IEEE Access
The journal for rapid open access publishing

IEEE Xplore Digital Library
http://ieeexplore.ieee.org/Xplore

# Node State Monitoring Scheme in Fog Radio Access Networks for Intrusion Detection

XINGSHUO AN[1], XING LÜ[1], LEI YANG[2], XIANWEI ZHOU[1], AND FUHONG LIN[1]*

***Abstract*—This paper studies intrusion detection for fog computing in Fog Radio Access Networks (F-RANs). As fog nodes are resource constrained, traditional intrusion detection system (IDS) cannot be directly deployed in F-RANs due to the communication overhead and computational complexity. To address this challenge, we propose a Skyline query based scheme that can analyze IDS log statistics of fog nodes and provide a complete data processing flow. Specifically, a three-step solution is proposed. First, a lightweight fog node filtering strategy (FNFS) is proposed to filter the raw data which can reduce the fog-cloud communication overhead. Second, a sliding-window-based mechanism is developed in the cloud server to efficiently process the asynchronous data flow. Then, using the pre-processed data, a set of seriously attacked nodes will be identified by Skyline query. Third, the security threat level of each individual fog node is calculated using the unascertained measure which can determine the degree of security threat. Numerical simulations show that the proposed scheme can significantly reduce communication overhead and computational complexity.**

***Index Terms*—5G, Fog computing, IDS, Skyline query, Node monitoring, FNFS**

## I. INTRODUCTION

5G needs to support high concurrency of user equipment access and guarantee its quality of service [1]. This requires the 5G network to support low latency, high mobility, high scalability, and real-time execution [2]. Fog computing extends cloud services to the network edge supporting these characteristics mentioned above [3], which provides a promising solution for 5G applications. The paradigm of F-RANs is an application scenario where fog computing can be deployed in 5G networks [4], [5].

A typical fog computing infrastructure consists of the cloud service layer, the fog service layer, and the user equipment (UE) layer [6] as illustrated in Figure 1. The UE layer is composed by the user end devices, which generate various application requests in F-RANs. The fog service layer is mainly responsible for providing services for UEs and sends necessary data to the cloud server. The cloud service layer acts as the manager of fog nodes, which provides resource scheduling and information processing services for fog nodes [7]. This three-layer structure can provide users with QoS guaranteed services.

However, compared with the cloud server, the fog nodes are resource constrained, and the fog computing platform operates in a heterogeneous environment [8], which renders many network security challenges. First, F-RANs are vulnerable to various types of network attacks from the user equipment, such as Distributed Denial-of-Service (DDoS), Remote to Local (R2L), Probing (Prob), user to root (U2R) [9]. Second, heterogeneous network environments and communication protocols can cause high security threats to F-RANs [10]. Last but not least, operating system and program vulnerabilities of fog nodes are easily exploited by intruders [11]. Therefore, security technologies need to be applied to fog computing networks to solve these challenges. Due to constrained resources [12]– [14] at the end devices, the lightweight intrusion detection system (IDS) is one of the promising methods to solve this problem [15]. Along this line, this paper aims to study a lightweight fog computing IDS framework (FC-IDS) [16], where the fog nodes perform the detection task, and the cloud server monitor the security state of the fog nodes in real time [17].

In practice, the scale of F-RANs grows increasingly, which renders a challenge for a cloud server to query all fog nodes in real time due to the high communication overhead and computational complexity [18]. To tackle this challenge, this paper proposes a priority-based Skyline query scheme to detect whether fog nodes are compromised or not. The Skyline query [19] is a multi-objective decision-making method which can identify the nodes that are not dominated by other nodes so as to find the most advantageous tuples from a multi-dimensional tuple set [20], [21]. It is worth noting that Skyline queries processing in F-RANs environment is quite different from traditional network environment. In the F-RANs environment, when Skyline queries are executed by the cloud server, the fog nodes need to transmit data (such as the network connection and fog node host state) to the cloud server, which would incur significant communication overhead [22]. Further, due to the heterogeneous networking environments, the data of each fog node may not be synchronized when communicating with the cloud server. This requires a synchronized processing step in the cloud server.

To address these challenges, we propose a fog node monitoring scheme. First, data is initially pre-processed on the fog nodes before being sent to the cloud server to reduce the fog-cloud communication overload. Then, the pre-processed data is transmitted from each fog node to the cloud server. Second, the cloud server processes the asynchronous data and outputs a real-time Skyline collection. Third, using the output of Skyline set, the security threat of each fog node can be calculated by the cloud server. The main contributions of the paper are summarized as follows:

1) In the proposed lightweight scheme, we develop a fog node filtering strategy (FNFS) on the fog node, in order to reduce the high communication overhead between fog nodes and the cloud server.

2) A sliding-window-based mechanism is proposed to efficiently process the data flow after being filtered by the fog nodes. Then, using the Skyline processing method, the

Digital Object Identifier: 10.1109/ACCESS.2019.2899017

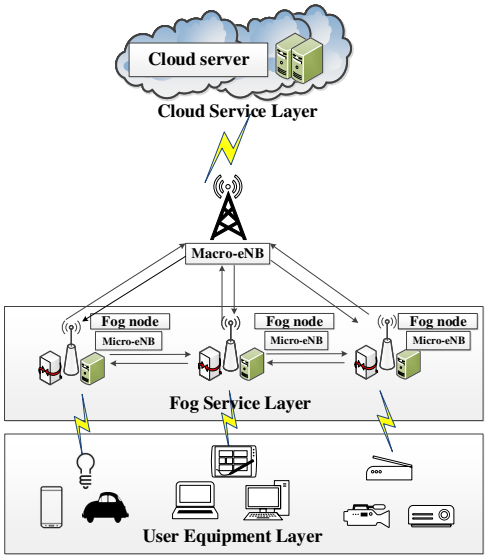



Fig. 1: The structure of fog computing-based F-RANs.

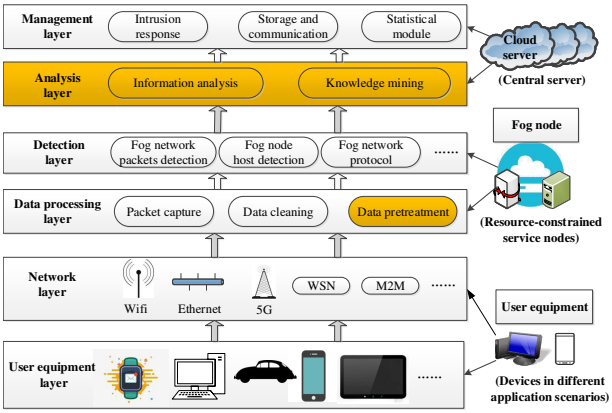



Fig. 2: The general framework for fog computing intrusion detection systems.

proposed scheme can identify a set of seriously attacked nodes.

3) Based on the outcomes of the proposed sliding-window-based mechanism, we measure the security threat of fog nodes by using the unascertained measure (UM) method [23].

