Journal of Network and Computer Applications 143 (2019) 167–177

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

BlockSecIoTNet: Blockchain-based decentralized security architecture for IoT network

Shailendra Rathore, Byung Wook Kwon, Jong Hyuk Park [*]

Department of Computer Science and Engineering, Seoul National University of Science and Technology, SeoulTech, Seoul, 01811, South Korea

A R T I C L E I N F O

Keywords:
Internet of things
Security attack detection
Edge computing
Fog computing
Blockchain
Deep learning
Software defined networking

A B S T R A C T

The exponential growth of the use of insecure stationary and portable devices in the Internet of Things (IoT) network of the smart city has made the security of the smart city against cyber-attacks a vital issue. Various mechanisms for detecting security attacks that rely on centralized and distributed architectures have already been proposed, but they tend to be inefficient due to such problems as storage constraints, the high cost of computation, high latency, and a single point of failure. Moreover, existing security mechanisms are faced with the issue of monitoring and collecting historic data throughout the entire IoT network of the smart city in order to deliver optimal security and defense against cyberattacks. To address the current challenges, this paper proposes a decentralized security architecture based on Software Defined Networking (SDN) coupled with a blockchain technology for IoT network in the smart city that relies on the three core technologies of SDN, Blockchain, and Fog and mobile edge computing in order to detect attacks in the IoT network more effectively. Thus, in the proposed architecture, SDN is liable to continuous monitoring and analysis of traffic data in the entire IoT network in order to provide an optimal attack detection model; Blockchain delivers decentralized attack detection to mitigate the "single point of failure" problem inherent to the existing architecture; and Fog and mobile edge computing supports attack detection at the fog node and, subsequently, attack mitigation at the edge node, thus enabling early detection and mitigation with lesser storage constraints, cheaper computation, and low latency. To validate the performance of the proposed architecture, it was subjected to an experimental evaluation, the results of which show that it outperforms both centralized and distributed architectures in terms of accuracy and detection time.

1. Introduction

Nowadays, the Internet of Things (IoT) is playing a prominent role in the real world by supporting operations and communications autonomously, thereby facilitating and advancing new services that are widely used in everyday life. With the advancement of Information and Communication Technology (ICT) and the proliferation of sensor technologies, the IoT is now being widely used in diverse fields - such as health care, smart cities, and smart power grids, etc. - for the purposes of efficient resource management and pervasive sensing. As described in the Statista report ("IoT: number of connected, 2012), the total number of connected devices in the IoT is escalating every year and is predicted to increase to 75.44 billion devices by 2025. Due to such a remarkable escalation in the number of IoT devices, it is becoming increasingly urgent to protect all these devices against cyber-attacks. Existing IoT ecosystems are vulnerable to diverse types of security attacks due to the accessibility of devices from anywhere over the internet and the prevalence of lower-level security protection strategies. The opportunities for attackers to control and damage critical infrastructures such as important sensors, moving cars, and nuclear facilities have made the security problem with the IoT ecosystem much more significant than that of the conventional network (Khan and Salah, 2018). An attacker can take control of devices and use them maliciously, ultimately resulting in privacy violations of devices and the IoT ecosystem. Therefore, it is necessary to conduct research on and devise innovative security defense strategies to deal with the numerous attacks on the IoT ecosystem. On the other hand, the distributed nature of the IoT ecosystem makes it difficult to monitor and collect historic data from that system, which in turn makes it somewhat challenging to develop a security attack detection mechanism capable of providing optimal security and defense in the IoT ecosystem (Rehman et al., 2018). Moreover, other issues such as the complex network topology, information uncertainties, and the heterogeneity of the IoT objects make it difficult to design an effective security defense system. The existing methods of security attack detection in the

* Corresponding author.
E-mail addresses: rathoreshailendra@seoultech.ac.kr (S. Rathore), rnjsqud123@seoultech.ac.kr (B. Wook Kwon), jhpark1@seoultech.ac.kr (J.H. Park).

https://doi.org/10.1016/j.jnca.2019.06.019
Received 14 December 2018; Received in revised form 18 June 2019; Accepted 27 June 2019
Available online 3 July 2019

IoT tend to be ineffective due to several challenges.

Centralization: The majority of security attack detection methods rely on a centralized architecture, wherein an attack is identified by deploying a centralized attack detection system at the central cloud server. These methods are unlikely to scale as a large number of devices are connected in the IoT ecosystem. In addition, the cloud sever will remain inefficient due to storage constraints, the high cost of computation, high latency, and the single point of failure.

Big data problem: Security attack detection is a big data problem in that it requires a large amount of data to be collected from communicating devices and the network for the purpose of real-time data analysis. Most real-time data give the accurate decision for attack detection. However, in the IoT, the collection and computation of real-time data will cost a massive amount of resources. Moreover, an attacker can inject false data and thereby escalate bandwidth consumption and latency (Rehman et al., 2018). All of these issues tend to decrease the accuracy of attack detection in the IoT.

Lack of privacy: The existing attack detection models do not examine user privacy. In other words, they perform data collection without the owner's permission. Furthermore, due to the issue of user privacy, current attack detection models are unable to collect sufficient data, thus leading to inaccurate detection decisions (Sharma et al., 2018a).

Recently, the concept of fog and edge computing has been introduced, wherein the computing paradigms carry out the necessary computation and storage nearer to the devices. It usually structures with the cloud computing in a layered architecture. It enables real-time data analysis by processing the data at the fog and edge layer and offers an encouraging approach to supporting the next generation of services and applications with high bandwidth and low latency (Yu et al., 2018), (Chiang et al., 2017). Meanwhile, in recent years, Blockchain has been used in many next-generation applications and has become an attractive choice for interested stakeholders in a wide range of industries (Novo, 2018). This is because the blockchain supports trust-free and decentralized solutions, wherein data are stored across the network in a distributed manner in the form of online-distributed ledgers. These online-distributed ledgers support the operation of applications in a decentralized manner instead of a trusted intermediary. Using the blockchain, untrusted individuals can connect in a peer-to-peer network and exchange data in a verifiable manner without the support of a trusted intermediary. In the IoT network, each fog node can transact with each other using the blockchain without relying on the central cloud authority to overcome the single point of failure problem. On the other hand, Software Defined Networking (SDN) supports the network in managing data remotely, adaptively and dynamically. The core objective of SDN is to provide a separation between the data plane and the control plane. Hence, SDN separates the execution of the data-forwarding decision from the logical centralized controller. In IoT security, the SDN-enabled switch can assist data collection and data analysis in providing a faster response for attack detection (Kalkan and Zeadally, 2018).

This paper addresses the issue of security attack detection by employing SDN, Fog and Edge computing and Blockchain in the IoT ecosystem. It proposes an SDN-based decentralized security architecture with a blockchain technology for the IoT ecosystem that can mitigate the recent challenges and detect attacks more efficiently. In the proposed architecture, attack detection is performed at the fog layer, and attacks are subsequently mitigated at the edge layer of the IoT network. The SDN-enabled switch provides dynamic flow management of traffic, which assists the process of attack detection by finding suspicious traffic flows and mitigates attacks by blocking suspicious flows. With the help of edge computing, SDN pushes attack detection to the edge of the network, thus ensuring early detection and mitigating attacks with a fast response. Such early detection also enables a reduced storage requirement and lower latency, and reduces resource wastage and the consumption of network bandwidth. All fog nodes and cloud server share data using the blockchain technology, and enable regular updates of the attack detection model at each fog node that improves the accuracy of attack detection.

Research contribution: The main contributions of the research work include:

- We design and develop a decentralized security architecture for the IoT ecosystem that mitigates the recent challenges in existing security architectures by employing the three emerging technologies of SDN, fog and edge computing, and blockchain to provide an efficient attack detection mechanism for the IoT ecosystem
- We implement the proposed architecture on the Ethereum blockchain technology and the Mininet emulator to detect attacks in the IoT network at the fog node using a deep learning algorithm, and subsequently mitigate attacks at the edge of the network.
- We compare the performance of the proposed architecture with two existing architectures, namely the cloud-based centralized model and the fog-based distributed model. We also consider and implement various attack scenarios such as DDoS, ICMP flooding, and TCP flooding with IoT devices to demonstrate the performance of our proposed architecture.

The rest of the paper is organized as follows. Section 2 describes the background of security attack detection in the IoT ecosystem and discusses problems with the existing attack detection architectures; Section 3 presents the SDN-based decentralized attack detection architecture for the IoT and an overview of the design and workflow of the proposed architecture; Section 4 presents the experimental evaluation of the proposed architecture and a comparison with the existing centralized and distributed architectures; and Section 5 presents the conclusion of this work.

2. Background and problem statement

In line with the sharp increase in security issues in the IoT ecosystem, a number of approaches to detecting and mitigating security attacks in the IoT have been proposed. In (Zarpelão et al., 2017), the authors summarize possible security threats and discuss various methods of detection including the anomaly-based, signature-based, and specification-based methods. Elsewhere, a centralized detection strategy based on the cloud data center is proposed in (Butun et al., 2015), while a distributed attack detection approach in which novel deep learning and fog computing paradigms are employed to detect cyber-attacks in the distributed IoT ecosystem is proposed in (Diro and Chilamkurti, 2018). Hosseinpour et al. (2016) used the smart data approach at the fog layer and introduced a distributed and lightweight intrusion detection system for the IoT. In (Pajouh et al., 2016), the authors presented a novel two-tier classification and dimensional reduction model to detect suspicious actions, such as User to Root (U2R) and Remote to Local (R2L) attacks in IoT backbone networks. Other studies focused on approaches to security attack detection in the IoT from the standpoint of game theory or information-theoretic, including collaborative security detection using a consensus mechanism (Wu and Wang, 2018) and threat mitigation using an automatic scheme (Ashraf and Habaebi, 2015).

