Generation of attack scenarios by modeling CSP for Evaluating and Testing Intrusion Detection System

Mohammed SABER, Toumi BOUCHENTOUF, Mohammed Ghaouth BELKASMI and Abdelhamid BENAZZI

Laboratory MATSI (ENSAO)
Laboratory Mathematics Applied, Treatment of the Signal and Computer Science,
Department of Computer Science
National School of Applied Sciences
Mohammed First University Oujda (Morocco),
ENSAO BP 669 Complex academic Al Qods Oujda 60000 (Morocco)

Summary

We focus in this paper to improve the level of intrusion detection system (IDS). This improvement is based on three research areas: classification of attacks, generation of attack scenarios and finally the evaluation methods. We will discuss in this article the second area, which is to seek meaningful scenarios to minimize the false and positive alerts reported by an IDS. We present a model of the generation of these scenarios based on constraint programming (CSP). The implementation will be done using the Java based library CHOCO.

Key words:
IDS, Evaluation, Attack, Scenario, CSP, CHOCO, Java

1. Introduction

Our main research area is to test and evaluate IDS (Intrusion Detection System). The objective is to develop a classification model of attacks (class model of attacks). Then model the attack process, and generate attack scenarios.

Regarding the classification model attacks in Saber and al. [1] and Saber and al. [2] we have presented a better classification model attacks Gad and al. [3] by eliminating duplication of class. This allowed us to reduce the number of meaningful classes by using the method CTM (Classification Tree Method) [5] using the tool CTE (Classification Tree Editor) [6].

The purpose of classification is to minimize false and positive alerts reported by an IDS. But the attacks follow several scenarios depending on the nature and purpose of the attack which makes the implementation of these different classifications very hard on an IDS.

In this paper, we focus on the generation of attack scenarios. We adopted the process model of malware attacks Gad and al. [4]. We propose an algorithm based on constraint satisfaction problem (CSP) to generate attack scenarios from this model. Our goal is to generate a significant minimum number of attack scenarios, which will facilitate the model integration in an IDS, and thus facilitate its evaluation. Indeed, the aim is that the IDS can detect an attack as quickly as possible before it becomes an intrusion.

This paper is composed as follows: in section two we will make a description of the model of attack process of Gad and al. [4]. We will detail in section three the modeling problem, we present the CSP algorithm based on CHOCO and used to generate scenarios in Section 4. In the fifth section we present the results and then we will start a discussion. We end with a conclusion and future work.

2. Model of Attacks Process

There are several models of attacks [8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18]. They are generally specific to the runtime environment, and therefore require a precise and detailed knowledge of architecture, topology and network vulnerabilities and considered system. Moreover, these models are based primarily on known vulnerabilities and ignore the attacks that may exploit still unknown vulnerabilities, which would constitute a serious limitation, since the robustness of IDS also depends on unknown vulnerabilities and new attacks.

In this paper we have adopted the model of the attack process of Gad and al. [4] which is based on a preliminary analysis of malicious attacks like the most prevalent viruses and worms. This choice is justified by the fact that this model is the result of the analysis of more than 70 malware from the CME List (Mitre's Common Malware

Manuscript received November 5, 2010
Manuscript revised November 20, 2010

Enumeration list) [7], which are representative of the most dangerous and more widespread attacks. Indeed, given that worms are autonomous, they must include all the steps in a process of attack. In addition, viruses such as worms can be seen as a class of automated attacks developed by skilled attackers, and this can help understand how interactive attacks can be conducted.

This model is described in Figure (Figure1). It distinguishes the following steps:

1. Recognition (Reconnaissance): it is logical for an attacker to find the necessary information on potential victims before targeting them with the most appropriate attack tools (exploit codes, toolkits).

2. Gain access : to achieve their objectives, attackers usually need access to victims resources, the level of access required will obviously depend on the attack. However, some types of attacks such as denial of service attacks, do not need access to the victim machine.

3. Privilege Escalation: Access originally obtained by the attacker is sometimes insufficient to achieve the attack, in which case, the attacker tries to increase its privileges to have more power (for example, switch from user mode to administrator mode to access the system resources).

4. Browsing Victim : after having acquired sufficient privileges, the attacker usually tries to explore the machine or the target network (eg, searching files and directories), to search for a particular account (as a guest account or an anonymous ftp account), to identify the hardware components, to identify installed programs or to search for trusted hosts (typically, those with certificates installed on the victim machine).

5. Principal Actions: as shown in Figure (Figure1), this step may take different forms, for example, an attacker can execute a denial of service attacks, install malicious code, compromising the integrity of data or run a program.