The rest of the paper is organized as follows. In Section II, we review the related literature on fog security, especially fog nodes monitoring and IDS. In Section III.A, we introduce the relationship between this work and Fog IDS. We define the concept of Skyline and describe the problem of fog nodes security state monitoring in Section III.B. In Section IV, we present our proposed monitoring scheme. Numerical simulation is carried out in Section V and conclusions are drawn in Section VI.

## II. Related work

Security technologies of fog computing are forward-looking research directions. As one of the key security technologies, IDS can detect and classify a wide range of attacks against IoT sensors. Yaseen et al. in [24] proposed a model that provides a global monitoring capability for tracing moving sensors and detecting malicious ones. Zhang et al. in [25] proposed an effective data acquisition approach with the ability to filter abnormal data and meet the real-time requirement. However, these methods are not applicable for the fog nodes with the limited computing resources, which require a lightweight algorithm. A new cloudlet mesh security framework was proposed [26] to protect the mobile cloud network, which provides a host-side IDS solution. A privacy-preserving framework in fog computing IDS was proposed by Wang et al. [27], and the proposed framework based on fog devices could help reduce the workload at the cloud's side. A three-layered fog IDS was proposed in [28]. It is a distributed and lightweight IDS based on an Artificial Immune System (AIS).

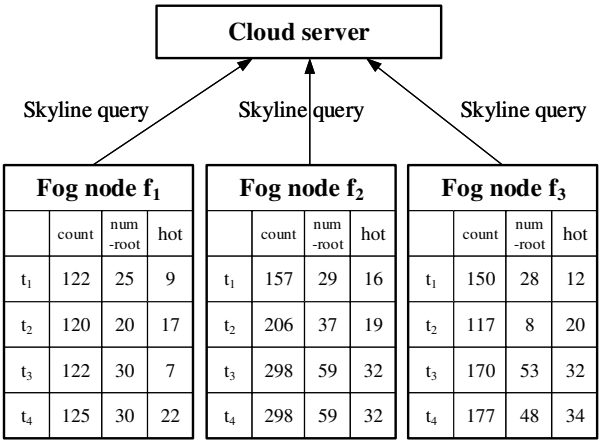
As mentioned above, some of the above works are carried out from the perspective of data processing. Some studies are applicable for an edge computing framework (Cloudlet). Few research focuses on the flow of IDS data in a fog environment. In [16], [29], we proposed an IDS framework for cloud server collaboration with fog nodes. Under this framework, we presented a new lightweight IDS algorithm called sample selected extreme learning machine (SS/ELM). The SS/ELM algorithm shows excellent performance in terms of detection accuracy. It contributes to solving the problems caused by resource constraints in fog computing. However, There are few studies on the monitoring of the security state of fog nodes. Some researchers consider the privacy of fog nodes [30], [31]. Angelopoulos et al. [32] presented an energy-based weight selection algorithm in mobile ad hoc networks (MANETs). The simulation results showed that the proposed method could reduce the monitoring overhead, improve the network performance, prolong the lifetime, and improve the security of networks. Wang et al. in [33] proposed a server-aided data stream monitoring scheme (DSM) to address the security challenges. This scheme is based on the Skyline query, and the users are able to verify the correlation scores obtained from the server. However, these methods are not designed the F-RANs environment, and do not take into account data redundancy processing and real-time performance. In contrast, this paper proposes a fog node state monitoring scheme, which accounts for data redundancy and real-time requirements of data processing.

## III. General FC-IDS description and problem formulation

In this section, we first introduce the FC-IDS framework and then discuss the node monitoring problem in this framework.

As shown in Figure 2, the FC-IDS framework can be divided into 6 layers, namely UE layer, network layer, data processing layer, detection layer, analysis layer, and management layer (The detailed description of each layer can be found in our previous work [16]). Comparing with the structure of F-RANs, the UE layer and network layer can be deployed on user equipment, the data processing layer and detection layer can be deployed on the fog nodes, and the processing analysis layer and management layer can be deployed on the cloud server. A typical working process is given as follows.

1) The fog node performs data pre-processing and sends pre-processed data to the cloud. In order to reduce the computing complexity on cloud servers, data is pre-processed first in fog




**Fog node $f_1$**

| | count | num -root | hot |
|---|---|---|---|
| $t_1$ | 122 | 25 | 9 |
| $t_2$ | 120 | 20 | 17 |
| $t_3$ | 122 | 30 | 7 |
| $t_4$ | 125 | 30 | 22 |

**Fog node $f_2$**

| | count | num -root | hot |
|---|---|---|---|
| $t_1$ | 157 | 29 | 16 |
| $t_2$ | 206 | 37 | 19 |
| $t_3$ | 298 | 59 | 32 |
| $t_4$ | 298 | 59 | 32 |

**Fog node $f_3$**

| | count | num -root | hot |
|---|---|---|---|
| $t_1$ | 150 | 28 | 12 |
| $t_2$ | 117 | 8 | 20 |
| $t_3$ | 170 | 53 | 32 |
| $t_4$ | 177 | 48 | 34 |

Fig. 3: The network diary transmission of fog nodes.

nodes. The goal of pre-processing is to filter out data that has no impact on the Skyline query.

2) The data flow is synchronized and processed on the cloud server, as a large number of fog nodes will send data at different time. When all the fog nodes send the preprocessed data to the cloud server, the asynchronous data will be generated. It is necessary to develop a mechanism for processing asynchronous data on the cloud server.

3) The cloud server analyzes the results of the data flow processing and measures the security threat levels of all fog nodes. A proper measuring method needs to be used to calculate the security state levels of fog nodes.

In practice, there are two key challenges for implementing this security state monitoring scheme: fog node data transmission and asynchronous data management.

1) **Fog node data transmission:** As mentioned above, if the cloud server performs the Skyline query, it needs to get data efficiently from the fog nodes. If the cloud server adopts the centralized Skyline query, the data of all fog nodes need to be collected and stored continuously. Due to the large amount of security log data generated by the fog nodes, the Skyline query will produce cloud-fog data transmission overload problem. This problem is further explained using the following example.

As shown in Figure 3, the cloud server executes Skyline query using the following data: COUNT, which is the number of connections in a fog node in a period of time; NUM_ROOT, which is the number of root users accessing a fog node over a period of time; and HOT, which is the number of sensitive files accessed to the system in a period of time. This scheme produces two sources of data redundancy. First, data generated by each fog node at different times may be the same which is called time induced data redundancy, such as the security log data of $f_2$ at $t_3$ and $t_4$. Second, there is a large amount of dominance relationship between data objects stored in a single fog node. The dominating data objects are not necessarily uploaded to the cloud server. We call this as dominance induced data redundancy.

The key challenge is how to effectively reduce the size of uploaded data. To tackle this challenge, we propose a data pre-processing scheme before the data is uploaded to the cloud server. The details of data pre-processing strategies and processes will be described in Section IV.

