resource data to enhance the training process and performance of the ML models. Particularly, by transferring "knowledge" from similar scenarios with a lot of high-quality data, TL can address the lack of labeled data for the target networks. Moreover, the TL can exchange "knowledge" even if the data features of the target and source networks are not very similar [37, 40]. However, if the data features are too different, TL might even make the learning process worse than that without using TL, i.e., negative transfer [37–39]. In the context of cyberattack detection for IoT networks, negative transfer might be a serious problem since different networks may have various types of devices generating different data.

In this work, we propose a novel collaborative learning framework that utilizes the strengths of both TL and FL to address the limitations of conventional DL-based cyberattack detection systems. Particularly, we consider a scenario with two different IoT networks[2]. The first network (source network) has an abundant labeled data resource, while the second network (source network) has very little data resource (and most of them are unlabeled). Here, unlike most of the current works that assume that the data at these networks have the same features [41], we consider a much more practical and general case in which data at these two networks may have different features. To address the problem of dissimilar feature spaces of the target and source networks, we propose to transform them into a new joint feature-space. In this case, at each learning round of the federated learning process, trained models of target and source networks can be exchanged through the joint feature-space. Thus, by periodically exchanging and updating the trained model, the target network can eventually achieve the converged trained deep neural network that can predict attacks with high accuracy (thanks to useful "knowledge" transferred from the source network). Besides the exchanging and updating the learning model iteratively, we use a small number of mutual samples between two networks to mitigate the negative

[2]The cases with multiple networks can be straightforwardly extended, e.g., by scheduling for networks to exchange information in order.

transfer learning. More importantly, unlike FL where networks try to train a joint global model, our proposed framework enables the participating networks to obtain their particular trained models that are specific to their networks, i.e., better predict attacks for particular networks with different data structures. Extensive experiments on recent real-world datasets, including N-BaIoT [42] [43], KDD [44], NSL-KDD [45] and UNSW [46] show that our proposed framework can achieve an accuracy of up to 99% and an improvement of up to 40% over the unsupervised learning approach. The main contributions of this work can be summarized as follows:

- We propose a novel collaborative learning framework that can effectively detect cyberattacks in decentralized IoT systems. By combining the strengths of FL and TL, our proposed framework can improve learning efficiency and the accuracy of cyberattack detection in comparison with the conventional DL-based cyberattack detection systems.
- We propose an effective transfer learning approach that can allow the deep learning model from the rich-data network to transfer useful knowledge to the low-data network even they have different features for cyberattack detection in IoT networks.
- We perform extensive experiments on recent real-world datasets including N-BaIoT, KDD, NSL-KDD, and UNSW to evaluate the performance of the proposed collaborative learning framework. The results show that our proposed approach can achieve an accuracy of up to 99% and an improvement of up to 40% over the unsupervised learning approach.

Related work

Deep Learning for Cyberattack Detection

There have been a rich literature proposing DL approaches for cyberattack detection. In [47], a deep neural network (DNN) model is developed to detect zero-day attacks based on two types of data, i.e., network activities and local system activities. The results show that for most of the datasets, the proposed DNN can achieve a higher detection accuracy and lower false-positive rate compared to those of the other conventional machine learning classifiers such as K-Nearest Neighbors (KNN), and Support Vector Machine (SVM). Another DL approach is proposed in [48] to detect cyberattacks in the mobile cloud computing environments. The main difference between [47] and [48] is that the approach in [48] consists of a feature analysis phase before the learning phase. In the analysis phase, the datasets are analyzed to identify meaningful features, thereby reducing the data dimension and computational complexity. Experiments on the KDD [44], NSL-KDD [45], and the UNSW [46] datasets show that the proposed approach can achieve a detection accuracy of up to 97.1%.

Federated Learning for Cyberattack Detection

With the advent of FL, the research focus has recently shifted towards applying this framework for cyberattack detection, especially in environments with numerous devices such as IoT and mobile edge networks. In [49], an FL framework is proposed for cyberattack detection in an edge network. In this network, the data for intrusion detection are stored locally at each edge node. The edge nodes train their data locally and send their models' weights to an FL server for aggregating. After aggregation, the FL server sends the weights back to all edge nodes. In this way, each edge node can benefit from the other nodes' data and training while protecting its privacy and reducing the network's communication burden. Experiments with the NSL-KDD datasets show that the proposed approach can achieve an accuracy of up to 99.2%. Another FL approach is proposed in [50] for attack detection in industrial cyber-physical systems. In the considered setting, there are multiple cyber-physical systems acting as FL nodes. However, unlike the previous frameworks, the authors propose a novel architecture combining a convolution neural network (CNN) and a gated recurrent unit for training at each FL node. Experiments with self-collected data show that the proposed approach can outperform other state-of-the-art approaches, e.g., [51–53], with an accuracy up to 99.2%. However, because of the limitations of FL as presented in the previous section, the learning model can only combine data with the same features and labels.

Transfer Learning for Cyberattack Detection

Although FL techniques can effectively address the privacy and communication load concerns of conventional ML for cyberattack detection, they are still facing some challenges. Particularly, FL approaches usually require high-quality and labeled data for training. However, collecting and labeling such data is expensive and time-consuming, especially for large-scale systems. On the other hand, unlabeled data are often abundant in environments such as IoT and mobile edge networks. Thus, a deep TL approach is proposed for IoT intrusion detection in [54] based on network activities, which can utilize both labeled and unlabeled data. In this approach, the authors employ two AEs. The first AE is trained with labeled data, while the second AE is trained with unlabeled data. Then, the knowledge is transferred from the first AE to the second AE by minimizing the Maximum Mean Discrepancy (MMD) distances between their weights. Experiments over nine IoT datasets were conducted to show that the proposed approach can achieve higher Area Under the Curve (AUC) scores compared to those of several other approaches.

Besides analyzing network traffic, another approach to detect cyberattacks is to analyze the devices' fingerprints. Particularly, attackers may try to impersonate a device in the system by copying its signal. For this kind of attack, ML techniques can be used to detect if the signals are coming from the real device or the malicious device. TL approaches

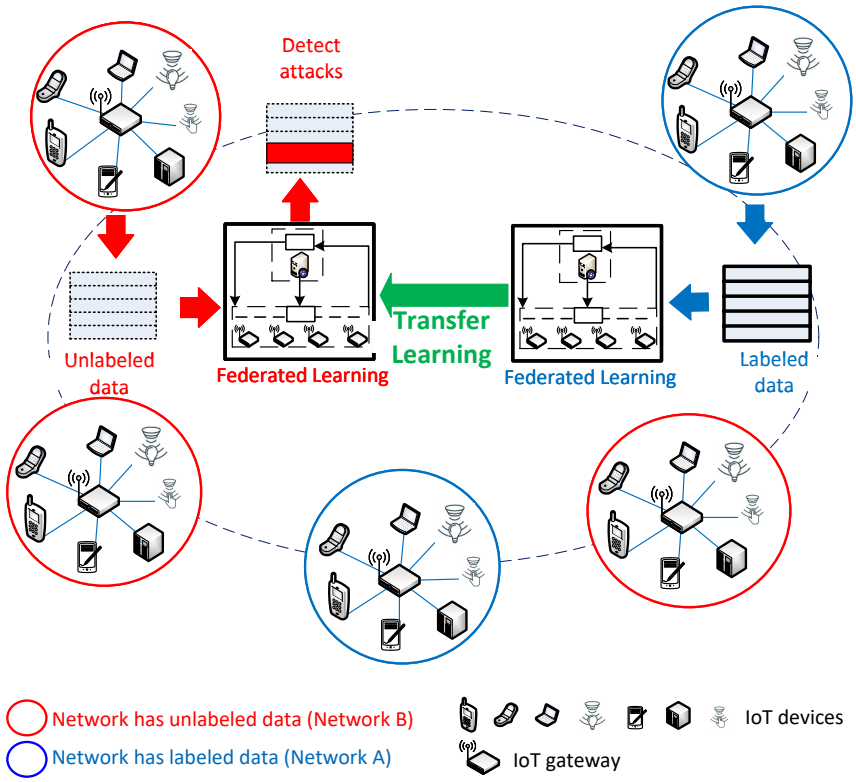


Figure 14: Illustration of a system model for cyber attack detection in IoT networks.

such as [55–58] are proposed to identify cyberattacks based on device fingerprints. Among them, [57] and [58] leverage the environmental effects to classify signals from devices. To improve the classification accuracy and address the lack of data, these approaches transfer the knowledge from nearby devices (since they share similar environmental effects). On the other hand, [55] and [56] leverage the knowledge from previous experiences, i.e., data collected in the past. These past data are then combined with the current data for training, thereby addressing the lack of fingerprint data.

Unlike all the abovementioned approaches, the collaborative learning framework proposed in this work can leverage the strengths of both FL and TL to address limitations of ML-based intrusion detection systems, e.g., lack of labeled data, privacy and heterogeneous data feature space. Moreover, in our approach, each IoT network has a separated model that is fine-tuned specifically for that network, therefore the model is more effective for that network's cyberattack detection compared to FL frameworks with a single model for all networks. Furthermore, our proposed system model can utilize knowledge from both source and target data in the network instead of only transferring knowledge from a single source as proposed in most of the mentioned TL frameworks [54–56, 59], thereby mitigating the negative transfer problem.

Proposed Federated Transfer Learning Framework for Cyberattack Detection in IoT Networks

System Model

The conventional FL model requires to use a centralized server to maintain and aggregate all the trained models in the whole learning process. However, this may lead to a high cost to maintain and may not be effective to deploy in IoT networks. Thus, in this work, we propose a federated transfer learning model that allows the learning process to be performed more flexibly and effectively in IoT environments. In particular, we consider a network which has unlabeled data (e.g., Network B as illustrated in Fig. 14), and it wants to learn more knowledge from other networks with abundant labeled data. In this case, this network will connect with a target network (e.g., Network A as illustrated in Fig. 14) and nominate itself as a centralized node which can train its own data as well as perform transfer learning to exchange knowledge with the target network.

We denote a labeled cybersecurity dataset $D_A = \{X^A, Y^A, F^A\}$ of Network A with $(X^A, Y^A) = \{x_1^A, y_1^A, x_2^A, y_2^A, ..., x_{M_A}^A, y_{M_A}^A\}$ where M_A is the number of samples of dataset A. In contrast, Network B has an unlabeled cybersecurity dataset $D_B = \{X^B, F^B\}$ with $(X^B) = \{x_1^B, x_2^B, ..., x_{M_B}^B\}$ where M_B is the number of samples of dataset B. F^A, F^B are the feature spaces of Network A and Network B, respectively. The proposed model will perform transfer learning between two neural network by minimizing the total loss J to predict the label $P(z^B)$ for the unlabeled dataset of Network B. In this way, the network can help to improve the accuracy in identifying network traffics by learning useful knowledge from other labeled networks. Each network can be managed by an IoT gateway and possesses its own private dataset. The IoT gateway uses its deep learning model to detect normal and abnormal traffics. It is important to note that, unlike conventional FL approaches [60], in this work, we consider a practical scenario in which the datasets of networks may have different features.

Proposed Federated Transfer Learning Approach for Cyberattack Detection

In this section, we propose a highly-effective federated transfer learning model that can exchange knowledge between an unlabeled network and multiple networks which may have different features. To better analyze the impact of our proposed approach, we consider a specific scenario in which one labeled network is used as a source network to support an unlabeled network (i.e., target network). The scenario with one unlabeled network and multiple labeled networks can be straightforwardly extended, and we leave it for future study. Fig. 15 describes the training and predicting processes of FTL algorithm that we use in this case. The table of notations is presented in Table 14. As described in previous section, Network A, Network B have their dataset D^A, D^B, respectively. They also have their model parameters called W^A and W^B. The outputs of two neural networks

Table 14: The table of notations.

Notation	Description
X	The total samples of a dataset
Y	The labels of a dataset
F	The feature space of a dataset
x	A dataset sample
D	Network
M_A, M_B	The number of samples of dataset A, B, respectively
M_C	The number of predicted labels
M_{AB}	The overlapping samples between dataset A and dataset B
W_A, W_B	The parameter matrices of models A and B, respectively
Z_A, Z_B	The outputs of models A and B, respectively
z	The output of an input sample after learning model
j	The loss of an input sample
J	The loss function
γ, λ	Weight parameters
w	Trainning parameters

are calculated as follows:

$$Z^A = W^A * X^A, \ Z^B = W^B * X^B. \tag{6}$$

We need to find the prediction function $P(z_j^B) = P(z_1^A, y_1^A, \ldots, z_{M_A}^A, y_{M_A}^A, z_j^B)$ to predict the output of Network B. To find a high-quality predict function, we first need to minimize the loss function using the labeled dataset as follows:

$$\arg \min_{W^A, W^B} J^B = \sum_{i}^{M_c} j^B(y_i^A, P(z_i^B)), \tag{7}$$

where M_c is the number of predicted labels, and j^B represents the loss of the loss function which depends on the type of output or mechanism, i.e., the logistic loss function [61] with the predicted value $\mathbf{z}$ and the labeled $\mathbf{y}$:

$$j^B(\mathbf{z}, \mathbf{y}) = \log\big(1 + \exp(-\mathbf{z} \times \mathbf{y})\big). \tag{8}$$

In addition, datasets A and B may have some overlapping samples, and thus we can use these samples to optimize the loss function. We denote M_{AB} as the overlapping samples between dataset A and dataset B. We need to minimize the alignment loss function

Algorithm 1 Federated Transfer Learning Algorithm: Training Process

Input: The learning rate η, the weight parameter γ, λ, the maximum iteration T, the tolerance t and Network A and Network B initialize model parameters W^A, W^B;
Output: The trained model parameter W^A, W^B;
$iteration = 0$
while $iteration \leq T$ **do**
  Network A performs:
  $z_i^A = h_i^A * x_i^A$ for $i \in D_A$;
  Send $\{z_i^A, y_i^A\}$ to Network B;
  Network B performs:
  $z_i^B = h_i^B * x_i^B$ for $i \in D_B$;
  Send $\{z_i^B\}$ to Network A;
  Network A performs:
  Compute $\frac{\partial J}{\partial w_i^A}$ and J^A, then send them to Network B;
  Network B performs:
  Compute $\frac{\partial J}{\partial w_i^B}$, J^B and J^{AB}, then send them to Network A;
  Network A performs:
  Update $w_l^A = w_l^A - \eta \frac{\partial J}{\partial w_i^A}$;
  Network B performs:
  Update $w_l^B = w_l^B - \eta \frac{\partial J}{\partial w_i^B}$;
  if $J_{prev} - J \leq t$ **then**
    Send stop signal to Network B;
    Break.
  else
    $J_{prev} = J$;
    $iteration = iteration + 1$;
    continue;
  end if
end while

between A and B as follows:

$$\underset{W^A, W^B}{\operatorname{argmin}} J^{AB} = -\sum_{i}^{M_{AB}} j^{AB}(z_i^A, z_i^B), \tag{9}$$

where j^{AB} represents the alignment loss function. The common alignment loss function can be represented in modulus $j^{AB} = ||z_i^A - z_i^B||^2$ or angle $j^{AB} = -z_i^A * z_i^B$. Lastly, we add the regularization $J_R^A = \sum_l^{L_A} ||w_l^A||^2$ and $J_R^B = \sum_l^{L_B} ||w_l^B||^2$ in which L_A and L_B are the numbers of layers in neural Network A and Network B, respectively, to find the final loss function that needs to be minimized:

$$\underset{W^A, W^B}{\operatorname{argmin}} J = J^B + \gamma J^{AB} + \frac{\lambda}{2}(J_R^A + J_R^B), \tag{10}$$

Algorithm 2 Federated Transfer Learning Algorithm: Predicting Process

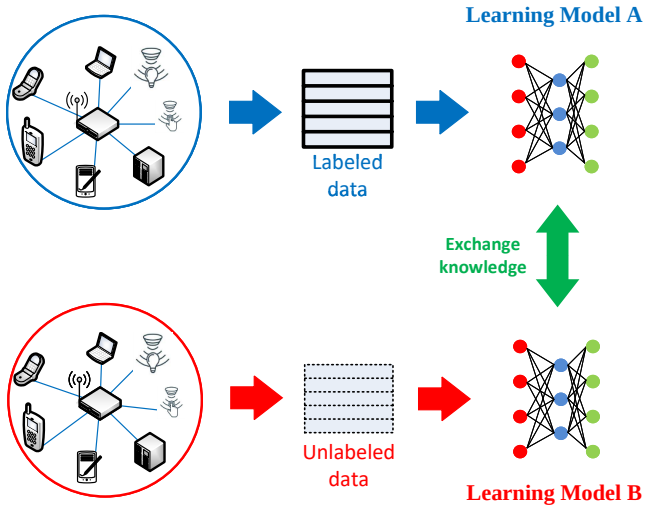
1: **Input:** The model parameters W^A, W^A and dataset X_B;
2: **Output:** The prediction Y^B;
3: Network B performs:
4: $z_i^B = h_i^B * x_i^B$ for $i \in D_B$;
5: Send $\{z_i^B\}$ to Network A;
6: Network A performs:
7: Compute $P(z_i^B) = W^A[z_i^B]$ and send it to Network B.

where γ and λ are the weight parameters. The gradient for updating W^A, W^B are calculated by the following formula:

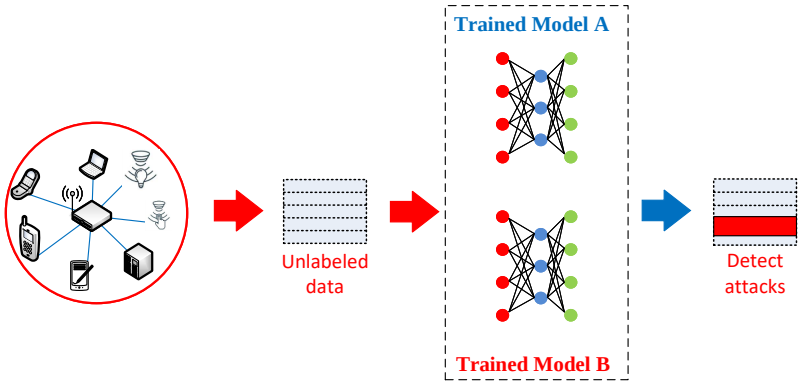
$$\frac{\partial J}{\partial w_l^i} = \frac{\partial J^B}{\partial w_l^i} + \gamma \frac{\partial J^{AB}}{\partial w_l^i} + \lambda w_l^i. \tag{11}$$

The training process is presented in Algorithm 1. Specifically, we first initialize W^A and W^B. Next, we calculate z_i^A and z_i^B from the input samples of dataset A (D^A) and dataset B (D^B) as shown in Equation (6). Then, Network A sends $\{z_i^A, y_i^A\}$ to Network B to calculate J^B, the alignment loss function J^{AB} and the gradients of J^B as shown in Equations (7), (9), (10) and (11), respectively. Similarly, Network B sends $\{z_i^B\}$ to Network A to calculate J^A as in Equation (10). In Equation (9), we use M_{AB} as the mutual samples of two datasets. For example, the same IoT devices are attacked by the same types of cyberattacks in different networks. Each network extracts the attack data with different features, e.g., Network A uses timeslot, packet header, ip address while Network B uses MAC address, error packets, frame header. The number of mutual samples is an important factor that strongly supports the learning process between two networks (we will explain it more details in Section (1)). After that, we calculate the final loss function J and the gradient as in Equation (10) and Equation (11). Finally, Network A and Network B update their model parameters based on the gradient and loss functions. This process continuously repeats until the system converges or reaches the maximum number of iterations to minimize the final loss function in Equation (10).

When the training completes, the prediction process described in the Algorithm 2 is called to predict the final result of the unlabeled dataset D_B. In this process, both Network A and Network B have their trained models. Similar to the training process, the dataset D_B firstly goes through the trained model of Network B to calculate Z^B. Then, Network B sends Z^B to Network A to archive the transfer learning knowledge from trained model of Network A. Network A predicts the results and sends them back to Network B to classify the attack and normal behaviors of the network.



(a) FTL training process.



(b) FTL predicting process.

Figure 15: The FTL algorithm.

Evaluation Methods

As mentioned in [62, 63], the confusion matrix is typically used to evaluate system performance, especially for intrusion detection systems. We denote TP, TN, FP, and FN to be "True Positive", "True Negative", "False Positive", and "False Negative", respectively. The Receiver Operator Characteristic (ROC) is created by plotting the TPR over FPR at different thresholds. Then, we use Area Under the Curve (AUC) to evaluate the performance of the algorithm in the following formulas:

$$\xi = \int_{x=0}^{1} \mathrm{TP}(\mathrm{FP}^{-1}(x))\, dx. \tag{12}$$

In our experiments, we randomly select samples from original dataset to test the algorithm. In this scenario, the p is often used to evaluate the resuts of random tests and is given by

$$p = F(\xi|\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{\xi} \mathrm{e}^{\frac{-(t-\mu)^2}{2\sigma^2}} \mathrm{d}\xi, \tag{13}$$

in which μ is the mean and σ is the standard deviation. The results are calculated by the significant number with the following formula:

$$Sig = F^{-1}(p|\mu,\sigma) = \{\xi : F(\xi|\mu,\sigma) = p\}, \tag{14}$$

where Sig is the significant number that represents the results of 30 random runs and the confidence of this number is calculated by $conf = 1 - p$. In a normal situation, the p is considered confidence when it has values around 0.01 and 0.05, corresponding to the confidence of significant numbers around 99% and 95%.

Data and Experiments

Datasets

In this experiment, we use four popular cybersecurity datasets, namely KDD [44], NSL-KDD [45] UNSW [46] and N-BaIoT [42] [43], to evaluate the performance of the proposed method. The Network-based Detection of IoT Botnet Attacks (N-BaIoT) dataset includes the information collected in the setup network about the normal and attack situation. The attack was performed by servers to nine IoT devices and the total network behavior was captured by the sniffer server to extract dataset. This dataset is characterized by 115 features for both normal and attack behaviors. In this dataset, the attack type is the Distributed Denial of Service (DDoS) which was implemented by two well-known botnets, namely Mirai and BASHLITE. The BASHLITE botnet includes 5 types of attacks, i.e., network scanning (scan), spam data sending (junk), UDP flooding (udp), TCP flooding (tcp), and the join of sending spam data and opening port to specific IP address (combo). Besides BASHLITE, the Mirai botnet also includes 5 types of attacks, i.e., scan, ACK flooding (ack), SYN flooding (syn), udp, and optimized UDP flooding (udpplain).

In addition to IoT datasets, we also want to evaluate our proposed solution on some classical intrusion detection datasets, i.e., KDD [44], NSL-KDD [45] and UNSW [46] datasets. The KDD dataset [44] includes many different kinds of network attacks simulated in military network environment. The KDD dataset has 41 features and it classifies attacks into 4 groups including Denial of Service (DoS), Probe, User to Root (U2R), Remote to Local (R2L). The NSL-KDD dataset [45] inherits the properties from KDD [44] dataset such as the features and types of attacks but eliminates the redundant samples

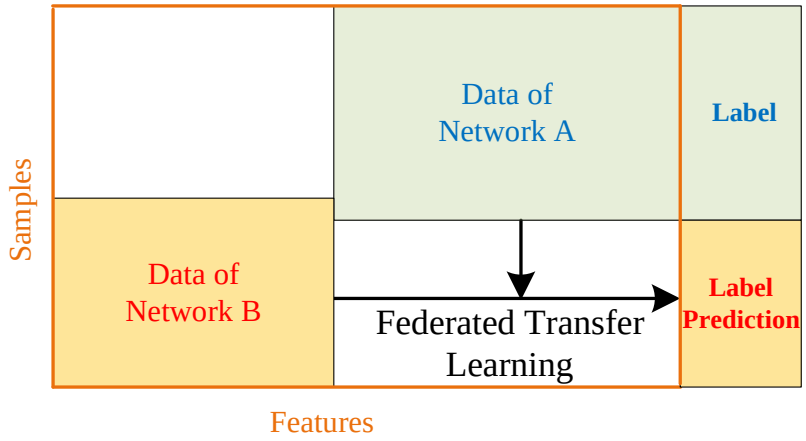


Figure 16: The data of participated networks used in this experiment.

Table 15: The results with multiple datasets in CASE 1.

(a) The results with p=1.

	FTL	UDL
IoT1	**85.771**	45.753
IoT2	**83.795**	63.171
IoT3	**94.286**	80.453
IoT4	**79.241**	77.885
IoT5	**90.605**	81.876
IoT6	**91.179**	82.703
IoT7	**90.670**	85.183
IoT8	**82.960**	65.256
IoT9	**83.222**	73.072
KDD	**99.315**	80.477
NSLKDD	**98.485**	83.025
UNSW	**97.072**	68.449

(b) The results with p=3.

	FTL	UDL
IoT1	**87.398**	49.770
IoT2	**85.672**	65.793
IoT3	**94.896**	81.070
IoT4	**81.672**	77.885
IoT5	**91.517**	82.013
IoT6	**92.059**	82.703
IoT7	**92.030**	86.013
IoT8	**85.197**	68.161
IoT9	**85.072**	73.078
KDD	**99.395**	81.304
NSLKDD	**98.534**	83.450
UNSW	**97.141**	69.124

(c) The results with p=5.

	FTL	UDL
IoT1	**88.259**	51.897
IoT2	**86.666**	67.181
IoT3	**95.220**	81.397
IoT4	**82.959**	77.885
IoT5	**92.000**	82.085
IoT6	**92.525**	82.703
IoT7	**92.750**	86.453
IoT8	**86.381**	69.700
IoT9	**86.052**	73.082
KDD	**99.438**	81.742
NSLKDD	**98.561**	83.675
UNSW	**97.177**	69.482

in the training dataset and the duplicated samples in the testing dataset. Although both KDD and NSL-KDD datasets are well-known and used in many research works, they were developed long time ago. Thus, some modern attacks were not involved. Therefore, a recent dataset, i.e., UNSW dataset [46], is considered in this work. Unlike KDD and NSL-KDD, the feature space of this dataset includes 42 types and 9 kinds of attacks, namely DoS, Backdoors, Worms, Fuzzers, Analysis, Reconnaissance, Exploits, Shellcode, and Generic.

Experiment Setup

In this section, we carry out experiments using all the aforementioned datasets to evaluate the performance of the proposed solution. In this experiment, we denote IoT1-9

Table 16: Dataset preparation

Dataset	Device name	Features of Network A	Features of Network B	Total features
IoT1	Danmini_Doorbell	85	30	115
IoT2	Ecobee_Thermostat	85	30	115
IoT3	Ennio_Doorbell	85	30	115
IoT4	Philips_B120N10_Baby_Monitor	85	30	115
IoT5	Provision_PT_737E_Security_Camera	85	30	115
IoT6	Provision_PT_838_Security_Camera	85	30	115
IoT7	Samsung_SNH_1011_N_Webcam	85	30	115
IoT8	SimpleHome_XCS7_1002_WHT_Security_Camera	85	30	115
IoT9	SimpleHome_XCS7_1003_WHT_Security_Camera	85	30	115
KDD	-	31	10	41
NSLKDD	-	31	10	41
UNSW	-	31	11	42

as the dataset names of nine IoT devices. Table 16 describes the total features and the representative names of datasets that we use in this experiment. Fig. 16 also describes the separated data in each dataset in this experiment. In this experiment, the participated data are randomly selected from the dataset. Then, the selected data are separated into label data (data of Network A) and unlabeled data (data of Network B) with different features as described in Table 16. These data have about 10% mutual samples of total dataset samples. We experiment with two cases, i.e., the first one is with 2000 unlabeled data and 9577 labeled data (CASE 1), the second one is with 10000 unlabeled data and 47893 labeled data (CASE 2).

In this setup, we consider a baseline solution with the state-of-the-art unsupervised deep learning model (UDL) which clusters the unlabeled data into normal and attack behaviors based on autoencoder and k-means techniques [64]. The unsupervised deep learning model includes an autoencoder and k-nearest neighbor to cluster the unlabeled data. In addition, we consider the second baseline solution that uses both supervised and unsupervised datasets to feed the FTL learning models. The FTL will exchange the knowledge from the supervised learning model and the unsupervised learning model to improve the accuracy of learning as well as increase the precise of identifying attack and normal behaviors of the unlabeled data. Then, we measure the AUC of this process 30 times to calculate the signification number of the AUC series results with both baseline solutions. Finally, we plot the reconstruction errors to analyze the convergence of the FTL algorithm for all datasets.

Experimental Results

In this section, we show the results of our experiments with different kinds of cyber-security datasets.

Table 17: The results with multiple datasets in CASE 2.

(a) The results with p=1.

	FTL	UDL
IoT1	**90.371**	49.783
IoT2	**68.193**	62.591
IoT3	**94.525**	83.411
IoT4	**87.050**	77.725
IoT5	**86.535**	81.954
IoT6	**87.214**	82.555
IoT7	**97.662**	79.517
IoT8	**84.609**	52.702
IoT9	**90.095**	63.803
KDD	**99.535**	84.333
NSLKDD	**98.858**	81.164
UNSW	**97.049**	66.329

(b) The results with p=3.

	FTL	UDL
IoT1	**91.497**	54.079
IoT2	**72.573**	65.681
IoT3	**95.073**	83.565
IoT4	**88.538**	77.781
IoT5	**88.150**	82.160
IoT6	**88.638**	82.664
IoT7	**97.928**	81.400
IoT8	**86.691**	57.318
IoT9	**90.959**	65.559
KDD	**99.562**	84.423
NSLKDD	**98.885**	81.976
UNSW	**97.121**	66.901

(c) The results with p=5.

	FTL	UDL
IoT1	**92.093**	56.354
IoT2	**74.892**	67.317
IoT3	**95.363**	83.647
IoT4	**89.326**	77.811
IoT5	**89.006**	82.269
IoT6	**89.392**	82.721
IoT7	**98.069**	82.397
IoT8	**87.793**	59.763
IoT9	**91.417**	66.489
KDD	**99.576**	84.471
NSLKDD	**98.900**	82.406
UNSW	**97.159**	67.203

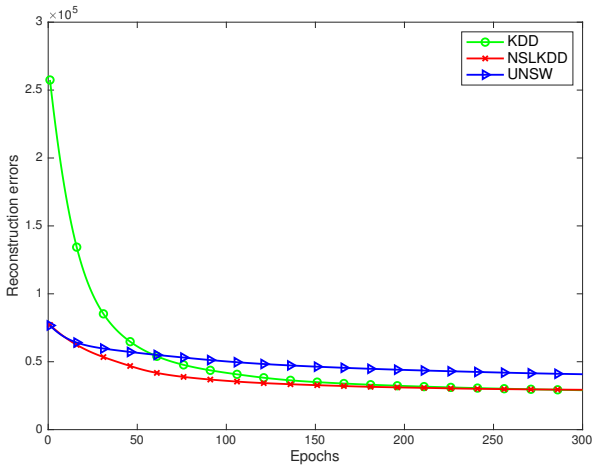
Accuracy Comparison

In this section, we compare the performance of FTL and the unsupervised deep learning (UDL) method in terms of the significant number of each p as explained in Section (1). Tables 15 and 17 describe the significant number of each dataset with $p = 1, 3, 5$ corresponding to the confidence of $99\%, 97\%, 95\%$.

