Intelligent System Security Event Description Method

Jun Hou[1], Qianmu Li[2](✉), Yini Chen[2], Shunmei Meng[2,5], Huaqiu Long[3], and Zhe Sun[4]

[1] Nanjing Institute of Industry Technology, Nanjing 210023, China
[2] School of Cyber Science and Technology, Nanjing University of Science and Technology, Nanjing 210094, China
qianmu@njust.edu.cn
[3] Intelligent Manufacturing Department, Wuyi University, Jiangmen 529020, China
[4] Jiangsu Zhongtian Technology Co, Ltd., Nantong 226463, China
[5] State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China

Abstract. In a cloud environment, the control logic and data forwarding of network devices are separated from each other. The control layer is responsible for the centralized management of network nodes. After it acquires the entire network topology, it can automatically generate a visualized network structure. The security analyst can grasp the connection status of the devices on the entire network in the control domain. The network topology generation method based on the control layer information is directly and efficiently, which can greatly simplify the description of security events in the cloud environment. At the same time, the separate structure also makes the specific details of the underlying network device hidden. Petri-net, as a formal description tool, can be used to describe such a structure. Based on the cloud environment structure, this paper combines the advantages of CORAS modeling and analysis with object-oriented Petri-net theory, and proposes a COP (CORAS-based Object Oriented Petri-net)-based intelligent system security event description method. Model the description of the complexity and dynamics of cloud environment security events.

Keywords: Security event description · CORAS modeling · Petri-net theory

1 CORAS Framework

CORAS is a model-based approach to security risk analysis that maintains and provides the results of the analysis. CORAS is mainly based on some traditional security analysis techniques, such as HazOp, FTA, FMEA, etc., and combines them with system development techniques such as UML to form a modeling analysis description method. CORAS is a graphics and model based approach that give CORAS the following advantages:

(1) CORAS can provide a precise description of the target system. Its syntax and all related security features are easy to use;

X. Zhang et al. (Eds.): CloudComp 2019/SmartGift 2019, LNICST 322, pp. 487–499, 2020.
https://doi.org/10.1007/978-3-030-48513-9_39

(2) The graphical representation of CORAS information enhances the communication and interaction of each participant in the analysis;
(3) CORAS facilitates the documentation of risk assessment assumptions and assessment results.

CORAS can be divided into three different components:

(1) The CORAS Risk Modeling Language: which includes the graphical grammar and textual grammar of the CORAS icon and related semantics;
(2) The CORAS Method: which includes a step-by-step description of the safety analysis process and a guide to constructing a CORAS chart;
(3) The CORAS Tool: This includes tools for documenting, maintaining, and reporting the results of risk analysis.

In addition to including descriptions and analytical methods, the CORAS approach also takes into account international standards for risk management.

1.1 Component-Based CORAS and Petri-Net

In recent years, CORAS has gradually begun to develop toward component-based risk analysis [1]. For complex system analysis tasks, reusable components should be utilized to reduce the amount of work involved, rather than analyzing from scratch. It contains development techniques including syntax, rules, and implementation guidelines for specifying the behavior and system architecture of components. This standardized the incremental analysis of the system. A simple example is given below to illustrate how component-based CORAS describes and analyzes security events. Figure 1 is an example of a modeling analysis of a threat scenario. The circle on the graph represents the threat scene that occurred. Playing files directly is one of the actions.

