Runtime and Routing Security Policy Verification for Enhanced Quality of Service of IoT Networks

Konstantinos Papachristou, Traianos Theodorou, Stavros Papadopoulos,
Aikaterini Protogerou, Anastasios Drosou, Dimitrios Tzovaras
Centre for Research & Technology Hellas, Thessaloniki, Greece
{*kostas.papachristou, theodorou, spap, k.protogerou, drosou, dimitrios.tzovaras*}*@iti.gr*

***Abstract*—The Internet of Things (IoT) is growing rapidly controlling and connecting thousands of devices every day. The increased number of interconnected devices increase the network traffic leading to energy and Quality of Service efficiency problems of the IoT network. Therefore, IoT platforms and networks are susceptible to failures and attacks that have significant economic and security consequences. In this regard, implementing effective secure IoT platforms and networks are valuable for both the industry and society. In this paper, we propose two frameworks that aim to verify a number of security policies related to runtime information of the network and dynamic flow routing paths, respectively. The underlying rationale is to allow the operator of an IoT network in order to have an overall control of the network and to define different policies based on the demands of the network and the use cases (e.g., achieving more secure or faster network).**

***Keywords*-security policy; quality of service; software-defined networking; internet of things**

I. Introduction

The recent technological development of Internet of Things (IoT) enables different devices and systems with computing and sensorial capabilities to collect and exchange data, thus becoming an integral part of the Internet [1]. IoT fosters the development of technologies and applications that enable easy access and interaction between a wide variety of devices (e.g., surveillance cameras, monitoring sensors, actuators, vehicles, etc) and effective processing of large amount and variety of generated data. Such services are used by citizens and companies in many different domains, such as industrial automation, smart health-care, intelligent transportation systems and smart energy systems [2]–[4].

In the traditional networks, the networking devices use different complex rules that cannot be modified in real-time, resulting in a static, decentralized and very complex networking infrastructure. To manage the above limitations, the Software-Defined Networking (SDN) is used [5] that facilitates network management and allows a dynamic network configuration and central control of the network in order to achieve higher network performance and monitoring capabilities. More specifically, while in the traditional networks each forwarder uses static routing tables (control plane) that are locally maintained in order to learn where the data packets (data plane) will be directed, the SDN technology separates these two planes, leaving the transportation of data to the forwarder and using the SDN controller to direct the data packets.

While SDN infrastructure is responsible for the data routing between IoT devices, Fog computing [6] provides the necessary infrastructure such that the enormous amounts of data are stored, processed and presented in a seamless and efficient way. It consists of Fog nodes near to the SDN forwarders and IoT devices which provide computational capabilities (e.g., virtual servers) to host IoT applications for data monitoring, storage, delivery, analysis and visualization. The Fog cloud is the main cloud infrastructure of the IoT network that communicates with all the Fog nodes.

One of the main challenges of an IoT network is to provide high security and Quality of Service (QoS) to the IoT users and applications [7]. For instance, there should be guarantee that the network data are accessible to authorized users only, or the network must not be influenced by denial-of-service attacks where a high demand for energy resources (e.g., by transmitting high volumes of packet) can lead to resource leakages and overloading. Various security policy framework have been proposed that enable network operators to define their own policies for improved security on the network management. In [8], a QoS policy enforcement framework has been proposed that enforces QoS in an OpenFlow-based network. In [9], the network operator is able to manage security policies, related to control rules access, defining priorities, delegating rights, etc. A security-related policy framework was presented in [10], which allows the network operator to create security policies using human-readable language. Finally, a theoretical analysis of policy-based management frameworks was proposed in [11].

