Hindawi
Mobile Information Systems
Volume 2017, Article ID 1750637, 13 pages
https://doi.org/10.1155/2017/1750637

Research Article

An Automata Based Intrusion Detection Method for Internet of Things

Yulong Fu,[1] Zheng Yan,[2,3] Jin Cao,[1] Ousmane Koné,[4] and Xuefei Cao[1]

[1]*School of Cyber Engineering, Xidian University, Xian, China*
[2]*Aalto University, Espoo, Finland*
[3]*The State Key Lab of ISN, Xidian University, Xian, China*
[4]*University of Pau and Academy of Bordeaux, Mont-de-Marsan, France*

Correspondence should be addressed to Yulong Fu; ylfu@xidian.edu.cn

Received 25 January 2017; Revised 12 March 2017; Accepted 28 March 2017; Published 2 May 2017

Academic Editor: Jing Zhao

Internet of Things (IoT) transforms network communication to Machine-to-Machine (M2M) basis and provides open access and new services to citizens and companies. It extends the border of Internet and will be developed as one part of the future 5G networks. However, as the resources of IoT's front devices are constrained, many security mechanisms are hard to be implemented to protect the IoT networks. Intrusion detection system (IDS) is an efficient technique that can be used to detect the attackers when cryptography is broken, and it can be used to enforce the security of IoT networks. In this article, we analyzed the intrusion detection requirements of IoT networks and then proposed a uniform intrusion detection method for the vast heterogeneous IoT networks based on an automata model. The proposed method can detect and report the possible IoT attacks with three types: jam-attack, false-attack, and reply-attack automatically. We also design an experiment to verify the proposed IDS method and examine the attack of RADIUS application.

1. Introduction

Due to the rapidly advancing technologies of network communication, the Internet is going to connect everything from everywhere. New concept of Internet of Things (IoT) appears and is associated with the future Internet of 5G. IoT connects a large number of heterogeneous devices, such as "instance cameras," "wireless sensor network" (WSN), "smart meters," and "vehicles," while providing open access to a variety of data generated by such devices to provide new services to citizens and companies [1]. However, as the resources of IoT's front devices are constrained, many security mechanisms are hard to be implemented to protect the IoT networks. Some lightweight encryption methods are considered as the core technology to build the security mechanism of IoT [2], but considering the increments of the hacker's computation ability (the usage of Cloud Computing, Distributed Computing, Quantum computation, etc.), those lightweight cryptography methods are going to be crushed in the foreseeable future. Other kinds of security enforcement methods, such as intrusion detection system should be considered to protect the IoT networks [3].

Intrusion detection system (IDS) is an efficient technique to detect attackers when cryptography is broken [4]. It can detect malicious activities or policy violations by monitoring the network traffics or system actives [5]. IDS is normally a stand-by device or third-part software which will not inquire many changes to the current system. It is suitable for the resource constrained or inherited systems to protect their network security.

Many recent works have noticed the security problem of IoT system, and a number of intrusion detection methods are proposed and developed, such as [4, 6–10]. However, most of the proposed methods are still limited to data mining and can only give an intrusion view of WSN, MANET, Zigbee, or other subnets of IoT, and a uniform intrusion detection method for the whole IoT networks is rarely discussed. Meanwhile, as the network packets digging and statistic

feature training usually require many computation resources, such methods are hard to be implemented in some cases of IoT environments.

In this article, we present an automata based intrusion detection method for the networks of Internet of Things. Our method uses an extension of Labelled Transition Systems to propose a uniform description of IoT systems and can detect the intrusions of IoT networks. The used automata model can describe the combination of heterogeneous networks with terms and graphs, and the proposed IDS structure and algorithm can detect the intrusions by comparing the abstracted actions flows, which can solve the aforementioned problems.

Paper Contribution. By using automata theory, many complicated problems can be described and solved. In this article, we use an extension of Input Output Labelled Transition System to solve the uniform description problem of the heterogeneous IoT networks and propose a corresponding intrusion detection mechanism for IoT network. To achieve this purpose, a set of procedures including collected data grouping, packet data translation, anomaly data detection, and intrusion classification are designed and proposed. Comparing with the existing methods, the benefits of our work can be listed as below:

(1) To our knowledge, this is the first time of using automata theory to model and detect the intrusions of IoT networks. By using the proposed automata methods, we can map the IoT system to an abstract space, where a uniform security evaluation structure can be built.

(2) We defined and proposed a set of intrusion detection mechanisms by using the proposed automata method.

(3) We developed a GUI tools to automatically analyze and graphically present the abstract action flows and to detect the possible intrusions.

(4) We also analyzed and classified the detected intrusions, and three kinds of attacks, including replay-attack, jam-attack, and fake-attack, can be distinguished in our method.

The following sections are organized as below: In Section 2, the background, problem description, and related works of developing the IDS system over IoT are discussed. In Section 3, the entire approach of the automata based intrusion detection method will be described. In Section 4, to illustrate the use of the proposed IDS methods, we present an example of using the proposed method to analyze a simplified IoT system, and the results demonstrate the correctness of our method. And finally, in Section 5, we conclude this work and discuss some possible future works.

2. Background, Problems, and Related Works

2.1. Internet of Things and Its Security

2.1.1. Internet of Things. IoT is the network of things, with clear element identification, embedded with software intelligence, sensors, and ubiquitous connectivity to the Internet [11]. IoT enables things or objects to exchange information with the manufacturer, operator, and other connected devices utilizing the telecommunications infrastructure of the Internet. It allows physical objects to be sensed (by providing the specific information such as the RFID tags and QR code) and controlled remotely across the Internet. IoT will create opportunities for more direct integration between the physical world and computer-based systems, resulting in improved efficiency, accuracy, and economic benefit, for example, monitoring and controlling things by experts such as telemedicine and searching for things (keys, passports) directly that search engines do not provide today.

Normally, three basic elements should be included by an IoT system: the unique identity per thing (e.g., IP address), the ability to communicate between things (e.g., wireless communications), and the ability to sense specific information about the things (sensors) [11]. Therefore, for an IP based system, the IoT gateway is a good solution to form the IoT networks. The IEEE 802.15 Task Group 4 has defined the personal area network (PAN) coordinator to take in charge of the network domain. The PAN allocates local addresses and acts as a gateway to other domains or networks [12]. IEEE 802.15.4 also defined two types of IoT devices: the full-function device (FFD), which implements all of the functions of the communication stack and allows it to communicate with any other device in the network; and the reduced-function devices (RFDs) which are meant to be extremely simple devices with very modest resource and communication capabilities. Hence, RFDs can only communicate with FFDs and can never act as PAN coordinators.

2.1.2. IoT Security Attacks. Considering the specific features of IoT networks, we found that the following three kinds of attack scenarios likely happen in the real world and are important to be studied.