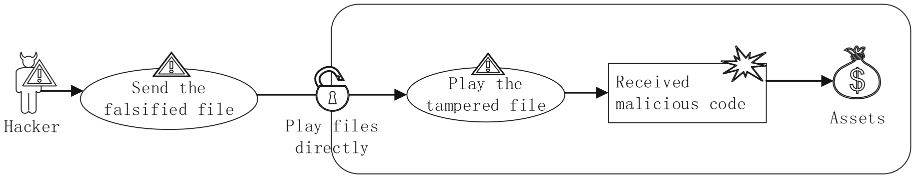

Fig. 1. CORAS modeling analysis of threat scenarios

By sending a tampered music file, the hacker uses the media player buffer overflow vulnerability to threaten the user's related media assets. When the receive file operation is invoked, that is the user plays the file directly, the channel interface calls the tampered music file from the interface of the media player. Once the file is played, it will use a buffer overflow vulnerability to overwrite the pointer address to point to malicious code, threatening the user's assets. In the above threat scenarios, scenarios, risks, and threat assets are defined as individual component objects. The description of the entire security event is done by connecting the calling relationships of the interfaces between the objects. The entire description process is very clear and concise, which is beneficial to the participants involved in the risk analysis and evaluation to understand and communicate the entire risk event. At the same time, related scenes are also very convenient for documenting preservation. If a new threat scenario is created, the entire modeled part is not necessary to make major changes, so the reusability of the model is also guaranteed. However, from the above examples, CORAS can also be found to have shortcomings such as insufficient formal description ability, excessive subjectivity, and insufficient dynamic analysis capability.

Petri-net is a graphical description method based on mathematical theory. It is a special directed graph consisting of the library, transition and flow relationship. And it uses Token to describe the state changes in the graph. The basic Petri-net is defined as following:

Definition 1. Basic Petri-net is a triple:

$$PN = (P, T, F) \tag{1}$$

Where:

(1) P is a finite set of spaces that represent the state of the system; T is a finite set of transitions that represent changes in behavior;
(2) $P \cup T \neq \varnothing,\ P \cap T = \varnothing$;
(3) $F \subseteq (P \times T) \cup (T \times P)$ is a lone set. It is the flow relationship of Petri-net, connecting libraries and transitions;
(4) $Dom(F) \cup Cod(F) = P \cup T$;
$Dom(F) = \{x | \exists y : (x, y) \in F\},\ Cod(F) = \{x | \exists y : (y, x) \in F\}$

2 COP Modeling Method

Definition 2. COP is a risk assessment process that defines it as a triple:

$$COOPN = \{SP,\ OG;\ OF\} \tag{2}$$

Where,

(1) $SP = \{sp_1,\ sp_2, \ldots,\ sp_n\}$ is a sub-process of the COP evaluation process, which can be regarded as a special library;
(2) $OG = \{og_1,\ og_2, \ldots,\ og_n\}$ is a collection of Outer Gate Transitions between sub-processes. In order to comply with the description of COP, this paper extends the transition T to G. G can be seen as a special kind of gate transition. This change has the nature of a gate. This paper introduces two different gate transitions, as shown in Fig. 2:

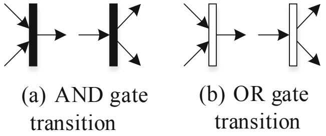
(a) AND gate transition (b) OR gate transition

Fig. 2. Gate transition symbol

(3) $OF = \{of_1, of_2, \ldots, of_n\}$ is a collection of all Outer Flows outside the sub-process, corresponding to the dependencies between the sub-processes.

Definition 3. The COP sub-process sp_i is internally defined as a triple:

$$inner(sp_i) = \{P, IG; IF\} \tag{3}$$

Where,

(1) $P = \{p_1, p_2, \ldots, p_n\}$ is a collection of all the places in the sub-process sp_i;
(2) $IG = \{ig_1, ig_2, \ldots, ig_n\}$ is a collection of all Inner Gate Transitions within sub-process sp_i;
(3) $IF = \{if_1, if_2, \ldots, if_n\}$ is a collection of Inner Flows between all the libraries and transitions in sub-process sp_i.