A. Motivation

Cyber or physical attacks on IoT network infrastructures aim to affect the QoS and the security of them, while the increased amount of network traffic produced by the interconnected devices leads to energy efficiency problems of the network [12]. Given that the IoT networks allow the communication between devices through time-varying communication links (i.e. flow routing paths), the selected flow routing path affects the performance of the network. This dynamic character of the IoT network gives the opportunity

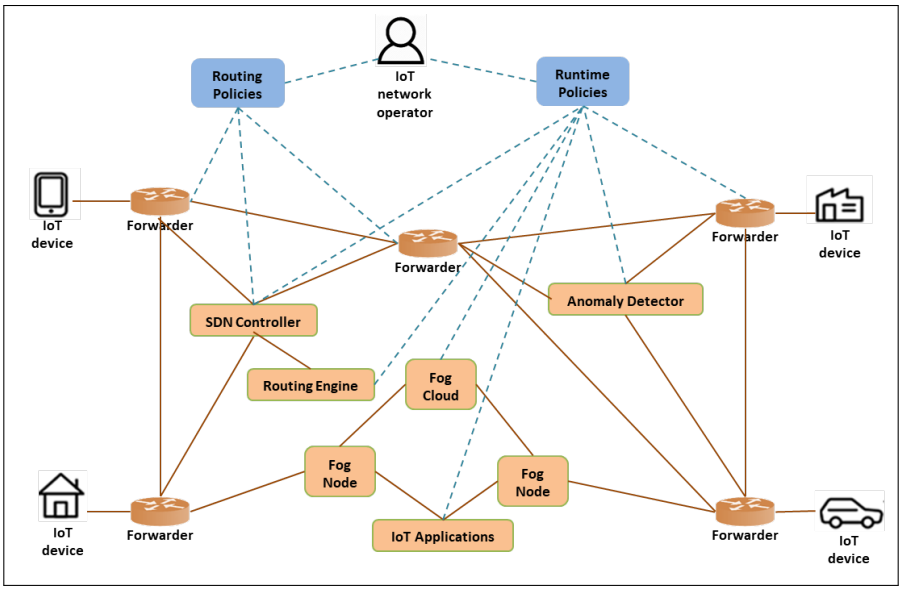

Figure 1. An IoT network enables different devices to collect and exchange data through forwarders. The routing engine generates the flow rules of the network which are used by the SDN controller to send the data packets to the forwarders accordingly. Fog computing (i.e. Fog nodes and cloud) manages and orchestrates the IoT applications that support data monitoring, storage, delivery, analysis and visualization. Anomaly detector performs attack detection and takes the appropriate mitigation actions. In this paper, we propose a number of policies related to the runtime behavior and flow routing of IoT network which ensure confidentiality, integrity and availability awareness. These policies enable the IoT network operator to have an overall control of the network and to define different policies based on the demands of the network and the use cases.

to the operator to use different policies based on the demands of the network and the use cases. For instance, in some cases the main priority of the operator is security of the data routing over the network. Alternatively, in other cases the operator aims to achieve a very fast network, without extra delays.

Additionally, the current approaches of IoT security focus mainly on the detection of attacks at specific parts of network and ignore the overall aspect of the network. However, the QoS of an IoT network also depends on the specials demands of use cases [13]. For example, the percentage of authorized users and the services run on the Fog cloud gives an degree of security, while differences between packets that have been send and received in a data flow indicate whether the integrity security property is violated. Therefore, it would be essential for the operator to be aware for the runtime behavior of the IoT network and to be able to define the constraints depending on the use cases.

B. Contribution

The recent advances of IoT security usually target only very specific security threats, whereas truly secure and reliable SDN-based networking has not yet been achieved. Moreover, the runtime behavior and dynamic flow routing of IoT networks have not yet been analyzed, to the best of our knowledge. We therefore propose two methodologies that will allow the operator of an IoT network to define and verify security policies related to the runtime behavior and flow routing of the network, as illustrated in Figure 1. Toward this end, we make the following technical contributions:

- A number of security policies for the runtime behavior and flow routing of IoT network ensuring confidentiality, integrity and availability awareness.
- A methodology to verify the security policy based on runtime information of the network.
- A methodology to verify that flow routing paths produced by the SDN routing system are aligned to the routing policy.
- A methodology to compute the optimal routing path based on a multi-objective optimization approach.