(i) Attack Scenario 1. For a given IoT network, such as the one presented in Figure 1, an authorized user, User1, may want to control the specific device in the IoT. The user needs to use the IoT networks to find the right device and to communicate with the device. For some security reason, the IoT device has to verify the authentication of User1. During this process, a cryptography method is normally needed to verify the authentication and to protect against the malicious attacks. However, a malicious user, User2, may be able to listen the communication between User1 and the corresponding IoT device. User2 may fake himself as User1 and create a replay-attack to the IoT system. To solve such problem, the RFD may ask FFD or PAN to help him to verify the authentication of the user and record the passed IDs of the user. A group authentication protocol and cryptography functions can help RFD to protect itself from such kind of attack. However,

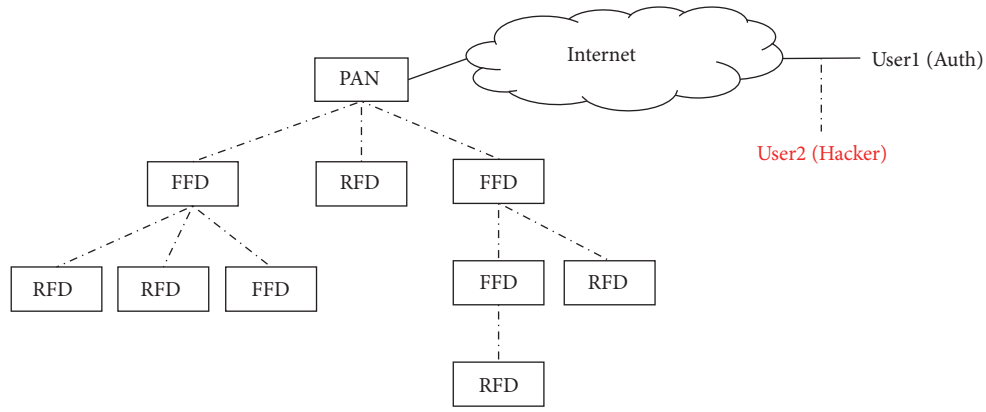

Figure 1: Attack scenario 1.

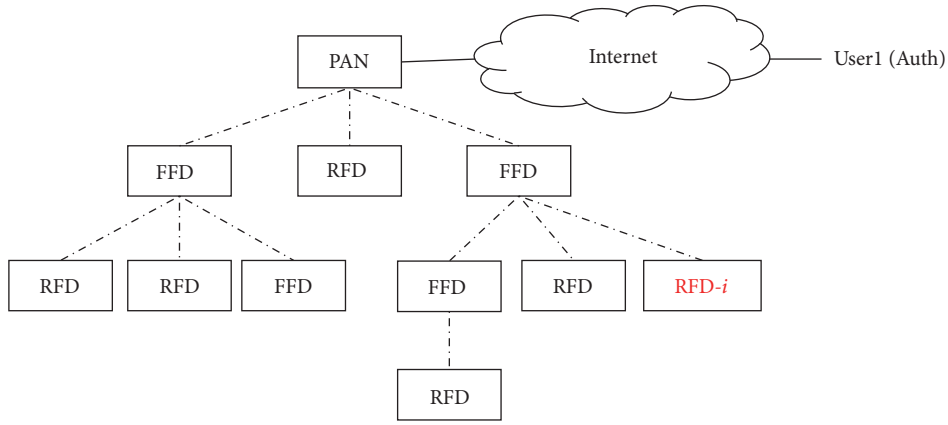

Figure 2: Attack scenario 2.

the FFD is also a resource constrained device, and the communication delay and calculation consuming will be too much for him to hold.

(ii) Attack Scenario 2. As most of the IoT networks are not closed, a malicious device may be able to present its willingness to join the IoT networks. For example, in Figure 2, a powerful device RFD-*i* (such devices can listen the communication channel of IoT devices), which is controlled by an attacker, may want to join the IoT network. Such powerful device can detect the communication information on the IoT networks and can execute many kinds of attacks such as DoS/DDoS to the corresponding FFD or PAN. Simply using the cryptography methods on IoT device will be hard to defense this kind of attacks.

(iii) Attack Scenario 3. Because the structure of IoT networks is dynamic, some authorized IoT device may be captured by the attacker. The attacker then can modify some functions or inject some virus and trojans to such device. Then the attacker can put such compromised devices to rejoint the IoT networks (see Figure 3). Because the device will be still recognized by the IoT system, it will pass the security verification of IoT network. This kind of attack is also difficult to be protected through the cryptography methods.

As we can see, by simply using the cryptography methods, some kinds of attack are hard to be detected in IoT networks. Although the usage of some complex security protocols may be able to achieve the security goals of IoT, they are hard to be implemented on the resource constrained IoT devices. Other ways of defensing the security of the system, such as the usage of intrusion detection system, should be considered for IoT network security.

2.2. Intrusion Detection System. The concept of intrusion detection was first proposed by Anderson in the year of 1980 [13] and is introduced to network system by Heberlein in 1990 [14]. After 2 decades of developing, the researches on IDS are becoming mature and have helped the industries to protect their system security for many years. An IDS may be either host or network-based [15]. A host based IDS analyzes events mainly related to OS information, while a network-based IDS analyzes network related events, such as traffic volume, IP addresses, and service ports. Meanwhile, according to the way of detecting the intrusion, two main categories of IDS are usually discussed: misuse IDS and anomaly IDS. The former uses the traces or templates of the known attacks, while the latter builds profiles of nonanomalous behaviors of computer system's active subjects. For example, IDIOT [16]

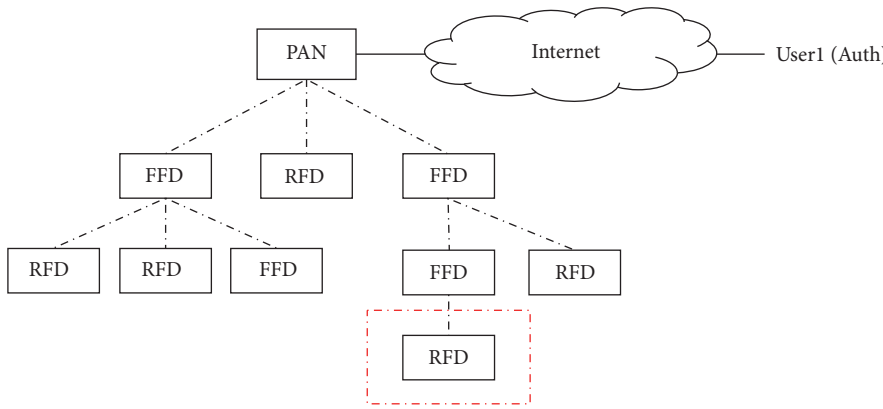

Figure 3: Attack scenario 3.

and STAT [17] use patterns of well-known attacks or weak spots in the system to match and identify known intrusions. The main advantage of misuse IDS is that it can accurately and efficiently detect instances of known attacks. The principal disadvantage is that it lacks the ability to detect the truly innovative attacks. On the other hand, anomaly IDS [18] does not require prior knowledge of intrusion and can thus detect new intrusions. But it may not be able to describe what the attack is and may have a high false positive rate.

An IDS normally contained four major components: Event Monitor, Event Database, Event Analyzer, and Response Unit [19]. The Event Monitor is responsible for detecting the system or environment actives and converts them as some specific formats and store them in the Event Database. The Event Analyzer retrieves the modeled actives from the Event Database and analyzes them in order to detect the intrusions. Once the unusual actives are detected, the Response Unit produces reports to a management station to warn a risk. IDS focuses on detecting and preventing the intrusive activities, which were not detected by conventional system security mechanisms. For some inherited systems, because of some historical or economic reasons, some powerful security mechanisms are hard to be deployed. However, the IDS can be used to solve this problem, because it needs nothing to change the target system.