2) **Asynchronous data processing on the cloud server:** The cloud server needs to perform real-time Skyline query on the security log to determine the real-time security state of fog node. Due to heterogeneous computing power and communication resource of each fog node, there are two problems that need to be solved. First, fog nodes with different computing power may upload different amounts of data during a query cycle. Second, owing to the different arrival time of uploaded data, the cloud server will suffer from data asynchronous problem.

To address these issues, we propose a sliding-window-based mechanism to handle these problems. Details of the mechanism will be described in Section IV.

## IV. NODE STATE MONITORING SCHEME

In this section, we introduce the security state monitoring scheme. As mentioned in Section III, the proposed scheme has three steps: 1) A local pre-processing strategy is proposed for fog nodes to reduce communication overhead between the cloud server and fog nodes; 2) A sliding window mechanism is proposed to process asynchronous data from the fog nodes; 3) A unascertained measure (UM) method is proposed to quantify and analyze the security threat of the fog nodes. This scheme is deployed in the data preprocessing and analysis layer of FC-IDS.

The overall overview of the proposed is illustrated in Figure 4. First, each fog node managed by $D(f_i)$ sends the outcomes processed by $FilterD(f_i)$ to the sliding window of the cloud server. Second, the sliding window processes the obtained data and gets two kinds of outputs. One is the real-time output of CSKY$(D, T_j)$ during period $T$, which relates to the fog nodes real-time security states and will be used to calculate the security threat level. The other one is the expired Skyline data, which will be discarded. The key notations are listed in Table 1.

### *A. PRE-PROCESSING LOCAL DATA WITH FOG NODE FILTERING STRATEGY (FNFS)*

For local data pre-processing, we propose a fog node filtering strategy (FNFS) to reduce the data redundancy. The FNFS is composed by two strategies. The first strategy processes the time induced data redundancy and the second strategy processes the dominance induced data redundancy.

**Strategy 1:** Store the network security diary data collected in the previous moment on the fog node. In a period $T$, if the latter collected data is the same as the former one, only the former one needs to be recorded, otherwise, both data need to be recorded. For example, in a period of $T$, if $d_{k-1}$, $d_k$ and $d_{k+1}$ are the same ($d_k$ is the kth collected data in $T$), only $d_{k-1}$ will be recorded. And if $d_{k-1}$ and $d_k$ are different, both $d_{k-1}$ and $d_k$ will be recorded.

**Strategy 2:** Adopt the Ed-Max filtering strategy. In the period $T$, the fog node $f_i$ has collected the data set $D_{p1} = \{Dt_{1_i}, Dt_{2_i}, \cdots Dt_{s_i}\}$ after being filtered by Strategy 1. Assuming that the data dimension of each $Dt_{i_n}$ is $m$, the set

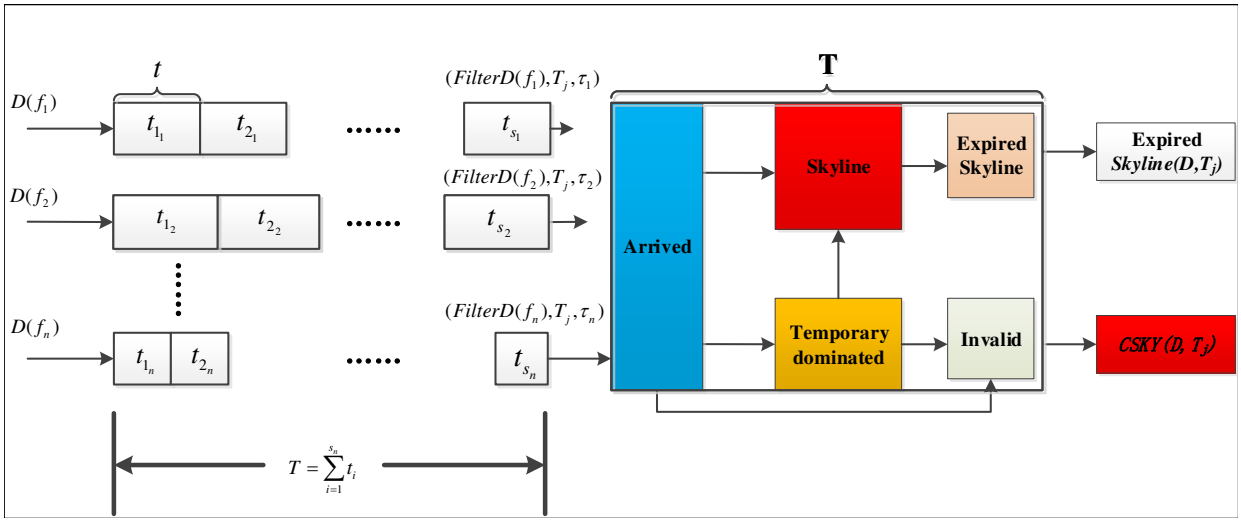



Fig. 4: The data flow of fog node security state monitoring.

TABLE I: NOTATIONS IN FOG NODE SECURITY STATE MONITORING

| Symbol | Notation |
|---|---|
| $T$ | The width of the sliding window. |
| $t_i$ | The sampling period on the fog node $f_i$ . |
| $s_i$ | The number of sampling in a $T$ period on the fog node $f_i$. |
| $D(f_i)$ | The security log data stored locally on the fog node $f_i$. |
| $FitlterD(f_i)$ | The data after preprocessing by the fog node $f_i$. |
| $\tau_i$ | The timestamp of the $FitlterD(f_i)$ sent by the fog node $f_i$ to the cloud server. |
| CSKY$(D, T_j)$ | The real-time Skyline set output through the sliding window of cloud services. |
| $(FitlterD(f_i, T_j, \tau_i)$ | The data set of fog node $f_i$ arriving at the cloud server at the $j$ th period $T$. |

$D_{\max} = \{D_{1-\max}, \cdots, D_{m-\max}\}$ represents the tuple of every $m$ dimensions. $(Dt_{s_i})_n$ is used to represent the value in the $n'$th dimension of the tuple $Dt_{s_i}$. Then, we define the Euclidean distance between the tuple and the origin as

$$Ed(Dt_{s_i}) = \sqrt{\sum_{n=1}^{m} [(Dt_{s_i})_n]^2}. \quad (1)$$

Then, $\max Ed(D_{p1})$ represents the largest value of the Euclidean distance in the data set $D_{p1}$ (except for the tuple of $D_{\max}$), and the corresponding tuple is denoted as $D_{\max Ed}$.

Assuming that the data space is an m-dimensional coordinate space, each tuple has a mapping point in that space. The Ed-Max filtering strategy uses an area encircled by $D_{\max}$, $D_{\max Ed}$, and the coordinates of the filtering data. We set the two-dimensional data as an example to illustrate the process of Ed-Max filtering strategy in Figure 5.

