BlockFW - Towards Blockchain-based Rule-Sharing Firewall

Wei-Yang Chiu and Weizhi Meng
SPTAGE Lab, Department of Applied Mathematics and Computer Science,
Technical University of Denmark, Denmark
Email: {weich, weme}@dtu.dk

***Abstract*—Central-managed security mechanisms are often utilized in many organizations, but such server is also a security breaking point. This is because the server has the authority for all nodes that share the security protection. Hence if the attackers successfully tamper the server, the organization will be in trouble. Also, the settings and policies saved on the server are usually not cryptographically secured and ensured with hash. Thus, changing the settings from alternative way is feasible, without causing the security solution to raise any alarms. To mitigate these issues, in this work, we develop BlockFW – a blockchain-based rule sharing firewall to create a managed security mechanism, which provides validation and monitoring from multiple nodes. For BlockFW, all occurred transactions are cryptographically protected to ensure its integrity, making tampering attempts in utmost challenging for attackers. In the evaluation, we explore the performance of BlockFW under several adversarial conditions and demonstrate its effectiveness.**

***Index Terms*—Network security, Firewall, Blockchain technology, Intrusion detection, Consensus algorithm**

I. Introduction

It is difficult to overlook security policies over large networks for network administrators. When attacks occurred from either internal or external network, it can be quite challenging for them to quickly take measures and deploy new policies [3], [16]. For example, performing penetration test toward multiple servers in a network can be quite simple [18], such as setting up scripts for automating the attack. However, it is quite an opposite situation for network administrators, since collecting information and deploying security solutions need to be done one-by-one. This is very time-consuming and labor-costly compared to performing an attack. To overcome this unfair situation, commercialized central-managed security solutions are provided by many security providers. These products give administrators a dashboard or a cockpit, making it easier to overview situations in the network. That is, information can be collected, and policies can be deployed at one-stop.

However, what these solutions are offering can also become a security breaking point of the system [12]. All endpoints, by default, must trust the decision and command coming from the central server of the security solution. If the management server is compromised, it can become a huge loophole of the security status in an organization [20]. For example, attackers can command all security solutions deactivated in order to reveal further exploits of the internal network.

Fig. 1 shows an example of a central-managed security solution with its settings stored in a mutable database. We

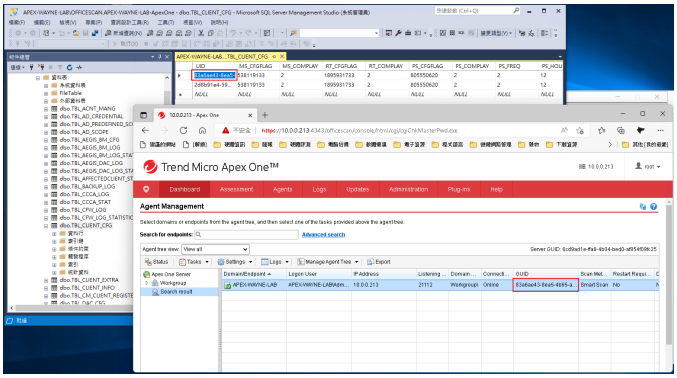

Fig. 1. A Centralized Security Solution with Database in Mutable Storage

Fig. 2. Security Solution Vulnerability

can perform some value changes, not through the security solution's management console, but through the database console. Then we notice the existence of toolkit that can directly access the offline database file, without any restrictions from the configured database management system. Although the attackers could not obtain the management console's access credential, they can change the security solution's settings through several alternative methods, which can be considered as unauthorized changes for the security solution.

In this case, although attackers may not be able to find the exploit to the security solution itself, they can still affect the security policies via different vulnerabilities on the server that holds the centralized management of the security solutions, as shown in Fig. 2.

The situation creates the need of having a second pair of eyes to closely monitor the management server itself, making organizations with centralized security solutions more insecure. However, as we closely inspect the example case we

are studying, we can see that the issue itself is more related to the underlying database. That is, changes of a security policy can be made through alternative routes that are outside the designed workflow. While other form of validation and monitoring from others does not exist.