Definition 4. Sub-process sp_i internal and external communication is defined as a four-tuple, defined as follows:

$$outer(sp_i) = \{IM, OM, OG; OF\} \tag{4}$$

(1) $IM = \{im_1, im_2, \ldots, im_n\}$ is a collection of all In-message queues outside of sub-process sp_i;
(2) $OM = \{om_1, om_2, \ldots, om_n\}$ is a collection of all Out-message queues outside of sub-process sp_i;
(3) The definition of $OG = \{og_1, og_2, \ldots, og_n\}$ and $OF = \{of_1, of_2, \ldots, of_n\}$ is defined in Definition 2;

In the description of modeling using the COP method, the COP model of each object is first given. Secondly, the message input and output interface are defined according to the flow relationship between the objects. Then connect the interfaces according to the flow relationship and initialize the COP model. Finally, a COP analysis was performed.

The COP model initialization algorithm is as follows:

Algorithm: COP model initialization algorithm

Input: Sub-process collection *SP*

Output: COP model initialization result *COP*

```
num← SizeOf($SP$);        // Get the number of sub-process collections
COP ← $\Phi$;                // COP network initialization
for $i$ ← 1 to num do{      // Establish sub-processes
    $sp_i$ ← Pop($SP$, $i$);     // New sub-process $sp_i$
    AddIM2SP($IM$, $sp_i$);  // Add $sp_i$ in-message queue $IM$
    AddP2SP($P$, $sp_i$);  // Add $sp_i$ internal library $P$
    AddIG2SP($IG$, $sp_i$);  // Add $sp_i$ Inner Gate Transition $IG$
    AddOM2SP($OM$, $sp_i$);    // Add $sp_i$ Out-message queue $OM$
    $IF$ ← LinkPandIG($P$, $IG$, $IM$, $OM$);   // calculate $sp_i$ Inner Flow $IF$
    AddIF2SP($IF$, $sp_i$);            // Add $sp_i$ Inner Flow $IF$
    AddSP2COP(COP, $sp_i$);        // Add COP sub-process $sp_i$
}
AddOG2COP(COP, $OG$);          // Add COP Gate Transition $OG$
$OF$ ← LinkPandIG($P$, $IG$);    // Calculate the COP flow relationship $OF$ from the sub-process $SP$ and the gate transition relationship.
AddOF2COP(COP, $OF$);     // Add COP flow relationship $OF$
```

3 Instance Verification

Unified management of the control layer in the cloud environment will introduce new threats. Using network nodes to launch DDoS attacks to controllers is one of them [2]. In order to verify the feasibility and effectiveness of the COP-based cloud environment security event description method, this paper combines the cloud environment structure

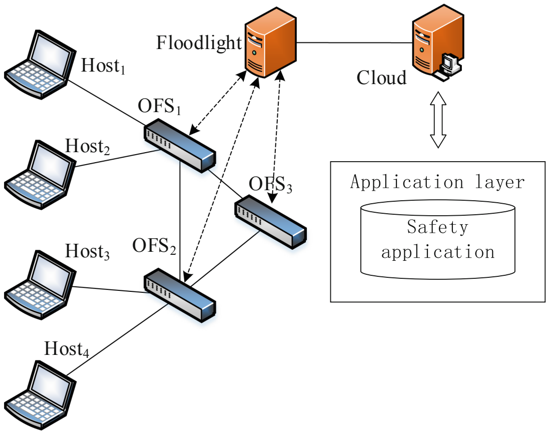

Fig. 3. Experimental environment network topology

proposed in the paper and builds the simulation network environment shown in Fig. 3 by using SDN technology. The paper carried out the DoS attack simulation in the cloud environment and described the security events triggered. The device layer includes multiple hosts, OpenFlow switches, controllers, and application servers. The control layer uses Floodlight as the SDN controller. The application layer runs a security application. The simulation software is mininet [3].

Data packet transmission information is shown in Table 1. After the request is sent, the stream data that is not matched by the OFS flow table will be packaged and delivered to Floodlight. After the Floodlight identifies the packet, it passes the packet to the application layer security application for processing. The security app sends the specified protection policy to Floodlight. Floodlight delivers new flow tables and settings to OFS. Finally, the OFS processes the packet according to the new command. The experiment collects the link bandwidth occupancy rate (*lbor*: link bandwidth occupancy rate), the client packet transmission rate (*psps*: package send per second), and the server-side packet reception rate (*prps*: package received per second) as statistical indicators. The *prps* responses to the attack strength and credibility of the attack. The greater the number of attacks, the more likely the attack is to be a real intrusion.

