Lin, Huaqing; Yan, Zheng; Fu, Yulong

Adaptive security-related data collection with context awareness

Published in:
Journal of Network and Computer Applications

DOI:
10.1016/j.jnca.2018.11.002

Published: 15/01/2019

Document Version
Publisher’s PDF, also known as Version of record

Please cite the original version:
Adaptive security-related data collection with context awareness

Huaqing Lin\textsuperscript{a,b}, Zheng Yan\textsuperscript{a,b,c,*}, Yulong Fu\textsuperscript{b}

\textsuperscript{a} State Key Laboratory on Integrated Services Networks, School of Cyber Engineering, Xidian University, Xi'an, China
\textsuperscript{b} School of Cyber Engineering, Xidian University, Xi'an, China
\textsuperscript{c} Department of Communications and Networking, Aalto University, Espoo, Finland

\textbf{ARTICLE INFO}

Keywords:
Security-related data
Adaptive data collection
Heterogeneous network
Network context

\textbf{ABSTRACT}

The huge economic loss resulting from network attacks and intrusions has led to an intensive study on network security. The network security is usually reflected by some relevant data that can be collected in a network system. By learning and analyzing such data, which are called security-related data, we can detect the intrusions to the network system and further measure its security level. Clearly, the first step of detecting network intrusions is to collect security-related data. However, in the context of 5G and big data, there are a number of challenges in collecting these data due to the heterogeneity of network and ever-growing amount of data. Therefore, traditional data collection methods cannot be applied in the next generation network systems directly, especially for security-related data. This paper presents the design and implementation of an adaptive security-related data collector based on network context in heterogeneous networks. The proposed collector solves the issue of heterogeneity of network system by designing a Security-related Data Description Language (SDDL) to instruct security related data collection in various networking contexts. It also applies adaptive sampling algorithms to reduce the amount of collected data. Furthermore, performance evaluation based on a prototype implementation shows the effectiveness of the adaptive security-related data collector in terms of a number of pre-defined design requirements.

1. Introduction

With the development of the next generation mobile networks and wireless systems (in short, 5G), Internet of Things and social networking (Peng et al., 2016; Jiang et al., 2016; Yang et al., 2017; Zhang et al., 2017), more and more attention has been paid to the security of a heterogeneous network system organized by different types of networks, such as the Internet, mobile cellular networks, self-organized Mobile Ad hoc Networks (MANET), Wireless Sensor Networks (WSN), etc. The network security is usually reflected by relevant data that can be collected in the network system. By studying the data related to network security events, the security of the network system can be quantified and measured. Thus, the first step in detecting network attacks and intrusions is data collection, especially collecting the data related to the occurrence of security events. In this paper, we define such data as \textit{security-related data} that indicates security threats and shows abnormality with regard to security, safety, privacy and trust. However, how to determine the security-related data, how to collect these data and how to use these data to measure network security in a heterogeneous network system are still open issues. In this paper, we focus on the collection of security-related data in heterogeneous networks.

Security-related data collection in heterogeneous networks is different from traditional networks due to its heterogeneity. In such networks, the amount of network data is huge and the data flow velocity is fast. The most important issue is heterogeneity of network types and heterogeneity of network nodes due to complex system architectures and network topologies. A heterogeneous network is composed of multiple different types of access network systems and can dynamically switch among them based on the demands of network users. It also contains a number of different types of node devices. All of these bring a lot of challenges to security-related data collection. Furthermore, in the context of big data, security-related data has 5 V (i.e., Volume, Variety, Value, Velocity and Veracity) characteristics, which pose tremendous difficulties in their collection.

Currently, there are few security-related data collection solutions for heterogeneous networks. A possible reason is that heterogeneous networks have not yet been popularized. Some work on data collection in heterogeneous network is to collect link information of access networks to achieve mobility management (Monger et al., 2009; Fouquet et al., 2009, 2011; Habeiba et al., 2011). Some other heterogeneous network data collection methods aimed at WSN, usually by designing or improving some transmission protocols to solve the issue of resource...
limitation in sensor nodes (Rohankar et al., 2015). However, the issues solved by these methods are different from our design goals as described in Section 3.2. A variety of solutions have been proposed for collecting security-related data in traditional network systems. Based on the used collection tools, data collection technologies can be divided into software-based data collection and hardware-based data collection (Lin et al., 2018). The software-based data collection has high flexibility and low cost. However, it cannot be applied into such scenarios that requires high collection efficiency because of its low performance. The hardware-based data collection is suitable for such application scenarios that require high performance, but it has high cost and is not flexible enough. Most of practically deployed security-related data collection solutions are based on hardware devices, but they are expensive and inflexible. Due to the complexity of heterogeneous networks, the flexibility of data collection solutions is increasingly demanding. Furthermore, because of the development and popularity of mobile devices, more efforts are needed to explore applicable schemes with regard to adaptive security-related data collection that can be implemented in any networking devices, including mobile terminals. Therefore, a low-cost, flexible, efficient and universal software solution for security-related data collection is urgently expected. In addition, in order to increase the collection efficiency and reduce the burden on network systems, some sampling-based methods are used to collect data (Paxson et al., 1998; Zhao et al., 2007). However, the traditional statistical sampling algorithms need to set the collection frequency in advance, which does not reflect the statistical characteristics of the data well. A novel data collector is highly needed in order to solve the above open issues and challenges.

In this paper, we propose an adaptive security-related data collector with context awareness for heterogeneous networks and present its design and implementation. We design a Security-Related Data Description Language (SDDL) based on XML. It specifies what kind of security-related data should be collected in which way and at what place based on the detection of networking context. SDDL also marks the tags about data processing methods and the target attacks that the data can be used to detect. By integrating the SDDL with network context detection, the proposed adaptive security-related data collector can collect data flexibly at any network nodes with context awareness in a large-scale heterogeneous network. By introducing adaptive sampling algorithms, data collection efficiency of the collector can be improved and the volume of the collected data can be reduced with the insurance of data collection accuracy. Furthermore, some security protection mechanisms are used to ensure the security of the collected data. After data collection, we can perform data composition and process for network security analytics based on the tags marked on the data. According to the tags attached to each piece of data collected, we know how to compose the data, how to select algorithm to process the data and what kind of attacks or security threats could be detected. With the detection result, the vulnerability and security measurement of the whole network system can be figure out.

To the best of our knowledge, our proposed collector is one of the first to adaptively and effectively collect security-related data in heterogeneous networks. Specifically, the contributions of this paper can be summarized as follows:

1) We specify key issues and design requirements of security-related data collection in a large-scale heterogeneous network system;
2) We propose the SDDL that describes the security-related data so as to facilitate data collection and processing and to solve the challenges caused by the heterogeneity of network types and network nodes;
3) We propose two adaptive sampling algorithms to collect security-related data to improve collection efficiency, ensure collection accuracy and reduce the amount of collected data in order to minimize the effect of data collection on the normal operations of a network system.
4) We implement a prototype collector in an Android mobile terminal and evaluate its performance. The results show the advantages of the adaptive security-related data collector regarding efficiency and adaptability.