Motivation. As blockchain becomes a constantly discussed topic recently, several of its characters can tackle the issues of central-managed security solutions [9], [13]. They are the immutability of occurred events, and evidence of transaction events is cryptographically strengthened so that data integrity will become extremely challenging to compromise, and the underlying consensus algorithm will be able to follow one version of the data with their recognition. Further, blockchain requires its participants to hold a partial or full copy of the network transaction log, called *ledger*. Transactions are collected and validated by network maintainers, such characters or equivalent may have different names in different platforms, before being cryptographically sealed into a basic storage unit, named *block*. Generated block contains the cryptographical proof (e.g., hashes) of the previous blocks. This creates a strengthened chain-like storage structure, which is challenging to break [6], [7]. For attackers that would like to alter the previously existed records, it will be extremely time-taking, making such operation infeasible.

If attackers deliberately change the database records by editing it forcefully, it will result in either the node being ditched out of the network due to tremendous differences, or the tampered database records will be restored from other nodes [17]. Both situations are not favorable to the attackers.

Contributions. In this paper, our main goal is to deploy a proof-of-concept of centralized security management on top of blockchain, in order to showcase the feasibility and resilience of such system under cyber-attacks. In particular, we develop BlockFW – a blockchain-based rule-sharing firewall, and investigate its performance under adversarial conditions. The results indicate its capability of lowering the cost of operating a security solution.

The rest of this paper is organized as follows. Section II introduces the background and related work. Section III details the design of BlockFW including the requirements and major components. Section IV presents the performance evaluation under some adversarial scenarios. Section V concludes the work with future work..

II. Background and Related Work

A. Blockchain

Blockchain, by its design and practice, is considered as a kind of decentralized ledger technology (DLT) [7], [9]. A block is the basic storing unit in the blockchain, which can be formed in a periodic way including the collected transactions within a time period. A consensus algorithm is applied in the network to allow everyone validating the blocks and to reach an agreement on the block version. Basically, consensus algorithm will select a sealer to seal the latest formed block with strong cryptography. The block is then distributed to all network participants for updating their local copies.

To ensure the unification of the decentralized database is the primary designing goal of a consensus algorithm. Below are two typical algorithms.

a) Proof-of-Work (PoW): A PoW-based system will generate a challenging computational problem, in which a difficulty control mechanism is involved. The level of difficulty can be adjusted according to the system's requirements. The participant who first solves the problem will win the turn.

Being the first consensus algorithm in Bitcoin [23] with the easy-to-understand design philosophy, PoW indeed dominates the market of cryptocurrencies. However, with the network participants increasing, many new challenges can be caused, i.e., the tremendous waste of computational power on completing transactions. Profitable mining activities may encourage the forming of mining pools. The concentration of computing power leads to the threat of 51% attack [22]. That is, when a particular group owns 51% or more computational power of the whole network, it has unsurpassed domination on manipulating future records [21].

b) Proof-of-Stake (PoS): As a possible solution to complement PoW consensus algorithm, PoS chooses sealers by rounds of selection rather than computing competitions. More specifically, PoS asks participants to take some of their assets (or coins) to join the election. The system chooses the preferable stake by conditions. The selected stake's owner wins the turn [15]. The criteria of how the system decides the preferable stake is crucial. For example, setting the criteria as preferring a larger stake may cause monopoly. For this issue, *coin-age* that measures a coin's stagnation in an account is considered as a promising solution [4].

PoS provides a more power-efficient method of reaching consensus and providing more fairness of sealer selection toward the participant with less computational power. However, it does not prevent the 51% attack. Though PoS does not suffer from the monopoly of computational power, it may suffer from the monopoly of wealth. As opposite to 51% of computational power, 51% of the wealth can provide unsurpassed advantages on winning the stake [5].

B. Related Work

The application of blockchain technology in developing a firewall is not new. In the literature, Steichen *et al.* [19] introduced ChainGuard, which could use SDN functionalities to filter network traffic for blockchain-based applications. Their system required that all traffic to the blockchain nodes has to be forwarded by at least one of the switches controlled by ChainGuard. Li *et al.* [11] then developed a blockchain-based filtration mechanism (similar to firewall) with collaborative intrusion detection to help protect the security of IoT networks by refining unexpected events. It is found that though some ideas have been proposed on blockchain-based firewall, they have not been widely implemented. This motivates our work to implement a prototype of blockchain-based firewall and examine its performance in a practical setup.