2.3. Existing Intrusion Detection Works on IoT Networks. In recent years, along with the development of Internet of Things, Intelligent Hardware, and Virtual Reality, the intrusion detection method under IoT has become a trend in the development of information technology. However, the researches on such problem are still in its infancy. As IoT can be thought of as a vast heterogeneous network, most of the existing works began to study the components of IoT to find a suitable intrusion detection method. In [1], based on the use of Game Theory, Sedjelmaci et al. proposed a hybrid intrusion detection method, which mixed the usage of signature and anomaly ways for IoT intrusion detection. By creating the game model of intruder and normal user, the Nash Equilibrium Value was calculated and was used to decide when to use the intrusion detection method of anomaly. In [20], J. Chen and C. Chen proposed a real-time pattern matching system for IoT devices by using the Complex Event Processing (CEP). The advantage of this method is that it uses the features of the events flows to judge the intrusions, which can reduce the false alarm rate comparing with the traditional intrusion detection methods. Although this method will increase the consumption of system computing resources, it can obviously reduce the feedback delay of the IDS system. In [7], Nadeem and Howarth summarized the intrusion detection methods for MANET, which is one kind of network structure of the IoT. By analyzing and comparing the attack methods and detection algorithms of MANET, this paper analyzes the existing CRADS, GIDP, and other intrusion detection frameworks for MANET.

Although these existing methods can solve the intrusion detection problems of IoT from different levels, a uniform intrusion detection method is still needed to give an entire intrusion view of the IoT networks. As what have been pointed by Gendreau and Moorman in their survey of [10], the research of intrusion detection system for IoT system should focus on solving the problems of "lacking complete interoperability between different IoT parts."

3. An Automata Based Intrusion Detection Approach for IoT Security

In order to give a complete intrusion view for the different cases of IoT networks, a uniform intrusion detection method is required. In this article, by using the proposed automata model, we can project the different cases of IoT to an abstract algebra space, where a uniform security evaluation structure can be built. Meanwhile, in the real word of IoT system, by adopting a data collector and analyzing the transmitting packets, the real-time actions flows of the IoT networks can be achieved and translated into the formal format of automata. Then by comparing the real-time action flows with the anomaly or standard libraries, we can detect the intrusions of IoT quickly and solve the aforementioned problems.

3.1. The Automata Model. A finite automata (or finite state machine) [21] can present the network system with a finite

number of states and transitions, where the states represent the current status of the device and the transitions represent the active actions between different states. The current state changes only if it receives the corresponding actions. An Input/Output Labelled Transition System (IOLTS) [22] is a special case of automata, which emphasizes the input and output interactions of the system. An IOLTS system can be presented as a 4-tuple algebra set $\langle S, L, T, s_0\rangle$, where S represents a countable, nonempty set of states; L represents a countable set of labels; T represents the set of transition relations, $T \subseteq S \times (L \cup \{\tau\}) \times S$ (here, τ represents an internal action of the system that will not be achieved from outside); and s_0 is the initial state. Notice that L contains two subsets: input label L_I and output label L_O ($L_I \cap L_O = \emptyset$, $L_I \cup L_O = L$). If $s \in S$, then we denote $\mathrm{In}(s)$ and $\mathrm{Out}(s)$ to represent the set of input and output labels of state s. A transition is denoted as $s_i \xrightarrow{!l} s_j$, where $s_i, s_j \in S$ and $l \in L$. The symbol ! or ? representing l is an output label or input label, respectively. IOLTS can be used to describe an interactive system and can present the system with a graphic view. However, as the IoT networks contain multiple components, an extension of IOLTS, the Glued-IOLTS [23], is needed to present the networked system.

In a Glued-IOLTS, in order to describe the communication medium between different components, a normal state $s \in S$ of IOTS(L) is defined as the following two levels:

(i) higher_level state s_i_u, which connects to the environment or other states of the same component;

(ii) lower_level state s_i_l, which connects to the states of other components.

And then, the communication medium can be defined by such transition, which begins from the lower_level state of one component and ends with the lower_level of initial state of another component. If we use S_i and L_i to denote the states and labels in IOTS(L_i) and S_j and L_j to denote the state and labels in IOTS(L_j), then if $\exists !l \in L_i$, $\exists s_i \in S_i$, $!l \in \mathrm{Out}(s_i)$, and $\exists s_j \in S_j$, $?l \in L_j$, $?l \in \mathrm{In}(s_j)$. The transition of the common medium between IOTS(L_i) and IOTS(L_j) is presented as $s_i\_l \xrightarrow{!l} s_0\_l$. We use S_{medium} and T_{medium} to denote the states and transitions in the medium, and we give the definition of Glued-IOLTS as below.

Definition 1 (Glued-IOLTS). A Glued-IOLTS represents a set of IOLTS $\langle S_i, L_i, T_i, s_{i0}\rangle$ ($i = 1, \ldots, n$) and a medium M, which is still a 4-tuple system $\langle S_{\mathrm{glu}}, L_{\mathrm{glu}}, T_{\mathrm{glu}}, s_{\mathrm{glu}_0}\rangle$, where

(i) $S_{\mathrm{glu}} = \langle S_1 \cup S_2 \cup \cdots \cup S_n \cup S_M\rangle$,

(ii) $L_{\mathrm{glu}} = \langle L_1 \cup L_2 \cup \cdots \cup L_n\rangle$,

(iii) $s_{\mathrm{glu}_0} = \langle s_1\_0, s_2\_0, \ldots, s_n\_0\rangle$ is the initial state,

(iv) $T_{\mathrm{glu}} \subset S_{\mathrm{glu}} \times L_{\mathrm{glu}} \times S_{\mathrm{glu}}$,

$$T_{\mathrm{glu}} = \Big\{\big(s_1, s_2, \ldots, s_i, \ldots, s_m\big) \xrightarrow{\alpha} \big(s_1, s_2, \ldots, s_i', \ldots, s_m\big) \mid \big(s_i, \alpha, s_i'\big) \in T_i \cup T_M\Big\}, \quad (1)$$

$$T_M = \Big\{\big(s_{il}, \mu, s_{jl}\big) \mid i \neq j, \mu \in \mathrm{Out}\big(s_{il}\big) \cap \mathrm{In}\big(s_{jl}\big)\Big\}.$$

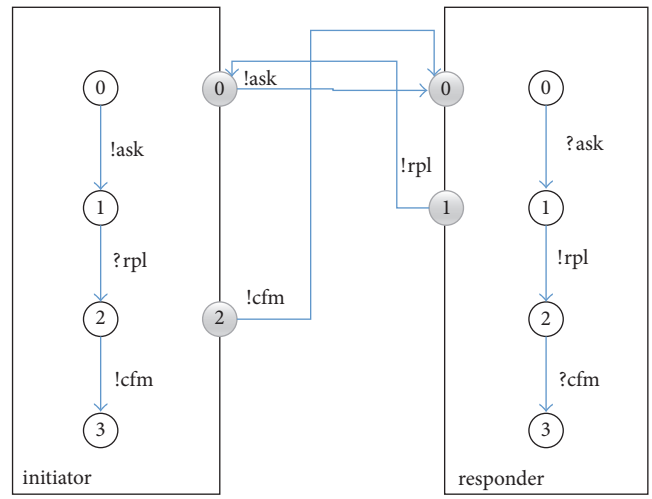

Figure 4: Glued-IOLTS of NSPK.

Example 2. The Needham-Shroeder Public Key (NSPK) protocol [24] is an asymmetric cryptography based authentication protocol, which defines the handshakes between two participations: the initiator i and the responder r. The brief protocol narrations can be presented with the three-message exchanging as below:

Msg 1 (Ask). $i \rightarrow r$: $\{n_i, i\}_{\mathrm{pk}_r}$

Msg 2 (Rpl). $r \rightarrow i$: $\{n_i, n_r\}_{\mathrm{pk}_i}$

Msg 3 (Cfm). $i \rightarrow r$: $\{n_r\}_{\mathrm{pk}_r}$

A networked security system implementing the NSPK protocol can be described and modeled with the Glued-IOLTS, and the result is presented in Figure 4.