The rest of the paper is organized as follows. Section 2 gives a brief review of related work on security-related data and an introductory technical overview of data collection. We introduce key issues, design requirements and our research methodology for implementing security-related data collector in Section 3. Then, we provide detailed design and implementation of our proposed adaptive security-related data collector in Section 4. In Section 5, we evaluate the performance and security of the collector based on the prototype implementation. Finally, conclusions and future work are presented in the last section.

2. Background and related work

2.1. Security-related data

Security-related data are mainly involved in the work about attack and intrusion detection (Buczak and Guven, 2016; Bhuyan et al., 2014; Nguyen and Armitage, 2008; Garcia-Teodoro et al., 2009; Sperotto et al., 2010). For example, Buczak and Guven (2016) reviewed the related work using machine learning and data mining methods to process and analyze security-related data for cyber security intrusion detection. However, prior to our work, there was no specific work to introduce security-related data and its collection methods. In order to comprehensively introduce the security-related data, we worked out four review papers (He et al., 2018; Liu et al., 2018; Xie et al., accepted; Jing et al., 2018) that introduce the security-related data in four main types of networks, including the Internet, LTE, MANET and WSN. Refer to (He et al., 2018; Liu et al., 2018; Xie et al., accepted; Jing et al., 2018) for more detailed information about security-related data. However, these four papers just tell us what security-related data should be collected for detecting which network attack or intrusion without regard to data collection technologies. Therefore, we introduce the relevant knowledge of data collection technologies in Section 2.3.

2.2. Heterogeneous networks

In this part, we introduce the main concepts of heterogeneous networks and outlines the key issues on security-related data collection in heterogeneous networks. 5G is coming and will bring great convenience to people. In particular, one of the most promising technologies in 5G is heterogeneous network architecture. It is considered to be among the most valuable enhancements to the network (Rong et al., 2016). A heterogeneous network is composed of multiple network systems with different operators, different business services and different access technologies, as well as different network topologies. It takes full advantage of the complementary characteristics of different network tiers, and thus becomes an inevitable trend for future development of information networks. An example heterogeneous network architecture is shown in Fig. 1. Multiple wireless access network systems with different characteristics are connected to the core network through dedicated gateways in order to meet abundant service requirements of subscribers.

Though deploying heterogeneous networks brings many benefits, it causes new issues that we have to handle, i.e., mobile management, access point selection, identity authentication, seamless handover (Monger et al., 2009). However, in this paper we focus on security-related data collection in heterogeneous networks.

A large-scale heterogeneous network system has the following specific characteristics:

a) Network topologies and system architectures are complex and different from traditional networks;

b) It contains a variety of heterogeneous network node devices;
related data collection in heterogeneous networks for network security.

Data can be collected in an economic and accurate way.Adjust data collection frequency according to the changes of collected data well. Therefore, we propose two algorithms to dynamically advance, which does not dynamically re-

Traditional statistical sampling algorithms set collection frequency in so forth (Lin et al., 2018; Zhou et al., 2018). Statistical sampling algo-

The characteristics of large-scale heterogeneous network systems have brought a number of challenges into the collection of security-

c) It supports dynamic handover among different types of networks in an adaptive way;
d) Network attacks are more complicated than a single network; e) Data are shared in heterogeneous network systems with a high speed; f) Security-related data are gigantic and have 5 V characteristics.

There are four steps to design a complete data collection solution in a traditional network system (Lin et al., 2018). The first step is to deter-

Existing studies lack investigation on a lightweight, flexible and efficient solution to achieve context-aware adaptive security-related data collection in heterogeneous networks with sound scalability and practi-

3. Problem statement

In this section, we summarize the key issues of security-related data collection in the heterogeneous networks. Based on the key issues, we set up our design requirements of collector implementation.

3.1. Key issues

By studying the characteristics of large-scale heterogeneous networks and the related work of data collection, we conclude the key issues to be addressed in the security-related data collection in the next generation network systems.
3.2. Design requirements

We propose the design requirements of security-related data collector in terms of both functionality and security based on our previous literature surveys about security-related data collection in main types of networks (Lin et al., 2018; He et al., 2018; Liu et al., 2018; Xie et al., accepted; Jing et al., 2018; Zhou et al., 2018; Li et al., 2018), i.e., the Internet, LTE-A cellular networks, wireless sensor networks and mobile ad hoc networks. The designed adaptive security-related data collector needs to meet the requirements as described below.

3.2.1. Functional requirements (FR)

Functional Requirements (FR) are functions that need to be implemented during security-related data collection and storage. The functional requirements of the adaptive security-related data collector are proposed as follows:

[FR1]: Must be able to justify which data would be collected, that is, to determine collection targets.
[FR2]: Must be able to store collected data in a storage medium.
[FR3]: Must be able to export data to other entities or allow other entities to access the collected data.
[FR4]: Must be universal and generic and be able to support a variety of networking application scenarios.
[FR5]: Must be able to collect required security-related data in a heterogeneous network system.
[FR6]: Must be flexible, efficient and stable when collecting security-related data.
[FR7]: Must not cost too much computation and storage resources in collecting security-related data.
[FR8]: Must not destroy or affect the normal operations of the network application system.
[FR9]: Must be automatic and adaptive in terms of networking context changes.

3.2.2. Security requirements (SR)

Security Requirements (SR) are the requirements about security issues relating to security-related data collection, storage and access. The security requirements of the adaptive security-related data collector are summarized as below:

[SR1]: Must be able to prevent data leakage, loss and failure in the storage of the security-related data collected (confidentiality, integrity and availability of the collected data).
[SR2]: Must not affect the accuracy of collected security-related data (data integrity).
[SR3]: Must be able to verify the integrity and authenticity of the collected data (data integrity and authenticity).
[SR4]: Must be able to verify the identity of data requesters when they are accessing the collected data (authenticity and non-repudiation).

4. Adaptive security-related data collector

In this section, we introduce the adaptive security-related data collector that is designed based on the above research methodology. We first introduce the collector by specifying its working procedure. Then, we introduce security-related data that can be collected in our prototype system based on an Android mobile platform, followed by SDDL design and adaptive sampling mechanisms applied for data collection.

4.1. Overview

The system model of the collector is given in Fig. 2. The adaptive data collector collects the security-related data at any network nodes and transmits the data to a server. Herein, there are many issues that need to be solved. For example, a mobile device equipped with an adaptive security-related data collector can access multiple network systems, thus can play as a network node in various types of networks that can be switched among each other based on mobile device user's demands. In order to support complex network topologies and dynamic switching among multiple types of network systems in a large-scale heterogeneous network, we propose an adaptive method for security-related data collection. The basic idea of our work is the data that need to be collected are expressed and described in XML files based on different network contexts. Based on network context detection, an XML parser parses a corresponding XML file. The parsing result provides the information of the data that should be collected in the underlying network context. It plays as an instructor to guide data collection (i.e., with which data collectors to collect what data), and data composition (with which

Fig. 2. A system model.
algorithms to compose what data) in order to measure the security of a network system.