Many research studies are focusing on the combination of blockchain technology with intrusion detection. For instance,

Meng *et al.* [14] designed a blockchain-based approach to help enhance the robustness of challenge-based intrusion detection against advanced insider attacks, where a trusted node may suddenly become malicious. Li *et al.* [9] introduced BlockCS-DN, a framework of blockchain-based collaborative intrusion detection for Software Defined Networking (SDN). A similar scheme was also proposed by Meng *et al.* [13], which used blockchain to enhance the robustness of trust management. Some more relevant studies can refer to surveys [2], [9], [10].

III. BlockFW - A Blockchain-based Rule-Sharing Firewall

This section introduces how our proposed blockchain-based rule-sharing firewall works. At first, we briefly describe how to choose and decide a blockchain platform for our case. Then we present the high-level architecture of our system including the major software components.

A. The Requirement for underlying Blockchain Platform

Although different blockchain platforms share similar concepts, the underlying implementation differences provide the platforms with various advantages separately. Not all platforms can become the data storage of our system. For our purposes and goals, we consider a suitable platform that should have the following characteristics:

- `Semi-Dynamic Network`: Servers may be added or removed according to the changes or expands in services. In the trend of X-as-a-Service, cloud, and virtualization, the action of adding or removing service entities can be dynamic. Though being dynamic, there are differences from the public network: authentication is mandatory. Nodes in the network cannot join or leave the network autonomously, authorization entity or authorized personnel must get involved and approve the operation. This specific characteristic creates a semi-dynamic all-known-nodes network. Furthermore, since all network nodes are responsible toward different tasks and may potentially be vulnerable in different ways, we have to assume that part of the network may become malicious. Hence the network we are trying to deploy must be Byzantine-resistible.
- `Stable Connection`: Since servers are regarded as critical infrastructure in IT-enabled businesses, they are usually either connected through the internal network, or the connections can be ensured by telecom SLA with the company. Compared with the wide area network, it has less flickering or instability issues. We consider that it can accept having a blockchain-platform with higher counts of exchanged messages during communication.
- `Timing-Sensitive`: When attacks occurred, we definitely expect that the traffic can be blocked as soon as possible when being a network administrator. However, even deploying security policies through many centralized security solutions may take a while to reach every client. Although it is unreasonable to have everything responded at instant, the actions have been taken will reach and execute by clients eventually. While the time consumption should be in a reasonable length from the command being given to the action being taken. Thus a blockchain system that completes transactions in an estimable time is important.

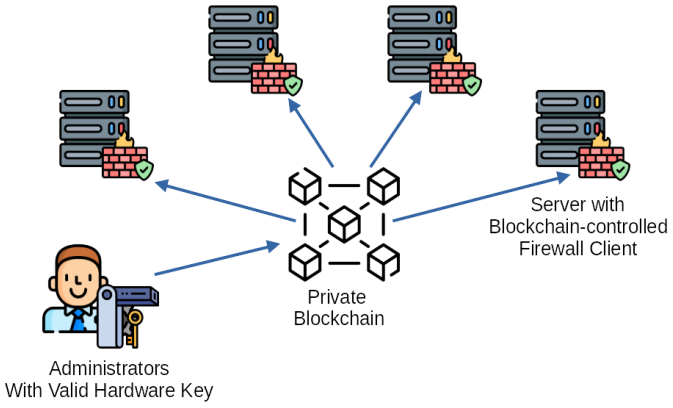

Fig. 3. The Overview Structure of BlockFW

Based on the above characteristics, we figure out that our BlockFW platform needs to be Byzantine tolerable with stable transaction speed, in which these requirements are usually satisfied in a private blockchain.

In this work, we decide to implement the system based upon the DevLeChain platform[1] – a blockchain development environment, which can be used to quickly and easily set up a desired environment [8]. In addition, it supports multiple different blockchain platforms. Hence, we can easily switch between platforms to observe the differences.