3.2. Intrusion Detection Approaches of IoT Networks. Although the proposed automata model can be used to describe the communications of an IoT system and can make the comparison of different subnets of IoT become possible, to adopt this model into an intrusion detection system, a set of cooperated devices and some existing approaches are also needed. Just like the general IDS system, the proposed automata based IDS of IoT networks also consist of four major components: Event Monitor, Event Database, Event Analyzer, and Response Unit. A general view of the proposed IDS can be presented in Figure 5. In this article, although the four components are developed in our system, our description will mainly focus on the Event Analyzer and Response Unit.

3.2.1. Event Monitor. For the purpose of collecting the data traffics through the IoT network, a network collector (the component labelled with C in Figure 5) should be implemented on the PAN coordinator or other IoT gateways to monitor the network traffic. Such collector will be embedded software or hardware to obtain the received and sent packets through the network device. The collector needs to record the transmitting data into digital files and send the files to the IDS Event Analyzer.

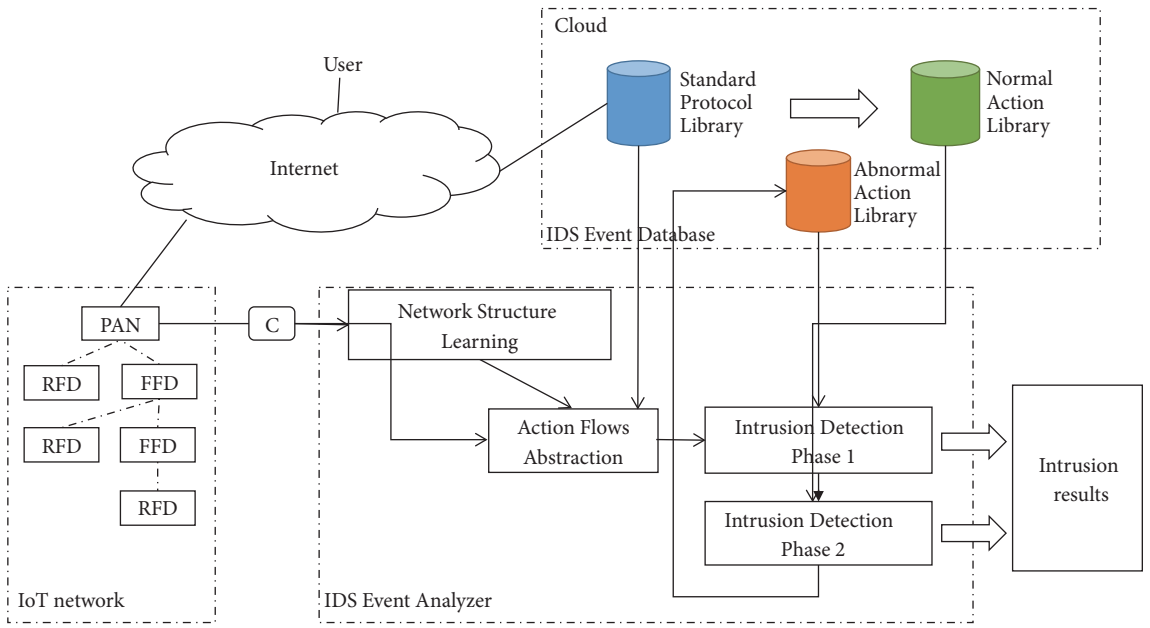

Figure 5: IDS structure.

3.2.2. Event Database. In our method, the network event is described as the abstract action flows, and such network actions are described with transitions of the proposed Glued-IOLTS model. Three databases should be implemented in our IDS: Standard Protocol Library, Abnormal Action Library, and Normal Action Libraries are required. The Standard Protocol Libraries store the description of the standard protocols through Glued-IOLTS. The Normal Action Libraries store the possible action flows which are created from the Standard Protocol Libraries. The Abnormal Action Libraries store the recognized anomaly actions flows for the system. These three databases should be stored on the cloud and can be visited directly by the Event Analyzer.

3.2.3. Event Analyzer. The IDS Event Analyzer is an important part of our IDS system. It contains three basic models: Network Structure Learning Model, Action Flows Abstraction Model, and Intrusion Detection Model.

(i) Network Structure Learning Model. In our method, the collected packet data should be sent to this model first to make the IDS system get a general view of the network topologies. As the IoT devices can be distinguished with the unique ID, by analyzing the collected information of the data packets, such as the source IP, destination IP, port number, timestamp, and protocol type, we can distinguish the IoT devices from the others. For example, because the IoT devices are usually connected to the same IoT gateway, the first three fields of the IPv4 address of such devices will be the same. In this case, by counting the frequency of each IPv4 field, we can achieve the IP segment of the IoT devices. These unique IDs of the IoT devices will be recorded and sent to the Action Flows Abstraction Model.

(ii) Action Flows Abstraction. The collected real-time packets from IoT also need to be sent to the Action Flows Abstraction Model. Through this model, the packets will be allocated according to the device belonging, session ID, timestamps, and protocol types which are recognized through the aids of Network Structure Learning Model and the Standard Protocol Library. Through the information detected, the network traffics can be classified into message sequences. However, if the IoT serves multiple customers, different sessions may happen in parallel, which may make the messages become hard to be distinguished. In this article, we assume that the network connections from different services happen sequently; then by using one selected window size N, by comparing the other detected information, such as IP address, protocol type, and info (see Figure 6), we can allocate the packets to be the message sequence. The selected window size N relates to the efficiency of the Event Analyzer. The greater the value of N is selected, the more accurate the sequence detection is. But at the same time, it also means more memory and computing times consuming. We suggest N should be considered bigger than the amount of messages which happened during one session of the protocol specification and less than the whole detected messages space of the Event Monitor.

After we can allocate the packets to be message, we need to translate these messages to abstract action flows. To do this, the help from the Standard Protocol Library is needed. From the results of the message allocation, together with the protocol type information of each packet, we can know the main protocol type of such selected message. Then after we get the protocol type of the selected message, we can search for the basic formal action primitives from the Standard Protocol Library. And by comparing with the Info information of each packet, we can represent the packets