6. Hiding Traces : the most experienced attackers generally use this last step to erase their tracks, thereby making detection more difficult.

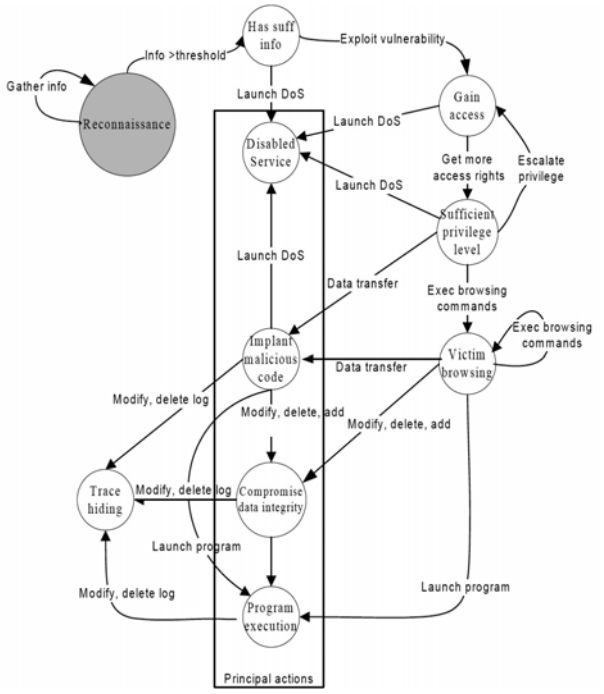

Fig 1. Model of attack process

To generate attack scenarios, Gad and al. proposed a simplified model called the state machine, illustrated in Figure (Figure2).

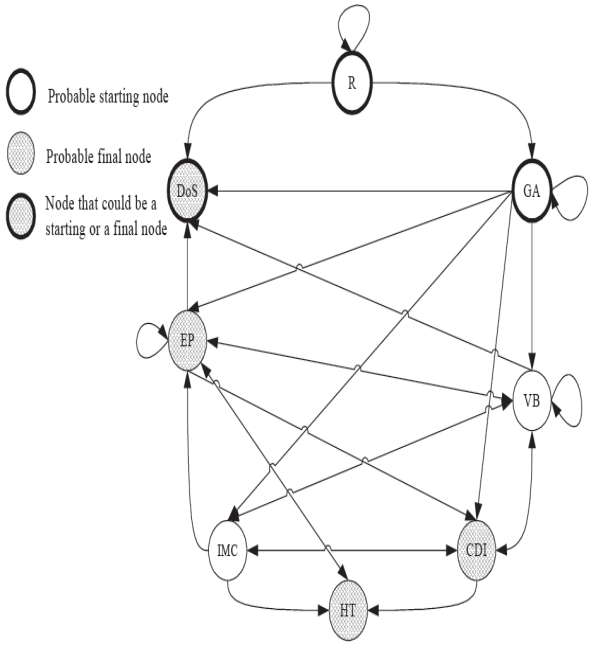

Fig 2. State machine representing the attack process

The steps taken by the malware attacks can be classified into only 8 primitive and each identified by a symbol, as indicated below:

- ✓ R: Recognition (Reconnaissance)
- ✓ VB: Exploration of the machine /or the network of the victim (Victim Browsing)
- ✓ EP: Program execution (Execute Program)
- ✓ GA: Gain Access
- ✓ IMC: Implementation of malicious code (Implant Malicious Code)
- ✓ CDI: Compromise of integrity (Compromise Data Integrity)
- ✓ DoS: DoS (Denial of Service)
- ✓ HT: erasing traces (Hide Traces)

The graph in Figure (Figure2) allows the generation of attack scenarios at an abstract level. By applying constraints on the paths between nodes and consecutive repetition of the same action (loops), as shown in the connection matrix of Figure (Figure3), we can find a set of valid abstract scenarios. A valid scenario is a combination of these nodes with constraints that lead to an attack or intrusion. Here are some examples of scenarios that we can generate from the graph in Figure (Figure2):

- ✓ Scén_1 = (R, GA, DoS) : Start by obtaining recognition and gain access and end with a DoS attack.
- ✓ Scén_2 = (R, DoS) : Start by recognizing and end with a DoS attack.
- ✓ Scén_4 = (R, R, R, DoS) : Start by recognizing several times and end with a DoS attack, this scenario is equivalent to the scenario Scén_2 = (R, back).