Typically, most of the existing researches on security attack detection in the IoT rely on the concept of the Intrusion Detection System (IDS), whereby an attack is detected by employing a centralized or distributed architecture. As illustrated in Fig. 1(a), in the centralized architecture (Butun, Kantarci, Erol-Kantarci), (Pajouh et al., 2016), a single IDS is deployed as an attack detection module at the central cloud server of the IoT network, which classifies network data as an attack or normal situation by employing a machine learning model on large-scale big data collected from the entire network. In the IoT network, the central cloud server collects large-scale big data from diverse IoT devices through the base station to enhance the performance of attack detection. However, the creation of large-scale big data is challenging due to the issue of the privacy of IoT devices or data providers. Due to the high probability of privacy violation, potential data providers might not be motivated to

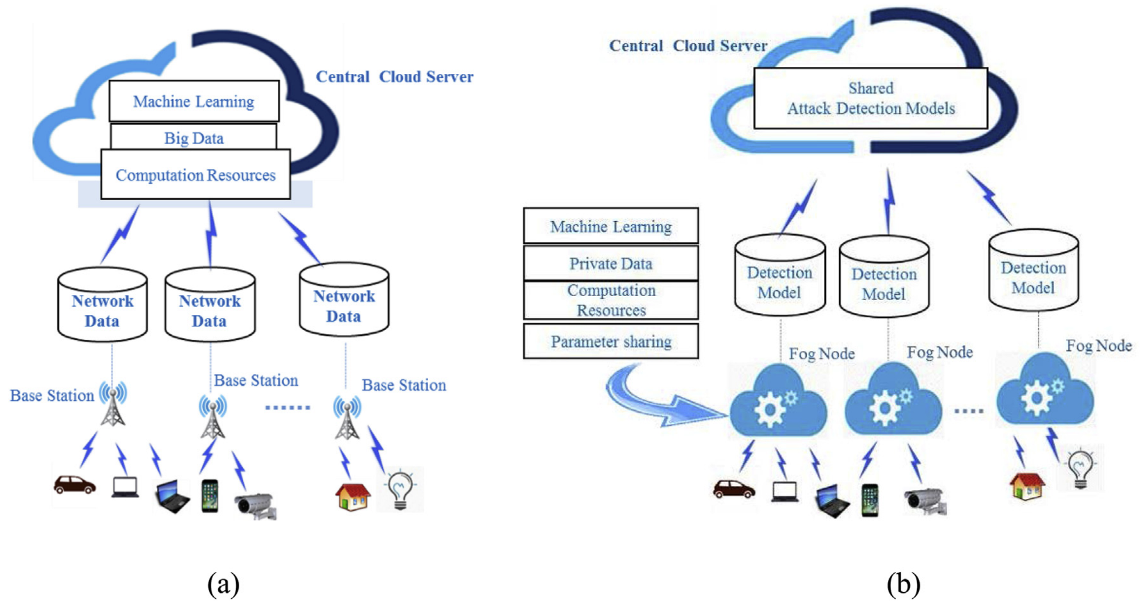


Fig. 1. (a) Centralized attack detection architecture; (b) Distributed attack detection architecture.

share their data. Moreover, a huge amount of communication bandwidth is required to accumulate a massive volume of data. The processing of such a massive volume of data in turn necessitates a large amount of computational power.

On the other hand, in the distributed attack detection architecture (Diro and Chilamkurti, 2018), the collection and processing of data is carried out in a distributed manner instead of a single cloud server, as illustrated in Fig. 1(b). Each individual fog node is responsible for collecting and processing its private data using a machine learning algorithm to prepare its own attack detection model, and then shares the individual model with the cloud server to implement distributed attack detection. In this architecture, each fog node does not share its data with the cloud server, which results in preserving the privacy of data at the fog level. Also, the higher communication bandwidth requirement is reduced due to the fact that only the parameter set of the attack detection model by each fog node is shared with the cloud server. Furthermore, data training to prepare the attack detection model is performed in a distributed manner with smaller sets of data at the fog layer nearer to the source, thus resulting in lower computation overheads and offloading storage, while providing a fast response time due to the computation nearer to the source devices.

Nevertheless, in the distributed attack detection architecture, the centralized cloud sever has full control of attack detection. Therefore, the distributed attack detection architecture presented in Fig. 1(b) is prone to the single point of failure problem. Moreover, in this architecture each fog node requires a sufficient quantity of data to make an accurate decision about attack detection. However, it is sometimes impossible to collect sufficient data for a fog node due to the smaller number of communication devices associated with the fog node or due to the issue of the privacy of connected devices, leading to a lower accuracy rate of attack detection. To address these problems, this paper upgrades the distributed attack detection approach, as shown in Fig. 2, and obtains a new decentralized and cooperative attack detection architecture. In the new architecture, each individual fog node has full control of its own

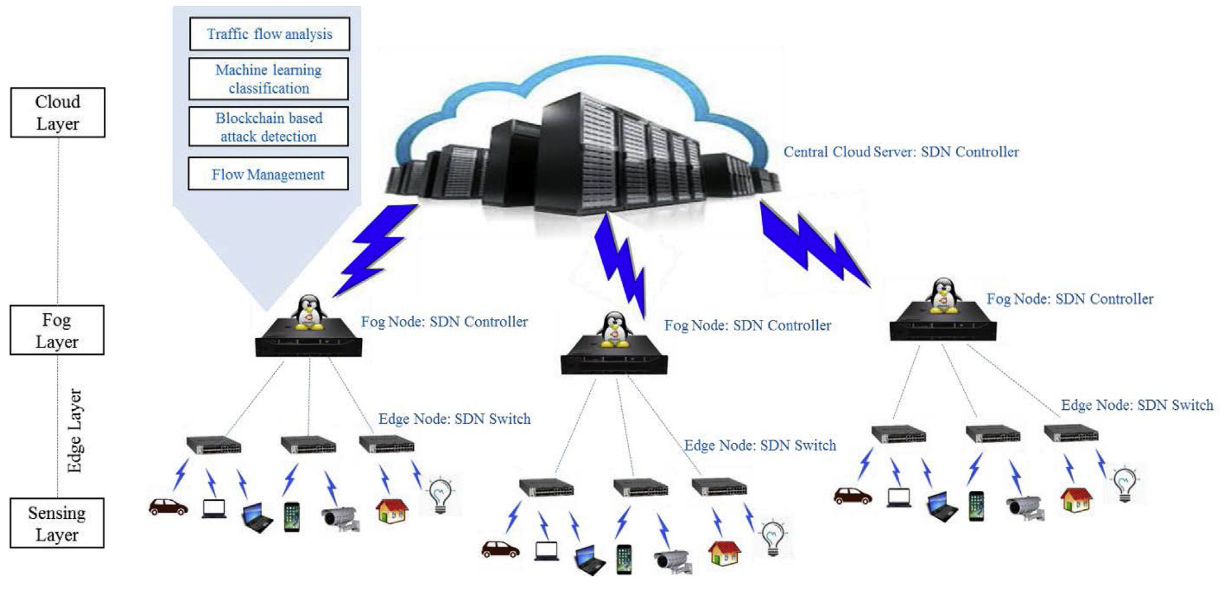


Fig. 2. Overview of the design of the proposed decentralized security architecture.

attack detection model, which it shares with other fog nodes and cloud servers so as to mitigate the single point of failure problem and ensure sufficient data availability. Thus, the sharing of the attack detection model provides an accurate detection model for the fog node with insufficient data, which further improves the accuracy of attack detection. To design and develop the new architecture, we use the Ethereum blockchain technology, which supports peer-to-peer transactions between fog nodes using a smart contract to enable decentralized and cooperative attack detection (Kosba et al., 2016). Moreover, the new architecture exploits the SDN and fog and edge computing paradigm to enable the detection and mitigation of attacks with a fast response. A number of results on the integration of SDN, fog and edge computing, and blockchain to improve security in the IoT ecosystem have been reported. In (Sharma et al., 2017), a DistBlockNet model is presented by combining the advantages of blockchain technology and SDN to provide an efficient and secure network architecture for the IoT. Recently, we introduced an innovative SDN-based distributed layered architecture for a sustainable edge computing network capable of supporting security at the fog layer to mitigate security threats in the distributed network (Sharma et al., 2018b).

3. Proposed BlockSecIoTNet architecture

Considering the problem setting described above, BlockSecIoTNet, a novel decentralized and cooperative attack detection architecture for IoT ecosystem is introduced. In this section, we explain the design overview and workflow of the proposed BlockSecIoTNet architecture.

