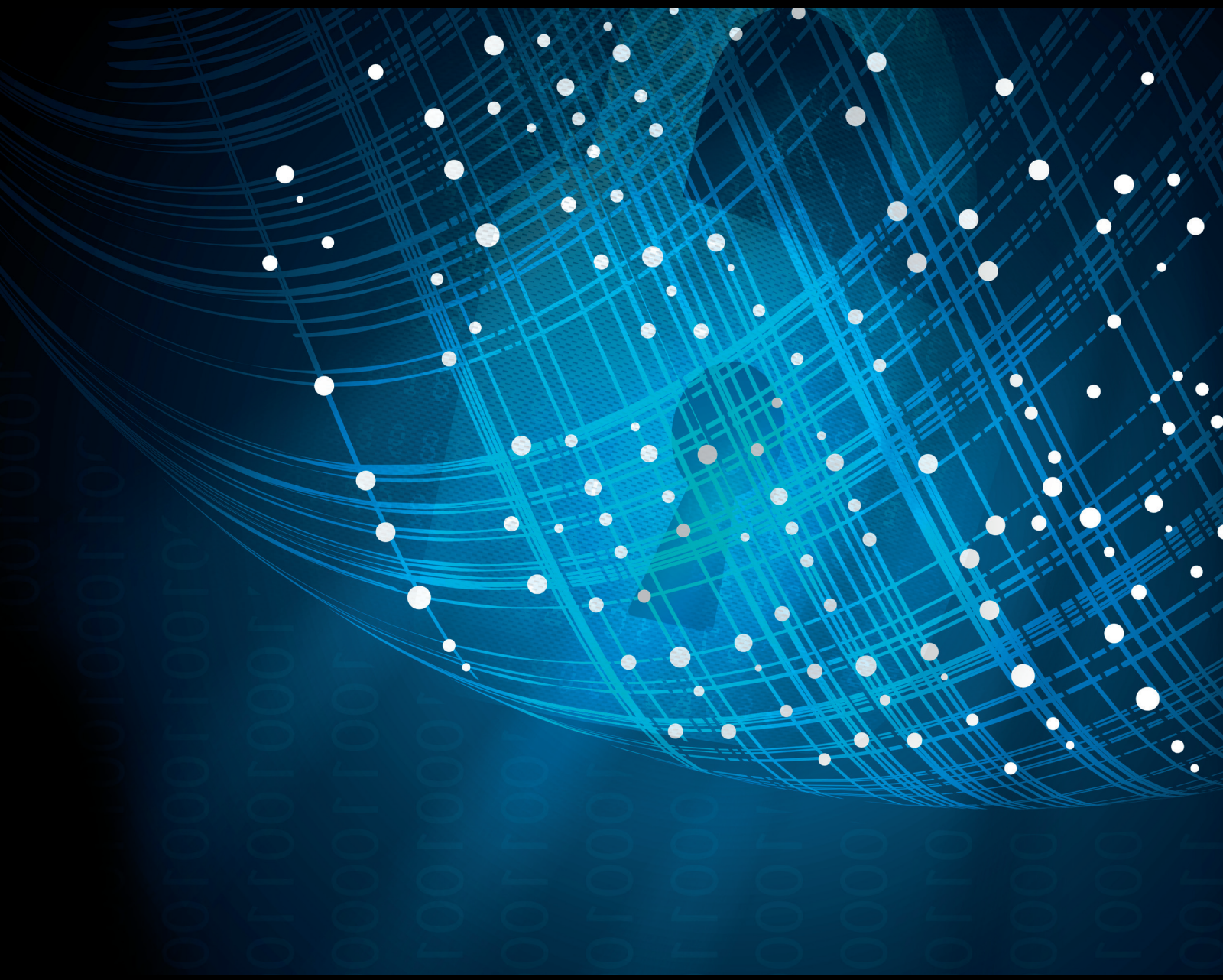
Security and Communication Networks

# Communication Security in Socialnet-Oriented Cyber Spaces 2022

Lead Guest Editor: Ilsun You
Guest Editors: Karl Andersson, Zheli Liu, Hao Peng, and Gökhan Kul



# Communication Security in Socialnet-Oriented Cyber Spaces 2022

Security and Communication Networks

# Communication Security in Socialnet-Oriented Cyber Spaces 2022

Lead Guest Editor: Ilsun You
Guest Editors: Karl Andersson, Zheli Liu, Hao Peng, and Gökhan Kul

# Contents

Hindawi
Security and Communication Networks
Volume 2023, Article ID 6013270, 13 pages
https://doi.org/10.1155/2023/6013270

*Research Article*

# FACSC: Fine-Grained Access Control Based on Smart Contract for Terminals in Software-Defined Network

**Bingcheng Jiang**,[1] **Qian He**,[1] **Mingliu He**,[1] **Zhongyi Zhai**,[1] **and Baokang Zhao**[2]

[1]*College of Computers and Information Security, Guilin University of Electronic Technology, Guilin, China*
[2]*College of Computers, National University of Defense Technology, Changsha, Hunan, China*

Correspondence should be addressed to Qian He; heqian@guet.edu.cn

Received 14 October 2022; Revised 3 March 2023; Accepted 13 April 2023; Published 15 May 2023

Academic Editor: Zhe-Li Liu



Physical terminals provide network services to upper-layer applications, but their limited memory and processing power make it challenging to perform security updates and patches, leaving them vulnerable to known security threats. Attackers can exploit these weaknesses to control the terminals and attack the network. To restrict unauthorized access to the network and its resources, appropriate access control mechanisms are necessary. In this paper, we propose a fine-grained access control method based on smart contracts (FACSC) for terminals in software-defined networking (SDN). FACSC utilizes the attribute-based access control (ABAC) model to achieve fine-grained control over terminal access networks. To ensure the security and reliability of access control policies and terminal-related attribute information, we utilize smart contract technology to implement the ABAC model. Furthermore, we leverage the programming protocol-independent packet processor (P4) to filter and forward packets in the data plane based on the packet option field, enabling rapid terminal access. Experimental results show that our proposed method achieves fine-grained secure authentication of terminals in SDN networks with a low authentication processing overhead.

## 1. Introduction

With the increasing adoption of emerging technologies in various fields, such as the Internet of Things, social networks, and mobile Internet, there is a growing need for proper management of large-scale dynamic networks [1]. Fortunately, software-defined networking (SDN) offers a viable solution to this pressing problem. SDN innovatively changes the existing network structure by dividing it into a data plane and a control plane [2], making it possible to optimize network resource allocation and improve network quality. However, the open and untrustworthy network environment of SDN leaves it vulnerable to attackers who may use forged user identities or malicious terminals to attack the network [3, 4].

Access control is a standard approach to safeguard valuable resources from illegal access by unauthorized users or improper use by authorized ones. However, the native SDN controller lacks access control mechanisms for terminal access and cannot perform authentication functions for terminals. As a result, malicious terminals can gain access to the SDN and launch various attacks, leaving the entire network vulnerable to known attacks, such as denial-of-service (DoS) attacks.

The identifier network [5] presents a new possibility for access control of terminals in SDN. By utilizing identifier network technology, all terminals can be uniformly bound with attributes, which makes each terminal unique by its set of attributes, providing support for developing access control policies for terminals. The attribute-based access control model (ABAC) has made a significant breakthrough in addressing complex access control policies, access control granularity, and dynamic scaling of terminal access [6, 7]. ABAC introduces the idea of entity attributes, which describe subject, object, operation, and environment attributes in a unified manner. This makes ABAC an appropriate solution for addressing the problem of secure and controllable network access for many terminals. The ABAC-based scheme [8–11] implements policy-based access

control, which combines various types of attributes (subject, object, operation, and environment attributes). These schemes grant access rights to subjects by defining a set of rules.

It is worth noting that in the aforementioned schemes, authentication of the subject's access rights is usually performed by a centralized entity, which is vulnerable to single points of failure. To avoid the aforementioned security problems, some attempts have been made in recent literature to solve the distributed authentication problem using blockchain technology [7, 12].

Blockchain can be technically understood as a distributed database without the problem of a centralized single point of failure. And the tamper-evident and traceable nature of blockchain can strongly endorse the data on the chain. Thanks to the invention of smart contracts (executable code residing in the blockchain), the blockchain has now evolved into a promising platform for developing distributed and trusted applications. It has attracted much attention from researchers in the SDN community [13, 14]. Predictably, blockchain technology is emerging as a key enabler for achieving distributed and trusted access control.

In this paper, the ABAC model is implemented as smart contracts with the help of blockchain technology, which makes the access control policy free from malicious tampering and enables secure and controlled access to the SDN network for terminals.

The main work of this paper consists of the following.

(1) This paper proposes a fine-grained access control mechanism based on smart contracts for terminals in SDN-based networks (FACSC). It leverages attributes to identify terminals uniquely, and network administrators combine multiple attributes to formulate access control policies based on the ABAC model.

(2) In FACSC, we improve token-based authorization by introducing blockchain and ABAC. Dedicated smart contracts are designed to encapsulate, distribute and verify tokens to satisfy terminals' decentralized, reliable, and flexible access control requirements.

(3) We introduce the P4 forwarding device to realize SDN data plane programmability, which enables fast packet filtering by parsing data streams in the P4 control plane.

The remainder of this paper is organized as follows: Section 2 reviews the related work in recent years. Section 3 introduces the preliminary P4, smart contracts, and ABAC situation. Section 4 introduces the system model. Section 5 presents the proposed ABAC access control scheme based on smart contracts. Section 6 presents the experiments and performance evaluation. Section 7 concludes this article.

## 2. Related Work

As SDN technology becomes increasingly mature, it has been widely used in production environments, such as Google Cloud Data Center, Huawei Cloud Data Center, etc. In addition, it has also been used in some higher education institutions, such as Stanford University and Tsinghua University, which have implemented SDN as the basic network architecture. However, SDN-based networks also have security threats, such as the lack of access control mechanisms for terminal access.

Duy et al. [15] construct an access control scheme for SDN northbound, introducing the B-DAC framework for decentralized authentication and fine-grained access control for northbound interfaces, which assists administrators in managing and protecting critical resources, indirectly enabling terminals access functionality. Kammoun et al. [16] propose a new SDN architecture based on IoT trust management and access control, where a predefined trust management algorithm calculates the terminal's trust value. Based on trust value, malicious devices are prevented from accessing the network. However, this scheme does not consider the security of the access control policy and is prone to problems such as policy leakage. Awasthi et al. [17] design a scalable, efficient, and cost-effective network architecture that not only meets the changing needs of users but also increases the number of accessing IoT devices, which embed network elements in software rather than dedicated hardware, making it easy to rent from the pool of available devices, enabling rapid device access. Matias et al. [18] propose FlowNAC, an access control scheme for SDNs, which grants users access to the network based on the user's requested target and implements fine-grained access control functionality. However, this scheme is time-consuming and relies on third-party authorization for data flow access, which is likely to be insecure. Benzekki et al. [19] propose an access authentication model based on an SDN network by improving the 802.1X protocol, in which a switch supporting the 802.1X protocol must exist and DHCP and RADIUS servers are connected to this switch to reduce the communication latency with the controller, but the disadvantage of this model is the lack of flexibility in deployment. Fathima and Vennila [20] propose a new algorithm for building IEEE 802.1X-based port authentication schemes, which extends the implementation of EAP from 802.1X to the application and control layers in IPv6, thus improving network throughput and terminals authentication efficiency. However, this scheme is highly targeted and cannot flexibly provide authentication services for more terminals. Ferrazani and Duarte [21] propose an access control model that combines information about users with OpenFlow flow tables, which solves the problem of fine-grained user access control. Still, the model has a single authentication method and is less scalable when the number of terminals is large. Hesham et al. [22] propose a simple authentication model for M2M, which provides different levels of access rights and bandwidth for users through controllers. Still, this model has a single authentication method and cannot be applied to large data centre networks. Yakasai and Guy [23] propose a virtualized network access control scheme based on SDN architecture, which provides a virtualized network access control scheme by combining a stateful role-based firewall with an authorization process to provide a solution for endpoint access control in enterprise networks; however,

this work is applied to a single domain and is not very pervasive.

Although access control policies are considered in some of the previous literature, they are often inflexible and not robust enough regarding security. Furthermore, these schemes do not address the issue of reducing access time overhead. As a result, problems such as high authentication overhead and tampering with control policies may arise.

## 3. Preliminaries

*3.1. Attribute-Based Access Control.* In response to the challenge of dynamic and fine-grained access control, which cannot be effectively addressed by traditional models, researchers have proposed the attribute-based access control model (ABAC) [6]. Unlike other models, the ABAC model determines a user's access control privileges based on their entity attributes rather than solely on their identity, eliminating the need for the explicit privilege granted to a subject. The structure of the ABAC model is shown in Figure 1. The core elements of the ABAC model include subject, object, environment, and operational constraints, all of which are described using attributes and attribute values. The generation of access control policies is composed of entity attributes in a flexible way, which improves the representability of access control policies and the model's flexibility. In addition, the ABAC model can also represent the permissions used to control roles and security in other access control models in the form of attributes.

Therefore, the ABAC model is suitable for controlling massive data access. In the terminals access control designed in this paper, attributes are used to identify terminals, making the terminal access flexible and controllable.

*3.2. Programming Protocol-Independent Packet Processor (P4).* SDN divides the traditional network architecture into the control plane and the data plane, which becomes more flexible than the traditional network but also has some drawbacks. OpenFlow was designed to control only the forwarding behaviour of switches and routers, which limits its ability to manage network traffic and resources. As networks grow larger and more complex, OpenFlow may be unable to handle the increased traffic and routing demands. Because OpenFlow allows for remote control of network devices, there are concerns about security vulnerabilities and potential attacks. OpenFlow is not a standardized protocol, which means that there may be interoperability issues between different vendors' devices and software [24]. To solve the problem of poor scalability caused by OpenFlow's own design, Bosshart et al. [25] proposed the Programming Protocol-Independent Packet Processors (P4) language and the corresponding forwarding model [26, 27]. With the data plane programming capability brought by P4, administrators can not only implement existing network device functions and network protocols such as bridges, routers, and firewalls but also easily support new protocols including VxLAN and RCP [28].

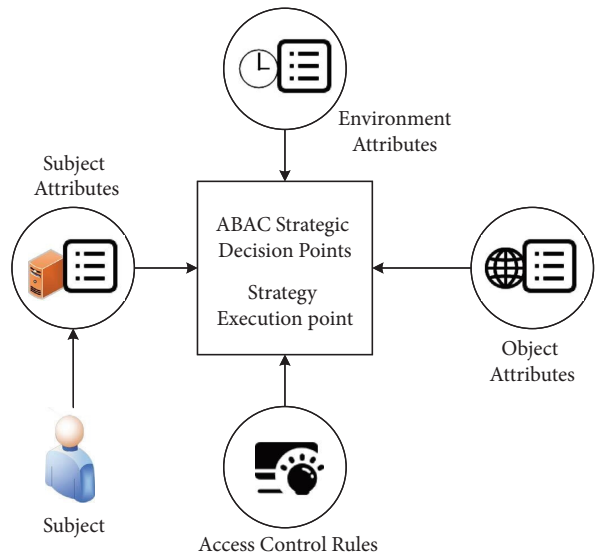



Figure 1: Model structure.

P4 has the language properties of reconfigurability, protocol independence, and platform independence. To this end, P4 defines a set of abstract forwarding models to support the above three language properties. The abstract forwarding model consists of three main components.

(1) **Header parser:** P4 enables developers to customize the packet header structure and parsing process and to configure the debugged P4 code into programmable hardware devices that support P4. This allows for flexible parsing of various packet formats. Upon receiving a message from a terminal, the P4 programmable device follows the message processing logic to separate the packet header from the payload. The information within the packet header is then stored in a self-defined field according to the state transfer rules of the parsing graph, allowing for matching with the flow table in the subsequent pipeline.

(2) **Multilevel pipeline:** This includes the ingress pipeline and the egress pipeline. The ingress pipeline is responsible for modifying the data grouping and determining the port from which the data is forwarded. The egress pipeline only has the function of modifying the attributes associated with the messages. If the researchers want to fulfil the custom business requirements, they have to customize the information in the P4 code such as the matching header field, the executive action and parameters, the number of flow tables in each match action table (MAT), and decide the execution order of the MAT [29].

(3) **Control program:** The written P4 program can be compiled by the P4 compiler to generate a control interface for data parsing or matching. Through this interface, data flow forwarding rules can be installed for the data plane, and hardware facilities such as counters and registers can also be configured

through this interface, as well as statistical information on the status of P4 forwarding devices during operation.

In this paper, we use P4 forwarding devices to implement packet processing in the data plane.

*3.3. Blockchain and Smart Contracts.* Blockchain is a decentralized digital ledger technology that allows data to be recorded and stored in a secure, transparent, and tamper-proof manner. It was originally developed for the cryptocurrency Bitcoin but has since been applied to a wide range of industries and use cases [7, 30–32]. Transaction information is stored in blocks containing timestamps and references to the previous block and grows as a chain, which is maintained by all participants, and the consistency of the ledger is ensured by consensus algorithms [33]. According to the access rules, blockchains can be divided into public blockchains and consortium blockchains. For public blockchains, participants are free to join and withdraw, and the number of participants is not fixed, as in the case of Bitcoin [34]. For consortium blockchain, only authorized users can join, and the set of participants is usually predefined, such as IBM's hyperledger fabric. With its transparent, traceable, and robust features, blockchain can establish reliable trust between unknown parties and is an effective solution to replace vulnerable central servers in insecure environments. As a blockchain with access control, the consortium blockchain is suitable for access control scenarios that require prevetted users and a relatively stable set of participants.

A smart contract is a concept introduced by cryptographer Nick Szabo in the 1990s. However, due to the lack of a trusted execution environment at that time, smart contracts were not widely applied or developed until the emergence of Ethereum. With the introduction of Ethereum, smart contracts were revitalized and began to gain more attention and use. Smart contracts are designed to eliminate reliance on traditional trusted third parties and are deployed on physical hardware to generate a variety of flexible and controllable smart assets. The life cycle of a smart contract consists of six phases: negotiation, development, deployment, operation and maintenance, learning, and self-destruction. Among them, the development phase includes functional testing of the contract to ensure the correctness of its results, and the learning phase includes operational feedback and updates to the smart contract. In the fabric network, the debugged contract is wrapped in the form of a Docker image, installed in the form of a Docker container in each peer node, and the Init method in the contract is executed after the installation. The installed contract will wait to be invoked by the related business.

In this paper, the ABAC model is implemented as a smart contract, and the contract interface is encapsulated as a Restful service using Fabric-Java-SDK and SpringBoot technology. The encapsulated Restful service is used to realize the functions of terminal access verification and data storage.

## 4. System Model

We propose a fine-grained terminal access control method based on ABAC and smart contracts to address the lack of effective terminal access control mechanisms in SDN-based networks. The system model, shown in Figure 2, comprises terminals/devices, P4 forwarding devices (P4FD), a blockchain platform, attribute management center (AMC), controllers, and OpenFlow Switches (OFS).

(i) **Terminal/Device:** A terminal is a client used by a user to access the SDN-based network. A device is the carrier of the network resource that the terminal wants to access. When the terminal tries to access the SDN-based network, it will put its own attributes into the *Options* field of the IP packet and send it to the P4FD.

(ii) **P4FD:** The P4FD is responsible for packet processing, including parsing IP packets sent by terminals, filtering out packets without *Options*, and forwarding packets with *Options*. Additionally, it can mirror the packets to the P4 control plane, enabling fast access for the terminals.

(iii) **Blockchain:** The blockchain is the core component of the access control model, and all nodes are required to be authenticated by the Certificate Authority when they join the blockchain system. In our scheme, the blockchain has the following two functions.

(1) The ABAC model is implemented through smart contracts, which mainly include three kinds of contracts, namely, policy contract (PC), device contract (DC), and access contract (AC). The PC formulates access control policies for terminal access to the SDN network based on terminal attributes, device attributes, operation attributes, and environment attributes and stores the policies on the blockchain. The DC stores the set of attributes submitted by the AMC in the blockchain state database and provides attribute support for the PC. The AC adjudicates whether the terminal has the authority to access SDN network resources according to the access control policy.

(2) Provide RestFul service for network administrators to implement smart contract addition, modification, deletion, and query operations.

(iv) **AMC:** AMC is divided into subject AMC (SAMC) and object AMC (OAMC), with two main functions.

(1) The SAMC manages the attribute sets of terminals and submits the attribute sets to the blockchain in batches to prevent terminals from interacting with the blockchain directly and improve the performance of the blockchain.

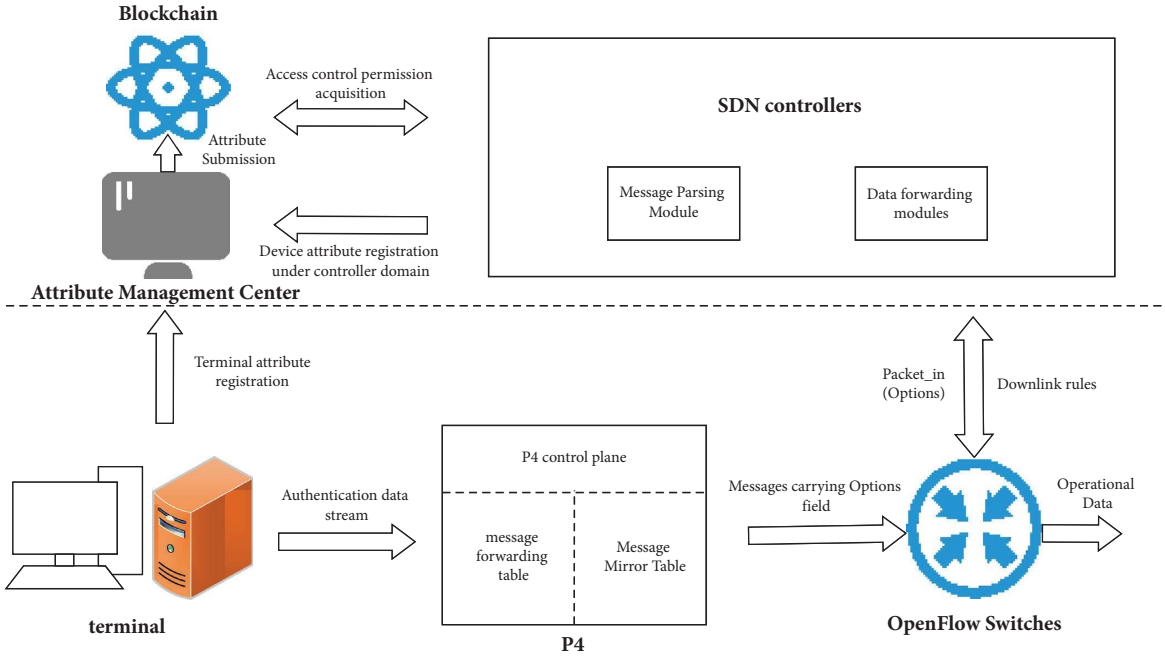



Figure 2: System model.

(2) The OAMC manages the attribute sets of devices under the SDN-controllers domain and submits them to the blockchain in batches.

(v) **Controller:** The main modules developed in the controller are the message parsing module and the data forwarding module.

The main function of the message parsing module is to get the IP packets carrying the *Options* field after filtering by the P4FD. The controller parses the value of the *Options* field, constructs the access request for the current terminal based on the parsing result, and submits the request to the AC through the encapsulated RestFul service. The AC verifies whether the terminal has permission and returns the response status code to the controller.

The main function of the data forwarding module is that the controller determines whether to issue a flow table to instruct the OpenFlow switch to forward messages based on the status code.

(vi) **OFS:** In this paper, we use OpenvSwitch (OvS) as OpenFlow switches, whose main function is to encapsulate packets sent by P4FD into Packet_in messages for forwarding to the controller and to forward the messages normally according to the flow table issued by the controller.

## 5. ABAC and Smart Contract-Based Access Control

To address the lack of effective access control for terminal access in software-defined networks, we propose an ABAC and smart contract-based access control method for terminal access. First, the ABAC model is formally defined, followed by a detailed description of the access control process.

*5.1. ABAC Model.* The attribute is the core concept of the ABAC model, which is described by a four-element set $\langle S_i, O_j, P_k, E_n \rangle$. The meaning of each element is explained as follows:

$S_i$ represents the *i-th* subject attribute, which is the terminals attribute, and uses $S_i = [A_{S_i}: V_{S_i}]$ to denote any one attribute item and attribute value in the subject, where $A_{S_i}$ denotes the subject attribute name, such as terminal Mac, IP, etc., and $V_{S_i}$ denotes the attribute value corresponding to the subject attribute name. $O_j$ represents the *j-th* object attribute, which is the devices attribute, and $O_j = [A_{O_i}: V_{O_i}]$ is used to denote any one attribute item and attribute value in the object. $P_k = [A_{P_k}: V_{P_k}]$ represents the operation of the subject on the object, such as read, add, execute, etc. Here, there are two values of $V_{P_k}$, when $V_{P_k}$ is 1, it means that the terminal is allowed to access the SDN network, and when $V_{P_k}$ is 0 or other values, the terminal is denied access to the SDN network. $E_n = [A_{E_n}: V_{E_n}]$ represents the environment attribute, which indicates the environment attribute required by the access control policy when the current subject accesses the object. $E_n$ represents environment attributes, indicating the environment properties required by the access control policy when the current subject accesses an object. $E_n = [A_{E_n}: V_{E_n}]$ represents any attribute item and value in the environment attributes. $A_{E_n}$ represents any attribute name in the environment attributes, such as the policy effective time, allowed terminal MAC, and IP information. $V_{E_n}$ represents the attribute value corresponding to the attribute name in the environment attributes.

*Definition 1.* An attribute group $AG_m = \{[A_{m_1}: V_{m_1}] \wedge [A_{m_2}: V_{m_3}] \wedge \cdots \wedge [A_{m_p}: V_{m_p}]\}$ represents a collection of attribute items of the same type, where $m \in \{S_i, O_j, P_k, E_n\}$.

*Definition 2.* An attribute access request $AAR = \{AG_{S_i} \wedge AG_{O_j} \wedge AG_{P_k} \wedge AG_{E_n}\}$ is a collection of subject attribute groups $AG_{S_i}$, object attribute groups $AG_{O_j}$, action attribute groups $AG_{P_k}$, and environment attribute groups $AG_{E_n}$. It indicates that the terminal with attribute group $AG_{S_i}$ is requesting operation $AG_{P_k}$ for device attribute group $AG_{O_j}$ under environment attribute group $AG_{E_n}$.

*Definition 3.* Attributed-based access control policy (ACP) is a collection of subject attributes, object attributes, operational attributes, and environment attributes formed by means of merging or parsing. $ACP = \{AG_{S_i} \wedge or \vee AG_{O_j} \wedge or \vee AG_{P_k} \wedge or \vee AG_{E_n}\}$ represents the access control rules of the subject to the object and represents the set of attributes required to access the protected object resources.

*5.2. Terminal Access Control.* Terminal access control has three main parts. In this section, the implementation steps of each part will be explained in detail.

(i) **Registration phase:** Terminals and devices register attributes in the AMC. Administrators generate access policies based on attribute sets (see Algorithm 1).

(1) The terminals and devices submit attributes to the attribute management centre, which uses a RestFul service to execute device contracts with the submitted attribute set and store them on the blockchain.
(2) The administrator obtains the set of attributes submitted by the terminal or the device under the controller domain through the device contract. Then, the access control policy is formulated for the terminal access to the SDN network based on $AG_{S_i}, AG_{O_j}, AG_{P_k}, AG_{E_n}$.
(3) During the formulation of the access control policy, a unique *Token* is generated for the terminal. The *Token* is created by encrypting the relevant attributes in the access control policy. All of these operations are performed in the chain to ensure the *Token* is not tampered with. Finally, the administrator signs the access control policy to ensure its validity.
(4) Once the administrator has defined the ACP, the RestFul service of the operation policy contract is used to add, delete, modify, and verify the access control policy.

(ii) **Authentication phase:** The controller calls the access contract, verifies that the terminal *AAR* requests, and returns the response to the SDN controller (see Algorithm 2).