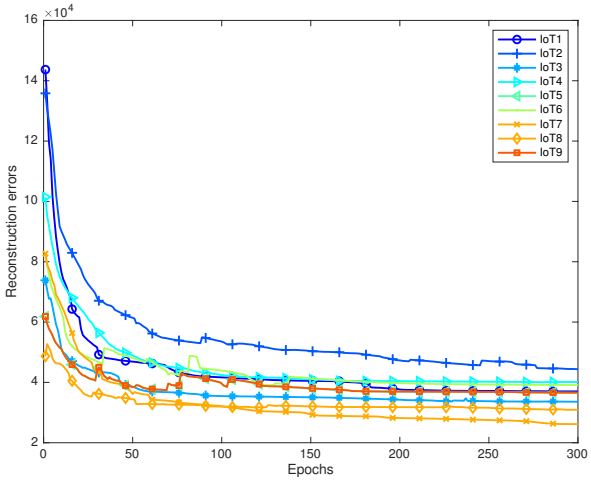
In general, Table 15 and Table 17 show that the significant numbers of all datasets increase as p increases. This is because in (14), we calculate the significant number based on a series of 30 continuous AUC results. When p increases, the AUC results increase in all tables. This demonstrates that most of the AUC results in 30 series are higher than the significant number in the case where $p = 1$.

Table 15(c) shows the significant numbers of participated datasets with $p = 5$ in CASE 1. In this table, the IoT1 and UNSW datasets show a significant gap of about 30% and 40% between FTL and UDL. These results show the difficulty of clustering in recognizing the groups of samples and the advantage of collaborative learning in these datasets. The other ten datasets have gaps of around 10-20% between the two methods, which demonstrate the stability of our proposed solution for any cybersecurity dataset.

In addition, Table 17(c) shows the significant numbers of multiple datasets with $p = 5$ in CASE 2. In this table, the significant numbers also have a gap of around 10-40% between the two solutions. It shows the common trend that the significant numbers increase for most datasets when the number of samples increases. However, in IoT2, IoT5, and IoT6 datasets, the significant numbers slightly decrease because of the randomly selected samples from the original dataset. It also can be demonstrated by the high fluctuation of the reconstruction errors of IoT2, IoT5, IoT6 datasets in Fig. 18(b) compared with other datasets. However, in all studied datasets, our proposed solution still



(a) KDD, NSLKDD, and UNSW datasets



(b) IoT datasets

Figure 17: Reconstruction errors in CASE 1.

performs much better than the state-of-the-art UDL solution. These results demonstrate that our solution can work efficiently in all IoT and conventional cybersecurity datasets in detecting cyberattacks in the network.

Reconstruction Error Analysis

In this section, we discuss the convergence of the FTL algorithm in each dataset. Fig. 17 describes the reconstruction errors of the nine IoT datasets and the conventional datasets like KDD, NSLKDD, and UNSW in CASE 1. Fig. 18 describes the reconstruction errors of study datasets in CASE 2.

In Fig. 17(a) and Fig. 18(a), we can see that at the first few epochs, the errors are

very high for KDD (up to 2.6×10^5 in CASE 1 and 12×10^5 in CASE 2), but this error dramatically reduces to 0.3×10^5 in CASE 1 and 1.5×10^5 in CASE 2 after only 200 epochs. For NSLKDD and UNSW, they have very similar trends with 0.75×10^5 in CASE 1 and 3.8×10^5 in CASE 2 at the beginning and gradually reduce to 0.4×10^5 in CASE 1 and 1.9×10^5 in CASE 2 after 200 epochs, respectively. After 200 epochs, the algorithm converges as all the reconstruction error curves are flattened.

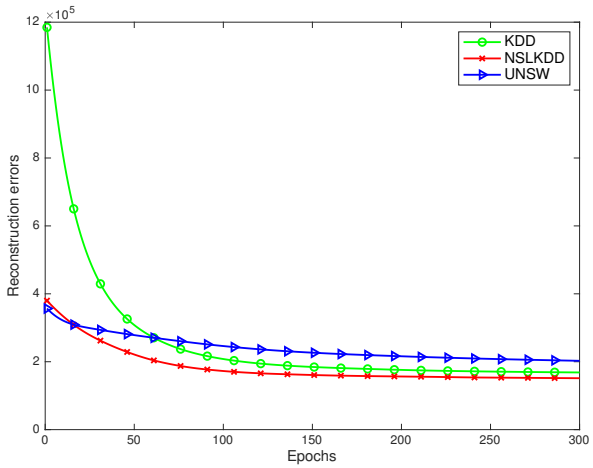
Fig. 17(b) and Fig. 18(b) show the reconstruction errors of nine IoT datasets in both CASE 1 and CASE 2. we can observe the same trend over all datasets, i.e., all errors gradually reduce when the number of epochs increases. However, it can be observed that the trend exhibits some fluctuations in comparison with the trends in Fig. 17(a) and Fig. 18(a) because of the heterogeneous distribution in IoT datasets. The high fluctuation of the reconstruction errors of IoT2, IoT5, IoT6 datasets in Fig. 18(b) also explains why their significant numbers reduce when the number of samples increases in CASE 2. However, the reconstruction errors of all studied datasets in our proposed solution dramatically decrease and become stable after 200 running epochs in both cases.

Mutual Information Analysis

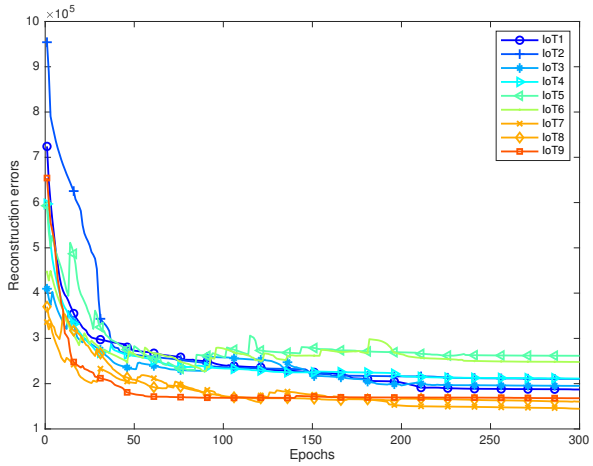
As mentioned in the previous section, network A and network B may share a number of mutual samples. The FTL algorithm exploits the information of these mutual samples to perform the prediction for unlabeled data of network B. This section provides the analysis results to identify how this mutual information can affect to the results of label prediction. In this section, we perform the simulation in CASE 2 with a larger number of samples than in CASE 1. Fig. 19 gives information about the variation of AUC when the percentage of mutual data increases.

Fig. 19(a) shows the increase of AUC on KDD, NSLKDD, and UNSW datasets when the percentage of mutual samples increases from 0.005% to 10%. The AUC of KDD and UNSW datasets sharply increase and remain stable at around 96% on the NSLKDD dataset with about 5% to 10% mutual samples. A similar trend happens with the IoT datasets in Fig. 19(b) when the AUCs of all nine IoT datasets increase and remain stable at approximately 10% of mutual samples. From these results, it can be observed that achieving high efficiency in AUC for IoT datasets may require at least 10% of mutual data.

In summary, the results with 12 cybersecurity datasets show the outperformance of our proposed model in comparison with the state-of-the-art unsupervised deep learning in term of accuracy as shown in Table 15 for CASE 1 and Table 17 for CASE 2, especially with IoT1 and UNSW datasets. Moreover, the reconstruction errors show a fluctuation of the IoT datasets when the number of samples increases due to noise from the collected datasets of some IoT devices. Finally, we vary the amount of mutual data between two networks to evaluate the accuracy of our proposed model. The results show that the proposed model can archieve high performance with 10% mutual data with all datasets.



(a) KDD, NSLKDD, and UNSW datasets

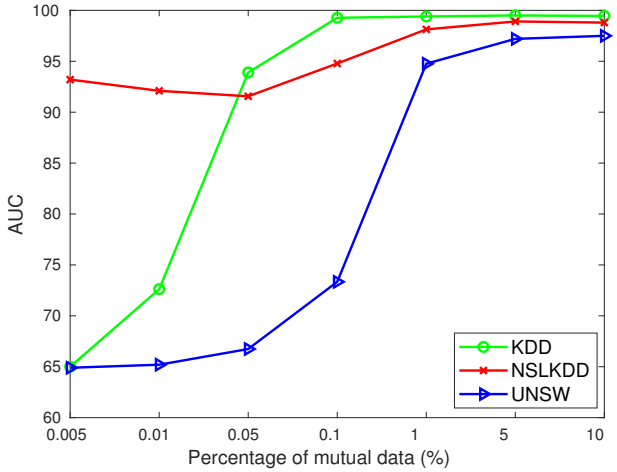


(b) IoT datasets

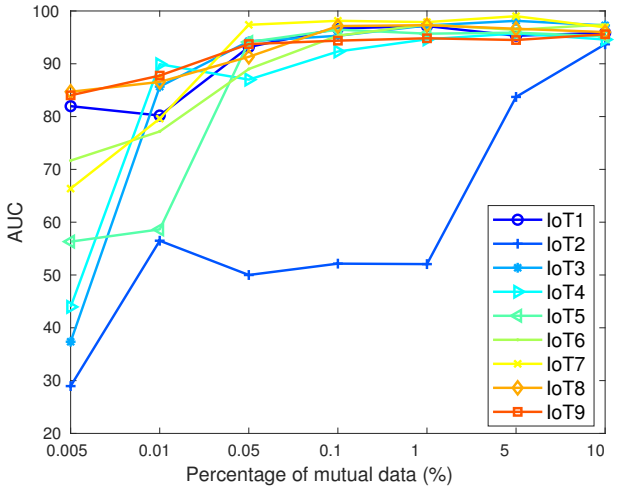
Figure 18: Reconstruction errors in CASE 2.

Conclusion

In this work, we have proposed a novel collaborative learning framework to address the limitations of current ML-based cyberattack detection systems in IoT networks. In particular, by extracting and transferring knowledge from a network with abundant labeled data (source network), the intrusion detection performance of the target network can be significantly improved (even if the target has very few labeled data). More importantly, unlike most of the current works in this area, our proposed framework can enable the source network to transfer the knowledge to the target network even when they are different data structures, e.g., different features. The experimental results then show that the accuracy of prediction of our proposed framework is significantly improved in comparison with the state-of-the-art unsupervised deep learning model. In addition,



(a) KDD, NSLKDD and UNSW datasets



(b) IoT datasets

Figure 19: AUC with different percentage of mutual information.

the convergence of the proposed collaborative learning model is also analyzed with various cybersecurity datasets. In future work, we can consider using other effective transfer learning techniques to make transfer learning processes more stable and achieve better performance, especially when the amount of mutual information is very limited.

PROBLEM 3: Effective Framework of Private Ethereum Blockchain Networks for Smart Grid

Introduction

In Industry 4.0, a huge amount of data is created from smart applications. This leads to a serious security threat to many entities, including users, service providers, and network operators [65]. Blockchain technology [66] has recently been introduced as a promising solution. It stores data on a distributed ledger and a consensus mechanism among the nodes to avoid the ledger being manipulated maliciously.

In 2009, Bitcoin was introduced by Satoshi Nakamoto and hailed as a radical development in money, being the first example of a digital asset. Since Bitcoin only intended to serve as a decentralized payment, the one's transaction will show up in a long time. That is why after Bitcoin, blockchain technologies are blooming with a famous project called "Ethereum" – a decentralized platform. This technology not only possesses advantages of Bitcoin's technologies, i.e., decentralization, transparency, immutability, and security-and-privacy but also has some great improvements, i.e., smart contract and GHOST (Greedy Heaviest Observed Subtree) protocol [67]. It takes only 15 seconds to confirm a new block, approximately 2.5% of Bitcoin [68]. As a result, the Ethereum network is applied in many impactful applications, such as smart agriculture [69, 70], Internet-of-vehicles [71], healthcare [72, 73].

Industry 4.0 also sees the smart grid as an attractive application, which stores, manages, and exploits the electric system [74]. A traditional grid is described in [75] with only one centralized server, controlled by the energy company, interacts with customers with symmetric or asymmetric schemes. A serious risk of the system is centralization, which leads to increasing latency and loss of data when cyber-attacks occur. The Ethereum network can be applied to overcome the risk [76–78]. A typical Ethereum-based smart grid is illustrated in Fig. 20. It is a potential system for the future electric network where users and energy markets are connected.

Several studies related to Ethereum-based smart grid are as follows. Zhuang *et al.* [76] reviewed the blockchain technology and showed the architecture and platform of a blockchain-based smart grid for cyber-security. Huang *et al.* [77] presented smart grid's protocols in theory and implemented a communication system using Sigfox devices, but did not apply the Ethereum network. Gao *et al.* [78] introduced a smart contract for their smart grid, but practical experiments were not investigated. Besides Ethereum, many studies applied the smart grid into other blockchain networks. [79] used the Hyperledger Fabric 1.0 without optimization. [80] proposed a structure of their SmartChain framework, which used the concept of Proof-of-Time (PoT), instead of Proof-of-Work (PoW), for good computational and propagation time versus conventional blockchain network in simulation, but without verification. [81] proposed an energy trading (ET) framework

without targeting a specific blockchain network, however, the performance was evaluated only in terms of the number of Hash functions to execute. Thus, it is unclear if this framework can be implemented in a real blockchain network. Generally, most of the studies reviewed above and others in the literature proposed system models or algorithms of the smart contracts for the smart grid or implementation of a testing system without the Ethereum network. This motivates us to focus, in this work, on developing a novel framework to implement a smart grid with secure data, and improve the efficiency of the private Ethereum network.

As shown in Fig. 20, a smart grid communication infrastructure can be separated into three layers: Home Area Node (HAN) which is the home electrical system, Neighborhood Area Network (NAN) which includes several HANs, Wide Area Network (WAN) which is a network of NANs. In this work, a prototype of an Ethereum-based smart grid is implemented at the HAN layer. This prototype includes the essential components of a smart grid, e.g., smart meters and an IoT Gateway [82]. To secure data inside the blockchain network, encryption methods are required. However, asymmetry schemes would make users reveal their secret keys to the nodes [83]. Therefore, a symmetric pre-encryption technique and a simple smart contract are considered to prevent the encrypted data from being duplicated and to avoid revealing the secret key.

The system is based on a private Ethereum network instead of a public Ethereum network. In a small-scale network, the mining time can be reduced while still ensuring security in the Ethereum network. By modifying the difficulty calculation method of the Ethereum consensus mechanism [84], the performance can be improved, with higher throughput, smaller latency, while keeping an uncle rate the same as that in the main Ethereum network. To experiment this method, we use BlockSim – a recently proposed framework for blockchain systems by Alharby and van Moorsel [85]. The input parameters for performance study by simulation are measured from the real system. Once we have obtained a suitable threshold of block interval in the consensus layer, the trade-off between latency and uncle rate, these parameters are then applied to the prototype to verify the system performance.

The main contribution of this work is to propose an effective framework to build a private Ethereum network for a smart grid. Firstly, a practical Ethereum-based smart grid is deployed with essential hardware at the home electrical system. Secondly, a smart contract for authentication in a securely multi-devices system is proposed. At last, a method to improve the efficiency of an Ethereum-based smart grid setup in practical work with the support of numerical experiments.

Overview of Ethereum Blockchain Technology

Ethereum Blockchain Technology

The Ethereum blockchain network was introduced by Vitalik Buterin in 2015. Typically, it can be divided into seven protocol layers: Storage, Data, Network, Protocol, Consensus, Contract, and Application. The Contract Layer and the GHOST protocol [67] in the Consensus Layer are greatly upgraded, in comparison with those in the Bitcoin network.

The smart contract in the Contract Layer is the first highlight of Ethereum. A smart contract, being simply a piece of code running on Ethereum, can be built with the Solidity language. The Solidity Compiler compiles the smart contract into Bytecode and Application Binary Interface (ABI). Both of them are packaged into a transaction and deployed into the Ethereum network. Bytecode is an executable code on Ethereum Virtual Machine (EVM) and Contract ABI is an interface to interact with EVM Bytecode.

GHOST is a PoW blockchain protocol like in Bitcoin, except for the way it resolves the correct blockchain. Instead of using the longest chain consensus rule in Bitcoin, GHOST follows the path of the sub-tree with the combined hardest proof of work/difficulty. The sub-tree is created because Ethereum allows us to reintroduce orphaned blocks to the chain as "uncles". These uncles in the chain allow the network to reduce the mining time while avoiding multiple forking of the ledger (51% attacks). But the uncle has no role in data storage, so if the uncle rate is too high, it will lead to unnecessary storage effort.

Types of Ethereum Nodes

A node is a device/program that communicates with the Ethereum network, also known as a client. In this prototype, there are two node types of the Ethereum network. A *full* node keeps a ledger, receives or broadcasts transactions to other nodes. Any full node can be used to confirm blocks and transactions and get rewards. In this case, it is also called "miner". A *boot* node keeps Ethereum Node Records (ENR) of many full nodes and is not responsible for keeping the ledger, mining, or broadcasting transactions. Any node that connects with a boot node would discover peers in the network.

Proposed private Ethereum network for smart grid

Private Ethereum Network and Hardware Implementation

At the present, 1 Ether (ETH) is approximately 1400 USD. If the smart grid is deployed in the public Ethereum network, at least transaction fees will be charged. There are two ways to deploy the system and send transaction fees in the Ethereum network; one uses Ethereum test-nets and the other creates an own private network.

In the former, test-nets are for free by given ETH coins in some vaults, and the benefit of this way is that we do not necessarily run our miners. However, the ETH coins obtained from vaults are limited. So for a large-scale system, Ethereum test-nets are not compatible.

In the latter, a private Ethereum network is preferred because it can overcome the disadvantage of the former. To deploy a private network, the Ethereum developer team provides a powerful open-source software named Go-ethereum [86] (Geth). The software includes a number of functions to make private nodes, make boot nodes, create new accounts, run full nodes, and so on. Geth v1.10.4 is used in our setup. The private Ethereum network, as shown in Fig. 21, consists of three full nodes which are personal computers with processor Intel Core i7-4800MQ @2.7 GHz, RAM of 16 GB. The network has been setup by a Cisco switch Catalyst 2950 with 100 Mbps bandwidth.

As shown in Fig. 20, the proposed prototype of the HAN layer includes the essential components, i.e., electrical loads, smart meters, and an IoT gateway CPS 200RE.

XTM35SC is the next generation of electricity meters (smart meters). It measures how much electricity has been used, and displays information on a handy in-home display. Furthermore, data collected from the smart meter can be exploited by other IoT devices which use the Modbus-RTU protocol. In the system, data collected from the smart meter will be exploited, decoded, encrypted, and transmitted through an IoT gateway. For simplicity, only consumed energy data will be collected.

CPS 200RE is an edge IoT gateway. It is fully integrated with Fieldbus accessibility, Modbus TCP/RTU, PROFINET® or EtherNet/IP, and so on, for extremely easy deployment of both centralized/decentralized field data implementation in the automation process. The IoT gateway is responsible for the collection of device identification, collection time, and value of consumed energy.

Security Enhanced Smart Contract

After collecting raw data from smart meters, the gateway encrypts the data to avoid malicious tapping of the data. This work is necessary because the mechanism of the blockchain makes all data public, requiring pre-encryption before transmission. Both symmetry and asymmetry schemes are considered. But in the blockchain, a classical symmetry scheme named AES-256-CTR [87] is used in the system because asymmetry schemes would make a user reveal his/her $key_{private}$ to Ethereum nodes [83]. In the blockchain network, a pair of key_{pub} and key_{pri} are provided when a user creates a new account. In this work, the $key_{private}$ used for AES-256-CTR is the same as the key_{pri}. At this stage, the privacy of the raw data is guaranteed.

The smart contract is given by Algorithm 1 in which the inputs are encrypted identification/collection time/value of consumed energy $\mathcal{F}(\ldots, key_{pri})$ and the outputs are

Algorithm 1
Input: msg.sender (address sending transaction)
_id ⟵ $\mathcal{F}(\_id, key_{pri})$
_time ⟵ $\mathcal{F}(\_time, key_{pri})$
_value ⟵ $\mathcal{F}(\_value, key_{pri})$
Output: push data to the Ethereum network

```
Class SmartFac {
  init_addr ⟵ None
  total_of_reco ⟵ 0
  struct Reco { id, time, value }
  reco[uint][Reco]
  trusted_acc[address][bool]
  Function constructor( ) {
    trusted_acc[msg.sender] ⟵ true
    init_addr ⟵ msg.sender
  }
  Function add_acc( _addr ) {
    if msg.sender ≠ init_addr
      return Error
    trusted_acc[_addr] ⟵ true
  }
  Function rm_acc( _addr ) {
    if msg.sender ≠ init_addr || _addr == init_addr
      return Error
    delete trusted_acc[_addr]
  }
  Event added_reco(addr, _id, _time, _value)
  Function new_reco( _id, _time, _value ) {
    if trusted_acc[msg.sender] ≠ true
      return Error
    total_of_reco ⟵ total_of_reco + 1
    reco[total_of_reco] ⟵ Reco {_id,_time,_value}
    emit added_reco(msg.sender, _id, _time, _value)
  }
}
```

stored inputs, saving the key_{pub} of the user account that has deployed the smart contract as the primary account. This account has permission to add/remove other accounts from the account list, allowing the added/removed account to push data in smart contract or not. The data flow of every smart meter to the private Ethereum network is summarized in Fig 22.

Performance Improvement via Throughput and Latency

The latency of the original Ethereum network is more than 12 seconds because of global scalability [68]. This is not suitable for low-latency applications. When applied to a smart grid, with smaller scalability, the system can be improved to obtain higher throughput and smaller block intervals (T).

Based on the analysis of $10,000$ consecutive blocks in the Bitcoin network, on average, the transmission time of a block which has just been produced from a miner to 50% and 95% of all nodes are 6.5 seconds and 40 seconds respectively, and the mean delay is around 12.6 seconds [88]. For a private Ethereum network with small scalability, the mining time can be reduced subject to block propagation, which is measured in this private Ethereum network.

Firstly, we consider the case when the public Ethereum network keeps the stable block interval described in the Ethereum yellow paper [84]:

$$D_i = \begin{cases} D_0 = 131072, & \text{if } i = 0, \\ \max(D_0, P_D + x \times \zeta + \epsilon), & \text{otherwise}, \end{cases} \tag{15}$$

with

$$x = \left\lfloor \frac{P_D}{2048} \right\rfloor, \tag{16}$$

$$\zeta = \max\left\{ y - \left\lfloor \frac{T}{9} \right\rfloor, -99 \right\}, \tag{17}$$

$$y = \begin{cases} 1, & \text{if } \|P_U\| = 0, \\ 2, & \text{otherwise}, \end{cases} \tag{18}$$

$$\epsilon = \left\lfloor 2^{\lfloor \max(i - 5000000, 0) \div 100000 \rfloor - 2} \right\rfloor, \tag{19}$$

where D is "difficulty" which is a scalar value corresponding to the difficulty level of this block, P is the parent of this block, P_D, P_s and P_U are "difficulty", "time_stamp", and "the number of uncles" of P, respectively, $\lfloor \cdot \rfloor$ denotes the the integer division operator, the index i indicates the current block number, T is the block interval, given by $T =$ current_block_time_stamp $- P_s$.

We focus on the case in which $\|P_U\| = 0$. By this constraint, we consider the effect of the propagation time to T. Simultaneously, when the number of uncles is reduced, the size of the ledger is decreased while still keeping all transactions. Moreover, reducing the number of uncles in the ledger also avoids "selfish mining" in the network when some miners are simply to mine uncles instead of blocks extending the best chain [89].

We can observe that the Ethereum consensus does not vary T directly, but indirectly through D and a threshold λ in replacement of value 9 in (17). In detail, Eq. (15) depends

Table 18: The parameters in the network are pre-setup based on the measured and our experiments.

Parameters	Values	Notes
Block gas limit	15,000,000 gas	Same as main Ethereum network at June-6-2021
Average transaction size	0.759808 kB	Based on our smart contract
Average block size	60 kB	Same as main Ethereum network at June-6-2021
Average block propagation delay	0.25 seconds	Based on our experiment measured
Sync mode	light	Just sync block header
Number of transactions created per second	100 transactions per second	
Number of node	3 miners	Each miner has 33.33% of the total computing power

on Eq. (17), so unless the current block is the first one (genesis block), if $T < \lambda$ then D is adjusted upwards by $(x + \epsilon)$, if $\lambda \leq T < 2\lambda$ then D is unchanged, and if $T \geq 2\lambda$ then D is adjusted downwards proportional to the timestamp difference by from $(-x + \epsilon)$ to $(-99 \times x + \epsilon)$. It can be seen that T is always desired between λ and 2λ seconds, subject to λ in (17). It should be well noted that this threshold is rooted from [88].

However, in a private Ethereum network of limited size, T can be reduced while ensuring that the block propagates through 95% of the nodes. This leads to our modification in the consensus mechanism to improve the block interval. We propose to modify the source code of Geth [86] by restoring the use of the threshold λ in [88] as in

$$\zeta = \max\left\{y - \left\lfloor \frac{T}{\lambda} \right\rfloor, -99\right\} \tag{20}$$

instead of fixing it to 9 as in (17), and obtain the value of λ based on the practical smart grid setup.

Experiments

Performance Study by Simulation

Recently, a simulation framework called BlockSim [85] is introduced. It is used to evaluate a Bitcoin/Ethereum blockchain network depending on input parameters. This work focuses on evaluating three parameters, i.e., block interval, throughput (transactions per second), and uncle rate. The others based on either the main (public) Ethereum network or measured data from the prototype, and are given in Table 18.

The simulation results are shown in Fig. 23. The line with marker of triangle and that of diamond present uncle rate and throughput versus block interval, respectively. The results are averaged over 100 experiments.

Generally, the throughput and the uncle rate both decrease as the block interval time increases. The system has maximum performance with throughput of more than 80 transactions per second (tx/s) and block interval of less than 2 seconds. However, 7-18% of uncle rate is not good because of increasing storage. Comparing with the main

Table 19: Experiment results versus the main Ethereum network

Parameters	Avg values in the private Eth	Avg values in the main Eth
Transactions per second	50.08 tx/s	16.25 tx/s
Uncle Rate	3.03%	4.81%
Block interval	2.7 seconds	13.48 seconds

Ethereum network of 4.81% on average [90], the proposed network has $T = 3$ seconds. At this value of block interval, throughput and uncle rate are 77.72 tx/s and 4.88%, compared with 29.95 tx/s and 1.47% at $T = 12$, respectively.

Besides, other simulation results of a main Ethereum network [88] with 3 full nodes are two scatter points as shown in Fig. 23 with throughput and uncle rate are 14.05 tx/s and 17.48%, respectively. This throughput is suitable with the real main Ethereum network.

Verification by Real Data

From the experiment by simulation, to obtain the block interval of approximately 3 seconds and the uncle rate of approximately 4.81%, as shown in Section (1), the threshold λ in Eq. (20) is found to be 3.

The experimental results from the real prototype are presented in Fig. 24 with $7,000$ consecutive blocks. The top line and bottom line show the average number of transactions per second in a block time and the block interval, respectively. Here, because the mining process depends on probability, so in this figure, we can see that both are not stable at a specific value. The average values of both on $7,000$ blocks, compared to that of the main Ethereum network (found at `www.etherscan.io/chart`), are shown in Table 19. According to the real experiment, the proposed framework can handle 50.08 tx/s and 2.7 seconds between two blocks while those of the main Ethereum network are 16.25 tx/s and 13.48 seconds. Moreover, the obtained uncle rate is 3.03%, which is smaller than that of the main Ethereum network.

The throughput of the real experiment is not as good as that of the simulated experiment by BlockSim but better than that of the main Ethereum network. While the latency is much smaller than that of the main Ethereum network. It is clear that the proposal of changing the consensus mechanism did give better performance.

However, decreasing T may cause risky scenarios, for example, a significantly faster miner joins the network and takes all mining jobs. It can lead to a 51% attack by confirming dishonest transactions. In a private network, there are a few technical methods provided by Geth to solve this problem, e.g., whitelist IP, set maxpeers; more details in [86].

Conclusion

We have proposed an effective framework to build a private Ethereum network for smart grid with an own private Ethereum network and essential hardware of a smart grid at the home electrical system. The AES-256-CTR standard is applied to pre-encrypt raw data and a smart contract for authentication has been proposed. Then, we have shown how to improve the efficiency of a practical smart grid setup and our verification system can obtain throughput of 50.08 tx/s and latency of 2.7 seconds at an uncle rate of 3.03%. These results clearly show that our proposed framework can outperform the original setup for a private Ethereum network. Moreover, this framework can be applied to any system used to store data in the Ethereum network with any scale. In the future, other factors, e.g., the number of nodes, transaction size, will be investigated to fully evaluate a private Ethereum network.

PROBLEM 4: Collaborative Learning for Cyberattack Detection in Blockchain Networks

Introduction

Blockchain [91, 92] has been emerging as a novel technology in storing and managing data with many advantages over conventional data management systems. In particular, unlike traditional centralized data management solutions, blockchain technology allows data to be stored in a distributed manner across multiple nodes. In this way, data can be accessed and processed simultaneously at multiple nodes, and thus avoiding the problem of bottlenecks and single point of failure. More importantly, one of the most important features of blockchain technology is to enable data to be stored in blocks, and once a block of data is verified and placed in the chain, it cannot be modified and/or deleted. In this way, the data's integrity can be protected thanks to outstanding features of blockchain, e.g., decentralization, immutability, auditability, and fault tolerance [91]. As a result, there are more and more applications of blockchain technology in our lives including finance, healthcare, logistics and IoT systems [91–93].

Due to the rapid success with a wide range of applications in most areas, especially in money transfer and cryptocurrency, blockchain-based systems have been becoming targets of many new generation cyberattacks. For example, in September 2020, KuCoin, a crypto exchange based in Singapore, announced that its system was hacked and the hackers stole over $281 million worth of coins and tokens [94]. In May 2019, Binance, one of the biggest cryptocurrency exchange companies in the world, reported to be hit by a major security incident. In particular, the hackers did break the exchange's security system and withdraw over 7,000 bitcoins from digital wallets, causing a total loss of approximately $40 million for the customers [94]. Most recently, in January 2022, Chainalysis reported that North Korean hackers performed seven attacks into cryptocurrency platforms and stole nearly $400 million from digital assets in 2021 [95]. Although most of current attacks target on virtual money exchange systems, a number of blockchain applications in critical areas such as healthcare [96] and food supply chains [97] could be potential for attackers in the near future. These attacks, if happen, not only cause huge losses on our assets, but can also lead to many serious issues related to human health and lives. Therefore, solutions to detect and prevent attacks in blockchain networks are becoming more urgent than ever.