The next Sections II and III present the proposed runtime security and flow routing policy verification frameworks, respectively, while conclusions and the future plans are given in Section IV.

II. Runtime Security Policy Framework

In this section, a number of runtime security policies are defined that represent the desired behavior of IoT components in real-time (i.e. runtime monitors). For each of the runtime security policies, we use as input the relevant runtime data from the monitors as well as the respective runtime security Key Performance Indicators (KPI) constraints defined by the IoT network operator. After the verification procedure, the results of the policy verification are produced that describe the percentage of deviation for each of the KPIs. In the next subsections, an overview of the runtime monitors are given and the runtime security policies are defined in detail.

A. Runtime Monitors

The main components of an IoT network (as can be seen in Figure 1) and their runtime data used for the verification

of runtime security policies are presented below:

- **SDN controller**: It sends the data packets to the forwarders according to flow rules generated by the routing engine.
- **SDN routing engine**: It generates the flow rules of the network (i.e. path configurations for every source-destination pair of network nodes).
- **Fog cloud and nodes**: It provides the virtual infrastructure such that privileged users access appropriate services for data monitoring, storage, delivery, analysis and visualization.
- **Anomaly detector**: It performs anomaly detection and takes the appropriate mitigation actions.
- **IoT applications**: Services for data monitoring, storage, delivery, analysis and visualization services which are provided by the Fog nodes and clouds.

B. Runtime Security Policies

Based on the synthesized runtime data generated by the above runtime monitors, the following policies should be satisfied depending on the constraints defined by the IoT network operator:

1) **Percentage of difference between packets should be lower than the defined constraint**: We investigate the similarity of the data in the receiver and sender side of the SDN controller for a specific flow, so as to validate the integrity security property.
2) **Delay and inter-arrival time should be lower than the defined constraint**: We use the SDN controller information regarding the packet delay and the inter-arrival time of the packets for validation of the integrity property.
3) **Percentage of authorized flows should be greater than the defined constraint**: Using data exported by the SDN routing engine, we consider counting the percentage of authenticated flow rules in comparison with the total number of flow rules, so as to determine whether confidentiality goal has been successful.
4) **Percentage of authorized users should be greater than the defined constraint**: We use the number of authorized user and the total number of users connected to the Fog infrastructure of the IoT network, so as to validate the confidentiality property.
5) **Percentage of authorized connections should be greater than the defined constraint**: Given the fact that all devices and services should be identified and authenticated to connect to the Fog cloud and nodes, we consider counting the number of authenticated established connections on the server and comparing it to the total number of devices requesting access to it. We then calculate the percentage of authenticated devices, which compared against the KPI constraint, so as to determine whether confidentiality goal has been successful.
6) **Availability percentage in time of an attack should be greater than the defined constraint**: We use the start and end time of an attack detected by the corresponding application. We use the start time and the lifetime of the Fog infrastructure to calculate the availability percentage of Fog infrastructure during the attack. In addition, we use the start time and the lifetime of the SDN controller to calculate the availability percentage of the SDN controller during the attack, for validation of the availability property.
7) **Availability percentage in time of a physical failure should be greater than the defined constraint**: We use the start time and the lifetime of the SDN controller as well as the duration of a physical attack detected by the corresponding application in order to calculate the availability percentage of the SDN controller in this kind of attacks, so as to validate the availability property.
8) **Processing time should be lower than the defined constraint**: To ensure robust performance and availability, we compare the corresponding processing time of IoT services against the KPI constraint, so as to determine whether availability goal has been successful.