A system structure of the adaptive security-related data collector based on network context is shown in Fig. 3. In the collector, the network context detector detects underlying network system context and drives the SDDL parser to parse the SDDL file corresponding to the network context. After parsing the SDDL file, the SDDL parser calls needed data collectors to collect security-related data. In order to ensure the confidentiality and integrity of data, the collected security-related data should be protected, e.g., by applying encryption and hash operation, as well as signing the signatures of collectors. The security-related data collected in a network node can be sent to the server to perform attack detection and measure the security of the entire network system.

In what follows, we introduce the detailed design and implementation of adaptive security-related data collector.

4.2. Security-related data

Determining collection target, i.e., what security-related data should be collected, is the first step in security-related data collection. Based on the literature study as reported in Section 2, we determined the collection target of our prototype, as shown in Fig. 4. Due to space limitation, we do not introduce the concept and function of each security-related data herein. Detailed information can refer to our previous work (He et al., 2018; Liu et al., 2018; Xie et al., accepted; Jing et al., 2018). Due to the limitations of experimental conditions, we use a multi-mode terminal in LTE and WLAN (to access the Internet) environments to simulate a heterogeneous network system. Therefore, we only focus on the security-related data at mobile terminals in these two networking environments. The integration of LTE and WLAN to form a heterogeneous network is the most popular heterogeneous networking mode nowadays. The value of a given LTE network can be enhanced by supplementing it with WLAN by providing additional services. For example, a smartphone can seamlessly complement an LTE network with a nearby WLAN network whenever there is weak or no LTE coverage (Singh et al., 2014).

Fig. 4 provides some example security-related data that should be collected in the above network context. We can see that in the above networking environments, security-related data include network data and host data. The network data contains the data collected in WLAN and LTE. In WLAN, such security-related data as network protocol, packet size in all network layers, source IP, destination IP in network layer, and destination port and source port in transport layer are collected. In LTE, SMS and SIP messages are collected for potential attack or intrusion detection. In terms of the host data that are collected in any networking contexts, system level data such as memory usage and CPU occupation time, hardware related data such as battery consumption, and application level data such as process count are collected for supporting network attack and intrusion detection.

4.3. Security-related Data Description Language

We designed a Security-related Data Description Language (SDDL) based on XML that can facilitate the description, collection, and processing of security-related data in heterogeneous networks. To this regard, it facilitates the formation of a unified standard and model for expressing security-related data. Additional security-related data and data processing and analysis algorithms can be added into the XML file to facilitate application of newly advanced mechanisms for network security measurement. The SDDL can be flexibly configured and extended to contain descriptions on security-related data based on practical needs. In short, the SDDL is used to describe security-related data to be collected in different access networks and instruct security-related data collection under a concrete network context.

XML is a plain-text markup language for expressing, storing, and transmitting information and it is platform independent. We design the SDDL based on XML, SDDL specifies what kind of security-related data should be collected in which way, under which networking context. It marks the tags about data processing methods and the target attacks that the data can be used to detect. As shown in Fig. 5, we describe the source IP of network packet with SDDL. According to an exemplary embodiment, in addition to the data name, other information elements are also marked in data description for the purpose of collection, such as data type, collection priority, collection method and so on. Here are some explanations of the exemplary information elements comprised in the SDDL:

Network-type: the specific access network in which the indicated network data need to be collected;
network-node-type: the specific network node in which the indicated network data need to be collected;

- network-protocol: the network protocol used in the network system;
- data-type: the storage type of the data;
- data-category: the category of data that is used in the classification for data composition and analytics;
- data-importance-level: the importance level of the data;
- collection-priority: the priority of the data in the collection process;
- collection-method (collector ID): the identifier of the collection method or a corresponding data collector;
- processing-algorithm: the algorithm used to process or pre-process the collected data;
- composition-tag: the tag that indicates the security threats or attacks that could be detected with the collected data.

A network context detector in adaptive security-related data collector is designed and developed, which can detect network context (e.g., an Internet host, a MANET node, an LTE Base Station, etc.) and make it as input of a SDDL parser to parse the SDDL described security-related data in order to figure out what kind of data should be collected in the underlying context and which data collection toolkits or applications should be triggered to collect them. Then, the collector can perform adaptive data collection based on the network context. As shown in Fig. 2, based on SDDL description, our collector can work in a heterogeneous network with dynamic handover between multiple access networks. The information encapsulated in the XML file can guide us to conduct adaptive security-related data collection and processing. After data collection, we can perform data composition and process for network security analysis based on the tags marked on the data. With the detection result, it becomes easy to figure out the security holes and measure the security level of the whole network system. In summary, the proposed SDDL has the following six advantages:

1) Describe security-related data to make it clearer for both network devices and users;
2) By marking the security-related data in a SDDL file, it is possible to achieve purposeful collection instead of collecting data without concrete targets;
3) Detect the network context and then collect expected security-related data according to the instruction specified in the SDDL file corresponding to the network context in order to overcome the heterogeneity issue of the heterogeneous network.
4) Theoretically, through the “network-node-type” encapsulated in SDDL, our collector can collect security-related data at any node in the heterogeneous network. Thus, it is a universal data collection solution.
5) By describing the attributes of the security-related data, it is possible to facilitate the subsequent storage, access, and processing of the security-related data.
6) By introducing SDDL, the coupling degree of data collection and network systems can be reduced, thereby improving the scalability and extensibility of the data collector.

Although the security-related data to be collected are described with the SDDL based on XML in this paper, it is also possible to describe the security-related data with other types of languages, such as various data exchange and markup languages like HTML, JSON, YAML, XHTML, etc. In practice, the selection of the descriptive language used to specify the network data may depend on system implementation preference and convenience.

4.4. Network context and adaptive collection strategy

In this paper, we study the issue of security-related data collection when access switching occurs and attempt to solve it through network
context detection. The notion and the use of the term context has represented an important issue in computer science, and more recently in several emerging domains, including heterogeneous networks, performance-oriented applications and autonomic communications (Makris et al., 2013a, 2013b; Marabissi and Fantacci, 2015). Context can be related to any information that can be used to characterize the situation of an entity, including its location, speed, identity, time, user preferences and activities. Its concrete connotation should depend on different application scenarios and requirements. Our purpose is to determine the access network type of the heterogeneous network in order to determine the security-related data that needs to be collected.

To demonstrate the functionality and performance of our collector, we implement our prototype in a smartphone that can connect to the Internet via WLANs or mobile data service, and send SMS or make phone calls over VoLTE. Therefore, we choose WLAN and mobile data service as the network context of the Internet and choose a mobile call and a SMS as the network contexts of LTE network in the design of the prototype collector. Of course, as the function of collector continues to expand, the network contexts of the LTE network in the design of the prototype collector. Of course, as the function of collector continues to expand, the network context before performing concrete security-related data collection, the network context that needs to be detected could be different.