Table 1. Packet transmission information in the experiment

Number	Send content
p_{51}	Host1 sends ICMP packets to cloud
p_{52}	Host2 sends ICMP packets to cloud
p_{61}	Host3 sends TCP packets to cloud
p_{62}	Host4 sends TCP packets to cloud

A gate threshold value ε can be set as a reference value for the number of alarms, which is dynamically adjusted by the application layer security application, whereby the probability λ of occurrence of a certain attack can be calculated.

$$\lambda_i = \begin{cases} \frac{prps_i}{\varepsilon_i} \ if(n_i < \varepsilon_i) \\ 1 \quad otherwise \end{cases} \tag{5}$$

For an attack, when the data is less than the set gate threshold ε_i, the probability value λ_i of the attack is represented by $\frac{prps_i}{\varepsilon_i}$. When the threshold ε_i is exceeded, the probability value λ_i of the attack is considered to be 1.

It is also possible to divide the transmission frequency $prps_i$ into different intervals according to the provisions of GB20984-2007 as the basis for the attack threat assignment. The division between intervals can be divided into non-equal divisions, as shown in Table 2. In this way, the probability λ of an attack occurring is calculated.

The experiment uses the first attack probability calculation method as the evaluation basis. First, Host1 sends ICMP packets at a lower frequency, and Host3 and Host4 send TCP packets at a lower frequency. Host2 sends ICMP packets at increasing frequency

Table 2. Attack probability assignment table

Assignment	Identification	Threat frequency	Frequency range	λ_i
5	Very high	Occur frequently	$> 50\% \cdot lbor$	1
4	High	Very likely to happen	$(20\% \sim 50\%) \cdot lbor$	0.5
3	Medium	Likely to happen	$(10\% \sim 20\%) \cdot lbor$	0.2
2	Low	Less likely to happen	$(5\% \sim 10\%) \cdot lbor$	0.1
1	Very low	Extremely rare	$< 5\% \cdot lbor$	0.01

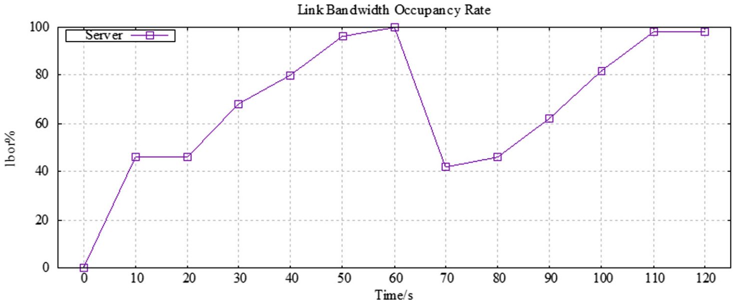

Fig. 4. Cloud link bandwidth occupancy rate

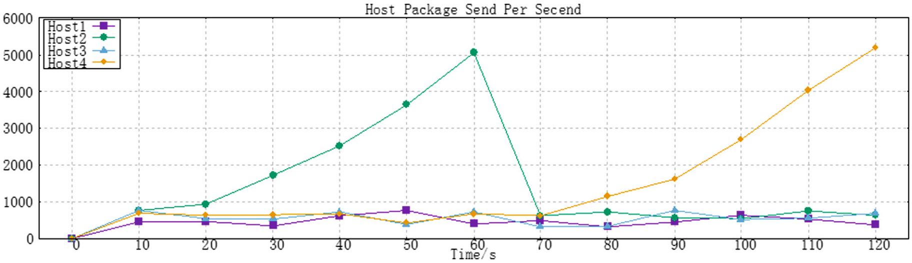

Fig. 5. Host packet transmission frequency

until it occupies all of the link bandwidth and then drops to normal. Host4 then sends TCP packets with increasing frequency until it occupies all of the link bandwidth and then drops to normal. The cloud link bandwidth occupancy, host packet transmission frequency, and cloud packet reception frequency in the experiment are shown in Fig. 4, Fig. 5 and Fig. 6.