3.1. BlockSecIoTNet design overview

Fig. 2 presents an overview of the design of the proposed architecture, which is a layered model consisting of four layers: sensing, edge, fog, and cloud layers. The sensing layer consists of many smart devices and widely distributed sensory nodes that monitor diverse environments and activities in the public infrastructure. It produces a huge amount of sensing data, which is forwarded to the edge layer. The edge layer consists of low-power high-performance SDN-enabled switches at the edge of the network. Each SDN-enabled switch at the edge layer is connected to a number of sensors at its premises and is liable to process and analyze traffic data from sensors. The processed data from the edge layer are reported to the fog layer, which is composed of a number of SDN controllers. Each SDN controller at the fog layer is associated with a cluster of SDN switches at the edge layer and is responsible for analyzing processed data to identify anomalous traffic flows. Based on an identified anomalous traffic flow, the SDN controller is also responsible for updating and managing the flow rules to their respective switches and instructing the switches to detect attacks with low latency. Moreover, each SDN controller announces the outcomes of its data processing to the cloud layer for monitoring large-scale and long-term anomalous behaviors and data analysis.

3.2. Methodological flow of BlockSecIoTNet

The workflow of the proposed architecture follows the bottom-up approach whereby operation starts from the sensing layer and ends at the cloud layer. During operation, each SDN-enabled switch at the edge layer continuously monitors the traffic flow of the sensor nodes connected to it at the sensing layer and reports the monitored traffic traces to its respective fog node at the fog layer. The traffic traces are learned and analyzed by the SDN controller at the fog node to identify malicious traffic flows from sensor nodes. The controller employs the historical behaviors of traffic, such as known attack patterns and the number of features to identify if the traffic flow is malicious or normal. After performing the analysis, the traffic flow rules for the switch are set in the SDN controller at the fog node. Then, the SDN controller sets the flow rules to their respective switches dynamically. Based on the rules set and defined by the SDN controller, the switches perform various actions on the traffic flow from IoT devices. These actions may consist in applying rate limits on the flow, or in blocking the flow partially or fully. Furthermore, the SDN controller at the cloud layer is updated by each fog node at regular intervals, if the fog node observes any significant patterns or events. Thus, the SDN controller at the cloud has information about events and patterns from various fog nodes, which helps discover identical traffic patterns from various fog nodes and provide decisions about traffic flow from IoT devices located anywhere in the network. Thus, the SDN controller can assist with the provision of security to devices remotely with no manual intervention by the end users, look for similar events from different clusters, and draw conclusions about incoming waves of attacks on IoT devices across its network. This intelligence can be helpful in securing devices remotely without the need for any manual intervention by the end users.

In the proposed architecture, the SDN controller at the fog node is composed of the following four components: traffic flow analyzer, traffic flow classifier, blockchain-based attack detection, and attack mitigation module. The first two components are responsible for identifying anomalous traffic so as to prepare an individual attack detection model for the fog node, while the third component contributes to the dynamic updating of the attack detection model using blockchain technology. The attack detection model is further used by the final component, i.e. the attack mitigation module, to mitigate attacks at the edge layer.

Traffic flow analyzer: The traffic flow analyzer dynamically observes the traffic flow from different IoT devices and collects traffic patterns, including the packet and flow-level features of traffic, such as device utilization at various time intervals, bandwidth utilization, total number of sent requests from a device, source of request, etc. During normal usage, the collected patterns are labeled as “legitimate” and delivered as training data to identify the legitimate traffic. In addition, the traffic flow analyzer also contains various known vulnerabilities of IoT devices, several known patterns of attack (e.g. DDoS, TCP flooding), and blacklisted source IP addresses in order to identify the attacks in the IoT ecosystem. Hence, the traffic flow analyzer acquires knowledge about malicious and legitimate traffic and provides this knowledge to the traffic flow classifier.

Traffic flow classifier: This component is responsible for preparing the attack detection model to classify the attack traffic at the fog node. Each fog node prepares the trained model using a machine learning algorithm over the data provided by their respective traffic flow analyzers. In the proposed architecture, deep learning is used as a machine learning algorithm for classification.

Blockchain-based attack detection: Based on the attack detection model obtained from the traffic flow classifier, this component dynamically updates the attack detection model at the fog node based on the attack detection model from other fog nodes. The updated attack detection model enables efficient attack detection and instructs the SDN switches to detect attacks. In order to dynamically update the attack detection model, a novel approach is proposed, as shown in Fig. 3. The proposed approach consists of two entities: manager and agents. A manager is an entity that manages the attack detection models from a set of fog nodes. It defines the data-driven tasks for attack detection, such as the structure of input data, and the estimated output from the attack detection models. In the proposed approach, the central cloud server acts as a manager that delivers a testing set of data and describes the estimated accuracy for proofing the attack detection model from each fog node, and then defines the financial compensation commitments for the decentralized attack detection model. Overall, the central cloud server recommends the structure of the attack detection model. Conversely, agents are entities that are responsible for processing and proofing the decentralized attack detection model. For the approach proposed in this paper, a fog node can act as a processing or proofing agent. As a processing agent the fog node is liable to process its own local data using a machine learning algorithm to prepare the attack detection model. The proofing agent is responsible for verifying and proofing the prepared attack detection model.

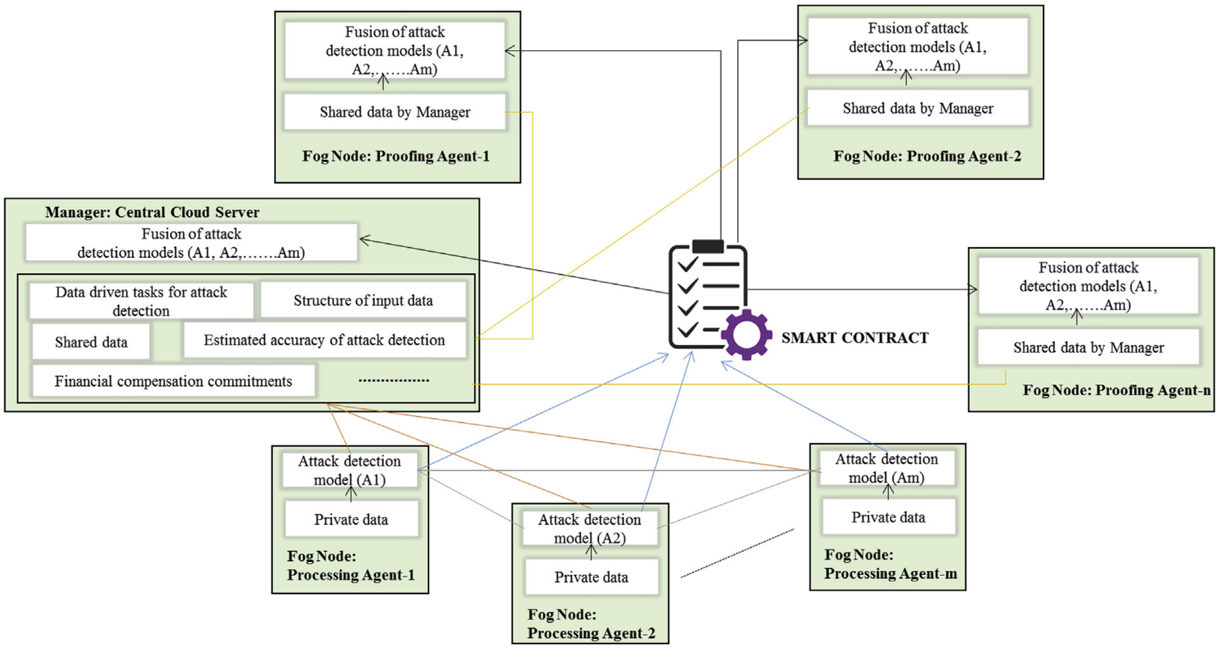

Fusion of attack detection models (A1, A2,……Am)
Shared data by Manager
Fog Node: Proofing Agent-1
Fusion of attack detection models (A1, A2,……Am)
Shared data by Manager
Fog Node: Proofing Agent-2
Manager: Central Cloud Server
Fusion of attack detection models (A1, A2,…….Am)
Data driven tasks for attack detection
Structure of input data
Shared data
Estimated accuracy of attack detection
Financial compensation commitments
……………
SMART CONTRACT
Fusion of attack detection models (A1, A2,……Am)
Shared data by Manager
Fog Node: Proofing Agent-n
Attack detection model (A1)
Private data
Fog Node: Processing Agent-1
Attack detection model (A2)
Private data
Fog Node: Processing Agent-2
Attack detection model (Am)
Private data
Fog Node: Processing Agent-m

Fig. 3. Overview of Blockchain-based attack detection in the IoT.