Detailed functions of the network context-based security-related data collection is shown in Fig. 6. When a SMS is received, we collect the content of the incoming SMS. When a call arrives, we collect SIP messages in the VoLTE communication data. When the smartphone accesses the Internet through WLAN or a mobile data service, we collect the Internet communication data packets. Herein, we collect the data shown in Fig. 4. They are also the security-related data we collect in the prototype system.

The proposed collector achieves adaptive security-related data collection by applying SDDL, detecting network context, and adaptively adjusting collection frequency. Detailed description of Adaptive Security-related Data Determination based on Network Context (ASDDNC) algorithm is described below. We provide comments in the algorithm to help easy understanding. The core idea of Algorithm 1 is to determine data collection targets by parsing the SDDL based on a detected network context before performing concrete security-related data collection, protect the collected data for secure storage and adjust data collection interval for later collection.

Algorithm 1 Adaptive Security-related Data Determination Based on Network Context (ASDDNC).

```matlab
Input: λ /* λ is the parameter for calculating the next sampling interval */
Input: SDDL_LTE, SDDL_Internet, SDDL_Host are the data contents described with SDDL in different network contexts */
/* parameter initialize phase */
Initialize: security_Data_SDDL /* defining data collected by collector */
Initialize: security_Data /* defining the initial value of the sampling interval */
Initialize: Sampling_Interval_Init /* defining the current value of the sampling interval */
Initialize: sampling_Interval_Cur ← Sampling_Interval_Init /* parse the SDDL that describes the host data */
security_Data_SDDL ← SDDLParser(SDDL_Host) security_Data_SDDL.append(security_Data_SDDL)
/* parse the SDDL that describes the Internet data */
security_Data_SDDL ← SDDLParser(SDDL_Internet) security_Data_SDDL.append(security_Data_SDDL)
/* parse the SDDL that describes the LTE data */
security_Data_SDDL ← SDDLParser(SDDL_LTE) security_Data_SDDL.append(security_Data_SDDL)
/* detect current network context */
If (network_context = Internet) Then
   /* parse the SDDL that describes the Internet data */
   security_Data_SDDL ← SDDLParser(SDDL_Internet) security_Data_SDDL.append(security_Data_SDDL)
End If
If (network_context = LTE) Then
   /* parse the SDDL that describes the LTE data */
   security_Data_SDDL ← SDDLParser(SDDL_LTE) security_Data_SDDL.append(security_Data_SDDL)
End If
/* data collection phase */
security_Data ← DataCollect(security_Data_SDDL) /* data storage phase includes data encryption, hashing, and storage */
security_Data_Enc ← EncryptedData(security_Data, key) security_Data_Enc, Hash ← HashData(security_Data_Enc)
DataStore(security_Data_Enc, security_Data_Enc, Hash)
/* calculate next sampling interval and different algorithms use different parameters */
sampling_Interval_Cur ← next_sampleInterval (λ, security_Data)
Wait(sampling_Interval_Cur)
4.5. Adaptive collection frequency adjustment strategies

In this part, we introduce three traditional statistical sampling algorithms and two proposed adaptive sampling algorithms to adjust collection frequency in order to improve collection efficiency and reduce the amount of collected data. Collection frequency adjustment strategies generally have three trigger modes: count trigger (i.e., counters), timing trigger (i.e., timers), and event trigger. In practice, the trigger mode needs to be selected according to networking scenarios.

4.5.1. Traditional statistical sampling algorithms

4.5.1.1. Simple random sampling. The simple random sampling is to generate a random number N before collecting one data and compare it with a threshold N_T set in advance (Hu et al., 2006). If the random number is smaller than the threshold, the next data is collected, otherwise, the next data is not collected. The choice of threshold determines the collection frequency. In this paper, we generate a random number within 0–1 and the threshold is the desired collection frequency. If we want to collect one data every 60 s or every 60 pieces of data, the
threshold $N_T$ is $1/60$. The formulas are as follows:

$$N = \text{random(seed)}$$  \hspace{1cm} (1)

$$\text{bool } b = N \leq N_T$$  \hspace{1cm} (2)

where random(seed) is a random number generation function and seed is the seed for random number generation. If b is true, data is collected; else, data is not collected.

4.5.1.2. Systematic sampling. Data collection based on systematic sampling is relatively simple (Ariyapala et al., 2016). We only need to set a fixed period length $T$ in advance, and then the collector collects data at a fixed collection frequency $T, 2T, 3T, \ldots, nT$. The choice of collection period directly affects the quality of the collection results. If we want to collect one data every 60 s or every 60 pieces of data, the threshold $T$ is 60.

4.5.1.3. Poisson sampling. Poisson sampling is a sampling algorithm recommended by Internet Engineering Task Force (IETF) RFC 2330 for network data collection in IP network measurement (Paxson et al., 1998). The Poisson sampling algorithm is based on Poisson distribution. Poisson distribution is a discrete probability distribution that expresses the probability of a given number of events occurring in a fixed interval of time or space. Its inverse function is the collection time or space interval, that is, the time period when there is no event. An event can occur at any random time or space. Its inverse function is the collection time or space interval, also called event rate parameter. The probability of observing $k$ events in an interval is given by the equation:

$$P(X(t) = k) = \frac{e^{-\lambda} \lambda^k}{k!}, \quad k = 0, 1, \ldots$$  \hspace{1cm} (3)

where $\lambda$ is the number 2.71828 … (Euler’s number), which is the base of the natural logarithms; $k$ takes values $0, 1, 2, \ldots, n$. When $k = 0$, that is, there is no event in the time interval:

$$k = 0, \quad P(X(t) = 0) = \frac{e^{-\lambda} \lambda^0}{0!} = e^{-\lambda}$$  \hspace{1cm} (4)

Then the inverse function of Formula (5) is the Poisson sampling formula:

$$t = \frac{1}{\lambda} \ln P(X(t) = 0)$$  \hspace{1cm} (5)

$$G_i(X_i) = \frac{1}{2} \ln X_i, \quad X_i \sim \text{Uniform}(0, 1)$$  \hspace{1cm} (6)

Because mathematical expectation of the exponential distribution is $1/\lambda$, $\lambda$ is equal to 1/(desired sampling interval) in the application. In other words, if the collector or user expects to sample once every 60 s or 60 pieces of data, then $\lambda = 1/60$. Assume that the time intervals of Poisson distribution are $T_1, T_2, T_3, T_4, T_5, \ldots, T_n$, then the collection time points are $T_1, T_1 + T_2, T_1 + T_2 + T_3, \ldots, T_1 + T_2 + T_3 + \ldots + T_n$.