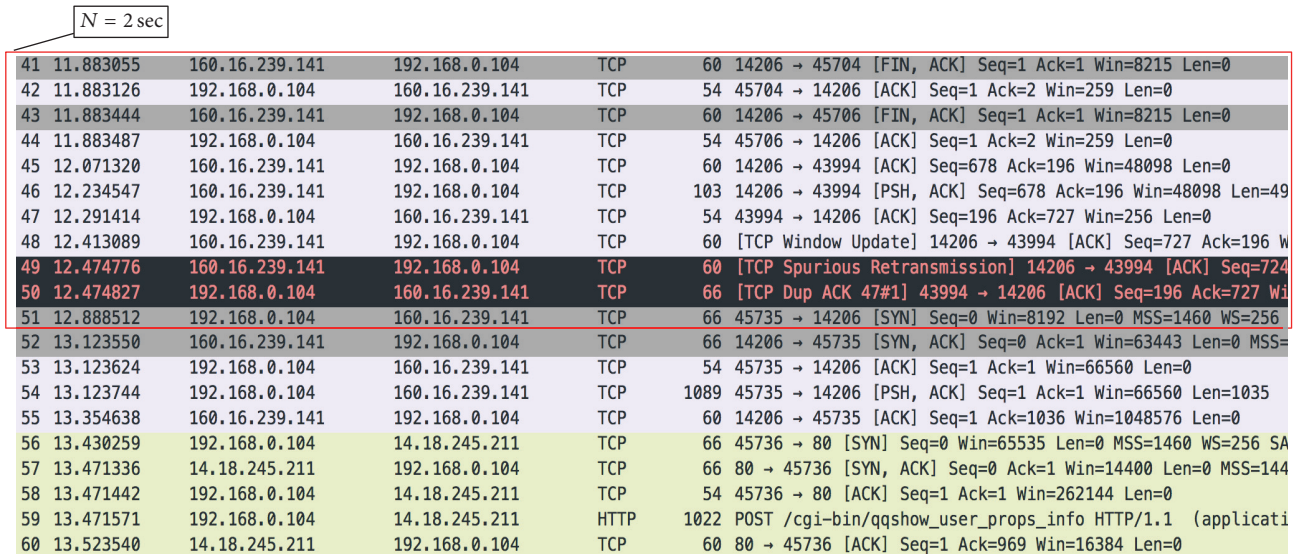

Figure 6: Example of selecting $N = 2$ sec.

to be the automata primitives. Then the abstracted action sequences can be achieved. For example, the selected message in Figure 7 can be translated as [?FIN, !ACK, ?ACK + FIN, !ACK, ?ACK, ?PSH, !ACK, ?UPDATE, !SYN] through the processes presented in Figure 7.

(iii) Intrusion Detection. The result of the Action Flows Abstraction Model will be the list of automata transition sequence of the target system. Such transition sequences are then taken as the input to the intrusion verification part. In our method, we have two phases of intrusion verification.

Intrusion Detection Phase 1. The results of Action Flows Abstraction Model are used to be checked with an Abnormal Action Library, which is stored in the Event Databases. This library is a predefined database that is stored on the cloud next to the IoT system (Fog Computing [11]). If the transition sequence matches with the one stored in the Abnormal Action Library, we remark such message as an intrusion and output it as the result of the intrusion detection system. If the input sequence does not match any stored sequences in the Abnormal Action Library, the action flows go to the second phase of the intrusion detection.

Intrusion Detection Phase 2. In the second phase of intrusion, an anomaly detection method will be used to check the intrusion. In this phase, a Normal Action Library will be used to check whether the input transition sequence is a normal one. The Normal Action Library is generated from the Standard Protocol Library, by using the techniques of Fuzzing [25] and Robustness Testing [26]. If the comparing results show that the input sequence is abnormal, we take such message as a suspected one and ask for a manual verification from the experts to avoid the false positive. If the suspected transition sequence is confirmed as intrusion by the experts, we then record such message into the Abnormal Action Library and use it for the next time of intrusion detection. The method of verifying transition sequences in the Normal Action Library is to find the *walk* in the Glued-IOLTS graph of the library. During the verification process, we may need to adapt some past transitions into the detected sequence to complete the walk in Glued-IOLTS; for the detailed algorithm, please check [27]. After doing this, if the transition sequence can find the corresponding walk, it means the detected messages traffics are normal messages. Otherwise, message traffic contains some possible attacks to the system.

3.2.4. Response Unit. The Response Unit produces reports to a management station to warn an intrusion risk to the IoT networks. In the report, the following three types of attacks are going to be classified, which correspond to the attack scenarios presented in Section 2.

(i) Replay-attack: this attack corresponds to the aforementioned attack scenario 1. In this kind of attack scenario, the attacker can listen the communication between an authenticated user and the IoT device; then the attacker uses the transition which happened to attack the system. This kind of attacks can be distinguished by our IDS because the corresponded transition sequence can not be found in the normal library. The *walk* will stop at an inopportune transition, and also this transition can be found in the past transitions.

(ii) Jam-attack: this attack corresponds to the aforementioned attack scenario 2. In this kind of attack, the powerful attacker can detect the communication information on the IoT networks and can execute attacks such as DoS/DDoS to the corresponding FFD or PAN to block the communication channel. In this case, on our IDS system, after translating the collected messages into automata transition sequences, the

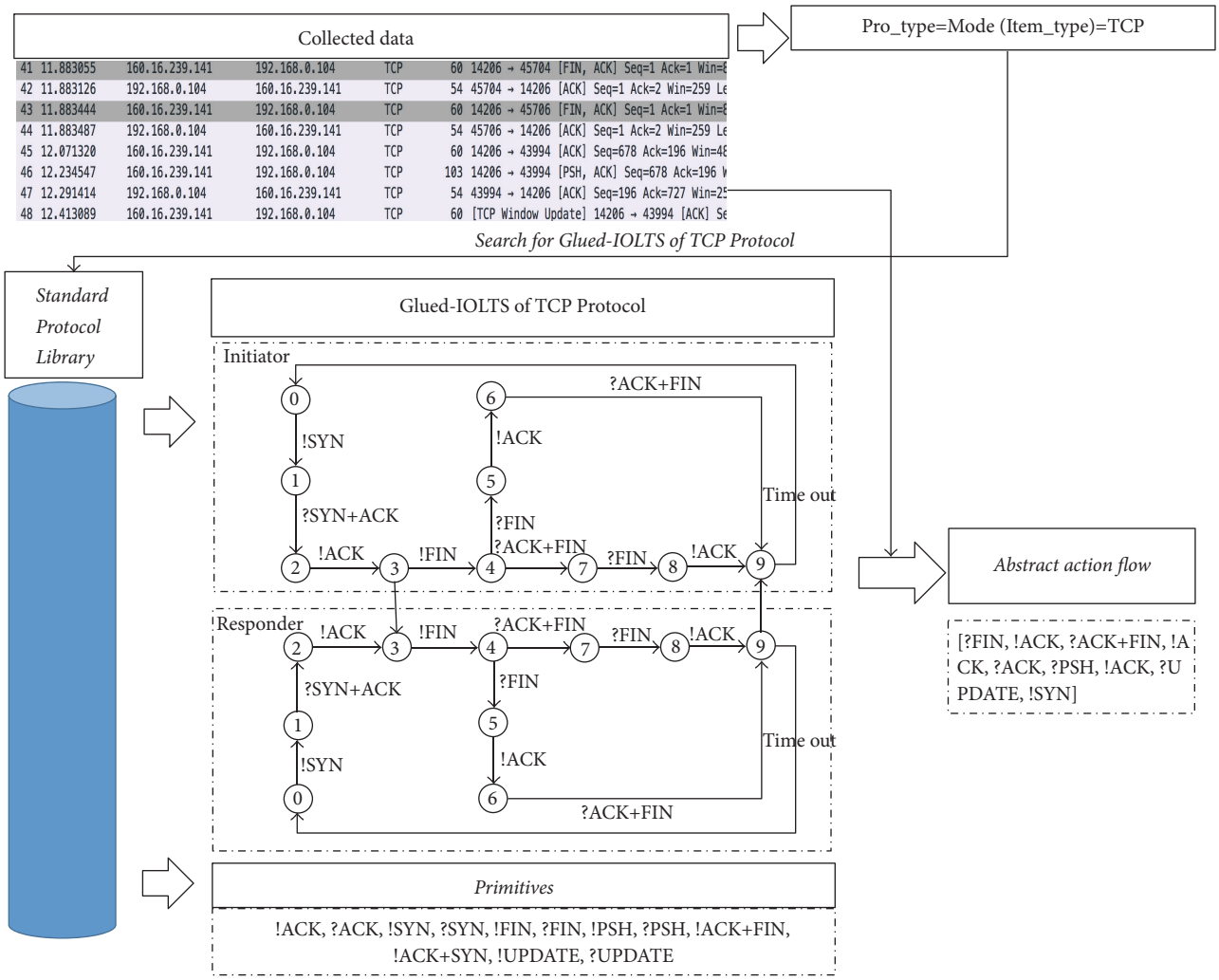

Figure 7: Example of translating abstract action flow.

corresponding *walk* can be found in the Glued-IOLTS graph, but the end state of this walk will not be the end state of the transition machine. It is a partial sequence of Glued-IOLTS.