B. The System Overview

As shown in Fig. 3, BlockFW features a simple and straightforward system structure, which consists of two major roles and three major pieces of software.

The two roles are:

- `Administrators`: They have the permission to set and alter firewall rules to the system. Each administrator will be given a hardware key that has been registered into the system. Existing administrators can set other keys as administrators. The hardware key is regarded as the wallet file of the administrator when interacting with the blockchain.
- `Clients`: These are endpoints that listen and monitor the given rules on the blockchain. They are installed with firewall software, which can act according to the rules on the blockchain.

The three major software components are:

- `Management Console`: The console is a command-line interface for administrators to add new firewall rules or manage existing firewall rules, as depicted in Fig. 4. It requires the administrator's hardware key to function correctly. If a non-registered hardware key is provided, any command given to the management console will fail.

[1]https://devlechain.compute.dtu.dk/.

```
Welcome to BlockFW Interactive Console
Firewall Rules is configured at : 0xc2fcb155aee59030ba74..
#> help
    port <port number> <block | unblock | unmonitor>
    list
    exit
#> list
    Port : 22       unblocked
    Port : 23       blocked
    Port : 80       blocked
    Port : 30303    unblocked
#>
```

Fig. 4. The BlockFW Management Console

```
Refresh
    Port 22 remains the same
    Port 23 blocked
    Port 80 blocked
    Port 30303 new monitored

Refresh
    Port 22 remains the same
    Port 23 remains the same
    Port 80 remains the same
    Port 30303 remains the same
```

Fig. 5. The BlockFW Management Console

This is because the system's backend smart contract is enforced with Access Control List, which contains the public-key-derived wallet addresses. Any non-registered key will result in transactions that are unacceptable to the smart contract, as it cannot be validated.

- `Firewall-Commander`: The firewall-commander is the middleware between the blockchain and the system. It monitors the blockchain for any changes periodically. If the current firewall state is different from what the blockchain has stated, it will synchronize the rules in local system firewall, as shown in Fig. 5.
- `Blockchain`: The blockchain is acted as the decentralized database among clients and administrators.

IV. Performance Evaluation

In this section, we present the environmental setup and evaluate the system under different adversarial scenarios.

A. System Configuration

To test how effective the proposed system is, we configured **three nodes** with client installed, **one administrator node** with hardware key, and **one attacking node** toward the network. Each node is given and configured with the information listed in Table I.

For concise and clear demonstration, we set up all entities under the same network, as illustrated in Fig. 6. Servers are

TABLE I
Environmental Platfrom

VM Reousrces		Software	
Item	Config.	Item	Version
CPU	Intel Xeon W-2133 @ 3.6GHz x2	Hypervisor	vmware ESXi 7.0 U3d
Memory	4GB ECC DDR4-2666	Guest OS	mxLinux 21
Storage	48GB HDD	Blockchain Platform	Ethereum 1.10.18
Network	vmware vSwitch 1G	Contract Platform	EVM

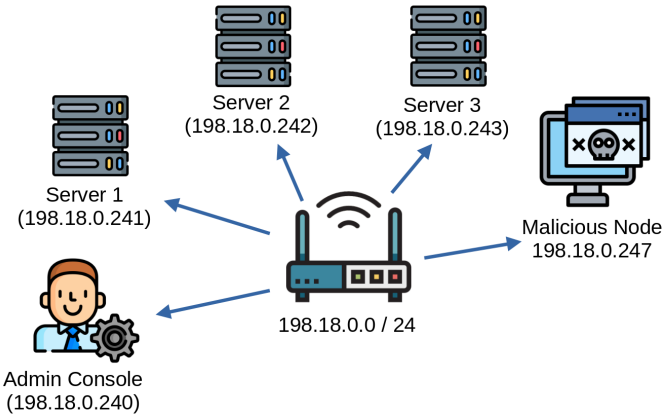

Fig. 6. The network configuration for the testing environment

running three common services: the SSH (Port 22), the Telnet (Port 23), and the HTTP (Port 80).