In network security, Machine Learning (ML) has been being considered as the most effective solution to detect cyberattacks with very high accuracies [98–100]. The main reasons for the outstanding advantages of using ML for intrusion detection problems compared with other conventional detection methods such as signature-based and abnormally-based are threefold. First, unlike conventional intrusion detection solutions which are usually designed to detect a specific type of attack (e.g., virus, trojan, spam and botnet), ML solutions allow to detect many types of attacks at the same time with very high accuracies. For example, Deep Learning (DL) allows to detect cyberattacks in

industrial automation and control system with an accuracy of up to 97.5% [101]. Second, while traditional solutions are often designed to detect known attacks, ML allows to detect attacks that have never been detected and reported before. For example, in [102], the authors showed that their DL model can detect attacks such as, the DDoS attack of Mirai and BASHLITE botnets, even though such types of attacks have never been learned/trained before by this model. Last but not least, ML algorithms, especially DL, can be deployed effectively, quickly and flexibly. For example, after a deep neural network is trained, it can be deployed in different intrusion detection systems at the same time to detect cyberattacks quickly with high accuracies. In addition, when data about new types of attacks is available, we can easily update new versions of deep neural networks through transfer learning techniques [103].

As a result, ML has been considering as a highly-effective solution to detect the cyberattacks for blockchain networks [104]. In particular, in [105], the authors proposed to use Random Forest and XGBoost to detect attacks in a blockchain-based IoT system. The results show that this solution can identify different types of attacks and normal behaviors with an accuracy of up to 99%. However, they only tested their results on the BoT-IoT dataset that is not real blockchain traffic. Similarly, in [106], the authors proposed an ML-based method, called bidirectional long short-term memory (BiLSTM) to detect attacks in an IoT network before the data is stored in the blockchain network. Although the results also showed that they can detect different kinds of attacks with an accuracy of up to 99%, they were validated only on conventional network datasets such as UNSW-NB15 and BoT-IoT datasets. These datasets are collected in conventional computer networks and thus cannot reflect actual traffic in blockchain networks. In particular, these datasets have just general attacks in computer networks without specific attacks in blockchains, e.g., changes in blockchain transactions, incorrect consensus protocol or the break of the chain of blocks.

To the best of our knowledge, there are only few works that consider to use the real blockchain traffic, e.g., try to generate artificial data or try to create data to simulate an attack for blockchain networks to train ML models such as [107–109]. Specifically, in [107] the authors proposed a method to collect blockchain traffic data. First, they captured traffic samples from a public Bitcoin node and used them as the normal network data. Then, for the malicious traffic data, the authors performed DoS and Eclipse attacks on a target device (this device was created to become a node in the Bitcoin network). After that, the collected data was used to train an autoencoder deep learning model. This solution showed an accuracy of attack detection up to 99%. In [108], the authors used a public dataset and a private dataset from their testbed. Then, they proposed to use a Long Short-Term Memory Network (LSTM) to learn the properties of normal samples in the datasets. After that, they deployed a Condition Generative Adversarial Networks (CGAN) model to generate the artificial Low-rate Distributed DoS (LDDoS) attack samples for their blockchain dataset. The results showed the accuracy of classification up to 93%.

In addition, in [109], the authors performed a DDoS attack namely Link Flood Attack (LFA) on a the simulation Ethereum network and collected the traceroute records of the network in both normal and attack behaviors. After that, the authors used the Recurrent Neural Network (RNN) to analyze the traceroute records to identify the attacks in the network. The results showed that the attack detection rate can achieve nearly 99%.

From all the above works and others in the literature, we can observe two main challenges for ML-based intrusion detection systems in blockchain networks which still have not been addressed. In particular, the first challenge is lacking of a synthetic data from laboratories for training ML models. Most of current works, e.g., [105] and [106] are using conventional cybersecurity datasets (e.g., UNSW-NB15 and BoT-IoT) to train data. However, these datasets were not designed for blockchain networks, and thus they are not appropriate to use in intrusion detection systems in blockchain networks. Other works, e.g., [107–109], tried to build their own datasets for blockchain networks, e.g., by obtaining the normal samples from the Bitcoin network [107], creating simulation experiment to detect the LFA [109] and generating artificial attack samples by CGAN [108]. However, these methods have several issues. First, normal samples of transactions from the Bitcoin network may include attacks from public blockchain network, but all collected data are classified and labeled to be normal data. Second, the simulation experiment in [109] was to generate traceroute records only for the LFA so they cannot extend to other attacks. Furthermore, it is difficult to evaluate the effects of artificial attack samples in [108] whether they can simulate a real attack into blockchain network or not. The another challenge we can observe here is that all of current ML-based intrusion detection solutions for blockchain networks are based on centralized learning models, i.e., all data is collected at a centralized node for training and detection. However, this solution is not suitable to deploy in blockchains as they are decentralized networks. Specifically, nodes in blockchain networks may have different data to train and due to privacy concerns, they may not want to share their raw data to a centralized node (or other nodes) for training processes. Moreover, sending a huge amount of data to the network will not only cause excessive network traffic, but also risk compromising the data integrity of blockchain networks.

This work aims to address the aforementioned challenges by first introducing ***a novel intrusion detection dataset named BNaT which stands for Blockchain Network Attack Traffic***, created from a real blockchain network in our laboratory and then proposing an effective decentralized collaborative machine learning framework to detect intrusions in the blockchain network. Specifically, to develop BNaT, we first setup and implement a blockchain network in our laboratory using Ethereum (an open-source blockchain software) and perform intensive experiments to generate blockchain data (including both normal and malicious traffic data). The main objectives of producing BNaT dataset are fourfold. Firstly, we collect the BNaT in a laboratory environment to have "clean" data samples (i.e., to ensure that the obtained data is not corrupted, error

and/or irrelevant), that is especially important for training ML models. Secondly, the BNaT can be easily extended to include to new kinds of blockchain attacks, e.g., 51% or double spending attacks. Thirdly, we perform experiments with real attacks in the considered blockchain network, and thus the BNaT can reflect better the actual attack behavior of network than simulations or by artificial attack data generated by GAN in the literature, e.g., [108]. Fourthly, we collect the data in different blockchain nodes to have a complete view of effects when the attacks are performed in a decentralized manner. After that, we develop a highly-effective collaborative learning framework to make it more effective in deploying in blockchain networks to detect attacks. In particular, in our proposed learning framework, working nodes in the blockchain network (e.g., mining nodes) can be used as learning nodes to collect blockchain data (e.g., observing its own traffic and classifying data). However, unlike centralized learning models, these nodes will train their learning models by using their own datasets instead of sending their datasets to a centralized node for processing. After that, these nodes can share their trained models to other nodes in the network to improve the accuracy of the trained models. In this case, even the nodes do not need to share their raw data, they still can learn useful information from other nodes in the network through extracting information from shared trained models. The main contributions of this work can be summarized as follows.

- We set up experiments in our laboratory to build a private blockchain network with the aims of not only obtaining real blockchain datasets, but also testing our proposed learning model in a real-time manner. To the best of our knowledge, this is the first dataset obtained from a laboratory for studying cyberattacks in blockchain networks, and thus we expect that our proposed BNaT dataset can promote the development of ML-based intrusion detection solutions in blockchain networks in the near future.
- We build an effective tool named Blockchain Intrusion Detection (BC-ID) to collect data in the blockchain network. This tool can extract features from the collected network traffic data, filter attack samples in network traffic, and exactly label them in a real-time manner.
- We propose a collaborative decentralized learning model to not only improve the accuracy of identifying attacks, but also effectively deploy in decentralized blockchain networks. This model enables nodes in the network to effectively share their trained models to improve cyberattack detection efficiency without sharing their raw data.
- We perform both intensive simulations and real-time experiments to evaluate our proposed framework. Both simulation and experimental results clearly show outperformance of our proposed framework compared with other baseline ML methods. Furthermore, our results reveal some important information in designing and

implementing learning models in blockchain networks in practice, e.g., real-time monitoring and detecting attacks.

Blockchain Network: Fundamentals and Proposed Network Model

Blockchain

Blockchain is a novel method in storing and managing data in a decentralized manner. In a blockchain network, multiple nodes are used to simultaneously process and store data. In particular, when a node in the blockchain network receives transactions (e.g., money exchange in the Bitcoin network), it will gather all the transactions and put in a block. This node will then start a mining process to find a "nonce" value for this block. It is important to note that thanks to the feature of the hash function, there is only a small set of satisfying nonce values for a block, and these values can only be found through an intensive searching process [66]. This mining process is a special process of blockchain networks to provide proofs for validated blocks, and thus this tamper-proof can significantly enhance security for blockchain networks. After the node finds the nonce value for the mining block, this new block will be broadcast and verified by other nodes in the network. Finally, if this block is verified, it will be put to the chain (linked to the hash value of the previous block inside its header). After the block is added to the chain, it is nearly impossible to change information in this block, and thus this property can guarantee the immutability of the blockchain. Another aspect of blockchain is traceability due to infeasible collision of the hash function, and thus any transaction or block can be tracked correctly. In summarize, blockchain can be termed as a decentralization, immutable, traceable, and time-stamped digital data chain (ledger).

Designed Blockchain Network at our Laboratory

In order to launch a blockchain network, there are two main kinds of blockchain nodes namely full node and bootnode. Firstly, full nodes take responsibility to store the ledger, participate in the mining process, and verify all blocks and states. Furthermore, they can be used to serve the network and provide data on request, e.g., netstats, which is a visual interface for tracking Ethereum network status (e.g., the block number, mining status and the number of pending transactions). Secondly, bootnode is a lightweight application used for the Node Discovery Protocol. The bootnodes do not synchronize blockchain ledger but help other Ethereum nodes discover peers to set up Peer-to-Peer (P2P) connections in the network.

The system model together with essential components of our designed blockchain network are set up as illustrated in Fig. 25. Specifically, the model includes K full nodes which are used to receive transactions, mining blocks, and keep the replica of ledger.

These nodes continuously synchronize their ledgers together by the P2P protocol with equal permissions and responsibilities for processing data [66]. In order to connect them together, a management node, known as bootnode, is setup. The full nodes connect and interrogate this bootnode for the location of potential peers in the blockchain network. After being connected, each full node can collect data (i.e., transactions) from its network. Transactions can come from different blockchain applications such as cryptocurrency, smart city, food supply chain, and IoT. First, when transactions are sent to a full node, they will be verified and packed into one block. After the node finds the nonce value for this block, it will broadcast the block together with this nonce value to other nodes in the network for verification. Finally, if the block is verified by majority nodes in the network, it will be added to the chain.

At our laboratory, we design a private blockchain network based on the Ethereum blockchain network. This network also uses the Proof-of-Work (PoW) consensus mechanism, but the block confirmation time is significantly faster than the older version in Bitcoin. Furthermore, the smart contract layer of Ethereum is suitable for flexible purposes of decentralized environments as mentioned above. In addition, at each node, several attacks, that can cause serious damages in the public blockchain network, will be considered. We then capture the traffic data to analyze their impacts on the blockchain network using BC-IDS. Note that, in practice, there is no software which supports automatically capturing the blockchain network traffic so far. Therefore, we have to analyze the blockchain network traffic data by using the Wireshark [110] and build a new collection tool, namely BC-IDS (more details will be explained in Section **??**). In this way, we can observe effects of these attacks on different nodes in the blockchain network.

Proposed Cooperative Learning Model

In this section, we introduce our proposed collaborative learning model which can collaborate to effectively and decentralizedly detect attacks in the blockchain network. Our proposed model aims to leverage knowledge learned from all the nodes in the network without revealing their raw data. To do so, we first design a framework in the decentralized blockchain network in which each participated learning node (i.e., fullnode in the blockchain network) will use a deep learning model (we will explain more details in the next section) to learn from its collected data and then share its trained model to a Centralized Server (CS). The CS can be a bootnode or any full node in the blockchain network. After that, the CS will aggregate all the trained models and send the aggregated model (i.e., the global model) back to the participated learning nodes. By doing this process iteratively, the learning nodes can gradually update their deep learning models and finally can reach the convergence (to the global training model). In this way, we can not only improve the accuracy of detecting cyberattacks in blockchain network but also eliminate the risks of exposing data over the network. Fig. 25 describes our

proposed framework for intrusion detection in the blockchain network. In this design, all the blockchain nodes can quickly detect and prevent attacks to them. In addition, thanks to the advantages of our proposed machine learning model, our framework can detect attacks very quickly with very high accuracies (more details and results will be explained in Section (1)). Therefore, our proposed model can perform offline training and real-time detection to quickly and efficiently prevent attacks in decentralized blockchain networks.

Proposed collaborative learning model for intrusion detection in blockchain network

In our proposed collaborative learning model, full nodes in the blockchain networks will be used as Learning Nodes (LNs) to learn data and share their learned knowledge to improve learning performance for the whole network. At each learning node, we propose to use a deep neural network to learn useful information from its collected data. Then, the learning nodes will share their trained learning models to the CS. After that, the CS will calculate the aggregated model (i.e., the global model) and share this model back to the LNs. When a learning node receives this aggregated model from the CS, it will integrate with its current learning model and train its local dataset. This process will be repeated until convergence or reaching a predefined maximum number of iterations. To the end, we can obtain the global learning model for all the learning nodes.

In our proposed model, each blockchain node has a set of local collected data, and we propose a deep neural network (DNN) using Deep Belief Network (DBN) to better learn knowledge from this data. The DBN is a type of deep neural network that is used as a generative model of both labeled and unlabeled data. Therefore, unlike other supervised deep neural networks which use labeled data to train the neural networks (e.g., convolutional neural networks [64]), the DBN has two stages in the training process. The first stage is the pre-training process where the DBN trains its neural network with unlabeled dataset. The second stage is fine-tuning process where DBN uses labeled dataset to train its neural network. Thereby, the DBN can represent better the characteristics of dataset, and thus it can classify the normal behavior and different types of attacks with very high accuracies. In addition, the DBN includes multiple Restricted Boltzmann Machines (RBM) layers for latent representation [111]. In the DBN training process, the current layer generates latent representation by using latent representation of previous layer as the input. Unlike other deep neural networks which also can process both labeled and unlabeled data (e.g., autoencoder deep learning network [64]), the DBN optimizes the energy function of each layer to have better latent representations of data on each RBM layer in each iteration. Thereby, the DBN is more appropriate to analyze the blockchain network traffic where the samples and features have relatively coherence with each other.

The whole processes of DBN are illustrated in Fig. 26. Like other DNNs, the structure of DBN has three layers including an input layer, an output layer and multiple hidden

layers. As can be seen in Fig. 26, the Gaussian Restricted Boltzmann Machines (GRBM) layer, a type of RBM which can process real values of data, is the input layer to receive and transform the input data into binary values. We denote $k = \{1, ..., K\}$ as the number of learning nodes in the collaborative learning model, $\boldsymbol{v}^k$ and $\boldsymbol{h}^k$ to be the vectors of visible and hidden layers of LN-k, respectively. In addition, M and N are the numbers of visible and hidden neurons of GRBM, and M' and N' are the numbers of visible and hidden neurons of RBM. As defined in [112] the energy functions of GRBM and RBM of LN-k are defined as follows:

$$E^k_{GRBM}(\boldsymbol{v}^k, \boldsymbol{h}^k) = \sum_{m=1}^{M} \frac{(v^k_m - b_{1,m})^2}{2\epsilon^2_m} - \sum_{m=1}^{M}\sum_{n=1}^{N} w_{m,n} h^k_n \frac{v^k_m}{\epsilon_m} - \sum_{n=1}^{N} b_{2,n} h^k_n, \tag{21}$$

$$E^k_{RBM}(\boldsymbol{v}^k, \boldsymbol{h}^k) = -\sum_{m=1}^{M'} b_{1,m} v^k_m - \sum_{m=1}^{M'}\sum_{n=1}^{N'} w_{m,n} v_m h^k_n - \sum_{n=1}^{N'} b_{2,n} h^k_n, \tag{22}$$

where $w_{m,n}$ is the weight between visible and hidden neurons, $b_{1,m}$ and $b_{2,n}$ indicate the bias of visible and hidden neurons, respectively, and ϵ_m represents the standard deviation of the neuron in the visible layer. From the energy functions, we can find the probability that is used in the visible layer of DBN [112] as follows:

$$p^k(\boldsymbol{v}^k) = \frac{\sum_{\boldsymbol{h}^k} e^{-E(\boldsymbol{v}^k, \boldsymbol{h}^k)}}{\sum_{\boldsymbol{v}^k}\sum_{\boldsymbol{h}^k} e^{-E(\boldsymbol{v}^k, \boldsymbol{h}^k)}}. \tag{23}$$

Then, we use the probability in (23) to calculate the gradients of each layer with the expectation value $\langle . \rangle$ as follows [112]:

$$\nabla g^k_{m,n} = \frac{\partial \log p^k(\boldsymbol{v}^k)}{\partial w_{m,n}} = \langle v^k_m h^k_n \rangle_{dataset} - \langle v^k_m h^k_n \rangle_{model}. \tag{24}$$

After learning with multiple GRBM and RBM layers, we define $\boldsymbol{X}^{g,r}_k$ as the output of the last hidden layer of LN-k. In this work, the output layer utilizes the softmax regression function to classify the data samples based on the probability. We denote $\boldsymbol{W}^o$ and $\boldsymbol{b}^o$ as the weight matrix and bias vector between the output and the last hidden layer, respectively. We then can define the probability of the output Z belonging to Class-t as follows:

$$p^o_k(Z = t | \boldsymbol{X}^{g,r}_k, \boldsymbol{W}^o, \boldsymbol{b}^o) = softmax(\boldsymbol{W}^o, \boldsymbol{b}^o) = \frac{e^{\boldsymbol{W}^o_t \boldsymbol{X}^{g,r}_k + \boldsymbol{b}^o_t}}{\sum_n e^{\boldsymbol{W}^o_n \boldsymbol{X}^{g,r}_k + \boldsymbol{b}^o_n}}, \tag{25}$$

where $t \in \{1, ..., t, ..., T\}$ is a class of the output and T refers to the total classes (including

different types of attacks and normal behavior). The prediction $\boldsymbol{Z}$ of the probability p^o is

$$\boldsymbol{Z}_k = \arg\max_t [p_k(Z = t|\boldsymbol{X}_k^{g,r}, \boldsymbol{W}^o, \boldsymbol{b}^o)], \tag{26}$$

where Z is the output prediction from $\boldsymbol{Z}$. Then, we can calculate the gradient between the output layer and the last hidden layer from (25) as follows:

$$\nabla g_k^o = \frac{\partial p_k^o(Z = t|\boldsymbol{X}_k^{g,r}, \boldsymbol{W}^o, \boldsymbol{b}^o)}{\partial \boldsymbol{W}^o}. \tag{27}$$

After that, the results of (24) and (27) are used to calculate the total gradient ∇g_k^t of DBN with multiple GRBM, RBM layers and the output layer of LN-k as follows:

$$\begin{aligned} \nabla g_k^t &= \sum_{\boldsymbol{v}^k, \boldsymbol{h}^k} \nabla g_k^{GRBM} + \sum_{\boldsymbol{v}^k, \boldsymbol{h}^k} \nabla g_k^{RBM} + \nabla g_k^{output} \\ &= \sum_{\boldsymbol{v}^k, \boldsymbol{h}^k} \sum_{m=1}^{M} \sum_{n=1}^{N} \nabla g_{m,n}^k + \sum_{\boldsymbol{v}^k, \boldsymbol{h}^k} \sum_{m=1}^{M'} \sum_{n=1}^{N'} \nabla g_{m,n}^k + \nabla g_k^o. \end{aligned} \tag{28}$$

In the training process, the DBN first trains its neural network with unlabeled data for pre-training. Then, DBN uses its labeled data to fine-tuning its neural network. At this stage, the DBN of LN-k calculates its gradient ∇g_k^t. After that, this gradient is sent to the CS to create an updated global model for all LNs as illustrated in Fig. 27. For example, at iteration i the CS receives gradients from all K LNs, the CS first performs the average gradient function [113] as follows:

$$\nabla g^* = \frac{1}{K} \sum_{k=1}^{K} \nabla g_k^t. \tag{29}$$

We then denote $\boldsymbol{\varphi}_i$ as the global model at iteration i which includes the weight matrix for all layers of the LN's deep learning model, and μ represents the learning rate. From (29), the CS can calculate a new global model at iteration $i + 1$ as follows:

$$\boldsymbol{\varphi}_{i+1} = \boldsymbol{\varphi}_i + \mu \nabla g^*. \tag{30}$$

Next, the CS sends the latest global model $\boldsymbol{\varphi}_{i+1}$ to the LNs to update their deep learning models. This process is repeated until it reaches the convergence or gets the maximum number of iterations. At this time, we can find the optimal global model $\boldsymbol{\varphi}_{opt}$ that includes the optimal weights of all layers. We denotes $\boldsymbol{W}_{opt}^o$ as the optimal weight matrix between the output layer and the last hidden layer from $\boldsymbol{\varphi}_{opt}$, the output prediction

Algorithm 4 The classification-based collaborative learning algorithm

while i $\leq$ maximum number of iterations or the training process is not converged **do**
  for $\forall k \in K$ **do**
    DBN of LN-k learns $\boldsymbol{X}_k$ and produces $\boldsymbol{Z}_k$.
    Calculate gradient ∇g_k^t.
    Send ∇g_k^t to CS.
  end for
  CS calculates average gradient ∇g^* and global model $\boldsymbol{\varphi}_i$.
  $i = i + 1$.
  CS updates global model $\boldsymbol{\varphi}_{i+1}$.
  CS sends global model $\boldsymbol{\varphi}_{i+1}$ to all LNs.
  LNs update their DBNs.
end while
DBN of LNs predict and classify $\boldsymbol{Z}_k$ from the training dataset $\boldsymbol{X}_k$ with the optimal global model $\boldsymbol{\varphi}_{opt}$.

$\boldsymbol{Z}_k$ of LN-k thus can be calculated as follows:

$$\boldsymbol{Z}_k = \arg\max_t [p_k(Z = t|\boldsymbol{X}_k^{g,r}, \boldsymbol{W}_{opt}^o, \boldsymbol{b}^o)]. \tag{31}$$

From (31), the softmax regression of each LN can classify its the blockchain network samples to be normal behavior or a type of attacks. Algorithm 4 summarizes the process of our proposed collaborative learning model.

Experiment Setup, Dataset Collection and Evaluation Method

This section will explain more details about experiment setup, data collection, and feature extraction over our designed blockchain system.

Experiment Setup

In our experiments, we setup a small-scale Ethereum blockchain network in our laboratory which includes three Ethereum full nodes, an Ethereum bootnode and a netstats server. All these nodes are connected to a Cisco Switch Catalyst 2950 as illustrated in Fig. 28. The details of these nodes are as follows:

- Ethereum full nodes are launched by *Geth v1.10.14* [114] - open-source software for implementation of the Ethereum protocol. These nodes share the same initial configuration of genesis block, i.e., PoW consensus mechanism, 8,000,000 gas for block gas limit, initial difficulty 1,000,000. Each node is running on a personal computer with processor Intel Core i7-4800MQ @2.7 GHz, RAM of 16 GB.

- Bootnode is also created by *Geth v1.10.14* and connected to all the three Ethereum nodes.
- Ethereum netstats is launched by an open-source software named "eth-netstats" on Github [115].

The normal traffic data is configured with the three trustful servers, while an attack device will execute abnormal/malicious activities to the blockchain network traffic. Each trustful server takes responsibility for generating data and sending transactions to the corresponding Ethereum node in its network as visualized in Fig. 28. In summary, in the normal state, the following tasks are scheduled or randomly occur in the network:

- The servers are scheduled to send transactions.
- The users call functions in the deployed smart contracts to explore the ledger. Besides transaction-related functions, the users also send requests to the Ethereum nodes for tracking their balances or status of miners. Both of these works are randomly made by HTTP requests to the Ethereum node API (Application Programming Interface).
- Ethereum nodes broadcast transactions and mined blocks to synchronize their ledgers. The packets of bootnode are also included in this field.
- WebSockets and JSON-RPC are used when netstats get information from *Geth* clients.
- HTTP requests and replies to view netstats interface and results of cyberattack detection.

Data Collection and Feature Extraction

In this section, we consider network layer aspects of the permissionless blockchain [116, 117] to detect cyberattacks in blockchain network. In general, the goals of an adversary are usually the monetary benefit, e.g., chain splitting, and wallet theft, or stability of the network, e.g., delay and information loss. Hence, in our network, we perform three specific types of cyberattacks that have been reported in blockchain networks, i.e., the brute password for wallet theft, denial of service for information loss, and flooding of transactions for consensus delay. More details are as follows:

- *Brute Password (BP) attack*: is derived from traditional cyberattack when hackers perform such attacks to steal blockchain users' accounts. In this way, the hackers can access the users' wallets and steal their digital assets of the users. As mentioned

Table 20: Features of the designed dataset.

#	Features name	T	Description
Basic features			
1	*duration*	C	length of the connection (seconds)
2	*protocol_type*	D	type of the protocol (i.e., tcp, udp, icmp)
3	*service*	D	network service (e.g., http, ssh, etc)
4	*src_bytes*	C	number of data bytes from source to destination
5	*dst_bytes*	C	number of data bytes from destination to source
6	*flag*	D	normal or error status of the connection
Statistical features			
Features refer to source IP-based Statistical			
7	*count*	C	number of connections to the same source IP as the current connection
8	*srv_count*	C	number of connections to the same service as the current connection
Features refer to these same source IP connections			
9	*serror_rate*	C	% of 'SYN' errors connections
10	*same_srv_rate*	C	% of same service connections
11	*diff_srv_rate*	C	% of different services connections
Features refer to these same service connections			
12	*srv_serror_rate*	C	% of 'SYN' errors connections
13	*srv_diff_host_rate*	C	% of different host connections
Features refer to destination IP-based Statistical			
14	*dst_host_count*	C	number of connections to the same destination IP as the current connection
15	*dst_host_srv_count*	C	number of connections to the same service as the current connection
Features refer to these same destination IP connections			
16	*dst_host_same_srv_rate*	C	% of same service connections
17	*dst_host_diff_srv_rate*	C	% of different services connections
18	*dst_host_same_src_port_rate*	C	% of same both source port and destination IP connections
19	*dst_host_serror_rate*	C	% of 'SYN' errors connections
Features refer to these same service connections			
20	*dst_host_srv_diff_host_rate*	C	% of different host connections
21	*dst_host_srv_serror_rate*	C	% of 'SYN' errors connections

in Section **??**, the BP attack on KuCoin caused the loss of up to $281 million [94]. To perform this attack, the attacker retries passwords of an Ethereum public key until it finds out the correct login information.

- *Denial of Service (DoS) attack*: is also another common type of attack in blockchain networks as it can be easily performed to attack blockchain nodes. For such kind of attack, the attackers will launch a huge amount of traffic to a target blockchain node in a short period of time. Consequently, the target node will not be able to work as normal, i.e., mining transactions, and even be suspended. In the real-world, Bitfinex [118] was temporarily suspended due to such kind of attack. Thus, in our setup, a simple DoS attack is simulated, i.e., an SYN flood attack, by repeatedly sending initial connection request (SYN) packets to an Ethereum node.

- *Flooding of Transactions (FoT) attack*: targets delay the PoW blockchain network by spamming the blockchain network with null or meaningless transactions. When the number of transactions per second in the Ethereum network suddenly hits the top, a mining node may face two following issues, i.e., too much traffic (similar as that of DoS), and the queue of pending transactions is full. It equates to the unnecessary time burden for mining process and block propagation [119]. In 2017, the Bitcoin mempool size was exceeded 115,000 unconfirmed transactions which led to $700 million worth of transaction stall [117]. In our work, FoT attack is implemented by continuously sending a large number of transactions to an existing smart contract.

In order to capture traffic data of these attacks, we build a dataset collection tool, named BC-IDS, which inherits the core of an open-source utility named "kdd99 feature extractor" [120] and our new designs to fit the considered Ethereum network, i.e., correct the service of packets related to Ethereum nodes, remove meaningless features, and automate label dataset samples based on some given properties. Especially, our goal is to achieve a trained model that can be applied to our proposed real-time blockchain attack detection system, when the number of samples in a prediction frame is limited. Thus, the BC-IDS collects the dataset frames in which each frame lasts for 2 seconds and extracts their features. The BC-IDS then puts all collected data in this frame into a single file. Finally, we merge all single files together to make the full dataset. In summary, Table 20 shows 21 features in the designed dataset, which are separated into two types, i.e., discrete (D) and continuous (C). For continuous features, they are calculated in 2 seconds time window (similar to that of the famous KDD99 dataset [121]).

In each Ethereum node, the separated dataset is collected in four states (classes), i.e., normal state (Class-0), BF attack (Class-1), DoS attack (Class-2), and FoT attack (Class-3). The normal state is captured in two hours, the rest of them in an hour through the designed BC-IDS tool. Note that, when a node is attacked, the normal traffic still exists. Therefore, the attack samples can be filtered out by features-based two properties, i.e., the *src_ip* and *dst_ip* of the attack device, *service*, *src_length*, and *dst_length* of the samples, which are analyzed by Wireshark [110]. To improve the diversity of the designed dataset, the normal traffic data in the attack state is mixed with traffic data

Table 21: The number of samples in the designed dataset.

Ethereum node / Class	Node-1 (samples)	Node-2 (samples)	Node-3 (samples)
Normal	30,000	30,000	30,000
BP	3,000	3,000	3,000
DoS	3,000	3,000	3,000
FoT	3,000	3,000	3,000

in the normal state. In our experiments, a number of random samples in each state are selected to reduce the size of the bulk dataset as shown in Table 21. In fact, we mix normal traffic data in an equal ratio, i.e., 7,500 samples per normal state, normal traffic data at BF, DoS, FoT, respectively.

Fig. 29 illustrates the visualization of our designed dataset using t-Distributed Stochastic Neighbor Embedding (t-SNE) [122] with three most important components. In 3D view, the DoS and FoT attack samples show a fairly clear separation from normal state points. Otherwise, the BP attack samples collide with the normal state samples. This indicates that discriminating BP samples from the normal data points would be significantly challenging.

Evaluation Method

The confusion matrix with accuracy, precision and recall proposed in [62, 63] is widely used to evaluate the performance of many machine learning systems. We denote TP, FP, TN, FN to be "True Positive", "False Positive", "True Negative", and "False Negative", respectively. The accuracy of total system with T classes of normal behaviors and different types of attacks as follows:

$$Accuracy = \frac{1}{T}\sum_{t=1}^{T}\frac{TP_t + TN_t}{TP_t + TN_t + FP_t + FN_t}. \tag{32}$$

In the next section, we also use the accuracy, precision, recall to evaluate and compare the performance of our proposed Collaborative Learning model (proposed CoL) with two other baseline methods, i.e., Centralized Learning model (CeL) and Independent Learning model (IL).

Simulation Results

In this section, we use the collected datasets of three nodes described in aforementioned section for the corresponding LNs. The dataset of each LN is randomly split into

Table 22: Simulation results.

	2 Learning Nodes (LNs)				3 Learning Nodes (LNs)				
	Proposed CoL	CeL	IL		Proposed CoL	CeL	IL		
			LN-1	LN-2			LN-1	LN-2	LN-3
Accuracy	**97.780**	97.923	96.444	96.175	**97.609**	97.507	96.034	95.788	95.225
Precision	**95.497**	95.946	92.713	93.145	**95.317**	94.894	91.630	92.514	90.583
Recall	**95.560**	95.846	92.889	92.350	**95.219**	95.014	92.068	91.575	90.450

training and testing dataset. All LNs use DBN with the same structure of neural network for learning and detecting attacks. However, the LNs have to work in different learning models in various scenarios. Each LN has itself training and testing dataset, and thus we can use these datasets to evaluate and compare the performance of the proposed CoL, the CeL and the IL in different scenarios.