(1) First, the terminal encapsulates the attribute set in the IP packet's *Options* field. When the P4 forwarding device receives the message, it quickly filters out messages without the *Options* field based on the *IHL* field value. If the packet carries the *Options* field, it's forwarded to the connected switch, which encapsulates it into a Packet_in message and sends it to the controller. The controller then parses the Packet_in message, retrieves the *Options* value, and uses it to construct the AAR request for the access contract for the terminal.
(2) The blockchain verifies whether the AAR request constructed by the controller satisfies the ACP and, if so, generates a response status code and returns it to the controller. At the same time, the controller sends the flow table to OVS, allows the terminal traffic to be forwarded, and caches the *Token* generated by the terminal corresponding to the access control policy with the key as the terminal ID and value as the *Token* value in the cache database and the blockchain. The Token value in the cache database is consistent with the *Token* on the blockchain, and when the *Token* on the chain changes, it will be synchronized to the cache database in real time. If the AAR request does not satisfy the access control policy, the blockchain returns an error message to the controller.

(iii) **Access phase:** If the terminal's connection is interrupted for external reasons, two situations will occur when it is accessed again: first-time access and nonfirst-time access (see Algorithm 3).

(1) For first-time access, the terminal needs to do the same operations as in the authentication phase.
(2) For nonfirst-time access, the terminal adds the Token value obtained for the first time to the *Options* field of the IP packet and initiates an access request to the SDN network. First, when the packet carrying the *Options* field arrives at the P4 forwarding device, the P4 forwarding device parses the packet and filters out the packets without the *Options* field in the IP packet using the *IHL* field. Then, the packet is mirrored to the P4 control plane through the *to_cpu* action, and the P4 control plane parses the *Options* field value and queries the corresponding *Token* to the cache database through the RestFul service. If the corresponding *Token* is queried and is within the validity period, a function similar to the *Packet_in* message is implemented in the P4 control plane, and the flow table is distributed in the control plane to allow the terminal to join the SDN network. Otherwise, IP packets carrying *Options* are

**Require:** Attribute set
(1) Terminal and device submit attributes to the AMC
(2) AMC submit all attributes to the device contract and store attributes in the blockchain
(3) Administrators generate an access control policy Policy = gen($AG_{S_i}$, $AG_{O_j}$, $AG_{P_k}$, $AG_{E_n}$)
(4) compute *Token* = *MakeToken* (*Policy.AE.AllowedMAC*, *Policy.AO.DeviceId*, *Policy.AS.TerminalId*)

Algorithm 1: Attributes registration phase.

(1) Terminal sends packets with Identification to P4FD
(2) **if** Option field is none **then**
(3) discards the packets
(4) **end if**
(5) Forwarding the packets to the controller
(6) Controller sends a AAR request to Access Contract
(7) Access Contract verifies the ACP and return a statuscode
(8) **if** statuscode = 200 **then**
(9) Controller sends flow rules to OFS
(10) **else**
(11) **return** Authentication failed
(12) **end if**

Algorithm 2: Authentication phase.

(1) Terminal add *Token* to the *Options* field of the IP packet
(2) Initiate access requests
(3) **if** The terminal is nonfirst-time access **then**
(4) P4FD parses the IP header of packets
(5) **if** *IHL* = *0x05* **then**
(6) Discard the packets
(7) **end if**
(8) The packet is mirrored to the P4 control plane
(9) The P4 parses the *Options* and get the *Token*
(10) Query the *Token* from the cache database
(11) **if** The *Token* exists and has not expired **then**
(12) Distribute the flow table in the p4 control plane
(13) **else**
(14) Resend the IP packet with *Options* to the controller
(15) **end if**
(16) **else**
(17) Perform the same operation as in the authentication phase
(18) **end if**

Algorithm 3: Access phase.

resent to the controller, which realizes terminal access to the SDN network after the operation of the authentication stage.

*5.3. Smart Contracts of ABAC.* This section provides details on the structure and interface of the ABAC smart contract, which is implemented using smart contracts and can be accessed by the application through the RestFul service.

(1) **Policy Contract:** The responsibility of the PC includes generating, updating, finding, and deleting access control policies. The *Policy* struct is used for this purpose, which consists of AS, AO, AP, and AE substructs. AS, AO, and AE represent the attributes of the terminal, device, and environment, respectively, while *AP* represents access permissions. The main functions of the PC are as follows.

(1) AddPolicy(): This method primarily generates an access control policy using ⟨AS, AO, AP, AE⟩ as the input parameters. The algorithm first validates the legitimacy of the input parameters and then uses the *parsePolicy* method to parse and match the policy JSON string with the policy structure, ensuring the type and number of attributes are correct. Following this, the

*CheckPolicy* method is called to verify whether the access control policy set by the network administrator satisfies the policy requirements. If it meets the requirements, the *MakeToken* method is called to create a *Token* using $SHA256(AAS.ID, AO.ID)$, which is then assigned to the Token field in the Policy structure. The formulated policy is stored in the blockchain state database as a key-value pair, where the key is generated using $SHA256(AS.ID, AO.ID)$ and the value is policy. Finally, *Policy.ID* is returned.

(2) QueryPolicy(): First, the method verifies the legitimacy of the input parameters and then queries the policy details in the blockchain state database based on *Policy.ID*.

(3) DeletePolicy(): This method executes the DelState method to remove the access control policy corresponding to *Policy.ID* from the blockchain state database.

(4) UpdatePolicy(): This method will override the original access control policy.

(5) QueryToken(): This method queries the corresponding *Token* based on the input *Policy.ID*.

(ii) **Device Contract:** The DC is responsible for adding and finding attributes related to terminals or devices, and it performs the following main functions.

(1) AddAS(): This method saves the registered attributes of the terminal in the blockchain state database. Initially, it validates the input parameters' accuracy and uses the parserAS function to parse the terminal's registered attributes to the AS struct. If the attribute value complies with the defined data type, the terminal's ID is obtained and used as the key, and the attributes are saved as the value. Finally, the blockchain state database stores the key-value pair $\langle ID, Attributes \rangle$.

(2) AddAO(): Similar to AddAS(), this method receives the device's attributes under the management domain of a particular SDN controller and stores them in the blockchain state database.

(3) GetAS(): This method queries the attributes from the blockchain based on the terminal ID and returns the details of the terminal attributes.

(4) GetAO(): Similar to GetAS(), this method queries the device's attributes.

(iii) **Access Contract:** The main function of the AC is to verify whether the terminal has the right to access the SDN network.

(1) AuthACP(): This method verifies the correctness of the struct for the input AAR.

(2) CheckAccess(): This method validates the access privileges of the terminal by examining the AAR request received from the controller. Initially, it validates the legitimacy of the passed parameters and uses the AuthACP() method to verify the AAR struct. Subsequently, the *GetAttrs* method is called to retrieve AO.ID, AS.ID, and AS.MAC from the verified *AAR*. The *QueryPolicy()* method is then used to retrieve the access control policy. If the value of *Policy.AP* is 1, indicating that the terminal has access rights, and access is granted. Otherwise, it is denied. Using the four AE parameters (*CreatedTime*, *EndTime*, *AllowedIP*, and *AllowedMAC*), the AC checks the current access time's validity and the legality of MAC and IP. Finally, the method returns the outcome of the access verification to the controller.

## 6. Evaluation

To evaluate the feasibility and performance of our scheme, we realized a prototype of its proof-of-concept using Mininet [35] and Hyperledger Fabric [36]. Mininet is a network simulation tool that rapidly creates large-scale SDN prototype systems on ordinary computers with limited resources. Hyperledger Fabric is an open-source consortium blockchain platform widely used in various domains.

*6.1. Simulation Setup.* As illustrated in Figure 3, we simulate an SDN network using Mininet with Floodlight as the SDN controller. We modify the message parsing module in each controller to parse the Options field of Packet in messages and the data forwarding module to implement the flow table for postauthentication distribution. To enable blockchain functionality, we combine each controller with a Fabric node. Additionally, we leverage a pastry-based dynamic load balancing algorithm [37] to ensure load balancing among controllers. The experiments are conducted on an Ubuntu-20.04 system running on VMware ESXi 6.5 with an Intel(R) Xeon(R) Silver 4114 CPU @2.20 GHz and 16 GB of memory.

*6.2. Comparative Summary.* In the comparative summary, we focus on four key features: decentralization, fine-grained access control, dynamic access control, and a programmable data plane. It should be noted that among all the schemes compared in Table 1, only our scheme meets all these features. The detailed explanations of the comparative summary are presented below.

*6.2.1. Decentralization.* Decentralization requires that the entire solution not rely on a central server. For example, in SDN, a single controller may not be able to handle the service requests from a large number of terminals. With distributed edge controllers, service requests from terminals are dispersed to closer controllers, effectively avoiding the vulnerability of a single point of failure.

*6.2.2. Fine-Grained Access Control.* Fine-grained access control in SDN environments allows administrators to control who can access the network, what they can access, and how they can access it. This helps prevent unauthorized

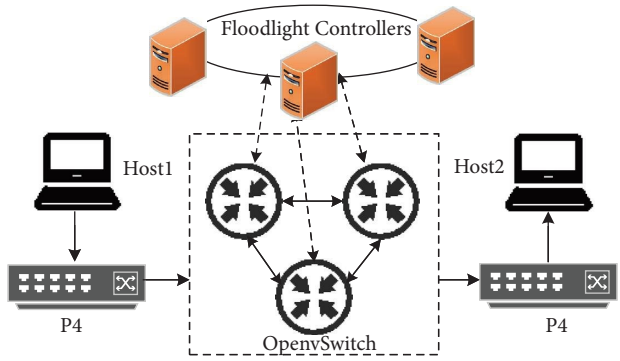



Figure 3: The proposed prototype topology.

Table 1: Comparative summary features.

| Schemes | Decentralization | Fine-grained access control | Dynamic access control | Programmable data plane |
|---|---|---|---|---|
| B-DAC [15] | √ | √ | × | × |
| FlowNAC [18] | × | √ | √ | × |
| SILedger [7] | √ | √ | √ | × |
| FGAC [38] | √ | √ | × | × |
| FACSC | √ | √ | √ | √ |

access to the network, which can help protect against cyber threats such as network intrusions, data breaches, and denial-of-service attacks.

*6.2.3. Dynamic Access Control.* Access policies can be updated and enforced in real time based on changes in the environment. This means that access control decisions can be made on the fly, which can help improve security and reduce risk. ABAC is a scalable access control model that can be easily applied to large, complex environments. This means that organizations can easily manage access control policies for a large number of users and resources.

*6.2.4. Programmable Data Plane.* Programmable data planes enable greater flexibility and control over how packets are processed and forwarded through the network. This can lead to improvements in network performance, security, and reliability, as well as enable the development of new network applications and services. With programmable data planes, network engineers can define how packets should be processed and forwarded through the network using a high-level programming language such as P4.

*6.3. Performance of ABAC Smart Contracts.* In this subsection, we conduct tests to fully assess the performance of the ABAC model. Specifically, we measure the average completion time of the three smart contracts in the ABAC model under varying concurrency levels of 10, 50, 100, 150, and 200.

*6.3.1. Policy Contracts.* Figure 4 shows the average completion time for add, delete, query, and update operations in the policy contract under varying concurrent requests. The figure indicates that the AddPolicy(), QueryPolicy(), DeletePolicy(), UpdatePolicy(), and QueryToken() functions have average response times of 139.4 ms, 36.6 ms, 100.2 ms, 197 ms, and 39 ms, respectively. We also conducted tests for a single operation of each function and found that the completion time for a single add or update operation was consistently between 80–140 ms, a single query operation was consistently between 25–50 ms, and a single delete operation was consistently between 55–90 ms. These results demonstrate that the performance of the policy contract can effectively meet the daily requirements of network administrators for policy add, delete, query, and update operations.

*6.3.2. Device Contracts.* The performance test results for the device contract are presented in Figure 5, which includes interfaces for adding and querying *AS* and *AO* attributes. As shown in the figure, the average completion times for AddAS(), AddAO(), GetAS(), and GetAO() are 129.6 ms, 141.4 ms, 36.1 ms, and 48 ms, respectively. Additionally, we conducted tests on individual add or query operations, with completion times for the add interface ranging from 70–125 ms and for the query interface ranging from 30–50 ms. These results indicate that the device contract's performance is sufficient for registering and querying device attributes.

*6.3.3. Access Contracts.* The performance test results of the access contract interface are presented in Figure 6. The average completion time for verifying terminal access rights is approximately 175.4 ms under different concurrent requests, while the completion time of the policy verification function interface remains stable at 82–150 ms during a single verification operation. This takes more time as it

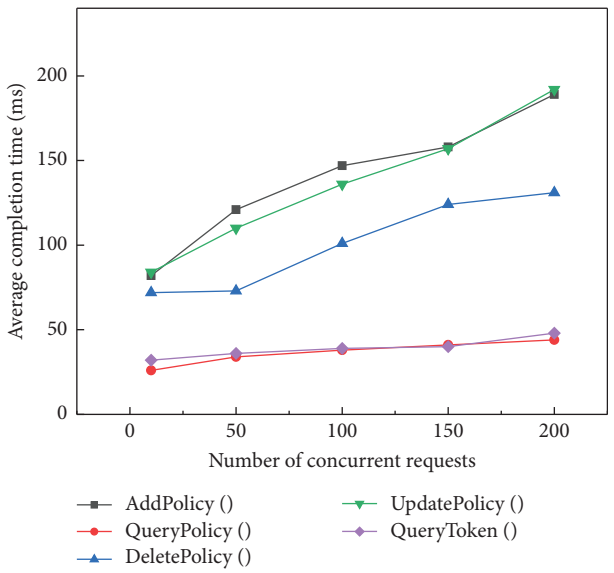



Figure 4: Average completion time of policy contract calls under concurrency.

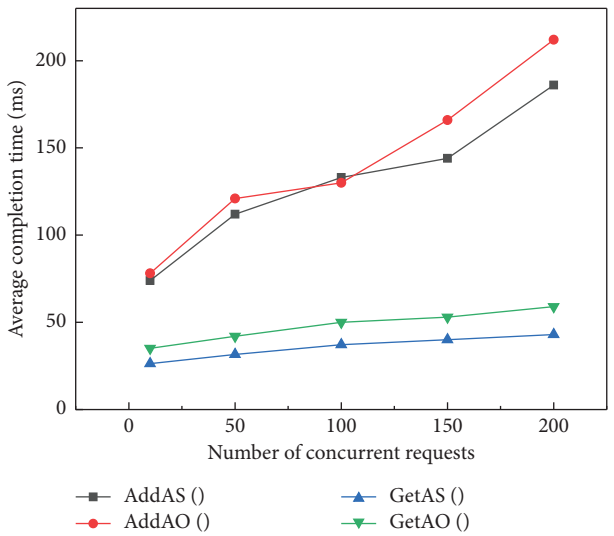



Figure 5: Average completion time of device contract calls under concurrency.

requires interchain code calls in the AC, such as calling the *QueryPolicy* function in the PC.

Based on the performance tests conducted on the policy contracts, device contracts, and access contracts, we can draw the following conclusions: (1) query operations have a minimal time overhead since they do not require consensus and do not need to be recorded on the blockchain. (2) Add and update operations have a significant time overhead because consensus is required among the blockchain nodes before data can be saved.

*6.4. Time Overhead for Terminal Access.* In our system, there will be two cases of terminal access to the network: first access and nonfirst access.

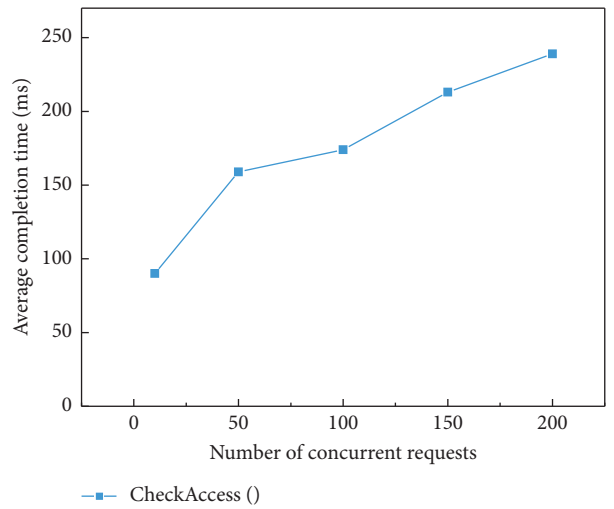



Figure 6: Average completion time of access contract calls under concurrency.

(1) First-time access: For the client-server experiment, we designated $Host_1$ as the client and $Host_2$ as the target server. We ran the client and server codes on their respective hosts. The client $Host_1$ encapsulated the IP packet with the *Options* field into a UDP packet and sent it to P4FD, which filtered the UDP packet and forwarded it to the OpenvSwitch switch. OvS then encapsulated the UDP packet into a *Packet_in* message and transmitted it to the Floodlight controller.

The floodlight controller parsed the message and constructed an *AAR* request. Upon receiving the *AAR* request, the blockchain called AC and returned the response status code to the controller. Finally, the controller sent the flow table to OvS according to the status code, thereby achieving the first access to the terminal.

(2) Nonfirst-time access: When the access is not the first time, $Host_1$ retrieves the previously obtained *Token* and inserts it into the *Verification_Token* field in *Options*, then initiates the *ping* operation. As shown in Figure 7, the terminal successfully passes the *Token* verification in the P4 control plane, and subsequently, the P4 control plane issues the flow table, enabling the successful execution of the *ping* command.

We compared the time overhead of first-time and nonfirst-time requests for terminal access to the SDN network. As shown in Figure 8, for first-time access, the average authentication completion time for different numbers of packets is approximately 197 ms. For nonfirst-time access, message parsing in the P4 control plane and verification of the *Token* are simulated, and the average completion time for verifying each packet authentication is approximately 35.6 ms for different numbers of packets. From the comparison results, it is evident that the authentication overhead for nonfirst-time access is much lower than that for first-time authentication. Therefore, the nonfirst-time terminal

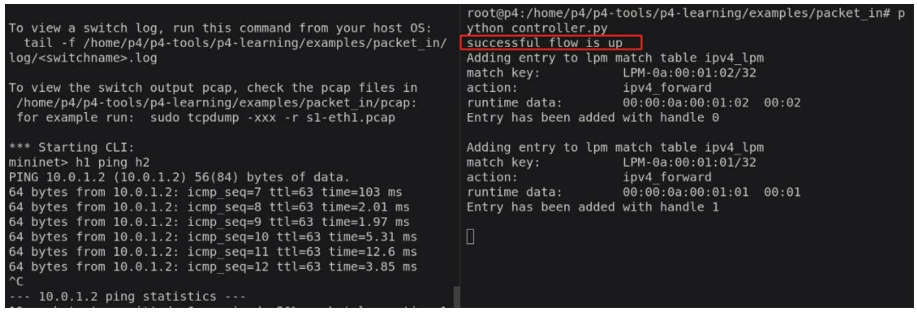



Figure 7: Flow table issued by P4 control plane.

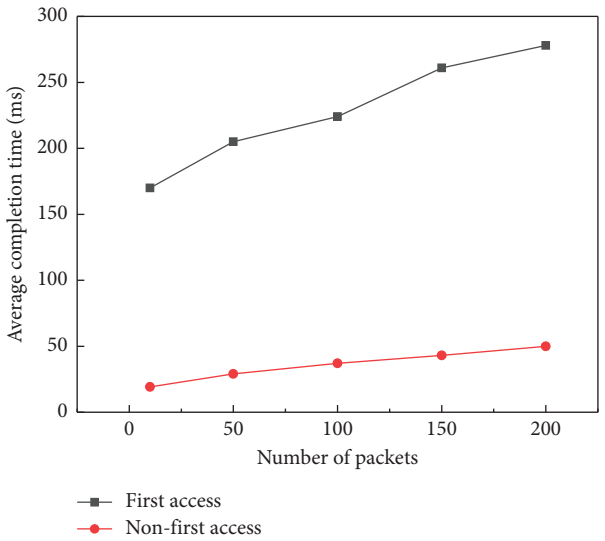



Figure 8: Comparison of time overhead for first-time and nonfirst-time access.

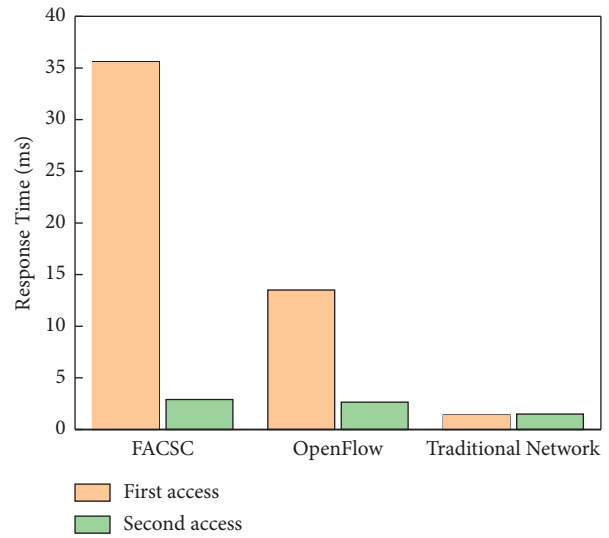



Figure 9: Comparison of forwarding latency between different schemes.

access method in this scheme can compensate for the time-consuming nature of first-time access.

*6.5. Data Forwarding Delay.* Considering that first-time access to the terminal requires permission verification from the blockchain, which consumes more time, we only compare the latency of nonfirst-time access for the following comparison. We compare the latency of performing two *ping* operations in the traditional network, the OpenFlow network, and FACSC.

In FACSC, when P4FD receives the first *ping* packet, the control plane of P4FD does not issue any flow rules, so it cannot forward the data. At this point, the P4 control plane calls the RestFul service to find the corresponding *Token* of the terminal from the cached database according to the terminal ID. If the *Token* is the same, the P4 control plane issues the flow table, and the traffic will be transmitted. Otherwise, the P4 control plane will refuse to issue the flow rule. The results of the latency evaluation for different schemes are shown in Figure 9.

Based on the comparison of time overhead for the first *ping* in the traditional and OpenFlow networks, it can be concluded that the Floodlight controller takes around 12.1 ms to process data forwarding. In our proposed solution, the time overhead for the first *ping* is 35.62 ms. This delay is higher because the terminal must retrieve and verify the *Token* from the cache database before accessing the network. However, the time overhead for the second *ping* in FACSC is similar to that of the traditional and OpenFlow networks since it only involves normal packet flow between terminals without complex authentication. Therefore, FACSC provides secure terminal access to the SDN network while meeting normal usage requirements for authentication delay.

## 7. Conclusion

Securing terminal access in SDN networks is crucial for ensuring network security. However, most SDN architectures lack effective access control methods, leaving the network vulnerable to malicious terminal attacks. To address this issue, we propose the Fine-Grained Access Control System for SDN (FACSC), which uses blockchain technology and the ABAC model to implement smart contracts that provide strong security and flexible control policies for terminal access. Additionally, we utilize the programmability characteristics of SDN networks and P4 forwarding devices to offer convenient, efficient, and secure

terminal access, further enhancing the network's security. Our experimental simulations demonstrate that FACSC enables secure, controllable, and traceable terminal access to SDN networks. In future work, we will focus on reducing the authentication time and cost for initial access and using P4 to directly transmit filtered packets to the controller. We also plan to deploy the ABAC model on multiple physical nodes in a real environment for performance testing.

## Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

## Conflicts of Interest

The authors declare that they have no conflicts of interest.

## Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant Nos. 62162018 and 61861013, in part by the Innovation Research Team Project of Guangxi Natural Science Foundation 2019GXNSFGA245004.

## References

[1] D. Chattaraj, S. Saha, B. Bera, and A. K. Das, "On the design of blockchain-based access control scheme for software defined networks," in *Proceedings of the IEEE INFOCOM 2020-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 237–242, IEEE, Toronto, ON, Canada, July 2020.

[2] O. I. Abdullaziz, L.-C. Wang, and Y.-J. Chen, "Hiauth: hidden authentication for protecting software defined networks," *IEEE Transactions on Network and Service Management*, vol. 16, no. 2, pp. 618–631, 2019.

[3] M. Bonola, G. Bianchi, G. Picierro, S. Pontarelli, and M. Monaci, "Streamon: a data-plane programming abstraction for software-defined stream monitoring," *IEEE Transactions on Dependable and Secure Computing*, vol. 14, no. 6, pp. 664–678, 2017.

[4] P. Krishnan, K. Jain, K. Achuthan, and R. Buyya, "Software-defined security-by-contract for blockchain-enabled mud-aware industrial iot edge networks," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 10, pp. 7068–7076, 2022.

[5] H. Zhang, W. Quan, H. C. Chao, and C. Qiao, "Smart identifier network: a collaborative architecture for the future internet," *IEEE network*, vol. 30, no. 3, pp. 46–51, 2016.

[6] V. C. Hu, D. R. Kuhn, D. F. Ferraiolo, and J. Voas, "Attribute-based access control," *Computer*, vol. 48, no. 2, pp. 85–88, 2015.

[7] W. Ren, Y. Sun, H. Luo, and M. Guizani, "Siledger: a blockchain and abe-based access control for applications in sdn-iot networks," *IEEE Transactions on Network and Service Management*, vol. 18, no. 4, pp. 4406–4419, 2021.

[8] N. Ye, Y. Zhu, R. C. Wang, R. Malekian, and Q. M. Lin, "An efficient authentication and access control scheme for perception layer of internet of things," *Applied Mathematics & Information Sciences*, vol. 8, no. 4, 2014.

[9] S. Bhatt, F. Patwa, and R. Sandhu, "Access control model for aws internet of things," in *International Conference on Network and System Security*Springer, Berlin, Germany, 2017.

[10] R. Zhang, G. Liu, S. Li, Y. Wei, and Q. Wang, "Absac: attribute-based access control model supporting anonymous access for smart cities," *Security and Communication Networks*, vol. 2021, Article ID 5531369, 11 pages, 2021.

[11] Y. Xu, W. Gao, Q. Zeng, G. Wang, J. Ren, and Y. Zhang, "A feasible fuzzy-extended attribute-based access control technique," *Security and Communication Networks*, vol. 2018, Article ID 6476315, 11 pages, 2018.

[12] Y. Zhang, S. Kasahara, Y. Shen, X. Jiang, and J. Wan, "Smart contract-based access control for the internet of things," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1594–1605, 2019.

[13] A. Yazdinejad, R. M. Parizi, A. Dehghantanha, Q. Zhang, and K.-K. R. Choo, "An energy-efficient sdn controller architecture for iot networks with blockchain-based security," *IEEE Transactions on Services Computing*, vol. 13, no. 4, pp. 625–638, 2020.

[14] A. Rahman, M. J. Islam, A. Montieri et al., "Smartblock-sdn: an optimized blockchain-sdn framework for resource management in iot," *IEEE Access*, vol. 9, p. 28361, 2021.

[15] P. T. Duy, H. D. Hoang, D. T. T. Hien, A. G. T. Nguyen, and V. H. Pham, "B-dac: a decentralized access control framework on northbound interface for securing sdn using blockchain," *Journal of Information Security and Applications*, vol. 64, Article ID 103080, 2022.

[16] N. Kammoun, R. Abassi, S. Guemara El Fatmi, and M. Mosbah, "A new sdn architecture based on trust management and access control for iot," in *Proceedings of the Workshops of the International Conference on Advanced Information Networking and Applications*, pp. 245–254, Springer, Sydney, Australia, April 2020.

[17] C. Awasthi, I. Sehgal, P. K. Pal, and P. K. Mishra, "Software-defined network (sdn) for cloud-based internet of things," in *Transforming Management with AI, Big-Data, and IoT*, pp. 185–213, Springer, Berlin, Germany, 2022.

[18] J. Matias, J. Garay, A. Mendiola, N. Toledo, and E. Jacob, "Flownac: flow-based network access control," in *Proceedings of the 2014 third European workshop on software defined networks*, pp. 79–84, IEEE, Budapest, Hungary, September 2014.

[19] K. Benzekki, A. El Fergougui, and A. El Belrhiti El Alaoui, "Devolving ieee 802 Devolving IEEE 802.1X authentication capability to data plane in software-defined networking (SDN) architecture: d," *Security and Communication Networks*, vol. 9, no. 17, pp. 4369–4377, 2016.