(iii) Fake-attack: this attack corresponds to the aforementioned attack scenario 3. In this kind of attack, the compromised IoT devices may modify the transmitting message and inject some malicious codes to the message and send it to the receiver. This kind of attack may contain many strategies of modification, but here, we only consider the modifications which causes the changes on the automata primitives (the model transition label will change). If a sequence contains the fake-attack, the verification cannot find the corresponding *walk* in the Glued-IOLTS. But the fake actions may happen at the transition which makes the walk stopped or may happen before.

In order to detect those attacks automatically, we propose an algorithm in Algorithm 1. The inputs to the algorithm are one of the modeled label sequences (l_{ids}) which is detected by the IDS monitors and the glued transition system (T_{sys}). First of all, the algorithm searches for the transitions in T_{sys}, which have the same label as the first label of l_{ids} and record the results in a transition list of t_temp. Then for each transition t_i in t_temp, the algorithm compares the label of the next transition of t_i and the next label of l_{ids}. Remove t_i from t_temp. If the transition with the same label can be found, record it in t_temp. Backup this t_temp as t_temp_bac. Repeat the process until the end of l_{ids} or the t_temp is empty. During the loop, the algorithm records the past labels of l_{ids} in l_{pass}. The algorithm will stop if it checks all of the items in l_{ids} or T_{sys}. When it stops, if it found all labels of l_{ids} in T_{sys}, we go to check the final state of the walk in T_{sys}. If the finial state is an "end" state, l_{ids} is secure. Otherwise, l_{ids} contains jam-attack. If the algorithm stops when comparing l_n of l_{ids} with result of the t_temp being empty, then for each transition t_j in t_temp_bac, compare the label of the next transition of t_j and the passed label l_i in l_{pass}. If l_i is the same as the label of the next transition of t_j, record the next transition of t_j in t_temp, backup t_temp to t_temp_bac, record l_i in l_{pass}. Then, compare l_n with the next transitions of t_temp.

```
Input:
Label Array l_ids; //one transition sequence detected by IDS.
Transition Array T_sys; //the transition system of the protocol.
Output:
secure, fake-attack, jam-attack, replay-attack
Begin
Transition Array t_temp;
Transition Array t_next;
Label Array l_pass;
String result;
int flag=0; Search l_ids[0] in T_sys and record the results in t_temp;
For each transition t_i in t_temp{
   record the next transition of t_i in t_next;
   record l_ids[0] in l_pass;
  }For (int i = 1; i < l_ids.length; i++){
   flag++;
   If (t_tempisnotempty){
      record the next transition of t_i in t_next;
      t_temp_bac=t_temp;
      remove t_i from t_temp;
      Search l_ids[i] in t_next and record the results in t_temp;
      record l_ids[i] in l_pass;     }else{
      For each l_k in l_pass{
         Search l_k in t_next and record the results in t_temp;
         If (t_tempisnotempty){
            continue;
         }
      }
      If (l_ids[i] in l_pass){
         result="replay-attack";
         return result;
      }
      else{
         result="fake-attack";
         return result;
       }
   }   }
If(flag==l_ids.length){
   If(t_i.nexState().getStatus.equals("end")){
      result="secure";
      return result;
   }else{
      result="jam-attack";
      return result;
      result="secure";
   }
}
End
```

Algorithm 1: Algorithm for intrusion detection.

If l_n can be found in the next transition, record l_n in l_{pass} and move to the next label of l_{ids}. Otherwise, reconsider the passed labels until the end of l_{pass}. If after considering the labels of l_{pass}, l_n still cannot be found in the transition sequence, then l_{ids} must contain some modifications. The algorithm returns "fake-attack." Meanwhile, if l_{pass} contains l_n, then l_{ids} contains a replay, and the algorithm returns "replay-attack."

4. An Experiment over a Tested IoT System

In order to verify the proposed intrusion detection method, we design a IoT experiment environment like Figure 8. In the tested environment, we use two *Raspberry Pi 3* as the reduced-function device, an *Android Phone* (HUAWEI Mate 9) as a full-function device, and a wireless router

```
type:RADIUS
source:c0 a8 01 84
dest:c0 a8 01 0a
time:16:16:09
data:01 00 00 14 74 68 69 73 20 69 73 20 63 6c 69 65 6e 74 20 31
category:send
type:RADIUS
source:c0 a8 01 0a
dest:c0 a8 01 84
time:16:16:12
data:0b 00 00 3c 4e 61 73 74 6f 63 6c 69 65 6e 74 63 68 61 6c 6c 12 1e
69 6e 70 75 74 20 75 73 65 72 6e 61 6d 65 20 61 6e 64 20 70 61 73 73
77 61 72 64 73 18 0a 33 32 37 36 39 34 33 30
category:receive
type:RADIUS
source:c0 a8 01 84
dest:c0 a8 01 0a
time:16:17:12
data:01 00 00 3a 74 68 69 73 20 69 73 20 63 6c 69 65 6e 74 20 31 01 08 79
75 6c 6f 6e 67 02 12 0d be 70 8d 93 d4 13 ce 31 96 e4 3f 78 2a 0a ee 04
06 c0 a8 01 84 05 06 00 00 12 0c
category:send
...
```

Box 1: An example of IDS1 records traffics.

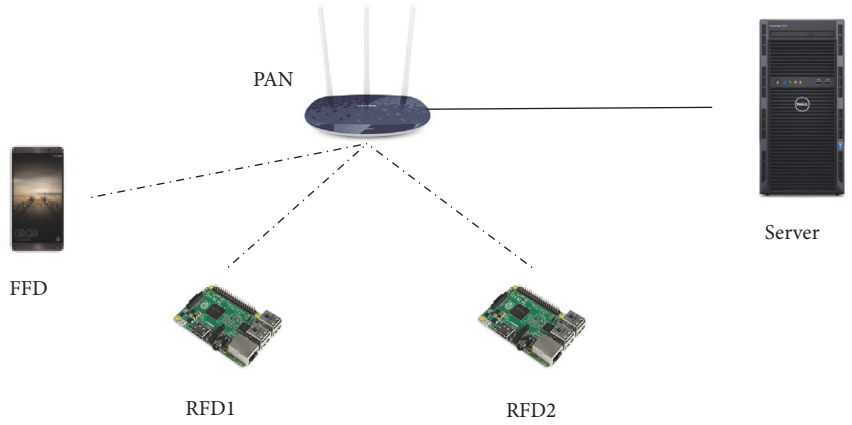

Figure 8: Experiment IoT networks.