We present the simulation results with the dataset of LNs in different learning models. The details of datasets using for simulation are as follows:

- **Proposed Collaborative Learning Model (proposed CoL)**: Each LN learns its training dataset and performs collaborative learning with other LNs to generate the global model. Then, we use the global model to test the merged testing dataset of all participated LNs.
- **Centralized Learning Model (CeL)**: The centralized node (e.g., one of the mining node in the network) is assumed to be able to collect data from all the nodes in the network and train the deep learning model on the collected datasets. Then, we use the trained model to test data based on the merged testing dataset of all participated LNs.
- **Independent Learning Model (IL)**: Each LN learns its training dataset without sharing knowledge with other LNs. Then, we use this model to test data based on the merged testing dataset of all participated LNs.

Fig. 30 describes the convergence of the proposed CoL, the CeL and the IL (in terms of accuracy) of three LNs in the training process. The proposed CoL is obtained at the LN-1 after obtaining the global model. The CeL has a large number of training samples from three LNs so it can reach the convergence with around 97% accuracy after 300 epochs. Besides, the proposed CoL and the IL converge nearly the same after 750 epochs. After 10,000 learning epochs, we can observe that the proposed CoL has a higher accuracy compared with that of the IL (i.e., 98% vs 96%). The reason is that the proposed CoL can obtain the exchange knowledge from DL models of other LNs. Thereby, it can achieve similar performance as that of the CeL.

Table 22 presents the simulation results in two cases, i.e., two participated LNs and three participated LNs. In both cases, we can observe the same trend when the accuracy,

Table 23: Real experimental results.

	2 Learning Nodes (LNs)				3 Learning Nodes (LNs)					
	Proposed CoL		CeL		Proposed CoL			CeL		
	LN-1	LN-2	LN-1	LN-2	LN-1	LN-2	LN-3	LN-1	LN-2	LN-3
Accuracy	**97.520**	**97.342**	97.164	97.186	**97.582**	**97.383**	**96.693**	96.709	96.976	96.429
Precision	**95.796**	**95.830**	95.765	95.926	**96.253**	**96.081**	**94.718**	93.985	94.662	93.172
Recall	**95.040**	**94.685**	94.328	94.373	**95.164**	**94.765**	**93.387**	93.418	93.951	92.859

precision and recall of the proposed CoL are higher than those of the IL and nearly equal to those of the CeL. In particular, the accuracy of the proposed CoL is higher than that obtained by LN-1 in IL (approximately 1.5%) and the precision of the proposed CoL is nearly 4% higher than that obtained by LN-1 in IL in the case of three participated LNs. These results demonstrate that the proposed CoL can exchange knowledge with other LNs to improve its ability of detection, so it can achieve better performance in classifying attacks in the blockchain network than those of the IL. It also demonstrates that the learning model of IL should not be used to classify the data of other LNs. In addition, without sharing LN's dataset to a center node for training (e.g., a cloud server), the proposed CoL can achieve nearly the same accuracy as those of the CeL in all the scenarios.

Experimental Results

In this section, we present the experimental results obtained through real-time experiments at our laboratory. In this experiment, each blockchain node is installed a learning model to become an LN. Each LN learns its local dataset and then performs real-time attack detection in the blockchain network. We consider the scenario of two LNs and three LNs with the proposed CoL and the CeL. In the training process, the proposed CoL and the CeL are fed with the similar datasets as explained in the previous section. We then implement the trained model to all the participated LNs to perform real-time attack detection for both learning models in the testing process.

Real-time capturing and processing

In a real-time system, the cyberattack detection system continuously receives a number of the Ethereum network traffic data. Therefore, the system has to perform capturing, collecting frames, extracting features, analyzing and predicting within a period of time, i.e, 2 seconds. Fig. 31 shows the timeline of the cyberattack detection program. The data frame is exploited by our feature extractor function (Fe_Ex) of BC-IDS tool, and this is input for the trained model to predict (Pred) and classify packets to be normal or attack. All processes have to complete in 2 seconds before the next data frame of IP packets coming. To verify the predicted results from the trained model, all frames and

prediction results are stored. These frames are merged into a full validation dataset and labeled by own designed BC-IDS. After that, these ground truth labels are compared with the prediction results to obtain a validation report.

Performance Analysis

Table 23 presents the experimental results of the proposed CoL and the CeL with different participated LNs. We obtain the same trends as those of the simulation results. The results obtained by two and three learning models of both proposed CoL and CeL are slightly lower than those of the simulations at about 0.2%. This is because each type of attack has different distributions of attack samples in a period of time. Table 24 presents the number of samples of each class collecting in 15 minutes. In this table, we can observe that Class-1 has a very small number of samples during this period, this can lead to a low accuracy in statistics for this class and reduce the total accuracy of the model. However, our proposed CoL still has better performance than those of the CeL and LN-1 in the three LNs case (i.e., up to 0.8% of accuracy, 2.3% precision and 1.7% recall). Overall, our proposed CoL always achieves the best performance with approximately 97.5% accuracy, 95.7% precision and 95% recall with two LNs and 97.5% accuracy, 96.2% precision and 95.1% recall with three LNs. These results demonstrate that our proposed CoL can detect attacks with higher accuracies for all participated LNs than those of CeL.

Real-time Monitoring and Detection Fig. 32 illustrates the real-time monitoring of our proposed CoL for normal state and three types of attacks in the network. Fig. 32(a) is the normal state (Class-0) of the network with a high number of normal samples and a low number of attack samples. Then, the BP attack is performed, and Fig. 32(b) shows a slightly increase in the number of BP attacks. This is because, the number of BP attack samples is much smaller than other states in a period of time as in Table 24. In this case, the detection mechanism is activated and alarms the network under the BP attack (Class-1). Similarly, the DDoS attack in the network is described in Fig. 32(c) with a high increase in the number of samples of DoS attack. Finally, Fig. 32(d) describes the FoT attack. Unlike other attacks, the FoT attack includes a large number samples, thus it increases both the number of normal and attack samples (above 300 traffic samples per 2 seconds) than other attacks (under 50 traffic samples per 2 seconds). Thereby, in all the cases, we can observe that our proposed intrusion detection system can detect attacks effectively in a real-time manner.

Real-time Processing Capacity In this experiment, we fix the number of input samples in our proposed model to find the maximum real-time processing capacity in capturing and detecting attacks. Fig. 33 illustrates the real-time processing capacity of our proposed model. The processing time τ is counted from the time our proposed model

Table 24: The number of samples on Ethereum node 1 in an hour.

	Class-0	**Class-1**	**Class-2**	**Class-3**
Number of samples	242,298	828	10,858	128,509
Percentage (%)	63.347	0.216	2.839	33.598

reads the file containing the samples, until completing classification and producing the output. This work is repeated 20,000 times to determine the stability of the detection time of our proposed model. We vary the number of input samples multiple times to find the appropriate number that is adapted to the condition in Fig. 31. In most of the cases (98%), our proposed framework can classify 85,000 samples with less than 2 seconds. These results demonstrate that our proposed detection framework can be very potential to deploy to detect attacks in real-world blockchain networks.

Conclusion

In this work, we have proposed a novel collaborative learning framework for a cyber-attack detection system in a blockchain network. First, we have implemented a private blockchain network in our laboratory. This blockchain network is used to (1) generate data (both normal and attack data) to serve the proposed learning models and (2) validate the performance of our proposed learning framework in real-time experiments. After that, we have proposed a highly-effective learning model that allows to be effectively deployed in the blockchain network. This learning model allows nodes in the blockchain can be actively involved in the detection process by collecting data, learning knowledge from their data, and then exchanging knowledge together to improve the attack detection ability. In this way, we can not only avoid problems of conventional centralized learning (e.g., congestion and single point of failure) but also protect the blockchain network right at the edge. Both simulation and real-time experimental results then have clearly shown the efficiency of our proposed framework. In future, we plan to continue developing this dataset with other emerging types of attacks and develop more effective methods to protect blockchain networks.

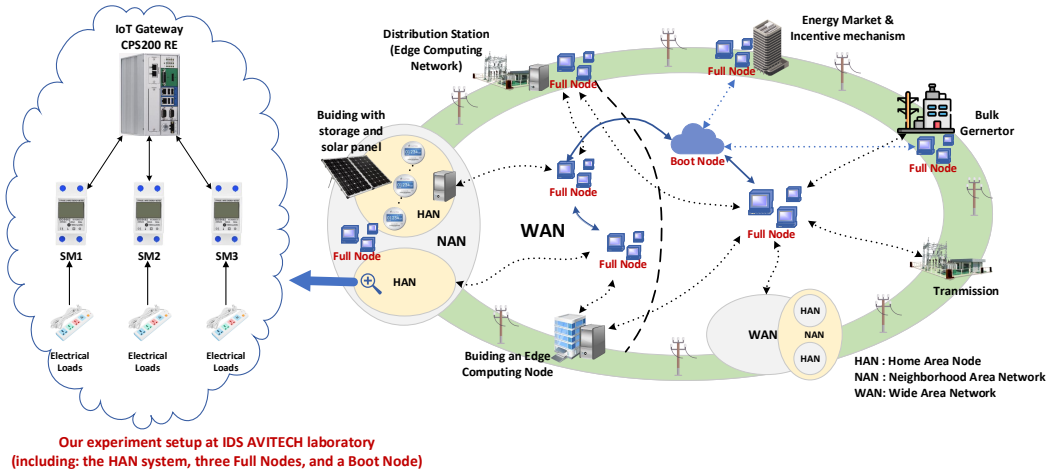


Figure 20: Model of a smart grid in a smart city.

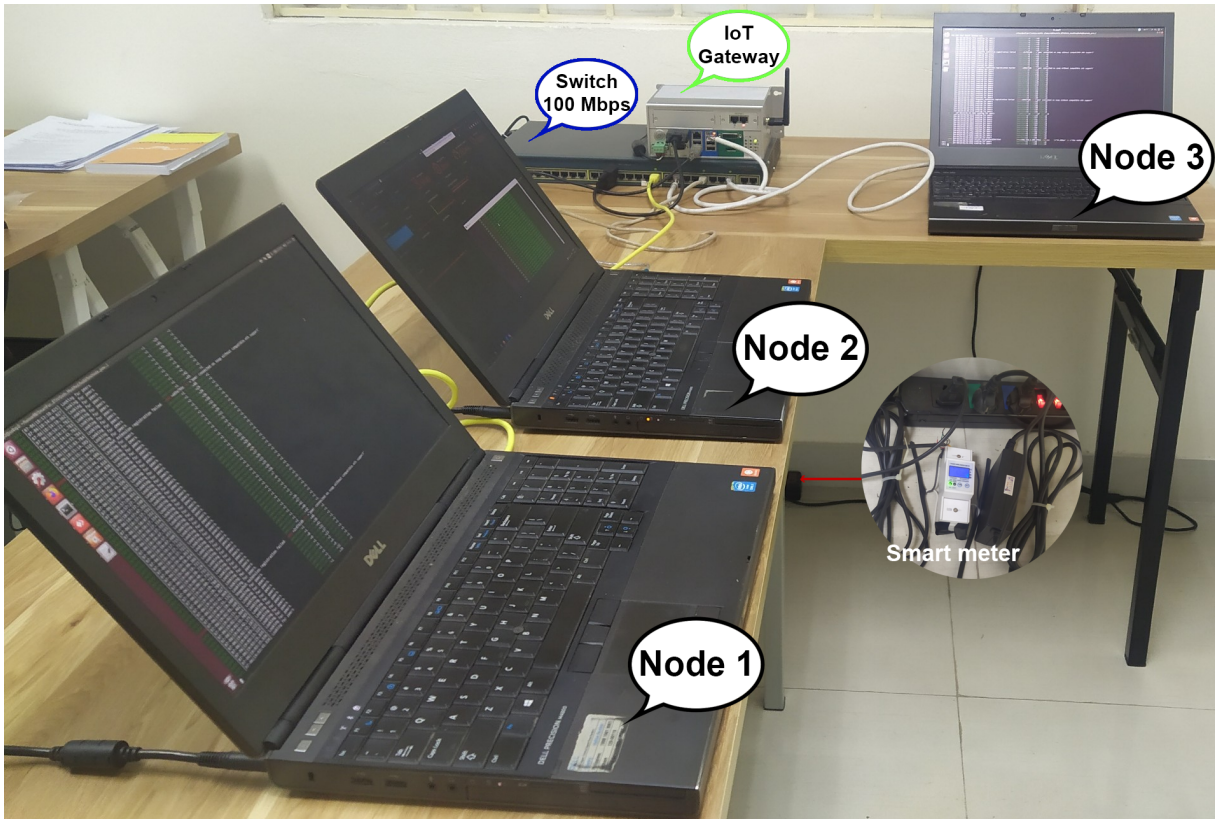


Figure 21: Our experiment system.

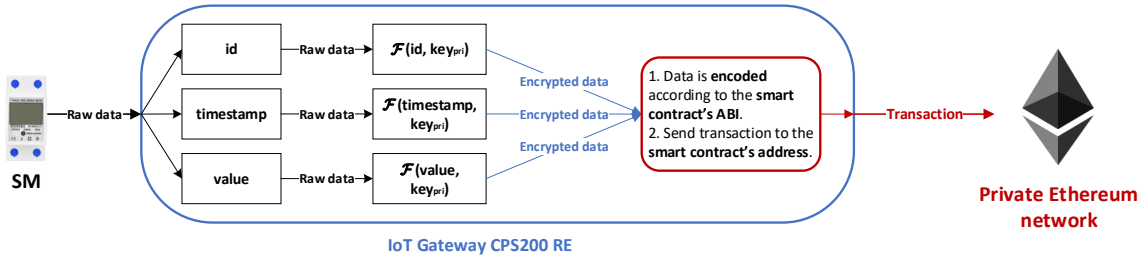


Figure 22: Data flow from a smart meter to the private Ethereum network.

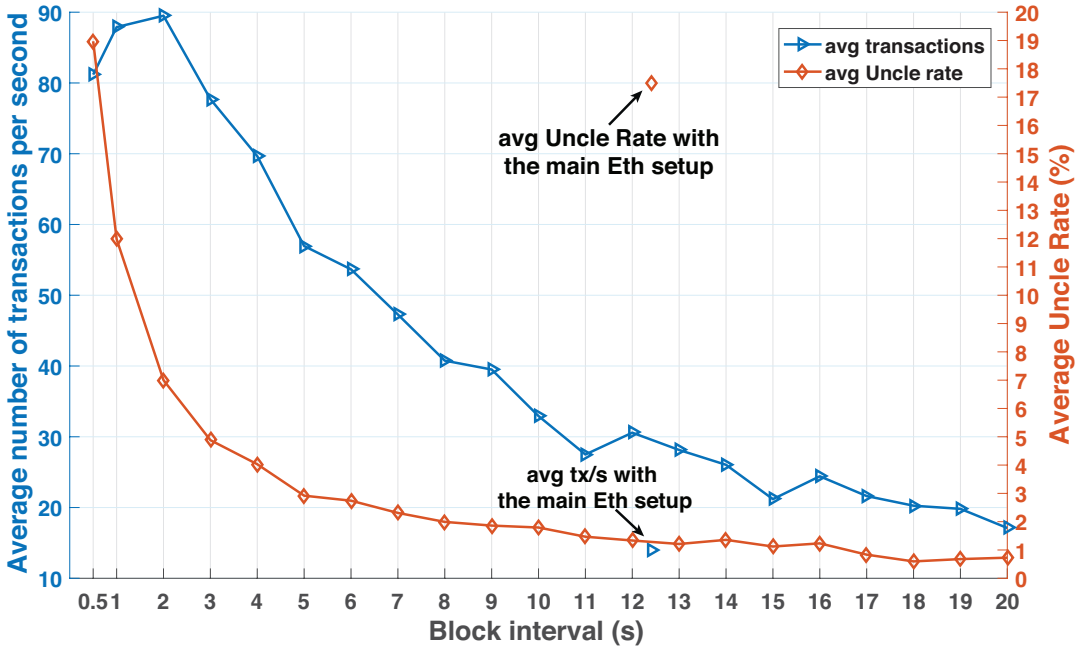


Figure 23: Performance by simulation: Throughput and uncle rate versus block interval.

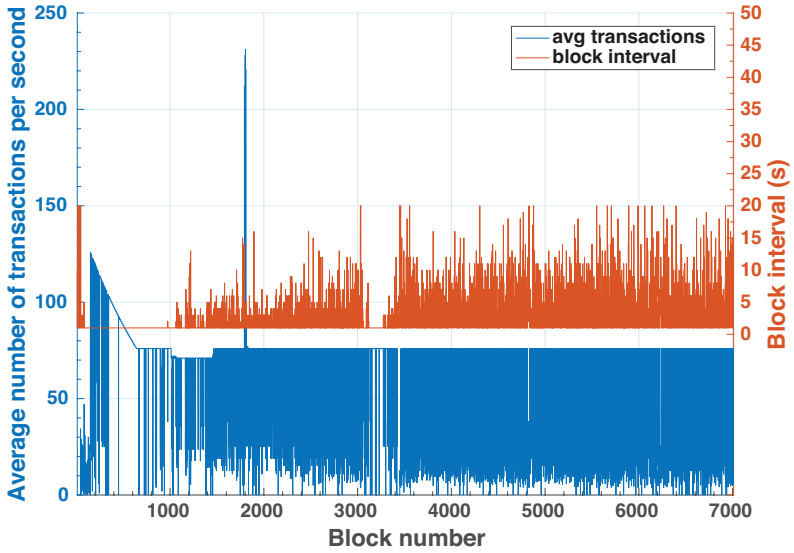


Figure 24: Real experiment: Throughput and block interval in $7,000$ blocks at $\lambda = 3$.

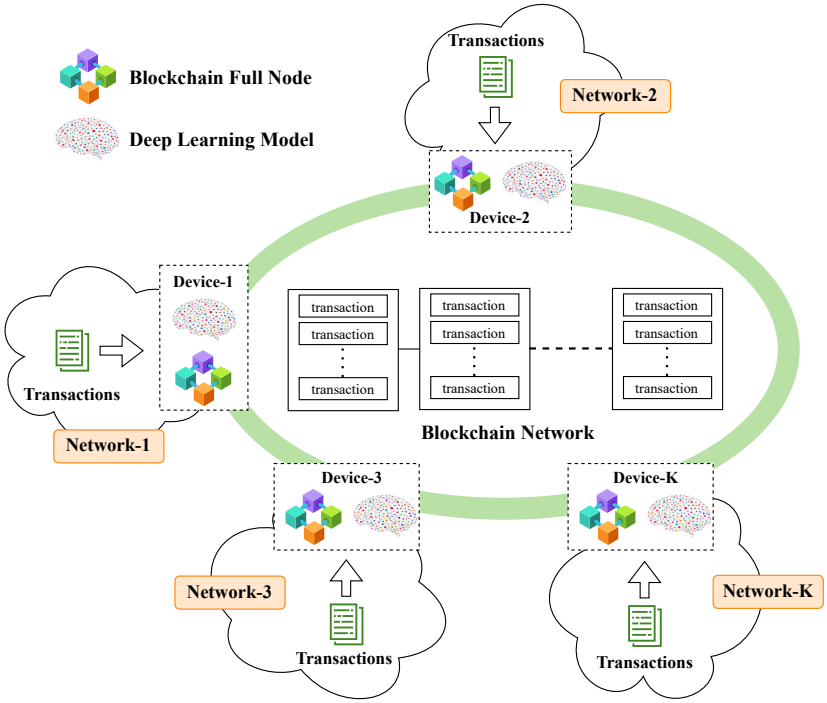


Figure 25: Proposed collaborative learning model for blockchain network.

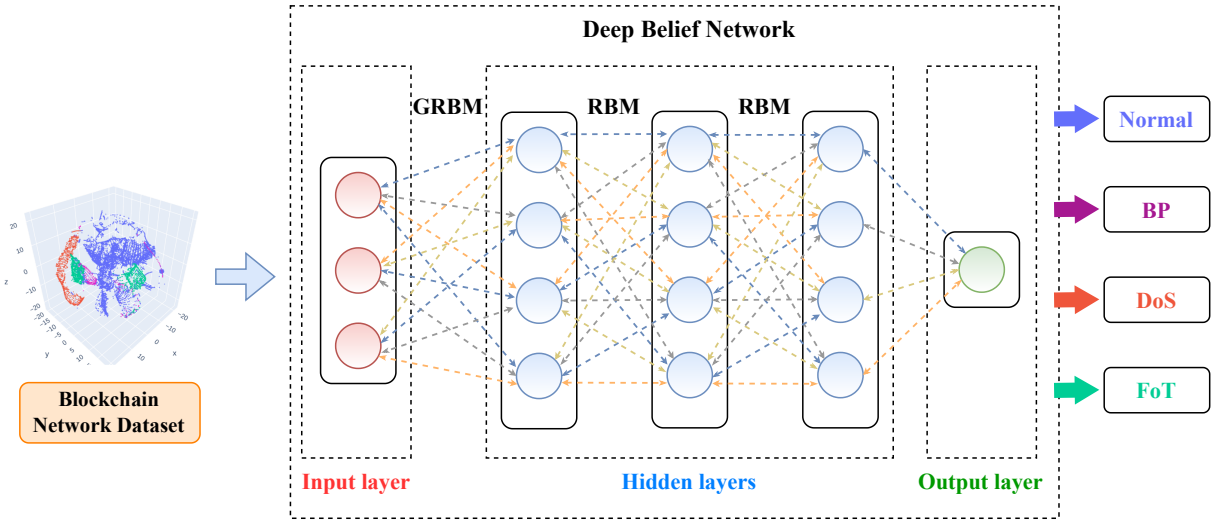


Figure 26: The structure of classification-based for intrusion detection learning model for the blockchain network.

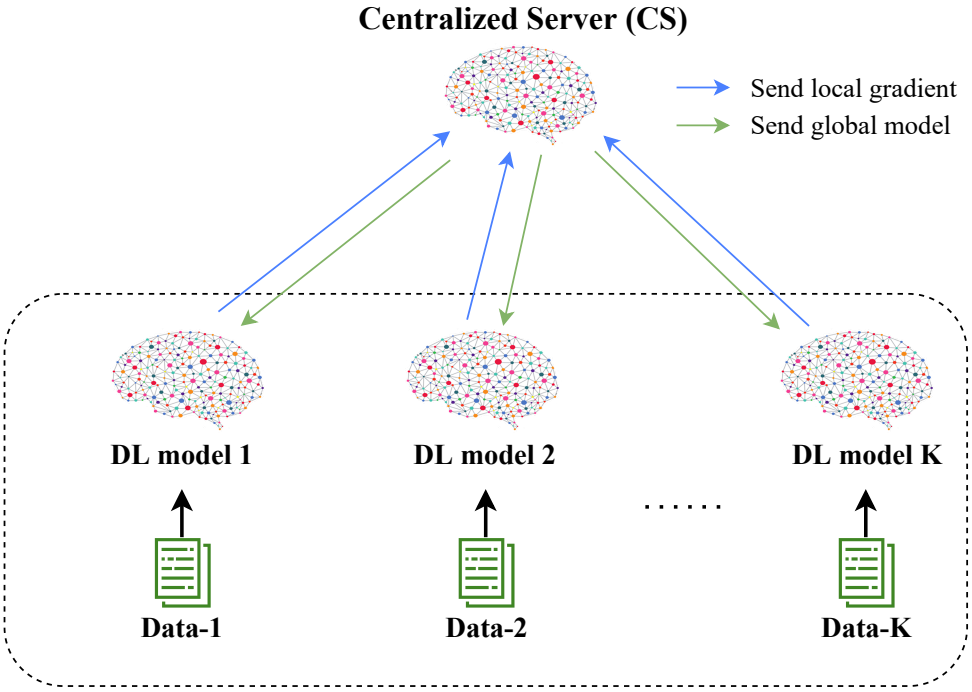


Figure 27: The illustration of the collaborative learning between DL models and the CS.

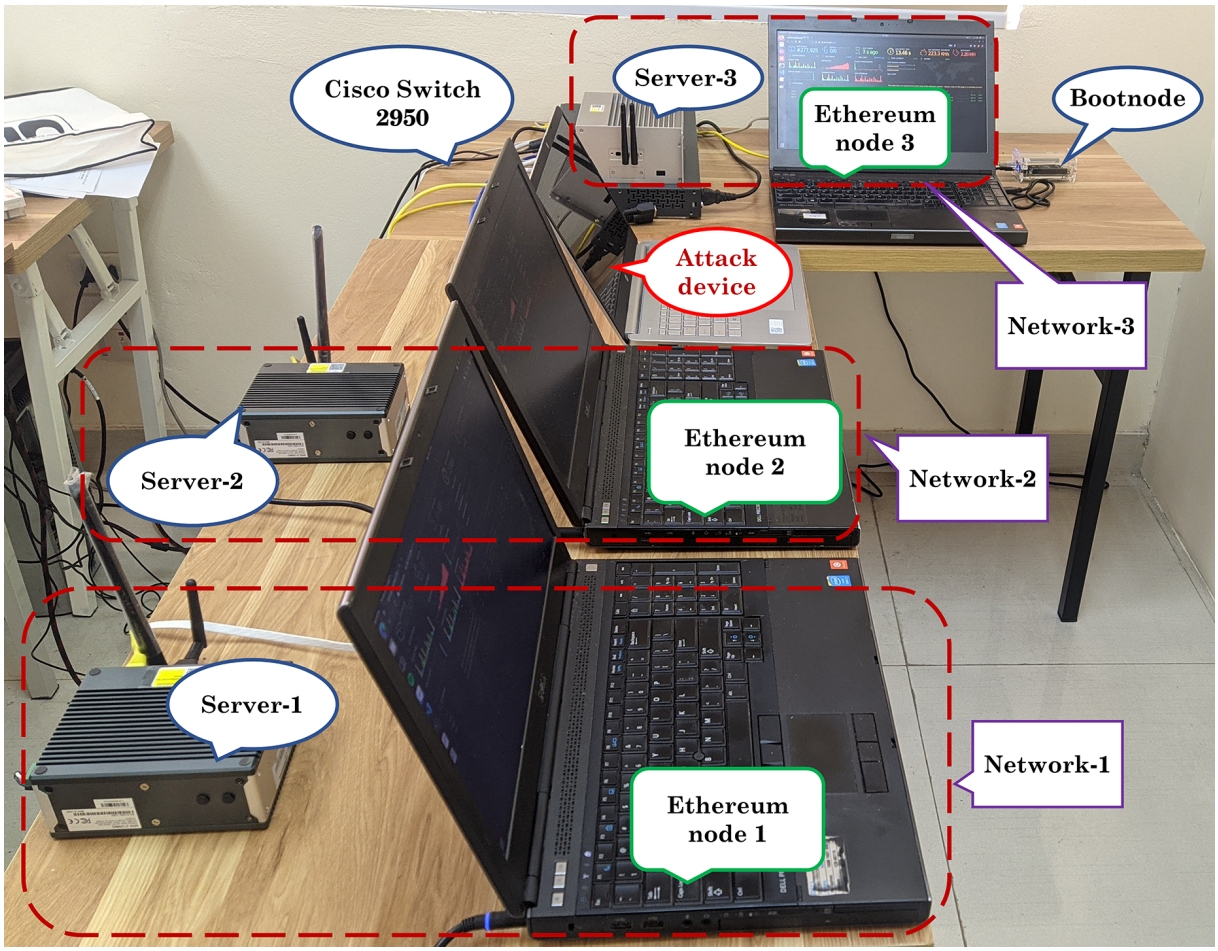
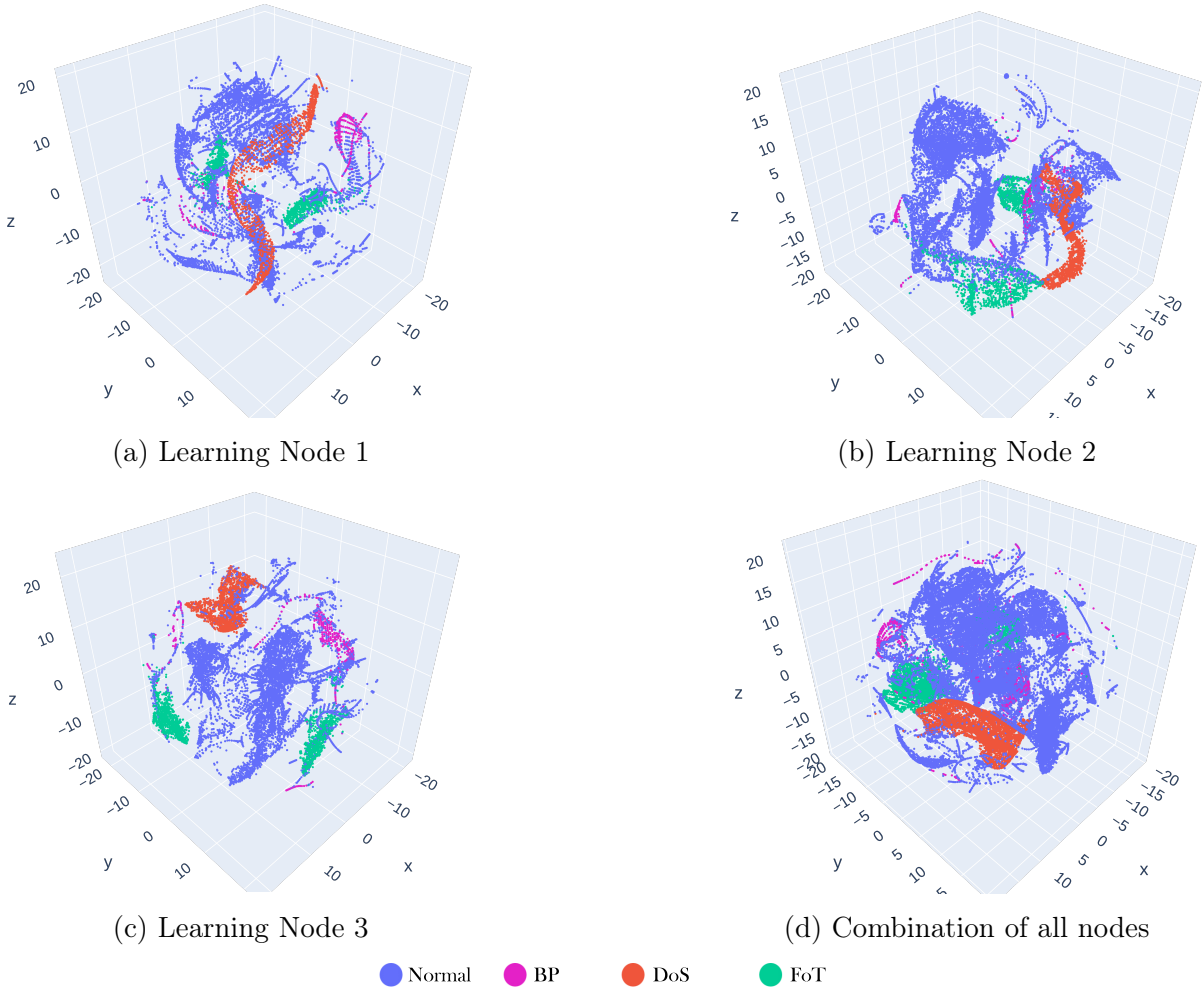


Figure 28: Experiment setup.