[20] T. Fathima and S. M. Vennila, "Emphasizing a productive and protective access control to improve authentication using 802.1 x with software-defined networks," in *Proceedings of the International Conference on Computing, Communication, Electrical and Biomedical Systems,*Springer, Berlin, Germany, 2022.

[21] D. M. Ferrazani Mattos and O. C. M. B. Duarte, "Authflow: authentication and access control mechanism for software defined networking," *Annals of Telecommunications*, vol. 71, no. 11-12, pp. 607–615, 2016.

[22] A. Hesham, F. Sardis, S. Wong, T. Mahmoodi, and M. Tatipamula, "A simplified network access control design and implementation for m2m communication using sdn," in *Proceedings of the 2017 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, pp. 1–5, IEEE, Seoul, South Korea, May 2017.

[23] S. T. Yakasai and C. G. Guy, "Flowidentity: software-defined network access control," in *Proceedings of the 2015 IEEE Conference on Network Function Virtualization and Software Defined Network (NFV-SDN)*, pp. 115–120, IEEE, San Francisco, CA, USA, November 2015.

[24] R. Bifulco and G. Rétvári, "A survey on the programmable data plane: abstractions, architectures, and open problems," in *Proceedings of the 2018 IEEE 19th International Conference on High Performance Switching and Routing (HPSR)*, pp. 1–7, IEEE, Bucharest, Romania, June 2018.

[25] P. Bosshart, D. Daly, G. Gibb et al., "P4: programming protocol-independent packet processors," *ACM SIGCOMM - Computer Communication Review*, vol. 44, no. 3, pp. 87–95, 2014.

[26] P. Bosshart, G. Gibb, H.-S. Kim et al., "Forwarding metamorphosis: fast programmable match-action processing in hardware for sdn," *ACM SIGCOMM - Computer Communication Review*, vol. 43, no. 4, pp. 99–110, 2013.

[27] S. Chole, A. Fingerhut, S. Ma et al., "drmt: disaggregated programmable switching," in *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, pp. 1–14, Beijing China, August 2017.

[28] N. Dukkipati, "Rate Control Protocol (RCP): congestion control to make flows complete quickly," *Cités*, vol. 12, no. 2, pp. 45–56, 2008.

[29] S. Kaur, K. Kumar, and N. Aggarwal, "A review on p4-programmable data planes: architecture, research efforts, and future directions," *Computer Communications*, vol. 170, pp. 109–129, 2021.

[30] S. Jiang, J. Cao, J. A. McCann et al., "Privacy-preserving and efficient multi-keyword search over encrypted data on blockchain," in *Proceedings of the 2019 IEEE International Conference on Blockchain (Blockchain)*, pp. 405–410, Atlanta, GA, USA, July 2019.

[31] M. Zhang, J. Cao, Y. Sahni, Q. Chen, S. Jiang, and L. Yang, "Blockchain-based collaborative edge intelligence for trustworthy and real-time video surveillance," *IEEE Transactions on Industrial Informatics*, vol. 19, no. 2, pp. 1623–1633, 2023.

[32] T. Wang, C. Zhao, Q. Yang, S. Zhang, and S. C. Liew, "Ethna: analyzing the underlying peer-to-peer network of ethereum blockchain," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 3, pp. 2131–2146, 2021.

[33] D. Huang, X. Ma, and S. Zhang, "Performance analysis of the raft consensus algorithm for private blockchains," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 50, no. 1, pp. 172–181, 2020.

[34] S. Nakamoto and A. Bitcoin, "A peer-to-peer electronic cash system," *Bitcoin*, vol. 4, p. 2, 2008.

[35] R. L. S. De Oliveira, C. M. Schweitzer, A. A. Shinoda, and L. R. Prete, "Using mininet for emulation and prototyping software-defined networks," in *Proceedings of the 2014 IEEE Colombian conference on communications and computing (COLCOM)*, pp. 1–6, IEEE, Bogota, Colombia, June 2014.

[36] E. Androulaki, A. Barger, V. Bortnikov et al., "Hyperledger fabric: a distributed operating system for permissioned blockchains," in *Proceedings of the Thirteenth EuroSys Conference*Springer, Berlin, Germany, 2018.

[37] B. Jiang, Q. He, X. Li, and H. Huang, "Qos control method based on sdn for mobile cloud service," in *Proceedings of the 2020 IEEE 13th International Conference on Cloud Computing (CLOUD)*, pp. 275–283, Honolulu, HI, USA, September 2020.

[38] Y. Zhu, X. Wu, and Z. Hu, "Fine grained access control based on smart contract for edge computing," *Electronics*, vol. 11, no. 1, p. 167, 2022.

Hindawi
Security and Communication Networks
Volume 2022, Article ID 5818879, 13 pages
https://doi.org/10.1155/2022/5818879

*Research Article*

# A Secure Certificateless Signature Scheme for Space-Based Internet of Things

**Tongwei Liu, Wei Peng, Kai Zhu, and Baokang Zhao**

*College of Computer, National University of Defense Technology, Changsha 410073, China*

Correspondence should be addressed to Baokang Zhao; bkzhao@nudt.edu.cn

Received 11 July 2022; Accepted 5 October 2022; Published 15 November 2022

Academic Editor: Hao Peng



Space-based Internet of things (S-IoT) can provide global services and connection capabilities. It has broad emerging application, including marine monitoring, forest monitoring, animal monitoring, disaster emergency response and other fields. However, owing to the openness of satellite communications, S-IoT is vulnerable to hijacking attacks, and malicious attackers can tamper with or forge transmitted messages. More seriously, due to limited S-IoT node resources, it is difficult to directly apply existing security solutions to terrestrial networks to the S-IoT. In this study, we propose CSP, a novel, secure, and efficient scheme based on certificateless signatures and bilinear pairings for S-IoT. CSP consists of six phases: system setup, partial private key settings, private key settings, public key settings, signing and verifying. In CSP, we especially design that part of the private key comes from the ground KGC and the other part is generated by the communication entity itself. We theoretically prove CSP is secure, and it is able to resist tampering or forgery attacks. Moreover, it can also ensure the authenticity, integrity, unforgeability, and non-repudiation of transmitted messages. We also conducted extensive experiments and compared CSP with the existing schemes. The experimental results demonstrate that CSP can significantly reduce the total scheme time consumption. Especially, it can reduce about 50–60% of the time in the signature verification phase.

## 1. Introduction

Space-based Internet of Things(S-IoT) has broad application prospects in the fields of disaster emergency, animal monitoring, air pollution monitoring, aerospace measurement and control, aviation and navigation, and mobile communications [1–6], as shown in Figure 1. At present, the IoT generally transmits information through terrestrial networks such as the Internet and mobile communication networks, which severely limits the application scope and makes it hard to achieve true interconnection of all things. Terrestrial networks mainly cover densely populated areas such as cities and towns. More than 70% of the Earth's area and more than 3 billion people are not covered by terrestrial networks. S-IoT can cover sparsely populated areas such as oceans, polar regions, and deserts [7, 8]. In addition, S-IoT can also be applied in situations where terrestrial infrastructure has been damaged, such as reconstruction in disaster areas. According to the forecast of McKinsey, an American consulting company, the output value of S-IoT will reach 560 billion to 850 billion dollars within the next 5 years. It is expected that the number of machine-to-machine (M2M) and IoT networks connected to S-IoT will reach 5.96 million by 2025 [9].

Authentication is one of the fundamental issues for security [10]. However, owing to the openness of satellite communications, identity authentication between nodes and message integrity authentication face significant challenges. First, in S-IoT, a single satellite usually needs to provide data transmission services for massive ground nodes. The particularity of the environment makes the messages transmitted by satellites vulnerable to security threats such as eavesdropping, tampering, and forgery. Therefore, the communication security between satellite nodes is indispensable. Second, since the communication bandwidth of a satellite is typically narrow and satellite storage resources are also limited, the authentication scheme must be efficient. Third, in S-IoT, scores of ground nodes are distributed in the

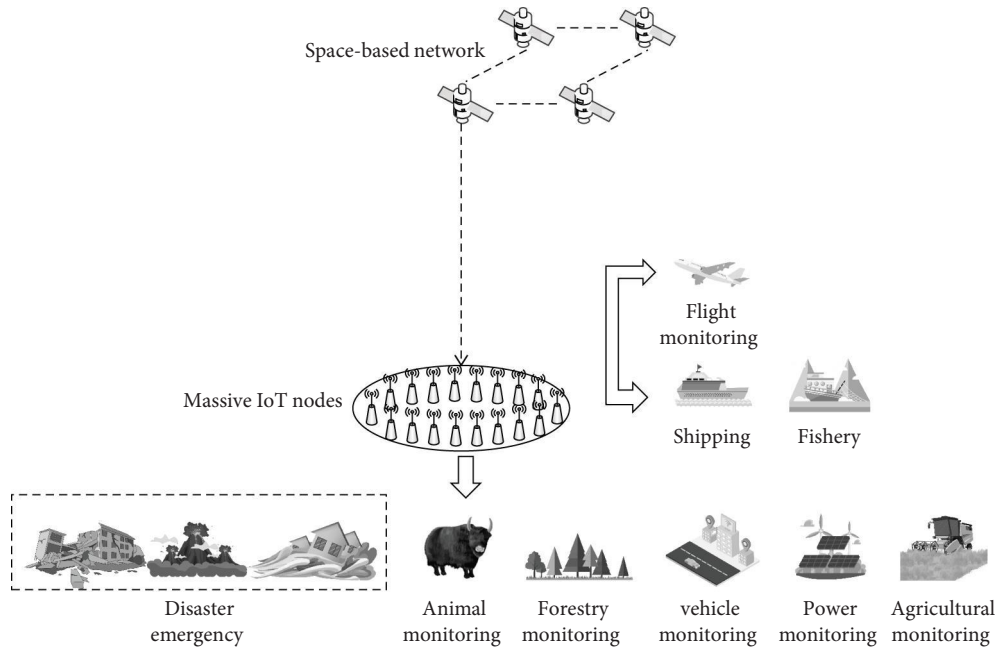



Figure 1: S-IoT application scenarios.

wild and may even be carried by animals. The computing, storage, and energy resources of ground nodes are severely limited due to the constraints of weight, volume, and deployment environment. These constraints require that the operations of ground nodes must be simple enough to operate [11]. Therefore, a secure and effective authentication scheme is urgently needed to ensure the development of S-IoT.

The main contribution of this study can be summarized as follows:

(1) We proposed CSP, a novel, secure, and efficient scheme based on certificateless signatures and bilinear pairings for S-IoT. To set the complete private key, a partial private key comes from the ground KGC and the other part of the private key is generated by the communication entity itself. In this way, CSP does not need a certificate authority and solves the key escrow issues.

(2) We have proved CSP, which has strong security and can effectively resist the attacks of external and internal adversaries. Moreover, CSP enables authenticity, integrity, unforgeability, and nonrepudiation of transmitted messages.

(3) We designed the CSP so that it only needs one hash function. Compared with the previous schemes, our scheme reduces the bilinear pairing operations in the verification phase. Performance evaluation shows that CSP can significantly reduce the total scheme time consumption. Especially, it can reduce about 50-60% of the time in the signature verification phase.

The organization of this study is summarized as follows. In Section 2, we review the related work. The S-IoT architecture, some principles of cryptography, and security model are described in Section 3. In Section 4, we elaborate on our proposed scheme for CSP. We present the security analysis of CSP in Section 5. In Section 6, we evaluate the performance. We finally conclude the study and discuss the future research direction in Section 7.

## 2. Related Work

The existing work on the certification of IoT can be roughly divided into three types. The first method is the authentication based on the public key infrastructure (PKI) mechanism, the second method is the authentication based on the identity-based cryptosystem (IBC), and the third method is the authentication based on the certificateless public key cryptography (CL-PKC).

In the traditional PKI, the certificate authority (CA) complexly manages the public key and identity information of all users and issues certificates to authenticated users. However, the generation, storage, distribution, verification, and revocation of certificates can be resource intensive.

Shamir [12] introduced IBC to try to solve the tedious problem of certificate management. As shown in [13], in addition to human users, computers and servers, smartphones, other mobile devices, and IoT devices also have their own identities. The IBC uses the user's identity information directly as the public key. The user's private key is generated by the key generation center (KGC) using the master key and the user's identity information. There have also been some studies [14, 15] on IBC in recent years. However, in an IBC system, the KGC must be highly trusted because it can encrypt and decrypt messages on behalf of any system user. This creates a key escrow problem inherent in the IBC system. Once KGC is invaded or breached, all users' private keys and identity information will be leaked. Then, the entire system will be paralyzed.

To solve these problems, Al-Riyam et al. [16] proposed CL-PKC. They designed a novel certificateless signature (CLS) scheme, in which KGC only provides part of the user's private key, and the other part is generated by the user, which can solve the key escrow problem. In addition, different from the traditional PKI system, users in the CL-PKC system do not need to be authenticated. Therefore, CLS do not need a certificate authority. CLS can be divided into two categories: one contains bilinear pairs and the other does not contain them.

Later, CLS technology has been greatly developed. In 1996, Bellare and Rogaway [17] proposed a random oracle model (ROM) to prove the security of the CLS scheme. This model can efficiently evaluate the scheme, but there are many loopholes. At the beginning of this century, Choudary Gorantla and Saxena [18] claimed to have proposed a high-efficiency CLS scheme. However, in 2006, Cao et al. [19] proved that their scheme is not secure against key substitution attacks. Zhang et al. [20] designed a new certificateless signature scheme using bilinear pairings and used the ROM to achieve strict security proofs under the assumption of computing the Diffie–Hellman Problem (DHP), but their algorithm's computational cost is significant.

In 2012, Tso et al. [21] proposed a certificateless short signature scheme. However, Du and Wen [22] point out that the scheme cannot resist the attack of the second type of strong adversary in the ROM model. There are also two short CLS schemes [23, 24], both of which have been proven to be secure. However, the scheme in [23] needs to perform two bilinear calculations during verification, while the scheme in [24] requires three bilinear calculations. Obviously, the computational cost is relatively high. The first CLS scheme without bilinear pairings is proposed in [25], but it is pointed out in [26] that this scheme is vulnerable to the second type of strong adversaries. After that, proposed CLS schemes without bilinear pairings were proposed in [27, 28], respectively. However, it is pointed out in [29, 30] that both schemes are vulnerable to the first type of super-adversaries. In 2015, Hassouna et al. [31] claimed to propose a strongly secure CLS scheme and proved its security under the assumption of two classes of strong adversary attacks. However, scholars soon proved that the scheme in [31] is insecure in the face of the attack of the first type of strong adversary. In addition, Wang et al. [32] proposed a novel, reliable, and efficient pairing-free certificateless scheme for the Industrial Internet of Things (IIoT) that utilizes the state-of-the-art blockchain technique and smart contracts. In 2018, Jia et al. [33] proposed an efficient and nonbilinear pairwise CLS scheme suitable for the IoT. In 2020, Du et al. [34] found that the scheme in [33] could not resist the attack of the first type of adversaries and proposed a new scheme. In the same year, a bilinear pair-free CLS scheme suitable for resource-constrained scenarios was proposed in [35]. However, in 2021, Xu et al. [36] found that the solution of [35] was vulnerable to signature forgery attacks and could not achieve its purpose.

Table 1 shows the main mechanisms and shortcomings of the existing works. From the above, we can see that the existing PKI and IBC mechanisms may not be suitable for S-IoT. The CLS scheme without bilinear pairings does not seem to be reassuring in terms of security. Especially, in recent years, the scheme proposed in a short period of time will be found to be unsafe. However, existing CLS schemes containing bilinear pairings have a large overhead. If these schemes are applied to the S-IoT, they will take up a large amount of resources. Therefore, we want to design a CLS scheme that balances security and computational overhead, which is suitable for the special environment of S-IoT.

## 3. Preliminaries

*3.1. S-IoT Architecture.* The typical S-IoT architecture [3] is shown in Figure 2. The S-IoT architecture consists of three parts: space segment, ground segment, and user terminal. The space segment consists of a constellation of satellites. The ground segment mainly includes the ground stations and the control stations. The user terminal refers to various terminals which are mainly used to send and receive signals. Information security is an important issue in S-IoT. There may be malicious nodes attacking the S-IoT system through eavesdropping, forgery, tampering, and other means.

*3.2. Elliptic Curve.* Elliptic curve cryptography (ECC) is a method of constructing cryptographic schemes from elliptic curves over finite fields. Elliptic curve cyphers can achieve the same strength with shorter keys than RSA; that is, elliptic curve cyphers have shorter key lengths but higher strength. In general, an elliptic curve cypher with a key length of 160 bits can achieve the same strength as RSA with a key length of 1024 bits.

Let $p$ be a large prime number of length $\lambda$, GF(p) represents a finite field, and an elliptic curve is a series of points satisfying the following equations:

$$G = \left\{(x, y):\ y^2 = x^3 + ax + b, 4a^3 + 27b^2 \bmod p \neq 0\right\} \cup O \quad a, b \in GF(p), \tag{1}$$

where $O$ represents the point at infinity.

*3.3. Bilinear Pairing.* Let the bilinear mapping be $e$: $G_1 \times G_2 \longrightarrow G_2$, where $G_1$ and $G_2$ are the additive cyclic group and the multiplicative cyclic group of order prime $p$, respectively. The generator of $G_1$ is $P$. The bilinear map satisfies the following properties:

(1) Bilinearity: $\forall Q, W, Z \in G_1$; there are $e(Q, W + Z) = e(Q, W) \cdot e(Q, Z)$ and $(Q + W, Z) = e\ (Q, Z) \cdot e(W, Z)$. Then, $\forall a, b \in Z_q^*$, $e(aQ, bW) = e(Q, W)^{ab}$

(2) Nondegenerate: $\exists P, Q \in G_1, e(P, Q) \neq 1$

(3) Computability: $\forall P, Q \in G_1$; there is a valid algorithm that can compute $e(P, Q)$

*3.4. Elliptic Curve Computational Diffie–Hellman Problem (ECDHP).* Given $aP, bP \in G_1, a, b, c \in Z_q^*$, where $P$ is the generator of $G_1$, it is hard to calculate $abP \in G_1$.

Table 1: Existing works.

| Method | Main mechanism | Services | Shortcomings |
|---|---|---|---|
| PKI | Public key certificate, certificate authority (CA), registrar authority (RA), etc. | Authentication, integrity, confidentiality, data fairness, nonrepudiation | Certificate management consumes a lot of resources |
| IBC | User ID is the user's public key (which can be their name, IP address, e-mail address, mobile phone number, etc.) | Authentication, integrity, confidentiality, and nonrepudiation | Key escrow issue |
| CL-PKC | The user's private key is divided into two parts; one is generated by KGC, and the other is generated by the user | Authentication, Integrity, Confidentiality, and nonrepudiation | Some existing schemes have low security or high overhead; it is not fully considered for S-IoT scenarios |

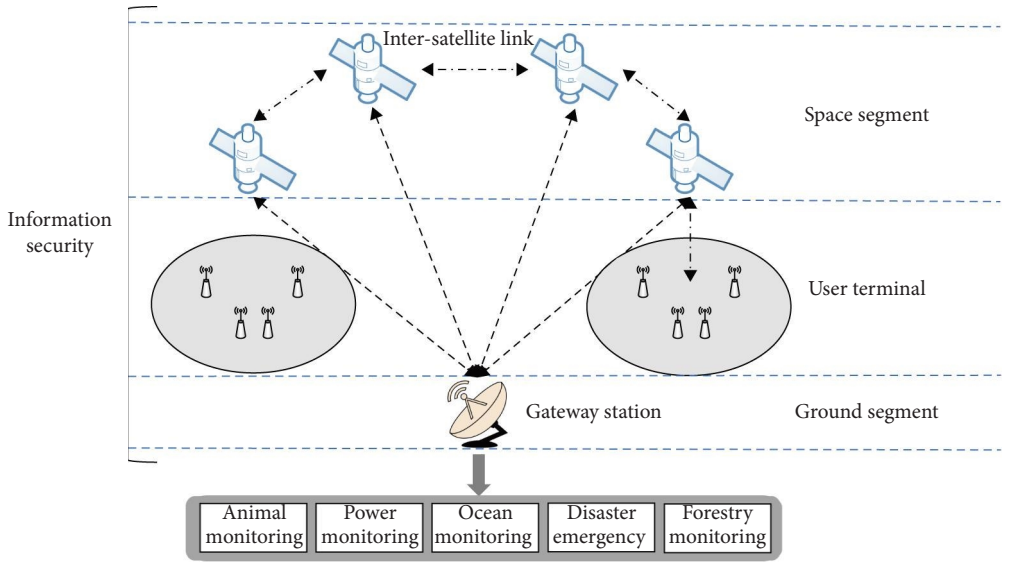



Figure 2: S-IoT architecture [3].

*3.5. Bilinear Diffie–Hellman Problem (BDHP).* Given $aP, bP, cP \in G_1, a, b, c \in Z_q^*$, where $P$ is the generator of $G_1$, it is hard to calculate $e(P, P)^{abc} \in G_2$.

*3.6. Certificateless Signature.* A certificateless signature scheme generally includes three entities: KGC, signer, and verifier. The general steps of the CLS are as follows:

(1) Setup: this step is performed by KGC. We input a security parameter $l$ and output the system master key $s$ and public parameter *params*. KGC securely keeps the system master key $s$ and makes *params* public.

(2) Set partial private key: this step is performed by KGC. We input the system master key $s$, the public parameter *params* and the signer's identity *ID*. Then, we output the partial private key $D$.

(3) Set private key: this step is performed by the signer. We input the public parameters *params*, the signer's identity *ID*, the partial private key $D$, and the signer's secret value. We output the private key *sk*.

(4) Set public key: this step is performed by the signer. We input the public parameter *params* and the signer's secret value and output the public key *pk*.

(5) Sign: this step is performed by the signer. We input public parameters *params*, message $m$, the signer's *ID*, the private key *sk*, and the public key *pk* and output the signature $\sigma$.

(6) Verify: this step is performed by the verifier. We input public parameters *params*, message $m$, signer *ID*, the public key *pk*, and the signature $\sigma$ and, finally, output
Verify $(m, \sigma, \text{params}, ID, pk) \longrightarrow$ true or false.

*3.7. Security Model of CLS.* As mentioned in [37, 38], traditional security controls and detection systems are often tailored against external threats, but insider attacks are also an ever-increasing threat to a system with dire consequences. For comprehensive consideration of safety, in a certificateless cryptosystem, external adversary and internal adversary $A_1$ and $A_2$ can be assumed [12]. At first, there are normal and strong adversaries in ROM. In [39], the adversary is expanded and classified into normal, strong, and super three levels. Through oracle queries, a normal adversary can only obtain the valid signature of the entity with the original public key. If the entity's public key is replaced, the normal adversary cannot obtain a valid signature. If the public key of a strong adversary has been replaced, the

adversary cannot obtain a valid signature until providing the associated secret value of the new public key. A super adversary can obtain the valid signature of the entity whose public key has been replaced without the new secret value. Here, we consider the case of super adversary attacks.

The first type of adversary $A_1$: these kinds of adversaries are also called external adversaries. The adversary can replace the public key of the target entity, but cannot obtain the master key of the KGC and partial private key of the entity.

The second type of adversary $A_2$: these kinds of adversaries are also called internal adversaries. The adversary knows the master key of KGC and the partial private key, but cannot replace the public key of the target entity.

#### 3.7.1. Type-I Model

Setup: challenger $C$ executes the algorithm to get the master secret key $s$ and public parameters *params*. Then, $C$ keeps $s$ as secret and makes the *parameters* public.

Queries: $A_1$ adaptively asks one of the following queries to $C$.

(1) Public key extraction query: $A_1$ obtains the public key $pk_i$ of $ID_i$

(2) Replace public key query: $A_1$ replaces $pk_i$ with $pk_i'$ which $A_1$ chooses

(3) Private key extraction query: $A_1$ obtains the private key $sk_i$ of $ID_i$

(4) Signature query: $A_1$ obtains a valid signature $\sigma$ for $(ID_i, m)$

Output: finally, $A_1$ outputs a valid forgery $\sigma^*$ for $(ID_i^*, m^*)$, where Verify$(m^*, \sigma^*, \text{params}, ID_i^*, pk_i') \longrightarrow$ true

#### 3.7.2. Type-II Model

Setup: challenger $C$ executes the algorithm to get the master secret key $s$ and public parameters *params*. Then, $C$ keeps $s$ as secret and makes *params* public.

Queries: $A_2$ adaptively asks one of the following queries to $C$.

(1) Public key extraction query: $A_2$ obtains the public key $pk_i$ of $ID_i$

(2) Partial private key extraction query: $A_2$ obtains the partial private key $D_i$ of $ID_i$

(3) Signature query: $A_2$ obtains a valid signature $\sigma$ for $(ID_i, m)$

Output: finally, $A_2$ outputs a valid forgery $\sigma^*$ for $(ID_i^*, m^*)$, where Verify$(m^*, \sigma^*, \text{params}, ID_i^*, pk_i) \longrightarrow$ true.

## 4. Proposed CSP Scheme

Our proposed scheme, CSP, is as follows:

Setup (KGC): the function of KGC is completed by the network control center (NCC) on the ground. KGC inputs security parameter $k$ and selects elliptic curve addition cyclic group $G_1$ and multiplication cyclic group $G_2$ with order $q$. The generator of $G_1$ is $P$. We set up bilinear mapping $e$: $G_1 \times G_1 \longrightarrow G_2$. KGC selects a random number $s \in Z_q^*$ as the system master key and calculates $P_{\text{pub}} = s \cdot P$ as the system public key. KGC selects a secure hash function $H_1$: $\{0,1\}^* \longrightarrow Z_q^*$. KGC securely saves the master key $s$ and makes the parameter params $= (q, G_1, G_2, e, P, P_{\text{pub}}, H_1)$ to the public.

Set partial private key (KGC): after KGC receives $ID_i$ from the entity(the satellite or the user on the ground) $S_i$, it calculates $Q_i = H_1(ID_i)$ and then calculates the partial private key $D_i = sQ_iP$ of $S_i$.

Set public/private key (entity): KGC sends $D_i$ to $S_i$. $S_i$ randomly selects the secret value $x_i, x_i' \in Z_q^*$, and then calculates $X_i = x_iP$, $Q_i = H_1(ID_i)$, and $Z_i = x_i'P, Y_i = x_iP_{\text{pub}}$. We take $(X_i, Y_i)$ as the public key $pk_i$ and take $(D_i, Z_i)$ as the private key $sk_i$.

Sign (entity): when $S_i$ needs to sign a message $m$, the specific description is as follows:

(1) $S_i$ randomly selects a large integer $a \in Z_q^*$
(2) $S_i$ calculates $M_i = x_i^2 H_1(m) \in Z_q^*$
(3) $S_i$ calculates $N_i = ax_i'Q_i \in Z_q^*$
(4) $S_i$ calculates $s_i = e(M_iD_i, Z_i)^a$
(5) $S_i$ sends $\sigma_i = (m, N_i, s_i)$ as a signature

Verify (entity): when another entity $S_j$ receives the message $m$ with the signature $\sigma_i$, it uses the public key $pk_i$ of $S_i$ to verify the signature. The specific description is as follows:

(1) $S_j$ calculates $M_j = H_1(m) \in Z_q^*$.
(2) $S_j$ calculates $e(M_jN_iX_i, Y_i)$.
(3) $S_j$ verifies that $s_i = s_j$. If the equation holds to prove that the signature is valid; otherwise, the verification fails, the message is discarded, and a reauthentication message $rm$ is returned.

Correctness analysis:

$$\begin{aligned} s_i &= e(M_iD_i, Z_i) \\ &= e(P,P)^{aM_isQ_ix_i'} \\ &= e(P,P)^{aH_1(m)sQ_ix_i'x_i^2} \\ &= e(H_1(m)ax_i'Q_ix_iP, sx_iP) \\ &= e(M_jN_iX_i, Y_i) \\ &= s_j. \end{aligned} \tag{2}$$

As shown in Figure 3, the steps of interaction between satellite nodes are as follows:

Step 1 (KGC): KGC generates system parameters params = $(q, G_1, G_2, e, P, P_{\text{pub}}, H_1)$ to the public. Then, KGC calculates partial private keys. KGC sends partial private keys to the corresponding satellites. In this scene, we take satellites $S_i$, $S_j$, and $S_k$ as an example.