III. Flow Routing Policy Framework

This section presents the methodology followed for the verification of the routing decisions taken by the SDN infrastructure of the IoT network. The effectiveness of the routing decisions is measured with respect to the following three pillars:

- **Energy consumption**: Every communication and packet transmission in the network consumes some amount of electrical energy, measured in watts. Since each watt consumed costs money, ideally the network operator would like to optimize the traffic in the network in order to minimize the total energy consumption, and thus, have lower operational costs.
- **Quality of Service (QoS)**: The QoS is a quantitative description of the overall performance of the services in a network. The QoS depends in several aspects in the network, such as packet loss, bit rate, throughput, transmission delay, availability, jitter, etc. The higher the QoS, the better for the network.
- **Security**: Ideally, in the IoT network, the forwarders should be malware free, and the flows should not contain data related to attacks. Since this is not a realistic assumption, the flows that are sensitive must be protected from alterations and eavesdropping.

The workflow of the verification methodology is illustrated in Figure 2. Based on the topology of the network (formulated as an undirected graph), we collect real-time statistics and use them as input to the multi-objective optimization algorithm along with the routing policies. This information

is used for the calculation of routing policy KPIs which estimate how good a routing decision is. The multi-objective optimization is able to identify the set of the best solutions (i.e. flow rules) which optimize the routing policy KPIs under consideration. The set of optimal flow rules is then compared with the flow rules created by the SDN, in order to compare them and provide a deviation metric that is the actual verification result. The verification is considered successful if the deviation similarity metric is within a predefined range of values. In the next subsections, we present an overview of the data collected from the SDN system, we define the flow routing policies and finally, the multi-objective approach for extraction of the optimal flow routing is provided.

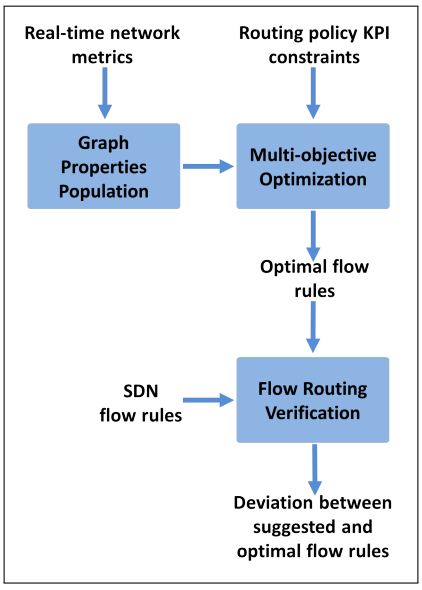

Figure 2. The workflow of the proposed flow routing policy verification methodology: based on the topology of the network (formulated as an undirected graph), and the routing policies, multi-objective optimization is applied in order to compute the optimal flow rules. These optimal rules are then compared with the flow rules of the SDN in order to identify discrepancies.

A. Real-time Network Metrics

The information collected from the SDN infrastructure of an IoT network is related to the forwarders, the connections between forwarders and data flows. Specifically, the following information is collected with respect to forwarder connections:

- **Delays of Cognitive Packets (CPs) between forwarders**: Delays between forwarders are measured by using Cognitive Packets (CPs) travelling from node to node.
- **Bandwidth of connections between forwarders**: Each connection has a static bandwidth metric, which measures the capacity to the connection with respect to the total number of bits it can serve per second.

The following information is collected with respect to each forwarder:

- **Energy usage within forwarder per packet (watts/packet)**: Every forwarder has hardware and software modules, which are able to measure energy usage. When a CP arrives in a forwarder, the forwarder energy usage is added to a list in CP's payload.
- **Confidence of forwarders**: Confidence represents the (inverse) probability that the forwarder is infected by a malware. High confidence corresponds to low probability of malware, while low confidence corresponds to high probability of malware. Malwares in the forwarder can affect the integrity and the confidentiality of the flows. In this respect, flows with high sensitivity should not use low confidence forwarders as part of the path they follow.
- **Sensitivity of forwarders**: Every forwarder has sensitivity that measures how sensitive a forwarder is with respect to different aspects of importance. Examples of sensitive forwarders include those forwarders that are central in the network, and thus, process large amounts of flows, or forwarders which are used to route sensitive (e.g. private/confidential) information. The higher the sensitivity of the forwarder is, the more we avoid low confidence flows routed through it.