	R	*GA*	*DoS*	*VB*	*CDI*	*EP*	*IMC*	*HT*
R	1	1	1	0	0	0	0	0
GA	0	1	1	1	1	1	1	0
DoS	0	0	0	0	0	0	0	0
VB	0	0	1	1	1	1	1	0
CDI	0	0	0	1	1	1	1	1
EP	0	0	1	1	1	0	1	1
IMC	0	0	0	1	1	1	0	1
HT	0	0	0	0	0	1	0	0

Fig 3. Connection Matrix

- ✓ 1 : there is a relationship between two nodes, a node is the son of an other node.
- ✓ 0 : no relationship between two nodes, one node is not the son of an other node.

It is important to note that this iterative approach for generating attack scenarios has overcome the problem of combinatorial explosion, inherent problem to conventional approaches to generating attack scenarios.

The problem we want to solve is to find efficient algorithms that can generate valid meaningful attack scenarios. It would be easier to incorporate the state machine model in an IDS, to test and evaluate it. The modeling of our problem is presented in the next section.

3. Modeling the Problem

In order to facilitate the modeling we will use Figure (Figure 4), And the Correspondence table (Table 1):

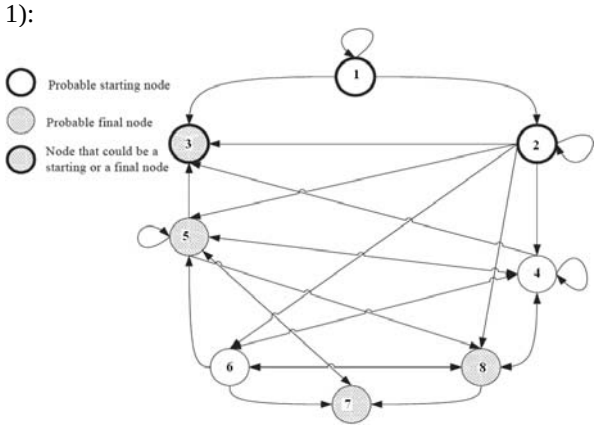

Fig 4. Matching state machine

Table 1: Correspondence table

Node	***Corresponding value***	***Node***	***Corresponding value***
R	1	EP	5
GA	2	IMC	6
DoS	3	HT	7
VB	4	CDI	8

ND is the set of all nodes $\mathrm{ND} = \{1, 2, 3, 4, 5, 6, 7, 8\}$,

$ND_{\text{departure}}$ is the set of starting nodes $ND_{\text{departure}} = \{1,2,3\}$,

ND_{final} is the set of final nodes $ND_{final} = \{3,5,7,8\}$,

L (ND) is the set of the subset of ND, c.à.d,

$X \in L(ND) \Leftrightarrow X \subset ND$

R is the relation defined by:

$ND \rightarrow L(ND)$

$R : x \rightarrow R(x) = X$

X is the set of the nodes son of x.

Example : $R(1)=\{1,2,3\}$; $R(2)=\{2,3,4,5,6,8\}$; $R(3)=\phi$ // empty set ; $R(4)=\{3,4,5,6\}$; $R(5)=\{3,4,5,7\}$; $R(6)=\{5,7\}$; $R(7)=\phi$ // empty set ; $R(8)=\{4,6,7\}$.

Definition 1: scenario

A scenario SN_K of size K is k-uple $(x_1, x_2, x_3, ..., x_k)$, such as:

x_i is the son of node x_{i-1} $\forall\, i \neq 1$

One notes: $SN_K = (x_1, x_2, x_3, ..., x_k)$ with $x_i \in R(x_{i-1})\ \forall\, i \neq 1$

For example :(x_1 = R) => (x_2 = GA) => (x_3 = DoS)

Definition 2: valid scenario

SN_K is a valid scenario if and only if:

$$\begin{cases} SN_K \quad est\ un\ sc?ario \\ x_1 \in ND_{departure} \\ x_k \in ND_{final} \end{cases}$$

S_K is the set of valid scenarios having K nodes:

We have S_K ={ SN_K / SN_K is a valid scenario of size K }

cardND = card(ND) : the number of the ND elements.

S is the set of all valid scenarios: $S = \bigcup_{k=1}^{+\infty} S_k$

Definition 3: equivalent scenarios

Two scenarios A and B are equivalent if and only if: $\exists\ p \in ND^n\ and\ q \in ND^m\ with\ (n,m) \in \aleph x \aleph$, such as A=(p,p,q) and B=(p,q) or A=(q, q,p) and B=(q,p).

