Poster: Modeling of APT Attacks through Transforming Attack Scenarios into DEVS Models

Jiyeon Kim
Department of Electrical and Computer Engineering
Carnegie Mellon University
Pittsburgh, USA
kimjy@andrew.cmu.edu

Hyung-Jong Kim
Department of Information Security
Seoul Women's University
Seoul, Republic of Korea
hkim@swu.ac.kr

***Abstract*—Detection and prevention of sophisticated cyber-attacks are challenging. Advanced persistent threat (APT) attacks are one of the most visible attacks that can show such attack trends. Predicting and defending APT attacks are difficult due to a variety of attacks conducted at each stage. Simulations can be a safer and cheaper way of developing countermeasures as well as analyzing such attacks. Modeling cyber-attacks depends on attack scenarios and abstraction levels according to simulation purposes. The simulation models would vary even when we model a same attack. It is very hard to model a variety of attack scenarios due to a lack of modeling methodology in cyber-security area. In this paper, we propose a method for modeling APT attacks by transforming attack scenarios into DEVS (Discrete EVent system Specification) simulation models. DEVS is a modular and hierarchical formalism to specify discrete event systems. Modeling cyber-attacks as DEVS models enables us to reuse the modularized models for other scenarios. It is also easy to implement and execute the models using DEVS simulation engines.**

Keywords—APT attacks; cyber-attack; cyber-security; attack simulation; DEVS formalism;

I. AUTHORING APT ATTACK SCENARIO

We author an example scenario based on most common mechanism of APT attacks. APT attacks usually consist of five steps: exploration, infection of malicious code, acquisition of authorization, and information leakage. In order to model such attacks, we should create a scenario for each step. [1] defines six elements needed to represent behaviors of cyber-attacks and defenses. The six elements are as follows: source element, attack/defense behavior, destination element, processing time, return state and output. Our scenario contains all the elements so that the scenario can represent the whole attack flow. We then map the scenario onto the modeling elements. This work helps modelers to develop a variety of scenarios as DEVS [2] models easily.

A. Scenario

Step 1. An ***attacker*** ***collects*** ***email*** addresses for ***100t*** (unit time). Repeat ***this step (collect)*** until this behavior succeeds.
Step 2. The ***attacker*** ***sends*** an ***email*** to ***TargetA***, one of the collected addresses for ***10t***. Repeat ***this step*** until this behavior succeeds.
Step 3. ***TargetA*** ***opens*** the ***email*** from the attacker for ***5t***. Repeat ***this step*** until this behavior succeeds. (As a result, the TargetA is infected by malicious code, and the attacker can acquire an authorization for TargetA's internal network, TargetB.)
Step 4. ***TargetA*** ***accesses*** to ***TargetB*** for ***20t***. Repeat ***this step*** until this behavior succeeds.
Step 5. ***TargetA*** ***requests*** a critical file by sending a request ***message*** to ***TargetB*** for ***15t***, in order to leak information. Repeat ***this step*** until this behavior succeeds.
Step 6. ***TargetB*** ***copies*** the ***file*** for ***30t***. Repeat ***this step*** until this behavior succeeds.
Step 7. ***TargetB*** ***transmits*** the ***file*** to ***TargetA*** for ***10t***. Repeat ***this step*** until this behavior succeeds.
Step 8. ***TargetA*** ***transmits*** the ***file*** to ***attacker*** for ***10t***. Repeat ***this step*** until this behavior succeeds.
Step 9. Repeat ***steps 5 to 8***.

B. Mapping a scenario onto modeling elements

Here we extract the six elements from the above scenario as shown in Table 1.

Table 1. Extraction the modeling elements from our scenario

Step	Source element	Attack behavior	Destination element	Processing time	Return state	Output
1	attacker	collect	email	100	collect	-
2	attacker	send	TargetA	10	send	email
3	TargetA	open	email	5	open	-
4	TargetA	access	TargetB	20	access	-
5	TargetA	request	TargetB	15	request	message
6	TargetB	copy	file	30	copy	-
7	TargetB	transmit	TargetA	10	transmit	file
8	TargetA	transmit	attacker	10	transmit	file

We can obtain the attack process from the fields of source element, attack behavior, destination element and output, as shown in Fig. 1.

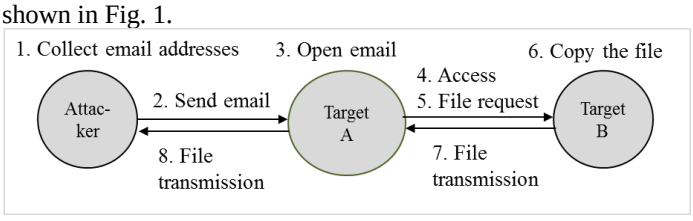

Fig. 1. Process of an APT attack derived from our scenario

This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education(NRF-2013R1A1A2062654)

II. DEVS MODELING OF APT ATTACK

The six elements correspond to all the DEVS elements.

