Blockchain-Based Programmable Fog Architecture for Future Internet of Things Applications

Sharmistha Nayak[1], Nurzaman Ahmed[2], Sudip Misra[3], and Kim-Kwang Raymond Choo[4]

[1]School of Nano Science & Technology
[2,3]Department of Computer Science & Engineering
Indian Institute of Technology, Kharagpur, India-721302
[4]Department of Information Systems and Cyber Security
University of Texas at San Antonio, San Antonio, TX 78249-0631, USA
[1]nayak.sharmistha@iitkgp.ac.in, [2]nurzaman@cse.iitkpg.ac.in, [3]sudipm@iitkgp.ac.in
[4]raymond.choo@fulbrightmail.org

***Abstract*—In this paper, we propose a secure programmable fog architecture for future Internet of Things (IoT) applications. The programmable feature in the proposed architecture enables us to achieve additional flexibility and support changes over the fog nodes deployed on a large scale network, where individual fog nodes are managed by the centralized fog controller. Specifically, the exchange of programmable content between the controller and the nodes is secured using blockchain. The performance evaluation of the proposed scheme shows significant improvements in reducing downtime and increasing reliability, as compared to conventional fog computing schemes.**

***Index Terms*—IoT, Blockchain, Fog Computing, Programmable Fog, SDN**

I. Introduction

Fog computing is fast becoming a norm, particularly in timely or latency-sensitive applications. There have also been recent attempts to integrate fog computing with other consumer technologies, such as Internet of Things (IoT) [1], [2]. However, challenges such as programmability and security have yet to be fully addressed in the current literature [3], [4]. For example, in a fog-based industrial IoT (IIoT) setting, we need to ensure the system has the capability to support machine configuration changes and updates, monitoring of system/network data/status (e.g., to detect failing IoT devices or other components, or potential attacks), management of sensor and actuator nodes, data exchanges between controllers and locally installed fog nodes and machines, etc. Besides, fog nodes are typically tasked to perform (preliminary) data analytics on the received data, and a range of other activities that require information exchange [5]. These activities reinforce the importance of robust programmability and secure data exchange technique. In addition, to facilitate future audits or forensic investigation, a trusted transaction log (that records the transactions associated with the various fog devices) should also be maintained.

There have been efforts dedicated to designing different architectures and solutions to address the different challenges in a fog computing deployment [6]. Large amount of unprotected data in IoT infrastructure demands for precise security system models in IoT deployments [7]. One particular research trend is to design blockchain-based solutions [8], partly due to its underpinning properties (e.g., decentralization, transparency, immutability). For example, such solutions maintain a distributed ledger among participating peer nodes in a P2P network in the form of a chain or sequence of individual units called blocks to keep a log of the accepted transactions. Each block is related to the previous block by its hash value, and thus any modifications can be detected.

While software defined networking (SDN) [9] is another viable solution, the programmability in SDN is limited to the network's dataplane. In other words, the programmability feature in fog computing can be used to achieve more efficient management for a task, resource, Quality of Service (QoS) in dynamic IoT (and IIoT) applications. Both runtime and dynamic reconfiguration of the fog nodes by remote programmability are critical tasks; therefore, it is important to ensure security while programming the nodes. Thus, we explore the utility of blockchain in this paper.

Specifically, we design a blockchain-based fog architecture for dynamic IIoT settings. In the proposed architecture, a central cloud/controller centrally manages all fog nodes, and remotely reprograms and re-configures new policy changes according to user/system requirements – see Section III. We also describe and discuss the performance evaluation of the proposed architecture in Section IV. In the next section, we will briefly review the related literature.

II. Related Literature

In the current fourth industrial revolution (also referred to as Industry 4.0), different technologies (e.g., 5G, IIoT and blockchain) have been integrated to achieve high-end provisioning of different industrial functionalities in a broad range of applications. One particular trend is extending fog computing to support Industry 4.0 applications, for example to facilitate local processing and minimizing latency for control and time-critical actions (e.g., through actuators and robots) – see also Table I. Fog computing can serve as a micro or

For personal use only

nano data center, a cloudlet, an edge device in the vicinity of the industrial environment. Sharma et al. [10], for example, proposed an SDN-based architecture for the distributed fog network. Blockchain has also been integrated in fog-based system to achieve other functionalities [11].