4.5.2. Adaptive sampling algorithms

Based on in-depth research and analysis, we find that when using the sampling method to collect data, the collection frequency needs to be determined according to specific conditions. If the data variation is large, the collection interval should be reduced. Otherwise, the variation trend of data cannot be reflected. If the data variation is small, the collection interval can be increased, so as to reduce the amount of data collected while ensuring the accuracy of data collection. The traditional statistical sampling algorithms cannot adaptively adjust the data collection frequency in different conditions, so it has shortcomings in terms of efficiency, economy and accuracy of data collection.

For adjusting the data collection frequency adaptively, we need to address two issues. First, how to calculate the data variation. Second, how to adjust the frequency of data collection according to the data variation. For the first issue, we can calculate data variation based on prediction. For the second issue, the data collection frequency should be adjusted according to the data variation. In order to address these two issues, we designed two adaptive collection frequency adjustment algorithms as described below.

4.5.2.1. ACFAS_PVR. We find that the data variation can be calculated by calculating the ratio of the predicted data value in the next interval to the data value already collected. Therefore, we propose an Adaptive Collection Frequency Adjustment Strategy Based on Predicted Variation Ratio (ACFAS_PVR). Regression algorithms can be used for prediction, such as linear regression, Support Vector Regression (SVR), logistic regression, KNN regression, etc.

First, we assume that some of the collected data is $y_i$. Where $i$ takes values of $t + 1, t + 2, t + 3, \ldots, t + N_T$ and $t$ is a certain time point. $y_i$ is the fitting data used for the prediction and $N_T$ is the data amount of the fitting data. We define that the data predicted by the fitting data $y_i$ is $\hat{y}_i$. Where $j$ takes values $k + 1, k + 2, k + 3, \ldots, k + N_p$ and $k = t + N_T$. The mean values of the fitting data and predicted data are:

$$M_f = \frac{1}{N_f} \sum_{i=t+1}^{t+N_T} y_i$$  \hspace{1cm} (7)

$$M_p = \frac{1}{N_p} \sum_{j=k+1}^{k+N_p} \hat{y}_j$$  \hspace{1cm} (8)

Therefore, the ratio of the mean value $R_M$ of the fitting data and predicted data can represent the data variation. $R_M$ is a value that changes around 1. When $R_M$ is 1, the data does not change substantially, but when $R_M$ is significantly greater or less than 1, the data changes obviously. Data collection frequency should be adjusted accordingly to the data variation. Formula (10) gives a specific collection frequency adjustment method based on data variation. Where $T_i$ represents the current sampling interval and $T_{i-1}$ represents the previous sampling interval. $T_{inc}$ represents the amount of change when the sampling interval is increased. $T_{dec}$ represents the amount of change when the sampling interval is reduced. $Th_{ir}$ and $Th_{re}$ are two thresholds for discriminating data variation. When $R_M = \frac{M_f}{M_p}$ is greater than $Th_{ir}$ and less than $Th_{re}$, which indicates that the data variation is small, the sampling interval should be increased in this situation. When $R_M$ is greater than $Th_{ir}$, or less than $Th_{re}$, which indicates that the data changes greatly, the sampling interval should be reduced in this case.

$$R_M = \frac{M_p}{M_f}$$  \hspace{1cm} (9)

$$T_i = \begin{cases} T_{i-1} + T_{inc}, & \text{if} \quad Th_{ir} \leq R_M \leq Th_{re} \\ T_{i-1} + T_{dec} \times ((R_M - 1) \times 10), & \text{if} \quad R_M > Th_{ir} \\ T_{i-1} + T_{dec} \times ((1 - R_M) \times 10), & \text{if} \quad R_M < Th_{re} \end{cases}$$  \hspace{1cm} (10)

Of course, we can also use the ratio of the variance $R_D$ to represent the data variation, as defined below. Then, we use $R_D$ to adjust the sampling interval.

$$D_f = \frac{1}{N_f} \sum_{i=t+1}^{t+N_T} (y_i - M_f)^2$$  \hspace{1cm} (11)

$$D_p = \frac{1}{N_p} \sum_{j=k+1}^{k+N_p} (\hat{y}_j - M_p)^2$$  \hspace{1cm} (12)
\[ R_p = \frac{D_p}{\bar{D}_p} \quad (13) \]

The above mean-based strategies and variance-based strategies are named ACFAS_PVR_M and ACFAS_PVR_V respectively. Detailed description of ACFAS_PVR is described in Algorithm 2.

**Algorithm 2**
Adaptive Collection Frequency Adjustment Strategy Based on Predicted Variation Ratio.

- **Input:** fit_num, predict_num, flag
- **Procedure:**
  - Initialize: \[ M, M_p, \bar{D}, \bar{D}_p \] (14).
  - If flag = True:
    - Calculate the next sampling interval:
      - Predicted Data:
        - \[ M = \frac{1}{N_p} \sum_{i=1}^{N_p} \tilde{y}_i \] (14).
        - \[ M_p = \frac{1}{N_p} \sum_{i=1}^{N_p} \hat{y}_i \] (15).
    - Therefore, the ratio of the mean value \( R_p \) of the predicted data and real data can represent the data variation. \( R_p \) is a value that changes around 1. When \( R_p \) is 1, the data does not change substantially, but when \( R_p \) is significantly greater or less than 1, the data changes obviously. The specific collection frequency adjustment method based on data variation in ACFAS_PAR is the same as ACFAS_PVR. Notably, depending on the application scenario, different collection frequency adjustment methods may be selected based on data variation.

- Of course, we can also use the ratio of the variance \( R_0 \) to represent the data variation.

\[ D_p = \frac{1}{N_p} \sum_{i=1}^{N_p} (y_i - M_p)^2 \] (17).
\[ D_p = \frac{1}{N_p} \sum_{i=1}^{N_p} (y_i - M_p)^2 \] (18).
\[ R_0 = \frac{D_0}{\bar{D}_p} \] (19).

The above mean-based strategies and variance-based strategies are named ACFAS_PAR_M and ACFAS_PAR_V respectively. Detailed description of ACFAS_PAR is described in Algorithm 3.

**Algorithm 3**
Adaptive Collection Frequency Adjustment Strategy Based on Predicted Accuracy Ratio.

- **Input:** fit_num, predict_num, flag
- **Procedure:**
  - Calculate the next sampling interval:
    - Predicted Data:
      - \[ M = \frac{1}{N_p} \sum_{i=1}^{N_p} \tilde{y}_i \] (14).
      - \[ M_p = \frac{1}{N_p} \sum_{i=1}^{N_p} \hat{y}_i \] (15).
    - Calculate the new sampling interval:
      - Predicted Data:
        - \[ M = \frac{1}{N_p} \sum_{i=1}^{N_p} \tilde{y}_i \] (14).
        - \[ M_p = \frac{1}{N_p} \sum_{i=1}^{N_p} \hat{y}_i \] (15).
    - Therefore, the ratio of the mean value \( R_p \) of the predicted data and real data can represent the data variation. \( R_p \) is a value that changes around 1. When \( R_p \) is 1, the data does not change substantially, but when \( R_p \) is significantly greater or less than 1, the data changes obviously. The specific collection frequency adjustment method based on data variation in ACFAS_PAR is the same as ACFAS_PVR. Notably, depending on the application scenario, different collection frequency adjustment methods may be selected based on data variation.