- Source/destination element –a multicomponent system M
- Attack behaviors/return state – a set of states S
- Output –sets of input X and output Y
- Processing time – a time base T

A. Specification

We can define the sets of DEVS elements for the APT attack as follows:

$M = \{attacker, TargetA, TargetB\}$

$X_{attacker} = \{file\}$
$Y_{attacker} = \{email\}$
$X_{TargetA} = \{email, file\}$
$Y_{TargetA} = \{file, message\}$
$X_{TargetB} = \{message\}$
$Y_{TargetB} = \{file\}$

Where s_0 is an initial state,
$S_{attacker} = \{s_0, collect, send\}$
$S_{TargetA} = \{s_0, open, access, request, transmit\}$
$S_{TargetB} = \{s_0, copy, transmit\}$

An external output coupling (EOC) is as follows:

$EOC = \{(attacker,TargetA), (TargetA,attacker), (TargetA,TargetB), (target, TargetA)\}$

B. Structure and state diagram

From the specification in Section 2.A and the fields of processing time and return state in Table 1, we can finally obtain DEVS models as shown in Fig. 2.

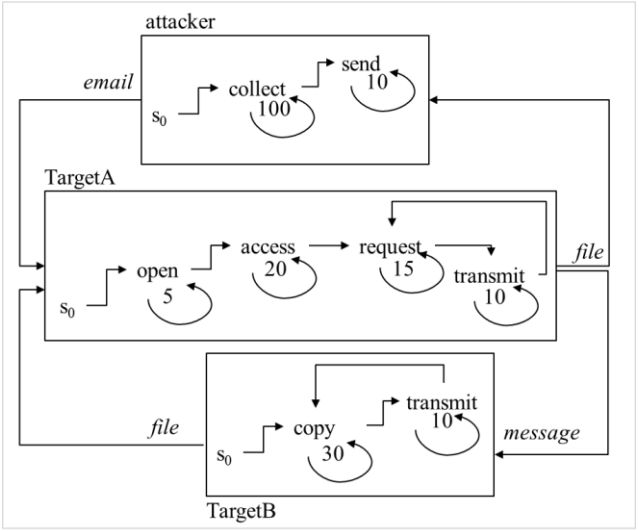

Fig. 2. Structure and state diagram of our APT attack model

III. ADVANTAGES OF USING DEVS FORMALISM

A modeling of discrete event systems is a process for tracing changing state variables by input/output events. Since a cyber-attack simulation progresses based on the interactions occurred by the attacks and defenses, we are able to observe the changes of state variables of cyberspace elements during the simulation. DEVS formalism is one of the specification methods for discrete event simulations. There are three advantages in modeling cyber-attacks based on DEVS. First, from the perspective of modeling theory, DEVS is a general modeling methodology that provides a mathematical frame to specify discrete event systems. This allows us to make use of DEVS regardless of the types of attack mechanisms or characteristics of the cyberspace elements. DEVS is also able to separate models from their interfaces by defining input/output interfaces such as internal, external and output functions. We do not need to change each model's design although its connections may change. Therefore, we can design modular models according to experimental purposes and can easily interwork with models such as those that represent cyber-attacks using differential equations. Furthermore, it is easy to transfer DEVS models into other types of discrete event system models, because they have general elements required for the modeling of discrete event systems. Second, from the perspective of model development, we can easily model by mapping entities that affect cyber-attacks in the real world to DEVS objects using an object-oriented concept. Lastly, from the perspective of model execution environments, DEVS models can be implemented and executed independent of the programming language or simulation engine.

REFERENCES

[1] J.Y and H.J., "Defining Security Primitives for Eliciting Flexible Attack Scenarios Through CAPEC Analysis." Information Security Applications. Springer International Publishing, 2014. 370-382.

[2] Zeigler, Bernard P., Herbert Praehofer, and Tag Gon Kim. Theory of modeling and simulation. 2nd edition. Academic Press, 2000.

[3] Giura, Paul, and Wei Wang. "A context-based detection framework for advanced persistent threats." Cyber Security (CyberSecurity), 2012 International Conference on. IEEE, 2012.

[4] Tankard, Colin. "Advanced Persistent threats and how to monitor and deter them." Network security 2011.8 (2011): 16-19.

[5] Baize, Eric. "Developing secure products in the age of advanced persistent threats." IEEE Security & Privacy 10.3 (2012): 0088-92.

[6] Aslan A., Stephen C. "Learning is Change in Knowledge: Knowledge-based Security for Dynamic Policies."2012 IEEE 25th Computer Security Foundations Symposium. (2012): 308-322.

[7] Guizani, Mohsen, et al. Network Modeling and Simulation: A Practical Perspective. Wiley. com, 2010.

[8] Kuhl, Michael E., et al. "Cyber attack modeling and simulation for network security analysis." Proceedings of the 39th Conference on Winter Simulation: 40 years! The best is yet to come. IEEE Press, 2007.

[9] Sood, Aditya K., and Richard J. Enbody. "Targeted cyberattacks: a superset of advanced persistent threats." IEEE security & privacy 11.1 (2013): 54-61.

[10] Smith, Allen M., and Nancy Y. Toppel. "Case study: Using security awareness to combat the advanced persistent threat." 13th Colloquium for Information Systems Security Education. 2009.