Step 2 ($S_i$, $S_j$, $S_k$): $S_i$ randomly selects the secret value $x_i, x_i' \in Z_q^*$, and then calculates $X_i, Q_i, Z_i, Y_i$ and $X_i', Q_i', Z_i', Y_i'$. We take $(X_i, Y_i)$ as the public key $pk_i$. We take $(X_i', Y_i')$ as the public key $pk_i'$. We take $(D_i, Z_i)$ as the private key $sk_i$. We take $(D_i, Z_i')$ as the private key $sk_i'$.

Step 3 ($S_i$): when $S_i$ needs to sign a message $m$, $S_i$ generates $\sigma_i = (m, N_i, s_i)$ as the signature of the message $m$ to $S_j$ and generates $\sigma_i' = (m', N_i', s_i')$ as the signature of the message $m'$ to $S_k$.

Step 4 ($S_j$, $S_k$): when satellites $S_j$ and $S_k$ receive the message $m$ and $m'$ with the signature $\sigma_i$ and $\sigma_i'$, $S_j$ uses the public key $pk_i$ to verify the signature $\sigma_i$ and $S_k$ uses the public key $pk_i'$ to verify the signature $\sigma_i'$.

As shown in Figure 4, the steps of interaction between satellite and ground nodes are as follows:

Step 1 (KGC): KGC generates system parameters params = $(q, G_1, G_2, e, P, P_{\text{pub}}, H_1)$ for the public. Then KGC calculates partial private keys. KGC sends partial private keys to the corresponding satellite and ground nodes. In this scene, take the satellite $S_i$ and the ground node $S_j$ as an example.

Step 2 ($S_i, S_j$): $S_i$ randomly selects the secret value $x_i, x_i' \in Z_q^*$, and then calculates $X_i$, $Q_i$, $Z_i$, and $Y_i$. $S_j$ randomly selects the secret value $x_j, x_j' \in Z_q^*$, and then calculates $X_j$, $Q_j$, $Z_j$, and $Y_j$.We take $(X_j, Y_j)$ as the public key $pk_j$ and take $(D_j, Z_j)$ as the private key $sk_j$.

Step 3 ($S_i$): $S_i$ generates $\sigma_i = (m, N_i, s_i)$ as the signature.

Step 4 ($S_j$): when the ground node $S_j$ receives the message $m$ with the signature $\sigma_i$, it uses the public key $pk_i$ of $S_i$ to verify the signature $\sigma_i$.

## 5. Security Analysis

**Lemma 1.** *Under the attack of the first type of super adversary $A_1$, it is assumed that $A_1$ can adaptively perform $q_H$ for $H_1$ oracle queries, $q_d$ for partial private key extraction queries, $q_{sk}$ for private key extraction queries, $q_{pk}$ for public key extraction queries, and $q_s$ for signature queries; there is an algorithm C that can solve the ECDHP problem with the advantage of $\varepsilon' \geq \varepsilon(1/q_H)(1-(1/q_H))^{q_{sk}+q_{pk}}$.*

*Proof.* Let $A_1$ be a super adversary, and we assume that the challenge for C is to know that $Z_i = x_i'P$ (which can be obtained in the private key query below), and $aP, a \in Z_q^*$. $C$ calculates $ax_i'P$ after interacting with $A_1$. We play the game as follows:

Game 1: challenger $C$ inputs the security parameter $l$, runs the system establishment algorithm to generate the system master key $s$ and system parameter *params*, then sends the params to $A_1$, and saves $s$ in secret.

After going through all the queries, $A_1$ outputs a forged signature $(m^*.N^*, S^*)$; if the forgery meets the following requirements, the super adversary $A_1$ is considered to win.

(1) $A_1$ has never submitted $(ID^*, m^*)$ to the signature oracle

(2) $A_1$ never submitted $ID^*$ to partial private key oracles

(3) Verify $(m^*, \sigma^*, \text{params}, ID^*, pk_i') \longrightarrow$ true

$H_1$ oracle query: $C$ maintains a list $L_{H_1}$ consisting of triples $(ID_i, Q_i, M_i)$, and the list is initially empty. When $A_1$ asks $C$ for $H_1$ with identity $ID_i$, if $ID_i$ has been stored in $L_{H_1}$, then $C$ returns the corresponding value to $A_1$; otherwise, $C$ calculates $Q_i = nP, n \in Z_q^*$; let $M_i = r_i, r_i \in Z_q^*$; we insert the new tuple $(ID_i, Q_i, M_i)$ into the list $L_{H_1}$ and return to $A_1$.

Setup: $C$ runs the system algorithm, selects a generator $P$, and calculates $P_{\text{pub}} = sP$, where $s$ is the system master key that $C$ does not know; in this game, $C$ randomly selects an identity $ID^*$, generates system parameters params = $(P, P_{\text{pub}}, H_1)$, and send to $A_1$.

Public key extraction query: $C$ maintains a list $L_{pk}$ consisting of triples $(ID_i, x_i, pk_i)$, and the list is initially empty. When $A_1$ inputs $ID_i$ to ask, if $ID_i$ has been stored in $L_{pk}$, $C$ returns the corresponding value to $A_1$; otherwise, $C$ calculates $X_i = x_iP, Y = x_iP, x \in Z_q^*$ and returns the value to $pk_i$. Then, $C$ inserts $(ID_i, x_i, pk_i)$ into the list $L_{pk}$.

Public key query: when $A_1$ enters $(ID_i, pk_i)$ query, if the tuple $(ID_i, x_i, pk_i)$ corresponding to $ID_i$ exists in the list $L_{pk}$, $C$ sets $pk_i = pk_i'$ and sets $(ID_i, x_i, pk_i)$ and returns the list $L_{pk}$; otherwise, $C$ performs the public key generation step to generate $(ID_i, x_i, pk_i)$, then sets $pk_i = pk_i'$, and returns $(ID_i, x_i, pk_i')$ to the list $L_{pk}$.

Private key extraction query: $C$ maintains a list $L_{sk}$ consisting of four tuples $(ID_i, x_i', sk_i, Z_i)$, and the list is initially empty. When $A_1$ asks with $ID_i$, if $ID_i$ has been stored in $L_{sk}$, $C$ returns the corresponding value to $A_1$; otherwise, $C$ calculates $Z = x_i'P, x_i' \in Z_q^*$, and converts the new tuple $(ID_i, x_i', sk_i, Z_i)$ which is inserted into the list $L_{sk}$ and returned to $A_1$.

Signature query: when $C$ receives a $(ID_i, m_i)$ signature query, it performs the following steps:

(1) If $ID = ID^*$, $C$ aborts the query and outputs an error; otherwise, $C$ queries $(ID_i, x_i', sk_i, Z_i)$, $(ID_i, x_i, pk_i)$, $(ID_i, Q_i, M_i)$ from $L_{sk}, L_{pk}, L_H$.
(2) Calculate $N_i = ax_i'Q_i \in Z_q^*, a \in Z_q^*$
(3) Calculate $s_i = e(M_iD_i, Z_i)^a$, signs the message using $(N_i, s_i)$. $C$ sends $(N_i, s_i)$ to $A_1$.

$A_1$ aborts the query and outputs the signature $\sigma = (N^*, S^*)$ of the identity $ID_i^*$ on the message $m^*$, which satisfies the verification condition:

$$\begin{aligned} S^* &= e(V, V') \\ &= s_j \\ &= e\left(M_j' N_i X_i, Y_i\right) \\ &= e\left(M_j' x_i P, x_i P_{\text{pub}}\right)^{a x_i' Q_i} \\ &= e\left(M_j' x_i P, x_i s P\right)^{a x_i' Q_i} \\ &= e\left(M_j' x_i^2 s Q_i P, a x_i' P\right). \end{aligned} \tag{3}$$

It can be known that $V' = ax_i'P$, which solves the ECDHP problem. Set events $E_1$, $E_2$ and $E_3$ are as follows:

$E_1$: $A_1$ goes through a series of queries and $C$ does not abort.

$E_2$: $A_1$ successfully forges a valid signature.

$E_3$: there is $ID = ID^*$ in forged signature.

We set $\Pr[E_2 \mid E_1] \geq \varepsilon$, then obviously we have:

$$\begin{aligned} \Pr[E_1] &\geq \left(1 - \frac{1}{q_H}\right)^{q_{sk}+q_{pk}}, \\ \Pr[E_2 \mid E_1] &\geq \varepsilon, \\ \Pr[E_3 \mid E_1 E_2] &\geq \frac{1}{q_H}. \end{aligned} \tag{4}$$

It can be calculated that $C$ can solve the ECDHP problem with a nonnegligible probability: $\varepsilon' \geq \varepsilon 1/q_H(1 - (1/q_H))^{q_{sk}+q_{pk}}$. This proof is unforgeable against adaptive selective message attacking the signature. Therefore, it is proved that the scheme can guarantee the authenticity and integrity of the message under the attack of the first type of super adversary $A_1$. □

**Lemma 2.** *Under the attack of the second type of super adversary $A_2$, it is assumed that $A_2$ can adaptively perform $q_H$ for $H_1$ oracle queries, $q_d$ for partial private key extraction queries, $q_{sk}$ for private key extraction queries, $q_{pk}$ for public key extraction queries, and $q_s$ for signature extraction queries; there is an algorithm C that can solve the BDHP problem with the advantage of $\varepsilon' \geq \varepsilon 1/q_H(1 - (1/q_H))^{q_d+q_{pk}}$.*

*Proof.* Let $A_2$ be a super adversary. We assume that the challenge for $C$ is that given the master key $s$, $D_i = sQ_iP$ (which can be obtained in the partial private key query below), $b = sQ_i, aP$ and $cP, a, c \in Z_q^*$ an calculate $S^* = e(P,P)^{\text{abc}}$ after interacting with $A_2$.

Game 2: challenger $C$ inputs the security parameter $l$, runs the system establishment algorithm to generate the system master key $s$ and system parameter *params*, then sends the params to $A_2$, and saves $s$ in secret.

After going through all the queries, $A_2$ outputs a forged signature $(m^*.N^*, S^*)$; if the forgery meets the following requirements, the super adversary $A_2$ is considered to win.

(1) $A_2$ has never submitted $(ID^*, m^*)$ to the signature oracle

(2) $A_2$ never submitted $ID^*$ to private key oracles

(3) $\text{Verify}(m^*, \sigma^*, \text{params}, ID^*, pk_i) \longrightarrow \text{true}$

$H_1$ oracle query: C maintains a list $L_{H_1}$ consisting of triples $(ID_i, Q_i, M_i)$, and the list is initially empty. When $A_2$ asks C for $H_1$ with identity $ID_i$, if $ID_i$ has been stored in $L_{H_1}$, then C returns the corresponding value to $A_2$; otherwise, C calculates $Q_i = nP, n \in Z_q^*$; let $M_i = r_i, r_i \in Z_q^*$, and we insert the new tuple $(ID_i, Q_i, M_i)$ into the list $L_{H_1}$, and return to $A_2$.

Setup: $C$ runs the system algorithm, selects a generator $P$, and calculates $P_{\text{pub}} = sP$, where $s$ is the system master key that $C$ does not know; in this game, $C$ randomly selects an identity $ID^*$, generates system parameters $\text{params} = (P, P_{\text{pub}}, H_1)$, and sends to $A_2$.

Public key extraction query: $C$ maintains a list $L_{pk}$ consisting of triples $(ID_i, x_i, pk_i)$, and the list is initially empty. When $A_2$ inputs $ID_i$ to ask, if $ID_i$ has been stored in $L_{pk}$, $C$ returns the corresponding value to $A_2$; otherwise, $C$ calculates $X_i = x_iP, Y = x_iP, x \in Z_q^*$ and returns the value to $pk_i$. Then, $C$ inserts $(ID_i, x_i, pk_i)$ into the list $L_{pk}$.

Partial private key extraction query: $C$ maintains a list $L_D$, consisting of triples $(ID_i, Q_i, D_i)$, and the list is initially empty. When $A_2$ asks $C$ with identity $ID_i$, if $ID = ID^*$, $C$ aborts and outputs an error; otherwise, if $ID_i$ has been stored in $L_D$, $C$ returns the corresponding value to $A_2$; if $ID_i$ is not stored in $L_D$, then $C$ extracts the tuple $(ID_i, Q_i, M_i)$ from the list $L_H$, calculates $D_i = sQ_iP$, and returns it to $A_2$.

Signature query: when $C$ receives a $(ID_i, m_i)$ signature query, it performs the following steps:

(1) If $ID = ID^*$, $C$ aborts the query and outputs an error; otherwise, $C$ queries $(ID_i, Q_i, M_i)$, $(ID_i, x_i, pk_i)$, $(ID_i, Q_i, M_i)$ from $L_D, L_{pk}, L_H$

(2) We calculate $N_i = ax_i'Q_i \in Z_q^*, a \in Z_q^*$

(3) We calculate $s_i = e(M_iD_i, Z_i)^a$ and sign the message using $(N_i, s_i)$. $C$ sends $(N_i, s_i)$ to $A_2$

$A_2$ aborts the query and outputs the signature $\sigma = (N^*, S^*)$ of the identity $ID_i^*$ on the message $m^*$, which satisfies the verification condition:

$$\begin{aligned} S^* &= e(V, V') \\ &= s_i \\ &= e\left(M_i D_i, Z_i\right) \\ &= e(P,P)^{a N_i s Q_i x_i / x_i^2}. \end{aligned} \tag{5}$$

The premise that the signature satisfies the verification conditions is the parameter $c = N_i x_i' x_i^2$ which is known to $A_2$. Thus, $A_2$ can calculate $S^* = e(P,P)^{\text{abc}}$ which solves the BDHP problem. We set events $E_1$, $E_2$, and $E_3$ are as follows:

$E_1$: $A_2$ goes through a series of queries and $C$ does not abort.

$E_2$: $A_2$ successfully forges a valid signature.

$E_3$: there is $ID = ID^*$ in forged signature.

We set $\Pr[E_2|E_1] \geq \varepsilon$; then, obviously we have

$$\begin{aligned} \Pr[E_1] &\geq \left(1 - \frac{1}{q_H}\right)^{q_d + q_{pk}}, \\ \Pr[E_2|E_1] &\geq \varepsilon, \\ \Pr[E_3|E_1E_2] &\geq \frac{1}{q_H}. \end{aligned} \tag{6}$$

It can be calculated that $C$ can solve the BDHP problem with a nonnegligible probability: $\varepsilon' \geq \varepsilon 1/q_H(1 - (1/q_H))^{q_d + q_{pk}}$. This proof is unforgeable against adaptive selective messaging attacks on the signature. Therefore, it is proved that the scheme can guarantee the authenticity and integrity of the message under the attack of the second type of super adversary $A_2$.

The above is a formal analysis in the ROM, which can ensure the strong security of CSP. The informal analysis is as follows.

Authenticity: CSP can realize the authentication of the message source and the authentication of the communication entity. This is determined by adding the identity information of the communication entity in the signature.

Integrity: CSP can guarantee that data have not been tampered with or damaged. This is determined by adding a hash function to the message in the signature. Once the message is changed, the corresponding hash function will change, which will result in authentication failure.

Unforgeability: CSP means that nobody except the communication entity itself can forge the signature. This is determined by the private key generation method of CSP. Only the ground KGC knows the secret value that generates the partial private key and only the communication entity itself knows the complete private key.

Nonrepudiation: CSP requires that neither the sender nor the receiver can deny the transmission. This is determined by adding the identity information of the sender in the signature. Besides, the receiver must reply with a message indicating whether the verification was successful. □

## 6. Performance Evaluation

In this section, we test the performance of CSP. We compare CSP with existing representative CLS schemes. To ensure a benchmark for comparison, CSP uses widely accepted parameters. The program runs on a virtual machine with an Intel(R) Core(TM) i7-9750H-CPU@2.60 GHz and 16 GB of RAM, using the Ubuntu18.04LT operating system. Using the Type-A type in the PBC library, its security level is comparable to that of 1024 bit RSA. The Type-A of this library is constructed on the elliptic curve $y^2 = x^3 + x$ in the finite field $GF(p)$, where $p$ is a large prime number of 160 bits. Assuming the message length is 128 bits.

Table 2 shows the time required for various operations in the simulation environment of this study. We can see that the bilinear pairing operation takes the longest, followed by the hash operation and the point multiplication operation. Since the scalar addition and multiplication operations take negligible time compared to other operations, we ignore the overhead of these two types of operations in the comparison. The value in the table is the average time of each operation 100 times.

Table 3 shows the efficiency comparison of CLS schemes for satellite-to-satellite nodes. In S-IoT, we first consider the case of authentication between satellite nodes. Since each satellite node provides services for massive ground nodes, we believe that the authentication of satellite nodes requires high security. Therefore, we consider that when a satellite wants to authenticate with other satellites, it needs to use different public and private keys. It mains that the satellite node $S_i$ uses $(sk_i, pk_i)$ for authentication with the satellite node $S_j$, while $S_i$ uses $(sk_i', pk_i')$ for authentication with the satellite node $S_k (j \neq k)$. In S-IoT, the roles of users and signers who generate public and private keys can only be assumed by the satellite itself, which cannot be delegated to a third party. Therefore, when calculating the overhead of the signing in this case, the overhead of generating public and private keys is included. Due to the difference in algorithms between CLS schemes with bilinear pairings and those without, the CLS schemes without bilinear pairings often include the point addition operation and the modular inverse operation. However, the time of these two operations is relatively short. For a more intuitive comparison, these two operations are ignored here.

The values of security against $A_1$ and $A_2$ are determined according to which level of adversary attack (normal, strong, and super) the corresponding scheme can resist in ROM. Among these schemes, it is mentioned in [36] that the scheme in [35] was vulnerable to signature forgery attacks and was insecure against $A_1$, and the scheme in [31] has recently been shown to be insecure in the face of the attack of $A_1$.

It can be seen from Figure 5 that, in the simulation environment of this study, except for the unsafe schemes in the table, the scheme in [34] has the shortest time of 8.727 ms when signing. CSP needs 10.638 ms for signing, which is in the middle level of these schemes. However, in the verification and total time comparison, the overhead of CSP is the smallest, which are 4.681 ms and 15.319 ms. Compared with the scheme in [34], CSP improves the efficiency by about 50% and 15%, respectively. In the CLS schemes with bilinear pairs, CSP has the smallest overhead in both the signature and verification phases, which improves efficiency by about 9% and 60%, respectively. Besides, CSP can prove a higher level of security. The reason is that the scheme in [31] does not fully use the user's public key when verifying the signature. It only uses part of the public key, which leads to the additional verification if the public key is authentic and valid. The authenticity and validity of the public key will increase the overhead of the bilinear pairing operations.

Table 4 shows the efficiency comparison of CLS schemes for satellite-to-ground nodes. When the satellite node is authenticated with the ground node in S-IoT, it is troublesome and resource-consuming to frequently update the

Table 2: Symbol description and operation time.

| Symbol | Operation | Time (ms) |
|---|---|---|
| $T_r$ | Generating a random number | 0.120 |
| $T_h$ | A hash operation | 1.943 |
| $T_{\text{pm}}$ | A point multiplication operation | 0.886 |
| $T_{\text{add}}$ | A scalar addition operation | 0.001 |
| $T_m$ | A scalar multiplication operation | 0.001 |
| $T_p$ | A bilinear pairing operation | 1.962 |

Table 3: Comparison of CLS schemes for satellite-to-satellite nodes.

| Scheme | Sign | Verify | Type | Security against $A_1$ | Security against $A_2$ | Signature size |
|---|---|---|---|---|---|---|
| Du's et al. [34] | $3T_{\text{pm}} + 3T_h + 2T_r$ | $4T_{\text{pm}} + 3T_h$ | Without bilinear pairs | Super | Super | $\|G_1\| + \|Z_q^*\|$ |
| Thumbur's et al. [35] | $3T_{pm} + 2T_h + 2T_r$ | $3T_{pm} + 2T_h$ | Without bilinear pairs | Insecure | Super | $\|G_1\| + \|Z_q^*\|$ |
| Xu's et al. [36] | $3T_{pm} + 3T_h + 3T_r$ | $4T_{pm} + 3T_h$ | Without bilinear pairs | Super | Super | $\|G_1\| + \|Z_q^*\|$ |
| Hassouna's et al. [31] | $4T_{pm} + 3T_h + 1T_p + 3T_r$ | $4T_p + 1T_h$ | With bilinear pairs | Insecure | Strong | $\|G_1\| + \|G_2\|$ |
| CSP | $5T_{pm} + 2T_h + 1T_p + 3T_r$ | $1T_p + 1T_{pm} + 1T_h$ | With bilinear pairs | Super | Super | $\|G_1\| + \|Z_q^*\|$ |

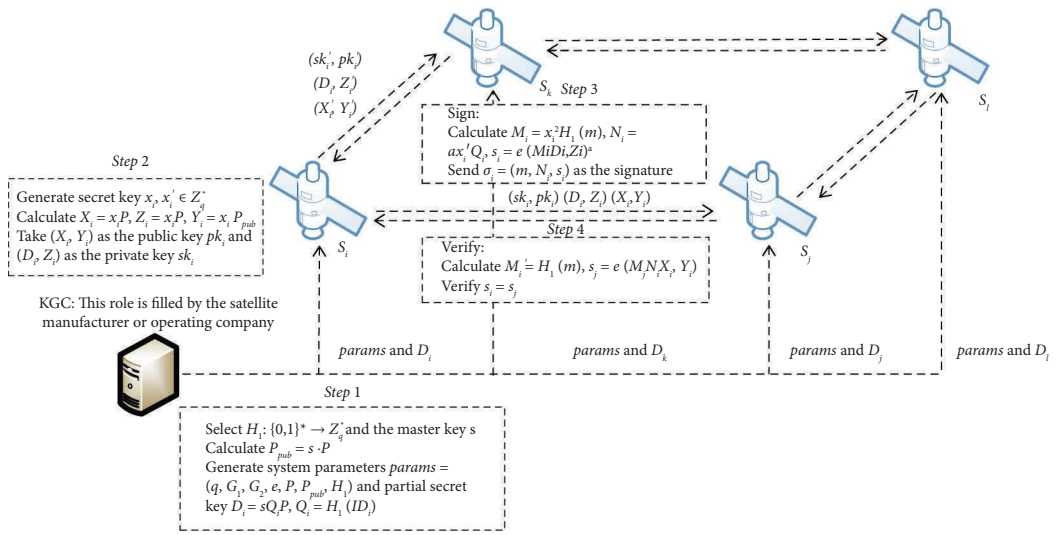



Figure 3: Interaction between satellite nodes.

Table 4: Comparison of CLS schemes for satellite-to-ground nodes.

| Scheme | Sign | Verify | Type | Security against $A_1$ | Security against $A_2$ | Signature size |
|---|---|---|---|---|---|---|
| Du's et al. [34] | $1T_{pm}$ | $4T_{pm} + 3T_h$ | Without bilinear pairings | Super | Super | $\|G_1\| + \|Z_q^*\|$ |
| Thumbur's et al. [35] | $1T_{pm}$ | $3T_{pm} + 2T_h$ | Without bilinear pairings | Insecure | Super | $\|G_1\| + \|Z_q^*\|$ |
| Xu's et al. [36] | $1T_{pm} + 2T_h$ | $4T_{pm} + 3T_h$ | Without bilinear pairings | Super | Super | $\|G_1\| + \|Z_q^*\|$ |
| Hassouna's et al. [31] | $1T_{pm} + 1T_h + 1T_p$ | $4T_p + 1T_h$ | With bilinear pairings | Insecure | Strong | $\|G_1\| + \|G_2\|$ |
| CSP | $1T_{pm} + 1T_h + 1T_p$ | $1T_p + 1T_{pm} + 1T_h$ | With bilinear pairings | Super | Super | $\|G_1\| + \|Z_q^*\|$ |

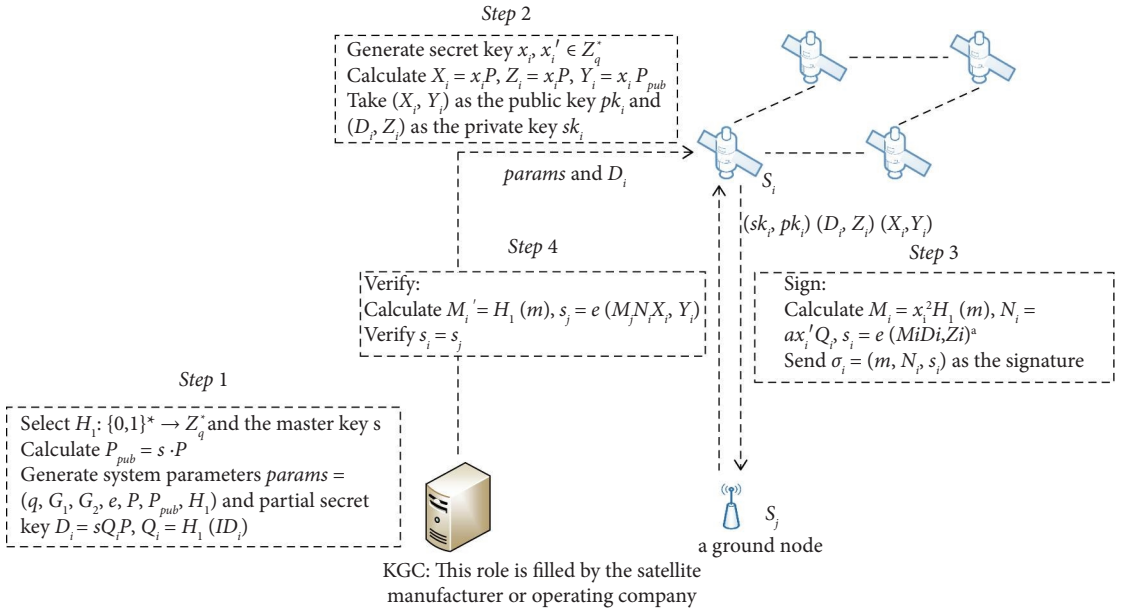



Figure 4: Interaction between the satellite and ground nodes.

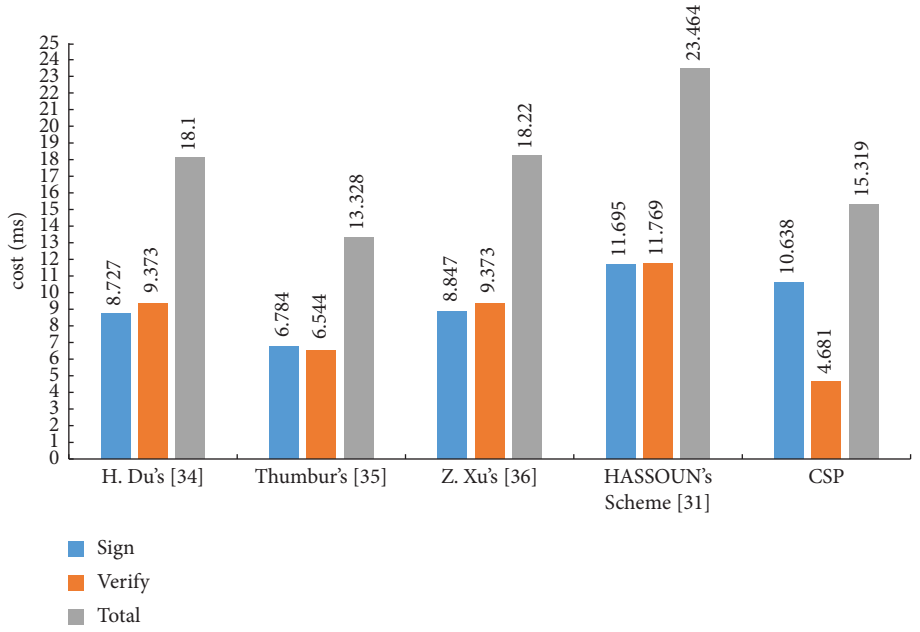



Figure 5: Comparison of CLS schemes for interaction between satellite nodes.

public and private keys. Therefore, when the satellite $S_i$ is authenticated with the ground node, it only uses $(sk_i, pk_i)$.