(a) Learning Node 1

(b) Learning Node 2

(c) Learning Node 3

(d) Combination of all nodes

Figure 29: Visualization using t-SNE for collected datasets.

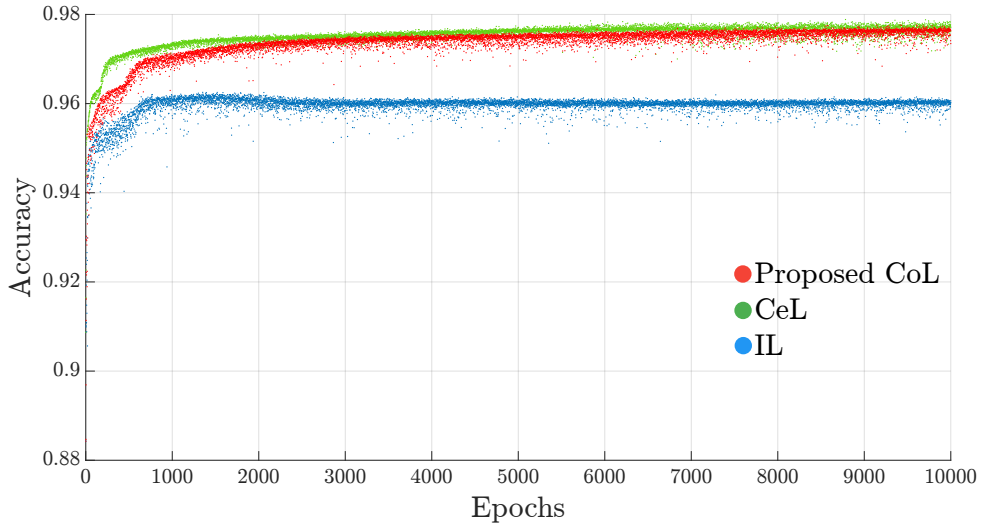


Figure 30: Training process of considered learning models.

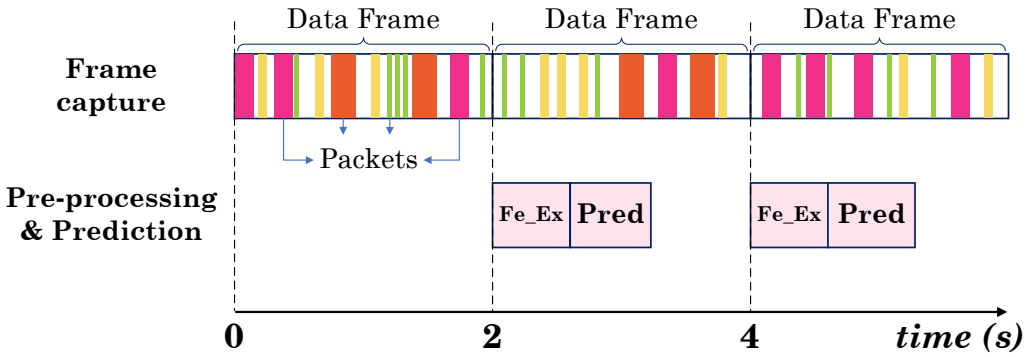


Figure 31: Timeline of verification phase.

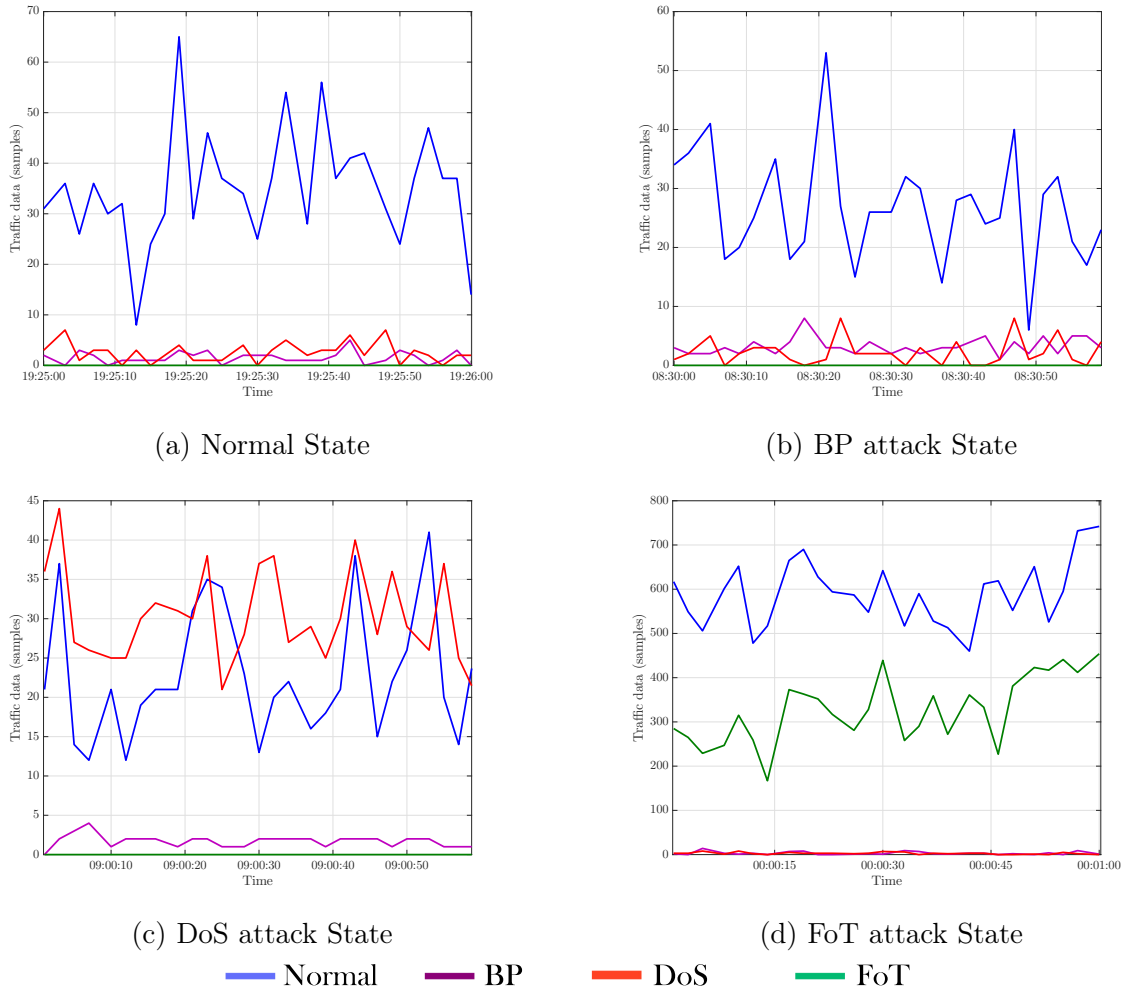


Figure 32: Real-time blockchain cyberattack detection: proposed CoL model of 3 LNs in Ethereum node 1.

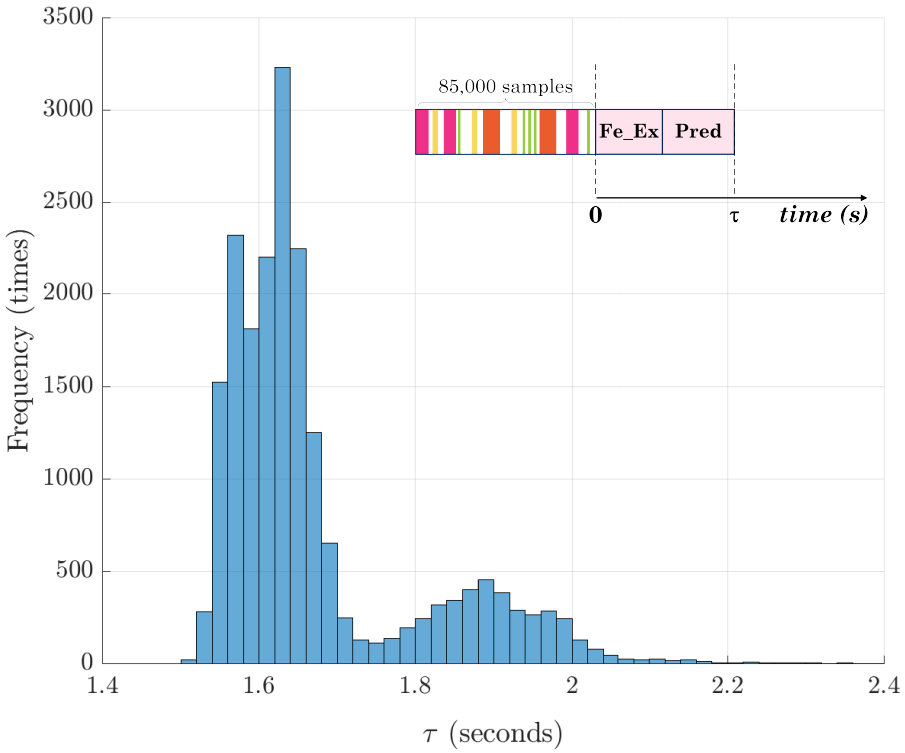


Figure 33: Histogram of real-time detection duration for 85,000 samples.

PROBLEM 5: Learning-based Friendly Jamming with Imperfect CSI for Security in MIMO Wiretap Channel

Introduction

Internet-of-Things (IoT) plays an essential role as a bridge to provide connectivity and data collections in many areas such as manufacturing, healthcare, etc. When the data exchange increase rapidly, it also raises cybersecurity issues such as data breach, man in the middle, eavesdropping, etc. There are two main challenges related to IoT security. Firstly, most IoT devices are wireless with a broadcast nature and thus more vulnerable to security attacks than wire-line devices. Secondly, IoT devices have limited capability in terms of computation and power, requiring low-complexity security solutions.

Conventional security solutions based on encryption techniques require infrastructure for the distribution of keys and the irreversibility of the underlying encrypted function such as Advanced Encryption Standard (AES), Rivest-Shamir-Adelman (RSA). However, those techniques rely on the computation limit of eavesdroppers, which may not guarantee perfect secrecy when eavesdroppers have a sufficient computational capability [123]. Hence, when legitimate transceivers have limited capabilities, such as in the case of IoT communication, as compared to eavesdroppers, legitimate communication links can be easily exploited by eavesdroppers.

Another solution, called Physical Layer Security (PLS), has been well developed to secure communication [124] based on random features of wireless channels. Two main PLS techniques are Key and Signal-to-Noise Ratio (SNR) based techniques as reviewed in [125]. The former extracts channel state information (CSI) to generate random keys that can be used for authentication purposes. The latter includes channel coding, channel-based adaptation, and artificial noise-based security. In this work, we consider an artificial noise technique or friendly jamming (FJ), wherein the transmitter uses a proportion of power to transmit a noise signal to degrade the eavesdropper's SNR while not impacting the intended receiver's SNR. Therefore, perfect secrecy can be achieved because there always exists a communication rate called secrecy capacity which guarantees that no information is leaked to the eavesdropper [126].

Most previous FJ studies (see for examples, [127–129]) assume that channel estimation is perfect; however, this assumption is not practical. To deal with ImCSI, a method in [130] uses second-order perturbation decomposition; however, this method requires high computation when the number of antennas at the transmitter, receiver and eavesdropper increase. Therefore, it is challenging to find a lightweight solution that is suitable for scalable IoT networks and can maintain secrecy performance under ImCSI.

As discussed above, FJ-based methods are highly sensitive with ImCSI. Therefore, we focus on dealing with the problem by incorporating a well-known deep learning model called autoencoder (AE) with the conventional FJ approach. In MIMO communication,

AE aims to learn a representation (*i.e.* encoding) for a set of data and, from the encoding, reconstructs a representation at the output close to the input[131, 132]. Our target is to perform secrecy optimization in that learning process for FJ-based communication security.

An AE-based security method over the Gaussian wiretap channel was proposed in [133]. In this work, Eve is assumed to be equipped with a NN that can cluster the constellation points with high probability. A coding scheme was proposed to make Eve suffer a high block error rate (BLER). However, this method will be undermined when the noise level at Bob is higher than that at Eve. In [134], using AEs to design finite block length wiretap codes was proposed. Also, a multi-objective programming function was proposed to simultaneously minimize the leakage of information to Eve and the BER at Bob. This method has high complexity because the number of needed parameters grows with the code parameters. Using AE for the MIMO Gaussian wiretap channel was demonstrated in [135]. However, the channel matrices at Bob and Eve and the number of antennas at Eve must be known as well. Besides, the learning in the above AE-based methods for security must take into account all the communication blocks (encoder, channel, decoder, etc.) from the input to the output (*i.e.* end-to-end learning) [136]. If there is a way to apply the AE for some but not all blocks, the training time can be shortened, especially, the method's design can be more flexible.

Secrecy capacity is defined as the difference of mutual information (MI) between the legitimate and illegitimate channels [124]. When the Eve's CSI is uncertain, then MI between Alice and Eve is not estimated accurately. Thus, we can only optimize MI between legitimate transmit and receive signals that can enhance security performance. Recently, mutual information neural estimation (MINE) has been proposed by Belghazi *et al.* in [137] and proved to very efficiently optimize MI. From this seminal work, MINE was applied to channel coding [138] by incorporating the MI maximization of channel inputs and outputs into the encoding process. The performance of this method is comparable with that of end-to-end learning. It is worth noting that only the learning process is not sufficient to guarantee the efficiency of the learning-based secrecy capacity. The authors in [139] prove that combining AE and MINE can provide better transmission performance, including decoding error and throughput. Another approach based on beamforming leverages the training process to produce beamformer [140] that can directly optimize channel capacity.

Inspired by the works in [130, 133, 137, 139], and [140], we have proposed a new FJ method for PLS using AE in case of ImCSI in [141] and, in this work, largely extend it. However, unlike the approaches based on CSI or channel estimation's error to cancel out FJ at receiver, ours allows the intended transmitter (Tx) and receiver (Rx) learning to null out the FJ signals while only degrading the eavesdropper channel. Further, we leverage MINE to secure the AE-based communications when the channel's distribution is unknown. This method can provide comparable secrecy performance to that in [127].

Our work has three main contributions as follows:

- We leverage the generalization features of neural networks to develop a MIMO FJ scheme that is robust to ImCSI due to issues such as time varying channels or the limited number of pilots [142]. Since DL-based communications frameworks have powerful generalization ability concerning the input data sets, our proposed method shows the competitive secrecy capacity compared to the conventional ones when the channel varies or with CSI errors. For that, secrecy optimization will be embedded into the learning process in cases of ImCSI.
- We leverage MINE-based FJ to demonstrate that it is possible to achieve a security performance comparable with the conventional FJ method without CSI. The training process is performed at the transmitter side to maximize the secrecy capacity between the sampled Tx and Rx signals. Compared to end-to-end learning like AE-based FJ, this security solution saves computation resources and saves energy consumption at the receiver. This method will be applicable for IoT devices that face resource constraints.
- This work also investigates the relationship between the MIMO secrecy optimization and detection tasks. In other words, maximizing secrecy rate and minimizing block/symbol error rate can be jointly optimized.

Notation: Vectors and matrices are denoted by bold lowercase and uppercase letters. The absolute value of a real number, the magnitude of a complex number and the complex conjugate transpose are, respectively, denoted by $\|\cdot\|$, $|\cdot|$ and $(\cdot)^\dagger$. A complex Gaussian random variable with mean μ and variance σ^2 is denoted by $\mathcal{CN}(\mu, \sigma^2)$. The expectation of a random variable X is denoted by $E[X]$.

Friendly Jamming

We first introduce the background of FJ, which is a baseline in this work. Further, AE-based MIMO communication and MINE is introduced because it is fundamental to our proposed method.

Multiple-Input-Multiple-Output (MIMO) FJ Scheme

In the MIMO FJ scenario, the number of antennas at Bob and Eve are N_T and N_E, respectively, are greater than 1.

The channel matrices at time k on Alice-Bob and Alice-Eve channel are $\mathbf{H}_k$ and $\mathbf{G}_k$ respectively. The elements of $\mathbf{H}_k$ and $\mathbf{G}_k$ are assumed to be i.i.d. and independent of

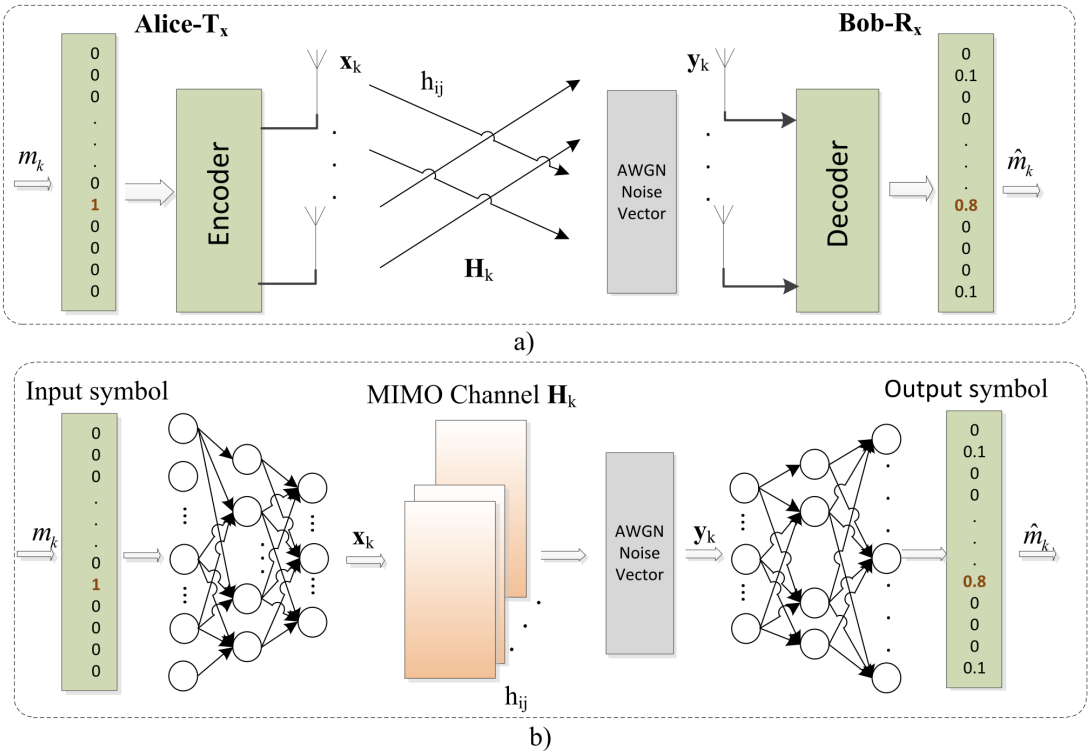


Figure 34: AE communication scheme for PLS.

each other and unchanged over a block of a large number of symbols. The received signals at Bob and Eve are presented as

$$\mathbf{y}_k = \mathbf{H}_k\mathbf{x}_k + \mathbf{n}_b = \mathbf{H}_k\mathbf{s}_k + \mathbf{H}_k\mathbf{w}_k + \mathbf{n}_b, \tag{33}$$

$$\mathbf{z}_k = \mathbf{G}_k\mathbf{s}_k + \mathbf{G}_k^{\dagger}\mathbf{w}_k + \mathbf{n}_e. \tag{34}$$

Conventional method [127, 143] assumes $\mathbf{H}_k$ is perfectly known at the Tx so the FJ signal $\mathbf{w}_k$ is chosen as $\mathbf{H}_k^{\dagger}\mathbf{w}_k = 0$ then $\mathbf{w}_k = \mathbf{Z}_k\mathbf{v}_k$. The elements of $\mathbf{v}_k$ are chosen to be i.i.d. complex Gaussian random variables with variance σ_v^2 which makes the elements of $\mathbf{w}_k$ are Gaussian distributed and $\mathbf{Z}_k$ is one of orthonormal matrix of the null space of $\mathbf{H}_k$.

To evaluate secrecy capacity, from (33) the covariance of noise at Eve is calculated as

$$\mathbf{K} = (\mathbf{G}_k\mathbf{Z}_k\mathbf{Z}_k^{\dagger}\mathbf{G}_k)\sigma_v^2 + \mathbf{I}\sigma_e^2. \tag{35}$$

Then the secrecy capacity C_s is presented as

$$\begin{aligned} C_s &\doteq I(A,B) - I(A,E) \\ &= \log(1+\mathrm{SNR_B}) - \log(1+\mathrm{SNR_E}) \\ &= \log\left(det(\mathbf{I}+\mathbf{H}_k\mathbf{Q}_s\mathbf{H}_k^\dagger\right) - \log\left(\frac{det(\mathbf{K}+\mathbf{G}_k\mathbf{Q}_s\mathbf{G}_k^\dagger)}{det(\mathbf{K})}\right), \end{aligned} \tag{36}$$

where $\mathbf{Q}_s = \mathbf{E}[\mathbf{s}_k\mathbf{s}_k^\dagger]$. Since Alice-Eve's channel is not available, only the first term in (36) is maximized by SVD-based method, where $\mathbf{H}_k$ is decomposed as

$$\mathbf{H}_k = \mathbf{U}_k\mathbf{\Gamma}_k\mathbf{V}_k^\dagger.$$

After pre-coding $\mathbf{r}_k = \mathbf{V}_k^\dagger\mathbf{s}_k$, the received signal $\mathbf{y}_k$ is multiplied with $\mathbf{U}_k^\dagger$. Then, the equivalent channel of (33) becomes

$$\tilde{\mathbf{y}}_k = \mathbf{\Gamma}_k^\dagger\mathbf{r}_k + \tilde{\mathbf{n}}_b,$$

where $\tilde{\mathbf{y}}_k = \mathbf{U}_k^\dagger\mathbf{y}_k$ then the transmitter chooses $\mathbf{Q}_r$ as

$$\mathbf{Q}_r = \mathbf{E}[\mathbf{r}_k\mathbf{r}_k^\dagger] = \mathrm{diag}(\sigma_{r,1}^2, \sigma_{r,2}^2, ..., \sigma_{r,N_T}^2)$$

where $\sigma_{r,i}^2$ is founded by the water filling solution with power constraint $\mathbf{P}_{info} \leq \mathbf{P}$ corresponding to the largest singular values of H_k. Then, the secrecy capacity C_s is given as

$$C_s = \log\left(\det(\mathbf{I}+\mathbf{\Gamma}_k\mathbf{Q}_r\mathbf{\Gamma}_k^\dagger)\right) - \log\left(\frac{\det(\mathbf{K}+\mathbf{F})}{\det(\mathbf{K})}\right), \tag{37}$$

where $\mathbf{F} = \mathbf{G}_k\mathbf{V}_k\mathbf{Q}_r\mathbf{V}_k^\dagger\mathbf{G}_k^\dagger$. Since C_s is a random variable, the average secrecy capacity C_{asc} will be used. The objective function is

$$\overline{C}_{asc} \doteq \max_{\mathbf{E}[\mathbf{x}_k\mathbf{x}_k^\dagger]\leq P} E[\log\left(\det(\mathbf{I}+\mathbf{\Gamma}_k\mathbf{Q}_r\mathbf{\Gamma}_k^\dagger)\right) - \log\left(\frac{\det(\mathbf{K}+\mathbf{F})}{\det(\mathbf{K})}\right)], \tag{38}$$

where the power constraint $\mathbf{E}[\mathbf{x}_k\mathbf{x}_k^\dagger] \leq P$ can be rewritten as $\mathrm{trace}(\mathbf{V}_k\mathbf{Q}_r\mathbf{V}_k^\dagger + N_{FJ}\sigma_v^2) \leq P$, and N_{FJ} denotes the number of dimensions used for FJ transmitting. From equation (35), it can be seen that to guarantee $det(\mathbf{G}_k\mathbf{Z}_k\mathbf{Z}_k^\dagger\mathbf{G}_k)\sigma_v^2 \neq 0$, the transmitter must use at least N_E antennas for FJ signals while the remaining ones can be used for transmitting information signals.

Autoencoder-based MIMO Communications Scheme

To model the complex signals by NN networks, the complex parameter need to be re-parameterized to real-valued ones as follows:

$$\hat{\mathbf{x}}_{\mathrm{k}} = \left[\ \mathrm{R_e}(\mathbf{x}_{\mathrm{k}}), \mathrm{I}_m(\mathbf{x}_{\mathrm{k}})\ \right],\ \hat{\mathbf{y}}_{\mathrm{k}} = \begin{bmatrix} \mathrm{R_e}(\mathbf{y}_{\mathrm{k}}) \\ \mathrm{I}_m(\mathbf{y}_{\mathrm{k}}) \end{bmatrix}$$
$$\hat{\mathbf{n}}_b = \begin{bmatrix} \mathrm{R_e}(\mathbf{n}_b) \\ \mathrm{I}_m(\mathbf{n}_b) \end{bmatrix},\ \hat{\mathbf{H}}_{\mathrm{k}} = \begin{bmatrix} \mathrm{R_e}(\mathbf{H}_{\mathrm{k}}) & -\mathrm{I}_m(\mathbf{H}_{\mathrm{k}}) \\ \mathrm{I}_m(\mathbf{H}_{\mathrm{k}}) & \mathrm{R_e}(\mathbf{H}_{\mathrm{k}}) \end{bmatrix}, \tag{39}$$

where $\hat{\mathbf{x}} \in \mathbb{R}^{2N_T}$, $\hat{\mathbf{y}} \in \mathbb{R}^{2N_R}$ and $\hat{\mathbf{H}} \in \mathbb{R}^{2N_R \times 2N_T}$ are the transmitted received vectors, and the channel matrix respectively with the real elements. The relationship between input and out put in (33) becomes [144]:

$$\hat{\mathbf{y}}_k = \hat{\mathbf{H}}_k^{\dagger} \hat{\mathbf{x}}_k + \hat{\mathbf{n}}_b. \tag{40}$$

Autoencoder based MIMO communication

The conventional MIMO and AE-based MIMO communication models are presented in Figure 34a as the MIMO channel $\mathbf{H}_k$ in Figure 34b, respectively. The flat Rayleigh fading as the channel distribution is used in our implementation. At time k, the message $m_k \in M = \{1, 2, \ldots, M\}$ is encoded into the transmitted vector $\mathbf{s}_k$. The power constraint is guaranteed by the normalization layer. The receiver blocks at Bob are based on the model in [134] with the last layer using the softmax function. This function gives a probability distribution $\hat{1}_m \in (0,1)^{\mathrm{card}(M)}$ over all of messages (card denotes cardinality), which is fed into the cross-entropy loss function. Then the maximum likelihood is used to estimate the sent signal [145] by using the cross-entropy loss function to optimize signals reconstruction error [146]. Hence, the index of the element of $\hat{1}_m$ with the highest probability will be the decoded symbol. In other words, the learning process is an optimization process in which the reconstruction error of the inputs is minimized.

MINE for Secrecy Evaluation

The concept of MINE is proposed in [137] that use to estimate the MI between two random variable X and Y without their knowledge of distribution functions. The MI between X and Y is denoted as

$$I(X,Y) = D_{KL}(P(X,Y) \parallel P(X) \otimes P(Y)), \tag{41}$$

where $D_{KL}(P(X,Y)$ is Kullback-Leibler divergence between the joint probability $P(X,Y)$ and the product of the marginal probability $P(X) \otimes P(Y)$. In [137], Donsker-Varadhan

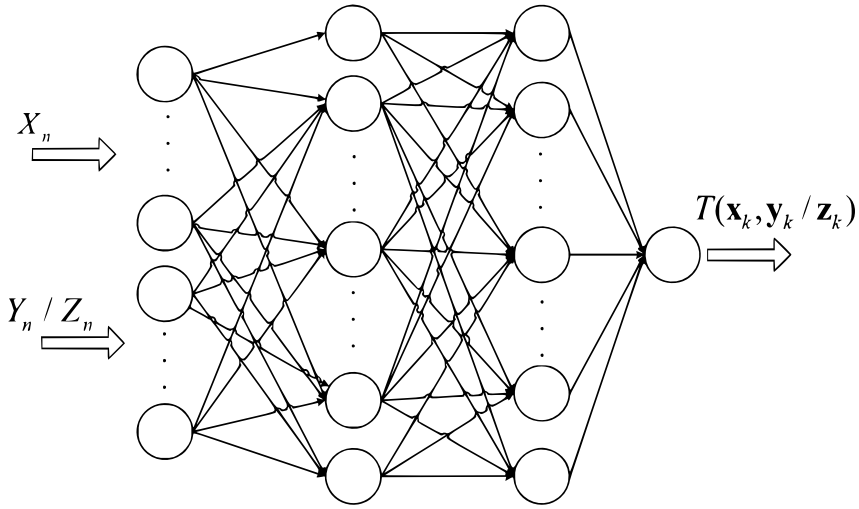


Figure 35: MI estimator.

representation was applied to represent the KL divergence as follows:

$$D_{KL}(P \parallel Q) = \sup_{f:\Omega \to R} E_P[T] - \log(E_Q[e^T]), \tag{42}$$

where the supremum is taken over all classes of the function f such that the expectation is finite. By choosing the function class, the term on the right hand side of (42) yields an optimal lower bound on the KL-divergence. In MINE, a deep neural network, called statistics network, is chosen as the function class, called $T_\theta : X \times Y \to \mathbb{R}$ with parameters $\theta \in \Theta$. We then have the following lower bound on KL-divergence:

$$D_{KL}(P \parallel Q) \geq \sup_{T \in F} E_P[T] - \log(E_Q[e^T]). \tag{43}$$

Since the inequality $I(X,Y) \geq I_\Theta(X,Y)$ [137], where $I_\Theta(X,Y)$ denotes the mutual information measure defined as

$$I_\Theta(X,Y) = \sup_{\theta \in \Theta} E_{P_{XY}}[T] - \log(E_{P_X \otimes P_Y}[e^T]), \tag{44}$$

the estimated mutual information between X,and Y is gained by maximizing $I_\Theta(X,Y)$ in (44). The MI neural estimator T_θ includes two fully connected hidden layers, and a linear output node as illustrated on Figure 35, where the inputs are the samples from the joint distribution of $P(X_n, Y_n)$ or $P(X_n, Z_n)$, and marginal distributions $P(X)$, $P(Y)$ or $P(Z)$, respectively, then taking the samples of these distributions and approximate the expectations by the sample average. The marginal distribution of the input can be derived by shuffling the joint distribution along with the batch axis [137]. Hence, the estimated

MI for N samples is given as follows [138]:

$$I_\Theta(X_n, Y_n) = \frac{1}{N}\sum_{i=1}^{N}[T_{\theta_1}(\mathbf{x}_i^n, \mathbf{y}_i^n)] - \log\frac{1}{N}\sum_{i=1}^{N}[e^{T_{\theta_1}(\mathbf{x}_i^n, \bar{\mathbf{y}}_i^n)}]. \tag{45}$$

Similarly, the MI between Alice and Eve or the leakage information to Eve is estimated as

$$I_\Theta(X_n, Z_n) = \frac{1}{N}\sum_{i=1}^{N}[T_{\theta_2}(\mathbf{x}_i^n, \mathbf{z}_i^n)] - \log\frac{1}{N}\sum_{i=1}^{N}[e^{T_{\theta_2}(\mathbf{x}_i^n, \bar{\mathbf{z}}_i^n)}]. \tag{46}$$

Then the secrecy rate is then given by

$$C_s^{MINE} = I_\Theta(X_n, Y_n) - I_\Theta(X_n, Z_n). \tag{47}$$

Proposed Learning Based Friendly Jamming

Proposed Autoencoder based MIMO FJ Security Scheme

The objective of PLS is to guarantee that no information leakage to Eve while Bob can recover the message without errors [147]. For that purpose, we leverage the AE-based communications scheme presented in Figure 34 for the goal that is learning to guarantee secrecy. The proposed AE-based FJ communication and security scheme is illustrated in Figure 36. The FJ signal is injected into the information signal via the FJ generator layer, making the final transmitted signals $\mathbf{x}_k$.