4.5.2.2. ACFAS_PAR. Data variation can also be represented by calculating the ratio of predicted accuracy. The prediction accuracy ratio is the ratio of the predicted value of the data to the real value of the data. When the predicted value of the data is close to the real value, it indicates that the data variation is small, and when the predicted value of the data is very different from the real value, the data variation is large. Therefore, we propose an Adaptive Collection Frequency Adjustment Strategy Based on Predicted Accuracy Ratio (ACFAS_PAR).

First, we assume that some of the collected data is \( y_i \). Where \( i \) takes values \( t+1, t+2, t+3, \ldots \), \( t+N_r \) and \( t \) is a certain time point. \( y_i \) is the real data and \( N_r \) is the amount of the real data. We define that the data predicted is \( \tilde{y}_i \) and \( N_p = N_r \). The mean values of the real data and the predicted data are:

\[ M = \frac{1}{N_r} \sum_{i=1}^{N_r} y_i \] (14).
\[ M_p = \frac{1}{N_r} \sum_{i=1}^{N_r} \hat{y}_i \] (15).

Therefore, the ratio of the mean value \( R_p \) of the predicted data and real data can represent the data variation. \( R_p \) is a value that changes around 1. When \( R_p \) is 1, the data does not change substantially, but when \( R_p \) is significantly greater or less than 1, the data changes obviously. The specific collection frequency adjustment method based on data variation in ACFAS_PAR is the same as ACFAS_PVR. Notably, depending on the application scenario, different collection frequency adjustment methods may be selected based on data variation.

\[ R_p = \frac{D_p}{\bar{D}_p} \] (19).

Of course, we can also use the ratio of the variance \( R_0 \) to represent the data variation.

\[ D_p = \frac{1}{N_p} \sum_{i=1}^{N_p} (y_i - M_p)^2 \] (17).
\[ D_p = \frac{1}{N_p} \sum_{i=1}^{N_p} (y_i - M_p)^2 \] (18).
\[ R_0 = \frac{D_0}{\bar{D}_p} \] (19).

The above mean-based strategies and variance-based strategies are named ACFAS_PAR_M and ACFAS_PAR_V respectively. Detailed description of ACFAS_PAR is described in Algorithm 3.
Algorithm 3 (continued)

Else If (len(y) % 2 > 0) Then
    sampling_INTERVAL ← Sampling_INTERVAL_Init
End If

// data collection phase
y ← dataCollect()
x_list.append(x)
y_list.append(y)
sampling_INTERVAL ← sampling_INTERVAL + 1
wait(sampling_INTERVAL)

End

/* next_sampling_INTERVAL: two functions that calculate the sampling interval */

Procedure: next_sampling_INTERVAL_Mean(x_list, y_list)

y_fit ← y_list[0:fit_num]
regressionModel.fit(x_list, y_fit)
x_predict ← x_list[fit_num:fit_num + predict_num]
y_predict ← regressionModel.predict(x_predict)

/* Calculate the next sampling interval */

x_real ← x_predict
y_real ← y_list[fit_num:fit_num + predict_num]
ratio ← mean(y_real)/mean(y_predict)

If ratio < upper_Threshold Then
    sampling_INTERVAL ← sampling_INTERVAL + Sampling_INTERVAL_Init
End If

/* data sampling interval */

sample_INTERVAL ← Sampling_INTERVAL_Adj_Inc + Sampling_INTERVAL_Cur

Else If ratio > lower_Threshold Then
    sample_INTERVAL ← Sampling_INTERVAL_Adj_Dec + (Sampling_INTERVAL_Init * (1-ratio))
End If

Else If ratio < upper_Threshold Then
    sample_INTERVAL ← Sampling_INTERVAL_Adj_Dec + (Sampling_INTERVAL_Init * (ratio-1))
End If

End

5. Performance evaluation

5.1. Applicability

5.1.1. Security-related data collection. Applicability refers to whether the designed adaptive security-related data collector is applicable, that is, whether it can be used to collect security-related data in a heterogeneous network. Fig. 7 shows the host security-related data, Internet security-related data and LTE security-related data collected by the adaptive security-related data collector in an Android smartphone. These security-related data are described in Part 3 of Section 4. The data collected above show that our collector is applicable and can be used to collect security-related data in a heterogeneous network.

5.1.2. Network context detection. In order to test the context awareness function of the collector, we designed an experiment to verify that it can adaptively collect security-related data when the network context changes. Fig. 8 shows the differences in data collection as the network context changes in time series. First, when the collector is started, host security-related data are collected until the collector terminates. When the device is connected to the Internet, the collector starts collecting Internet security-related data until the device disconnects from the Internet. When the network context switches to LTE, that is, when a call or SMS arrives, the collector will collect the related data about call and SMS.

5.1.2. Efficiency

In order to evaluate the efficiency of the adaptive security-related data collector, we test its memory footprint, CPU usage and power usage at run time.

5.1.2.1. Memory footprint. Memory footprint of the adaptive security-related data collector running for 30 min is shown in Fig. 9. We can see that the memory footprint of the security-related data collector is around 50 MB and the ratio is 1.22% for our test equipment, which is not high. This means that our proposed collector occupies a very low memory while working on an Android smartphone.

5.1.2.2. CPU usage. Fig. 10 shows the CPU usage of the adaptive security-related data collector. We can see that the CPU usage of the adaptive security-related data collector is very low because there is almost no computational requirement in the collection process except for the cryptographic operations and hash operations after data collection.

In this section, we evaluated the performance of the adaptive security-related data collector based on prototype implementation. Our implementation is based on the Android operating platform in the context of the Internet and LTE network environments. The test equipment is Huawei Honor 6X with an eight-core CPU of Kirin 655, 4*2.1 GHz work CPU frequency, a RAM of 4 GB and a ROM of 32 GB. Next, we provide performance evaluation results based on the prototype in terms of applicability, efficiency, and accuracy.
Fig. 8. Adaptive security-related data collector log records and corresponding data collected when the network contexts change.

Fig. 9. Memory footprint of adaptive security-related data collector.

Fig. 10. CPU usage of adaptive security-related data collector.

Fig. 11. Power usage of adaptive security-related data collector.
5.1.2.3. Power usage. Fig. 11 shows the power usage of the adaptive security-related data collector running for 30 min. Adaptive security-related data collector consumes 7.84–3.9 = 3.94 mAh in 30 min. That is, it consumes 0.002189 mAh per second and if we do not consider the battery consumption of the Android system and other applications, it can run continuously for 424 h in the testing equipment. This is a very power-economic.