It can be seen that as the two DoS attacks progress, the bandwidth is occupied in a large amount, normal traffic cannot be sent, and the connection cannot be established. First, the experimental results were analyzed, and the 27th second of the experiment was selected as the analysis time point, as shown in Fig. 7. In the figure, the red horizontal

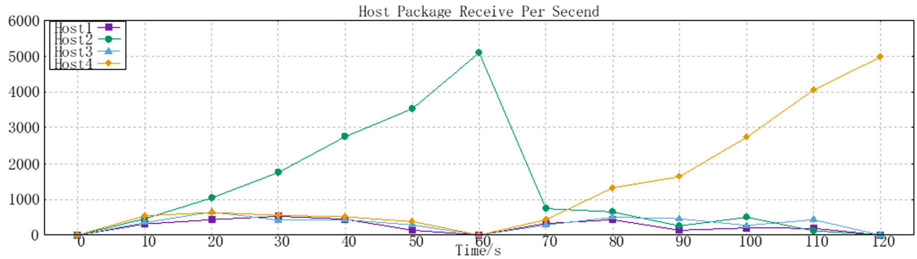

Fig. 6. Cloud packet receiving frequency

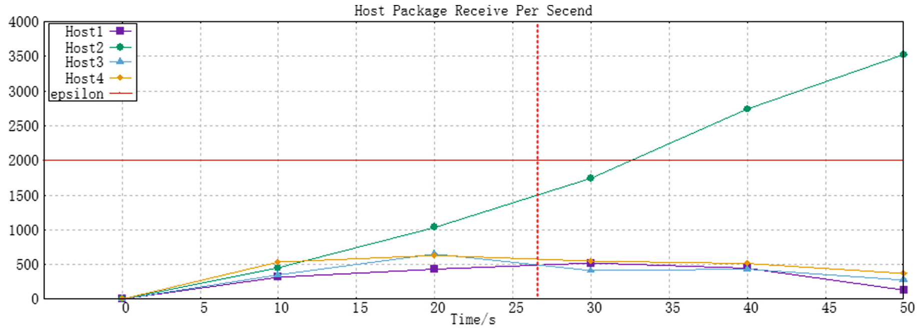

Fig. 7. Cloud packet receiving frequency fragment

Table 3. The possibility of an attack at this moment

Number	Packet acceptance frequency *psps*	Gate threshold value ε	Attack possibility λ
p_{51}	489	2000	0.244
p_{52}	1507	2000	0.753
p_{61}	502	2000	0.251
p_{62}	587	2000	0.294
$p_{51}\ \ p_{52}$	Total: 1996	Total: 3085	Proportion: 0.647
$p_{61}\ \ p_{62}$	Total: 1089		Proportion: 0.353

line is the gate threshold value ε, and the red vertical line is the 27th second of the experiment. Assume that both the ICMP gate threshold ε_1 and the ε_2 of TCP are 40% of the bandwidth occupied by the Cloud packet.

Table 3 lists the data on the likelihood of an attack occurring at the red vertical dashed line.

3.1 Attack Scene COP Modeling Definition

The moment is modeled and analyzed according to the COP modeling step:

The moment contains five sub-processes, in which Host$_1$ ~ Host$_4$ are recorded as potential attack initiators as sub-process $sp_1 \sim sp_4$. Two different potential attack behaviors ICMP and TCP belong to two different sub-processes sp_5 and sp_6. The attacked server is the target Recorded as sub-process sp_7.