The principle of security by FJ discussed in Section (1) is using a precoding technique to make the FJ signal orthogonal with the perfectly-known channel $\mathbf{H}_k$. However, since the channel coefficient is not perfectly known, we replace the precoding with the training process. More specifically, the encoder and decoder learn to cancel out FJ injected in the Tx signals. Once Alice and Bob have learned to maximize the likelihood between the input and output symbols, it will maximize secrecy capacity and minimize BLER. Note that the parameters in the hyper-parameters in channel/fading layers of Alice-Bob and Alice-Eve are set up to be i.i.d., respectively. By doing that, Alice and Bob learn to cancel out the injected FJ signals while Eve tries to decode the message simultaneously. Next, we will consider secrecy rate optimization and the security loss function to secure the communication.

AE-based Secrecy Capacity Optimization

A straightforward approach to optimize the secrecy capacity is computed via the difference of mutual information between Bob's and Eve's channel, as given in (38). Once the channel $\mathbf{H}_k$ is perfectly known, the optimum average secrecy rate achieved based

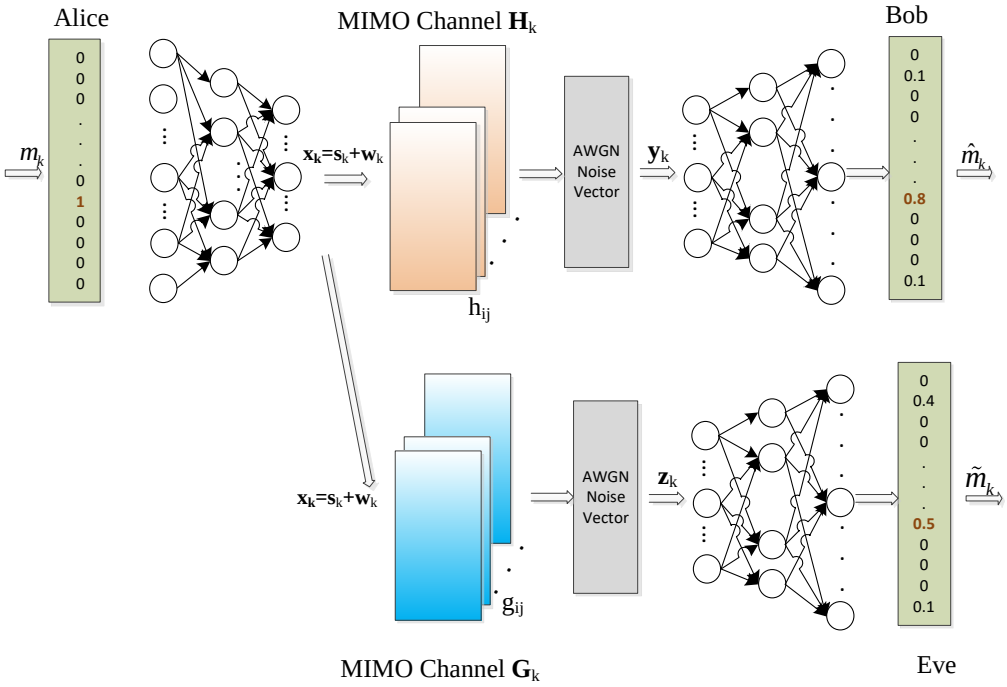


Figure 36: Autoencoder based Friendly jamming with an example of error decoding at Eve.

on maximizing channel capacity or $I(A, B)$ on the main channel and the pre-coded FJ signals $\mathbf{w}_k \in null(\mathbf{H}_k)$, which will degrade Eve's channel with the highest probability. On the other hand, since the channel coefficients are not perfectly known, the perfect null space of the channel is not available. So, it is impossible to find the optimum FJ signals that are canceled out completely on Bob's channel but only affect Eve's channel. In the AEFJ network, $I(A, B)$ can be maximized by minimizing the cross-entropy H_{AB}. When the true distribution $p(X/Y)$ is unknown, the encoding of X can be based on another distribution Y as a model that approximates P, which is represented by cross-entropy $H(X, Y)$. Further, the relationship between MI and cross-entropy can be represented via the KL divergence or relative entropy, which is the difference between the cross-entropy $H(X, Y)$ and the entropy $H(X)$, given as

$$\begin{aligned} KL(X||Y) = H(X,Y) - H(X) = & \quad -\sum_i x_i \log y_i - (-\sum_i x_i \log x_i) \\ = \sum_i x_i(\log x_i - \log y_i) = & \quad \sum_i x_i \log\left(\frac{x_i}{y_i}\right) \geq 0 \end{aligned} \tag{48}$$

The KL divergence represents a measure of the error of using Y to approximate X, in terms of the amount of information lost, due to the lack of channel information. In order to obtain the best model Y that optimally approximates X, we need to minimize $KL(X||Y)$. Further, the mutual information mentioned above can be expressed as the following KL divergence:

$$I(X,Y) = \sum_i \sum_j P(x_i, y_j) \log \frac{P(x_i, y_j)}{P(x_i)P(y_j)} \tag{49}$$

In [148], the mutual information $I(X,Y)$ can be rewritten as follow:

$$I(X,Y) = E_{(X,Y)} \log \left(\frac{P(y \parallel x)}{P(y)} \right) \tag{50}$$

where $E_{(X,Y)\sim P(X,Y)}$, and the variation bound [149] of $I(X,Y)$ via a variational distribution Q is presented as

$$I(X,Y) = E_{(X,Y)} \log(\frac{P(y \parallel x)}{P(y)}) \geq E_{(X,Y)} \log \left(\frac{Q(X,Y)}{P(X)P(Y)} \right). \tag{51}$$

That inequality holds since KL divergence maintains non-negativity. That lower bound is tight when $Q(X,Y)$ converges to $P(X,Y)$. The authors in [150] reparameterise and define $Q(X,Y)$ in terms of an unconstrained function $g(\phi)$. The authors in [151] prove that the infimum of the expected cross-entropy loss along with softmax function is equivalent to the mutual information between input and output variables up to constant $log(M)$ under uniform label distribution. We leverage that idea to maximize the lower bound of mutual information on the main channel model by using cross-entropy with the softmax function with the assumption that the inputs and corresponding labels have the same uniform distribution.

From that, the secrecy optimization can be transformed to the cross-entropy optimization between the transmitted and received signals. To do that, we construct a new cross-entropy loss function to optimize the MIMO autoencoder channel's parameter.

Security Loss Function

The idea of security loss functions is first used in [133]. In our work, cross-entropy loss function, as follow

$$L = (1-\alpha)H(p_A(s_k), p_B(s_k)) + \alpha H(p_A(w_k), p_B(w_k)) \tag{52}$$

where $p_A(s_k)$ and $p_B(s_k)$ is the probability mass function of the information signals, and $p_A(w_k)$ and $p_B(w_k)$ are the resulting probability mass functions of the FJ signals at Alice

Algorithm 5 MINE based Friendly Jamming Algorithm

At time k: Generate $\mathbf{w}_k$ which is orthogonal to $\mathbf{s}_k$
Alice transmits $\mathbf{x}_k = \mathbf{s}_k + \mathbf{w}_k$
Bob receives $\mathbf{y}_k$
for *i=1 to iteration* **do**
    Maximize the i^{th} MI_{ABi}, see Section (1)
    Save the estimated MI_{ABi}
    Load the MI_{ABi} as the loss to optimize the encoder weights by performing gradient descent steps
    $i = i + 1$
    If $MI_{AB_{i+1}}$ is not higher than MI_{AB_i}:
    end
end for

and Bob respectively. $H(.)$ denotes cross-entropy, and α is a parameter that controls the trade-off between security and the communication rate.

For that, minimizing $H(p_A(s_k), p_B(s_k))$ or maximizing the output probability of symbol s_{ik} will decrease the output probability of all other symbols at Bob. In contrast, maximizing $H(p_A(s_k), p_E(s_k))$ forces the system to reduce the output probability of the symbol s_{ik} and therefore randomly forces a higher probability on other symbols $s_{i\neq j}$. Hence, we can gain optimal secrecy capacity by neural optimization. More specifically, after the training process, the secrecy capacity is evaluated by both equations below. As discussed above, the AE classification task with softmax and cross-entropy loss function is equivalent to the maximum mutual information between the input and output. Hence, minimizing the loss function (52) will minimize the effect of FJ on Bob and maximize $I(A, B)$. By contrast, the eavesdropper's channel will be degraded with a high probability since Eve does not have the information about the FJ signals.

MINE-based FJ

Motivated by the work in [137], we leverage MINE for the two objectives: estimate secrecy capacity and maximize it by optimizing the encoder at Alice when CSI is not available at Alice in terms of channel distribution and samples. To deal with the problem, we leverage MINE to maximize the mutual information between Alice and Bob (MI_{AB}) while minimizing the mutual information between Alice and Eve (MI_{AE}) with the assumption that the number of the antenna at Alice is larger than that of Bob and Eve. The security framework is demonstrated as in Algorithm 5. At the time k, we assume that Eve has full knowledge of the CSI and the distance between Eve and Bob satisfy the assumption of $\mathbf{H_k}$ and $\mathbf{G_k}$ are ii.d. By doing that, we leverage MINE to optimize channel capacity and secrecy capacity simultaneously.

The structures of the encoder and MI estimator, I_Θ network, remain unchanged as

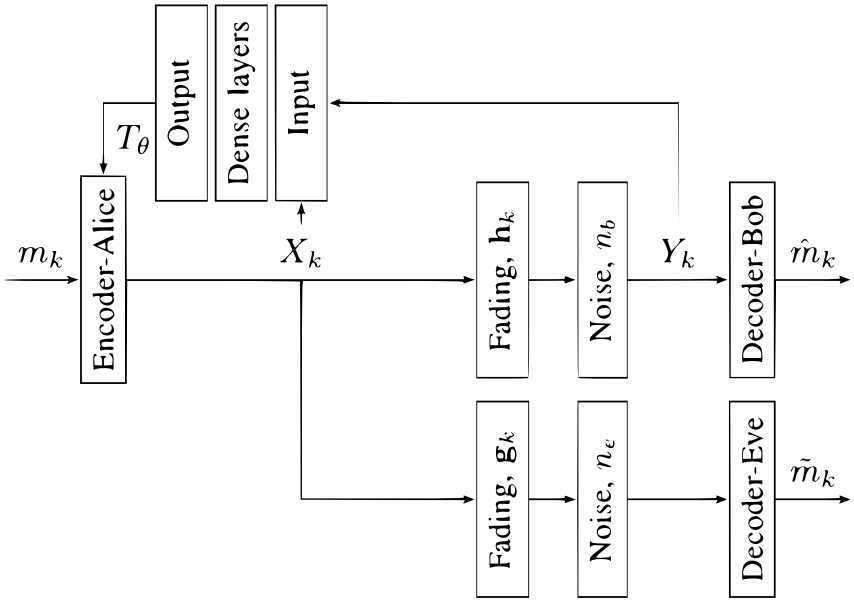


Figure 37: MI based FJ

described in Sub section (1). Regarding security purposes, MINE-based FJ network will be the combination of the MI estimator and encoder with the injected FJ signal. The main advantage of that is the learning works with the channel density distribution and rather estimates a function of the channel. Our security model is presented in Figure 37. To achieve the security requirement and reliable transmission, we use a new security loss function based on MI with the control coefficient β as follows:

$$L_{\text{MINE}} = \beta I_\Theta(X_n, Y_n) + (1-\beta) I_\Theta(X_n, Z_n), \tag{53}$$

From the equation (47) the loss function can be presented as

$$\begin{aligned} L_{\text{MINE}} = & \frac{\beta}{N}\sum_{i=1}^{N}[T_\theta(\mathbf{x}_i^n, \mathbf{y}_i^n)] - \beta \log \frac{1}{N}\sum_{i=1}^{N}[e^{T_\theta(\mathbf{x}_i^n, \bar{\mathbf{y}}_i^n)}] \\ & - \frac{(1-\beta)}{N}\sum_{i=1}^{N}[T_\theta(\mathbf{x}_i^n, \bar{\mathbf{z}}_i^n)] + (1-\beta)\log \frac{1}{N}\sum_{i=1}^{N}[e^{T_\theta(\mathbf{x}_i^n, \bar{\mathbf{z}}_i^n)}] \end{aligned} \tag{54}$$

where the coefficient β represents the trade-off between the communication rate and secrecy rate.

As mentioned above, to avoid approximating the channel probability distribution itself, the mutual information between the samples of the channel input and output and optimize the encoder weights will be estimated by maximizing the MI between them, see Figure 38.

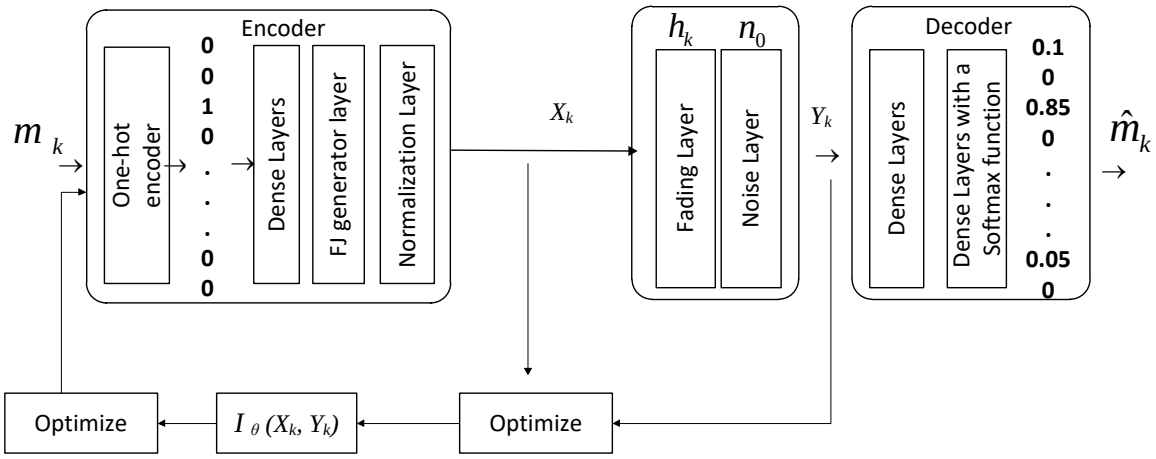


Figure 38: Optimization at the encoder

Implementation

In this part, we will examine the secrecy capacity in the practical scenarios of ImCSI as follows:

- Unknown static CSI, $\mathbf{H}_k$, with the block channel fading assumption.
- Channel information error with $\mathbf{\Delta H}_k$ is modeled through independent identical distributed complex Gaussian distribution with zero mean and scaled identity covariance matrix

In this scenario, the channel coefficients of both Bob's and Eve's channels are unknown and remain constant for a coherence interval of transmit symbol periods. The conversion in (39) is used for the case of the received signals $\mathbf{z}_k$ and the channel matrix $\mathbf{G}_k$ at Eve. Similar to conventional FJ method, the transmitted signals $\hat{\mathbf{x}}_k = \hat{\mathbf{s}}_k + \hat{\mathbf{w}}_k$, where $\hat{\mathbf{w}}_k$, and $\hat{\mathbf{s}}_k$ are FJ and information-bearing signals individually. From (40), the received signals received at Bob and Eve respectively, are

$$\begin{aligned} \hat{\mathbf{y}}_k &= \hat{\mathbf{H}}_k^{\dagger}\mathbf{s}_k + \hat{\mathbf{H}}_k^{\dagger}\hat{\mathbf{w}}_k + \hat{\mathbf{n}}_b, \\ \hat{\mathbf{z}}_k &= \hat{\mathbf{G}}_k^{\dagger}\mathbf{s}_k + \hat{\mathbf{G}}_k^{\dagger}\hat{\mathbf{w}}_k + \hat{\mathbf{n}}_e. \end{aligned} \tag{55}$$

To satisfy the following power constraint of the transmitted signal $\hat{\mathbf{x}}_k$:

$$E[\hat{\mathbf{x}}_k^{\dagger}\hat{\mathbf{x}}_k] = E[\hat{\mathbf{s}}_k^{\dagger}\hat{\mathbf{s}}_k] + E[\hat{\mathbf{w}}_k^{\dagger}\hat{\mathbf{w}}_k] \leq P,$$

the normalization layer is designed right before the channel/fading layer. The main layers that represent the proposed security communication model as

- *Input Layer/Encoder*: encode a message m_k to $\tilde{s}_k$
- *FJ layer*: generate $\mathbf{w}_k$ orthogonal to $\tilde{s}_k$
- *Power constraints/norm*: Normalization of average power
- *Channel Layers*: generate a random complex channel $\mathbf{H}_k$
- *Lamda Layer*: Matrix multiplication $\mathbf{x}_k$ with $\mathbf{H}_k$
- *Hidden Layers*: Simulate the Alice-Bob channel that estimate the mapping function: $Q(\mathbf{x}_k, \tilde{\mathbf{H}}_k)$
- *Power constraints/norm*: Normalization of average power
- *Loss Function*: the Equation (52)
- *Optimizer*: Adam optimizer

Next, we consider that the estimated channel state information at the transmitter side can not be perfect in general because channel feedback is not error-free. For purposes of our analysis, we denote $\mathbf{H}_k$ to be the ImCSI at Tx and the mathematical expression is given by

$$\tilde{\mathbf{H}}_k = \mathbf{H}_k + \Delta\mathbf{H}_k, \tag{56}$$

where $\Delta\mathbf{H}_k$ is an i.i.d. complex Gaussian distribution with zero mean and scaled identity covariance matrix given as $\Delta\mathbf{H}_k \sim \mathcal{CN}(0, \rho_e^2 \mathrm{I}_{N_R})$, and $\rho_e^2 = \frac{N_T}{N_P E_P}$ with N_p and E_p representing the number and the power of training symbols respectively.

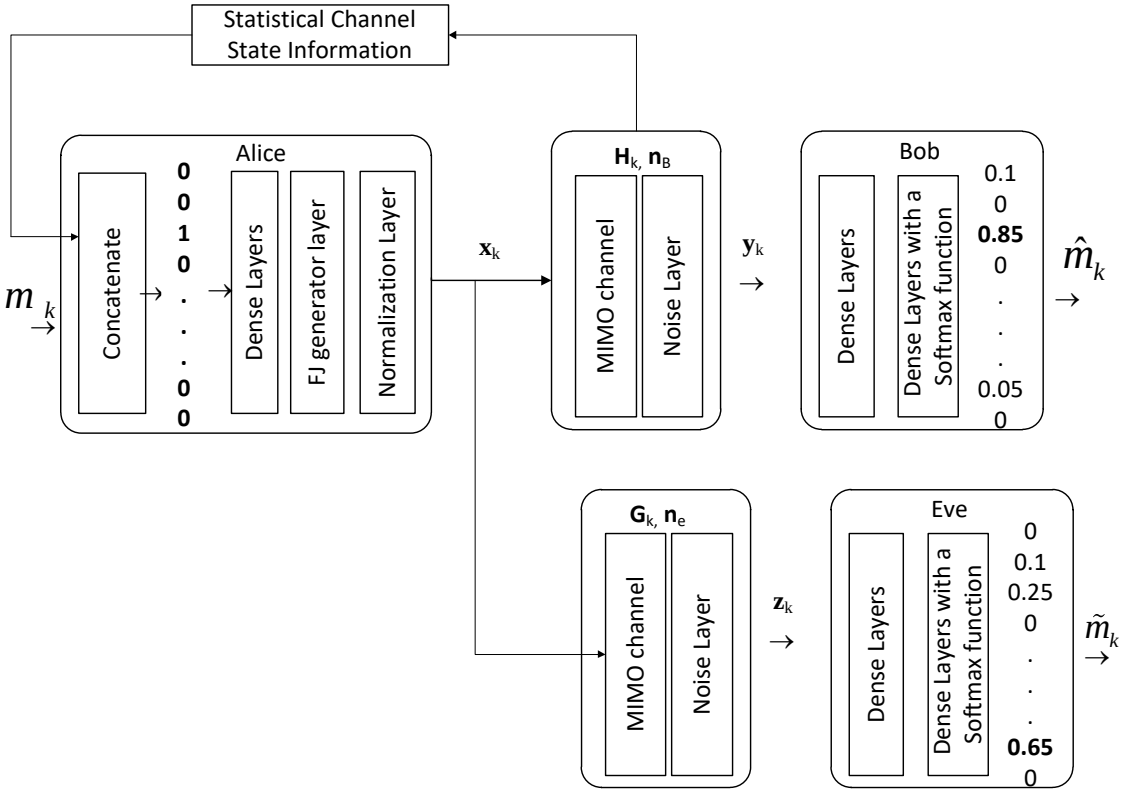


Figure 39: Learning based FJ with statistical CSI

Unlike the optimization in Equation (38) based on the SVD of the channel, we assume that only statistical information of the channel and CSI errors is available at Alice as seen in Figure 39. Compared to the random static channel and perfect CSI cases, two factors contribute to the decrease of secrecy capacity as the information-bearing signal will leak to Eve's channel, and the friendly jamming signals will interfere Bob's.

The impacts of the ImCSI on secrecy capacities in the traditional method can be seen in the equation (36) where the error is taken into account as

$$\overline{C}_s \doteq \max_{\mathbf{E}[\mathbf{x}_k \mathbf{x}_k^\dagger] \leq P} \log\left(det(\mathbf{I} + (\mathbf{H}_k + \Delta\mathbf{H}_k)\mathbf{Q}_s(\mathbf{H}_k + \Delta\mathbf{H}_k)^\dagger\right) - \log\left(\frac{det(\mathbf{K} + \mathbf{G}_k\mathbf{Q}_s\mathbf{G}_k^\dagger)}{det(\mathbf{K})}\right). \quad (57)$$

The optimization in (57) can be resolved by partitioning the SVD of $\mathbf{H}_k$ and second-order perturbation analysis [130]. However, the solution requires exponential complexity when the number of antennas increases. Thus, we leverage the non-convex optimization capability provided by NNs then directly solve the problem with sufficient training data. The architectures of our NN will be as

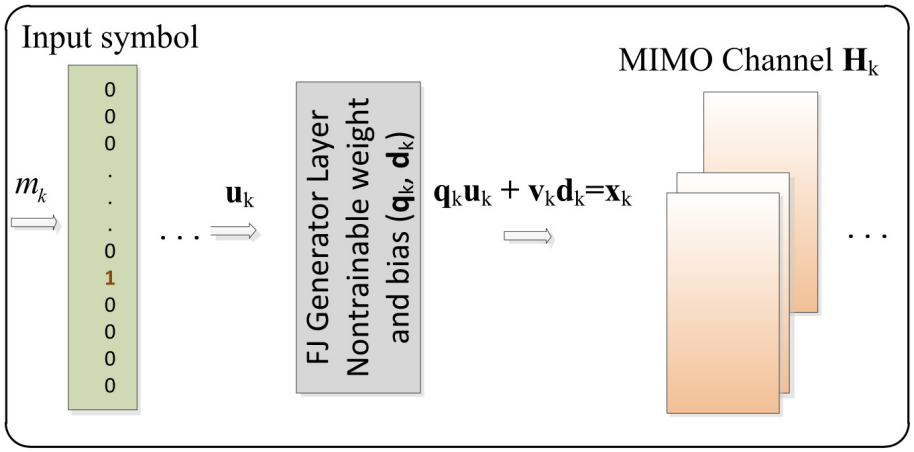


Figure 40: FJ generator scheme

- *Input Layer*: Concatenation of $\mathbf{x}_k$ and $\tilde{\mathbf{H}}_k$ which have been converted to real domain from complex domain
- *Power constraints/norm*: Normalization of average power
- *Channel Layers*: generate a imperfect complex channel $\tilde{\mathbf{H}}_k$
- *Lamda Layer*: Matrix multiplication $\mathbf{x}_k$ with $\tilde{\mathbf{H}}_k$
- *Hidden Layers*: Simulate the Alice-Bob channel for estimating the mapping function: $Q(\mathbf{x}_k, \tilde{\mathbf{H}}_k)$
- *Output Layer*: $\hat{\mathbf{x}}_k$, and the activation function is soft-max for a reconstruction problem
- *Optimizer*: Adam optimizer
- *Loss Function*: the Equation (52)

By considering the CSI error as an input for training, our network will be trained to maximize secrecy capacity and reconstruct the signals as well.

FJ is automatically generated by the FJ generator layer, which is shown in Figure 40. As proposed above, the FJ signal $\hat{\mathbf{w}}_k$ is orthogonal with the information bearing signals $\hat{\mathbf{s}}_k$ that $\hat{\mathbf{s}}_k = \hat{\mathbf{q}}_k\hat{\mathbf{u}}_k$, and $\hat{\mathbf{w}}_k = \hat{\mathbf{v}}_k\hat{\mathbf{d}}_k$, where $\hat{\mathbf{u}}_k$ denotes the information signals. The parameter $\hat{\mathbf{q}}_k$ and $\hat{\mathbf{v}}_k$ are the non-trainable weight and bias in the layer and orthogonal with each other. The elements of $\hat{\mathbf{d}}_k$ is chosen as i.i.d. Gaussian random variable.

Experiments and Discussion

We use the state-of-the-art deep learning library Tensor Flow with Adam optimizer as tools for the training process. To yield the BLER and average secrecy capacity C_s, we

use Monte-Carlo simulation with the flat Rayleigh fading channel [152, 153] by including a fading layer right before the normalization layer. The components of $\mathbf{H}_k$ and $\mathbf{G}_k$ are $h_{i,j}$ and $g_{i,j}$, respectively, and are assumed to be i.i.d. Gaussian with $E(|h|^2_{i,j}) = E(|g|^2_{i,j}) = 1$. Further, we assume that the power constraint P has been normalized by the power of adaptive white noise variables n_b and n_e. For this simulation, SNR of 7 dB is set up during the training phase. For the analysis purpose, Eve is assumed to have the same neural architecture decoder as Bob, in the Figure 36. The input and output layer has 16 neurons, which represent a symbol of 4 bits. The channel layer includes N_T neurons width representing the number of transmit antenna. The AE at Alice-Bob channel inducing FJ and NN at Eve are trained at the same time as a one-input-two-output NN network with the security loss function (52).

MIMO-AEFJ with Perfect CSI

We first implement the AE-based FJ model in case of perfect CSI. The aim of this task is to make a comparison of our method, based on neural network (NN) training, with the baseline work, based on exhaustive search in [127]. Regarding AE architecture, the channel matrix is used as an extra input parameter at the encoder and decoder in the AEFJ model.

Figure 41 compares the secrecy rates in AEFJ and that in the work [127], which uses exhaustive search, presented in equation (38). The competitive results in both cases of the number of transmit antennas increase from $N_T = 10$ to $N_T = 20$. Though the curve of the former is approximately that of the latter, it can be explained that the exhaustive search is considered as the most resource-consuming solution.

MIMO-AEFJ with ImCSI

In this experiment, our target is to make a comparison of the average secrecy rate achieved in our method with the one proposed in [130] when full CSI is not available at Alice. In the previous work, authors use second-order statistics analysis (So) of SVD of the imperfect channel to examine secrecy capacity. To do it, the channel error $\Delta\mathbf{H}_k \sim \mathcal{CN}(0; \sigma I_{N_R})$ is included in the two model above, where $\sigma = 0.1$ represents the perturbation level.

Firstly, we consider the case of unknown static CSI with the stochastic channel model as seen in Figure 39. Secondly, the channel error is applied to examine the secrecy capacity in this ImCSI scenario. Figure 42 compares the resulting average secrecy rate with ImCSI in different SNR between AEFJ based and So based method [130]. It can be seen that the average secrecy rate of the former is higher than that of the latter. We remark that the secrecy rate increase linearly at low SNR but tends to be stable at high normalized transmit power, from 10 to 25 dB. Moreover, the higher number of transmit

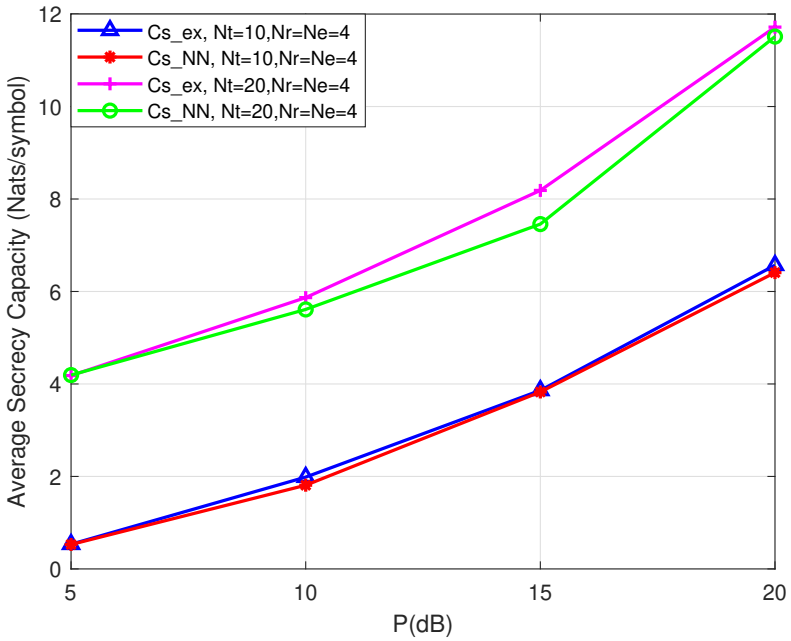


Figure 41: Average secrecy capacity with full CSI

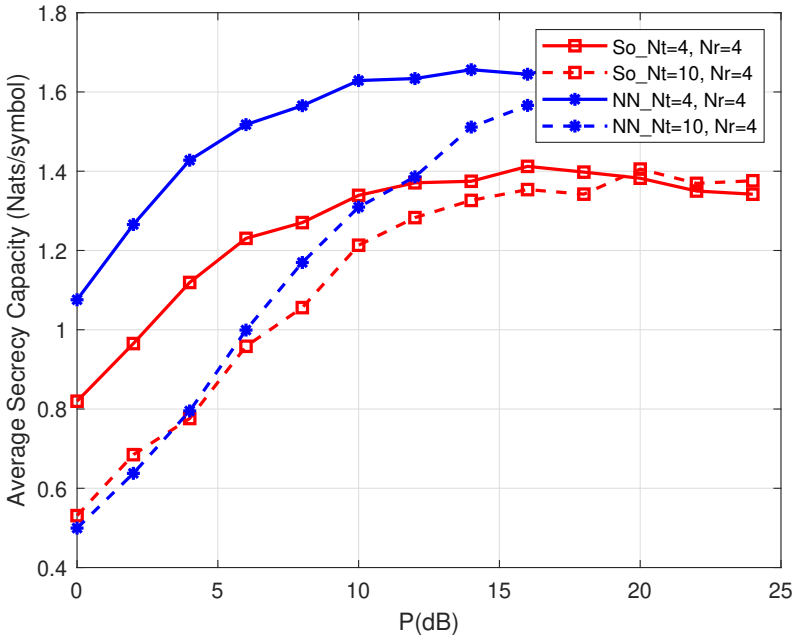


Figure 42: Secrecy rate versus P.

antennas, the higher the secrecy rate at low SNR. The secrecy rate tends to be stable at the SNR of about 15 to 25dB when the number of antennas increases.