The following information is collected with respect to each data flow:

- **Confidence of flows**: Confidence represents the (inverse) probability that the flow carries malicious traffic, i.e. the flow is part of an attack curried out in the IoT network. High confidence corresponds to low probability of anomalous traffic and low confidence to high probability. The flows with high confidence should not be sent through forwarders with high sensitivity, in order to limit the effect of the attack to the network.
- **Sensitivity of flows**: Each flow is characterized by a sensitivity measure. Examples of flows with high sensitivity include but are not limited to business critical, or private/confidential traffic. Flows with high sensitivity must be protected more than flows with low sensitivity. In this respect, highly sensitive flows must avoid using routes that include forwarders with low confidence.
- **Path followed by the flows**: Each flow is eventually assigned to one or more paths of forwarders over time. Each time a data packet corresponding to the flow is send, it uses the paths assigned to the flow in order to traverse the IoT network. The information regarding which paths are assigned into which flow, is important for policy verification, since highly sensitive flows should avoid low confidence forwarders, and sensitive forwarders should not be used in the routes of low confidence flows.

B. Flow Routing Policies

This subsection presents four runtime security policies that concern security, QoS and energy consumption infor-

mation related to the forwarders and the flows in the IoT network in order to ensure confidentiality, integrity and availability. In more detail, the following policies must be followed:

1) **Maximize the network QoS**: We use the delays of CPs and the bandwidth of connections between forwarders to measure the QoS with respect to the delays experienced in the network. The QoS KPI must be as large as possible.
2) **Minimize energy consumption**: The energy usage within forwarder per packet is used to measure the total energy consumed by the network. The Energy Consumption KPI should be as small as possible.
3) **Protect sensitive flows along the paths they follow by avoiding low confidence forwarders**: To quantitatively describe of how much sensitive flows use routes with low confidence forwarders, we use the sensitivity of flows and the confidence of forwarders. The Flow Security Violation KPI should be as small as possible.
4) **Protect sensitive forwarders from possible malware infections by avoiding sending low confidence flows through them**: In order to quantitatively describe of how much sensitive forwarders are included in routes serving low confidence flows, we use the confidence of flows and the sensitivity of forwarders. The Forwarder Security Violation KPI should be as small as possible.

C. Multi-objective Optimization Approach

The above optimization objectives may be contradicting, i.e. optimizing the value of one objective may be affecting negatively the values of other objectives. For example, the best route for protecting sensitive flows might be comprised of a large number of forwarders, and thus, induce an extra delay that lowers the QoS. The approach followed in order to tackle this problem is based on multi-objective optimization that tries to optimize multiple objectives simultaneously. In such cases, there are not exist a single trivial solution, but instead the multi-objective optimization identifies a set of optimal solutions. The solutions included in this set are called Pareto optimal [14]. Without additional subjective preferences regarding the significance of the optimization objectives, all the solutions within the Pareto front are considered as equally good, and there is no way to pick a single solution as the best overall one. An example of Pareto optimal solutions is illustrated in 3. In this example, two objectives must be minimized simultaneously, J_1 and J_2. S_F represents the set of all possible solutions and S_P the set of Pareto optimal solutions. Two solutions selected from the Pareto are p_1 and p_2. Solution p_1 has larger J_2 value than p_2, but also smaller J_1 value than p_2.

In the general case, the multi-objective optimization problem is formulated as follows:

$$\begin{aligned} &\arg\min_{x} && (J_1(x), J_2(x), \cdots, J_N(x)) \\ &\text{subject to} && J_i^{min} \leq J_i(x) \leq J_i^{max}, \forall i \in [1, N], x \in X, \end{aligned}$$