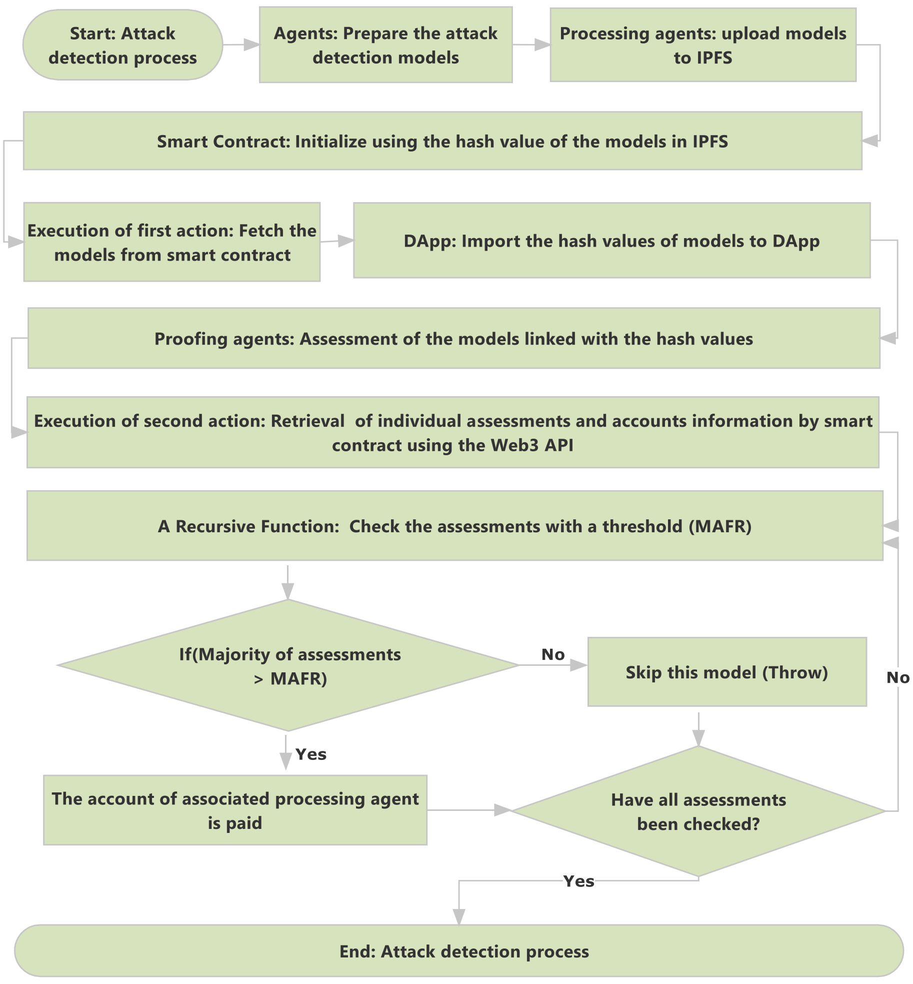

Start: Attack detection process
Agents: Prepare the attack detection models
Processing agents: upload models to IPFS
Smart Contract: Initialize using the hash value of the models in IPFS
Execution of first action: Fetch the models from smart contract
DApp: Import the hash values of models to DApp
Proofing agents: Assessment of the models linked with the hash values
Execution of second action: Retrieval of individual assessments and accounts information by smart contract using the Web3 API
A Recursive Function: Check the assessments with a threshold (MAFR)
If(Majority of assessments > MAFR)
No
Skip this model (Throw)
No
Yes
The account of associated processing agent is paid
Have all assessments been checked?
Yes
End: Attack detection process

Fig. 4. Work flow of the blockchain in the proposed architecture.

All of the participants, including manager, processing agent, and proofing agent, interact with each other through transactions effectuated using the Ethereum blockchain (bitfly.at and "Etherchain,", 2018). Initially, the manager specifies the data-driven tasks for attack detection, such as the structure of input data, the estimated output of the attack detection models, and the financial compensation commitments. According to the specified data-driven tasks, the processing agents train their own private data using a machine learning algorithm to prepare their own attack detection model. The prepared attack detection is published to the proofing agents to verify whether the overall performance of the attack detection task has been improved by fusing this attack detection model. The proofing agents also evaluate the contribution of the processing agents. A majority vote among the various proofing agents is employed to compute the contribution of each corresponding processing agent. According to the contribution, each processing agent receives compensation from the manager. In addition, the manager provides compensation to the proofing agents for their efforts. After compensating the processing and proofing agents with financial rewards, the manager gets to access the rewarded attack detection models based on the smart contract defined in the Ethereum blockchain. Later, the manager can exploit the same strategy employed by the proofing agents in order to fuse the individual attack detection models and thereby obtain a fused attack detection model.

To implement the Ethereum blockchain in the proposed approach, the truffle development suite (Trufflesuite, 2018), Web3.js API ("ethereum/wiki," GitHub., 2018), and Oraclize API (Team and OraclizeI, 2015) were used as development tools. In addition, smart contracts were written in solidity language to perform the transactions between all of the participants in the blockchain for decentralized attack detection. As shown in Fig. 4, two actions are executed in the smart contracts. In the first action, the manager initiates the attack detection process by specifying the data-driven tasks, such as the structure of input data, the estimated output accuracy, and the financial compensation commitments. For the given data-driven tasks, the processing agents prepare the attack detection models by performing off-chain machine learning training on their local data. Further, they use the InterPlanetary File System (IPFS) as a decentralized storage (Benet, 2014) to record the parameters of their attack detection models in the form of hash values. The hash values from the IPFS are then imported to the Decentralized Application (DApp) (Buterin, 2014) in order to announce the prepared models. Once the proofing agents have received the broadcasted attack detection models, they begin an off-chain assessment of the models and announce the results of their assessment through DApp. Then, the second action is executed, whereby the smart contract uses the Web3.js API to retrieve the individual assessments obtained by the proofing agents, and then checks the retrieved assessments within the threshold defined by the manager as the Minimum Acceptable Fitness Rate (MAFR). As regards the majority of assessments that are higher than MAFR, the Ethers are monogamously transmitted to the accounts of the associated processing agent by the payable function employed in the Ethereum blockchain. Furthermore, all of the proofing agents receive financial compensation from the manager for their efforts.

Model fusion strategy: As shown in Fig. 3, the progress of the proposed approach needs an effective strategy to fuse the individual attack detection models of the individual processing agents so as to obtain an efficient, high-performing attack detection model. Each processing agent prepares its attack detection model by training its private data using a deep learning mechanism. Both manager and proofing agents require an effective fusion strategy to fuse the various models. Research on multi-model fusion is currently being conducted in the field of machine leaning (Vielzeuf et al., 2017). In the literature, several fusion strategies for fusing multiple models trained using deep learning have been proposed. There are two main types of fusion strategy: (1) score fusion (late fusion), in which a fused model is obtained based on the predictions given by each trained model; and (2) feature fusion (early fusion), in which a model fusion relies on the vectors of the latent features given by each trained model (Ergun et al., 2016). Since deep learning supports excellent performance at features learning and their representation, early fusion was found to be the most optimal technique and has shown state-of-the-art performance in fusing deep learning models (Neverova et al., 2016). Moreover, deep learning in IoT provides the learning of large-scale big data that converts the raw data space into a higher-level feature space to deliver more abstract expression of the data while maintaining reasonable computation and memory overhead. For example, A. A. Diro et al. (Diro and Chilamkurti, 2018) demonstrated that deep learning obtains good performance in attack detection on an IoT ecosystem. The early fusion of deep learning for our proposed approach was used under the following assumptions: (a) The attack detection task is a deep learning classification; (b) Each processing agent prepares its own classification model according to the attack detection task given to it by the manager; (c) The processing agents are rational and only use data that are consistent with the properties specified by the manager in the smart contract.

Given an unlabeled dataset $a = \{a_1, a_2, \ldots, a_n\}$ for deep learning model A_k, where n input neuron for the first layer of model A_k describes the encoding process as follows:

$$h_1 = \mathcal{F}(w_1 a + b_1)$$

where $\mathcal{F}$ refer to an activation function and $w_1, b1$ represent weight matrix and bias vector between the input layer and the hidden layer, respectively. We use sigmoid function as an activation function that can be described as follows:

$$\mathcal{F}(z) = 1/[1 + \exp(-z)]$$

In deep learning model, the first hidden layer output h_1 is used as an input to the next hidden layer h_2 for training the network parameters w_2 and b_2. The h_2 represents the feature extracted as the second hidden layer. The process of training is repeated until the given N^{th} hidden layer h_N to train network parameter W_N. The h_N gives the feature extracted at the N^{th} layer of the model A_k. For ease of presentation, we use $w = [w_1, w_2, \ldots, w_N]$, and $b = [b_1, b_2, \ldots, b_N]$ to represent the network parameters (weight matrix and bias vector) on the N^{th} hidden layer of deep learning model A_k.

The network parameters in the training process of deep learning are updated through the gradient descent algorithm ("Deep Learning," Deep Lea, 2018) that can be mathematically described as below:

$$w_{1,l+1} = w_{1,l} - \beta \frac{\vartheta}{\vartheta w_1} J(w_1, b_1), \quad l = 1, 2, \ldots L$$

$$b_{1,l+1} = b_{1,l} - \beta \frac{\vartheta}{\vartheta b_1} J(w_1,\ b_1), l = 1, 2, \ldots L$$

Where L, β are the maximum number of iterations and learning rate of standard gradient descent algorithm, respectively. J represents the loss function that is minimized using stochastic gradient descent mechanism.

Thus, the training process of each attack detection model A_k is carried out using the above mentioned procedure, and features of all models $(f_1, f_2, \ldots, f_k, \ldots, f_m)$ are extracted from the last hidden layer h_N of each model. Based on the extracted features, the early fusion of all attack detection models $(A_1, A_2, \ldots, A_k, \ldots, A_m)$ is performed as shown in Fig. 5, wherein two layers are used to define a fully connected structure with a single hidden layer. The weighted sum of the values at each layer is calculated to achieve the corresponding values of the next layer.