(OpenWrt router) to be the IoT gateway (PAN coordinator). The router is connected with a server, and on the server, we use MySQL to build three database tables: Standard_Protocol, Abnormal_table, and Normal_table, which are corresponding to the three databases in our IDS methods. We use port mirroring on the router (a plug-in is needed to be installed on the OpenWrT router) and mirror the packets of WAN to the connected server. We install Wireshark [28] on the server side to collect and analyze the forwarded transmitting packets from IoT gateway. In our experiment, the RADIUS applications are taken as the services executed on the tested IoT networks [29]. The RADIUS protocol is an application layer protocol, which transmits data through UDP traffics. It uses the port number 1812 or 1645 to communicate. So when the monitor (Wireshark) obtains the IP traffics, by checking the port number of the UDP messages, the RADIUS messages can be distinguished.

For the simplicity of the experiment, we make the FFDs and RFDs only execute the RADIUS applications: we install the FreeRADIUS [30] on the server and the RADIUS client (NTRadPing [31]) on the client side (RFD1, RFD2 and FFD) to construct an experiment environment. We take the FFD device as an attacker and send the RADIUS requests as we need. Because the IoT gateway mirrored all of the WAN ports packets to the server, the Wireshark can record the sent/received data of each of the IoT devices, analyze them, and restore them. For better understanding, we select several packets and write them as the format of Box 1.

The IDS Event Analyzer in this experiment is an application we developed with Java. It can concatenate

Wc1	Wc2	Wc3	Wc4	Wc5	Lc1	Lc2	Lc3	R1	S1
xxxx !Ac_req_w1									
								?Ac_req_w1 !Ac_req_n_w1	
									?Ac_req_w1 !Ac_req_n_w1
								?Ac_accept_n_w1 !Ac_accept_w1	
?Ac_accept_w1 xxxx									
	xxxx !Ac_req_w2								
		?Ac_req_w2 !Ac_req_w2							
								?Ac_req_w2 !Ac_req_n_w2	
									?Ac_req_n_w2 !Ac_accept_n_w2
								?Ac_accept_n_w2 !Ac_accept_w2	
	?Ac_accept_w2 xxxx								
					xxxx !Ac_req_n				
								?Ac_req_l1 !Ac_req_n_l1	
				.....					

Figure 9: Message concatenation.

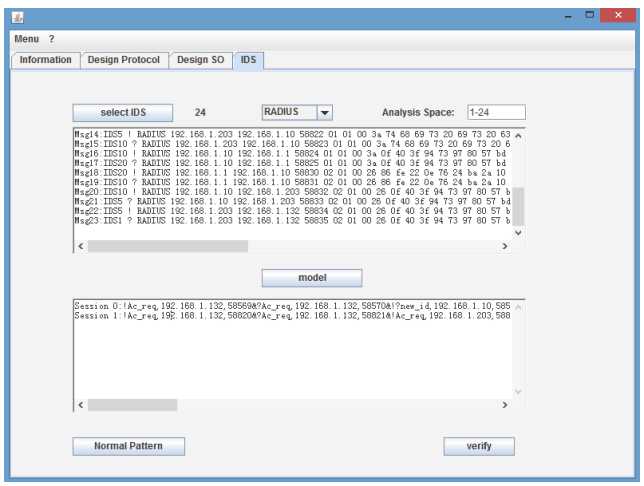

Figure 10: GUI of IDS.

the IDS detected messages as sequences, model those message sequences, and implement our algorithm to detect the possible intrusion (see Figure 10). As the network traffics happen sequently, the detected traffic data from different IoT devices may happen as Figure 9, where Wc1, Wc2, and Wc3 represented the RFD1, RFD2, and FFD of Figure 9, respectively. R1 represents the router, and S1 represents the server. For example, we choose a window size of 1 sec and found three modeled message sequences: {xxxx, !Ac_req_w1, ?Ac_req_w1, !Ac_req_w1_n, ?Ac_req_n_w1, !Ac_accept_n_w1, ?Ac_accept_n_w1, !Ac_accept_w1, ?Ac_accept_w1, xxxx}, {xxxx, !Ac_req_w2, ?Ac_req_w2, !Ac_req_w2, ?Ac_req_w2, !Ac_req_n_w2, ?Ac_accept_n_w2, !Ac_accept_w2, ?Ac_accept_w2, xxxx}, and {xxxx, !Ac_req_l1}. In this case, the first transition sequence is a normal connection sent from the client Wc1 to the server. The second sequence is a connection from Wc2 to Wc3 (this is maybe because the Wc3 declares himself as a NAS server); then Wc3 forwards the request of Wc2 to the real server. This sequence contains a replay-attack. And the third sequence is not a complete sequence. If the IDS only verifies the signature of the message, it will not find the problem of the second transition sequence. In our IDS approach, we only need to search this transition trace in the corresponding reachable graph, which is a nonanomalous profile of the target system.

The proposed Java tools will visit the Standard_Protocol table (the Standard Protocol Library) on MySQL database, and the nonanomalous profile of RADIUS protocol can be presented as the Glued-IOLTS of Figure 11. In this selected experiment, the verified traffics contain two RADIUS sessions and after the "message concatenation and classification," two different message sequences are obtained (they are listed in the bottom-left of Figure 11). Then through

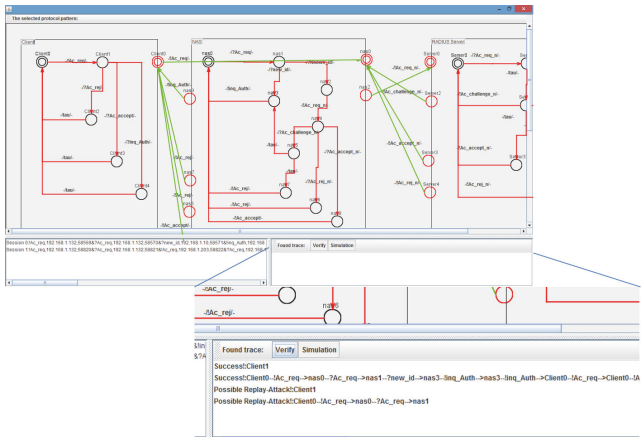

Figure 11: IDS verification panel.

the algorithm proposed, the program can verify the detected traffics automatically. The verification results of each detected sequence are presented in the bottom-right of Figure 11 (which identified that the first sequence is normal and the second sequence contains "replay-attack," and an alarm will be triggered when verifying the second message traffics).

5. Advances of the Proposed Method

The proposed intrusion detection method uses automata transitions to describe the network traffic flows and can map the different subnets of IoT to the same algebra space. In this case, different types of IoT, such as WSN, MANET, and Zigbee, can be described and compared with the same IDS method. Meanwhile, the way of using transition and graphic also makes the Standard Library, Anomaly Action Library, and Normal Action Library become easy to be implemented. However, because, in the process of finding abnormal action flows, the algorithm we used is a state based algorithm, which may cause the "state space explosion" problem, the complicity of the analyzed system should not be too much high. In fact, as the IoT devices are resources contained, the complexity of the IoT system is normally simple, and our IDS methods will be fine for the IoT intrusion detection.

6. Conclusion

Internet of Things is an important part of the future 5G, and the security of IoT will relate to many important scenarios of the future 5G and has become the core requirement of the network development. However, as the resources of IoT devices are constrained, many security mechanisms are hard to be implemented to protect the security of IoT networks. In this article, based on the automata theory, we proposed a uniform intrusion detection method for the vast heterogeneous IoT networks. Our method uses an extension of Labelled Transition Systems to propose a uniform description of IoT systems and can detect the intrusions by comparing the abstracted actions flows. We designed the intrusion detection approach, built the Event Databases, and implemented the Event Analyzer to achieve the IDS approaches. The result of the proposed IDS detects three types of IoT attacks: jam-attack, false-attack, and reply-attack. We also design an experiment environment to verify the proposed IDS method and examine the attack of RADIUS application in this article.