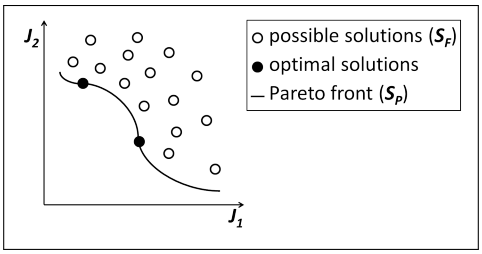

Figure 3. An example of Pareto optimal solutions [14]: Two objectives, J_1 and J_2, must be simultaneously minimized. The set of optimal solutions is represented as S_P. S_F represents the set of possible solutions.

where X is the possible set of solutions, $J_i, \forall i \in [1, N]$ are the objective functions that must be minimized simultaneously and $J_i^{min} \leq J_i(x) \leq J_i^{max}$ are optional constraints that the objectives might have. In the case of formal routing policy verification, the objectives are the four KPIs:

- Objective 1: $J_1 = -$QoS
- Objective 2: $J_2 =$Energy consumption
- Objective 3: $J_3 =$Flow Security Violation
- Objective 4: $J_4 =$Forwarder Security Violation

All the objectives have been formulated as minimization objectives. This is the reason why $J_1 = -$QoS, since QoS must be maximized, while the other KPIs must be minimized. In the case of formal verification, the possible set of solutions X is defined as the mapping of data flows to paths of forwarders. Each such mapping results in different values in the objectives (KPIs).

In case that the solution proposed by the SDN is not in the set of Pareto optimal solutions, we conclude that the policy verification failed and the degree of deviation from the optimal solution should be reported. More specifically, the deviation degree is the mean square distance of the SDN solution from the closest solution within the Pareto set. In case that the SDN solution is within the Pareto set, then this distance is zero.

D. Evaluation Scenario

To illustrate the above, we assume that there are three alternative paths (i.e. flow rules) of forwarders namely x_1, x_2 and x_3, which represent the communication between two network-connected devices, as presented in Table I. The final decision on which path is the optimal path is taken using the sensitivity and confidentiality values of the forwarders and the flow, as well as the energy consumption within the forwarders and the connection delays. More specifically, each flow rule has different flow routing KPI values corresponding to policies of subsection III-B, which must be compared in order to identify the set of Pareto optimal solutions. The flow rule x_2 has the worse values for all the KPIs, and thus, it is not part of the Pareto front. The other two flow rules, x_1 and x_3, are both in the Pareto front, since x_3 is better with respect to the Energy Consumption, Flow Security Violation and Forwarder Security Violation

KPIs, but worse with respect to QoS KPI, when compared to x_1.

As noted earlier, the selection of optimal solution depends on the special needs of the use cases of the IoT network. For instance, if the two IoT devices correspond to two automated vehicles, there is need for immediate responses so as to assure the safety of automated vehicles due to possible collisions. In such a case, the solution $x1$ should be selected, since it is the best with respect to QoS KPI. Alternatively, if the IoT network is part of a health care system, the security of personal data of the patients should be guaranteed. The solution $x3$ is the optimal in that case, considering it avoids forwarders with low confidence and high sensitivity.

Flow Rule	QoS	Energy	Flow Security	Forwarder Security
x_1	**7**	4	5	2
x_2	2	7	5	2
x_3	6	**1**	**2**	**1**

Table I
AN EXAMPLE OF ALTERNATIVE SOLUTIONS FOR A SPECIFIC DATA FLOW. EACH SOLUTION HAS DIFFERENT KPI VALUES AND CORRESPONDS TO A DIFFERENT PATH SELECTED FOR THE DATA FLOW. THE FLOW RULE x_2 HAS THE WORSE VALUES FOR ALL THE KPIS, AND THUS, IT IS NOT PART OF THE PARETO FRONT. THE OTHER TWO FLOW RULES, x_1 AND x_3, ARE BOTH IN THE PARETO FRONT, SINCE x_3 IS BETTER WITH RESPECT TO THE ENERGY CONSUMPTION, FLOW SECURITY VIOLATION AND FORWARDER SECURITY VIOLATION KPIS, BUT WORSE WITH RESPECT TO QOS KPI, WHEN COMPARED TO x_1. DEPENDING ON THE SPECIAL USE CASES OF THE IOT NETWORK (E.G., MORE SECURE OR FASTER NETWORK), THE OPTIMAL SOLUTION MAY BE DIFFERENT (E.G., x_3 OR x_1 RESPECTIVELY).