Figure 43 compares the BLER for the proposed method between Bob and Eve. It

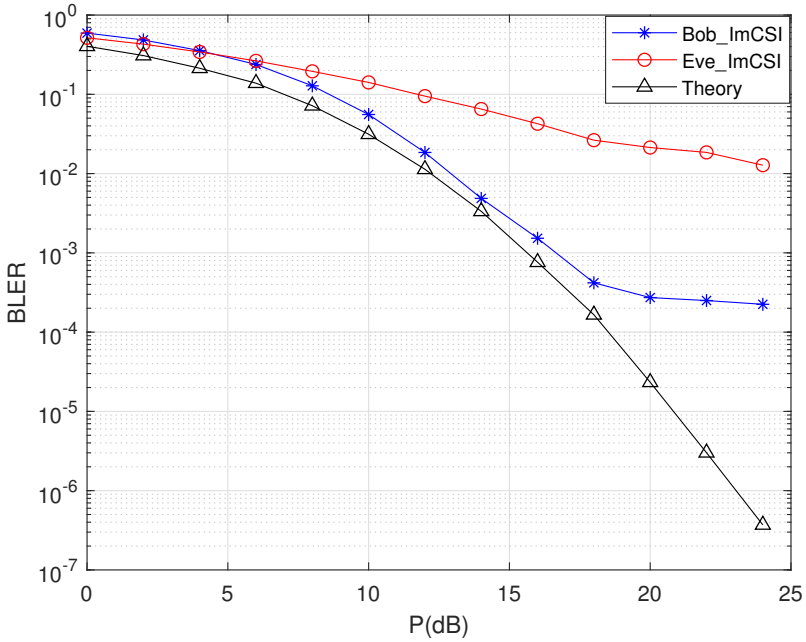


Figure 43: BLER with ImCSI

is worth mentioning that the BLER of Eve's decoder with the same NNs, in this case, is much higher than that of Bob's. Although the BLER of the proposed method is higher than that of the ML method, BLER theory, using our method for MIMO detection, has much lower complexity than ML algorithm.

MINE-FJ for Unknown CSI

Regarding MINE based security approach, the structure of our model is unchanged with the MINE security function (54). Figure 44 demonstrates the secrecy rate with two different values of β, given the number of transmit antennas $N_T = 3$, each is with 400 iterations and the batch size of $20,000$. We observe that the higher the value of β, the higher the secrecy rate. There is a trade-off between the communication rate and the secrecy rate due to the influence of the FJ signal. Figure 45 shows BLER at Bob and Eve, before and after a secure communication by AEFJ, for some SNR values. We observed a significant increase in BLER of Eve by using AEFJ compared to the one without AEFJ. Meanwhile, Bob's BLER change before and after applying AEFJ is negligible. This means the proposed method shows high performance against the physical-layer security thread by eavesdropping.

The resulting secrecy rate curves in different SNR levels can be seen in Figure 47 by using the security model in Figure 37. It can be seen that the gaps of secrecy capacity in the low levels are bigger than those in high levels of SNR. The relationship between

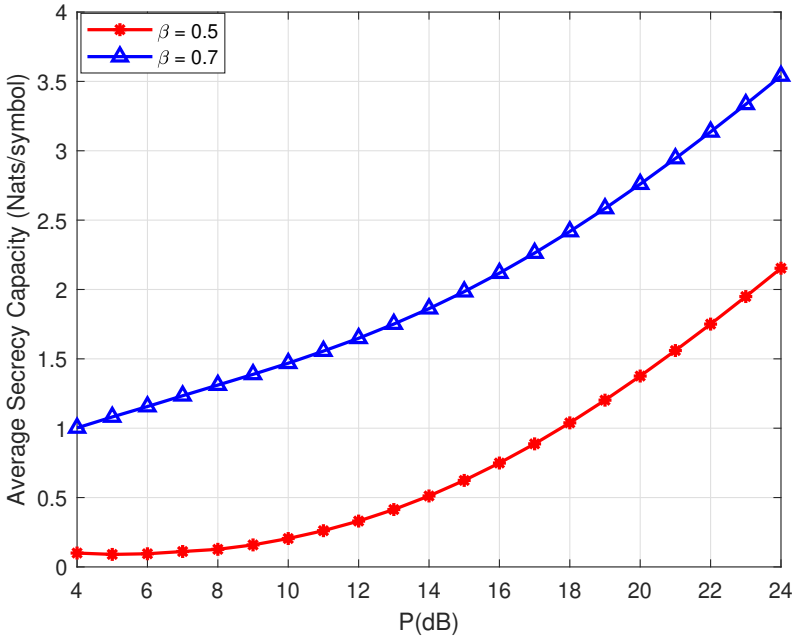


Figure 44: Average secrecy capacity with different values of β.

secrecy rate and the number of transmit antennas can be seen in Figure 46. The figures show that the values of secrecy rate increase linearly with the number of antennas N_T but tend to be stable although N_T raises from 10 to 25. Figure 47 demonstrates the variant of average secrecy rate with a total transmit power and number of iteration to be convergent for the MINE-based FJ model. We observe that the average secrecy capacity regularly increases with the low normalized transmit power, from 2 to 10 dB. The secrecy capacity at higher power levels converges to a constant.As expected, MINE-based FJ gains better secrecy performance than an end-to-end learning setup in AEFJ because the encoder basically uses the expert information about which performance function (i.e., the mutual information) it needs to optimize the encoding in order to perform well. This is in contrast to the end-to-end learning approach, where the neural network system needs to learn this information on its own. Furthermore, our method can be implemented without changes at the receiver side and is therefore suitable for fast deployment. This means that the secrecy performance will not increase along with the power allocation. Regarding the relationship between the average secrecy rate and BLER of the receiver at Bob, Figure 49 shows a decrease in the average secrecy rate when BLER increases. The decoding errors at Bob decrease along with the secrecy rate and vice versa.

Complexity

In this subsection, we will compare the computational complexity in terms of the number of floating-point operations (FLOPs) between the proposed method and conven-

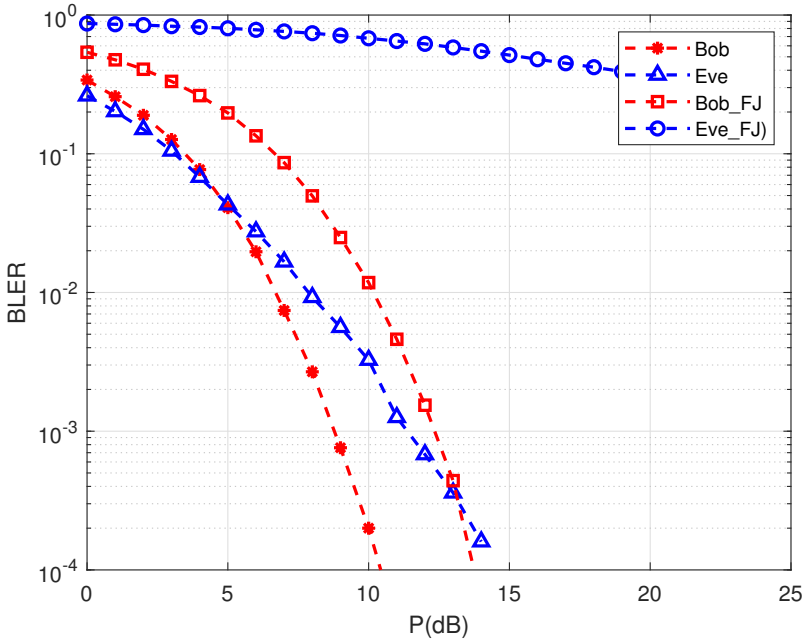


Figure 45: The block error rate versus P in AE based FJ.

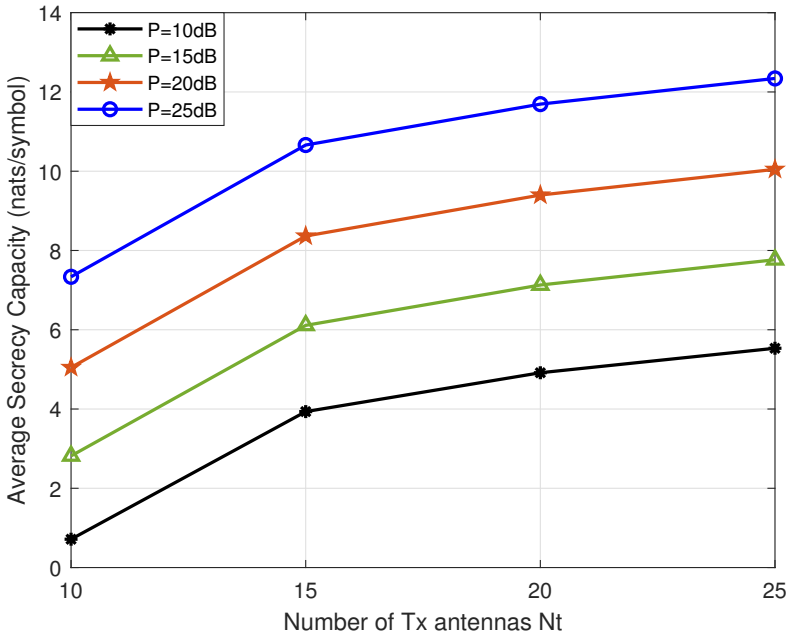


Figure 46: Secrecy rate versus number of transmit antenna.

tional one. We only count the complexity in the deployment stage since the training stage can be seen as an offline step. The number of FLOPs of a dense layer is referred to the work in [154] which is equal to $(2N_{Input} - 1)N_{Output}$, where N_{Input} and N_{Output} denote the

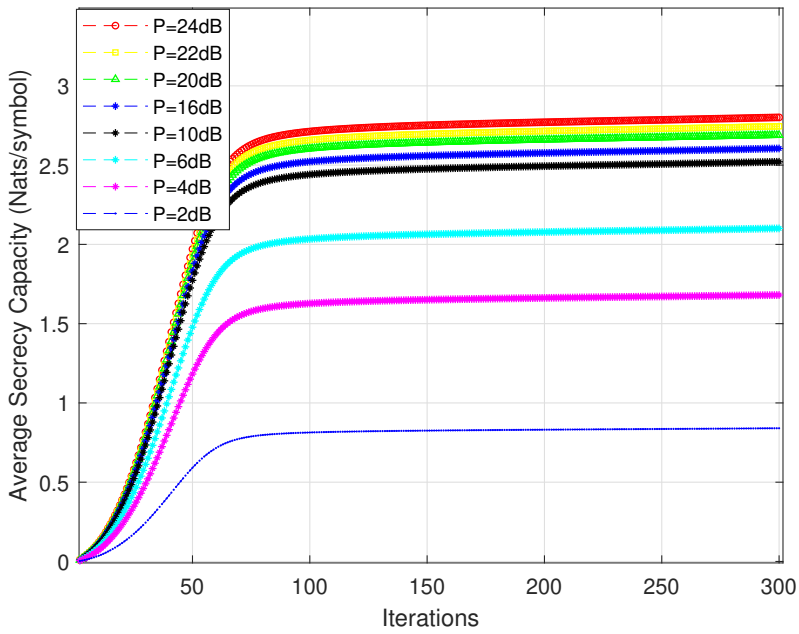


Figure 47: Secrecy rate versus the number of iteration with $N_T = 4, N_R = 2, N_E = 2$.

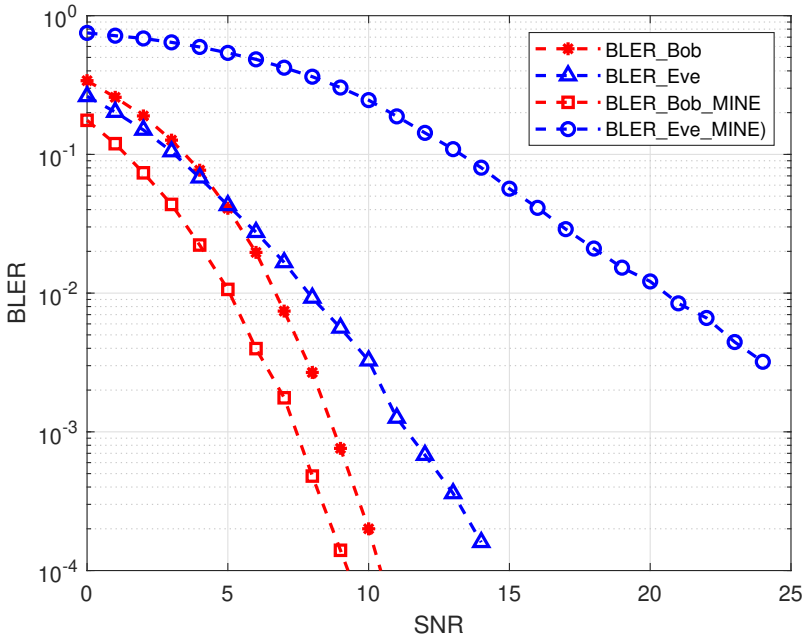


Figure 48: BLER at Bob and Eve using MINE security.

input and output dimensions, respectively. For example, when $N_T = 20$ and the transmit symbol is encoded into a one-hot vector and considering the architecture of our model in Table 25, the FLOPs is 6.144.

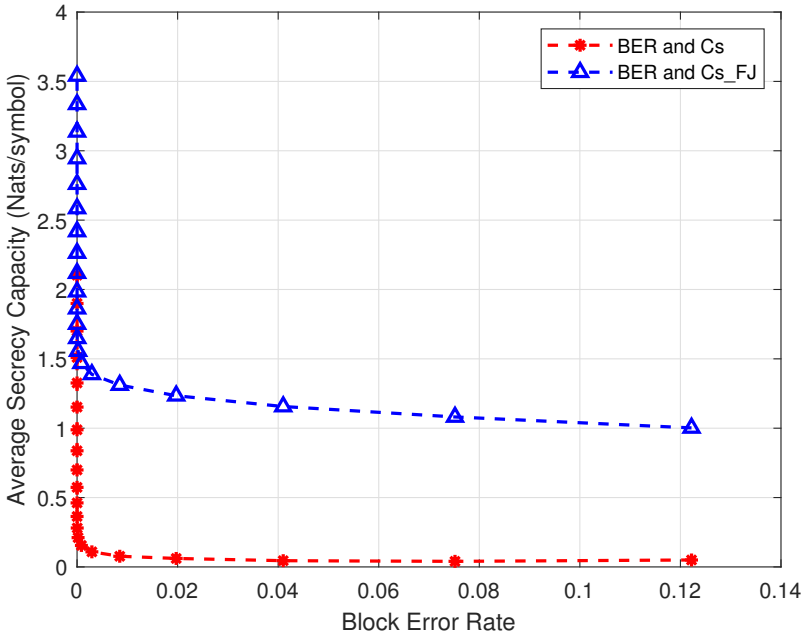


Figure 49: Secrecy rate and BLER.

Table 25: Main Parameter details of AEFJ Model.

Name of Layers	Input, Output Dimension	FLOPs
Input Layer	16,64	2048
Dense Layer 2	64,8	1024
Dense Layer 3	8,64	1024
Output layer	64,16	2048

For conventional methods based on exhausting search and So mentioned above, the asymptotic computational complexity is in the order of $O(N_T)^3$ as they include the singular value decomposition and matrix inversion even taking the coefficient in the complexity order to be 1.

Discussions

The idea of learning a well-known MIMO scheme for securing MIMO encoding across a range of different assumptions on CSI information, time-slots, antenna counts, and information densities is really quite appealing. In our results, we have shown that the proposed method is competitive with the conventional method in the case of ImCSI, with perfect CSI. Such a system can be readily partitioned into a real-world distributed communications system to guarantee secrecy efficiently.

This work considers the problems of secret MIMO communications in terms of ImCSI. It can be seen that the conventional works which based on partial CSI and statistical analyst, then based on an iterative method to optimize power allocation for information and FJ signals. In contrast, our proposed method is based on the learning process to build a pre-train model. Basically, that learning process is the NNs based optimization process with the loss function is designed for maximizing secrecy capacity. That trained model shows robustness in channel errors better than the previous work.

The simulation results presented above in various ImCSI scenarios in case of channel estimation error and without channel estimation. In the former situation, our proposed method is compared with the work in [127]. Other key considerations and challenges related to learning accuracy need to be examined. For the training process, we still base on a known stochastic channel model to learn the features of wireless channels. The more channel models are used for offline training, the higher performance can be achieved in the testing/running step. In other words, if the method can be used in real-world implementation, the data of various channels will be necessary to improve system performance. Moreover, the FJ approach has to satisfy the condition that the number of the antenna at the transmitter must be larger than that of eavesdropper (or concluded eavesdroppers). From this work, we can see the relationship between the learning method and system identification. MINE exemplifies the situation that from samples/observations without channel assumption, we can construct a model for an unknown system.

Conclusion

In this work, we have presented a new deep learning-based friendly jamming approach to deal with the eavesdropping issues in wireless communication. In the AEFJ scheme, where learning end-to-end is at both transmitter and receiver, we have shown that communication secrecy and reliability can be achieved simultaneously compared to the conventional model. In addition, we leverage the mutual information neural estimator to optimize the security scheme. This modification shows comparable security performance as compared to the AEFJ with the cross-entropy security loss function. Further, using the mutual information neural estimator, we can optimize the secrecy rate independently at the transmitter and the receiver, which is a shortage of the conventional AEFJ model. This method is promising for applications that require fast deployment and lightweight security, such as IoT networks.

(2) Leveraged Resources and Participants

This project is composed of researchers from Vietnam National University Hanoi (VNU), Nanyang Technological University (NTU) and University of Technology Sydney (UTS), which are prestigious universities in Vietnam, Singapore and Australia respectively.

The results of the project include a testbed, a website for Risk assessment in IIoT, 4 conference papers, and 5 journal articles.

NICT supported the main funding, for all equipment, organization of meetings and workshops, research visit to NICT.

VNU supported almost main activities of the project such as offices for meetings and testbed, salary for Do Hai Son (research assistant, master student), manpower for project management (Prof. Nguyen Viet Ha, Prof. Nguyen Linh Trung), manpower project implementation (Prof. Nguyen Viet Ha, Prof. Nguyen Linh Trung and Dr. Tran Thi Thuy Quynh, Dr. Ta Duc Tuyen), manpower for organization of meetings and workshops (staff of the Department of Science, Technology, and International Relations).

NTU supported research advice in security in general and in blockchain technology in particular (Prof. Dusit Niyato).

UTS supported scholarships for PhD students (Bui Minh Tuan, Tran Viet Khoa), manpower for project management (Prof. Eryk Dutkiewicz), manpower for project implementation (Dr. Diep Nguyen and Dr. Dinh Thai Hoang, Prof. Eryk Dutkiewicz).

Research collaboration was held on regular basis with meetings between UTS and VNU, especially for student supervision (Bui Minh Tuan, Tran Viet Khoa, Do Hai Son), and sometimes with workshops for all sides.

Contributors to the project are presented in Table 26.

(3) Findings and Outcomes

The findings and outcomes of this project can be summarized as follows:

Research problem 1

- We provided a brief review of methodologies and existing standards used for cyber risk assessment, primarily focusing on OT and recommendations for improving cyber risk assessment for information technology (IT) and operation technology (OT) systems in Industry 4.0 in Vietnam.
- We proposed a possible framework for Industrial IoT (IIoT) risk assessment in Vietnam. The proposed framework considers IT, OT, and IIoT system. We built an experiment that simulates an IIoT network and compare with several

Table 26: Project Member Contributions.

Content	Implementer	Advisor
Task 1	Bui Minh Tuan	Nguyen Linh Trung, Tran Thi Thuy Quynh, Nguyen Viet Ha
Task 2	Tran Viet Khoa	Nguyen Linh Trung, Tran Thi Thuy Quynh, Nguyen Viet Ha
Task 3	Do Hai Son, Tran Viet Khoa, Bui Minh Tuan	Nguyen Linh Trung, Tran Thi Thuy Quynh, Nguyen Viet Ha
Task 4	Bui Minh Tuan	Diep Nguyen, Nguyen Linh Trung, Dinh Thai Hoang, Nguyen Viet Ha, Eryk Dutkiewicz, Ta Duc Tuyen
Task 5	Do Hai Son	Nguyen Linh Trung, Dinh Thai Hoang, Tran Thi Thuy Quynh, Nguyen Viet Ha, Diep Nguyen, Eryk Dutkiewicz, Dusit Niyato
Task 6	Tran Viet Khoa	Dinh Thai Hoang, Nguyen Linh Trung, Diep Nguyen, Nguyen Viet Ha, Eryk Dutkiewicz
Task 7	Do Hai Son, Tran Viet Khoa	Nguyen Linh Trung, Dinh Thai Hoang, Tran Thi Thuy Quynh, Nguyen Viet Ha, Diep Nguyen, Eryk Dutkiewicz, Dusit Niyato

existing frameworks. The results show that our method gives the same severity level as OWASP.

Research problem 2

- We proposed a novel collaborative learning framework that can effectively detect cyberattacks in decentralized IoT systems, by combining the strengths of federated learning (FL) and transfer learning (TL).
- We proposed an effective transfer learning approach that can allow the deep learning model from the rich-data network to transfer useful knowledge to the low-data network even they have different features for cyberattack detection in IoT networks.
- We performed extensive experiments on recent real-world datasets including N-BaIoT, KDD, NSL-KDD, and UNSW to evaluate the performance of the proposed collaborative learning framework.

Research problem 3

- We developed a practical Ethereum-based smart grid with essential hardware in a home electrical system.

- We proposed a smart contract for authentication in a securely multi-devices system.
- We proposed a method to improve the efficiency of an Ethereum-based smart grid setup in practical work with the support of numerical experiments.

Research problem 4

- We leveraged the generalization features of neural networks to develop a MIMO friendly-jamming (FJ) scheme that is robust to imperfect channel state information (CSI) due to issues such as time varying channels or the limited number of pilots.
- We leveraged MINE-based FJ based on mutual information neural estimation (MINE) to demonstrate that it is possible to achieve a security performance comparable with the conventional FJ method without CSI.
- We also investigated the relationship between the MIMO secrecy optimization and detection tasks. In other words, maximizing secrecy rate and minimizing block/symbol error rate can be jointly optimized.

Research problem 5

- We set up experiments in our laboratory to build a private blockchain network (BNaT) with the aims of not only obtaining real blockchain datasets, but also testing our proposed learning model in a real-time manner.
- We built an effective tool named Blockchain Intrusion Detection (BC-ID) to collect data in the blockchain network.
- We proposed a collaborative decentralized learning model to not only improve the accuracy of identifying attacks, but also effectively deploy in decentralized blockchain networks.
- We performed both intensive simulations and real-time experiments to evaluate our proposed framework.

The findings and outcomes have been presented in 4 conference papers (Table 27) and 5 journal articles (Table 28).

(4) Broader Impact

One of the results for Problem 1, in the form of a website for cyber risk assessment may be used by Vietnamese organizations.

The theoretical results for Problems 2, 3 and 5 can be extended/applied to broader scenarios, especially for dealing with more types of attacks.

Table 27: Conference papers.

No	Paper title	Author names	Affiliation	Conference name	date	venue
1	Network Coding with Multimedia Transmission: A Software-Defined-Radio based Implementation [Problem 4]	TTT Quynh, TV Khoa, LV Nguyen, NL Trung.	VNU-UET	International Conference on Recent Advances in Signal Processing, Telecommunications and Computing	March, 2019	Hanoi, Vietnam
2	Collaborative Learning Model for Cyberattack Detection Systems in IoT Industry 4.0 [Problem 2]	TV Khoa, YM Saputra, DT Hoang, NL Trung, DN Nguyen, NV Ha, E Dutkiewicz	VNU-UET, UTS	IEEE Wireless Communications and Networking Conference	May, 2020	Seoul, South Korea
3	Autoencoder based Friendly Jamming [Problem 4]	BM Tuan, TD Tuyen, NL Trung, NV Ha	VNU-UET	IEEE Wireless Communications and Networking Conference	May, 2020	Seoul, South Korea
4	An effective framework of private ethereum blockchain networks for smart grid [Problem 3]	DH Son, TTT Quynh, TV Khoa, DT Hoang, NL Trung, NV Ha, D Niyato, DN Nguyen, E Dutkiewicz	VNU-UET, UTS, NTU	2021 International Conference on Advanced Technologies for Communications (ATC) (Best student paper award)	Oct, 2021	Ho Chi Minh, Vietnam

Table 28: Journal papers.

No	Paper title	Author	Affiliation	Journal	Publisher	Volume, Number, Pages
1	A Survey on Consensus Mechanisms and Mining Strategy Management in Blockchain Networks [Problem 3]	W Wang, DT Hoang, P Hu, Z Xiong, D Niyato, P Wang, Y Wen, D Kim	NTU, UTS	IEEE Access	IEEE	vol. 7, pp. 22328-22370, 2019
2	Deep transfer learning: A novel collaborative learning model for cyberattack detection systems in IoT networks [Problem 2]	TV Khoa, DT Hoang, NL Trung, CT Nguyen, T DN Nguyen, NV Ha, E Dutkiewicz	VNU-UET, UTS	IEEE Internet of Things Journal	IEEE	Accepted, 2022
3	Collaborative Learning for Cyberattack Detection in Blockchain Networks [Problem 5]	TV Khoa, DH Son, DT Hoang, NL Trung, TTT Quynh, DN Nguyer NV Ha, E Dutkiewicz	VNU-UET, UTS	IEEE Transactions on Systems, Man, and Cybernetics: Systems	IEEE	Submitted, 2022
4	A New Framework for Cyber Risk Assessment for Industry 4.0 and Recommendations for Vietnam [Problem 1]	BM Tuan, TV Khoa, DH Son, NL Trung, TTT Quynh, NN Hoa, ND Tho, NV Ha	VNU-UET, UTS	REV Journal on Electronics and Communications	REV	Submitted, 2022
5	Learning-based Friendly Jamming with Imperfect CSI for Security in MIMO Wiretap Channel [Problem 4]	BM Tuan, NL Trung, DN Nguyen, M Krunz, NV Ha, DT Hoang, E Dutkiewicz	VNU-UET, UTS	IEEE Transactions on Communications	IEEE	Submitted, 2022

The proposed BNaT dataset for Problem 5 can be used and extended for further study and development of blockchain security.

The security solutions studied in this project is rather general and can be applied to different applications (smart factory, smart agriculture, ...)

(5) Future Developments

The built website can be used by academic institutes and industrial companies to evaluate the risks of their systems. Based on that evaluations, we can build a dataset or a map of the risk level and potential risks of them for future research.

In addition, the built BNaT dataset can be used for various studies on cyberattack detection in blockchain networks in the future. Besides, we continue to research to find out malicious actions in the blockchain application layer, build a dataset on that layer, and find the method to solve that problem. In detail, the smart contract layer is known as the application layer in the blockchain network, which has many vulnerabilities [155]. Hackers exploited these vulnerabilities to steal digital assets from the deployed smart contracts. In the future, we will consider setting up a medium-scale Ethereum blockchain network to capture the first cyberattack dataset in the application layer from the laboratory environment. The following are types of attacks that we plan to execute on our network:

1. Integer Overflow and Underflow: In the Ethereum blockchain network, the largest number is 2^{256}. An integer overflow will happen if any operator tries to handle a number bigger than 2^{256}. If the smart contract uses unsafe math operators, hackers could use this weakness to generate a lot of invalid tokens. The same thing when a zero value is subtracted by another value, the result will be 2^{256}. In real-world, the attackers had successfully got 10^{58} BEC tokens [156] exploiting a vulnerability caused by integer overflowing. In 2018, 866 ETH vanished from the PoWH [157] contract caused by integer underflow weakness.

2. Reentrancy: A reentrancy attack occurs when a function makes an external call to another untrusted contract. A famous real-world Reentrancy attack is the DAO attack [158] which caused a loss of $60 million.

3. Delegatecall to Untrusted Call: To reduce the development cost, many smart contracts inherit functions from other deployed smart contracts, which are known as called “Delegatecall”. However, if the main smart contract is secured but the delegated smart contract has unsecured functions. Hackers can call the weakness function of delegated smart contracts from the main smart contract, and this weakness will also affect the main smart contract. In fact, Parity Multisig Wallet (Second Hack) is known as the famous attack caused by this vulnerability.

4. DoS With Block Gas Limit: The Ethereum blockchain limits the maximum gas for a block by a parameter in the previous block header. For example, at the time of writing this report, the block gas limit is around 30 million gas. That means if a function in the smart contract requires more than the block gas limit, it would be reverted by EVM and never be executed. Hence, if hackers found this

vulnerability in a smart contract, they could send many transactions to disable this smart contract. In 2016, the GovernMental Jackpot [159] smart contract was disabled caused by the paying out function exceeded the block gas limit at around 5 million gas. Hence, 1100 ETH never be paid to the winner.

5. Function Default Visibility: Functions in Solidity have visibility specifiers which dictate how functions are allowed to be called. Functions default to public allowing users to call them externally. Incorrect use of visibility specifiers can lead to some devastating vulnerabilities in smart contracts. Parity Multisig Wallet (First Hack) is reported as the first attack.

Finally, we can use intelligent reflecting surfaces (IRS) to perform a number of passive beamforming to enhance security performance of based on friendly jamming.

3) Social Contribution

This project has contributed to the society in a number of ways:

- The website developed in Problem 1, in the form of a website for cyber risk assessment, may be used by Vietnamese organizations [https://www.youtube.com/watch?v=ckgckAUGO7Q].
- The results for Problem 3 were recognized in the scientific community in the form of a "Best student paper award" at the 2021 International Conference on Advanced Technologies for Communications.
- The results for Problem 5 were widely disseminated via our participation in the 2022 Vietnam AI Awards, with a security solution called "Collaborative learning for cyberattack detection in blockchain network" [https://www.youtube.com/watch?v=uW6Khv_aPLg].

References

[1] "The asean ict masterplan 2020." [Online]. Available: https://asean.org/wp-content/uploads/images/2015/November/ICT/15b%20--%20AIM%202020_Publication_Final.pdf

[2] B. C. Ervural and B. Ervural, "Overview of cyber security in the industry 4.0 era," in *Industry 4.0: managing the digital transformation*. Springer, 2018, pp. 267–284.

[3] "Gartner says by 2020, more than half of major new business processes and systems will incorporate some element of the internet of things." Gartner, 2016, accessed: Feb. 14, 2022. [Online]. Available: https://www.gartner.com

[4] "Information technology —- security techniques -— information security risk management," International Organization for Standardization, Standard ISO/IEC 27005:2008, 2008.