B. Experiment-1: Attacking toward a Group of Servers

In this test, we assume that malicious node can brute-force the SSH and Telnet, while sending invalid HTTP packet to the web server. If any centralized security solution has not been implemented, then administrators have to do it one by one. In the comparison, our blockchain-based solution can complete this task more quickly. For example, the administrators can use the following commands via the management console, as demonstrated in Fig. 7.

In particulary, we configured the Firewall-Commander to refresh the rules every 5 seconds, as Clique consensus algorithm can finish packaging and generate blocks very quickly, as

```
#> port 22 block
Port 22 blocked
#> port 23 block
Port 23 blocked
#>
```

Fig. 7. Blocking Port 22 and Port 23 through the Console

```
Refresh
    Port 22 blocked
    Port 23 blocked
```

Fig. 8. Client updating firewall rules

shown in Fig. 8. It is guaranteed that the Firewall-Commander can reach the updated rules within the refreshing period.

After updating the firewall rules, the attack could be instantly stopped as shown in Figure 9. The attacker cannot performs either SSH or Telnet to the protected servers.

Fig. 9. The NMAP scanning result of the server

C. Experiment-2: When the Network is under Stressed

Many centralized security solutions can be often affected under the Denial-of-Service (DoS) attack. If the traffic flow was stressed out the centralized management server, it becomes difficult for clients to send or receive heartbeat toward and from the server. Hence, the deployment of security rules may become challenging.

Although blockchain is, theoretically, not affected much from DoS attacks toward single node, we still have to know how much it may affect the system. Consensus algorithms, especially those for private chains, have intensive message-exchange protocols. In this case, if the message could not be effectively exchanged, it will affect the rule deployment.

However, it is difficult to perform the experiment by really stressing the nodes with loads, as they are all on the same machine, and even the network switch is emulated. However, it does not mean that we could not emulate the environment through different ways. In this experiment, we deliberately configured the vSwitch [24] to emulate an unreliable network environment, as shown in Fig. 10. We configured the network with the following parameters:

- `Bandwidth`: 128 kbps Full-Duplex
- `Packet Loss`: 15.0%

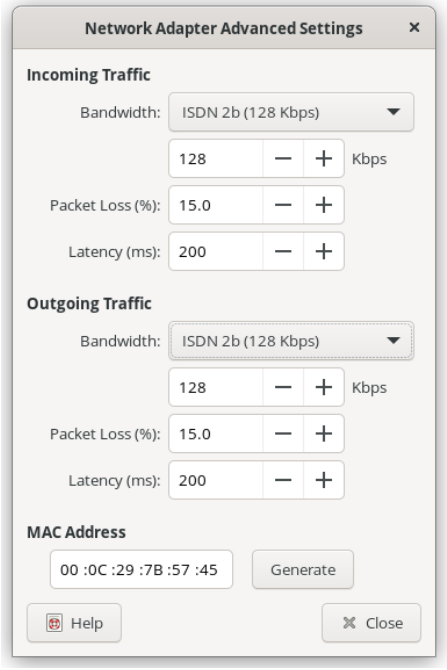

Fig. 10. Creating an unreliable network

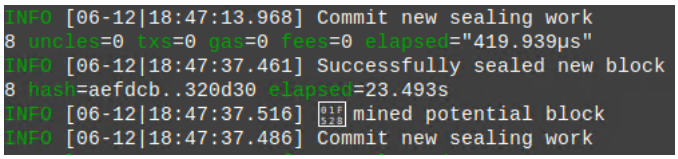

Fig. 11. The mining output from the console

- `Latency`: 200 ms

As shown in Fig. 11, the time of generating a new block instantly bumped up to around 23 seconds. Other nodes that do not join the mining took another 2-3 seconds to receive the new block. On the Firewall-Commander console, it took around 30 seconds on average to complete the deployment of new firewall rules.

Overall, it is found that our BlockFW system can still work under a stressed network, while the speed of making a policy may slow down. On the positive side, though it is becoming slower, the policy is still reachable to the endpoint.

D. Experiment-3: When a Server is Tampered

As long as the administrator's hardware key is removed from the system, the smart contract on the blockchain cannot be altered. However, we still tried to deliberately corrupt the ledger copy in one of the servers, in order to investigate how the system will react under this condition.