We can see from Figure 6 that the scheme in [34] has the shortest times of 0.886 ms in the signing phase, except for the unsafe scheme. CSP needs 4.791 ms, which is similar to the schemes in [31, 36]. In the verification phase and total time comparison, CSP still has the best performance, which improves efficiency by about 50% and 7%, respectively.

Finally, we discuss the issue of CSP complexity. For the scenario in Figure 5, to complete the authentication between satellite nodes, three authentication messages need to be transmitted. The communication cost of each authentication message is 480 bits. The signature cost is 10.638 ms, and the verification cost is 4.681 ms. For the scenario in Figure 6, three authentication messages between the satellite and the ground node need to be transmitted. The communication cost of each authentication message is 480 bits. The signature cost is 4.971 ms. The verification cost is 4.681 ms. The CSP performance is shown in Table 5.

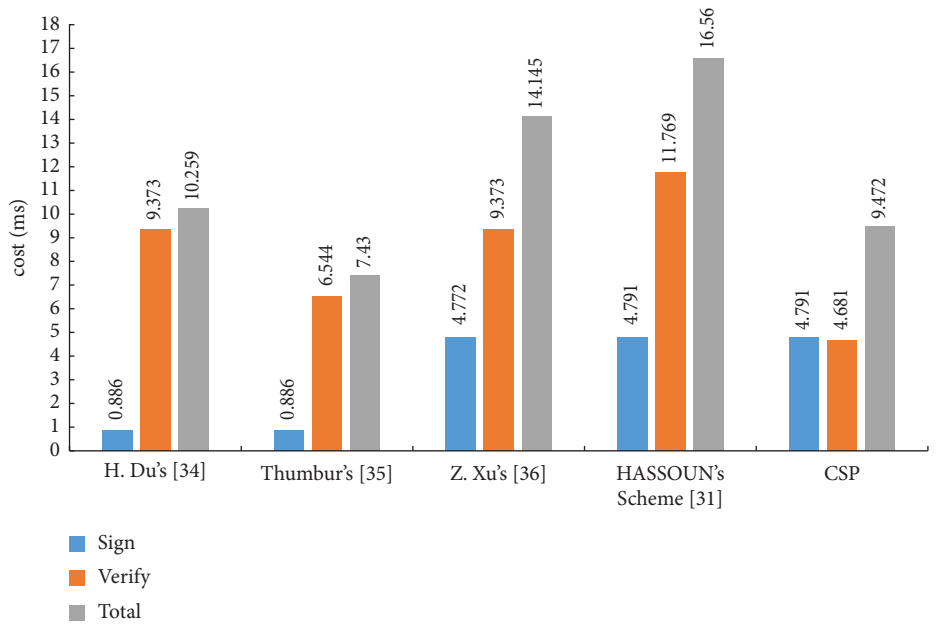



Figure 6: Comparison of CLS schemes for interaction between satellite-to-ground nodes.

Table 5: CSP performance.

| Scenario | The number of transmitted authentication messages | Computational operations | Total cost (ms) | Communication cost (bits) |
|---|---|---|---|---|
| Satellite-to-satellite nodes | 3 | $6T_{pm} + 3T_h + 2T_p + 3T_r$ | 15.319 | 480 |
| Satellite-to-ground nodes | 3 | $2T_{pm} + 2T_h + 2T_p$ | 9.472 | 480 |

## 7. Conclusions and Future Work

Due to the special environment and restricted resources, the security solution used by the terrestrial network for communication cannot be directly applied to the S-IoT. This study proposes a strong, secure certificateless signature scheme with bilinear pairings named CSP, which is suitable for S-IoT. Before the satellite is connected to the space-based network, the manufacturer or the company that undertakes the work of KGC inputs the partial private key and public parameters to the satellite. The satellite uses its own identity information and secret value to calculate the public key, the private key, and the signature. CSP can ensure the authenticity, integrity, unforgeability, and nonrepudiation of transmitted messages. The CSP solves the problems of complicated certificate management in the traditional PKI system key escrow in the IBC system. In the future, the secure access of satellites facing a large number of ground nodes is an important issue. In recent years, an ultra-superfast authentication protocol for electric vehicles, charging by utilizing the characteristics of extended chaotic maps, has been proposed in [40] which can resist man-in-the-middle attacks, replay attacks, and impersonation attacks. This work provides us with new ideas on how to perform rapid authentication when considering a large number of nodes that want to access satellites. Besides, analysis shows that compared with similar schemes, CSP can achieve a higher security level and lower overhead. However, CSP still needs a relatively long time in the signature phase, which is the research direction of future work.

## Data Availability

The data that support the findings of this study can be obtained from the corresponding author upon reasonable request.

## Conflicts of Interest

The authors declare that they have no conflicts of interest.

## Acknowledgments

This research was partially supported by the National Natural Science Foundation of China (Grant no. 61972412).

## References

[1] D. Yang, Y. Zhou, W. Huang, and X. Zhou, "5G mobile communication convergence protocol architecture and key technologies in satellite internet of things system," *Alexandria Engineering Journal*, vol. 60, pp. 465–476, 2021.

[2] M. Bacco, L. Boero, P. Cassará et al., "IoT applications and services in space information networks," *IEEE Wireless Communications*, vol. 26, pp. 31–37, 2019.

[3] Z. Qu, G. Zhang, H. Cao, and J. Xie, "LEO satellite constellation for internet of things," *IEEE Access*, vol. 5, pp. 18391–18401, 2017.

[4] M. De Sanctis, E. Cianca, G. Araniti, I. Bisio, and R. Prasad, "Satellite communications supporting internet of remote things," *IEEE Internet of Things Journal*, vol. 3, no. 1, pp. 113–123, 2016.

[5] J. Kalajdjieski, B. R. Stojkoska, and K. Trivodaliev, "IoT based framework for air pollution monitoring in smart cities," in *Proceedings of the 2020 28th Telecommunications forum (TELFOR'20)*, pp. 1–4, IEEE, Belgrade, Serbia, November 2020.

[6] E. M. Culpa, J. I. Mendoza, J. G. Ramirez, A. L. Yap, E. Fabian, and P. V. Astillo, "A cloud-linked ambient air quality monitoring apparatus for gaseous pollutants in urban areas," *Journal of Internet Services and Information Security (JISIS)*, vol. 11, no. 1, pp. 64–79, 2021.

[7] T. Wei, W. Feng, Y. Chen, C. Wang, N. Ge, and J. Lu, "Hybrid Satellite-Terrestrial Communication Networks for the Maritime Internet of Things: Key Technologies, Opportunities, and Challenges," 2019, http://www.w3.org/1999/xlink:href 1903.11814.

[8] D. Palma and R. Birkeland, "Enabling the internet of arctic things with freely-drifting small-satellite swarms," *IEEE Access*, vol. 6, pp. 71435–71443, 2018.

[9] NSR, "M2M and IoT via Satellite," 2017, http://www.nsr.com/research-reports/satellite-communications-1/m2 m-and-iot-via-satellite-7th-edition/.

[10] H. Anada and Y. Ueshige, "Anonymous deniable predicate authentication scheme with revocability," *Journal of Internet Services and Information Security (JISIS)*, vol. 11, no. 3, pp. 1–15, 2021.

[11] C. Fei, B. Jiang, K. Xu, L. Wang, and B. Zhao, "An intelligent load control-based random access scheme for space-based internet of things," *Sensors*, vol. 21, no. 4, 2021.

[12] A. Shamir, *Identity-Based Cryptosystems and Signature Schemes*, Springer, Berlin, Heidelberg, 1985.

[13] D. Pöhn and W. Hommel, "Universal identity and access management framework for future ecosystems," *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA)*, vol. 12, no. 1, pp. 64–84, 2021.

[14] Q. Xing, B. Wang, X. Wang et al., "Unbounded Revocable hierarchical identity-based encryption with adaptive-ID security," in *Proceedings of the 2016 IEEE 18th International Conference on High Performance Computing and Communications; IEEE 14th International Conference on Smart City; IEEE 2nd International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, Sydney, NSW, Australia, December 2016.

[15] P. Chen, Y. Wu, J. Su, and X. Wang, "Comparing performance of hierarchical identity-based signature schemes," *IEICE - Transactions on Info and Systems*, vol. 99, no. 12, pp. 3181–3184, 2016.

[16] Al-Riyami, S. Sattam, and K. G. Paterson, *Certificateless Public Key Cryptography*, Springer, Berlin, Heidelberg, 2003.

[17] M. Bellare and P. Rogaway, *The Exact Security of Digital Signatures-How to Sign with Rsa and Rabin*, Springer, Berlin, Heidelberg, 1996.

[18] M. Choudary Gorantla and A. Saxena, "An efficient certificateless signature scheme," *Computational Intelligence and Security*, vol. 3802, 2005.

[19] X. Cao, K. G. Paterson, and W. Kou, "An attack on a certificateless signature scheme," *IACR Cryptology ePrint Archive*, vol. 367, 2006.

[20] Z. Zhang, D. S. Wong, J. Xu, and D. Feng, *Certificateless Public-Key Signature: Security Model and Efficient Construction*, Springer, Berlin, Heidelberg, 2006.

[21] R. Tso, X. Huang, and W. Susilo, "Strongly secure certificateless short signatures," *Journal of Systems and Software*, vol. 85, no. 6, pp. 1409–1417, 2012.

[22] H. Du and Q. Wen, "Security analysis of two certificateless short signature schemes," *IET Information Security*, vol. 8, no. 4, pp. 230–233, 2014.

[23] D. He, B. Huang, and J. Chen, "New certificateless short signature scheme," *IET Information Security*, vol. 7, no. 2, pp. 113–117, 2013.

[24] Y.-C. Chen, R. Tso, G. Horng, C.-I. Fan, and R. H. Hsu, "Strongly secure certificate less signature: cryptanalysis and improvement of two schemes," *Journal of Information Science and Engineering*, vol. 31, pp. 297–314, 2015.

[25] D. He, J. Chen, and R. Zhang, "An efficient and provably-secure certificateless signature scheme without bilinear pairings," *International Journal of Communication Systems*, vol. 25, no. 11, pp. 1432–1442, 2012.

[26] M. Tian and L. Huang, "Cryptanalysis of a certificateless signature scheme without pairings," *International Journal of Communication Systems*, vol. 26, no. 11, pp. 1375–1381, 2013.

[27] P. Gong and P. Li, "Further improvement of a certificateless signature scheme without pairing," *International Journal of Communication Systems*, vol. 27, no. 10, pp. 2083–2091, 2014.

[28] L. Wang, K. Chen, Y. Long, X. Mao, and H. Wang, "A modified efficient certificateless signature scheme without bilinear pairings," in *Proceedings of the Paper presented at the 2015 International Conference on Intelligent Networking and Collaborative Systems*, Taipei, Taiwan, September 2015.

[29] K. H. Yeh, K. Y. Tsai, and C. Y. Fan, "An efficient certificateless signature scheme without bilinear pairings," *Multimedia Tools and Applications*, vol. 74, no. 16, pp. 6519–6530, 2015.

[30] K. H. Yeh, C. Su, K. K. R. Choo, and W. Chiu, "A novel certificateless signature scheme for smart objects in the internet-of-things," *Sensors*, vol. 17, 2017.

[31] M. Hassouna, E. B. M. Bashier, and B. I. A. Barry, "A strongly secure certificateless digital signature scheme in the random oracle model," *International Journal on Network Security*, vol. 18, pp. 938–945, 2016.

[32] W. Wang, H. Xu, M. Alazab, T. R. Gadekallu, Z. Han, and C. Su, "Blockchain-based reliable and efficient certificateless signature for IIoT devices," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 10, pp. 7059–7067, 2022.

[33] X. Jia, D. He, Q. Liu, and K. K. R Choo, "An efficient provably-secure certificateless signature scheme for internet-of-things deployment," *Ad Hoc Networks*, vol. 71, pp. 78–87, 2018.

[34] H. Du, Q. Wen, S. Zhang, and M. Gao, "A new provably secure certificateless signature scheme for internet of things," *Ad Hoc Networks*, vol. 100, Article ID 102074, 2020.

[35] G. Thumbur, G. S. Rao, P. V. Reddy, N. B. Gayathri, and D. V. R. K. Reddy, "Efficient pairing-free certificateless signature scheme for secure communication in resource-constrained devices," *IEEE Communications Letters*, vol. 24, no. 8, pp. 1641–1645, 2020.

[36] Z. Xu, M. Luo, M. K. Khan, K. K. R. Choo, and D. He, "Analysis and improvement of a certificateless signature scheme for resource-constrained scenarios," *IEEE Communications Letters*, vol. 25, no. 4, pp. 1074–1078, 2021.

[37] A. Wall and I. Agrafiotis, "A Bayesian approach to insider threat detection," *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA)*, vol. 12, no. 2, pp. 48–84, 2021.

[38] F. L. Greitzer, J. Purl, and P. J. Sticha, "Use of expert judgments to inform bayesian models of insider threat risk," *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA)*, vol. 12, no. 2, pp. 3–47, 2021.

[39] X. Huang, Y. Mu, W. Susilo, D. S. Wong, and W. Wu, "Certificateless signatures: new schemes and security models," *The Computer Journal*, vol. 55, no. 4, pp. 457–474, 2011.

[40] W. Wang, Z. Han, M. Alazab, T. R. Gadekallu, X. Zhou, and C. Su, "Ultra super fast authentication protocol for electric vehicle charging using extended chaotic maps," *IEEE Transactions on Industry Applications*, vol. 58, no. 5, pp. 5616–5623, 2022.

Hindawi
Security and Communication Networks
Volume 2022, Article ID 5021125, 11 pages
https://doi.org/10.1155/2022/5021125

*Research Article*

# Comparative Experiment on TTP Classification with Class Imbalance Using Oversampling from CTI Dataset

**Heejung Kim**[1] **and Hwankuk Kim**[2]

[1]*Department of Electronics Information and System Engineering, Sangmyung University, Cheonan 31066, Republic of Korea*
[2]*Department of Information Security Engineering, Sangmyung University, Cheonan 31066, Republic of Korea*

Correspondence should be addressed to Hwankuk Kim; rinyfeel@smu.ac.kr

Received 7 September 2022; Accepted 28 September 2022; Published 12 October 2022

Academic Editor: Zhe-Li Liu



Cyber threat intelligence (CTI) refers to the real-time collection of threat information and analysis of these acquired data to identify the situation and attack mechanism of a security threat. In a CTI analysis, it is important to have a standardized attack model. Recently, the MITRE adversarial tactics, techniques, and common knowledge (ATT&CK) framework has been widely used as the de facto standard security threat modeling technique. However, analyzing a large amount of data using the tactics, techniques, and procedures (TTP) of ATT&CK with a limited number of security personnel is time-consuming. To solve this cost-sensitive issue, research on automated classification of TTP from CTI data using artificial intelligence techniques is currently underway but remains challenging. This is because CTI data are domain-specific, and therefore, it is difficult to obtain labeling data to be used as training data for AI models. Hence, the distribution of training data related to TTP labeling is imbalanced. Thus, the current accuracy of ML-based TTP classification is still around 6080%. This study aims to improve the TTP classification accuracy from unstructured CTI data using machine learning while mainly focusing on solving the problems of small training datasets and TTP class imbalance. Therefore, we proposed a TTP classification method by applying easy data argumentation (EDA) and compared its performance with those of previous studies. By applying the proposed methodology, a 60–80% improvement was observed compared to the reference baseline model, TRAM. This indicates that the preprocessing methodology of applying the EDA technique is effective at improving the performance of TTP classification from unstructured CTI data in the CTI domain.

## 1. Introduction

The security operations center (SOC) collects security threat data to protect an organization's ICT infrastructure from internal and external cyber threats while monitoring and responding to security breaches. However, with the gradual expansion and ever-increasing number of cyberattacks, it is becoming more challenging for the SOC to promptly handle security solution events and respond to security breaches. This is because the time required to analyze a large amount of data and to provide a sophisticated response is long, and there is a dearth of skilled security personnel and resources. Security orchestration, automation, and response (SOAR) technology [1], a new paradigm of security control technology, solves these issues by automating various security threat response processes to effectively reduce repetitive tasks of security personnel and helps to quickly and accurately respond to various security events.

The core of SOAR is the integration and automation of security, orchestration, and automation (SOA), security incident response platform (SIRP), and threat intelligence platform (TIP) features [2]. SOA is a feature that interlocks and automates different workflows between numerous security solutions. The SIRP enables the automation of the process of responding to a security incident according to the response policy for each type of security incident. The TIP enables real-time collection and correlation analysis of internal and external threat data. A cyber threat intelligence (CTI) analysis is becoming crucial in quickly and effectively responding to advanced cyberattacks.

Cyber threat intelligence (CTI) data comprise of various information related to cyber threats, including information on attackers, attack procedures, and attack methods and consist of threat data analyzed by security experts, data collected from various threat sensors (such as threat data and detection data), and other related data [3]. Artificial intelligence (AI) models trained using such data are being increasingly used in the detection of new threats. In the recent past, the MITRE adversarial tactics, techniques, and common knowledge (ATT&CK) framework has often been used when analyzing cybersecurity threats and establishing a response strategy [4]. This is because the ATT&CK framework is an open-source project that is easily interoperable with other security-threat-related frameworks, such as CVE, CVSS, CAPEC, and CPE, developed by MITRE, and can be updated regularly whenever new attack techniques and patterns are discovered.

In contrast, using CTI data in conjunction with the tactics, techniques, and procedures (TTP) of MITRE ATT&CK is difficult. This is because extracting TTP information from CTI data, which are often in the form of a report, is cost-sensitive and time-consuming because CTI reports, such as the advanced persistent threat (APT) report, are unstructured threat data provided in sentence form. Manually converting these explanatory TTP sentences into the TTP naming or ID format of the ATT&CK structure is time-consuming and requires strong expertise [5]. To address these problems, there have been several efforts since 2018 to identify (extract) TTP information from CTI reports or to automatically classify the tactics and techniques in TTP.

However, several issues must be addressed to automatically increase TTP extraction or classification performance from CTI reports using AI models [6]. The first issue is insufficient training data. Training data composed of labeled TTP data, which are output data related to CTI data and are required as the input data for machine learning models, are not sufficiently available. The second issue is that of generalization error due to miss detection. As attackers constantly vary their attacks and use more advanced attack techniques, the continuous updating of TTP classification for CTI reports with new attack techniques may result in significant generalization errors and inaccurate results.

The purpose of this study is to improve TTP classification performance with insufficient training data by comparing and testing various data sampling methods.

The contributions of this study are as follows:

(i) In order to address the issues of insufficient dataset size and class imbalance in the field of CTI, two oversampling techniques, namely, synthetic minority oversampling technique (SMOTE) and easy data augmentation (EDA), were utilized, and changes in the TTP classification performance for sentence units of CTI reports were measured.

(ii) The experiment results showed better precision, recall, and $F1$ scores at the sentence level than previous works, which were reference models. An experiment with three datasets was conducted to show the generalization performance.

The structure of this study is as follows. Section 2 describes previous studies related to MITRE ATT&CK modeling and machine learning (ML)-based TTP classification. Section 3 defines the problem and describes the proposed methodology of this study. Section 4 describes the experimental design and evaluation metrics. Section 5 discusses the results and the comparative analysis with previous studies for verification of the proposed method. Finally, Section 6 describes the conclusions, implications, and future research directions.

## 2. Preliminary

*2.1. MITRE TTP Modeling.* In CTI analysis, security threat modeling is a key step for developing and evaluating defense systems against targeted attacks, such as APT attacks and spear phishing. Security threat modeling, covering the various cyber kill chains, tactics, techniques, and procedures used by attackers to carry out attacks has long been studied, and well-known examples include STRIDE, Cyber Kill Chain, and MITRE ATT&CK modeling. Table 1 shows the characteristics of the three modeling approaches.

STRIDE [7], developed by Praerit Garg and Loren Kohnfelder of Microsoft in 1999, was the first model to identify computer security threats and it was the model with the highest level of abstraction. We modeled six representative security threats that infringe on the three major elements of information protection, namely, confidentiality, integrity, and availability.

Cyber Kill Chain [8] was announced by Lockheed Martin in 2009 and is a strategic model for blocking APT attacks infiltrating the company in seven stages, namely, reconnaissance, weaponization, delivery, exploitation, installation, command and control, and exfiltration. The cyber kill chain model makes more specific attack steps than STRIDE, and defenders can utilize the cyber kill chain model when establishing a step-by-step defense strategy against APT attacks.

The MITRE ATT&CK framework [4, 9] is a modeling technique developed by MITRE in 2018. As shown in Figure 1, ATT&CK consists of tactics, techniques, and procedures related to attack techniques used to analyze the lifecycle of cyber attackers and achieve attack goals in the pre- and post-attack exploit operational stages. Currently, the enterprise ATT&CK matrix has 14 tactics and around 200 techniques (in the case of techniques, there are about 578 in total, which includes subtechniques).

*2.2. Related Work.* The analysis of advanced attack technologies is becoming crucial for responding quickly and effectively to intelligent cyber threats. To effectively analyze cyberattacks, the information used in cyberattacks (e.g., malicious code, IP, domain, and vulnerability), the similarity between resources, attack techniques, attack targets, and activity times should be analyzed.

To identify TTP from CTI data using ML techniques, the type of CTI data used as input data is important. CTI data can be categorized as structured data and unstructured data.

Table 1: Characteristics of the three threat modeling methodologies.

| Characteristics | Stride | Cyber kill chain | MITRE ATT&CK |
|---|---|---|---|
| Source | 1999, Microsoft | 2009, Lockheed Martin | 2018, MITRE |
| Level of abstraction | High | Medium | Low (detail) |
| Level of modeling | Attack type level | Tactics level | Tactics, techniques, and procedures |
| Features | Spoofing, tampering, repudiation, information disclosure, denial of service, and the elevation of privilege | Reconnaissance, weaponization, delivery, exploitation, installation, command and control, and exfiltration | Reconnaissance, resource development, initial access, execution, persistence, privilege escalation, defense evasion, credential access, discovery, lateral movement, collection, command and control, exfiltration, and impact |

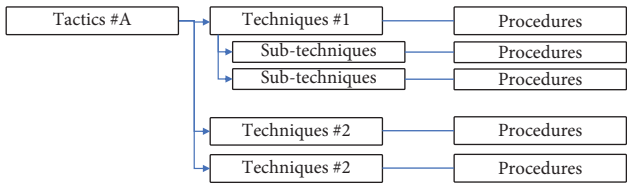



Figure 1: MITRE ATT&CK components (tactics, techniques, procedures).

Structured CTI data can express and contain TTP information in standardized formats, such as STIX, Database, and JSON, making it easier to identify TTP data from structured data than from unstructured data. However, the TTP data must be entered in advance in the specification field. Unstructured CTI data can have various forms, including reports and web pages, and when new threats arise, they are often shared in the form of reports. Therefore, studies on the use of AI and natural language processing (NLP) techniques for automated TTP identification or classification from unstructured data began in 2017.

TRAM [6] released an open-source TRAM that can automatically identify and classify TTP from CTI reports using machine learning at MITRE. This model makes the greatest contribution by disclosing proof-of-concept codes and data networks that can automatically classify tactics, techniques, and procedures with machine learning and NLP techniques. TRAM built its own dataset in which the output performs multiclassification at the techniques level of TTP by receiving input from the CTI report at a sentence unit from the input layer. The classification performance ranged between 50% and 60%.

Husari et al. proposed TTPDrill [10] and ActionMiner [11]. TTPDrill aimed to collect CTI reports from its website to identify ATT&CK techniques and CAPEC attack patterns at the document level. This approach extracts and weighs threat action-related candidate information, namely, subject, verb, and object, from each CTI report through part-of-speech tagging, and then generates 187 techniques and 19 tactics and converts them into a STIX structure. In addition, the ActionMiner model was published as a follow-up study. The purpose of this model was to find the same threat information in CTI reports by extracting object-verb pairs related to malicious software using entropy and mutual information from Wikipedia.

Legoy et al. [12] proposed the rcATT model, which is an ML model used for automatically identifying TTP from sentence units in unstructured CTI reports. This approach uses term frequency-inverse document frequency (TF-IDF) and Word2Vec as word embedding techniques, and the decision tree, support vector classifier, and AdaBoost models as classifiers. The multiclass classification performance was measured as 79.3% at the tactics level and 72.22% at the techniques level.

Ayoade et al. [13] proposed a TTP classification model using TF-IDF and support vector machines. The proposed model uses the Symantec dataset as the training dataset and APT reports as the test dataset to extract attacker actions from various CTI reports. In addition, classification performance experiments were conducted by applying various bias correction methods. The classification performance obtained was 63% at the tactics level and 96.3% at the kill chain level.

Nakanishi et al. [14] proposed the SECCMiner model. This model is not an ML-based TTP automatic classification model, and its purpose is to identify TTP-related keywords included in CTI reports using the TF-IDF NLP technique.

Kim et al. [15] proposed a technology to collect indicators of compromise (IoC) from CTI reports using NLP techniques. The IoC data and attack techniques (TTP) used for cyberattacks were extracted using the SyntaxNet technique from Google. Evaluation of 190 reports based on the $F1$ score showed an average performance of 76%.

You et al. [5] proposed a TTP intelligence mining model that extracts and classifies TTP information from unstructured CTI reports. For this model, Sentence-BERT embeddings were used in the feature extraction step, and a two-dimensional convolutional neural network and bidirectional long short-term memory network were used as classifiers, and a high $F1$ score of 0.97 was obtained. In particular, a model that focuses on embedding techniques related to the text in unstructured CTI reports was proposed. Experiments were conducted to classify six attack classes based on 6,061 TTP-related sentences for the dataset.

In summary, previous studies mainly utilized two approaches. They could be categorized as studies that aimed to find TTP and IoC data from CTI reports using various NLP techniques, and studies that classified TTP in the MITRE ATT&CK framework using ML techniques. However, the

performance of identifying threat information or classifying it as TTP showed results of 70% to 80%. In addition, previous studies have suggested that for the automation of CTI analysis, it is necessary to solve the issues relating to securing quality training data and minimizing the generalization error between training data and actual data. This indicates that research on technology to automatically identify or classify cyber threat information using AI techniques is still in the early stages and that there are many open issues to solve.

## 3. Proposed Model

*3.1. Problem Definition.* The biggest issues facing automated TTP classification in CTI are related to the quality of training data, such as small dataset size and class imbalance. The performance of ML models depends on the quality of the data for training the models. If the training data are not balanced across different classes, the performance of the ML model is significantly degraded. Although most learning models perform learning under the assumption that the proportion of the training classes is similar and provides high-performance results, however, in practice, predictive accuracy increases for classes with large data distributions and decreases for classes with small distributions, which lowers the overall performance.

CTI data are in the form of a report in unstructured text sentences. Features $X$, in which this type of report is entered into the TTP classification, uses sentences or documents that make up the CTI report itself as an input. Output $Y$ can also be classified as tactics or techniques of attacking TTP. However, the biggest problem with CTI is that samples comprising input training data with TTP labels are extremely rare. This is because the CTI data and TTP information are domain-specific, and therefore, there are not much learning data for label information. Since TTP information is the result of analyzing cyber threat information by security experts, such as log information of security equipment or hacker's attack techniques, it takes several months to analyze TTP. Therefore, it takes a long time for training data with TTP labels to be opened, so it is difficult to collect training data. Moreover, it is difficult to obtain labeling datasets because ML-based TTP classification studies are in the early stages. In addition, because TTP consists of 12 tactics and 200 techniques, it has unbalanced data characteristics that inevitably result in significantly fewer data instances for each class. Currently, the only training dataset used to automatically classify TTP in unstructured CTI data is the training dataset provided by TRAM of MITRE.

*3.2. Class Imbalance Issues.* In supervised learning, the problem of class imbalance can arise when there is an unbalanced distribution of classes in the training dataset [16]. While imbalances in class distributions can vary, severe imbalances are more difficult to model and may require specialized skills. The general solutions for addressing unbalanced data sampling currently include oversampling and undersampling [17].