In Fig. 5, to fuse m shared models, initially, a concatenated feature f_c is obtained by concatenating the feature vectors f_k of each k^{th} model (Feichtenhofer et al., 2016). Then, the weighted sum of the concatenated feature vector f_c is calculated to obtain the hidden layer H with the size $\sum_{k=1}^{m} |c_k|$, where $|c_k|$ is the number of labeled classes in the k^{th} model. Two fully connected weight matrices W_1 and W_2 are used to calculate the hidden layer output H and the final output Y, respectively. Initially, both

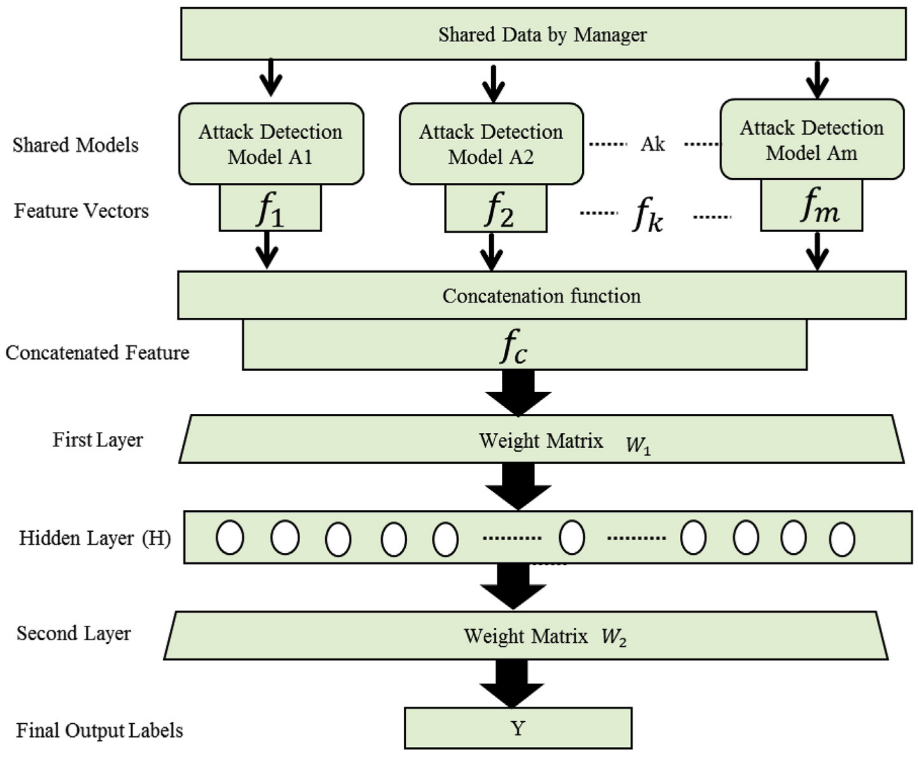


Fig. 5. The structure of the model fusion strategy.

weight matrices are initialized randomly, and the optimum values for both matrices are obtained using the back-propagation algorithm ("Deep Learning," Deep Lea, 2018),

$$H_i = \sum_{k=1}^{|f_c|} W_{1ki}^T \cdot f_{c_k}$$

Where f_c is the concatenated feature vector and W_1 is a weight matrix.

To calculate the final output Y, the softmax function is used, which is shown as follows:

$$Y_i = \frac{\exp\left(\sum_{k=1}^{|H|} W_{2ki}^T \cdot H_k\right)}{\sum_{j=1}^{d}\exp\left(\sum_{k=1}^{|H|} W_{2kj}^T \cdot H_k\right)}$$

Where W_2 is the weight matrix.

Attack Mitigation: Based on the attack detection model from the previous component, this security component is responsible for setting appropriate flow rules dynamically through the SDN controller in order to notify the switch to perform a suitable action in the case of an anomalous event. The anomalous events in the IoT network can be categorized into four general types: (a) Remote to Local (R2L), in which an attacker imitates as a local user and attempts to get access to the victim network or host machine; (b) User to Root (U2R), wherein a malicious user might escalate his/her privilege from limited access to root user access by employing various traditional exploitation techniques, such as malware infection and stolen credentials; (c) Denial of Service (DoS) that exploits the network access of the local user and attempts to make the service access unavailable to the user; (d) Probe, wherein an adversary scans the network or host activities and tries to get information concerning the network. To define appropriate flow rules for mitigating these anomalous events, our attack detection model relies on some input features. According to NSL-KDD dataset (Tavallaee et al., 2009) and our recent research (Rathore and Park, 2018), four categories of features, their name and data type are described as shown in Table 1. The host and time-based traffic features are represented as 1–10 and 11–19, respectively. In addition, the content and some basic features are depicted as 20–31 and 32–41, respectively. Based on the input features, the rule-setting task involves three major scenarios: (1) If the attack detection model classifies the traffic flow as normal traffic, then the SDN controller sets the rules for notifying the switch to pass the traffic flow in an uninterrupted manner; (2) When the traffic flow is detected as a suspicious or attack flow, then the SDN controller applies a list of rules. Initially, the switch is instructed to block suspicious traffic fully with instant effect. Second, blacklisting is applied to the source of the attack. Furthermore, the SDN controller at the cloud layer is updated with the blacklisted source to apply the blacklisting at a higher level, which prevents the attacker from affecting other devices in the IoT ecosystem. The updated rules in the SDN controller at the cloud also help to prevent situations in which certain types of device come under attack; (3) When the pattern of the traffic flow is not detected by the attack detection model, then a

Table 1
Input features to attack mitigation in IoT.

Category	Features	Data type
Host-based traffic features	(1) dst_host_count, (2) dst_host_srv_rerror_rate, (3) dst_host_srv_count, (4) dst_host_rerror_rate, (5) dst_host_same_srv_rate, (6) dst_host_srv_serror_rate, (7) dst_host_diff_srv_rat, (8) dst_host_serror_rate, (9) dst_host_same_src_port_rate, (10) dst_host_srv_diff_host_rate	Numeric
Time-based traffic features	(11) srv_diff_host_rate, (12) srv_count, (13) count, (14) diff_srv_rate, (15) serror_rate, (16) same_srv_rate, (17) srv_error_rate, (18) srv_rerror_rate, (19) rerror_rate	Numeric
Content features	(20) logged_in, (21) is_guest_login, (22) is_host_login	Binary
	(23) num_failed_logins, (24) num_outbound_cmds, (25) num_compromised, (26) num_access_files, (27) root_shell, (28) num_shells, (29) su_attempted, (30) num_file_creations, (31) num_root	Numeric
Basic features	(32) Protocol_type, (33) Service, (34) Flag	Nominal
	(35) src_bytes, (36) Duration, (37) dst_bytes, (38) hot, (39) wrong_fragment, (40) urgent.	Numeric
	(41) Land	Binary

further investigation of the traffic flow is required, and the SDN controller instructs the switch to limit the rate of traffic so as to reduce the effects of suspicious traffic.

4. Experimental analysis

This section presents the results of the evaluation of the performance of the proposed BlockSecIoTNet architecture. For the evaluation, Mininet (M. Team, 2018) was used as an emulation environment to emulate the open vSwitches and various functional nodes, such as IoT devices, in the proposed architecture. On the Linux server, we set up Mininet using 15 desktops, where the configuration of each desktop involved a 64 GB DDR3 RAM and an Intel i7 processor. We used POX (Noxrepo, 2017) as a controller to implement the machine learning classification for analyzing traffic behavior for attack detection. The controllers were run in separate VMs hosted on a Linux server for the fog and edge nodes. The Amazon EC2 cloud data center was used as a cloud server. To enable decentralization in our architecture, we implemented the architecture using the Ethereum blockchain technology, where an oracle was used in the private chain by employing the Ethereum Bridge in the broadcast mode (Oraclize, 2018). Additionally, the Truffle development suite was used to compile and deploy the DApp developed for our work. In the experimental evaluation, we considered that the manager initiates data-driven tasks and that each processing agent develops its attack detection model by employing a deep learning algorithm over its local private data. After a certain regular interval, the processing agents published their attack detection model to the proofing agents for assessment. Based on the results of the assessment, the manager provided financial rewards to the processing agents and accessed the attack detection model shared by the rewarded agent. In the simulations, for each processing agent, we provided private data from the NSL-KDD dataset, which is an intrusion detection dataset (Tavallaee et al., 2009). The private data contained the normal and attack patterns of traffic from the IoT devices. Moreover, we assumed that ten fog nodes acted as processing agents and that the remaining five fog nodes acted as proofing agents. Each processing agent contained 500 different training data and 500 testing data. Each proofing agent received the same dataset from the manager (cloud server) for verification. The distribution of data among the processing and proofing agents is shown in Table 2.

The performance of the proposed architecture was evaluated in terms of different measures, such as Accuracy, Positive Predictive Value (PPV), Detection Rate (DR), F-score, Mathew Correlation Coefficient (MCC), Detection Time (DT), and the area under the Receiver Operating Characteristic (AUC) curve, which are the most important evaluation metrics for attack detection (Diro and Chilamkurti, 2018), (Rathore et al., 2017). Based on these metrics, we first evaluated and compared the performance of the proposed decentralized architecture with two existing architectures, namely, the cloud-based centralized model and the fog-based distributed model. Secondly, we analyzed the performance of all three architectures by considering different attack scenarios.