More specifically, we deliberately blank out one of the blockchain database files, and see how the system reacts. As shown in Fig. 12, it is found that the blockchain client detected these anomalies in the local ledger, and immediately started to sync with other nodes.

In conclusion, although an attacker can deliberately tamper a local ledger copy, the blockchain client will instantly notice the

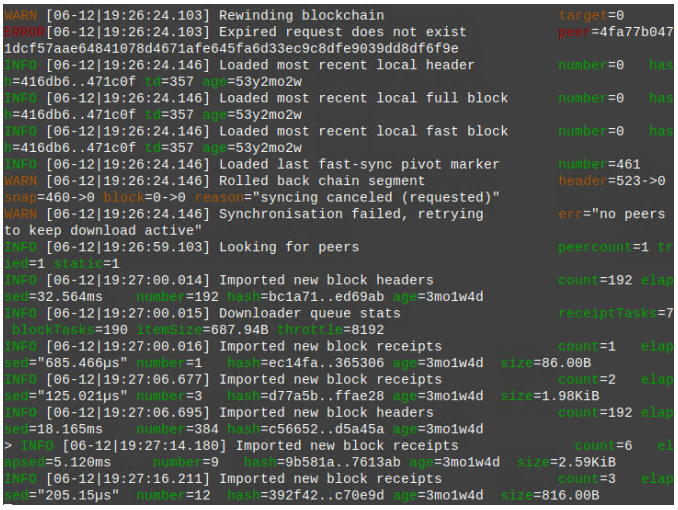

Fig. 12. Blockchain Synchronization Triggered

anomalies, start downloading chain data from other nodes and replacing the corrupted local copy. In this case, our BlockFW can be more robust than a centralized security solution, if the server is under attack.

V. Conclusion and Future Work

In this paper, we developed a blockchain-based rule-sharing firewall (called BlockFW) that can offer validation and monitoring among multiple nodes. In the evaluation, we tested BlockFW in several harsh network conditions and investigated whether it can perform better than a traditional central-managed security solution. Based on the results, it is found that our blockchain-based solution can continue to serve correctly under a stressful network condition. Also, as no central server exists in our system, there is no use for attackers to stress out one of the servers to crash the system. We further demonstrated the adversarial scenario when attackers tried to modify the policies by directly editing the blockchain storage file on one node, and identified that our system could recover itself from other reachable nodes, making the attacker's tampering trial unsuccessful. These provide a good evidence that making blockchain as the underlying database for the security solution is viable with particular advantages.

However, the BlockFW system we are developing requires some further improvements. On functionality phase, the implementation is less than a traditional firewall has, in which we are actively developing a new version to overcome this issue. Another important topic that we have not discussed is whether BlockFW can handle a large network the same as the current central-managed security solutions. This is because permission-based blockchain has to utilize voting-based consensus algorithms that require to exchange many messages to reach consensus compared with a traditional lottery-based consensus algorithm (e.g., PoW / PoS). Too many nodes may result in slowdown and a waste of network resources. Thus, the scalability issues are always important when developing a blockchain-based solution.

Acknowledgment

This work was funded by the European Union H2020 DataVaults project with GA Number 871755. The source code of BlockFW is available at SPTAGE Lab: https://nopkirouter1.compute.dtu.dk/project/blockfw.zip.