TABLE I
EXISTING FOG- AND CLOUD-BASED SOLUTIONS FOR IIOT: A COMPARATIVE SUMMARY

Works	Security	Programmabilty	Dynamicity	Fog
[12], [13]	✗	✗	✗	✗
[14]	✓	✗	✓	✗
[15]	✗	✓	✓	✗
[16]	✓	✗	✓	✓
[17]	✗	✗	✓	✓
[11]	✓	✗	✗	✓
This Work	✓	✓	✓	✓

Existing solutions are generally designed to provide dynamicity in task scheduling, resource allocation, actuation strategies, and management functionalities within the fog node's pre-programmed configuration. A centralized, programmable solution for the fog nodes can benefit large scale fog-based network. However, there is the single point of failure / attack risk, since the control lies with some central entity. Hence, this reinforces the importance of distributed systems to minimize such risks (inherent of centralized architectures); thus, our choice of blockchain in this paper. Specifically, using blockchain a copy of all transactions are distributively present in all fog nodes.

However, designing a secure programmable fog architecture, particularly in the presence of dynamic IoT and IIoT system requirements, is a challenging task.

III. PROGRAMMABLE AND SECURE FOG ARCHITECTURE

Here, we present our programmable fog architecture for supporting the dynamic requirements in IoT and IIoT systems. In the proposed architecture, the fog controller centrally manages the fog/edge nodes for placing, programming, and reconfiguring different rules according to current requirements. As the fog nodes may have limited computation and storage capabilities, it is essential to incorporate the programmability feature into the fog nodes.

As shown in Fig. 1, the proposed architecture is similar to a normal cloud-fog architecture, with the exception of the programmable feature that is extended from the cloud/controller. The controller can remotely and dynamically reprogram the fog nodes based on the (changing) requirements. Security is always a key consideration in any system, including blockchain- and fog-based systems. Transactions while creating or reconfiguring task, resource, actuation, and management can be considered as related parameters involved in programmable fog. A private Blockchain is integrated across the network of fog nodes for different transactions involved in a programmable cloud-fog architecture.

A. Programmable Fog Architecture

As IoT deployments can be dynamic, as previously discussed,, we can periodically or as per demand schedule the requested services at the fog nodes. A central controller takes charge of delivering different tasks at the fog node, as per the controller schedule/request from fog nodes. The task running on the fog nodes is on-demand from the device-level network, based on the application context. It is, however, not always possible to have a skilled professional to continuously monitor the deployed nodes / devices and attend to the changes physically. Hence, there is a need to add a programmable feature to fog nodes, in order to facilitate the dynamic management of fog nodes.

Fog nodes are programmed by passing arguments through the programmable application programming interfaces (APIs). For example, there are different types of tasks performed by the fog nodes at different times. The same fog is allocated to various tasks on-demand by changing the argument parameter in the programmable API. The addressed programmable feature includes different types of services.

We will discuss the services based on a generalized industrial use case here. This can include services like adopting the best possible actuation strategy under different situations to control either the machines or to meet the industrial safety requirements. We may also need to run different monitoring services like machine health, production management, local filtering of data before sending it to the cloud, and other various analytic services. The programmable API can be a simple transmission control protocol (TCP) socket or some heavy orchestration tools.

In the present work, we used client-server-based TCP sockets to manage the programmable feature in the fog nodes. Table I presents the comparative summary of existing fog and cloud-based solutions for IoT deployments. Unlike these existing approaches, our proposed architecture enables programmabilty in fog nodes (e.g., dynamic reprograming and reconfiguation of fog nodes features). Along with the programmable feature, the proposed architecture utilizes blockchain to allow secure data transmission. Thus, we achieve QoS parameters like security, programmability, dynamicity.