Some works (Jiewu et al., 2014; Chen et al., 2013) use Support Vector Regression (SVR) to predict traffic in order to adaptively collect data. However, the computational complexity of SVR is much higher than that of linear regression. J. L. García-Dorado et al. (2008) propose a mechanism to down sample traffic volume using multi-resolution analysis with wavelets with high computational complexity.

5.1.3. Accuracy

We expect to analyze the collected security-related data for attack and intrusion detection. However, the quality of any analysis relies on the quality of the data being analyzed. An important concern of data collection is data quality (Zhang et al., 2018). Collected security-related data is often incomplete, sometimes inaccurate, and generally prone to biases, e.g., due to the collection methods or strategies. Therefore, we evaluate the accuracy of security-related data collected by different sampling algorithms in this part. First, we give several evaluation criteria related to data collection.

5.1.3.1. Sampling times. Sampling times refers to the number of times the data is collected by the collector, which affects the accuracy of the data. The more sampling times, the higher the accuracy of data, but the greater the amount of data collected, the heavier the burden brought to an application system. So, the evaluation of data accuracy needs to be based on the same sampling times.

5.1.3.2. Evaluation function. We use vertical distance between sampling data curve and real data curve to represent residual error:

\[ e = |f(x) - f(x)| \]

where \( f(x) \) is the real data curve, and \( f(x) \) is the sampling data curve. Based on the concept of residual error, we give several evaluation functions that can evaluate the accuracy of the data.

**SAE (Sum of Absolute Error) or SAD (Sum of Absolute Difference):**

\[ E_{SAE}(n) = \int_{x_1}^{x_n} |f(x) - f(x)| \, dx \]

\[ E_{SAD}(n) = \sum_{i=1}^{n} |f(x_i) - f(x_i)| \]

**MAE (Mean Absolute Error) or MAD (Mean Absolute Difference):**

\[ E_{MAE}(n) = \frac{1}{x_n - x_1} \int_{x_1}^{x_n} |f(x) - f(x)| \, dx \]

\[ E_{MAD}(n) = \frac{1}{n} \sum_{i=1}^{n} |f(x_i) - f(x_i)| \]

**SSE (Sum of Squared Error) or SSD (Sum of Squared Difference):**

\[ E_{SSE}(n) = \int_{x_1}^{x_n} (f(x_i) - f(x_i))^2 \, dx \]

Then, based on the above evaluation functions, we design three experiments to evaluate the accuracy of the three traditional sampling algorithms and the two proposed adaptive sampling algorithms. In the experiments, we use the system memory footprint as a target data for collection. The sampling algorithm needs to initialize some parameters before it is applied. Tables 1 and 2 give the parameter values of Algorithm 2 and Algorithm 3 that were used in the three experiments, respectively. The parameter setting is very flexible and can be configured according to the specific application scenario.

5.1.3.3. Experiment 1. Fig. 12 shows the sampling results of the collector. Due to space limitations, we only test the mean-based ACFAS_PVR and mean-based ACFAS_PAR where linear regression was used as a predictive model.

### Table 1

<table>
<thead>
<tr>
<th>Parameter settings of Algorithm 2.</th>
<th>E-1</th>
<th>E-2</th>
<th>E-3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Notation</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>fit_num</td>
<td>5</td>
<td>10</td>
<td>5</td>
</tr>
<tr>
<td>predict_num</td>
<td>5</td>
<td>10</td>
<td>5</td>
</tr>
<tr>
<td>Max_Sampling_Interval</td>
<td>25s</td>
<td>80s</td>
<td>120s</td>
</tr>
<tr>
<td>Min_Sampling_Interval</td>
<td>10s</td>
<td>10s</td>
<td>20s</td>
</tr>
<tr>
<td>Sampling_Interval_Init</td>
<td>5s</td>
<td>10s</td>
<td>20s</td>
</tr>
<tr>
<td>Sampling_Interval_Adj_Inc</td>
<td>5s</td>
<td>10s</td>
<td>20s</td>
</tr>
<tr>
<td>Sampling_Interval_Adj_Dec</td>
<td>20s</td>
<td>80s</td>
<td>120s</td>
</tr>
<tr>
<td>upper_Threshold</td>
<td>1.20</td>
<td>1.20</td>
<td>1.20</td>
</tr>
<tr>
<td>lower_Threshold</td>
<td>0.80</td>
<td>0.80</td>
<td>0.80</td>
</tr>
<tr>
<td>predict_Atom_Interval</td>
<td>10s</td>
<td>10s</td>
<td>10s</td>
</tr>
<tr>
<td>configurationModel</td>
<td>LR</td>
<td>LR</td>
<td>LR</td>
</tr>
<tr>
<td>next_samplingInterval</td>
<td>M</td>
<td>M</td>
<td>M</td>
</tr>
</tbody>
</table>

E: Experiment; LR: Linear Regression; M: mean-based ACFAS_PVR; s: second.

### Table 2

<table>
<thead>
<tr>
<th>Parameter settings of Algorithm 3.</th>
<th>E-1</th>
<th>E-2</th>
<th>E-3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Notation</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>fit_num</td>
<td>15</td>
<td>10</td>
<td>5</td>
</tr>
<tr>
<td>predict_num</td>
<td>15</td>
<td>10</td>
<td>5</td>
</tr>
<tr>
<td>Max_Sampling_Interval</td>
<td>50s</td>
<td>80s</td>
<td>120s</td>
</tr>
<tr>
<td>Min_Sampling_Interval</td>
<td>5s</td>
<td>20s</td>
<td>20s</td>
</tr>
<tr>
<td>Sampling_Interval_Init</td>
<td>5s</td>
<td>10s</td>
<td>20s</td>
</tr>
<tr>
<td>Sampling_Interval_Adj_Inc</td>
<td>5s</td>
<td>20s</td>
<td>20s</td>
</tr>
<tr>
<td>Sampling_Interval_Adj_Dec</td>
<td>50s</td>
<td>80s</td>
<td>120s</td>
</tr>
<tr>
<td>upper_Threshold</td>
<td>1.10</td>
<td>1.15</td>
<td>1.20</td>
</tr>
<tr>
<td>lower_Threshold</td>
<td>0.90</td>
<td>0.85</td>
<td>0.80</td>
</tr>
<tr>
<td>configurationModel</td>
<td>LR</td>
<td>LR</td>
<td>LR</td>
</tr>
<tr>
<td>next_samplingInterval</td>
<td>M</td>
<td>M</td>
<td>M</td>
</tr>
</tbody>
</table>

E: Experiment; LR: Linear Regression; M: mean-based ACFAS_PAR; s: second.
expected value of the collection interval is small.

5.1.3.4. Experiment 2. Fig. 13 shows the sampling results of the collection methods based on the five sampling algorithms and Table 4 shows the accuracy evaluation results when the expected value of the collection interval is 50. Based on this experiment, we can find that as the collection interval increases, simple random sampling algorithm is significantly worse than the other four algorithms. Furthermore, the two proposed adaptive sampling algorithms perform better than the three traditional statistical sampling methods with regard to sampling times and errors.

5.1.3.5. Experiment 3. Fig. 14 shows the sampling results of the collection methods based on the five sampling algorithms and Table 5 shows the accuracy evaluation results when the expected value of the collection interval is 100. Similar to the results of Experiment 2, the two proposed adaptive sampling algorithms perform better than the three traditional statistical sampling methods with regard to sampling times and errors.