Undersampling is the process of reducing the sample size of the majority class, which has a higher proportion, to balance the amount of sample data belonging to each class. However, performance degradation may occur due to the loss of useful data because data belonging to the majority class is omitted. Oversampling is a technique that supplements the training data with multiple copies of some minority classes to increase the sample size of the minority class. Existing oversampling techniques include random oversampling, SMOTE, and data augmentation.

SMOTE [18] is an oversampling technique proposed by Chawla et al. in 2002. This technique involves the use of the k-nearest neighbor algorithm to find close neighbors of data instances belonging to the minority class and to generate virtual data through interpolation, such that the virtual data corresponds to the minority class and is not identical to the original data. In other words, a sample of a class with a small number of instances is taken and a random value is added to create a new instance, which is then added to the data [6].

EDA [19], published at the EMNLP (2019) conference, is a technique for increasing the amount of data by transforming the currently available data and is used when the amount of training data is insufficient or when a class imbalance occurs. In text classification tasks, the EDA technique improves the performance of classification models with only a small amount of data without requiring additional external data or generation models. The methods used in EDA include synonym replacement, which replaces a specific word with a synonym; random deletion, which deletes a random word; random insertion, which selects words within a sentence and inserts them into any position in the sentence; and random swap, which repositions any two words in a sentence [20, 21]. The training data can be increased in various forms using the EDA technique, which improves the performance of AI models. The above four methods can be used to produce $4 + \alpha$ augmented sentences for one sentence. It was also proved in the study that the sentences made in this way preserve the label of the original sentence, that is, the original meaning. Also, when generating sentences, noise is properly generated, which can suppress the overmatching phenomenon that may occur in data shortage problems.

Back translation [22] was first introduced in the 2016 ACL. This study attempted to improve the performance of machine translation using monolingual data. One of the methods to improve the performance of machine translation was the back translation, which was suggested to be effective. The machine translation model has an encoder-decoder structure. The source sentence is inputted to the encoder and the target sentence is inputted to the decoder and then the training of the translation model is proceeded. The author of this study proposed a methodology to create artificial source sentences with no perfect sentence format using target sentences. In other words, the original sentence is translated into another language and then retranslated to create a poor source sentence. Based on this concept, back translation techniques have been used in several studies to increase the amount of training data for performance improvement in text classification models.

In this study, the oversampling technique was used to solve the class imbalance problem of limited training data for TTP automatic classification. A small number of class distributions degrade the performance of the classification model. The reason is that the training results of the unbalanced data can bring biased results for a number of class data. However, if a small number of class data are removed and only a large number of class data are used, it may be difficult to properly classify a small number of TTP techniques. The oversampling technique was used as a methodology that can sufficiently utilize limited training data. In addition, the performance improvement results and effectiveness of the classification model using the oversampling technique were verified.

## 4. Experiments Design

*4.1. Baseline Model.* In this study, we used the TRAM model from MITRE as a baseline to compare the effectiveness of the proposed method. The TRAM model was used as the baseline model because the classifier and training dataset used were open, making it easier to compare the results in terms of the reproducibility and feasibility of the model.

*4.1.1. Classifier.* The classifiers used in TRAM are logic regression (LR), Naive Bayes (NB), and multilayer perception (MLP). The input $X$ contains the CTI-related sentence text after conversion using the countVectorizer data representation method. The output $Y$ of the classifier is the result of classifying the data into multiple classes in units of techniques of TTP.

*4.1.2. Datasets.* We prepared three training datasets for improving the generalization problem of the experiment. Table 2 summarizes the features of the three datasets.

The first dataset was a training dataset provided by TRAM. This training data comprised of 1,410 CTI-related sentences and 100 classes corresponding to the techniques level of TTP.

The second dataset was the data we prepared. The amount of training data provided by TRAM was small and the number of techniques to be classified was 100. Information provided by MITRE ATT&CK was collected to organize 578 techniques (including subtechniques) related to a total of 4,250 sentences. However, the number of instances per class was limited to 24. The last dataset was a combined dataset, comprising of the TRAM dataset and our dataset, which consisted of 578 techniques related to 5,660 sentences. Figure 2 shows the distribution plot of the three datasets.

As shown in Figure 2, the combined dataset data distribution was reinforced compared to the distribution of samples per TTP class provided by TRAM.

*4.2. Experimental Procedure.* Figure 3 shows the experimental procedure in two steps. The first step is a preprocessing stage that involves data preprocessing, sentence representation, and oversampling. Sentence representation was performed using the countVectorizer, a bag-of-words technique that expresses text as a numerical feature vector. Then, oversampling techniques such as SMOTE and EDA were applied to the training set, and the dataset was split into training and testing sets in an 8 : 2 ratio. In the second step, the classification and model evaluation was performed by training the classification model using the training set. Then, the performance of the classification model was evaluated on the testing set.

Table 2: Temperature and wildlife count in the three areas covered by the study.

| | $X$ (sentences) | $Y$ (techniques) | $X$ per $Y$ |
|---|---|---|---|
| TRAM dataset | 1,410 | 100 | 1–95 |
| Proposed dataset | 4,250 | 578 | 1–24 |
| Combined dataset | 5,660 | 578 | 2–103 |

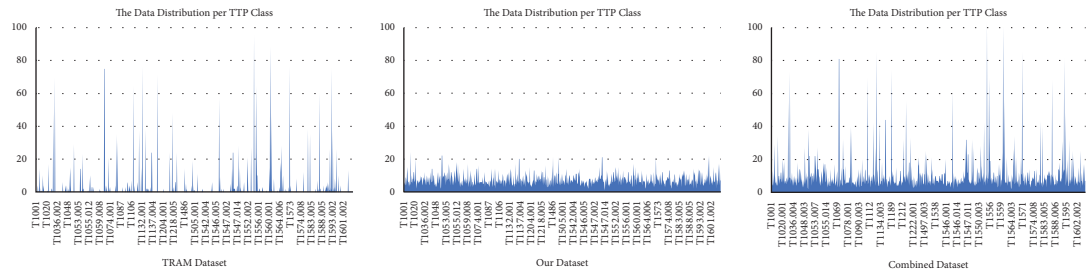
This experiment was processed in two ways. The first experiment measured the classification performance of the baseline model using the three datasets to provide a baseline for a comparative analysis. The second experiment measured the classification performance of the ML models with oversampling techniques. In this experiment, the SMOTE and EDA oversampling techniques were used.

*4.3. Evaluation Methods.* This section explains the evaluation method for the proposed model using our dataset. This experiment used accuracy, precision, recall, and the $F1$ score of the confusion matrix as performance indicators of the classification model. Since this experiment is a multiclassification and an imbalanced class problem, we focused on the $F1$ score and micro/macro average scores as performance metrics. In classification problems, the precision, recall, and $F1$ scores change depending on the number of instances in the target class. Therefore, we used the microaverage and macroaverage metrics, which are methods for averaging the performance for each target class and evaluating the performance of classification models with imbalanced classes. The microaverage is a method of taking the average that considers the number of instances in each class when calculating the average and is a metric that can respond sensitively to class imbalance. The macroaverage is a method of taking the average regardless of the number of instances in each class and is an indicator that can evaluate the overall performance of the model.

## 5. Results and Discussion

*5.1. Experiment Results.* This section describes the experimental results of baseline models, and the experimental results applied by oversampling techniques, SMTOE, and EDA, respectively.

*5.1.1. Results with Baseline Model.* This result is the baseline experiment to compare the effectiveness of our approach, oversampling techniques. The results of the first experiment are shown in Table 3. Using the training data provided by the




(a) (b) (c)

Figure 2: The distribution per TTP classes in 3 datasets: (a) dataset by TRAM, (b) our dataset, and (c) combined dataset (TRAM dataset + our dataset).

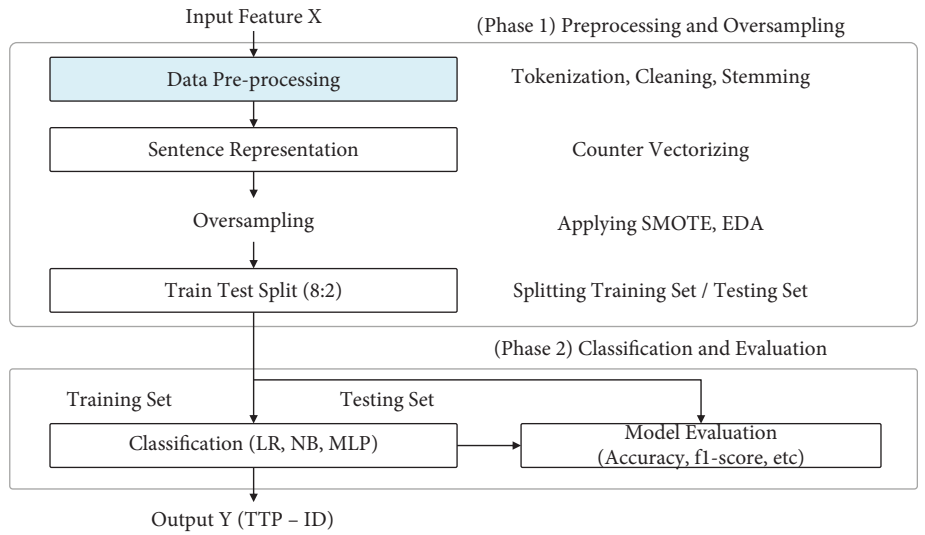



Figure 3: The procedure.

Table 3: The result of baseline experiment (%).

| Data sets | Model | Accuracy (%) | Microaverage (%) | | | Macroaverage (%) | | |
|---|---|---|---|---|---|---|---|---|
| | | | Precision | Recall | *F*1 | Precision | Recall | *F*1 |
| TRAM dataset (1,510) | LR | 59.60 | 59.60 | 59.60 | 59.60 | 34.39 | 34.39 | 34.39 |
| | NB | 50.99 | 50.99 | 50.99 | 50.99 | 29.41 | 28.92 | 29.16 |
| | MLP | 61.26 | 61.26 | 61.26 | 61.26 | 39.97 | 38.85 | 39.40 |
| | Avg. | 57.28 | 57.28 | 57.28 | 57.28 | 34.59 | 34.05 | 34.32 |
| Proposed dataset (4,250) | LR | 29.06 | 29.06 | 29.06 | 29.06 | 23.09 | 23.55 | 23.32 |
| | NB | 21.29 | 21.29 | 21.29 | 21.29 | 15.09 | 17.06 | 16.02 |
| | MLP | 27.88 | 27.88 | 27.88 | 27.88 | 22.35 | 22.89 | 22.62 |
| | Avg. | 26.08 | 26.08 | 26.08 | 26.08 | 20.18 | 21.17 | 20.65 |
| Combined dataset (5,760 | LR | 36.81 | 36.81 | 36.81 | 36.81 | 24.50 | 25.67 | 25.07 |
| | NB | 25.26 | 25.26 | 25.26 | 25.26 | 13.81 | 14.28 | 14.04 |
| | MLP | 35.59 | 35.59 | 35.59 | 35.59 | 26.48 | 28.31 | 27.36 |
| | Avg. | 32.55 | 32.55 | 32.55 | 32.55 | 21.60 | 22.75 | 22.16 |

TRAM model, which was used as the baseline model, a classification accuracy between 32.55% and 57% at the techniques level was shown. The micro-*F*1 score was between 26.08% and 57.28% and the macro-*F*1 score was between 20.65% and 34.32%. The reason for the poor performance is that the total amount of data samples is insufficient, which is affected by a small set of data, making it unsuitable.

*5.1.2. Results with SMOTE.* Table 4 shows the classification performance with the SMOTE sampling technique. When applying the SMOTE algorithm, the value of the neighbor *k* value parameter was set to 1. The meaning of the *K* value of 1 is the minimum number of samples per class for oversampling in SMOTE technique. Compared to the baseline model, the results of applying the SMOTE showed that the TRAM dataset was between 40.98 and 57.71%, and the

Table 4: The results of experiments with SMOTE oversampling techniques.

| Datasets | Model | Accuracy (%) | Microaverage (%) | | | Macroaverage (%) | | |
|---|---|---|---|---|---|---|---|---|
| | | | Precision | Recall | *F*1 | Precision | Recall | *F*1 |
| TRAM dataset (1,510) | LR | 58.50 | 58.50 | 58.5% | 58.50 | 43.12 | 44.55 | 43.82 |
| | NB | 56.46 | 56.46 | 56.46 | 56.46 | 38.56 | 40.42 | 39.47 |
| | MLP | 58.16 | 58.16 | 58.16 | 58.16 | 38.99 | 40.31 | 39.64 |
| | Avg. | 57.71 | 57.71 | 57.71 | 57.71 | 40.22 | 41.76 | 40.98 |
| Proposed dataset (4,250) | LR | 26.82 | 26.82 | 26.82 | 26.82 | 22.55 | 23.81 | 23.16 |
| | NB | 26.82 | 26.82 | 26.82 | 26.82 | 20.57 | 22.38 | 21.44 |
| | MLP | 25.65 | 25.65 | 25.65 | 25.65 | 20.12 | 22.84 | 21.39 |
| | Avg. | **26.43** | 26.43 | 26.43 | 26.43 | 21.08 | 23.01 | 22.00 |
| Combined dataset (5,760) | LR | 29.86 | 29.86 | 29.86 | 29.86 | 22.55 | 25.20 | 23.80 |
| | NB | 28.82 | 28.82 | 28.82 | 28.82 | 19.95 | 22.01 | 20.93 |
| | MLP | 26.22 | 26.22 | 26.22 | 26.22 | 19.87 | 22.30 | 21.01 |
| | Avg. | 28.30 | 28.30 | 28.30 | 28.30 | 20.79 | 23.17 | 21.92 |

Table 5: The result of performance with EDA oversampling techniques.

| Datasets | Model | Accuracy (%) | Microaverage (%) | | | Macroaverage (%) | | |
|---|---|---|---|---|---|---|---|---|
| | | | Precision | Recall | *F*1 | Precision | Recall | *F*1 |
| TRAM dataset (30,160) | LR | 98.24 | 98.24 | 98.24 | 98.24 | 97.56 | 96.32 | 96.94 |
| | NB | 92.08 | 92.08 | 92.08 | 92.08 | 73.56 | 66.84 | 70.04 |
| | MLP | 98.76 | 98.76 | 98.76 | 98.76 | 97.27 | 96.83 | 97.05 |
| | Avg. | 96.36 | 96.36 | 96.36 | 96.36 | 89.46 | 86.66 | 88.01 |
| Proposed dataset (84,870) | LR | 93.64 | 93.64 | 93.64 | 93.64 | 93.88 | 93.58 | 93.73 |
| | NB | 85.06 | 85.06 | 85.06 | 85.06 | 88.71 | 82.73 | 85.62 |
| | MLP | 93.70 | 93.70 | 93.70 | 93.70 | 94.04 | 93.59 | 93.81 |
| | Avg. | 90.80 | 90.80 | 90.80 | 90.80 | 92.21 | 89.97 | 91.05 |
| Combined dataset (115,030) | LR | 94.75 | 94.75 | 94.75 | 94.75 | 94.31 | 93.70 | 94.01 |
| | NB | 83.61 | 83.61 | 83.61 | 83.61 | 88.58 | 78.29 | 83.12 |
| | MLP | 94.95 | 94.95 | 94.95 | 94.95 | 94.38 | 94.01 | 94.20 |
| | Avg. | 91.11 | 91.11 | 91.11 | 91.11 | 92.43 | 88.67 | 90.44 |

proposed dataset was between 22% and 26.43%, and the combined dataset was between 21.92% and 28.30%. The performance of the experiment with SMOTE showed little improvement compared to the reference model. Oversampling with the SMOTE technique is affected by adjacent k values, so the number of samples per class is required. Since the dataset used in this experiment contains classes with a small number of samples, it seems that the experiment was conducted with the k = 1, resulting in low performance.

*5.1.3. Results with EDA.* Data augmentation techniques for oversampling exist in text modification and generation methods. One of the modification methods is easy data augmentation (EDA), which augments text without external data using four text editing techniques and taking back translation and conditional pretraining as generation methods. In this study, we perform the experiment by using EDA and back translation.

Table 5 shows the classification performance with the EDA-BT (back translation) technique. Compared with the baseline model, the results of applying the EDA technique showed that the classification performance in the case of the TRAM dataset was between 88.01% and 96.36%, in the case of the proposed dataset it was between 90.80% and 91.05%, and in the case of the combined dataset it was between 90.44% and 91.11%.

*5.2. Discussion: Comparative Analysis.* This section compares the experimental results described in the previous section and analyses the results of previous studies and current studies.

*5.2.1. Experiments Comparison.* The experimental comparison of each technique is the result of comparison before and after using of oversampling with SMOTE and EDA. Here, accuracy and *F*1 score were used as predictive performance metrics and ROC-AUC metrics were used to evaluate the effectiveness of the model. As shown in Table 6, the classification results using EDA oversampling are 90% to 95% on an average, and this result shows good classification performance.

These results show that the EDA technique compared to the baseline model has significantly improved on an average in accuracy and micro/macro-*F*1 score regardless of dataset type.

Figure 4 is a graph comparing the performance results of the baseline model, applying SMOTE and EDA, respectively, with accuracy, micro-*F*1 scores, and macro-*F*1 scores for multiple classifications. Here, the *X*-axis is divided into three datasets and the *Y*-axis is the result of each performance metric. Figure 4(a) is the classification performance with the baseline model and Figure 4(b) shows

Table 6: Comparison of experiments (ACC: accuracy, macro-$F1$, and AUC: ROC-AUC).

| Models | | TRAM dataset | | | Proposed dataset | | | Combined dataset | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | ACC | $F1$ | AUC | ACC | $F1$ | AUC | ACC | $F1$ | AUC |
| The base model | LR | 59.6 | 59.6 | 0.797 | 29.1 | 29.1 | 0.645 | 36.8 | 36.8 | 0.684 |
| | NB | 51.0 | 51.0 | 0.753 | 21.3 | 21.3 | 0.606 | 25.3 | 25.3 | 0.626 |
| | MLP | 61.3 | 61.3 | 0.808 | 27.9 | 27.9 | 0.647 | 35.6 | 35.6 | 0.668 |
| | Avg | 57.3 | 57.3 | 0.786 | 26.1 | 26.1 | 0.633 | 32.6 | 32.6 | 0.659 |
| With SMOTE | LR | 58.5 | 58.5 | 0.791 | 26.8 | 26.8 | 0.634 | 29.9 | 29.9 | 0.649 |
| | NB | 56.5 | 56.5 | 0.781 | 26.8 | 26.8 | 0.634 | 28.8 | 28.8 | 0.644 |
| | MLP | 58.2 | 58.2 | 0.791 | 25.7 | 25.7 | 0.620 | 26.2 | 26.2 | 0.627 |
| | Avg | 57.7 | 57.7 | 0.788 | 26.4 | 26.4 | 0.629 | 28.3 | 28.3 | 0.640 |
| With EDA + BT | LR | 98.2 | 98.2 | 0.991 | 93.6 | 93.6 | 0.968 | 94.8 | 94.8 | 0.974 |
| | NB | 92.1 | 92.1 | 0.960 | 85.1 | 85.1 | 0.925 | 83.6 | 83.6 | 0.918 |
| | MLP | 98.8 | 98.8 | 0.994 | 93.7 | 93.7 | 0.969 | 95.0 | 95.0 | 0.975 |
| | Avg | 96.4 | 96.4 | 0.982 | 90.8 | 90.8 | 0.954 | 91.1 | 91.1 | 0.956 |

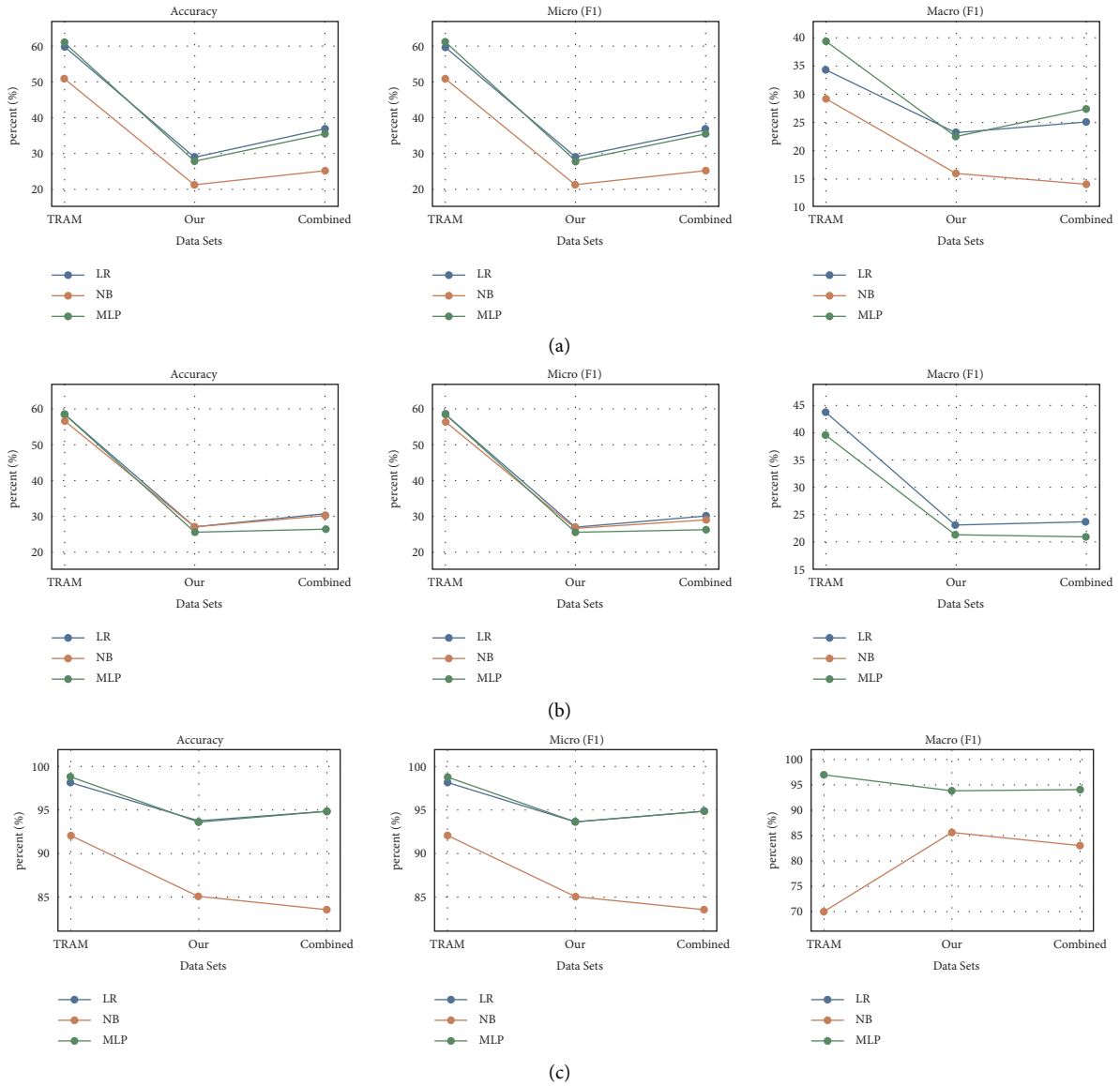



Figure 4: A graph comparing the performance results of the baseline model, applying SMOTE and EDA. (a) Performance of classification with the baseline model. (b) Performance of classification with SMOTE. (c) Performance of classification with EDA.

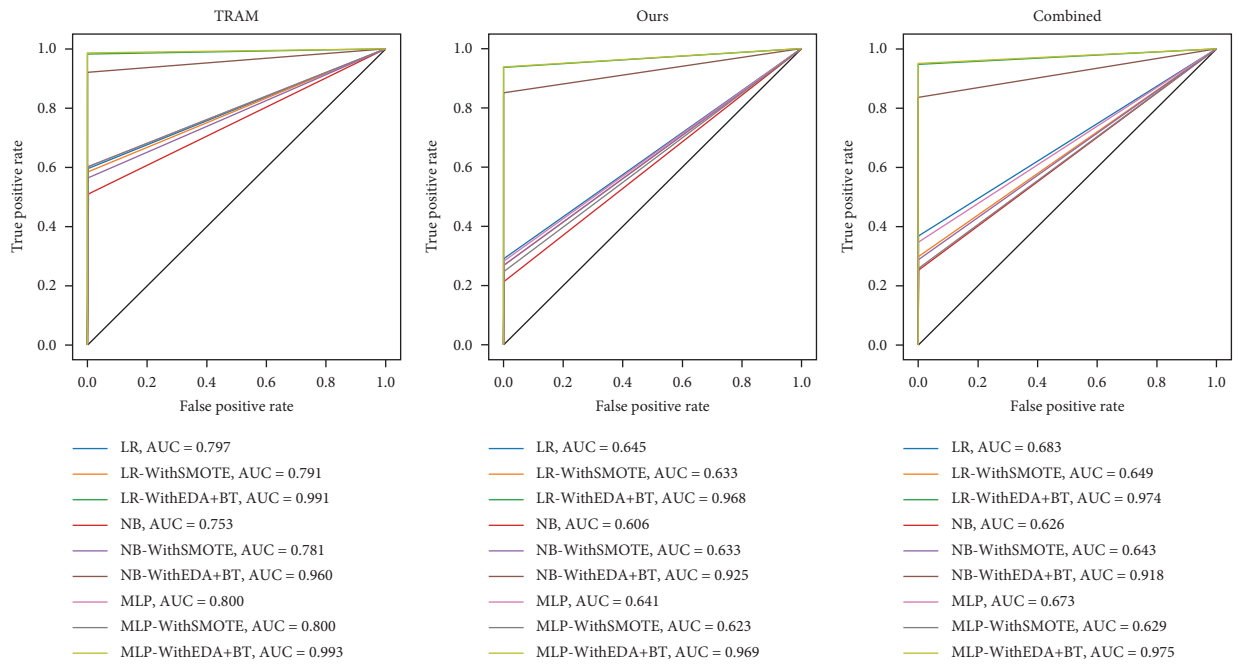



Figure 5: The result of ROC-AUC with each approach in three datasets.

Table 7: The comparison analysis with related works (legend: input = doc (document level), SEN (sentence level), metrics = ACC (accuracy), *F*1 (*F*1score), poc (proof of code), and IoC (indicator of compromise)).

| Works | Objectives | Input | Approaches | Metrics | Performance |
|---|---|---|---|---|---|
| Husari et al. [10, 11] | CTI report ⟶ CAPEC ⟶ STIX conversion | DOC | NLP and ML | PoC | 187 techniques 19 tactics |
| rcATT [12] | CTI report ⟶ TTP classification | DOC | TF-IDF, Word2Vec/AdaBoost | ACC | Tactics: 79.3% Techniques: 72.22% |
| Ayoade et al. [13] | CTI report ⟶ TTP attack actions | DOC | Similarity (TF-IDF) | ACC | Tactics: 63% Kill chain: 96.3% |
| Nakanishi et al. [14] | CTI report ⟶ TTP keywords | DOC | Similarity (TF-IDF) | PoC | 445 reports |
| Kim et al. [15] | CTI report ⟶ IoC | DOC | NLP (SyntexNet) | *F*1 | Keyword extraction: 76% |
| You et al. [5] | CTI report ⟶ TTP classification | SEN | Sentence-BERT embeddings/ LSTM classifier | *F*1 | TTP level: 91% |
| TRAM (baseline) | CTI report ⟶ TTP classification | SEN | Basic ML (LR, NB, and MLP) | *F*1 | Techniques: 50%~ 60% |
| The proposed model (EDA + BT) | CTI report ⟶ TTP classification | SEN | Oversampling/TRAM-based classifier | *F*1 | Techniques: 90.8%~ 96.4% |

the classification performance by using smote oversampling. Figure 4(c) shows the classification performance by applying EDA oversampling. As shown in the figure, the results of EDA improved by about 40% compared to the baseline model before applying the oversampling technique.

Figure 5 shows the ROC-AUC results of the experiment with oversampling. Comparing the ROC-AUC results, which are indicators for evaluating the discriminant power of classification models, the AUC values of SMOTE and baseline models are from 0.62 to 0.78 indicating that the discriminant performance of the model is average. In the case of EDA, the AUC value is from 0.95 to 0.98, which means that it has the best discrimination performance of the model.