(1) Initialize the COP network, assign Φ;
(2) New a sub-process sp_1. sp_1 does not have a library and transitions that need to be described in detail, add sp_1 to the COP network. Similarly, new a sub-process $sp_2 \sim sp_4$. $sp_2 \sim sp_4$ does not have a library and transitions that need to be described in detail. $sp_2 \sim sp_4$ is added to the COP network;
(3) Create a new subprocess sp_5. The behavior im_{51} that initiates the attack within the A sub-process is taken as the input of sp_5. It can be seen from Table 1 that sp_5 includes p_{51}, p_{52} suspected of initiating an ICMP (ig_{51}) attack. Since p_{51}, p_{52} belong to the same ICMP attack ig_{51}, they conform to the "AND" relationship, so add the AND transition ig_{51} to sp_5. Finally, the consequences of the attack are taken as the output om_{51} of sp_5 and added to sp_5. Calculate the internal *IF* of sp_5. Add the internal stream relationship *IF* to sp_5. Add sp_5 to the COP network. Similarly, modeling can get sp_6 and add sp_6 to the COP network.
(4) It can be seen from Table 1 that $sp_1 \sim sp_4$. randomly initiates an attack can make a affection of sp_7, so there is a logical OR relationship between the attack behaviors. Add OR gate transitions og_1, og_2 and og_3 to the COP. Calculate OF based on the relationship between the elements and add to the COP.
(5) Improve the COP network;

The modeling results are shown in Fig. 8.

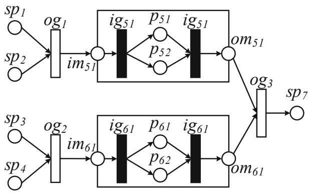

Fig. 8. COP model generated based on attack scenario information

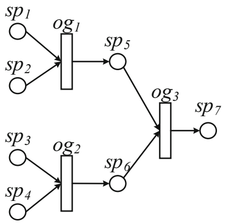

Fig. 9. Dynamically scaled COP model

3.2 COP Method Analysis

(1) **Qualitative description**
In the qualitative description, this way of independent scaling of sub-processes and the describing way of completing the closing and opening of the detail implements

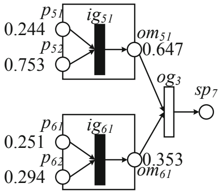

Fig. 10. COP model with attack probability

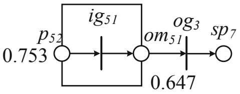

Fig. 11. COP model with an attack gate threshold of 0.6

a description of the different levels of refinement. The sub-processes that have completed the analysis at the same time can be saved independently as the analysis results. Portions of the same analysis content encountered in other analyses can be directly replaced to achieve reuse of the model.

The qualitative results are shown in Fig. 9. It can be clearly seen that $sp_1 \sim sp_4$ initiates two different attacks sp_5 and sp_6 against sp_7. The results of each attack analysis can be saved separately to implement model reuse. The attack process can be scaled independently to achieve a different level of description.

(2) **Quantitative description**

In the quantitative description, the analysis can be performed based on the connection relationship in the COP network. Suppose that the risk of an object being attacked is F. It can be seen from the definition of COP that in the case of the transition of the AND gate, the value of F is determined by the sum of the possibility of initiating the attack precondition. In the case of an OR gate transition, the value of F is determined by the maximum probability of initiating an attack precondition. Bring the possibility of potential attack at this moment in Table 3 to Fig. 8. The possibility of each attack content and attack type is shown in Fig. 10.

According to the definition, the risk value of the possible attack node sp_7 is calculated as follows:

$$\begin{aligned} F(sp_7) &= MAX[0.647 \cdot SUM(0.244,\ 0.753), 0.3 \cdot SUM(0.251,\ 0.294)] \\ &= MAX[0.997,\ 0.545] \\ &= 0.997 \end{aligned} \tag{6}$$

Assuming that the probability of attack to be analyzed exceeds 0.6, the new COP model is shown in Fig. 11.