Comparison of performance of different architectures: Here, we compared the proposed decentralized architecture with two different architectures, the cloud-based centralized model and the fog-based distributed model. For the centralized architecture, we employed a deep learning paradigm for attack detection at the Amazon EC2 cloud data center. Subsequently, we applied the deep learning paradigm at the fog node to perform attack detection in the distributed architecture. Fig. 6 shows a comparison of the performance of the proposed decentralized architecture and the centralized and distributed architectures in terms of standards evaluation metrics, clearly demonstrating that the distributed architecture outperformed the centralized architecture, and that the decentralized architecture showed superior performance to both the centralized and distributed architectures. In the centralized architecture, attack detection is performed by processing the data from IoT devices at the cloud server, whereas the distributed architecture performs attack detection at the fog level by processing the data from IoT devices connected to each fog node, which share the detection load among several fog nodes and improve the performance of attack detection.

However, in our decentralized architecture, the attack detection model at each fog node is updated dynamically by employing the blockchain technology, which improves the attack detection performance. Furthermore, Fig. 7(a) shows the variations in the detection time against the total quantity of data traffic. With the increase in data traffic, the detection time always increases in all three architectures. However, the decentralized architecture always obtained a lower detection time than the distributed and centralized architectures. This could be due to the fact that, in the decentralized architecture, the fog node prepares the attack detection model and regularly updates the flow rule in the SDN switch at the edge for attack detection, which is nearer to the IoT devices, and shortens the detection time. Similarly, Fig. 7(b) shows variations in accuracy against the total quantity of data traffic. The increasing volume of data traffic provides more data for the deep leaning classification task in attack detection, thus leading to more accurate attack detection in all three architectures. However, the decentralized architecture uses the blockchain technology to update dynamically the attack detection model at each fog node, resulting in more accurate attack detection than the distributed and centralized architectures. Overall, the decentralized architecture outperforms both centralized and distributed architectures for security attack detection in the IoT network, suggesting that decentralized attack detection using blockchain technology is an effective approach for detecting attacks in smart IoT ecosystem such as self-driving car (Karnouskos, 2018), where more accurate and real-time decisions are required.

Analysis of different attack scenarios: In order to evaluate the performance of our proposed architecture for attack detection and mitigation, we considered three different attacks as use case scenarios: a) TCP flooding attack; b) ICMP flooding attack; c) Distributed Denial of Service (DDoS) attack. The network topology for the simulation is shown in Fig. 8, where two IoT devices (A, B) and one attacker node (C) are configured to test different attacks.

a) *TCP flooding attack:* In this attack, IoT device comes under attack by attacker node C. In particular, node C tries to perform a DoS attack on device using TCP flooding. We detected and mitigated the TCP flooding attack in this scenario using the centralized and distributed approaches and the proposed decentralized approach. In the simulation, 1 Mbps bandwidth is allocated and shared by devices A and B. Initially, devices A and B utilize bandwidths of 0.5 and 0.3 Mbps, respectively. Unfortunately, attacker node C sends numerous TCP SYN packets to IoT device B, which has the effect of increasing the bandwidth consumption by device B while decreasing the throughput of device A. As shown in Fig. 9(a), our decentralized approach detects suspicious and unusual behavior at $t = 6s$ and blocks the suspicious flow of device B at $t = 12s$, and the bandwidth of device A is restored. Thus, the attack is mitigated with a recovery time of approximately 6s. However, the recovery time in the case of the distributed and centralized architectures is 7s and 10s, respectively, demonstrating that our proposed architecture mitigates the TCP flooding attack in a shorter time than the centralized and distributed architectures.
b) *ICMP flooding attack:* Here, an attacker compromises an IoT device and uses it to launch an attack in the IoT network. In our simulation, attacker node C compromises IoT device A, which generates a large volume of data traffic and begins to act adversely. Thus, device A is exploited to attempt an ICMP flooding attack. In the simulation, 1 Mbps bandwidth is allocated and shared by devices A and B.

Table 2
Distribution of data among agents.

Agent	No. of training instances	No. of testing instances
Processing agent	500	500
Proofing agent	2000	500

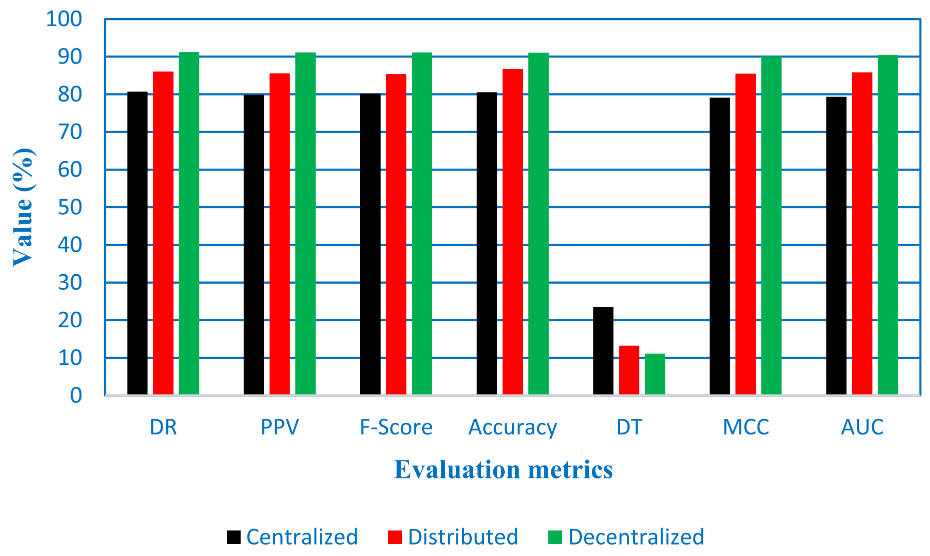
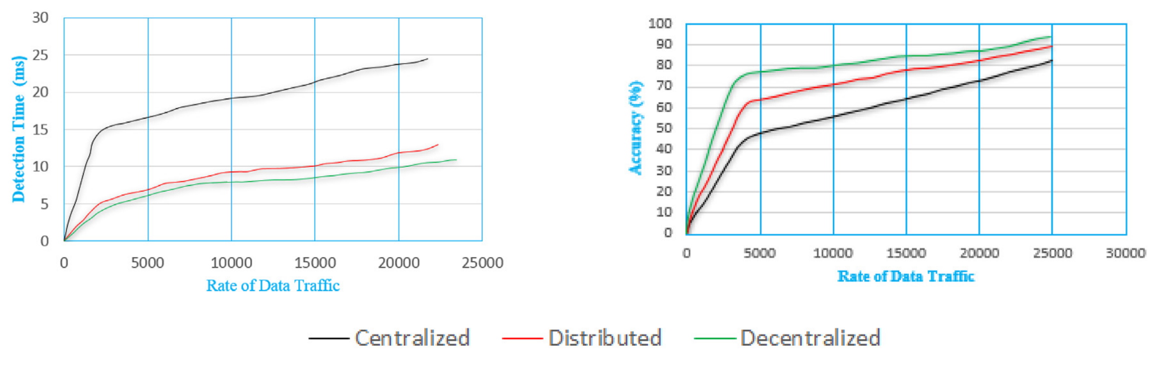


Fig. 6. Comparison of the performance of the different architectures.



(a) (b)

Fig. 7. Attack detection performance of different architectures with a varying rate of data traffic versus: a) detection time: b) accuracy.

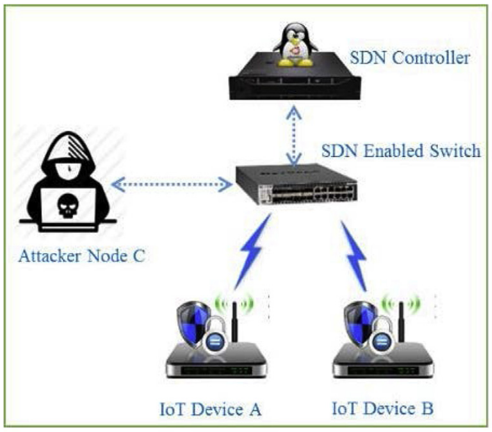


Fig. 8. Network topology to test the different attack scenarios.