*5.2.2. Comparison of Previous Studies.* Table 7 is the result of a comparative analysis between the current work and existing studies. As mentioned in previous studies, the comparative analysis targeted the ML-based TTP automatic classification model.

The purpose of previous studies is similar to the current study with the aim of extracting keywords of TTP from CTI reports, classifying strategies/technologies, or extracting behaviours of attacks. However, the difference from the current study is that different techniques have been applied for each study. Compared to the current approach, it is common to focus on data preprocessing techniques, but there are differences in detailed data preprocessing methods and application models, such as TF-IDF and embedding techniques.

There are also differences in input/output relationships. In five studies, including Husari et al. [10, 11], the input data were used as the document level of the CTI report. These studies aim to define the full content of the CTI report as a TTP or attack model. In three studies, including the current study, the input data were used as sentence-unit texts in the CTI report. It aims to define one sentence as a TTP or attack model. Evaluation metrics also differ from the proposed approach. Husari et al. [10, 11] and Nakanishi et al. [14] focused on implementation over evaluation metrics. In the current work, we selected *F*1 scores as evaluation metrics because we were addressing the class imbalance problem, but rcATT [12] and Ayoade et al. [13] presented evaluation metrics using accuracy indicators.

Direct comparative analysis with previous studies is difficult because the datasets, models, input/output relationships, and purposes used in each study are different. However, since the accuracy of TTP classification is not high at 20–50% on average in text-style CTI reports, and the results of previous studies to solve this problem show performance improvement at 60–90%, thus the results of this study showed good improvement compared to previous studies.

## 6. Conclusions

In this study, we present an automated classification of TTP from CTI data. As the occurrence of cybersecurity threats increases rapidly, it is necessary to quickly identify and respond to attacks. CTI information is mainly used to understand these threat situations and attack mechanisms. It is important to define a large amount of CTI information as a standardized attack model. However, analyzing a large amount of CTI data with a limited number of security personnel is time-consuming. Therefore, in this study, we present an automated method for classifying TTP from unstructured CTI data using machine learning. It is expected that this will enable faster identification and response to security threats.

In this study, we also focus on improving TTP classification accuracy while solving the problem of small training datasets and TTP class imbalances. Imbalanced data is one of the most important problems to be solved in machine learning. We present the comparative experimental results of TTP identification and classification performance by using data augmentation techniques during data preprocessing to address insufficient training data issues in CTI domains. As a result, when the training data augmentation technique was used based on the TRAM model, which is a reference baseline model, a performance improvement of about 60%–80% for the *F*1score was achieved.

However, a limitation of this work is that it is highly prone to generalization errors. In particular, due to the nature of the cybersecurity domain, the accuracy of ML models is bound to vary depending on the content and amount of unknown new security threats or CTI reports, as attackers continue to find new attack techniques to bypass existing defense models. Therefore, after solving this generalization error and classifying TTP from known information through rule-based matching, we believe that the proposed model can be applied to unmatched CTI information through machine learning. Future studies need to consider various improvements, such as quality training data generation, word embedding methods, model selection, and optimization, to improve automated TTP classification performance.

## Data Availability

The experimental data supporting the current study are available from the corresponding author upon request.

## Conflicts of Interest

The authors declare that they have no conflict of interest..

## Acknowledgments

This work was supported by the Institute of Information and Communications Technology Planning and Evaluation (IITP) grant funded by the Korea Government (MSIT) (No. 2021- 0-00358, AI·Big Data Based Cyber Security Orchestration and Automated Response Technology Development).

## References

[1] C. Brooks, "Security orchestration, automation and response (SOAR)-The pinnacle for cognitive cybersecurity," *Security Essentials*, p. 678, Alien Vault, 2018.

[2] F. Gartner, "Security orchestration, automation and response (SOAR)," Availablehttps://www.gartner.com/en/information-technology/glossary/security-orchestration-automation-response-soar[Online; accessed on.

[3] T. D. Wagner, K. Mahbub, E. Palomar, and A. E. Abdallah, "Cyber threat intelligence sharing: survey and research directions," *Computers & Security*, vol. 87, no. 10, pp. 101589–101592, 2019.

[4] S. Mitre, "MITRE ATT@CK framework," Available:https://attack.mitre.org/Online; accessed on, 2018.

[5] Y. You, J. Jiang, Z. Jiang et al., "TIM: threat context-enhanced TTP intelligence mining on unstructured threat data," *Cybersecurity*, vol. 5, no. 1, pp. 3–17, 2022.

[6] S. Yoder, "Automating mapping to att&ck: the threat report att&ck mapper (tram) tool," Available at:https://medium.com/mitre-attack/automating-mapping-to-attack-tram-1bb1b44bda76accessed, 2019.

[7] B. Potter, "Microsoft SDL threat modelling tool," *Network Security*, vol. 29, pp. 15–18, 2009.

[8] J. Straub, "Modeling attack, defense and threat trees and the cyber kill chain, att&ck and stride frameworks as blackboard architecture networks," in *Proceedings of the 2020 IEEE International Conference on Smart Cloud (SmartCloud)*, pp. 148–153, IEEE, Washington, DC, USA, 06-08 November 2020.

[9] A. Georgiadou, S. Mouzakitis, and D. Askounis, "Assessing mitre att&ck risk using a cyber-security culture framework," *Sensors*, vol. 21, no. 9, p. 3267, 2021.

[10] G. Husari, E. Al-Shaer, M. Ahmed, B. Chu, and X. Niu, "Ttpdrill: automatic and accurate extraction of threat actions from unstructured text of cti sources," in *Proceedings of the*

*33rd annual computer security applications conference*, pp. 103–115, New york, December 2017.

[11] G. Husari, X. Niu, B. Chu, and E. Al-Shaer, "Using entropy and mutual information to extract threat actions from cyber threat intelligence," in *Proceedings of the 2018 IEEE international conference on intelligence and security informatics (ISI)*, pp. 1–6, IEEE, Miami, FL, USA, 09-11 November 2018.

[12] V. Legoy, M. Caselli, C. Seifert, and A. Peter, "Automated retrieval of att&ck tactics and techniques for cyber threat reports," *Discover*, vol. 204, p. 14322, 2020.

[13] G. Ayoade, S. Chandra, L. Khan, K. Hamlen, and B. Thuraisingham, "Automated threat report classification over multi-source data," in *Proceedings of the 2018 IEEE 4th International Conference on Collaboration and Internet Computing (CIC)*, pp. 236–245, IEEE, Philadelphia, PA, USA, 18-20 October 2018.

[14] A. Niakanlahiji, J. Wei, and B. T. Chu, "A natural language processing-based trend analysis of advanced persistent threat techniques," in *Proceedings of the 2018 IEEE International Conference on Big Data (Big Data)*, pp. 2995–3000, IEEE, Seattle, WA, USA, 10-13 December 2018.

[15] N. Kim, M. Kim, S. Lee et al., "Study of natural language processing for collecting cyber threat intelligence using syntaxnet," *International Symposium of Information and Internet Technology*, pp. 10–18, Springer, Cham, 2018.

[16] S. Kotsiantis, D. Kanellopoulos, and P. Pintelas, "Handling imbalanced datasets: a review," *GESTS international transactions on computer science and engineering*, vol. 30, no. 1, pp. 25–36, 2006.

[17] R. Mohammed, J. Rawashdeh, and M. Abdullah, "Machine learning with oversampling and undersampling techniques: overview study and experimental results," in *Proceedings of the 2020 11th international conference on information and communication systems (ICICS)*, pp. 243–248, IEEE, Irbid, Jordan, 07-09 April 2020.

[18] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, no. 2002, pp. 321–357, 2002.

[19] J. Wei and K. Zou, "Eda: easy data augmentation techniques for boosting performance on text classification tasks," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pp. 6382–6388, EMNLP-IJCNLP), 2019.

[20] B. Li, Y. Hou, and W. Che, "Data augmentation approaches in natural language processing: a survey," *AI Open*, vol. 3, pp. 71–90, 2022.

[21] P. Liu, X. Wang, C. Xiang, and W. Meng, "A survey of text data augmentation," in *Proceedings of the 2020 International Conference on Computer Communication and Network Security (CCNS)*, pp. 191–195, IEEE, Xi'an, China, 21-23 August 2020.

[22] R. Sennrich, B. Haddow, and A. Birch, "Improving neural machine translation models with monolingual data," *Annual Meeting of the Association for Computational Linguistics (ACL)*, vol. 1, pp. 86–96, 2016.

Hindawi
Security and Communication Networks
Volume 2022, Article ID 9569703, 10 pages
https://doi.org/10.1155/2022/9569703

*Research Article*

# CAN Signal Extinction-based DoS Attack on In-Vehicle Network

**Yousik Lee**[1] **and Samuel Woo**[2]

[1]*ETAS Korea, Gyeonggi 13494, Republic of Korea*
[2]*Department of Software Science, Dankook University, Gyeonggi 16891, Republic of Korea*

Correspondence should be addressed to Samuel Woo; samuelwoo@dankook.ac.kr

Received 2 July 2022; Revised 12 August 2022; Accepted 24 August 2022; Published 26 September 2022

Academic Editor: Hao Peng



As automobiles become more electrified, more and more Electronic Control Units (ECU) are installed in vehicles. ECUs communicate with each other through dedicated protocols such as a controller area network (CAN), but these protocols do not have their own security measures. Many cyberattacks have exploited this weakness, but an intrusion detection system (IDS) is emerging as an effective countermeasure. In this study, we introduce a new attack method that existing IDS cannot detect. CAN signal extinction-based DoS attack (CEDA) is a new attack method that uses a voltage drop to erase the CAN signal. When the target ECU transmits a signal, adding a resister that lowers the differential voltage to an undefined gray zone causes the other ECU to ignore the signal being sent from the target ECU. In cybersecurity, denial of service (DoS) is defined as restricting an authorized entity from accessing a resource or delaying a time-critical system. This attack is a kind of a DoS attack since the adversary can make the target ECU bus-off through a CEDA. CEDA could be a serious problem as it has not been detected by any known IDS to date. In this study, we use laboratory and vehicle tests to detail the attack methods and introduce appropriate security measures.

## 1. Introduction

Modern vehicles are developing into huge information technology (IT) systems of software as the convergence of vehicles and information & communication technology (ICT), represented by connected cars and autonomous vehicles, becomes active [1, 2]. However, more software means more potential for cyberattacks on the vehicle [3–5]. After the first recall of vehicles due to a cyberattack in 2015, manufacturers began to equip security functions in their vehicles [6, 7]. And related organizations such as governments, associations, and societies have implemented standards, guidelines, and regulations related to automotive security.

One of the most notable security measures is the intrusion detection system (IDS) because it is effective against cyberattacks on vehicles, and many regulations recommend the installation of IDS [8–10]. Recently, artificial Intelligence and machine learning technologies have been actively introduced into the latest IDS research [11–13]. They will soon be adopted for automotive IDS as well. An electronic control units (ECUs) communicate with each other through an in-vehicle network using a protocol such as a controller area network (CAN). An application of the ECU generates data and sends it to the CAN controller. Then, the CAN controller hands the data to the CAN transceiver, and it transforms the data to an electrical signal and sends it to the CAN bus. Conversely, the CAN transceiver of the receiving ECU receives the signal to convert into logical bits, which the CAN controller further converts into a message that the application can recognize.

After that, an application reads the converted message. Figure 1 shows the architecture of the CAN compared with an open systems interconnection reference model (OSI) layer. Since most cyberattacks are performed in the application layer, an IDS is installed on the application layer. However, if the attack is conducted on the physical layer, IDS cannot detect it.

In this paper, we introduce CAN signal extinction-based DoS attack (CEDA) that erases messages by using a voltage drop by increasing the resistance. The differential voltage must be within the range defined by the standard so that the

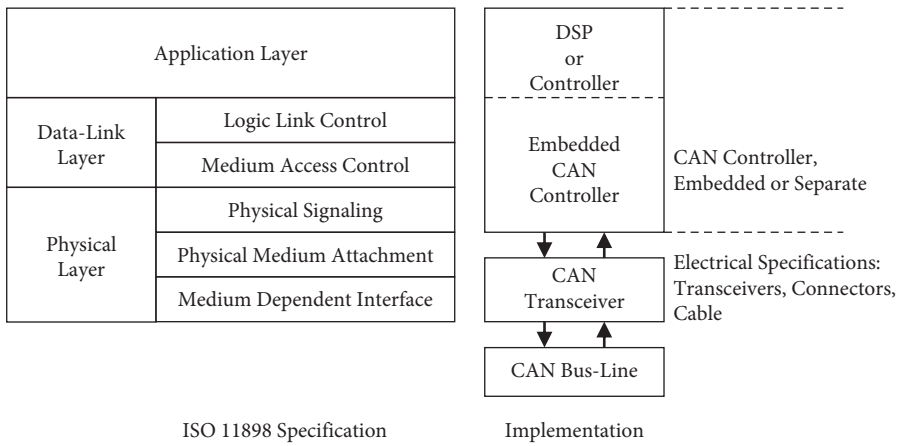



Figure 1: The layered ISO 11898 standard architecture [14].

receiving ECU can recognize the signal as a 0 or 1, but it can be made outside this range by simply adding a resistor. We propose to call the area outside the range defined in the standard a "gray zone." The gray zone is not defined in the standard, so other ECUs will ignore the signal if the differential voltage is in this zone. Therefore, if an adversary lowers the differential voltage to the gray zone by adding a resistor when the target message is transmitted, all ECUs in the in-vehicle network will ignore the signal. No existing IDS can detect the CEDA because:

(1) It is a signal-based attack, not a message-based attack, and it is ignored by the CAN transceiver, so no message is passed to the application layer where the IDS is installed.

(2) Attack device does not send messages and does not communicate with other ECUs.

In addition, since the attack device we proposed can be manufactured for less than 20 US dollars, this attack is quite realistic with the catastrophic consequences. We prove the proposed attack technique through the following three experiments.

(1) Feasibility check in the laboratory.

(2) Simulation test in the laboratory.

(3) Attack on a real vehicle.

This paper is structured as follows. "Background" reviews the background of the in-vehicle network architecture, CAN protocol, and related studies. We explain the mechanism of CEDA and attack model in "Proposed attack technique." We then describe the test on the laboratory and the real vehicle, and respective countermeasures in "Practical attack experiment." "Conclusion" concludes the paper and proposes future work area.

## 2. Background

*2.1. Communication Protocols for In-Vehicle Networks.* As vehicles evolve into connected cars and autonomous vehicles, more and more components are required to communicate with each other. Information is collected through sensors or other components and processed by the respective electronic control unit (ECU). The processed data is then transmitted to other ECUs. The ECUs that require data control the vehicle through an actuator or display the information on the devices. The ECUs that do not require data ignore the data. This is the reason why the in-vehicle network (IVN) is essential to the vehicle. Modern cars carry about 150 ECUs [15]. ECUs are classified into domains according to their functions or physical configurations, and communicate with each other via protocols such as CAN, CAN flexible data rate (CAN FD), local interconnect network (LIN), media oriented systems transport (MOST), FlexRay, and Ethernet. Figure 2 shows the traditional IVN architecture.

*2.2. Controller Area Network.* A CAN is a serial data communications bus developed by Robert Bosch GmbH for the vehicular embedded system in the early 1980s. CAN is a multi-master broadcast protocol based on sender IDs. It allows ECUs to communicate with data rates up to 1 Megabit per second. CAN is divided according to the communication speed into high-speed CAN and low-speed CAN. This paper provides all explanations based on the high-speed CAN. In the CAN bus system, each ECU uses a data frame to transfer information to other ECUs.

All ECUs are connected to each other through two dedicated wires. The wires are called CAN high (CANH) and CAN low (CANL). The CAN bus system must have bus Termination resistors 120 Ω at both endpoints of the physical network wires. The CAN Bus topology is shown in Figure 3(a).

ECUs generate a dominant bit (0) and a recessive bit (1) using a CAN transceiver to transmit the data frame. In the recessive state, both CANH and CANL are at the same level of 2.5 voltage potential (V), while CANH is at 3.5 V and CANL is at 1.5 V in the dominant state. The bit representation of the CAN transceiver is shown in Figure 3(b).

*2.3. Related Work.* Although CAN is the most widely used communication protocol for an in-vehicle network, it does not have its own security measures. For this reason, many attack techniques have been introduced since CAN was

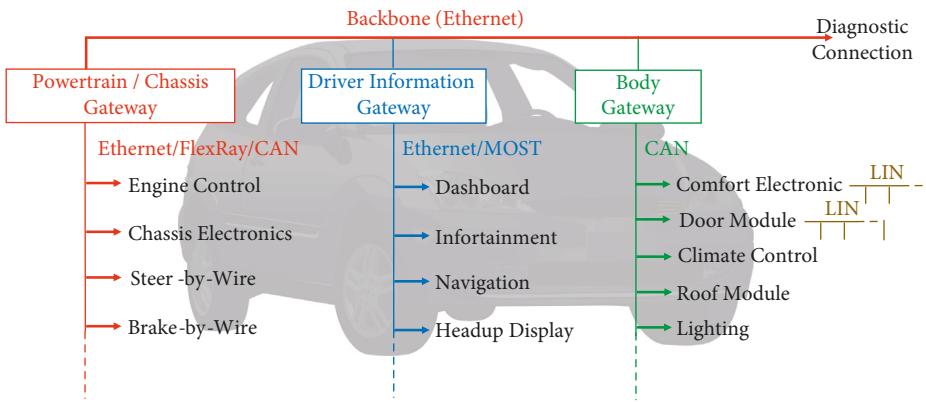



Figure 2: Traditional topology of in-vehicle network.

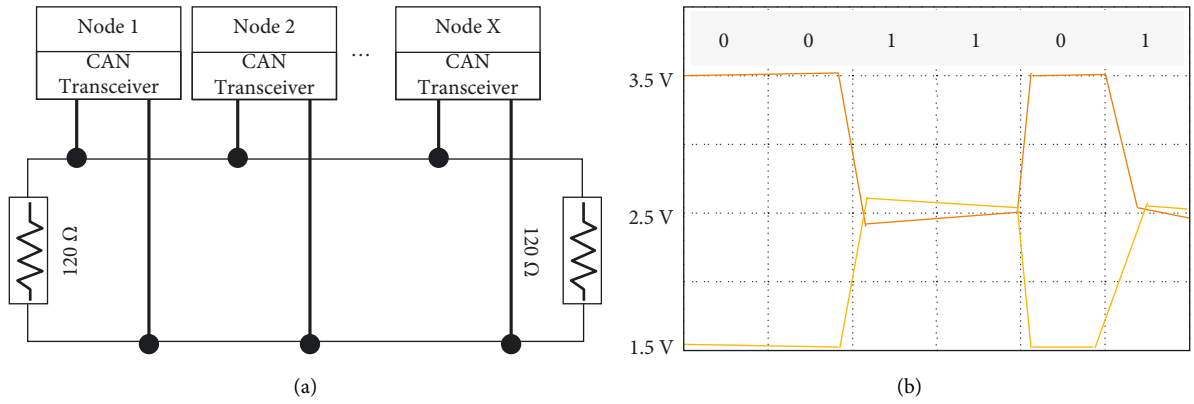



Figure 3: CAN BUS topology and nominal bus levels. (a) CAN BUS topology, (b) normal bus levels.

invented. In the early days of automotive cybersecurity research, system hackers from the traditional IT environment entered the automotive field, and there were a lot of SW-based attacks like in the IT environment. As cybersecurity became one of the important factors in the automotive industry, researchers have begun to use the characteristics of vehicles to expose their weaknesses, especially in-vehicle network protocols.

Miller and Valasek hacked a vehicle running on the highway with only a laptop and a smartphone [7]. They used the vulnerability of the head unit, which communicates with the outside to obtain administrator rights. Then, they replaced the firmware of the head unit with theirs and sent an attack message to the vehicle. As a result of this attack, the vehicle manufacturer recalled 1.4 million related vehicles, which was the first recall case due to a cyberattack [6]. The importance of the automotive cybersecurity increased due to this attack, and vehicle manufacturers began to implement countermeasures against cyberattacks on their vehicles. Government and related organizations started to enforce regulations, guidelines, and standards.

Palanca et al. proposed a new attack technique using a weakness of the CAN protocol [16]. In order to cause an error, they modulated a recessive bit into a dominant bit when a sender transmits a data frame. CAN is a carrier sense multiple access/collision detection (CSMA/CD) protocol, which means every node on the network can send a message. If two nodes start transmitting at the same time, the nodes will detect the collision and perform a nondestructive bitwise arbitration. If the attacker injects the dominant bit when the legitimate ECU sends the recessive bit, the recessive bit can be changed to the dominant bit.

Lee et al. introduced the app repackaging attack [17]. The researchers attacked the vehicle with OBD-II dongle and an app for operating it. The attack was made with a device that can be easily purchased in the market and downloaded apps from Google Play that can operate the device, which shows that the attack is realistic. They demonstrated unauthorized vehicle control such as opening a locked door and halting the engine. As countermeasures, they proposed obfuscation to prevent app tampering and message filtering to prevent receiving messages that control the vehicle from the outside.

To protect CAN-based network against cyberattacks, an intrusion detection system (IDS) was proposed [18]. An IDS is effective in detecting malicious messages since most messages in a CAN protocol have a fixed length and sending frequency. But as IDS has become more sophisticated, they are looking for ways to circumvent it. Attackers have begun to exploit software-based IDS by using the physical characteristics of the CAN protocol. Accordingly, IDS has also evolved to search for malicious messages using physical characteristics of the CAN protocol.

Cho and Shin proposed a mechanism for detecting an attack and identifying the specific ECU using clock skew that reflects the hardware characteristics of the clock source constituting the ECU [19]. Even if two ECUs transmit messages in the same period, they have different clock skew due to the characteristics of the hardware. The authors introduced a technique for detecting this clock skew as an attack if it fluctuates beyond a critical value while monitoring it. In addition, they proposed a voltage-based attacker identification (VIDEN), which is based on the characteristic of CAN signals transmitted by ECUs [20]. This characteristic is unique due to the difference in voltage supplied to each ECU, but it has limitations in mass-produced vehicles because an oscilloscope is required for the detection.

Sagong et al. introduced the hardware-based intrusion response system (IRS) [21]. They demonstrated vulnerabilities of the voltage-based IDS with three types of attacks:

(1) Overcurrent attack: supplying a current that exceeds the range the microcontroller can accommodate.

(2) Denial-of-service attack: letting CAN bus be in the idle state in a way that zeroes all signal and causes an error frame.

(3) Forced retransmission attack: forcing the ECU to send the message repeatedly.

An IRS is proposed to defend the attack which can circumvent the voltage-based IDS. In order for IDS to detect a malicious message or an attack device, it must receive a message or signal from the device. But, since the CAN signal extinction attack we proposed simply lowers the differential voltage of the signal transmitted from the target device, it is not detected by the existing IDS.

## 3. Proposed Attack Technique

*3.1. Attack Mechanism.* As described in section above, CAN has two logical states-a recessive state and a dominant state. In the recessive state, both CANH and CANL are at the same level of 2.5 voltage potential (V), while CANH is at 3.5 V and CANL is at 1.5 V in the dominant state [22]. The logical state of the bus can be determined by subtracting the voltage potential of CANH and CANL, which is called the differential voltage. However, since the differential voltage of each state can change according to various variables such as device characteristics, wire length and location, and vehicle driving conditions, it is not always possible to pinpoint 2.5 V and 0 V. Therefore, the CAN standard tolerates a certain amount of margin of error.

If the differential voltage is less than 0.5 V, the bus will be considered as the recessive state, and the bus will be regarded as the dominant state when the differential voltage is greater than 0.9 V.

However, if the differential voltage is between 0.5 V and 0.9 V, it is neither a dominant state nor a recessive state. In this case, the bus state is not defined according to the CAN standard [22]. It means that ECUs do not take any actions if they receive an undefined state. We propose to call this area the gray zone. Thus, if attackers can place the differential voltage in the gray zone when the target ECU sends a message, other ECUs ignore the message from the target ECU, and the target ECU generates an error frame. When attackers conduct this attack to a specific ECU distinguished by ID, the ECU continuously generates an error frame. And when the number of an error frame reaches the threshold, the ECU becomes a "bus-off" state and the ECU in the bus-off state cannot be operated normally. This is a DoS attack that can be conducted on the CAN-based in-vehicle network.

According to formula (1) and (2), the differential voltage is inversely proportional to the resistance, so increasing the resistance can decrease the differential voltage. Therefore, if an appropriate resistance can be calculated according to the in-vehicle network characteristics of the target vehicle and the corresponding resistor can be installed in the vehicle so that the differential voltage is located in the gray zone, the specific message of the vehicle can be erased. In other words, while monitoring messages in the CAN-based in-vehicle network, if a message with target ID appears, the resistance is increased so that the differential voltage is located in the gray zone. This can cause other ECUs to ignore the message and lead to disable certain functions.

*3.2. Attack Model.* The idea of the attack we propose comes from the structural architecture of the CAN-based in-vehicle network. This attack method is a kind of a DoS attack. The goal of a DoS attack is to make the target system unusable. We chose this method to remove the target system from the network instead of making the target system unavailable by sending a large amount of traffic to the system. When the ID of the target system appears while monitoring the CAN-based in-vehicle network, attackers make the message invalid by adjusting the voltage. We propose the following attack model and make a few assumptions that are required for the attack to be successful.

*3.2.1. Attacker's Ability.* Attackers can create the monitoring device and monitor messages in the CAN-based in-vehicle network. Based on this, attackers can find the CAN ID of the target function or device and add the resistance to prevent other ECUs from receiving messages from the target ECU. Attack that needs additional devices requires the attacker to equip the attack device to the target vehicle. Thus, it is assumed that the attackers can equip their device to the vehicle.

*3.2.2. Target Vehicle.* It is assumed that the in-vehicle network of the target vehicle includes a CAN. It is also assumed that the target vehicle is equipped with an ECU with the function that the attacker wants to exploit.

*3.2.3. Attack Model.* The attack method we propose can consider two attack models. The first attack model is the supply chain attack. The supply chain is very complex and layered. Most vehicle manufacturers cannot produce the cars by themselves and are provided parts, systems, and services

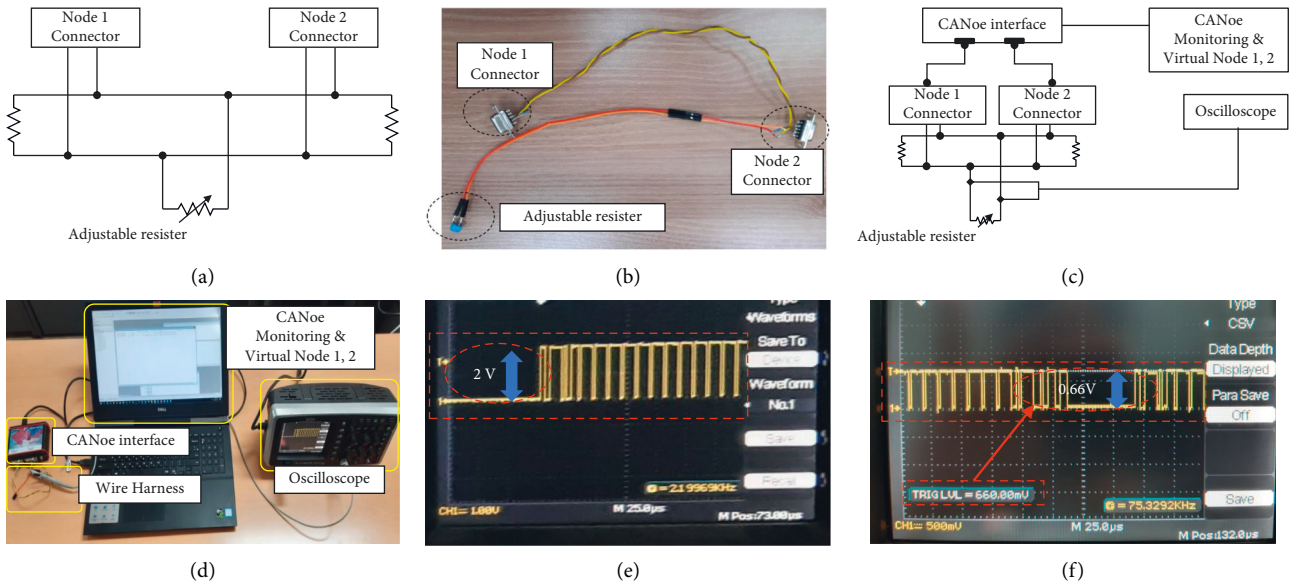



FIGURE 4: Experiment structures and environments for feasibility testing. (a) Block diagram of wire harness, (b) wire harness, (c) block diagram of testbed, (d) testbed, (e) normal state, (f) attack case.

from various suppliers. Suppliers also purchase parts and systems from other partners. Securing the entire supply chain is difficult because attackers can exploit any part of the complex supply chain. Attackers may add features or devices to enable our proposed attack method in certain parts of the supply chain, or even some vendors can be the attackers in this model. The second attack model is terrorism. Attack devices can be attached to a specific vehicle to compromise the safety of specific targets.