Among them, ig_{51}, og_3 degenerates into a normal gate transition. At this time, the risk value of sp_7 is:

$$F(sp_7) = MAX[0.647 \cdot SUM(0.753)] = 0.487 \tag{7}$$

(3) **Strategic response**

Once it is detected that the actual risk value of the relevant asset exceeds the acceptable risk value (assumed to be 0.5), the application-level security application performs the flow table update according to the set rules. Then, depending on the magnitude of the risk value, a new forwarding path can be set to offload, limit or block certain stream data. In the experiment, if the gate threshold is exceeded, the forwarding request of the relevant network segment is discarded, and the stream data is discarded. After setting the rules, the Cloud link bandwidth occupancy, Host packet transmission frequency, and Cloud packet reception frequency are shown in Fig. 12, Fig. 13 and Fig. 14. It can be seen that in the case where the transmission packet law is unchanged in the simulation network, the transmission source with the attack intention is blocked, the link occupancy rate of the Cloud end is significantly reduced, and the normal service is guaranteed.

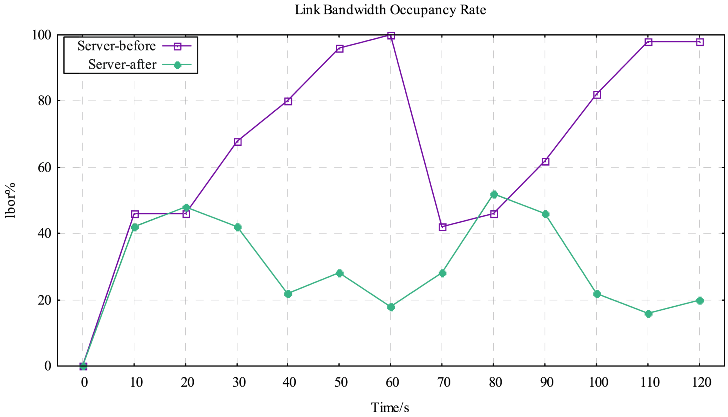

Fig. 12. Cloud link bandwidth usage

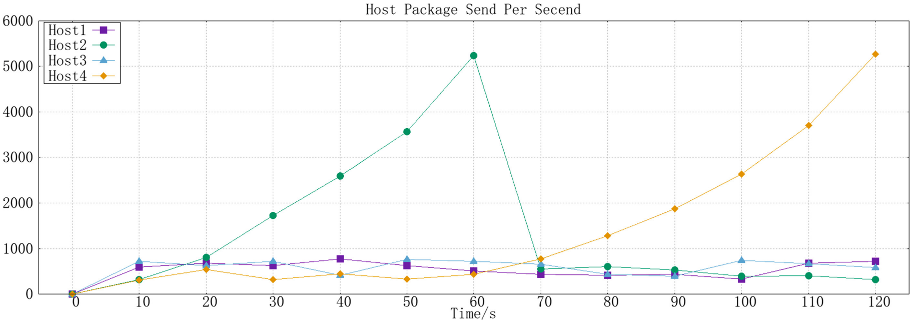

Fig. 13. Host packet transmission frequency

COP inherits CORAS's graphical description, reusability and refined description of the advantages, and uses object-oriented Petri-net to increase the advantages of formal description, scalability and dynamic verification. At the same time, the data source of CORAS quantitative analysis is transformed from subjective expert evaluation into objective scanning analysis, which reduces the human factors in the analysis process and makes the results more reliable.