B. A Blockchain-based Secure Fog

As previously discussed, we utilize a private blockchain to facilitate the secure exchange of data (e.g., programmable contents) in the proposed architecture. There are two types of transactions involved, namely: the program request from fog node to controller, and program delivery to requested/scheduled fog node. The blockchain ledger is distributively maintained in all the participating nodes, and every information or data exchange is recorded in the form of unique transactions in the blocks. A transaction here contains information like timestamp, unique program ID and recipient fog ID. The recipient node verifies each digitally signed transaction received from the sender node. Only upon successful verification will the ledger be updated with a new transaction. This process repeats until the maximum permissible storage

For personal use only

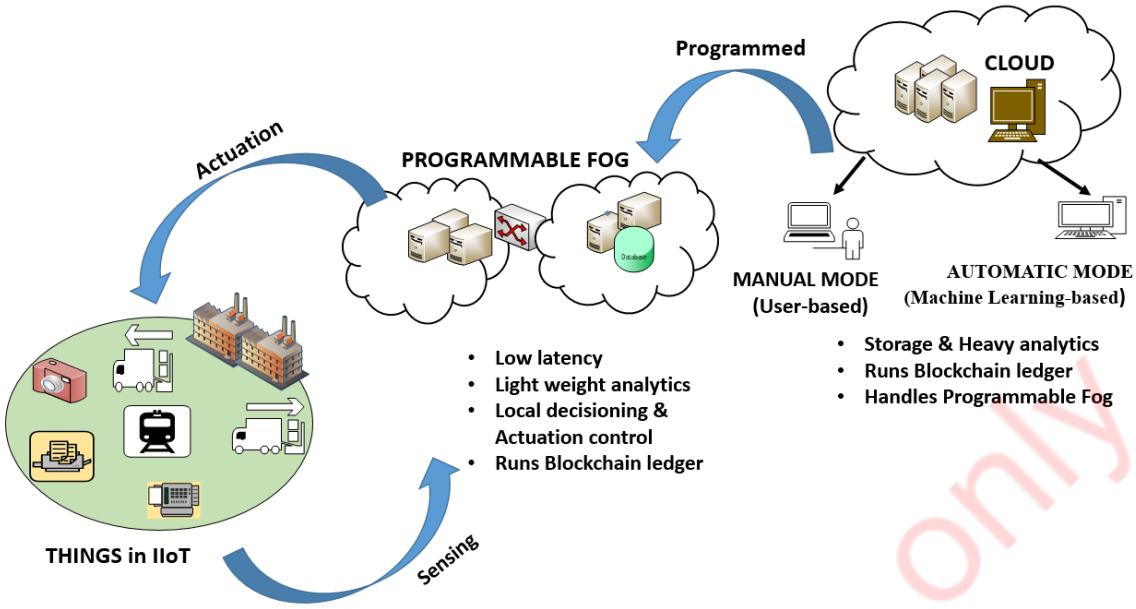

Fig. 1. Proposed Programmable Fog architecture

capacity of the ledger is reached. The participating miners work to generate a unique hash key and show their proof of work in order to form a block, which is then added to the blockchain.

Tampering a single transaction can only be possible if the hash value of all the blocks in a Blockchain can be tampered with, which is a time-consuming and expensive process. In addition, each computing node has a copy of the blocks of transactions recorded. Hence, it is computationally challenging for an attacker to modify any data without detection, since all the other nodes in the network have a local copy of the blockchain. This also facilitates auditing and traceability. We also remark that in such a design, if the controller is replaced, the new controller can download the entire history from the blockchain. Fig. 2 presents the blockchain-based programmable fog network.

Algorithm1 presents the program at the controller, where the controller first selects a node from the set of available fog nodes F_x. It checks if there is a request for any changes in the program demanded from the selected fog. If the condition is fulfilled, the desired program or programming parameter P_l is sent to the selected fog through some API. Before sending, the controller digitally signs the transaction. In case the node is found to be faulty, the task running at the faulty node will be sent to another available fog node.

A service request can also be sent to the concerned service provider using a smart contract, which is presently not included in this paper. If the controller does not receive a positive acknowledgment from the receiver node, it re-initiates the service associated with the transaction. This process repeats for all fog nodes in periodically.