As illustrated in Figure 5, there are 11 tuples in the data space. Among them, $f_3$ is the tuple of the largest value in the dimension of the "Number of vulnerabilities" ($f_3 \in D_{\max}$). Similarly, we have $f_9 \in D_{\max}$. After calculating $Ed(F)$, we can get $f_8 = D_{\max Ed}$. The shadow region encircled by $f_3$, $f_9$, $f_8$ and the coordinate axis is the dominant region. Fog nodes filter the other tuples by using this strategy. If some tuple falls into the domination region, then the tuple will be filtered. The

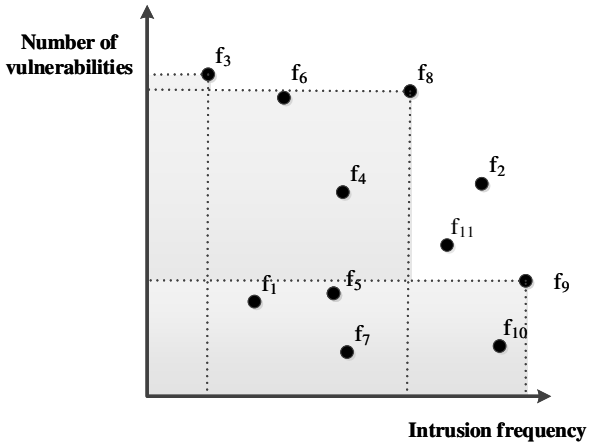



Fig. 5: Ed-Max filtering strategy

Ed-Max filtering strategy can reduce the amount of data that fog nodes transmit to the cloud server.

**Theorem 1.** *$D_{\max Ed}$ in a data set is not a dominated point. In other words, $D_{\max Ed}$ belongs to the Skyline set.*

*Proof:* If there exists a tuple $D' \succ D_{\max Ed}$, the value of $D'$ should be greater than that of $D_{\max Ed}$ in at least one datadimension, and the value should be equal or greater than that of $D_{\max Ed}$ in other data dimensions. Therefore, we have $Ed(D') > Ed(D_{\max Ed})$, and then $D_{\max Ed}$ is not the tuple that has the largest Euclidean distance in the data set. It contradicts the requirements, which concludes the proof. ■

Using these two proposed strategies, the redundant data can be significantly reduced and the pseudocode of the FNFS is given in Algorithm1 as follows.

**Algorithm 1 FNFS**

```
# DATA is an array of M dimensions, and each
DATA value is a sample value once
list = []
pre_data = zeros(m)
for i in range(S)
  if DATA != pre_data
list.append(DATA)
pre_data = DATA # list is a S sample point array
  end if
end for
max_list = []
temp = Array(m, S) # An array of m rows
and S columns
  for j in range(m)
    temp[j]. append(item[j]
  end for
for item in temp
  dimension = item.dimension
  index = max(item).position
  max_list.append((list[index], dimension))
end for
# Find the max Euclidean distance
Ed = []
for item in list
  distance = Euclidean(item) # Euclidean distance
  Ed.append(distance)
end for
index = max(Ed).position
max_list.append(list[index])
# FNFS Filter
for item in max_list
  for item2 in list
    if item2 ¡= item
      list.pop(item2)
    end if
  end for
end for
# Then the last sample value uploadedis the
sample value in list
```

### *B. PROCESSING THE DATA ON CLOUD SERVER WITH SLIDING WINDOW MECHANISM*

The asynchronous data is caused by irregular data arrival and heterogeneous data size. This requires the cloud server to be flexible enough to handle this asynchronism problem. Therefore, a sliding window mechanism is developed to manage the data flow in the cloud server.

As shown in Figure 6, the sliding window mechanism has 4 processing steps: P_1 to P_4. During the process, there are 6 states:

• **Arrived:** It represents the state of the data object sent by each fog node that arrives at the cloud server, and the object is waiting to enter the cloud server to be implemented by the Skyline calculation.

• **Skyline:** It represents the state of not being dominated by other data points at the sliding window with width $T$. It is denoted as Skyline_2 and Skyline_3 partly in P_2 and P_3. Actually, both Skyline_2 and Skyline_3 belong to the Skyline set that is computed in real time, and we record them as CSKY set. Since our aim is to monitor highly vulnerable fog nodes, we pay more attention to the data from fog nodes in CSKY. According to CSKY, we can calculate the security threat level of fog nodes.

• **Temporary Dominated:** It represents the state of a data object being temporarily dominated by other data objects in P_2.

• **S-Dominated:** It represents the state change of a data object from no domination to being dominated in P_3.

• **Invalid:** It represents the state of a data object always being in domination throughout the whole process. When the Invalid is expired, the object will be stored in the database of the cloud server as Strong Dominated.

• **Expired Skyline:** It represents the Skyline point when the data object is removed from the sliding window. Here we focus on when the fog node's security is threatened. We will mark the data in the cloud server as Expired Skyline when it has expired. In other words, the Expired Skyline is the stored set of historic security data.

It is significantly meaningful to the historical query of the fog node security state and can be applied in obtaining evidence and other aspects. The four processes are introduced as follows.

• Process 1(P_1): Arrived: this process represents that the data object will enter the cache's waiting state.

• Process 2(P_2): Preliminary calculation: this process, the latest data object will be first processed the Skyline queries. If the data object $P$ is dominated by the original Skyline point, the data object's state "Temporary Dominated". If the $P$ is not dominated by the original point, the data object's state will be marked as "Skyline", and the Skyline set in sliding window will be updated.

• Process 3(P_3): subsequent calculation: It describes the changing operating state when the data object in the sliding window becomes earlier object. As shown in Figure 6, during P_2, the state of the data object $P$ may become either Skyline or Temporary Dominated. When the later object $Q$ arrives at the sliding window, the state of $P$ may change. During P_2, if the data object of the Skyline dominates the later object $Q$, then during P_3, $P$ is still the element of the Skyline set and the state of $Q$ is recorded as Temporary Dominated. During P_2, if data object $P$ is dominated by the later object $Q$, then $P$ will change from the original Skyline to S-Dominated and the state of $Q$ will be recorded as Skyline. During P_2, if the original data object dominating $P$ of the Temporary Dominated is expired and $P$ is not dominated by other objects, then the state will change from the original Temporary Dominated into the Skyline. During P_2, if the original data object dominating $P$ of Temporary Dominated is expired and it is still dominated by the later object $Q$ or the object with the same age, then the state of $P$ will change from Temporary Dominated into Invalid.