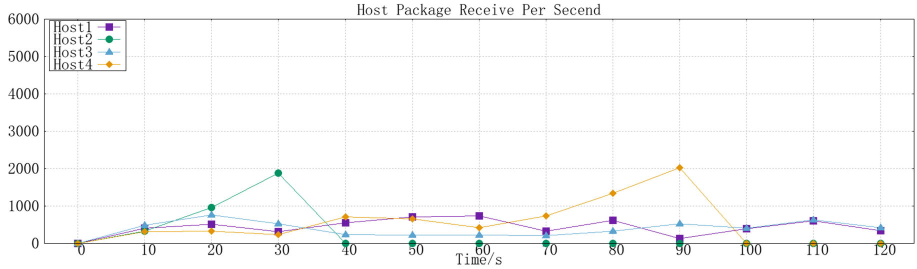

Fig. 14. Cloud packet receiving frequency

4 Conclusion

In this paper, a model-based static security event modeling description method CORAS and object-oriented Petri-net are combined to propose a COP-based security risk modeling method. Compared with the existing model-based methods, the proposed COP model not only inherits the existing model's extensibility, reusability, and refinement description but also enhances the formal description and dynamic analysis capabilities. In the cloud environment structure, the description of the entire network in the control domain can be directly generated based on the control layer information, and the efficiency is far superior to the topology discovery technology in the traditional network. The attack simulation experiment proves that COP can effectively describe the cloud environment security incidents, and can further carry out risk strategy response based on the description results.

This work was supported in part by the Fundamental Research Funds for the Central Universities (30918012204), Military Common Information System Equipment Preresearch Special Technology Project (315075701), 2019 Industrial Internet Innovation and Development Project from Ministry of Industry and Information Technology of China, 2018 Jiangsu Province Major Technical Research Project “Information Security Simulation System”, Shanghai Aerospace Science and Technology Innovation Fund (SAST2018-103).

References

1. Hong, J.E., Bae, D.H.: Software modeling and analysis using a hierarchical object-oriented Petri net. Inf. Sci. Int. J. **130**, 131–164 (2000)
2. Brændeland, G., Dahl, H.E.I., Engan, I., Stølen, K.: Using dependent CORAS diagrams to analyse mutual dependency. In: Lopez, J., Hämmerli, B.M. (eds.) CRITIS 2007. LNCS, vol. 5141, pp. 135–148. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-89173-4_12
3. De Oliveira, R.L.S., Shinoda, A.A., Schweitzer, C.M., et al.: Using mininet for emulation and prototyping software-defined networks. In: IEEE Colombian Conference on Communications and Computing, pp. 1–6. IEEE, Bogota (2014)
4. Liu, X., Wang, H., Lai, J., et al.: Multiclass support vector machines theory and its data fusion application in network security situation awareness. In: 2007 International Conference on Wireless Communications, Networking and Mobile Computing, pp. 6349–6352. IEEE, Shanghai (2007)

5. Shin, S., Yegneswaran, V., Porras, P., et al.: AVANT-GUARD: scalable and vigilant switch flow management in software-defined networks. In: ACM SIGSAC Conference on Computer & Communications Security, pp. 413–424. ACM (2013)
6. Xu, X., Liu, Q., Zhang, X., Zhang, J., Qi, L., Dou, W.: A blockchain-powered crowdsourcing method with privacy preservation in mobile environment. IEEE Trans. Comput. Soc. Syst. **6**(6), 1407–1419 (2019)
7. Qi, L., Chen, Y., Yuan, Y., Fu, S., Zhang, X., Xu, X.: A QoS-aware virtual machine scheduling method for energy conservation in cloud-based cyber-physical systems. World Wide Web **23**, 1275–1297 (2020). https://doi.org/10.1007/s11280-019-00684-y
8. Qi, L., et al.: Finding all you need: web APIs recommendation in web of things through keywords search. IEEE Trans. Comput. Soc. Syst. **6**(5), 1063–1072 (2019)
9. Li, Q., Meng, S., Wang, S., Zhang, J., Hou, J.: CAD: command-level anomaly detection for vehicle-road collaborative charging network. IEEE Access **7**, 34910–34924 (2019)
10. Li, Q., Meng, S., Zhang, S., Hou, J., Qi, L.: Complex attack linkage decision-making in edge computing networks. IEEE Access **7**, 12058–12072 (2019)
11. Li, Q., et al.: Safety risk monitoring of cyber-physical power systems based on ensemble learning algorithm. IEEE Access **7**, 24788–24805 (2019)
12. Li, Q., Wang, Y., Pu, Z., Wang, S., Zhang, W.: A time series association state analysis method in smart internet of electric vehicle charging network attack. Transp. Res. Rec. **2673**, 217–228 (2019)