For the future work, we plan to continue enrich date types in our Standard Protocol Library and to improve the fuzzy method to make the creating of Normal Action Library become more efficient and accurate. Another line of our future research is to develop the suitable method to describe and evaluate the contents of the translating packets.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is sponsored by the National Key R&D Program of China (Grant 2016YFB0800700), the NSFC (Grants 61602359 and 61402354), the China Postdoctoral Science Foundation Funded Project (no. 2015M582618), the 111 project (Grant B16037), and the Fundamental Research Funds for the Central Universities (JB150115 and JB161508).

References

[1] H. Sedjelmaci, S. M. Senouci, and M. Al-Bahri, "A lightweight anomaly detection technique for low-resource IoT devices: a game-theoretic methodology," in *Proceedings of the IEEE International Conference on Communications (ICC '16)*, pp. 1–6, IEEE, Kuala Lumpur, Malaysia, May 2016.

[2] N. Boggs, W. Wang, S. Mathur, B. Coskun, and C. Pincock, "Discovery of emergent malicious campaigns in cellular networks," in *Proceedings of the 29th Annual Computer Security Applications Conference (ACSAC '13)*, pp. 29–38, New Orleans, La, USA, December 2013.

[3] C. X. Wang, X. Gao, X. You et al., "Cellular architecture and key technologies for 5g wireless communication networks," *IEEE Communications Magazine*, vol. 5, no. 2, pp. 122–130, 2014.

[4] B. Arrington, L. Barnett, R. Rufus, and A. Esterline, "Behavioral modeling intrusion detection system (BMIDS) using internet of things (IoT) behavior-based anomaly detection via immunity-inspired algorithms," in *Proceedings of the 25th International Conference on Computer Communication and Networks (ICCCN '16)*, pp. 1–6, Waikoloa, Hawaii, USA, August 2016.

[5] A. R. Baker and J. Esler, *Snort Intrusion Detection and Prevention Toolkit*, Andrew Williams, Norwich, NY, USA, 1st edition, 2007.

[6] C. Liu, J. Yang, Y. Zhang, R. Chen, and J. Zeng, "Research on immunitybased intrusion detection technology for the internet of things," in *Proceedings of the 7th International Conference on Natural Computation (ICNC '11)*, Shanghai, China, 2011.

[7] A. Nadeem and M. P. Howarth, "A survey of manet intrusion detection & prevention approaches for network layer attacks," *IEEE Communications Surveys and Tutorials*, vol. 15, no. 4, pp. 2027–2045, 2013.

[8] Z. Yan, R. Kantola, G. Shi, and P. Zhang, "Unwanted content control via trust management in pervasive social networking," in *Proceedings of the 12th IEEE International Conference on*

Trust, Security and Privacy in Computing and Communications (TrustCom '13), pp. 202–209, Melbourne, Australia, July 2013.

[9] C. Modi, D. Patel, B. Borisaniya, H. Patel, A. Patel, and M. Rajarajan, "A survey of intrusion detection techniques in cloud," *Journal of Network and Computer Applications*, vol. 36, no. 1, pp. 42–57, 2013.

[10] A. A. Gendreau and M. Moorman, "Survey of intrusion detection systems towards an end to end secure internet of things," in *Proceedings of the IEEE 4th International Conference on Future Internet of Things and Cloud (FiCloud '16)*, pp. 84–90, Vienna, Austria, August 2016.

[11] A. Rayes and S. Samer, *Internet of Things—From Hype to Reality*, Springer International Publishing, Cham, Switzerland, 2017.

[12] Z. Hanzálek and P. Jurčík, "Energy efficient scheduling for cluster-tree wireless sensor networks with time-bounded data flows: application to IEEE 802.15.4/ZigBee," *IEEE Transactions on Industrial Informatics*, vol. 6, no. 3, pp. 438–450, 2010.

[13] J. P. Anderson, "Computer security threat monitoring and surveillance," Tech. Rep., 1980.

[14] L. T. Heberlein, "A network security monitor," in *Proceedings of the IEEE Computer Society Symposium, Research in Security and Privacy*, pp. 296–303, Oakland, Calif, USA, 1990.

[15] P. García-Teodoro, J. Díaz-Verdejo, G. Maciá-Fernández, and E. Vázquez, "Anomaly-based network intrusion detection: techniques, systems and challenges," *Computers and Security*, vol. 28, no. 1-2, pp. 18–28, 2009.

[16] S. Kumar and E. H. Spafford, "A software architecture to support misuse intrusion detection," in *Proceedings of the 18th National Information Security Conference*, pp. 194–204, Baltimore, Md, USA, October 1995.

[17] K. Ilgun, R. A. Kemmerer, and P. A. Porras, "State transition analysis: a rule-based intrusion detection approach," *IEEE Transactions on Software Engineering*, vol. 21, no. 3, pp. 181–199, 1995.

[18] T. Lunt, A. Tamaru, F. Gilham et al., "A real-time intrusion detection expert system (ides)-final technical report," Technical Report, Computer Science Laboratory, SRI International, Menlo Park, Calif, USA, 1992.

[19] S. Staniford-Chen, B. Tung, P. Porras et al., "The common intrusion detection framework-data formats," Internet draft draft-staniford-cidf-dataformats-00.txt, 1998.

[20] J. Chen and C. Chen, "Design of complex event-processing IDS in internet of things," in *Proceedings of the 6th International Conference on Measuring Technology and Mechatronics Automation (ICMTMA '14)*, pp. 226–229, January 2014.

[21] D. Lee and M. Yannakakis, "Principles and methods of testing finite state machines—a survey," *Proceedings of the IEEE*, vol. 84, no. 8, pp. 1090–1123, 1996.

[22] J. Tretmans, "Conformance testing with labelled transition systems: implementation relations and test generation," *Computer Networks*, vol. 29, no. 1, pp. 49–79, 1996.

[23] Y. Fu and O. Koné, "Security and robustness by protocol testing," *IEEE Systems Journal*, vol. 8, no. 3, pp. 699–707, 2014.

[24] G. Lowe, "Breaking and fixing the Needham-Schroeder Public-Key Protocol using FDR," in *Tools and Algorithms for the Construction and Analysis of Systems*, vol. 1055 of *Lecture Notes in Computer Science*, pp. 147–166, Springer, Berlin, Germany, 1996.

[25] P. Tsankov, M. T. Dashti, and D. Basin, "SECFUZZ: fuzz-testing security protocols," in *Proceedings of the 7th International Workshop on Automation of Software Test (AST '12)*, pp. 1–7, Zurich, Switzarland, June 2012.

[26] B. Lei, X. Li, Z. Liu, C. Morisset, and V. Stolz, "Robustness testing for software components," *Science of Computer Programming*, vol. 75, no. 10, pp. 879–897, 2010.

[27] Y. Fu and O. Koné, "Validation of security protocol implementations from security objectives," *Computers and Security*, vol. 36, pp. 27–39, 2013.

[28] Wireshark, "Wireshark network protocol analyzer.," 2017, http://www.wireshark.org/.

[29] C. Rigney, S. Willens, and A. Rubens, "Remote authentication dial in user service (radius)," Tech. Rep. RFC2865, The Internet Society, Reston, Va, USA, 2000.

[30] FreeRADIUS, "Freeradius-the world's most popular radius server," 2017, http://freeradius.org/.

[31] mastersoft, "Ntradping-radius test utility," 2017, http://www.mastersoft-group.com/.