World Academy of Science, Engineering and Technology
International Journal of Aerospace and Mechanical Engineering
Vol:1, No:12, 2007

DEVS Modeling of Network Vulnerability

Hee Suk Seo, and Tae Kyung Kim

***Abstract*—**As network components grow larger and more diverse, and as securing them on a host-by-host basis grow more difficult, more sites are turning to a network security model. We concentrate on controlling network access to various hosts and the services they offer, rather than on securing them one by one with a network security model. We present how the policy rules from vulnerabilities stored in SVDB (Simulation based Vulnerability Data Base) are inducted, and how to be used in PBN. In the network security environment, each simulation model is hierarchically designed by DEVS (Discrete EVent system Specification) formalism.

***Keywords*—**SVDB, PBN, DEVS, Network security.

I. Introduction

NETWORKS are developed to make computers more accessible to the outside world. Making computers more accessible to the outside is a mixed blessing. Network security is a problem that has gotten larger with the growth of the Internet [1]. For the simulation of the policy-based framework environment, ID agent models, network component models and Firewall models are constructed based on the DEVS (Discrete EVent system Specification) formalism [2]. Since evaluating the performance of a security system directly in real world requires heavy costs and efforts, an effective alternative solution is using the simulation model. In concrete terms, using the model we can build various simulation situations, perform iterative runs, and decide which security configuration is effective in meeting the change of network environment [3]. Every network is made up of a variety of elements, but they must still work together. Because of this heterogeneity and the lack of complete standardization, managing a network with more than a handful of elements can require a significant amount of expertise. The network manger is faced with the difficult task of meeting internal and external security requirements while still providing easy and timely access to network resources to authorized user. The solution to these issues lies in policy-based management. A network manager creates policies to define how resource or services in the network can (or cannot) be used. The policy-based management system transforms these policies into configuration changes and applies those changes to the network. The simplification and automation of the network management process is one of the key applications of the policy framework [2]. The paper describes the design and modeling of network security agents based on policy-based framework which has some merits. The need arises for systems to coordinate with one another, to manage diverse attacks across networks and time.

Manuscript received October 15, 2007.

H. S. Seo is with the Korea University of Technology and Education, Byungcheon, Chungnam 330-708 Korea (e-mail: histone@kut.ac.kr).

T. K. Kim is with the Seoil College of Department of Information Electronics, Seoul 77005 Korea. (corresponding author to provide phone: +82-2-490-7241; e-mail: tkkim@ seoil.ac.kr).

This work was supported by the Korea Research Foundation Grant funded by the Korean Government(MOEHRD)" (KRF-2007-331-D00450).

II. Network Model

Policy-based networking [4 is decomposed into five sub-components: PC, PR, PDP, PEP, and PMP model. Policy console model helps an administrator edit policy rules and configurations. PR model stores the policies. PDP model is decomposed into Supervision, Policy Transform, and Policy Distributor model. Policy Transform model is responsible for ensuring that the high-level policies specified by the network administrator and mutually consistent, correct, and feasible with the existing capacity and topology of the network. Policy Distributor model is responsible for ensuring that the low-level policies are distributed to the various devices in the network. PEP model is specialized into six sub-components: Gateway model, Router model, Switch model, IDS model, firewall model, and Authentication model. PMT model applies the policies that is defined by the network management policies and stores the information to the policy repository model. It is decomposed again into Resource Discovery, Model Interface, and Model Valid Check. Model Interface is specialized into Model Interface Administrator, Model Interface PR, Model Interface IDS, Model Interface Firewall, and Model Interface VDB. Vulnerability database (VDB) contains the network vulnerability. Valid Check model inspects the attributes that is set by the network management policies and checks if the new policy conflicts with the used it. Resource Discovery model determines the topology of the network, the users, and applications operational in the network.