Initially, devices A and B utilize bandwidths of 0.5 and 0.3 Mbps, respectively. After a certain time, IoT device A starts flooding a certain destined node with ICMP messages. The quantity of ICMP messages is considerably higher than usual, which further decreases the throughput of device B and implies that IoT device A has been compromised. As shown in Fig. 9(b), our decentralized approach identified compromised device A at the time t = 5s and the ICMP flow was fully blocked at t = 12s. Furthermore, device B is restored legitimate TCP traffic and the recovery time is approximately 7s. The centralized and distributed architectures mitigated the ICMP flooding attack in 8s and 11s, respectively, i.e. slower than the proposed decentralized approach.

c) *DDoS attack:* In this attack scenario, multiple IoT devices are compromised to launch a DDoS attack. In our experiment, attacker node C compromises devices A and B in succession with a time difference of a few seconds. The two compromised devices perform a DDoS attack by operating as a botnet entity. As shown in Fig. 9(c), initially, the IoT device A is compromised and its bandwidth increases from 0.4 to 0.6 Mbps at t = 5s. Then, at t = 14s another device B is compromised, and its bandwidth is triggered to rise from 0.2 to 0.8 Mbps. Our decentralized architecture detected the abrupt flow of traffic and identified the compromised devices. The traffic flow of each of devices A and B was blocked at t = 10s and t = 18s, respectively. In this scenario, initially, the compromised device A performs a DDoS attack and is recovered. After some time, however, compromised device B starts launching the attack. Here, then, two successive attacks are promptly mitigated and the abrupt traffic flow is blocked to recover the system to its normal state. As shown in Fig. 9(c), the average recovery time for the decentralized architecture is 5s. However, the centralized and distributed architectures take slightly longer times of 8 and 14s, respectively, to recover the system to its normal state.

In the three different attack scenarios, we observed lower time for

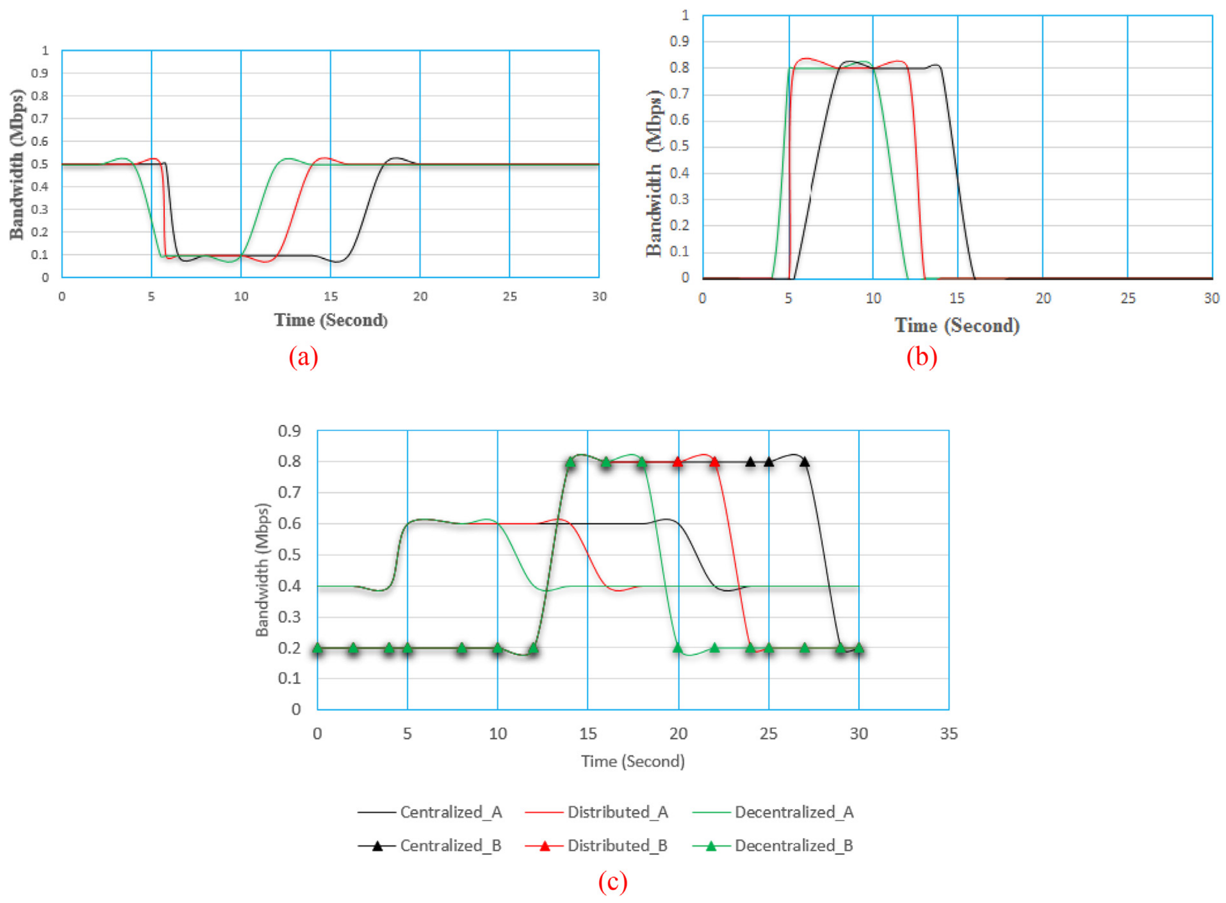


Fig. 9. Comparison of the performance of the different architectures in detecting and mitigating: a) TCP flood attack; b) ICMP flooding attack; c) DDoS attack.

Table 3
Attack mitigation time for different architectures.

Attack scenario	Architecture	Centralized	Distributed	Decentralized
TCP flooding		10s	7s	6s
ICMP flooding		11s	8s	7s
DDoS attack		14s	8s	5s

attack mitigation when decentralized architecture is used as shown in Table 3. This could be due to the fact that the decentralized architecture uses the blockchain technology to update dynamically the attack detection model at each fog node, which provides updating of the flow rule quickly and accurately in the SDN switch at the edge for attack mitigation, resulting in shorter attack mitigation time than the distributed and centralized architectures. In summary, decentralized architecture provides an effective way for mitigating attacks in smart IoT ecosystem such as self-driving car (Karnouskos, 2018), where fast mitigation of an attack is required.

4.1. Overhead of blockchain operation

We evaluated the overall complexity of the proposed architecture based on the blockchain operation. As compared to centralized and distributed architectures, the blockchain operation causes additional overhead in the proposed architecture. We observed the average usage of computational resources (CPU and memory) by the fog nodes during the blockchain operation as shown in Fig. 10. The fog nodes consume slightly higher memory and CPU resources during the blockchain operation for committing and packing the transactions into new blocks in the blockchain. Since the proposed decentralized architecture outperforms both centralized and distributed architectures in terms of accuracy and detection time, slightly extra overhead can be considered acceptable.

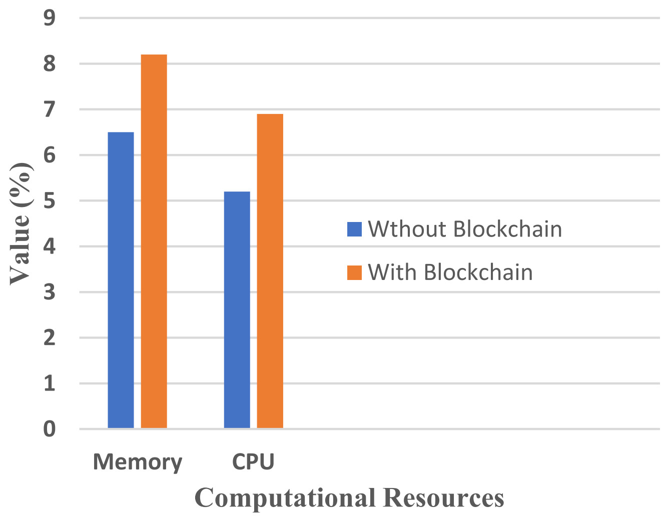


Fig. 10. Computational complexity of the proposed architecture.

5. Conclusion

In this study, we proposed a decentralized security architecture that detects and mitigates security attacks in the IoT ecosystem. The proposed architecture has made three new contributions in the area of IoT security. First, the proposed architecture uses SDN to continuously monitor and analyze traffic data in the entire IoT ecosystem, thus mitigating the issue of data unavailability in security detection and providing optimal security defense. Second, the architecture employs the Blockchain technology which supports decentralized attack detection to overcome the single point of failure problem inherent to the centralized and distributed architectures. Finally, the architecture relies on the layered structure, where attacks are detected at the fog node and subsequently mitigated at the edge node, which contributes to shortening the time taken to detect and mitigate attacks. The results of our evaluation demonstrate that the proposed decentralized security architecture outperforms the centralized and distributed architectures and takes less time to mitigate attacks in the IoT ecosystem. Our findings also suggest that the architecture could be deployed with the IoT ecosystem as a security detection component that detects and mitigates potential attacks by monitoring and analyzing the traffic data of the entire IoT ecosystem.

Declaration of interests

√ The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☐ The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Acknowledgments

This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No.2018-0-00644, Linux Malware Dynamic Detection & Protection Solution on Embedded Device).