• Process 4(P_4): Leave: this process represents the data object in the storage state by the cloud server after leaving the

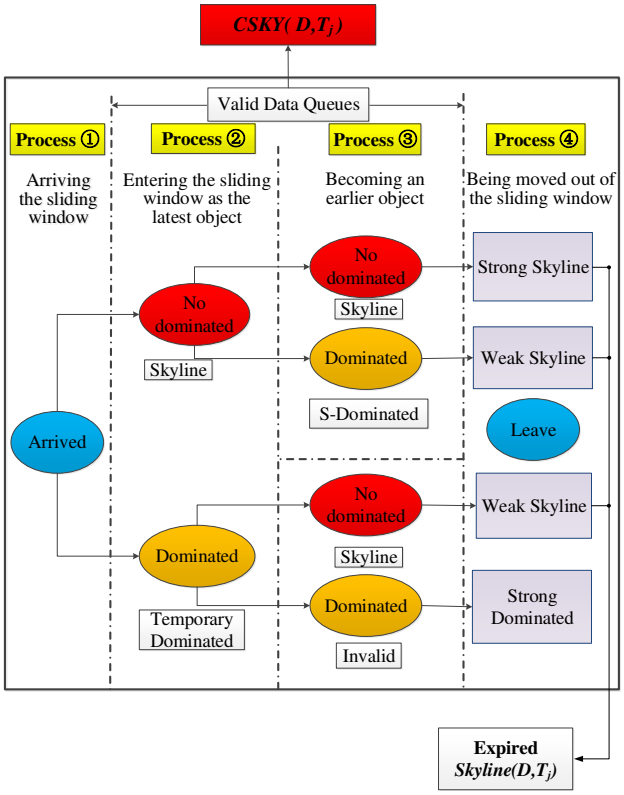



Fig. 6: Sliding window mechanism in cloud.

sliding window. As shown in Figure 6, if the data object $P$ leaves the state of Skyline, the data object will be stored as Strong Skyline. If data object 'P' maintains the Skyline state in the sliding window, the data object will be stored as Weak Skyline when leaving. If the state of $P$ is not Skyline from P_1 to P_4, it should be stored as Strong Dominated when leaving.

As mentioned above, P_2 and P_3 are the procedures for when data operates in a sliding window. Throughout the whole process, the cloud server outputs two parts of data. One part is the CSKY that is real-time calculated by the cloud server storage. The other part is the Expired Skyline transmitted to and stored in the disk array of the cloud server. Our aim is to real-time monitor the fog node groups of greater vulnerability among the cluster. Therefore, we pay more attention to the data output of the CSKY. The CSKY output is the set of data points of each fog node in the sliding window. Next, we will introduce how to calculate the security threat level of fog nodes using the CSKY output.

### *C. CALCULATING FOG NODE SECURITY THREAT LEVEL WITH UM*

To evaluate the security threat of each fog node, the cloud server calculates the index weight of each fog note in the set based on the unascertained measure (UM) method. The UM is an objective assignment method index [22] that can efficiently calculate security threat level calculation.

During the UM adoption process, a measure function will be built. However, for the security threat calculation problem of fog nodes, there is no existing standard to construct the function. In this paper, we leverage the outputted statistical results of CSKY to construct this function Table 2.

In the output of CSKY, we care only about the fog nodes to which data objects belong. Assume the number of fog nodes being $n$. The sampling interval corresponding to the fog node is $S = \{s_1, s_2, \cdots, s_n\}$ which is a parameter that can be self-adjusted by the fog node. It depends on the computational performance of the fog node. Among the outcomes of CSKY at one period $T$, the number of data that belongs to fog node is $Q$=$\{q_1,q_2,\cdots,q_n\}$.

Since the computing power of each fog node is different, each fog note has a different sampling interval. Considering the effects of the sampling interval on the security threat level, we define the fog node Skyline report rate $Y$ as

$$Y = \{y_1, y_2, \cdots, y_n\}, \tag{2}$$

where

$$y_i = \left(\frac{q_i}{s_i} - \min\frac{q_i}{s_i}\right) / \left(\max\frac{q_i}{s_i} - \min\frac{q_i}{s_i}\right).$$

First, we build a linear measuring function based on [14], which provides the expression of the measurement function $U(x)$. Specifically, the expressions corresponding to the uncertain measurement functions $\mu$ are

$$\begin{cases} \mu_i(x) = \begin{cases} 1 - \dfrac{1-e^{x-a_i}}{1-e^{a_{i+1}-a_i}} & a_i < x \le a_{i+1} \\ 0 & x > a_{i+1} \end{cases} \\ \mu_{i+1}(x) = \begin{cases} 0 & x < a_i \\ \dfrac{1-e^{x-a_i}}{1-e^{a_{i+1}-a_i}} & a_i < x \le a_{i+1}. \end{cases} \end{cases} \tag{3}$$

According to the measure in Table 2, we format the measurement function $U(x)$ as:

$$\underset{C0\sim C1}{\mathrm{U(x)}} \begin{cases} \mu_{C0}(x) = \begin{cases} 1 - \dfrac{1-e^{x}}{1-e^{0.1}} & 0 < x \le 0.1 \\ 0 & x > 0.1 \end{cases} \\ \mu_{C1}(x) = \begin{cases} 0 & x = 0 \\ \dfrac{1-e^{x}}{1-e^{0.1}} & 0 < x \le 0.1, \end{cases} \end{cases} \tag{4}$$

$$\underset{C1\sim C2}{\mathrm{U(x)}} \begin{cases} \mu_{C1}(x) = \begin{cases} 1 - \dfrac{1-e^{x-0.1}}{1-e^{0.2}} & 0.1 < x \le 0.3 \\ 0 & x > 0.3 \end{cases} \\ \mu_{C2}(x) = \begin{cases} 0 & x < 0.1 \\ \dfrac{1-e^{x-0.1}}{1-e^{0.2}} & 0.1 < x \le 0.3, \end{cases} \end{cases} \tag{5}$$

$$\underset{C2\sim C3}{\mathrm{U(x)}} \begin{cases} \mu_{C2}(x) = \begin{cases} 1 - \dfrac{1-e^{x-0.3}}{1-e^{0.3}} & 0.3 < x \le 0.6 \\ 0 & x > 0.6 \end{cases} \\ \mu_{C3}(x) = \begin{cases} 0 & x < 0.3 \\ \dfrac{1-e^{x-0.3}}{1-e^{0.3}} & 0.3 < x \le 0.6, \end{cases} \end{cases} \tag{6}$$

TABLE II: NOTATIONS IN FOG NODE SECURITY STATE MONITORING

| | **C0** | **C1** | **C2** | **C3** | **C4** |
|---|---|---|---|---|---|
| $F$ | 0 | 0.1 | 0.3 | 0.6 | 0.85 |

$$\underset{C3\sim C4}{\mathrm{U(x)}} \begin{cases} \mu_{C3}(x) = \begin{cases} 1 - \dfrac{1-e^{x-0.6}}{1-e^{0.25}} & 0.6 < x \leq 0.85 \\ 0 & x > 0.85 \end{cases} \\ \mu_{C3}(x) = \begin{cases} 0 & x < 0.6 \\ \dfrac{1-e^{x-0.6}}{1-e^{0.25}} & 0.6 < x \leq 0.85. \end{cases} \end{cases} \tag{7}$$

According to measurement functions (4), (5), (6) and (7), we let $Y = x$ and calculate $U(Y)$. In this way, we can get the measurement recognition matrix of fog node $F = \{f_1, f_2, \cdots, f_n\}$ as:

$$\begin{array}{c} \\ f_1 \\ f_2 \\ \vdots \\ f_n \end{array} \begin{array}{c} \begin{array}{cccc} C1 & C2 & C3 & C4 \end{array} \\ \begin{bmatrix} \mu_{11} & \mu_{12} & \mu_{13} & \mu_{14} \\ \mu_{21} & \mu_{22} & \mu_{23} & \mu_{24} \\ \vdots & \vdots & \vdots & \vdots \\ \mu_{n1} & \mu_{n2} & \mu_{n3} & \mu_{n4} \end{bmatrix} \end{array}.$$

According to UM method, we calculate the intermediate value of the security threat level of fog nodes. Let

$$v_i = 1 + \frac{1}{2} \sum_{l=1}^{4} \mu_{il} \log \mu_{il}. \tag{8}$$

The security threat level of each fog node can be calculated as:

$$w_i = v_i / \sum_{k=1}^{n} v_k \ . \tag{9}$$

Accordingly, the security threat vector of the fog cluster is $w = (w_1, w_2, \cdots, w_n)^T$, $\sum_{i=1}^{n} w_i = 1$.