Algorithm 2 describes the computations at the fog nodes. Fog nodes request for new program and check for acknowledgment from the controller. If there is an acknowledgment, then the fog nodes wait and receive the program P_l from the controller. It will verify the integrity of the received transaction. Following which the transaction T_n is recorded within a current block line B_l is less than maximum block size, B_{max}. Otherwise, the nodes run a consensus algorithm

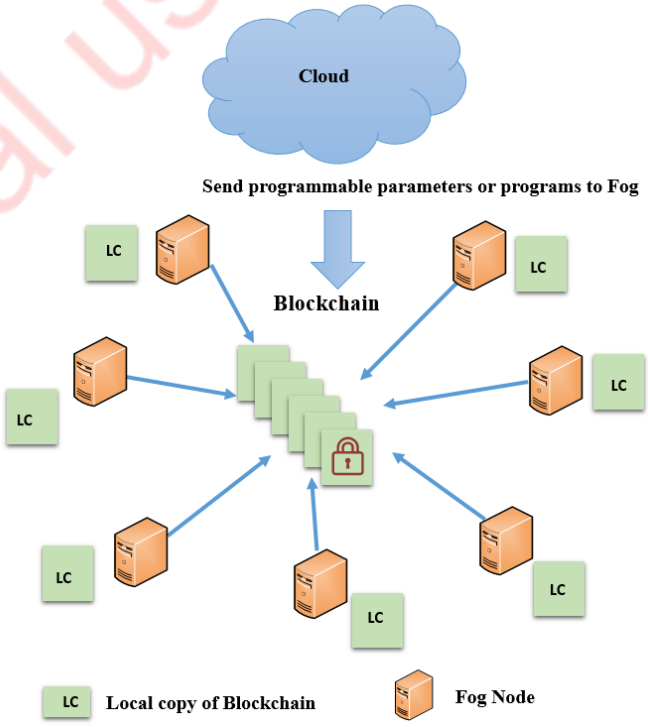

Fig. 2. A Blockchain-based Industrial Fog Network

For personal use only

Algorithm 1: Secure Programmable Controller

Input: Central controller: C_i ; Fog: F_x Programmable Parameters(P_l)
, **Output:** Program the Target Fog node

Procedure *Program target nodes*
- Select F_x
- **if** *request for change* **then**
  - choose P_l
  - Select available Fog nodes API (P_l)
    `// programmable parameters or program files are passed as arguments through API to the selected Fog node`
- **if** *fault* **then**
  - find available Fog
  - API(P_l)
- **else**
  - Repeat from step4

Digitally signs transaction
if *Recieve(positive Ack)== True* **then**
- Record(T_n)

else
- Resend(T_n)

Repeat from step 4

Algorithm 2: Secure Programmable Fog nodes

Input: Central controller: C_i ; Fog: F_x Programmable Parameters(P_l)
, **Output:** Recieve programmable parameters, Blockchain update

Procedure *Fog nodes recieve programmable parameters from cloud and runs the recieved changes.*
- **if** *RecieveAPI (P_l)* **then**
  - Broadcasts to all Fog nodes `// programmable parameters are passed as arguments through API to the Fog nodes`
- **if** $B_l < B_{max}$ **then**
  - verification(T_n)
  - Record (T_n) `// verifies recieved transaction and Records new transaction in the current block`
- **else**
  - Run Consensus Algo
  - Record (T_n) `// Records new transaction in new block`

Repeat from step 4

and then record the transaction into a new block.

IV. Performance Analysis

We evaluate the performance of our proposed approach using lab-based experiments. Our setup comprises Raspberry Pis programmed as the fog nodes, and each of these nodes runs a predefined task. A central controller is run on a personal computer with core i5 processor, in order to manage the programmable requirements of the fog nodes. In a conventional fog architecture, human intervention is required to monitor, maintain and replace fog nodes. This increases the downtime in the case of non-programmable fog architecture. We define the downtime as the time between faulty behavior of a fog node and reactivation of the normal service. Below are metrics that can be used to quantify the performance of our proposed architecture.