From the above experimental results, we can see that both ACFAS_PVR and ACFAS_PAR can perform better than simple random sampling, systematic sampling and Poisson sampling, regardless of the length of the sampling interval. With the collection interval increasing, the performance of Poisson sampling and systematic sampling becomes worse. However, ACFAS_PVR and ACFAS_PAR still exhibit high collection accuracy and can reduce the amount of collected data while ensuring a certain collection accuracy. ACFAS_PVR can reduce 35% of SAE or MAE and reduce 50% of SSE or MSE compared to the three traditional statistical sampling methods. ACFAS_PAR can reduce 40% of SAE or MAE and reduce the 60% of SSE or MSE compared to traditional statistical sampling methods. Therefore, ACFAS_PAR performs better than ACFAS_PVR. One possible reason is that the accuracy ratio of prediction

<table>
<thead>
<tr>
<th>Evaluation Criteria</th>
<th>Sampling Algorithms</th>
<th>Sampling times</th>
<th>SAE</th>
<th>MAE</th>
<th>SSE</th>
<th>MSE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Simple Random Sampling (Hu et al., 2006)</td>
<td>(P = 1/25)</td>
<td>212</td>
<td>33881.564</td>
<td>6.795</td>
<td>5274699.752</td>
</tr>
<tr>
<td></td>
<td>Poisson Sampling (λ = 1/25)</td>
<td></td>
<td>190</td>
<td>23268.160</td>
<td>4.708</td>
<td>1838469.879</td>
</tr>
<tr>
<td></td>
<td>ACFAS_PVR</td>
<td></td>
<td>(205-10 = 195)</td>
<td>17550.083</td>
<td>3.524</td>
<td>1134542.025</td>
</tr>
<tr>
<td></td>
<td>ACFAS_PAR</td>
<td></td>
<td>(208-30 = 178)</td>
<td>15015.228</td>
<td>3.029</td>
<td>500305.234</td>
</tr>
</tbody>
</table>

Table 3: Results of experiment 1.
data is more accurate in terms of calculating the change of data than the variation ratio of prediction data. However, ACFAS_PVR is easier to implement than ACFAS_PAR. Furthermore, the computational complexity of these two adaptive sampling algorithms is higher than the three traditional statistical sampling algorithms. Therefore, these five sampling algorithms deployed in our collector should be selected based on the application conditions and requirements.

In summary, our proposed collector can collect security-related data in heterogeneous networks and has high collection performance and accuracy, with reduced collected data volume. Furthermore, it also has high flexibility, adaptability and low cost. It meets the nine functional requirements presented in Section 3.

R. Gad et al. (2015) also proposed a traffic collection approach named Self-adaptive Sampling which is adapted dynamically depending on the current collector load. The goal of this approach is to avoid system performance degradation due to overload. However, in order to ensure the operation of the system without considering the collection accuracy, so it does not meet our collection accuracy requirements.

5.2. Security mechanism

In what follows, we conduct security analysis of the adaptive security-related data collector based on the proposed security requirements in terms of confidentiality, integrity and availability. After data collection, we use symmetric encryption algorithms (such as AES) to encrypt data so as to avoid data leakage. When the data is encrypted, we digest the collected data through the hash algorithm (such as SHA-256). After that, we record each hash value of collected data in the file called “Hash_File”. Before processing the collected data, we first verify the integrity of the security-related data by comparing the hash value recorded in the Hash_File with the hash value of the collected data. After ensuring the integrity of the collected data, the data can be further processed and analyzed to ensure the accuracy of analysis results. We also need to ensure that the data storage medium of local system (such as a database) is safe. Furthermore, when collected data needs to be shared, the collector signs the data with a digital signature and applies access control mechanisms to prevent unauthorized access. Therefore, our proposed collector can meet the four security requirements presented in Section 3.

6. Conclusions

This paper presents the design and implementation of an adaptive security-related data collector in order to efficiently and adaptively collect security-related data in a large-scale heterogeneous network system. We apply SDDL and adaptive sampling algorithms to achieve adaptive security-related data collection based on network contexts. The adaptive security-related data collector can reduce the amount of collected data while ensuring the accuracy of data collection. The system performance evaluation based on a prototype implementation further shows the efficiency and effectiveness of our collector in terms of a number of pre-defined design requirements.

With regard to future work, we are going to further improve and extend the current implementation to make the data collector support self-learning capabilities combined with machine learning. Due to the

<table>
<thead>
<tr>
<th>Evaluation Criteria</th>
<th>Sampling algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Sampling times</strong></td>
<td>96</td>
</tr>
<tr>
<td>SAE</td>
<td>44243.884</td>
</tr>
<tr>
<td>MAE</td>
<td>9.029</td>
</tr>
<tr>
<td>SSE</td>
<td>4206916.374</td>
</tr>
<tr>
<td>MSE</td>
<td>858.554</td>
</tr>
</tbody>
</table>

Fig. 13. Collected data curve and real data curve in Experiment 2 (T = 50).
limitation of experimental conditions, we only evaluate the proposed collector in the contexts of the Internet and LTE. In the future, we plan to test the proposed collector in 5G core network devices and in mobile ad hoc networks. We also plan to integrate the adaptive security-related data collector and data analysis functions to implement a synthetic attack detection system.

Acknowledgements

This work is sponsored by the National Key Research and Development Program of China (grant 2016YFB0800704), the National Natural Science Foundation of China (grants 61672410, 61802293 and U1536202), the Academy of Finland (grant 308087), the Project Supported by Natural Science Basic Research Plan in Shaanxi Province of China (Program No. 2016ZDJC-06), the National Postdoctoral Program for Innovative Talents of China (grant BX20180238), the Project funded by China Postdoctoral Science Foundation (grant 2018M633461), and the China 111 project (grants B16037 and B08038).

References


Table 5
Results of experiment 3.

<table>
<thead>
<tr>
<th>Evaluation Criteria</th>
<th>Simple Random Sampling (Hu et al., 2006) (P = 1/100)</th>
<th>Systematic Sampling (Ariyapala et al., 2016) (T = 100)</th>
<th>Poisson Sampling (λ = 1/100)</th>
<th>ACFAS_PVR</th>
<th>ACFAS_PAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sampling times</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SAE</td>
<td>86818.736</td>
<td>50558.819</td>
<td>51849.106</td>
<td>37836.400</td>
<td>35396.227</td>
</tr>
<tr>
<td>MAE</td>
<td>17.574</td>
<td>10.216</td>
<td>11.034</td>
<td>7.625</td>
<td>7.103</td>
</tr>
<tr>
<td>SSE</td>
<td>8092972.809</td>
<td>6264628.955</td>
<td>5344381.416</td>
<td>3398695.965</td>
<td>2967174.725</td>
</tr>
<tr>
<td>MSE</td>
<td>1638.253</td>
<td>1137.344</td>
<td>684.944</td>
<td>595.460</td>
<td></td>
</tr>
</tbody>
</table>

Fig. 14. Collected data curve and real data curve in Experiment 3 (T = 100).