$\dot{S}_K$ is the set of the valid equivalent scenarios: we notes

$\dot{S}_K$ ={ SN_K / SN_K is a valid scenario of size K }

$\dot{S}$ is the set of valid equivalent scenarios, given: $\dot{S} = \bigcup_{k=1}^{cardND} \dot{S}_k$

Particular case K=1:

$SN_1 = (x_1)$, is valid if and only if

$x_1 \in ND_{depart} \bigcap ND_{final}$, thus

$x_1 = ND_{depart} \bigcap ND_{final}$.

We propose in the next section a presentation of our solution using constraint programming (CSP).

4. Presentation of CSP and CHOCO

A CSP "constraint satisfaction problem" is modeled as a set of constraints imposed on variables, each of these variables taking values in a domain. More formally, a CSP will be defined by a triplet (V, D, C) such that:

- ✓ $V = (x_1, x_2, x_3, ..., x_k)$ is the set of variables (unknowns) of the problem.
- ✓ D is the function that maps each variable x_i to its domain , That is to say $D(x_i)$, wich means all possible values of x_i .

$$\begin{cases} x_1 \in ND_{departure} \\ x_k \in ND_{final} \\ D(x_i) = \bigcup_y R(y) \quad \forall y \in D(x_{i-1}) \quad ou \quad i \neq 1 et i \neq k \end{cases}$$

- ✓ C is the set of constraints. Each constraint is a relation between certain variables V, restricting the values that these variables can take simultaneously.

The implementation of solving algorithms for this problem uses several languages such as Mozart [19], Jacop [20], ILOG [21] and the java library of Choco solver [22]. We have adopted the CHOCO library to generate our algorithm.

The resolution of the CSP thus formulated, has been implemented in Java using the API (Application Programming Interface) CHOCO. This solver requires the submission of a description of the variables, their domains and the set of constraints as shown in Figure (Figure5).

Fig 5. CHOCO Operating principle

5. Obtained Results and Discussion

We implemented the solution obtained by developing an application using the framework Zk [23] for data acquisition and the library CHOCO to display the results. Indicating the size of the desired valid scenario, we get the number of valid scenarios and the time taken to search. So we have calculated the number of valid scenarios of size k, k varying from 1 to 8, and the time needed to find a scenario of size k. The following table summarizes the results.

Table 2: Obtained Results

Scenario size k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Number of valid scenarios	1	4	13	34	63	73	42	8
Time needed (ms)	1	10	15	24	49	68	63	213

Scenarios thus generated are 238 valid scenarios, and required an average of 405 milliseconds. We also find that the largest the scenario is the more time the generation of valid scenarios takes. So that this solution can be implemented in an IDS, we must first implement the model mentioned in the Figure (Figure 1) and then evaluate it in relation to other IDS that treat malware attacks.

6. Conclusion & Future Works

In this paper we presented an algorithm based on CSP to generate meaningful attack scenarios based on the model proposed by Gad and al. [4] to represent attacks like malware (viruses, Trojan.).

After modeling the generation of attack scenarios as a CSP problem, we implement a solution based on CHOCO using the Java language.

This solution provided the number of valid scenarios of size k, k varying from 1 to 8, 8 being the maximum number of nodes in the model proposed by GAD for malware.

A very interesting perspective would be to apply the algorithm on other models of attacks, but also to look for another algorithm in order to compare and sort the best performing. Another perspective is to implement an IDS prototype implementing the model of Gad for malware and the solution presented in this article and compare it with other IDS.

References

[1]Mohammed Saber, Toumi Bouchentouf, Abdelhamid Benazzi and Mostafa Aziz "Amelioration of Attack Classifications for Evaluating and Testing Intrusion Detection System" *Journal of Computer Science 6(7): 716-722 , 2010*

[2] Mohammed Saber, Toumi Bouchentouf, Abdelhamid Benazzi and Mostafa Azizi, "ATTACKS CLASSIFICATION FOR EVALUATING INTRUSION DETECTION SYSTEM", IADIS International Conferences Informatics 2010, Wireless Applications and Computing 2010 and Telecommunications, Networks and Systems 2010, page 166 – 170.

[3] Mohammed S. Gadelrab, Anas Abou El Kalam and Yves Deswarte, "Defining categories to select representative attack test-cases", *Proceedings of the 2007 ACM workshop on Quality of protection (QoP '07)*, Alexandria, Virginia, USA, pp. 40-42, 2007.