Note that through the normalization treatment of (9), the security threat of the fog cluster we calculate is a relative value. That is to say, the $w_i$ of fog node $f_i$ is affected by other fog nodes.

## V. NUMERICAL SIMULATION

The simulation environment is built based on [16]. We adopt the KDD CUP99 dataset [34] to measure and classify the results according to the attack types. There are four types of intrusion data: DoS, R2L, U2L, and PROBE. The counting diary records the counting results of attack types in unit time.

First, we validate the effects of reducing communication costs of fog nodes using FNFS. The counting diary data are time-independent random distribution. We extract "abnormal" attacks from the KDD CUP99 data as the measurement set to implement the measurements. Among the set, there are 4000 pieces of PROBE, 10000 pieces of DoS, 200 pieces of U2R, and 5000 pieces of R2L. We simulate the counting diary of the fog node using periodic sampling. The fog nodes realize the data flow transmission by Storm [35]. The data flow speed of Spout in Storm is 100 numbers/s, and the frequency is 10 times per second on the fog node. In other words, there are 10 data counts every second. We set up a period $T$=60s, and when the data dimension is 2, we compare the effects of the Ed-Max filtering algorithm and MinMax filtering algorithm [36] that we constructed on the data size after filtering.

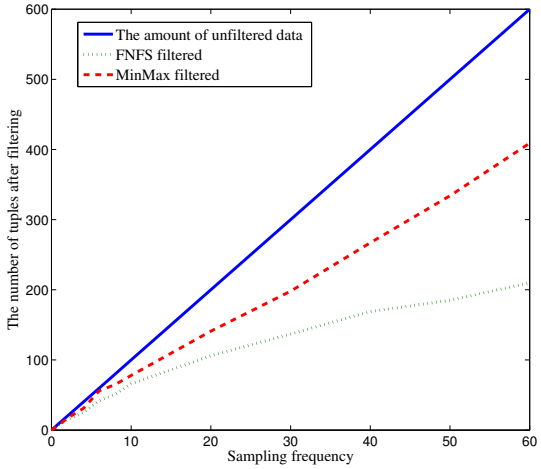



Fig. 7: The filtering effect of FNFS in different sampling frequencies.

As shown in Figure 7, the proposed FNFS filtering strategy has significantly reduced the original data size when the data flow dimension is 2. The experiment compares MinMax and FNFs, and the results indicate a better performance of FNFS. There are two main reasons, and the primary one is that the Ed-Max algorithm in the FNFS has greater filtering area to the data. MinMax is the improved version of the threshold method and it looks for only one data point in the two-dimensional space. Meanwhile, the Ed-Max algorithm that we raised has $(m+1)$ data points in the data space as the datum points of data filtering measurement. Another reason is that the FNFS in strategy 1 has the effect of removing weight. Especially when the fog node sampling period is smaller and the change in the network environment is not large, the fog node may produce more repeated data. The repeated data can be filtered to a certain extent using strategy 1. The experiment verified that when the data sampling period was bigger, the extent of data size reduction would be bigger.

We also designed a group of experiments to assess the filtering effects of FNFS under different data dimensions. Define the fog node data filtering rate as the ratio of the number of filtered tuples and the number of overall tuples.

$$\eta = \frac{q_i - (q_f)_i}{q_i} \cdot 100\%, \tag{10}$$

where $(q_f)_i$ is the number of filtered tuples that belongs fog node $f_i$.

Under the situation of fog node sampling frequency being fixed, we analyze the data filtering rate when the data dimension is 2, 3 and 4 in Figure 8. The increased data dimension will reduce the data filtering effects. The reason is that when some dimension has been introduced, new points without dominations at this dimension will be created, and thus the data points will not fall into the region filtered by

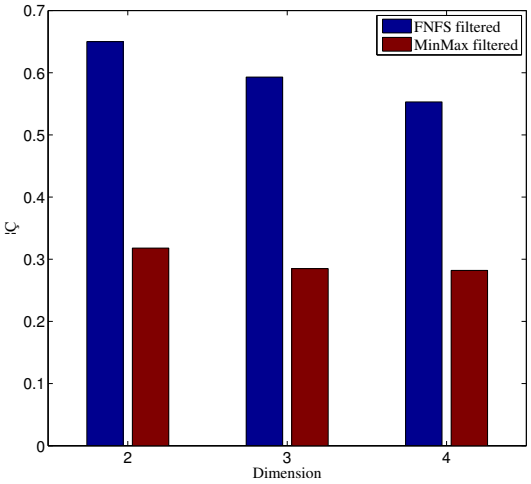



Fig. 8: The filtering effect of FNFS in different dimensions.

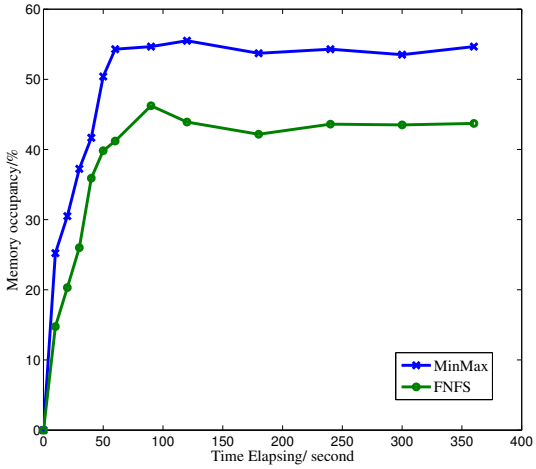



Fig. 9: Analysis of FNFS spatial complexity.

Ed-Max. However, the difference between FNFS and MinMax will reduce with the increase of the dimensions. From two-dimension to four-dimension, the data filtering rate of FNFS decreases by approximately $14.9\%$ and that of MinMax decreases by approximately $11.3\%$. Although the range of the MinMax reduction is smaller with the increase of dimensions, we can still conclude through the outcomes that the FNFS has significantly better filtering effects than MinMax. Therefore, from the point of data filtering effects, the FNFS is much more appropriate to the fog computing environment.