[5] "Operational technology (ot)," Gartner, accessed: Feb. 14, 2022. [Online]. Available: https://www.gartner.com/en/information-technology/glossary/operational-technology-ot

[6] M. Le, H. D. Huu, T. N. Ngoc, and et al., "An assessment model for cyber security of vietnamese organization," *VNU Journal of Science: Policy and Management Studies*, vol. 33, no. 2, pp. 97–103, 2017.

[7] L. V. Ha, P. V. On, and N. N. Hoa, "Information security risk management by a holistic approach: a case study for vietnamese e-government," *IJCSNS International Journal of Computer Science and Network Security*, vol. 20, no. 6, pp. 72–88, 2020.

[8] Y. Zheng and S. Zheng, "Cyber security risk assessment for industrial automation platform," in *2015 International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, Adelaide, SA, Australia, 2015, pp. 341–344.

[9] X. Lyu, Y. Ding, and S.-H. Yang, "Safety and security risk assessment in cyber-physical systems," *IET Cyber-Physical Systems: Theory & Applications*, vol. 4, no. 3, pp. 221–232, 2019.

[10] I. Lee, "Internet of things (iot) cybersecurity: Literature review and iot cyber risk management," *Future Internet*, vol. 12, no. 9, p. 157, 2020.

[11] Y. Cherdantseva, P. Burnap, A. Blyth, P. Eden, K. Jones, H. Soulsby, and K. Stoddart, "A review of cyber security risk assessment methods for scada systems," *Computers & Security*, vol. 56, pp. 1–27, 2016.

[12] G. Culot, F. Fattori, M. Podrecca, and M. Sartor, "Addressing industry 4.0 cybersecurity challenges," *IEEE Engineering Management Review*, vol. 47, no. 3, pp. 79–86, 2019.

[13] H. Well, "The 4th industrial revolution," *Report*, 2018, accessed: Feb. 14, 2022. [Online]. Available: https://kemptechnologies.com/

[14] J. Lee, B. Bagheri, and H.-A. Kao, "A cyber-physical systems architecture for industry 4.0-based manufacturing systems," *Manufacturing letters*, vol. 3, pp. 18–23, 2015.

[15] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (iot): A vision, architectural elements, and future directions," *Future generation computer systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[16] A. Ustundag and E. Cevikcan, *Industry 4.0: managing the digital transformation*. Springer, 2017.

[17] M. Ghobakhloo, "The future of manufacturing industry: a strategic roadmap toward industry 4.0," *Journal of Manufacturing Technology Management*, vol. 29, no. 6, pp. 910–936, 2018.

[18] "Unified Architecture," OPC Foundation, accessed: Feb. 14, 2022. [Online]. Available: opcfoundation.org/about/opc-technologies/opc-ua

[19] A. Mishra, N. Gupta, and B. B. Gupta, "Security threats and recent countermeasures in cloud computing," in *Modern principles, practices, and algorithms for cloud security*. IGI Global, 2020, pp. 145–161.

[20] J. Prinsloo and et al., "A review of industry 4.0 manufacturing process security risks," *Applied Sciences*, vol. 9, no. 23, p. 5105, 2019.

[21] A. Gilchrist, *Industry 4.0: the industrial internet of things*. Springer, 2016.

[22] "The industrial internet of things volume g1: Reference architecture," in *Industrial Internet Consortium*, 2019.

[23] "Itu internet of thing report, 2005," https://www.itu.int/net/wsis/tunis/newsroom/stats/The-Internet-of-Things-2005.pdf.

[24] R. Khan, S. U. Khan, R. Zaheer, and S. Khan, "Future internet: The internet of things architecture, possible applications and key challenges," in *2012 10th International Conference on Frontiers of Information Technology*, 2012, pp. 257–260.

[25] A. Shameli-Sendi, R. Aghababaei-Barzegar, and M. Cheriet, "Taxonomy of information security risk assessment (isra)," *Computers & Security*, vol. 57, pp. 14–30, 2016.

[26] P. P. Roy, "A high-level comparison between the nist cyber security framework and the iso 27001 information security standard," in *2020 National Conference on Emerging Trends on Sustainable Technology and Engineering Applications*, Durgapur, India, 2020, pp. 1–3.

[27] "Common Vulnerability Scoring System (CVSS)," Forum of Incident Response and Security Teams, accessed: Feb. 14, 2022. [Online]. Available: https://www.first.org/cvss/

[28] "CVE site," https://cve.mitre.org/.

[29] M. U. Aksu, M. H. Dilek, E. İ. Tatlı, K. Bicakci, H. I. Dirik, M. U. Demirezen, and T. Aykır, "A quantitative cvss-based cyber security risk assessment methodology for it systems," in *2017 International Carnahan Conference on Security Technology (ICCST)*. IEEE, 2017, pp. 1–8.

[30] "Architecture alignment and interoperability," in *Industrial Internet Consortium*, 2019.

[31] B. Mburano and W. Si, "Evaluation of web vulnerability scanners based on owasp benchmark," in *2018 26th International Conference on Systems Engineering (ICSEng)*. IEEE, 2018, pp. 1–6.

[32] Y. Keshet, "Half of the malware detected in 2019 was classified as zero-day threats, making it the most common malware to date," Mar. 2020. [Online]. Available: https://www.cynet.com/blog/half-of-the-malware-detected-in-2019-was-classified-as-zero-day-threats-making-it-the-most-common-malware-to-date/

[33] S. Morgan, "Global ransomware damage costs predicted to hit $11.5 billion by 2019," Mar. 2021. [Online]. Available: https://cybersecurityventures.com/ransomware-damage-report-2017-part-2/

[34] T. V. Khoa, D. H. Son, D. T. Hoang, N. L. Trung, T. T. T. Quynh, D. N. Nguyen, N. V. Ha, and E. Dutkiewicz, "Collaborative learning for cyberattack detection in blockchain networks," *arXiv preprint arXiv:2203.11076*, Mar. 2022.

[35] W. Y. B. Lim, N. C. Luong, D. T. Hoang, Y. Jiao, Y.-C. Liang, Q. Yang, D. Niyato, and C. Miao, "Federated learning in mobile edge networks: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 2031–2063, Apr. 2020.

[36] M. Aledhari, R. Razzak, R. M. Parizi, and F. Saeed, "Federated learning: A survey on enabling technologies, protocols, and applications," *IEEE Access*, vol. 8, pp. 140 699–140 725, July 2020.

[37] C. T. Nguyen, N. Van Huynh, N. H. Chu, Y. M. Saputra, D. T. Hoang, D. N. Nguyen, Q.-V. Pham, D. Niyato, E. Dutkiewicz, and W.-J. Hwang, "Transfer learning for future wireless networks: A comprehensive survey," *arXiv preprint arXiv:2102.07572*, Feb. 2021.

[38] S. Niu, Y. Liu, J. Wang, and H. Song, "A decade survey of transfer learning (2010–2020)," *IEEE Transactions on Artificial Intelligence*, vol. 1, no. 2, pp. 151–166, Oct. 2020.

[39] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, "A comprehensive survey on transfer learning," *Proceedings of the IEEE*, vol. 109, no. 1, pp. 43–76, July 2020.

[40] M. Xu, D. T. Hoang, J. Kang, D. Niyato, Q. Yan, and D. I. Kim, "Secure and reliable transfer learning framework for 6g-enabled internet of vehicles," *IEEE Wireless Communications*, May 2022.

[41] M. A. Ferrag, O. Friha, L. Maglaras, H. Janicke, and L. Shu, "Federated deep learning for cyber security in the internet of things: Concepts, applications, and experimental analysis," *IEEE Access*, vol. 9, pp. 138 509–138 542, Oct. 2021.

[42] Y. Meidan, M. Bohadana, Y. Mathov, Y. Mirsky, A. Shabtai, D. Breitenbacher, and Y. Elovici, "N-BaIoT-Network-based detection of IoT botnet attacks using deep autoencoders," *IEEE Pervasive Computing*, vol. 17, no. 3, pp. 12–22, July 2018.

[43] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune: An ensemble of autoencoders for online network intrusion detection," *arXiv preprint arXiv:1802.09089*, Feb. 2018.

[44] "KDD dataset," http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

[45] "NSL-KDD dataset," https://www.unb.ca/cic/datasets/nsl.html.

[46] N. Moustafa and J. Slay, "UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *2015 Military Communications and Information Systems Conference*, Nov. 2015, pp. 1–6.

[47] R. Vinayakumar, M. Alazab, K. Soman, P. Poornachandran, A. Al-Nemrat, and S. Venkatraman, "Deep learning approach for intelligent intrusion detection system," *IEEE Access*, vol. 7, pp. 41 525–41 550, Apr. 2019.

[48] K. K. Nguyen, D. T. Hoang, D. Niyato, P. Wang, D. Nguyen, and E. Dutkiewicz, "Cyberattack detection in mobile cloud computing: A deep learning approach," in *2018 IEEE Wireless Communications and Networking Conference*, Apr. 2018, pp. 1–6.

[49] A. Abeshu and N. Chilamkurti, "Deep learning: The frontier for distributed attack detection in fog-to-things computing," *IEEE Communications Magazine*, vol. 56, no. 2, pp. 169–175, Feb. 2018.

[50] B. Li, Y. Wu, J. Song, R. Lu, T. Li, and L. Zhao, "Deepfed: Federated deep learning for intrusion detection in industrial cyber-physical systems," *IEEE Transactions on Industrial Informatics*, Sep. 2020.

[51] T. D. Nguyen, S. Marchal, M. Miettinen, H. Fereidooni, N. Asokan, and A.-R. Sadeghi, "Dïot: A federated self-learning anomaly detection system for iot," in *2019 IEEE 39th International Conference on Distributed Computing Systems*, July 2019, pp. 756–767.

[52] W. Schneble and G. Thamilarasu, "Attack detection using federated learning in medical cyber-physical systems," in *28th International Conference on Computer Communications and Networks*, July 2019, pp. 1–8.

[53] Y. Chen, X. Qin, J. Wang, C. Yu, and W. Gao, "Fedhealth: A federated transfer learning framework for wearable healthcare," *IEEE Intelligent Systems*, vol. 35, no. 4, pp. 83–93, Apr. 2020.

[54] L. Vu, Q. U. Nguyen, D. N. Nguyen, D. T. Hoang, and E. Dutkiewicz, "Deep transfer learning for IoT attack detection," *IEEE Access*, vol. 8, pp. 107 335–107 344, June 2020.

[55] Y. Sharaf-Dabbagh and W. Saad, "Transfer learning for device fingerprinting with application to cognitive radio networks," in *2015 IEEE 26th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications*, Aug. 2015, pp. 2138–2142.

[56] C. Zhao, Z. Cai, M. Huang, M. Shi, X. Du, and M. Guizani, "The identification of secular variation in IoT based on transfer learning," in *2018 International Conference on Computing, Networking and Communications*, Mar. 2018, pp. 878–882.

[57] Y. Sharaf-Dabbagh and W. Saad, "On the authentication of devices in the internet of things," in *2016 IEEE 17th International Symposium on A World of Wireless, Mobile and Multimedia Networks*, June 2016, pp. 1–3.

[58] Y. S. Dabbagh and W. Saad, "Authentication of wireless devices in the internet of things: Learning and environmental effects," *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 6692–6705, Apr. 2019.

[59] T. Wen and R. Keyes, "Time series anomaly detection using convolutional neural networks and transfer learning," *arXiv preprint arXiv:1905.13628*, May 2019.

[60] T. V. Khoa, Y. M. Saputra, D. T. Hoang, N. L. Trung, D. Nguyen, N. V. Ha, and E. Dutkiewicz, "Collaborative learning model for cyberattack detection systems in iot industry 4.0," in *2020 IEEE Wireless Communications and Networking Conference*, May 2020, pp. 1–6.

[61] Y. Liu, Y. Kang, C. Xing, T. Chen, and Q. Yang, "A secure federated transfer learning framework," *IEEE Intelligent Systems*, vol. 35, no. 4, pp. 70–82, Apr. 2020.

[62] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, June 2006.

[63] D. M. Powers, "Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation," *Journal of Machine Learning Technologies*, pp. 37–63, Oct. 2011.

[64] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[65] V. C.-M. V. Alcacer, "Scanning the industry 4.0: A literature review on technologies for manufacturing systems," *Engineering science and technology, an international journal*, vol. 22, no. 3, pp. 899–919, 2019.

[66] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[67] Y. Sompolinsky and A. Zohar, "Secure high-rate transaction processing in bitcoin," in *International conference on financial cryptography and data security*. Springer, 2015, pp. 507–527.

[68] V. Buterin, "Toward a 12-second blocktime," 2014, Accessed: Feb. 14, 2022. [Online]. Available: https://blog.ethereum.org/2014/07/11/toward-a-12-second-block-time/

[69] G. Mirabelli and V. Solina, "Blockchain and agricultural supply chains traceability: Research trends and future challenges," *Procedia Manufacturing*, vol. 42, pp. 414–421, 2020.

[70] M. Shyamala Devi, R. Suguna, A. S. Joshi, and R. A. Bagate, "Design of iot blockchain based smart agriculture for enlightening safety and security," in *International Conference on Emerging Technologies in Computer Engineering*. Springer, 2019, pp. 7–19.

[71] R. Jabbar, M. Kharbeche, K. Al-Khalifa, M. Krichen, and K. Barkaoui, "Blockchain for the internet of vehicles: A decentralized iot solution for vehicles communication using ethereum," *Sensors*, vol. 20, no. 14, 2020.

[72] D. Ivan, "Moving toward a blockchain-based method for the secure storage of patient records," in *ONC/NIST Use of Blockchain for Healthcare and Research Workshop. Gaithersburg, Maryland, United States: ONC/NIST*. sn, 2016, pp. 1–11.

[73] e. a. W. Hersh, A. Totten, "Health information dissemination from hospital to community care: Current state and next steps in ontario," *Journal of Medical Systems*, vol. 63, no. 50, pp. 425–432, 2016.

[74] M. E. El-hawary, "The smart grid-state-of-the-art and future trends," *Electric Power Components and Systems*, vol. 42, no. 3-4, pp. 239–250, 2014.

[75] Z. Guan, Y. Zhang, L. Zhu, L. Wu, and S. Yu, "Effect: An efficient flexible privacy-preserving data aggregation scheme with authentication in smart grid," *Science China Information Sciences*, vol. 62, no. 3, pp. 1–14, 2019.

[76] P. Zhuang, T. Zamir, and H. Liang, "Blockchain for cybersecurity in smart grid: A comprehensive survey," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 1, pp. 3–19, 2020.

[77] Z. Huang, K. Suankaewmanee, J. Kang, D. Niyato, and N. P. Sin, "Development of reliable wireless communication system for secure blockchain-based energy trading," in *2019 16th International Joint Conference on Computer Science and Software Engineering (JCSSE)*. IEEE, 2019, pp. 126–130.

[78] J. Gao, K. O. Asamoah, E. B. Sifah, A. Smahi, Q. Xia, H. Xia, X. Zhang, and G. Dong, "Gridmonitoring: Secured sovereign blockchain based monitoring on smart grid," *IEEE Access*, vol. 6, pp. 9917–9925, 2018.

[79] K. Gai, Y. Wu, L. Zhu, M. Qiu, and M. Shen, "Privacy-preserving energy trading using consortium blockchain in smart grid," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 6, pp. 3548–3558, 2019.

[80] G. Bansal, A. Dua, G. S. Aujla, M. Singh, and N. Kumar, "Smartchain: a smart and scalable blockchain consortium for smart grid systems," in *2019 IEEE International Conference on Communications Workshops (ICC Workshops)*. IEEE, 2019, pp. 1–6.

[81] A. Kumari, R. Gupta, S. Tanwar, S. Tyagi, and N. Kumar, "When blockchain meets smart grid: Secure energy trading in demand response management," *IEEE Network*, vol. 34, no. 5, pp. 299–305, 2020.

[82] A. Ghasempour, "Internet of things in smart grid: Architecture, applications, services, key technologies, and challenges," *Inventions*, vol. 4, no. 1, p. 22, 2019.

[83] M. Raikwar, D. Gligoroski, and K. Kralevska, "Sok of used cryptography in blockchain," *IEEE Access*, vol. 7, pp. 148 550–148 575, 2019.

[84] G. Wood, "Ethereum: A secure decentralised generalised transactionledger, petersburg version 41c1837," 2021, Accessed: Feb. 14, 2022. [Online]. Available: https://ethereum.github.io/yellowpaper/paper.pdf

[85] M. Alharby and A. van Moorsel, "Blocksim: An extensible simulation tool for blockchain systems," *Frontiers in Blockchain*, vol. 3, p. 28, 2020.

[86] "Ethereum vision," Accessed: Feb. 14, 2022. [Online]. Available: https://ethereum.org/en/eth2/vision

[87] M. J. Dworkin, "Sp 800-38a 2001 edition. recommendation for block cipher modes of operation: Methods and techniques," Gaithersburg, MD, USA, Tech. Rep., 2001.

[88] C. Decker and R. Wattenhofer, "Information propagation in the bitcoin network," in *IEEE P2P 2013 Proceedings*, 2013, pp. 1–10.

[89] H. Kang, X. Chang, R. Yang, J. Misic, and V. B. Misic, "Understanding selfish mining in imperfect bitcoin and ethereum networks with extended forks," *IEEE Transactions on Network and Service Management*, vol. 18, no. 3, pp. 3079–3091, 2021.

[90] E. Statistics, "Ethereum uncle rate," Accessed: Feb. 14, 2022. [Online]. Available: https://ycharts.com/indicators/ethereum_uncle_rate

[91] M. S. Ali, M. Vecchio, M. Pincheira, K. Dolui, F. Antonelli, and M. H. Rehmani, "Applications of blockchains in the Internet of Things: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 2, pp. 1676–1717, Dec. 2018.

[92] J. Xie, H. Tang, T. Huang, F. R. Yu, R. Xie, J. Liu, and Y. Liu, "A survey of blockchain technology applied to smart cities: Research issues and challenges," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2794–2830, Feb. 2019.

[93] S. Biswas, K. Sharif, F. Li, Z. Latif, S. S. Kanhere, and S. P. Mohanty, "Interoperability and synchronization management of blockchain-based decentralized e-health systems," *IEEE Transactions on Engineering Management*, vol. 67, no. 4, pp. 1363–1376, June 2020.

[94] "The 10 Biggest Crypto Exchange Hacks In History," Accessed: Feb. 14, 2022. [Online]. Available: https://crystalblockchain.com/articles/the-10-biggest-crypto-exchange-hacks-in-history

[95] "North Korean Hackers Have Prolific Year as Their Unlaundered Cryptocurrency Holdings Reach All-time High," Accessed: Feb. 14, 2022. [Online]. Available: https://blog.chainalysis.com/reports/north-korean-hackers-have-prolific-year-as-their-total-unlaundered-cryptocurrency-holdings-reach-all-time-high

[96] C. T. Nguyen, D. T. Hoang, D. N. Nguyen, D. Niyato, H. T. Nguyen, and E. Dutkiewicz, "Proof-of-stake consensus mechanisms for future blockchain networks: fundamentals, applications and opportunities," *IEEE Access*, vol. 7, pp. 85 727–85 745, Jun. 2019.

[97] K. Salah, N. Nizamuddin, R. Jayaraman, and M. Omar, "Blockchain-based soybean traceability in agricultural supply chain," *IEEE Access*, vol. 7, pp. 73 295–73 305, May 2019.

[98] N. Chaabouni, M. Mosbah, A. Zemmari, C. Sauvignac, and P. Faruki, "Network intrusion detection for IoT security based on learning techniques," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2671–2701, Jan. 2019.

[99] N. Sultana, N. Chilamkurti, W. Peng, and R. Alhadad, "Survey on SDN based network intrusion detection system using machine learning approaches," *Peer-to-Peer Networking and Applications*, vol. 12, no. 2, pp. 493–501, Mar. 2019.

[100] M. Idhammad, K. Afdel, and M. Belouch, "Distributed intrusion detection system for cloud environments based on data mining techniques," *Procedia Computer Science*, vol. 127, pp. 35–41, Jan. 2018.

[101] K.-D. Lu, G.-Q. Zeng, X. Luo, J. Weng, W. Luo, and Y. Wu, "Evolutionary deep belief network for cyber-attack detection in industrial automation and control system," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 11, pp. 7618–7627, Nov. 2021.

[102] L. Vu, V. L. Cao, Q. U. Nguyen, D. N. Nguyen, D. T. Hoang, and E. Dutkiewicz, "Learning latent representation for iot anomaly detection," *IEEE Transactions on Cybernetics*, pp. 1–14, Sep. 2020.

[103] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, "A survey on deep transfer learning," in *Int. Conf. on Artificial Neural Networks*. Rhodes, Greece: Springer, Oct. 2018, pp. 270–279.

[104] M. U. Hassan, M. H. Rehmani, and J. Chen, "Anomaly detection in blockchain networks: A comprehensive survey," *arXiv preprint arXiv:2112.06089*, Dec. 2021. [Online]. Available: https://arxiv.org/abs/2112.06089

[105] P. Kumar, R. Kumar, G. Gupta, and R. Tripathi, "A distributed framework for detecting DDoS attacks in smart contract-based Blockchain-IoT systems by leveraging fog computing," *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 6, pp. 1–31, June 2021.

[106] O. Alkadi, N. Moustafa, B. Turnbull, and K.-K. R. Choo, "A deep blockchain framework-enabled collaborative intrusion detection for protecting IoT and cloud networks," *IEEE Internet of Things Journal*, vol. 8, no. 12, pp. 9463–9472, June 2020.

[107] J. Kim, M. Nakashima, W. Fan, S. Wuthier, X. Zhou, I. Kim, and S.-Y. Chang, "Anomaly detection based on traffic monitoring for secure blockchain networking," in *2021 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, Sydney, Australia, May 2021, pp. 1–9.

[108] Z. Liu and X. Yin, "LSTM-CGAN: towards generating low-rate DDoS adversarial samples for blockchain-based wireless network detection models," *IEEE Access*, vol. 9, pp. 22 616–22 625, Feb. 2021.

[109] W. Cao, Y. Huang, D. Li, F. Yang, X. Jiang, and J. Yang, "A blockchain based link-flooding attack detection scheme," in *2021 IEEE 4th Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)*, vol. 4, Chongqing, China, June 2021, pp. 1665–1669.

[110] "Wireshark Network Analysis," Accessed: Feb. 14, 2022. [Online]. Available: https://www.wireshark.org

[111] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, July 2006.

[112] ——, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[113] J. Konecny, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *arXiv preprint arXiv:1610.05492*, 2016.

[114] Ethereum, "Official Go implementation of the Ethereum protocol," Accessed: Feb. 14, 2022. [Online]. Available: https://github.com/ethereum/go-ethereum

[115] M. Oance, "Ethereum Network Stats," Accessed: Feb. 14, 2022. [Online]. Available: https://github.com/cubedro/eth-netstats

[116] T. Neudecker and H. Hartenstein, "Network layer aspects of permissionless blockchains," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 838–857, Sep. 2019.

[117] M. Saad, J. Spaulding, L. Njilla, C. Kamhoua, S. Shetty, D. Nyang, and D. Mohaisen, "Exploring the attack surface of blockchain: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 1977–2008, Mar. 2020.

[118] "Bitfinex restored after DDoS attack," Accessed: Feb. 14, 2022. [Online]. Available: https://www.computerweekly.com/news/450431741/Bitfinex-restored-after-DDoS-attack

[119] J. Otávio Chervinski, D. Kreutz, and J. Yu, "Analysis of transaction flooding attacks against Monero," in *2021 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, Sydney, Australia, May 2021, pp. 1–8.

[120] "kdd99_feature_extractor," AI-IDS, Accessed: Feb. 14, 2022. [Online]. Available: https://github.com/AI-IDS/kdd99_feature_extractor

[121] "KDD Cup 1999 Data," The Fifth International Conference on Knowledge Discovery and Data Mining, Accessed: Feb. 14, 2022. [Online]. Available: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[122] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of Machine Learning Research*, vol. 9, no. 86, pp. 2579–2605, Nov. 2008.

[123] A. Mukherjee, S. A. A. Fakoorian, J. Huang, and A. L. Swindlehurst, "Principles of physical layer security in multiuser wireless networks: A survey," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 3, pp. 1550–1573, 2014.

[124] M. Bloch and J. Barros, *Physical-layer security: from information theory to security engineering*. Cambridge University Press, 2011.

[125] J. M. Hamamreh, H. M. Furqan, and H. Arslan, "Classifications and applications of physical layer security techniques for confidentiality: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 2, pp. 1773–1828, 2018.

[126] A. D. Wyner, "The wire-tap channel," *Bell system technical journal*, vol. 54, no. 8, pp. 1355–1387, 1975.

[127] S. Goel and R. Negi, "Guaranteeing secrecy using artificial noise," *IEEE transactions on wireless communications*, vol. 7, no. 6, pp. 2180–2189, 2008.

[128] B. Akgun, O. O. Koyluoglu, and M. Krunz, "Exploiting full-duplex receivers for achieving secret communications in multiuser miso networks," *IEEE Transactions on Communications*, vol. 65, no. 2, pp. 956–968, 2016.

[129] J. Choi, "A robust beamforming approach to guarantee instantaneous secrecy rate," *IEEE Transactions on Wireless Communications*, vol. 15, no. 2, pp. 1076–1085, 2015.

[130] A. Mukherjee and A. L. Swindlehurst, "Robust beamforming for security in mimo wiretap channels with imperfect csi," *IEEE Transactions on Signal Processing*, vol. 59, no. 1, pp. 351–361, 2010.

[131] T. Erpek, T. J. O'Shea, and T. C. Clancy, "Learning a physical layer scheme for the mimo interference channel," in *2018 IEEE International Conference on Communications (ICC)*. IEEE, 2018, pp. 1–5.

[132] T. J. O'Shea, T. Erpek, and T. C. Clancy, "Deep learning based mimo communications," *arXiv preprint arXiv:1707.07980*, 2017.

[133] R. Fritschek, R. F. Schaefer, and G. Wunder, "Deep learning for the gaussian wiretap channel," in *ICC 2019-2019 IEEE International Conference on Communications (ICC)*. IEEE, 2019, pp. 1–6.

[134] K.-L. Besser, C. R. Janda, P.-H. Lin, and E. A. Jorswieck, "Flexible design of finite block-length wiretap codes by autoencoders," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 2512–2516.

[135] X. Zhang and M. Vaezi, "Deep learning based precoding for the mimo gaussian wiretap channel," *arXiv preprint arXiv:1909.07963*, 2019.

[136] S. Dorner, S. Cammerer, J. Hoydis, and S. t. Brink, "Deep learning based communication over the air," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 1, pp. 132–143, 2018.

[137] M. I. Belghazi, A. Baratin, S. Rajeswar, S. Ozair, Y. Bengio, A. Courville, and R. D. Hjelm, "Mine: mutual information neural estimation," *arXiv preprint arXiv:1801.04062*, 2018.

[138] R. Fritschek, R. F. Schaefer, and G. Wunder, "Deep learning for channel coding via neural mutual information estimation," *arXiv preprint arXiv:1903.02865*, 2019.

[139] N. A. Letizia and A. M. Tonello, "Capacity-driven autoencoders for communications," *IEEE Open Journal of the Communications Society*, vol. 2, pp. 1366–1378, 2021.

[140] T. Lin and Y. Zhu, "Beamforming design for large-scale antenna arrays using deep learning," *IEEE Wireless Communications Letters*, vol. 9, no. 1, pp. 103–107, 2019.

[141] B. M. Tuan, T. D. Tuyen, N. L. Trung, and N. V. Ha, "Autoencoder based friendly jamming," in *2020 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2020, pp. 1–6.

[142] C. Wang, E. K. S. Au, R. D. Murch, W. H. Mow, R. S. Cheng, and V. Lau, "On the performance of the mimo zero-forcing receiver in the presence of channel estimation error," *IEEE Transactions on Wireless Communications*, vol. 6, no. 3, pp. 805–810, 2007.

[143] R. Negi and S. Goel, "Secret communication using artificial noise," in *VTC-2005-Fall. 2005 IEEE 62nd Vehicular Technology Conference*, vol. 3. IEEE, 2005, pp. 1906–1910.

[144] E. Telatar, "Capacity of multi-antenna gaussian channels," *European transactions on telecommunications*, vol. 10, no. 6, pp. 585–595, 1999.

[145] Y. Bengio, I. Goodfellow, and A. Courville, *Deep learning*. MIT Proess, 2017, vol. 1.

[146] T. O'Shea and J. Hoydis, "An introduction to deep learning for the physical layer," *IEEE Transactions on Cognitive Communications and Networking*, vol. 3, no. 4, pp. 563–575, 2017.

[147] C. E. Shannon, "Communication theory of secrecy systems," *Bell system technical journal*, vol. 28, no. 4, pp. 656–715, 1949.

[148] B. Poole, S. Ozair, A. Van Den Oord, A. Alemi, and G. Tucker, "On variational bounds of mutual information," in *International Conference on Machine Learning*. PMLR, 2019, pp. 5171–5180.

[149] D. B. F. Agakov, "The im algorithm: a variational approach to information maximization," 2004.

[150] D. McAllester and K. Stratos, "Formal limitations on the measurement of mutual information," in *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, S. Chiappa and R. Calandra, Eds., vol. 108. PMLR, 26–28 Aug 2020, pp. 875–884. [Online]. Available: http://proceedings.mlr.press/v108/mcallester20a.html

[151] Z. Qin, D. Kim, and T. Gedeon, "Rethinking softmax with cross-entropy: Neural network classifier as mutual information estimator," 2020.

[152] B. Sklar, "Rayleigh fading channels in mobile digital communication systems. ii. mitigation," *IEEE Communications magazine*, vol. 35, no. 7, pp. 102–109, 1997.

[153] H. Harada and R. Prasad, *Simulation and software radio for mobile communications*. Artech House, 2002.

[154] P. Molchanov, S. Tyree, T. Karras, T. Aila, and J. Kautz, "Pruning convolutional neural networks for resource efficient inference," *arXiv preprint arXiv:1611.06440*, 2016.

[155] "SWC Registry - Smart Contract Weakness Classification and Test Cases," accessed: Feb. 14, 2022. [Online]. Available: https://swcregistry.io/

[156] "A disastrous vulnerability found in smart contracts of BeautyChain (BEC)," accessed: Feb. 14, 2022. [Online]. Available: https://medium.com/secbit-media/a-disastrous-vulnerability-found-in-smart-contracts-of-beautychain-bec-dbf24ddbc30e

[157] "How $800k Evaporated from the PoWH Coin Ponzi Scheme Overnight," accessed: Feb. 14, 2022. [Online]. Available: https://medium.com/@ebanisadr/how-800k-evaporated-from-the-powh-coin-ponzi-scheme-overnight-1b025c33b530

[158] "Reentrancy Attack," accessed: Feb. 14, 2022. [Online]. Available: https://hackernoon.com/hack-solidity-reentrancy-attack

[159] "GovernMental Jackpot," accessed: Feb. 14, 2022. [Online]. Available: http://governmental.github.io/GovernMental/