The downtime in the proposed architecture is calculated as:

$$T_{down} = T_{detect} + T_{prop} + T_{trans} + T_{prog} \quad (1)$$

In the above equation, $\mathsf{T}_{\mathsf{detect}}$ represents the time taken by the controller to detect any fault or failure in any fog node, $\mathsf{T}_{\mathsf{prop}}$ is the propagation time, $\mathsf{T}_{\mathsf{trans}}$ is the transmission time, and $\mathsf{T}_{\mathsf{prog}}$ is the time taken to start execution of the program at the selected fog node. For the downtime, we perform a comparative study of downtime in proposed programmable and traditional non-programmable fog architectures. In our model, upon some node failing, the controller identifies the failure and immediately programs another available fog node to run a computation with the failed node's program to maintain the availability of the latter's service. The controller also receives a request from the user to provide a new program as per needed. We repeat our study for 100 different instances.

As shown in Fig.3, our proposed architecture has significantly less and constant downtime than the traditional architecture (since downtime depends on the response time of the engineer / support staff). The significant difference in the downtime is due to the independent and non-predictive time taken by the engineer / support staff to resolve the problem. However, for programmable fog the downtime is taken as the average of the downtime over all the number of runs.

Equation 2 and Equation 3 denote the cost associated with the existing fog architecture and our proposed architecture, respectively. Specifically, the cost associated with the traditional fog-based architecture can be calculated as:

$$C_{fog} = n_1 \times (C_{fn} + C_{dep}) + n_2 \times C_{down} \times (\frac{T_{down}^{fog}}{60}) \quad (2)$$

In the above equation, C_{fn} is the unit cost price of a fog node, C_{dep} is the deployment cost, and C_{down} is the cost associated with downtime (T_{down}^{fog}, T_{down}^{prop}) of Fog nodes.

The cost function for proposed architecture is given by:

$$C_{prop} = n_1 \times (C_{fn} + C_{dep}) + n_2 \times C_{down} \times (\frac{T_{down}^{prop}}{60}) + n_3 \times C_{tr} - (n_3/B_{size}) \times C_{reward} \quad (3)$$

For personal use only

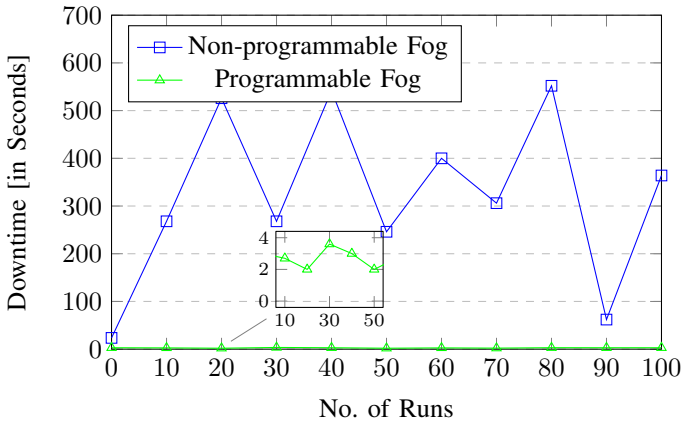

Fig. 3. Downtime comparison between programmable and non-programmable Fog

In the above equation, n_1 and n_2 denote the total number of fog nodes deployed and number of nodes that was interrupted or turned down until a given time instant. n_3 and B_{size} represent the number of transactions and block size, respectively. C_{reward} is the reward cost gained by fog nodes in the blockchain network after every successful mining and C_{tr} is the cost per unit transaction.

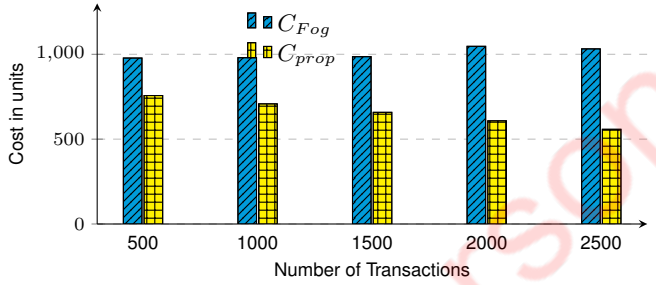

Fig. 4. Cost computation between proposed programmable and traditional Fog architecture