Fog nodes have limited resources. Therefore, the computational complexity of the FNFS needs to be considered. We measure the space complexity mainly through the memory usage of the fog nodes. Figure 9 shows the results of the experiment. It can be observed that the memory usage rate of the FNFS filtering scheme is low. Therefore, the FNFS is a lightweight filtering scheme that is suitable for deployment on fog nodes.

Finally, we show the monitoring results of fog node security status using the proposed scheme. Figure 10 shows the security threat level of the CSKY output of five fog nodes. From the monitoring results, we can see that the security threat of FN3 is relatively low. The security status of other nodes has varying degrees of fluctuations.

From these experiments, we can see that the monitoring scheme of fog node's security state is effective for F-RANs. The FNFS has excellent data filtering performance. The sliding window scheme on the cloud server also meets the real-time requirement.

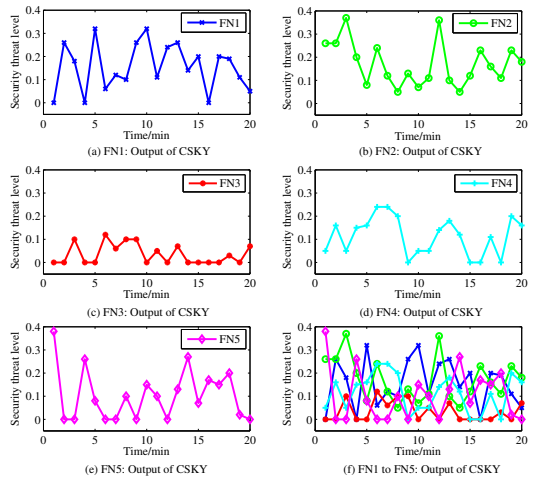



Fig. 10: The security threat level monitoring results of 5 fog nodes.

## VI. CONCLUSIONS

Fog nodes, as the key components of F-RANs, undertake the task of providing services for terminal devices [37]. Since it is closer to the user, the security threat faced by fog nodes is more serious. The security state monitoring of fog nodes is particularly important in the IDS of fog computing. This paper first analyzes the problems of the security state monitoring of fog nodes. Then, we propose an integrated monitoring scheme for the security of fog nodes where the real-time security threat level of fog node clusters can be calculated instantaneously on the cloud server. Experimental results show that the FNFS can effectively reduce the communication overhead between the cloud server and fog nodes.

## REFERENCES

[1] M. Peng, Y. Li, Z. Zhao, and C. Wang, "System architecture and key technologies for 5G heterogeneous cloud radio access networks," *IEEE Network*, vol. ED-29, no. 2, pp. 6-14, Mar. 2015.

[2] Z. Zhao, M. Peng, Z. Ding, W. Wang, and H. V. Poor, "Cluster content caching: An energy-efficient approach to improve quality of service in cloud radio access networks," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 5, pp. 1207-1221, May. 2016.

[3] M. Chiang and T. Zhang, "Fog and IoT: An Overview of Research Opportunities," *IEEE Internet Things J.*, vol. 3, no. 6, pp. 854-864, Dec. 2016.

[4] 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, Aug. 2016.

[5] M. Peng and K. Zhang, "Recent advances in fog radio access networks: Performance analysis and radio resource allocation," *IEEE Access*, vol. 4, pp. 5003-5009, Aug. 2016.