[4] Mohammed S. Gadelrab, Anas Abou El Kalam and Yves Deswarte, "Execution Patterns in Automatic Malware and Human-centric Attacks" *NCA 2008: Proceedings of the 2008 Seventh IEEE International Symposium on Network Computing and Applications 29-36.*

[5] Classification Tree Method 2010 : http://www.systematic-testing.com.

[6] Classification Tree Editor 2010 : http://www.systematic-testing.com.

[7]Mitres Common Malware Enumeration list http://cme.mitre.org/

[8] L. Chen, "Modeling Distributed Denial of Service Attacks and Defenses", *PhD thesis*, Carnegie Mellon University, USA, 2003.

[9] S. Cheung, U. Lindqvist and M. Fong, "Modeling Multistep Cyber Attacks for Scenario Recognition", *Proceedings of the Third DARPA Inofrmation Survivability Conference and Exposition (DISCEX III)*, Washington, DC, USA, pp. 284-292, 2003.

[10]Ashish Garg, Shambhu Upadhyaya and Kevin Kwiat, "Attack Simulation Management for Measuring Detection Model Effectiveness", *Proceedings of The Second Secure Knowledge Management Workshop (SKM 2006)*, Brooklyn, NY, USA, 2006.

[11]T. Tidwell, R. Larson, K. Fitch and J. Hall, "Modeling Internet Attacks", *Proceeding of the IEEE Workshop on Information Assurance and Security*, West point, NY, USA, pp. 54-59, 2001.

[12] B. Schneier, "Attack Trees: Modeling Security Threats", *Dr. Dobb's Journal*, Vol. 24, Number 12, pp. 21-29, 1999.

[13] O. Sheyner, J. Haines, S. Jha, R. Lippmann and J. M. Wing, "Automated generation and analysis of attack graphs", *Proceeding of 2002 IEEE Symposium on Security and Privacy*, Oakland, California, USA, pp. 273-284, 2002.

[14] J. Steven Templeton and Karl Levitt, "A requires/provides model for computer attacks", *Proceedings of the 2000 workshop on New security paradigms (NSPW '00)*, NY, USA, pp. 31- 38, 2000.
[15]J. P. McDermott, "Attack net penetration testing", *Proceedings of the 2000 workshop on New security paradigms (NSPW '00)*, New York, USA, pp. 15-21, 2000.
[16] Ole Martin Dahl and Stephen D. Wolthusen, "Modeling and Execution of Complex Attack Scenarios using Interval Timed Colored Petri Nets", *Proceedings of the Fourth IEEEInternational Workshop on Information Assurance (IWIA '06)*, Washington DC, USA, pp. 157- 168, 2006.
[17]Marc Dacier and Yves Deswarte, "Privilege Graph: an Extension to the Typed Access Matrix Model", *Proceedings of the Third European Symposium on Research in Computer Security(ESORICS '94)*, London, UK, pp. 319-334, 1994.
[18]Mohamed Kaaniche, Y. Deswarte, Eric Alata, Marc Dacier and Vincent Nicomette, "Empirical analysis and statistical modeling of attack processes based on honeypots", *Proceeding of Workshop on Empirical Evaluation of Dependability and Security (WEEDS), DSN'2006*,Philadelphia, USA, pp. 119-124, 2006.
[19][MOZART 2010] : http://www.mozart-oz.org.
[20] http://jacop.cs.lth.se/
[21]http://www-01.ibm.com/software/integration/optimization/cplex-cp-optimizer/
[22] http://www.emn.fr/z-info/choco-solver/
[23] http://www.zkoss.org/

Mohammed SABER engineer in networks and system at the Computer Science Department, National School of Applied Sciences, Mohammed First University Oujda (Morocco), doctoral researcher at the MATSI laboratory (Applied Mathematics, Signal Processing and Computer Science Laboratory) in the field of computer security. I am interested in intrusion detection system and attacks

Toumi BOUCHENTOUF Professor in computer science, responsible for the computer engineering department. from the National School of Applied Sciences, Mohammed First University Oujda (Morocco). Interested in software engineering, management and monitoring of IT project, computer security in intrusion detection system and attacks, and multi-agent system.

Mohammed Ghaouth BELKASMI Engineer in computer science, graduated, in 2004, from the National School of Applied Sciences, Oujda, Morocco. Head of it department of the Mohammed First University, Oujda, Morocco, since 2006. Doctoral researcher at the Applied Mathematics, Signal Processing and Computer Science Laboratory (MATSI), Mohammed First University, Oujda, Morocco. Interested in software engineering, management and monitoring of IT project.