The cost associated with the proposed architecture is then evaluated, and a comparative cost analysis is shown in Fig. 4. We consider $C_{fn} = 30$, $C_{deploy} = 10$, $C_{down} = 5$, $C_{tr} = 0.25$, and $C_{reward} = 2$ as respective cost prices for the simulation setup. The cost factor associated with the proposed architecture is much lower than the existing fog architecture. In the present application of a private blockchain network, we do not consider prioritization transactions. Hence, each transaction costs minimal gas price per transaction (C_{tr}) which is fixed. The total effective cost in the proposed architecture is much lower than the traditional fog architecture, due to cumulative increase in reward price over every transaction and significant reduction in loss incurred due to downtime. With increased transactions and minimal downtime, annual profit also increases.

To quantify utility and benefits of having a programmable fog architecture, we will take a simple parameterized service

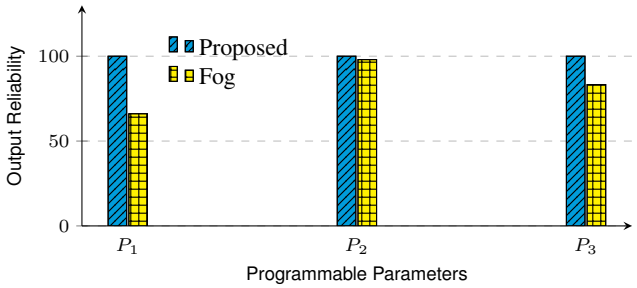

Fig. 5. Reliability check in programmable and traditional Fog architecture

model for the proposed programmable fog architecture. Let $S = f(a, b, c)$ represents a service function, where, a, b, c are factors affecting the service function. P= $[P_1, P_2, P_3]$ represent the programmable parameters associated with the factors a, b and c respectively. These factors depend on the programmable parameters in different ways (P_1-square, P_2-inverse, P_3-square root). Figure 5 shows the comparative analysis of output reliability for the model mentioned above between the proposed architecture and traditional fog architecture. With confirmed reconfiguration before execution of service, the proposed scheme achieves 100 percent reliability. In some scenarios, effective modification of these parameters is an essential requirement for achieving better services. In such situations, the traditional fog architecture fails to deliver more accurate output performance/service to the user.

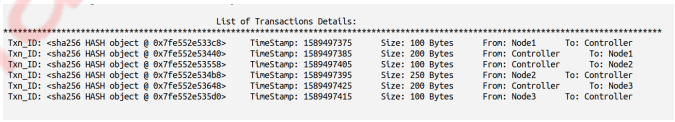

Fig. 6. List of transaction details in a block ledger

Fig. 7. An example instance of a Blockchain contents within the distributed nodal network

Finally, we demonstrate the security benefits associated with the use of blockchain. In the proposed architecture, all transactions (e.g., programmable parameter via the API; see also Fig. 6) are made through a private blockchain network. An example instance of a blockchain can be seen in Fig. 7, where we achieve data confidentiality, integrity, and availability. The unique program ID ensures confidentiality as only the concerned nodes know the details and the original program

For personal use only

is not revealed to other nodes in the network. Immutable recording of data in the blockchain ensures data integrity. No external attacker can successfully tamper with the data in the blockchain. Due to the distributed nature of the blockchain, availability of information regarding the transactions (programmable data content) is ensured whenever required. In case of any failure in the fog node or the controller, a copy of the Blockchain data is available with other nodes in the P2P network.

V. Conclusion

We proposed a new concept of programmable fog network in an IIoT setting. The proposed scheme can provide a wide range of flexibility, dynamicity, and programmability for performing different services with relatively few number of fog nodes. This is because the same fog nodes perform as a separate function as per demand. The programmability feature of fog allows for fog nodes to be programmed from a centralized controller. This work also uses ledger-based blockchain at all fog nodes that provide high-end security to protect the programmable contents shared between the controller and the fog nodes. This eases auditing and ensures trustworthiness in industrial applications.

In the future, we aim to integrate machine learning algorithms to automate the programmable feature in the proposed approach.