## 4. Practical Attack Experiment

In this chapter, we describe the attack experiment in laboratory environments and in a real vehicle. The following three experiments were conducted to prove our proposal.

(1) Feasibility test in a laboratory environment
(2) Attack simulation in laboratory environment
(3) Attack on a real vehicle

Finally, we describe the countermeasures against the attack we proposed.

*4.1. Feasibility Test.* In this section, we prove our idea through a simple device and facilities in a laboratory environment. We only need two nodes of CAN network for this attack. One node is a victim node that sends messages to another node. Since messages coming from the victim node will be erased by adjusting the resister, the contents of the messages are not important. Figure 4(a) shows the concept of a wire harness schematic, and Figure 4(b) shows the actual wire harness according to the schema in Figure 4(a). The nodes are created virtually on the laptop using CANoe that is an ECU simulation and test tool made by Vector Informatik GmbH, and they transmit the data through each node connector. Therefore, each node connector may be regarded as an individual node in order to simplify the system.

We assume that the "Node 1" is the victim system. Communicated messages can be monitored via a "Monitoring laptop" that is connected with a wire harness using a "CAN interface." Also, an "Oscilloscope" is used to check the voltage potential and the difference between them. We sent messages from "Node 1" to "Node 2" and monitored the transmitted messages to see whether "Node 2" could receive the messages. Figures 4(c) and 4(d) show the testbed environment. Without the proposed attack, the difference of the voltage potential is 2 V in the dominant state as shown in Figure 4(e). To simulate the attack, we gradually increased the resistance by controlling the adjustable register. The communication failed when 12.1 Ω was applied to the testing environment and the value of voltage potential was 0.66 V (Figure 4(f)), which means that the value was greater than 0.5 V and less than 0.9 V. Through this experiment, we proved that the CEDA is possible to attack an in-vehicle network. If attackers remove the signal regarding the brake system, the vehicle cannot slowdown, which could seriously endanger the safety of passengers and pedestrians.

*4.2. Attack Simulation in a Laboratory Environment.* In this section, we introduce a vehicle simulation test in a laboratory environment. The laboratory testing was performed at the automotive security living lab that was established by the Korea Internet & Security Agency (KISA) [23]. We confirmed in previous experiments that our idea is feasible. Our next step was to check whether the CEDA is possible in a vehicle simulator. In order to conduct the test, we need to consider the following procedures.

(1) Find the CAN ID of a target function
(2) Find out the voltage drop due to turn-on resistance of field effect transistor (FET) switches (3) Calculate

additional resistance to place the differential voltage between CANH and CANL in the gray zone so that the CAN signals dissipate

(3) Add the resistance calculated above to the CAN BUS

To accomplish the 4th step, we developed new device that can add a resistor programmable.

*4.2.1. Reversing to Find CAN IDs.* As described in "Attack Mechanism," we perform an attack that removes data when a specific message appears on the in-vehicle network. We selected a function that controls the motor-driven power steering (MDPS). In order to attack, we need to identify the CAN ID of the related message. In general, CAN specification includes CAN IDs and the data frame structures is one of the intellectual properties of respective vehicle manufacturers. Thus, we should reverse engineer the data frame structure to find the CAN ID of the message. The process is as follows:

(1) To monitor messages on the in-vehicle network, connect the monitoring tool to the vehicle. CANoe TM of Vector was used in this study.

(2) After turning on the ignition, leave the vehicle alone for a while so that the vehicle is in a stable state.

(3) A stable state means the ECUs in the vehicle send the same value or send a repeating predictable value periodically.

(4) Look for the message whose value changes significantly by manipulating the handle.

(5) Fixing the ID of the found message in the monitoring tool, and verify the CAN ID by checking the values when the steering wheel is being operated and not.

The CAN ID of the MDPS control message analyzed by the above process is shown in the following Table 1. Since the target messages are removed, we do not analyze the contents of the message.

*4.2.2. Calculating the Voltage Drop due to Turn-On Resistance of FET Switches.* To calculate the proper resistance to place the differential voltage in the gray zone, we must calculate the voltage drop due to turn-on resistance of FET switches. In order to make this calculation, we also need to know the structure of the CAN transceiver. Figure 5(a) shows the structure of the transceiver. According to Ohm's law, we can calculate the differential voltage between CANH and CANL using the following formula: formula,

$$V_{\text{diff}} = \frac{R_r}{R_r + R_1 + R_2} * V_{in}, \tag{1}$$

where, $R_{\text{r}}$ = resultant resistance. $R_1$, $R_2$ = voltage drop due to turn-on resistance in FET switches, (B) in Figure 5(a). $V_{\text{diff}}$ = a differential voltage between CANH and CANL. $V_{\text{in}}$ = input voltage, (A) in Figure 5(a).

In the case of in-vehicle network, a terminating resistance is 120 Ω in general [22].

Table 1: CAN ID of MDPS functions found by reversing.

| CAN ID | Description |
|---|---|
| 0x381 | MDPS (Motor-driven power steering) |

Thus, a resultant resistance can be calculated by the formula.

$$R_r = \frac{1}{1/R_a + 1/R_b}, \tag{2}$$

where, $R_a$, $R_b$ = resistances which are in parallel connection in a circuit, in this experiments 120 Ω.

A terminating resistance $R_r$ is 60 Ω in the normal state (not the attacked state), and can be changed if an attacker puts additional resistance. We calculate the input voltage ($V_{\text{in}}$) to be 3.3 V through the data sheet of VP230 transceiver which is used in this study [24]. The differential voltage between CANH and CANL ($V_{\text{diff}}$) can be checked using an oscilloscope at the automotive security living lab; and we calculated the value to be 2.44 V as shown in Figure 5(b). Now, we can calculate $R_1 + R_2$ and the value is 21.15 Ω. As you can see in Figure 5(a), $R_1$ , $R_2$ are only affected by the input voltage. Since the input voltage is a constant value fixed at 3.3 V in this study, it is meaningless to figure out each value.

*4.2.3. Calculating Additional Resistance to Attack the Vehicle.* Again, our goal is to place the differential voltage ($V_{\text{diff}}$) between 0.5 V and 0.9 V so that other ECUs cannot recognize the message from the target ECU. To do this, we must calculate the additional resistance using Formula (1).

Since $V_{\text{diff}}$, $V_{\text{in}}$, and $R_1 + R_2$ are known values, we can calculate $R_r$. And the resistance we want to know can be calculated from $R_r$ using Formula (2). The calculated additional resistance that places the differential voltage in the gray zone was $3.55\ \Omega \le R_{\text{b}} \le 7.00\ \Omega$. Thus, we chose 6 Ω as the additional resistance. To attack the CAN-based network, we developed a device as shown in Figure 5. Figure 5(d) shows the overall appearance of the KISA's automotive security living lab, and Figure 5(e) shows how to connect the vehicle simulator and the device. Figure 5(c) is the schematic of the device structure.

The device consists of an additional resistance to attack the target and a field programmable gate array (FPGA), which gives the additional resistance to the network if the received ID is the target ID. Details of each part are shown in Table 2.

When the FPGA receives signals through the CAN transceiver, it checks whether the received ID is the target ID or not. If the received ID is the target ID, the FPGA approves the attack resistance to the network by turning on the switch.

Figure 6(a) shows the screen capture of the oscilloscope after an attack, and Figure 6(b) is the chart used to find the exact value. To check the exact value, we downloaded the data from the oscilloscope and drew a chart with time and voltage. As you can see Figure 6(b), if 22 Ω resistance is added to the network, the differential voltage is 0.72 V, which is in the range of the gray area. This means that the

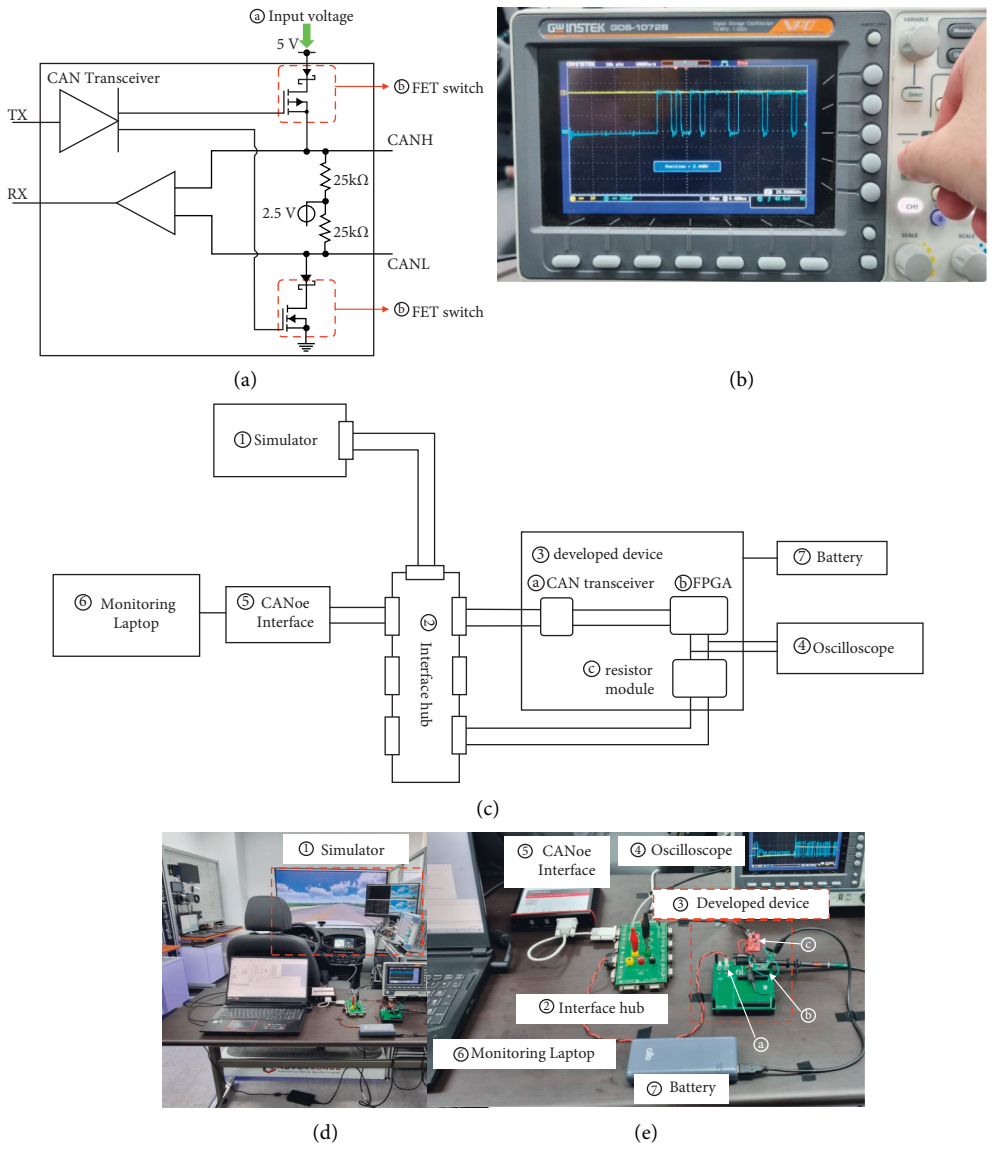


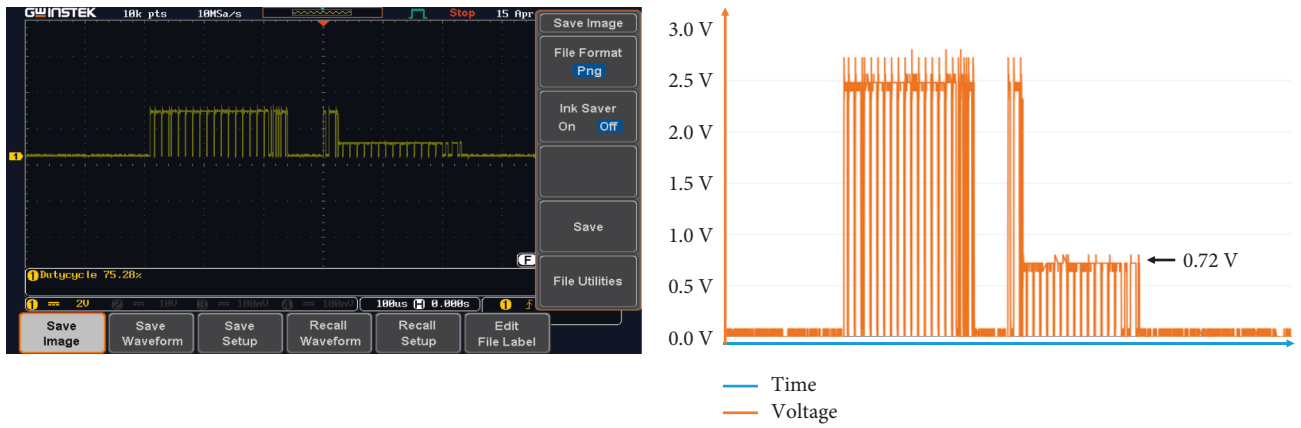

Figure 5: Experiment structures and environments for laboratory simulation testing. (a) CAN transceiver structure, (b) The differential voltage between CANH and CANL, (c) Detailed structure of the device, (d) Overall appearance of the lab, (e) The Developed device.

Table 2: Tools used for the attack experiment.

| Tool | Product info |
|---|---|
| Adjustable resister | P080 3590S |
| Oscilloscope | GDS-1072B, 70 MHz |
| CAN interface | CANcaseXL CANoe to CAN BUS |
| Monitoring and data transmission | CANoe (for CAN BUS) |
| Simulator | Automotive security living lab [23] |
| Developed device | CAN transceiver resistor module FPGA |
| CAN transceiver | SN65HVD230 TEXAS Instruments |
| Resistor module | 2N3904 |
| FPGA | TinyFPGA AX2 |
| Interface hub | CAN BUS terminal |
| Vehicle | Midsize car (2021 model) |

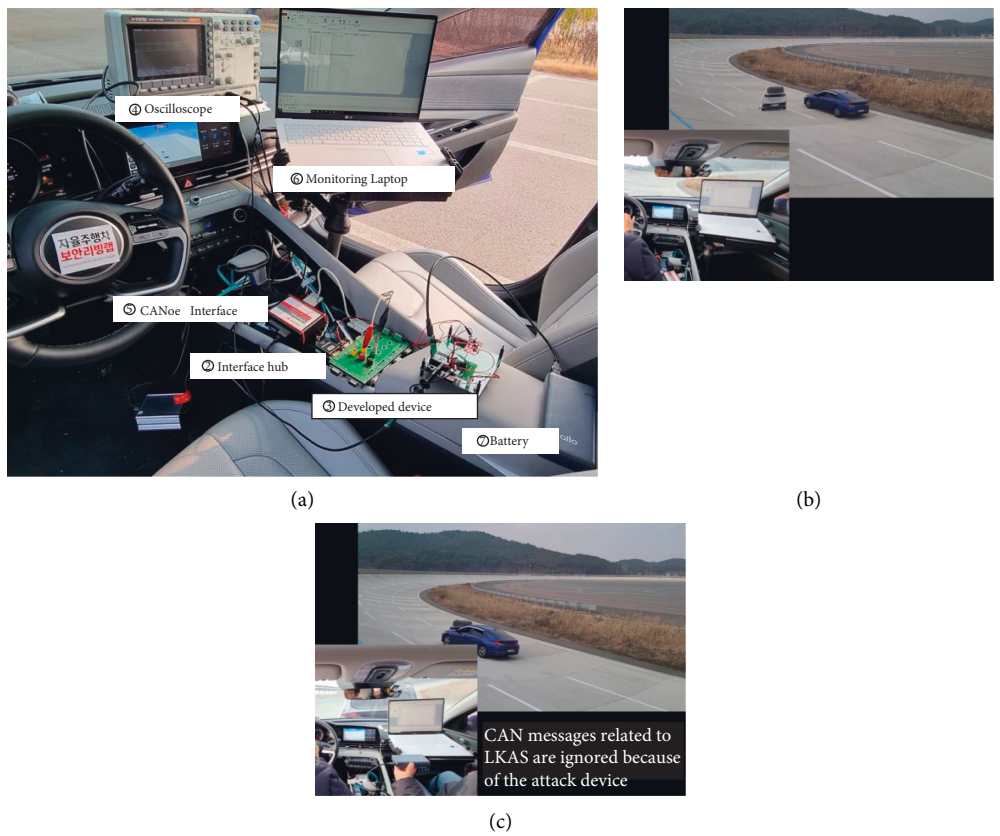
MDPS function is invalidated, and we confirmed that a lane keeping assist system (LKAS) does not work even though it is activated through the simulator.

*4.3. Attack on Real Vehicle.* In the "Attack simulation in a laboratory environment" section, we showed that the proposed attack is feasible in a simulator that simulates a real vehicle. In this section, we show that the attack we proposed is possible in a real-life setting and therefore dangerous. We applied the device developed in the laboratory environment to a real vehicle. Hyundai Avante (Code name CN7) was used for this experiment. According to the manufacturer, the vehicle has a LKAS named Lane Maintenance Assist function that helps keep the vehicle within the chosen driving lane [25]. The vehicle was supported by KISA living lab [23]. This attack we proposed does not use the weakness of the specific vehicle. It will affect all




(a) (b)

Figure 6: Experimental result of laboratory simulation testing. (a) oscilloscope view, (b) detailed waveform created based on recorded data.




(a) (b)

(c)

Figure 7: Attack on real vehicle. (a) overall view of the vehicle with attack device installed, (b) normal driving, (c) driving under attack.

vehicles that use the CAN-based in-vehicle network. We installed our device in the vehicle (Figure 7(a)). Then, we compared driving in normal and attack situations and recorded the video [26]. The video data used to support the findings of this study have been deposited in the GitHub repository (https://github.com/team-aegis/ceda). As you can see in the video, the first part is driving under normal conditions with the driver's hands off the steering wheel, and the vehicle is driving well between the lanes with the LKAS (Figure 7(b)). In the second part, after the attack starts (by connecting the battery and supplying power), the LKAS related message is not recognized in the in-vehicle network. The vehicle ignores the lane and collides with another vehicle driving in the next lane (Figure 7(c)).

*4.4. Countermeasures.* As discussed in section 3. B, CEDA can be realized through a supply chain attack or terrorism. In the case of a supply chain attack, MITRE tries to address it by generating a catalog of attack patterns that provides a structure for maturing aspects of supply chain risk

management [27]. Potential countermeasures against supply chain attacks are a good illustration of the catalog. The attack with ID CM-2 is named Prevent or Detect Critical Component Tempering, and the mitigation approach is to prevent or detect tampering with critical hardware or firmware components while in transit, across all lifecycle phases, through use of state-of-the-art anti-tamper devices [27]. In addition, the attack with ID CM-11 is named Multiple Suppliers, and the mitigation approach is Use multiple suppliers for key critical components [27]. As you can see examples in the catalog, almost all countermeasures are managerial measures rather than technical ones. The United Nations (UN) recently enacted a regulation related to vehicle security, UN Regulation No.155, and ISO/SAE 21434 supported the regulation [28, 29]. This regulation consists of two certification programs—Cyber security management system (CSMS) and Vehicle type approval (VTA). The CSMS is a regulation for the security governance and all countries under the 1958 agreement of the UN must enact and follow the relevant laws. You can see why the regulation focuses on supply chain management through security governance, which is consistent with the countermeasures against supply chain attacks proposed by MITRE, are consistent.

In the case of terrorism, the attack is much more difficult to detect. Since the attack device we developed just inspects the received message and increases the resistance in the network, it cannot be detected by a function such as component identification [30]. The IDS also cannot detect it because the device does not send the message to the other ECUs. Therefore, a practical countermeasure is to continuously monitor the voltage in the network and notify the IDS when a voltage is in the gray zone. In this case, a false positive must be considered and additional investigation is needed.

## 5. Conclusion

In this study, we have shown that it is easy to attack a CAN-based in-vehicle by controlling the resistance and eliminating specific and/or whole messages on the network. As described in Table 2, the attack device can be manufactured at a low cost of less than 20 US dollars. Also since this attack uses the weak point of the CAN-based network protocol, it is hard to detect. It means the attack we proposed can have a huge ripple effect in the real world. Therefore, to protect vehicles from this kind of attack, we need to consider designs based on "security by design" and "defense in depth" and carefully select the security features through security risk assessment [31]. In addition, we need to consider the supply chain management that is required by UN regulation No. 155 and ISO/SAE 21434 to mitigate the risk that comes from supply chain attacks. Furthermore, we believe the monitoring resistance of the network is an appropriate countermeasure against the CAN signal extinction-based DoS attack.

In the future, we will study this attack as an intrusion protection system (IPS). If the IDS can perfectly detect the attack message, the attack message can be completely removed using the CAN signal extinction mechanism we proposed. In the case of a firewall, only the ECUs are located.

## Data Availability

The video data used to support the findings of this study have been deposited in the GitHub repository (https://github.com/team-aegis/ceda).

## Conflicts of Interest

The authors declare that they have no conflicts of interest.

## Acknowledgments

The present research was supported by the research fund of Dankook University in 2019.

## References

[1] A. Saad and U. Weinmann, "Automotive software engineering and concepts," *GI. Jahrestagung*, vol. 34, pp. 318-319, 2003.

[2] M. Arfizurrahmanl, M. S. H. Ahmad, M. S. Hossain, M. A. Haque, and K. Andersson, "Real-time non-intrusive driver fatigue detection system using belief rule-based expert system," *J. Internet Serv. Inf. Secur.*, pp. 44–60, 2021.

[3] K. Koscher, A. Czeskis, F. Roesner et al., "Experimental security analysis of a modern automobile," in *Proceeding of the. 2010 IEEE Symposium on Security and Privacy*, pp. 447–462, IEEE, Oakland, CA, USA United Syate of America, May 2010.

[4] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, and S. Savage, "Comprehensive experimental analyses of automotive attack surfaces," in *Proceeding of the. 20th USENIX Security Symposium*, San Francisco, CA, August 2011.

[5] G. Lacava, A. Marotta, F. Martinelli et al., "Cybsersecurity issues in robotics," in *J. Wirel. Mob. Networks Ubiquitous Comput. Dependable Appl*Springer, Berlin/Heidelberg, Germany, 2022.

[6] FCA US LLC, "safety recall R40/NHTSA 15V-461 radio security vulnerability," 2015, https://static.nhtsa.gov/odi/rcl/2015/RCRIT-15V461-7681.pdf.

[7] C. Miller and C. Valasek, "Remote exploitation of an unaltered passenger vehicle," in *Proceeding of the. Black Hat USA*, Black Hat USA, United state of america, August 2015.

[8] H.R.3711, "safely ensuring lives future deployment and research in vehicle evolution act," 2019, https://www.congress.gov/bill/117th-congress/house-bill/3711.

[9] S.2182, "security and privacy in your car act of 2019," 2019.

[10] M. Komisarek, M. Pawlicki, R. Kozik, and M. Choras, "Machine learning based approach to anomaly and cyber-attack detection in streamed network traffic data," *J. Wirel. Mob. Networks Ubiquitous Comput. Dependable Appl*, pp. 3–19, 2021.

[11] M. Elshrkawey, M. Alalfi, and H. Al-Mahdi, "An enhanced intrusion detection system based on multi-layer feature reduction for probe and DoS attacks," *J. Internet Serv. Inf. Secur.*, pp. 61–78, 2021.

[12] P. Nowakowski, P. Zórawski, K. Cabaj, and W. Mazurczyk, "Detecting network covert channels using machine learning, data mining and hierarchical organisation of frequent sets,"

*Mob. Networks Ubiquitous Comput. Dependable Appl*, vol. 12, pp. 20–43, 2021.

[13] D. Bae and J. Ha, "Performance metric for differential deep learning analysis," *J. Internet Serv. Inf. Secur.*, pp. 22–33, 2021.

[14] S. C. Hpl, "Introduction to the Controller Area Network (CAN)," *Application Report SLOA101*, pp. 1–17, 2002.

[15] J. Deichmann, B. Klein, G. Scherf, and R. Stützle, "The Race for Cybersecurity: Protecting the Connected Car in the Era of New Regulation," 2019, https://mck.co/2xcXm4G.

[16] A. Palanca, E. Evenchick, F. Maggi, and S. Zanero, "A stealth, selective, link-layer denial-of-service attack against automotive networks, Detection of Intrusions and Malware, and Vulnerability Assessment," in *Proceeding of the. International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, pp. 185–206, Bonn, Germany, June 2017.

[17] Y. Lee, S. Woo, J. Lee, Y. Song, H. Moon, and D. Lee, "Enhanced Android app-repackaging attack on in-vehicle network," *Wireless Communications and Mobile Computing*, pp. 185–206, 2019.

[18] P. S. Groza and B. Groza, "Source identification using signal characteristics in controller area networks," *IEEE Signal Processing Letters*, vol. 21, no. 4, pp. 395–399, 2014.

[19] K. T. Cho and K. G. Shin, "Fingerprinting electronic control units for vehicle intrusion detection," in *Proceeding of the 25th USENIX Security Symposium*, pp. 911–927, ktcho, kgshin, August 2016.

[20] K. T. Cho and K. G. Shin, "VIDEN: attacker identification on in-vehicle networks," in *Proceeding of the. 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1109–1123, New York, NY, United States, October 2017.

[21] S. U. Sagong, X. Ying, R. Poovendran, and L. Bushnell, "Exploring attack surfaces of voltage-based intrusion detection systems in controller area networks," *SAVE Proceedings. 2018 ESCAR Europe*, pp. 1–13, 2018.

[22] ISO, *ISO11898-2:2016, Road Vehicles - Controller Area Network (CAN) - Part 2: High-Speed Medium Access Unit*, International Organization for Standardization, Geneva, Switzerland, 2003.

[23] KISA, "the automotive security living lab," https://www.kisa.or.kr/1040404, 2020.

[24] TI, VP230 datasheet - 3.3-v CAN transceivers, 2022, https://www.digchip.com/datasheets/parts/datasheet/477/vp230.php.

[25] Hyundai, "avante catalog," 2022, https://www.hyundai.com/kr/en/sedan/avante/20fc/price.

[26] Team-AEGIS, "CEDA: can signal extinction-based DoS attack," 2022, https://github.com/team-aegis/ceda.

[27] J. F. Miller, *Supply Chain Attack Framework and Attack Patterns*, MITRE CORP MCLEAN VA, Colshire Dr, McLean, VA 22102, USA, 2013.

[28] UN Regulation, No.155, *Uniform Provisions Concerning the Approval of Vehicles with Regards to Cyber Security and Cyber Security Management System*, UN, New York, 2021.

[29] ISO, *Road Vehicles — Cybersecurity Engineering*, International Organization for Standardization, Geneva, 2021.

[30] A Weimerskirch, C Paar, and M Wolf, "Cryptographic component identification: enabler for secure vehicles," in *Proceeding of the. IEEE Vehicular Technology Conference*, vol. 62, no. 2, p. 1227, September 2005.

[31] S. R. Ronald, M. Michael, and C. O. Janet, "Systems Security Engineering: Considerations for a Multidisciplinary Approach in the Engineering of Trustworthy Secure Systems," *NIST Special Publication*, pp. 800–160, 2016.