References

[1] N. Atzei, "A survey of attacks on ehtereum smart contracts (sok)," *Proc. International conference on principles of security and trust*, pp. 164-186, 2017.
[2] O. Al-Kadi, N. Moustafa, and B.P. Turnbull, "A Review of Intrusion Detection and Blockchain Applications in the Cloud: Approaches, Challenges and Solutions," *IEEE Access* 8, pp. 104893-104917, 2020.
[3] A.A. Almutairi, S. Mishra, and M. Alshehri, "Web Security: Emerging Threats and Defense," *Comput. Syst. Sci. Eng.* 40(3), pp. 1233-1248, 2022.
[4] F. Baldimtsi, V. Madathil, A. Scafuro, and L. Zhou, "Anonymous Lottery In The Proof-of-Stake Setting," *Proc. CSF*, pp. 318-333, 2020.
[5] W.Y. Chiu and W. Meng, "Mind the Scraps: Attacking Blockchain based on Selfdestruct," *Proc. the 26th Australasian Conference on Information Security and Privacy (ACISP)*, pp. 451-469, 2021.
[6] W.Y. Chiu and W. Meng, "EdgeTC - A PBFT Blockchain-based ETC Scheme for Smart Cities," *Peer-to-Peer Networking and Applications*, vol. 14, pp. 2874-2886, 2021.
[7] W.Y. Chiu, W. Meng, and C.D. Jensen, "My Data, My Control: A Secure Data Sharing and Access Scheme over Blockchain," *Journal of Information Security and Applications*, vo. 63, 103020, 2021.
[8] W.Y. Chiu and W. Meng, "DevLeChain - an Open Blockchain Development Platform for Decentralized Applications," *Proc. The 5th IEEE International Conference on Blockchain (Blockchain)*, 2022.
[9] W. Li, Y. Wang, W. Meng, J. Li, and C. Su, "BlockCSDN: Towards Blockchain-based Collaborative Intrusion Detection in Software Defined Networking," *IEICE Transactions on Information and Systems*, vol. E105.D, no. 2, pp. 272-279, 2022.
[10] W. Li, W. Meng, and L.F. Kwok, "Surveying Trust-based Collaborative Intrusion Detection: State-of-the-Art, Challenges and Future Directions," *IEEE Communications Surveys and Tutorials*, vol. 24, no. 1, pp. 280-305, 2022.
[11] W. Li, Y. Wang, and J. Li, "Enhancing blockchain-based filtration mechanism via IPFS for collaborative intrusion detection in IoT networks," *J. Syst. Archit.* 127, 102510, 2022.
[12] W. Meng, W. Li, and L.F. Kwok, "EFM: Enhancing the Performance of Signature-based Network Intrusion Detection Systems Using Enhanced Filter Mechanism," *Computers & Security*, vol. 43, pp. 189-204, 2014.
[13] W. Meng, W. Li, and J. Zhou, "Enhancing the Security of Blockchain-based Software Defined Networking through Trust-based Traffic Fusion and Filtration," *Information Fusion*, vol. 70, pp. 60-71, 2021.
[14] W. Meng, W. Li, L.T. Yang, and P. Li, "Enhancing challenge-based collaborative intrusion detection networks against insider attacks using blockchain," *Int. J. Inf. Sec.* 19(3), pp. 279-290, 2020.
[15] W. Meng, E.W. Tischhauser, Q. Wang, Y. Wang, and J. Han, "When Intrusion Detection Meets Blockchain Technology: A Review," *IEEE Access*, vol. 6, no. 1, pp. 10179-10188, 2018.
[16] Y. Meng and L.F. Kwok, "A Framework for Protocol Vulnerability Condition Detection," *Proc. SECURWARE*, pp. 91-96, 2011.
[17] R. Mukta, H.Y. Paik, Q. Lu, and S.S. Kanhere, "A survey of data minimisation techniques in blockchain-based healthcare," *Comput. Networks* 205, 108766, 2022.
[18] M. Rak, G. Salzillo, and D. Granata, "ESSecA: An automated expert system for threat modelling and penetration testing for IoT ecosystems," *Comput. Electr. Eng.* 99, 107721, 2022.
[19] M. Steichen, S. Hommes, and R. State, "ChainGuard - A firewall for blockchain applications using SDN with OpenFlow," *Proc. IPTComm*, pp. 1-8, 2017.
[20] R.L. Trope and E.K. Ressler, "Mettle Fatigue: VW's Single-Point-of-Failure Ethics," *IEEE Secur. Priv.* 14(1), pp. 12-30, 2016.
[21] 51% Attack, https://dci.mit.edu/51-attacks
[22] Frankenfield, J., 51% Attack Definition, https://www.investopedia.com/terms/1/51-attack.asp
[23] Nakamoto, S., Bitcoin: A Peer-to-Peer Electronic Cash System, https://bitcoin.org/bitcoin.pdf
[24] Open vSwitch, https://www.openvswitch.org/