References

Ashraf, Q.M., Habaebi, M.H., 2015. Autonomic schemes for threat mitigation in Internet of Things. J. Netw. Comput. Appl. 49, 112–127.
Benet, J., 2014. Ipfs-content addressed, versioned, p2p file system arXiv preprint arXiv:1407.3561.
bitfly.at, “Etherchain,” Charts - etherchain.org - the ethereum blockchain Explorer. [Online]. Available: https://www.etherchain.org/[Accessed: 30-Nov-2018].
Buterin, V., 2014. A Next-Generation Smart Contract and Decentralized Application Platform. *white paper*, pp. 1–36.
Butun, I., Kantarci, B., Erol-Kantarci, M., Jun 2015. Anomaly detection and privacy preservation in cloud-centric Internet of Things. In: Proc. IEEE International Conference on Communication Workshop (ICCW), pp. 2610–2615.
Chiang, M., Ha, S., I, C.-L., Risso, F., Zhang, T., 2017. Clarifying fog computing and networking: 10 questions and answers. IEEE Commun. Mag. 55 (4), 18–20.
Diro, A.A., Chilamkurti, N., 2018. Distributed attack detection scheme using deep learning approach for Internet of Things. Future Gener. Comput. Syst. 82, 761–768.
Ergun, H., Akyuz, Y.C., Sert, M., Liu, J., 2016. Early and late level fusion of deep Convolutional neural networks for visual concept recognition. Int. J. Semantic Comput. (IJSC) 10 (03), 379–397.
Feichtenhofer, C., Pinz, A., Zisserman, A., 2016. Convolutional two-stream network fusion for video action recognition. In: Proc. 2016 IEEE Conference on Computer Vision and Pattern Recognition. CVPR), pp. 1933–1941.
Hosseinpour, F., Amoli, P.V., Plosila, J., Hämäläinen, T., Tenhunen, H., 2016. An intrusion detection system for fog computing and IoT based logistic systems using a smart data approach. International Journal of Digital Content Technology and its Applications 10 (5), 34–46.
Kalkan, K., Zeadally, S., 2018. Securing internet of Things with software defined networking. IEEE Commun. Mag. 56 (9), 186–192.
Karnouskos, S., 2018. Self-driving car acceptance and the role of Ethics. IEEE Trans. Eng. Manag. 1–14.
Khan, M.A., Salah, K., 2018. IoT security: review, blockchain solutions, and open challenges. Future Gener. Comput. Syst. 82, 395–411.
Kosba, A., Miller, A., Shi, E., Wen, Z., Papamanthou, C., May 2016. Hawk: the blockchain model of Cryptography and privacy-preserving smart contracts. In: Proc. 2016 IEEE Symposium on Security and Privacy (SP), pp. 839–858.
M. Team. “Mininet walkthrough,” Mininet: an instant virtual network on your laptop (or other PC) - Mininet [Online]. Available: http://mininet.org/walkthrough/. (Accessed 30 November 2018).
Neverova, N., Wolf, C., Taylor, G., Nebout, F., Jan. 2016. ModDrop: adaptive multi-modal gesture recognition. IEEE Trans. Pattern Anal. Mach. Intell. 38 (8), 1692–1706.
Novo, O., 2018. Blockchain meets IoT: an architecture for scalable access management in IoT. IEEE Internet of Things Journal 5 (2), 1184–1195.
Noxrepo, 23-Nov-2017. noxrepo/pox,” GitHub [Online]. Available: https://github.com/noxrepo/pox. (Accessed 30 November 2018).
Oraclize, 07-Jul-2018. oraclize/ethereum-bridge,” GitHub [Online]. Available: https://github.com/oraclize/ethereum-bridge. (Accessed 30 November 2018).
Pajouh, H.H., Javidan, R., Khayami, R., Ali, D., Choo, K.-K.R., 2016. A two-layer dimension reduction and two-tier classification model for anomaly-based intrusion detection in IoT backbone networks. IEEE Transactions on Emerging Topics in Computing, 1–1.
Rathore, S., Park, J.H., 2018. Semi-supervised learning based distributed attack detection framework for IoT. Appl. Soft Comput. 72, 79–89.
Rathore, S., Sharma, P.K., Park, J.H., 2017. XSSClassifier: an efficient XSS attack detection approach based on machine learning classifier on SNSs. Journal of Information Processing Systems 13 (4), 1014–1028.
Rehman, M.H.U., Ahmed, E., Yaqoob, I., Hashem, I.A.T., Imran, M., Ahmad, S., 2018. Big data analytics in industrial IoT using a Concentric computing model. IEEE Commun. Mag. 56 (2), 37–43.
Sharma, P.K., Singh, S., Jeong, Y.-S., Park, J.H., 2017. DistBlockNet: a distributed blockchains-based secure SDN architecture for IoT networks. IEEE Commun. Mag. 55 (9), 78–85.
Sharma, P.K., Ryu, J.H., Park, K.Y., Park, J.H., Park, J.H., Oct. 2018. Li-Fi based on security cloud framework for future IT environment. Human-centric Computing and Information Sciences 8 (1).
Sharma, P.K., Rathore, S., Jeong, Y.-S., Park, J.H., 2018. Energy-efficient distributed network architecture for edge computing. IEEE Commun. Mag. 2–9.
Tavallaee, M., Bagheri, E., Lu, W., Ghorbani, A.A., Jul. 2009. A detailed analysis of the KDD CUP 99 data set. In: Proc. 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, pp. 1–6.
O. D. Team, oraclize API [Online]. Available. http://www.oraclize.it/–. (Accessed 30 November 2018).
Trufflesuite, 29-Nov-2018. “trufflesuite/truffle,” GitHub [Online]. Available: https://github.com/trufflesuite/truffle. (Accessed 30 November 2018).
Vielzeuf, V., Pateux, S., Jurie, F., Nov. 2017. Temporal multimodal fusion for video emotion classification in the wild. In: Proc. 19th ACM International Conference on Multimodal Interaction - ICMI 2017, pp. 569–576.
Wu, H., Wang, W., 2018. A game theory based collaborative security detection method for internet of Things systems. IEEE Trans. Inf. Forensics Secur. 13 (6), 1432–1445.
Yu, W., Liang, F., He, X., Hatcher, W.G., Lu, C., Lin, J., Yang, X., 2018. A survey on the edge computing for the Internet of Things. IEEE Access 6, 6900–6919.
Zarpelão, B.B., Miani, R.S., Kawakani, C.T., de Alvarenga, S.C., 2017. A survey of intrusion detection in Internet of Things. J. Netw. Comput. Appl. 84, 25–37.
Deep learning,” Deep learning. [Online]. Available: http://www.deeplearningbook.org./. [Accessed: 30-Nov-2018].
ethereum/wiki,” GitHub. [Online]. Available https://github.com/ethereum/wiki/wiki/JavaScript-API. [Accessed: 30-Nov-2018]
IoT: Number of Connected Devices Worldwide 2012-2025,” *Statista*. [Online]. Available: https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/. [Accessed: 03-Dec-2018]

Mr. Shailendra Rathore He is a research scholar in the Department of Computer Science at Seoul National University of Science and Technology (SeoulTech.), Seoul, South Korea. He works in the Ubiquitous Computing & Security Research Group under the supervision of Prof. Jong Hyuk Park. Prior to beginning the Ph.D. program, he worked as a researcher at Crompton Greaves Global R&D Center, Mumbai, India. He received his master's degree in Computer Science from the Thapar University, India in 2014. His current research interests are focused on the areas of Artificial intelligence, Blockchain, Information security, and IoT. He is the author of various top journal and magazine articles in the field of computer science. He is reviewer of IEEE Transaction of Industrial Informatica, IEEE wireless communication magazine, IEEE Transactions on Network and Service Management, and FGCS.

Mr. Byung Wook Kwon is M.S. student in Seoul National University of Technology, Seoul, Korea. Currently, He is working in Ubiquitous Computing Security (UCS) Lab under the guidance of Professor Jong Hyuk Park. His research interests include information and cybersecurity, cloud computing, IoT, network security, and artificial intelligence. Prior to joining M.S., he received B.Tech from SeoulTech., Graduated from Department of Computer Engineering, Software Engineering, Dongseo University.

Dr. James J. (Jong Hyuk) Park received Ph.D. degrees in Graduate School of Information Security from Korea University, Korea and Graduate School of Human Sciences from Waseda University, Japan. From December 2002 to July 2007, Dr. Park had been a research scientist of R&D Institute, Hanwha S&C Co., Ltd., Korea. From September 2007 to August 2009, He had been a professor at the Department of Computer Science and Engineering, Kyungnam University, Korea. He is now a professor at the Department of Computer Science and Engineering and Department of Interdisciplinary Bio IT Materials, Seoul National University of Science and Technology (SeoulTech), Korea. Dr. Park has published about 200 research papers in international journals and conferences. He has been serving as chair, program committee, or organizing committee chair for many international conferences and workshops. He is a steering chair of international conferences – MUE, FutureTech, CSA, CUTE, UCAWSN, World IT Congress-Jeju. He is editor-in-chief of Human-centric Computing and Information Sciences (HCIS) by Springer, The Journal of Information Processing Systems (JIPS) by KIPS, and Journal of Convergence (JoC) by KIPS CSWRG. He is Associate Editor / Editor of 14 international journals including JoS, JNCA, SCN, CJ, and so on. In addition, he has been serving as a Guest Editor for international journals by some publishers: Springer, Elsevier, John Wiley, Oxford Univ. press, Emerald, Inderscience, MDPI. He got the best paper awards from ISA-08 and ITCS-11 conferences and the outstanding leadership awards from IEEE HPCC-09, ICA3PP-10, IEE ISPA-11, PDCAT-11, IEEE AINA-15. Furthermore, he got the outstanding research awards from the SeoulTech, 2014. His research interests include IoT, Human-centric Ubiquitous Computing, Information Security, Digital Forensics, Vehicular Cloud Computing, Multimedia Computing, etc. He is a member of the IEEE, IEEE Computer Society, KIPS, and KMMS.