[6] J. Su, F. Lin, X. Zhou and X. Lü, "Steiner tree based optimal resource caching scheme in fog computing," *in China Commun.*, vol. 12, no. 8, pp. 161-168, August 2015.
[7] O. Osanaiye, S. Chen, Z. Yan, R. Lu, K. R. Choo and M. Dlodlo, "From Cloud to Fog Computing: A Review and a Conceptual Live VM Migration Framework," *IEEE Access*, vol. 5, pp. 8284-8300, 2017.
[8] Y. Huo, Y. Tian, L. Ma, X. Cheng, and T. Jing, "Jamming Strategies for Physical Layer Security," *IEEE Wirel. Commun.*, vol.25, no.1, pp. 148-153, 2018.
[9] Y. Yang, H. Cai, Z. Wei, H. Lu, K. K. R. Choo, "Towards Lightweight Anonymous Entity Authentication for IoT Applications," in Proceedings of 21st Australasian Conference on Information Security and Privacy, Cham, Switzerland, 2016, pp. 265-280.
[10] R. Rodrigo, J. Lopez, and M. Mambo, "Mobile edge computing, Fog et. al.: A survey and analysis of security threats and challenges," *Futur. Gener. Comp. Syst.*, vol.78, pp.680-698, 2018.
[11] S. Barbarossa, E. Ceci and M. Merluzzi, "Overbooking radio and computation resources in mmW-mobile edge computing to reduce vulnerability to channel intermittency," 2017 European Conference on Networks and Communications (EuCNC), Oulu, 2017, pp. 1-5.
[12] F. Lin, X. Lü, I. You and X. Zhou, "A Novel Utility Based Resource Management Scheme in Vehicular Social Edge Computing," IEEE Access, vol. 6, pp. 66673-66684, 2018.
[13] X. An, F. Lin, S. Xu, L. Miao and C. Gong, "A novel differential game based intrusion response," *Secur. Commun. Netw*, vol.2018, Article ID 1821804, 9 pages, 2018. 8. https://doi.org/10.1155/2018/1821804.
[14] F. Lin, Y. Zhou, G. Pau and M. Collotta, "Optimization-Oriented Resource Allocation Management for Vehicular Fog Computing," IEEE Access, vol. 6, pp. 69294-69303, 2018.
[15] D. Denning, "An Intrusion-Detection Model," *IEEE Trans. Softw. Eng.*, vol. SE-13, no. 2, 1987, pp. 222-232.
[16] X. An, X. Zhou, X. Lü, F. Lin, and L. Yang, "Sample Selected Extreme Learning Machine Based Intrusion Detection in Fog Computing and MEC," *Wirel. Commun. Mob. Comput.*, vol.2018, pp.1-10, 2018.
[17] X. An, J. Su, X. Lü and F. Lin, "Hypergraph Clustering Model-based Association Analysis of DDOS attacks in Fog Computing Intrusion Detection System," *EURASIP J. Wirel. Commun. Netw.*, 2018 (Accpeted).
[18] Y. Huo, C. Yong, and Y. Lu, "Re-ADP: Real-time Data Aggregation with Adaptive w-event Differential Privacy for Fog Computing," Wirel. Commun. Mob. Comput., vol. 2018, pp. 1-13, 2018.
[19] S. Borzsony, D. Kossmann and K. Stocker, "The Skyline operator," in Proceedings 17th International Conference on Data Engineering, Heidelberg, 2001, pp. 421-430.
[20] O. Papapetrou and M. Garofalakis, "Continuous fragmented skylines over distributed streams," in 2014 IEEE 30th International Conference on Data Engineering, Chicago, IL, 2014, pp. 124-135.
[21] C. Yu, Y. Tsou, C. S. Lu and S. Y. Kuo, "Practical and Secure Multidimensional Query Framework in Tiered Sensor Networks," IEEE Trans. Inf. Forensic Secur., vol. 6, no. 2, pp. 241-255, June 2011.
[22] S. Yang, "IoT Stream Processing and Analytics in the Fog," *IEEE Commun. Mag.*, vol. 55, no. 8, pp. 21-27, 2017.
[23] F. Lin, Z. Pang, X. Ma, Q. Gu, "User Access Management Based on Network Pricing for Social Network Applications," *Sensors* vol. 2, no.18, pp. 664, 2018.
[24] Q. Yaseen, F. AlBalas, Y. Jararweh and M. Al-Ayyoub, "A Fog Computing Based System for Selective Forwarding Detection in Mobile Wireless Sensor Networks," in 2016 IEEE 1st International Workshops on Foundations and Applications of Self Systems (FASW), Augsburg, 2016, pp. 256-262.
[25] G. Zhang and R. Li, "Fog computing architecture-based data acquisition for WSN applications," *China Commun*, vol. 14, no. 11, pp. 69-81, Nov. 2017.
[26] Y. Shi, S. Abhilash and K. Hwang, "Cloudlet Mesh for Securing Mobile Clouds from Intrusions and Network Attacks," 2015 3rd IEEE International Conference on Mobile Cloud Computing, Services, and Engineering, San Francisco, CA, 2015, pp. 109-118.
[27] Y. Wang, L. Xie, W. Li, W. Meng, J. Li, "A Privacy-Preserving Framework for Collaborative Intrusion Detection Networks Through Fog Computing," in International Symposium on Cyberspace Safety and Security, Cham, 2017, pp. 267-279.
[28] F. Hosseinpour, P. V. Amoli, J. Plosila, T. Hämäläinen, H. Tenhunen "An Intrusion Detection System for Fog Computing and IoT based Logistic Systems using a Smart Data Approach" International Journal of Digital Content Technology & Its Applications, vol.5, no. 10, 2016.
[29] F. Lin, Y. Zhou, X. An, I. You, K. Choo, "Fair Resource Allocation in Intrusion Detection System for Edge Computing," *IEEE Consum. Electron. Mag.*, vol.7, no. 6, pp.45-50.
[30] W. Tang, K. Zhang, J. Ren, Y. Zhang and X. Shen, "Lightweight and Privacy-Preserving Fog-Assisted Information Sharing Scheme for Health Big Data," in GLOBECOM 2017 -2017 IEEE Global Communications Conference, Singapore, 2017, pp. 1-6.
[31] J. Ni, K. Zhang, X. Lin and X. S. Shen, "Securing Fog Computing for Internet of Things Applications: Challenges and Solutions," *in IEEE Commun. Surv. Tutor.*, vol. 20, no. 1, pp. 601-628, First quarter 2018.
[32] I. Angelopoulos, E. Trouva and G. Xilouris, "A monitoring framework for 5G service deployments," in 2017 IEEE 22nd International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), Lund, 2017, pp. 1-6.
[33] Y. Wang, H. Pang, Y. Yang, X. Ding, "Secure server-aided top-k monitoring," Inf. Sci., vol. 420, 2017, pp. 345-363.
[34] KDD CUP 99 data set Available: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html
[35] W. Yang, X. Liu, L. Zhang and L. Yang, "Big Data Real-Time Processing Based on Storm," in 2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, Melbourne, VIC, 2013, pp. 1784-1787.
[36] H. Chen, S. Zhou, and J. Guan. "Towards Energy-Efficient Skyline Monitoring in Wireless Sensor Networks," in European Conference on Wireless Sensor Networks Springer-Verlag, 2007, pp. 101-116.
[37] Y. Huo, C. Hu, X. Qi, and T. Jing, "LoDPD: A Location Difference-Based Proximity Detection Protocol for Fog Computing," *IEEE Internet of Things J.*, vol. 4, no. 5, pp. 1117-1124, Oct. 2017.

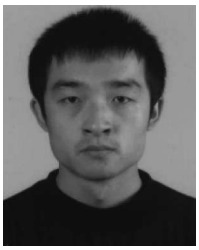
**Xingshuo An** was born in Shandong province, China in 1988. He received his Master degree from University of Science and Technology Beijing, in 2014. He is currently a Ph. D student in School of Computer and Communication Engineering, University of Science and Technology Beijing, P. R. China. His research direction is fog computing and network security.

**Xing LÜ** received his Ph. D degree from Beijing University of Posts and Telecommunications, Beijing, P. R. China, in 2012, in computer science and technology. Now he is a professor in department of Computer and Communication Engineering, University of Science and Technology Beijing, P. R. China. His research interests include soliton theory, symbolic computation and optical soliton communication.

**Lei Yang** (M13) received the B.S. and M.S. degrees in electrical engineering from Southeast University, Nanjing, P. R. China, in 2005 and 2008, respectively, and the Ph.D. degree from the School of Electrical, Computer, and Energy Engineering (ECEE), Arizona State University, Tempe, AZ, USA, in 2012.He was a Postdoctoral Scholar with Princeton University, Princeton, NJ, USA. He has worked as an Assistant Research Professor with the School of ECEE, Arizona State University, since 2013. His research interests include stochastic optimization and big data analytics for renewable energy integration, grid integration of plug-in electric vehicle, networked control of cyber-physical systems, modeling and control of power systems, network security and privacy, and network optimization and control.

Dr. Yang received the Best Paper Award Runner-up of IEEE INFOCOM 2014

**Xianwei Zhou** received his M.S. degree from Zhengzhou University in 1992. He obtained Ph.D. degree in Department of Transportation Engineering from Southwest Jiaotong University, P. R. China in 1999. He was engaged in postdoctor study at Beijing Jiaotong University, China, from 1999 to 2000. Now, he is a professor in School of Computer and Communication Engineering, University of Science and Technology Beijing. His research interests include the security of communication networks, cloud computing and game theory.



**Fuhong Lin** received his M.S. degree and Ph.D. degree from Beijing Jiaotong University, Beijing, P. R. China, in 2006 and 2010, respectively, both in Electronics Engineering. Now he is an associate professor in department of Computer and Communication Engineering, University of Science and Technology Beijing, P. R. China. His research interests include Edge/Fog Computing, Network Security, and Big Data. He won "Provincial and Ministry Science and Technology Progress Award 2" in 2017. His two papers won "Top 100 most Cited Chinese Papers Published in International Journals" in 2015 and 2016. He served as co-chair of the first and third IET International Conference on Cyberspace Technology, and general chair of the second IET International Conference on Cyberspace Technology. He was the leading editor of the Special issue "Recent Advances in Cloud-Aware Mobile Fog Computing" for Wireless Communications and Mobile Computing. Currently, he also serves as a reviewer more than 10 international journals including IEEE Transactions on Industrial Informatics, IEEE Access, Information Sciences, IEEE Internet of Things Journal, The Computer Journal and China Communications. He received the track Best Paper Award from IEEE/ACM ICCAD 2017.

*The corresponding author, email: FHLin@ustb.edu.cn