IV. CONCLUSION

In this paper, we presented two frameworks for verification of the runtime behavior and routing decisions of an IoT network. Conclusively, the proposed runtime security and flow routing policies involve security, QoS and energy information to ensure confidentiality, integrity and availability security properties. The policy constraints can be defined by the IoT network operator depending on the demands of the network and the use cases. In future work, the presented verification frameworks will be evaluated using real scenarios with heterogeneous IoT platforms and devices including domains such as surveillance, smart transport systems and flexible manufacturing. Finally, we will study how real-time network metrics used in routing policies can be utilized to detect attacks employing artificial intelligence approaches.

ACKNOWLEDGMENT

This work has been partially supported by the European Commission through project SerIoT funded by the European Union H2020-IOT-2016-2017 (H2020-IOT-2017) Program under Grant Agreement no. 780139. The opinions expressed in this paper are those of the authors and do not necessarily reflect the views of the European Commission.

REFERENCES

[1] A. Whitmore, A. Agarwal, and L. Da Xu, "The internet of things—a survey of topics and trends," *Information Systems Frontiers*, vol. 17, no. 2, pp. 261–274, Apr 2015.

[2] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, "Internet of things for smart cities," *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 22–32, Feb 2014.

[3] S. R. Islam, D. Kwak, M. H. Kabir, M. Hossain, and K.-S. Kwak, "The internet of things for health care: a comprehensive survey," *IEEE Access*, vol. 3, pp. 678–708, 2015.

[4] J. Sherly and D. Somasundareswari, "Internet of things based smart transportation systems," *International Research Journal of Engineering and Technology*, vol. 2, no. 7, pp. 1207–1210, 2015.

[5] S. Bera, S. Misra, and A. V. Vasilakos, "Software-defined networking for internet of things: A survey," *IEEE Internet of Things Journal*, vol. 4, no. 6, pp. 1994–2008, Dec 2017.

[6] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing*, ser. MCC '12, 2012.

[7] F. A. Alaba, M. Othman, I. A. T. Hashem, and F. Alotaibi, "Internet of things security: A survey," *Journal of Network and Computer Applications*, vol. 88, pp. 10 – 28, 2017.

[8] M. F. Bari, S. R. Chowdhury, R. Ahmed, and R. Boutaba, "Policycop: An autonomic qos policy enforcement framework for software defined networks," in *IEEE SDN for Future Networks and Services (SDN4FNS)*. IEEE, 2013.

[9] D. Rosendo, P. T. Endo, D. Sadok, and J. Kelner, "An autonomic and policy-based authorization framework for openflow networks," in *13th IEEE International Conference on Network and Service Management*, 2017.

[10] A. Lara and B. Ramamurthy, "Opensec: Policy-based security using software-defined networking," *IEEE transactions on network and service management*, vol. 13, no. 1, pp. 30–42, 2016.

[11] K. Sood, K. K. Karmakar, V. Varadharajan, U. Tupakula, and S. Yu, "Analysis of policy-based security management system in software-defined networks," *IEEE Communications Letters*, 2019.

[12] S. Scott-Hayward, S. Natarajan, and S. Sezer, "A survey of security in software defined networks," *IEEE Communications Surveys Tutorials*, vol. 18, no. 1, pp. 623–654, 2016.

[13] M. Conti, A. Dehghantanha, K. Franke, and S. Watson, "Internet of things security and forensics: Challenges and opportunities," *Future Generation Computer Systems*, vol. 78, pp. 544 – 546, 2018.

[14] K. Deb and D. Kalyanmoy, *Multi-Objective Optimization Using Evolutionary Algorithms*. New York, NY, USA: John Wiley & Sons, Inc., 2001.