VI. Acknowledgement

This work was partially supported by the research grant obtained from University Grants Commission (UGC)-UK India Education Research Initiative (UKIERI) Joint Research Programme (UKIERI-III) (Grant No: 184-17/2017(IC))

References

[1] Y. Jie, C. Guo, K. R. Choo, C. Z. Liu, and M. Li, "Game-theoretic resource allocation for fog-based industrial internet of things environment," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 3041–3052, 2020.

[2] C. Perera, Y. Qin, J. C. Estrella, S. Reiff-Marganiec, and A. V. Vasilakos, "Fog computing for sustainable smart cities: A survey," *ACM Computing Surveys (CSUR)*, vol. 50, no. 3, p. 32, 2017.

[3] M. Peng, S. Yan, K. Zhang, and C. Wang, "Fog-computing-based radio access networks: Issues and challenges," *IEEE Network*, vol. 30, no. 4, pp. 46–53, 2016.

[4] S. Homayoun, A. Dehghantanha, M. Ahmadzadeh, S. Hashemi, R. Khayami, K. R. Choo, and D. E. Newton, "DRTHIS: deep ransomware threat hunting and intelligence system at the fog layer," *Future Gener. Comput. Syst.*, vol. 90, pp. 94–104, 2019.

[5] A. Kumari, S. Tanwar, S. Tyagi, N. Kumar, R. M. Parizi, and K. R. Choo, "Fog data analytics: A taxonomy and process model," *J. Netw. Comput. Appl.*, vol. 128, pp. 90–104, 2019.

[6] C. Mouradian, D. Naboulsi, S. Yangui, R. H. Glitho, M. J. Morrow, and P. A. Polakos, "A comprehensive survey on fog computing: State-of-the-art and research challenges," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 1, pp. 416–464, 2017.

[7] P. Sarigiannidis, E. Karapistoli, and A. A. Economides, "Modeling the internet of things under attack: A g-network approach," *IEEE Internet of Things Journal*, vol. 4, no. 6, pp. 1964–1977, 2017.

[8] G. Kumar, R. Saha, M. K. Rai, R. Thomas, and T.-H. Kim, "Proof-of-work consensus approach in blockchain technology for cloud and fog computing using maximization-factorization statistics," *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 6835–6842, 2019.

[9] K. Kirkpatrick, "Software-defined Networking," *Communications of the ACM*, vol. 56, no. 9, pp. 16–19, 2013.

[10] P. K. Sharma, M.-Y. Chen, and J. H. Park, "A software defined fog node based distributed blockchain cloud architecture for IoT," *IEEE Access*, vol. 6, pp. 115–124, 2017.

[11] E. N. Lallas, A. Xenakis, and G. Stamoulis, "A generic framework for a Peer to Peer Blockchain based Fog Architecture in Industrial Automation," in *South-East Europe Design Automation, Computer Engineering, Computer Networks and Social Media Conference (SEEDA-CECNSM)*. IEEE, 2019, pp. 1–5.

[12] N. Jazdi, "Cyber physical systems in the context of Industry 4.0," in *International conference on automation, quality and testing, robotics*. IEEE, 2014, pp. 1–4.

[13] O. Chenaru, A. Stanciu, D. Popescu, V. Sima, G. Florea, and R. Dobrescu, "Open cloud solution for integrating advanced process control in plant operation," in *Mediterranean Conference on Control and Automation (MED)*. IEEE, 2015, pp. 973–978.

[14] M. N. Sishi and A. Telukdarie, "Implementation of industry 4.0 technologies in the mining industry: A case study," in *International Conference on Industrial Engineering and Engineering Management (IEEM)*. IEEE, 2017, pp. 201–205.

[15] Y.-W. Ma, Y.-C. Chen, and J.-L. Chen, "SDN-enabled network virtualization for industry 4.0 based on IoTs and cloud computing," in *International conference on advanced communication technology (ICACT)*. IEEE, 2017, pp. 199–202.

[16] M. Aazam, S. Zeadally, and K. A. Harras, "Deploying fog computing in industrial internet of things and industry 4.0," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 10, pp. 4674–4682, 2018.

[17] L. Chen, P. Zhou, L. Gao, and J. Xu, "Adaptive fog configuration for the industrial Internet of Things," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 10, pp. 4656–4664, 2018.

For personal use only