The policy architecture as defined in the IETF consists of four basic elements. PMT (Policy Management Tool) is used by an administrator to input the different policies that are active in the network. The PMT takes as input the high-level policies that a user or administrator enters in the network and converts them to a much more detailed and precise low-level policy description that can apply to the various devices in the network. PDP (Policy Decision Point) makes decisions based on policy rules and it is responsible for policy rule interpretation and initiating deployment. Its responsibilities may include trigger detection and handling, rule location and applicability analysis, network and resource-specific rule validation and device adaptation functions. PEP (Policy Enforcement Point) is the

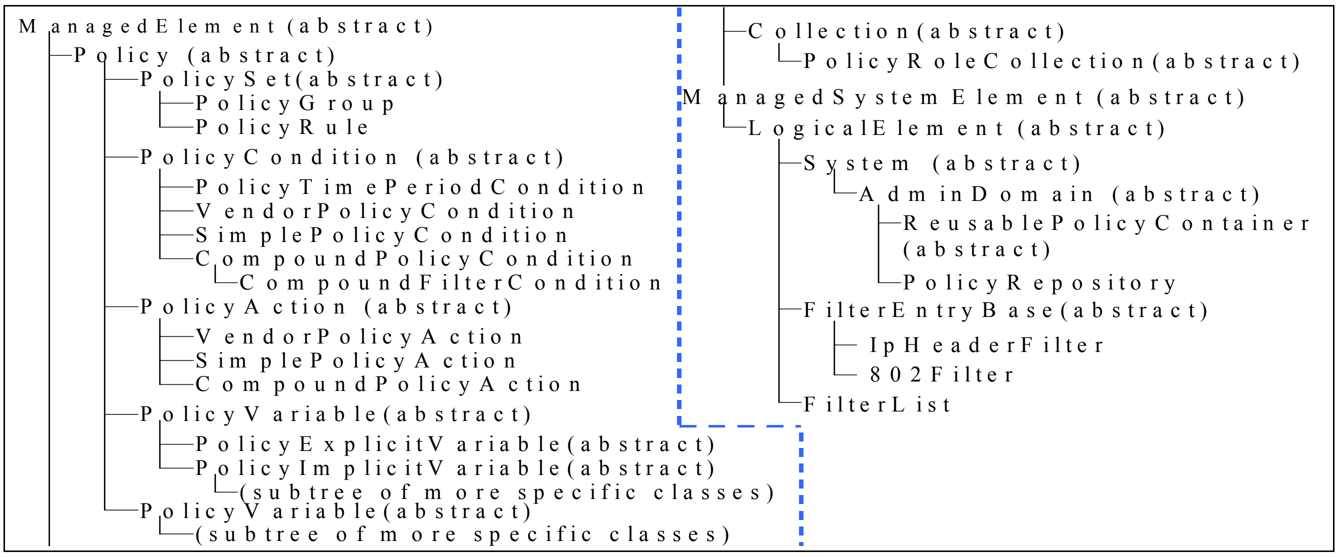

Fig. 1 Class Inheritance Hierarchy for PCIME

network device that actually implements the decisions that the PDP pass to them. The PEP is also responsible for monitoring any statstics or other information relevant to its operation and for reporting it to the appropriate places. Policy Repository is used to store the policies generated by the management tool. Either a directory or a database can store the rules and policies required by the system. In order to ensure interoperability across products from different vendors, information stored in the repository must correspond to an information model specified by the policy framework working group.

III. Network Model

The high-level and low-level policies required for network management can be specified in many different ways. From a human input standpoint, the best way to specify a high-level policy would be in terms of a natural-language input. Although these policies are very easy to specify, the current state of natural-language processing needs to improve significantly before such policies can be expressed in this manner. The next approach is to specify policies in a special language that can be processed and interpreted by a computer. When policies are specified as a computer interpretable program, it is possible to execute them. A simpler approach is to interpret the policy as a sequence of rules, in which each rule is in the form of a simple condition-action pair (in an if-then-else format). The IETF has chosen a rule-based policy representation. IETF Policy Framework WG works especially on the “condition action” part to define Policy Core Information Model [5] for the representation of policy information. The PCIM is the object-oriented information model for representing policy information. This model defines representing policy information and control of policies, and association classes that indicate how instances of the structural classes are related to each other. Policies can either be used in a stand-alone fashion or aggregated into policy groups to perform more elaborate functions. Fig. 1 illustrates the inheritance hierarchy for PCIME (PCIM extensions) [6].

A vulnerability is a condition or weakness in (or absence of) security procedures, technical controls, physical controls, or other controls that could be exploited by a threat [7]. The theme of vulnerabilities analysis is to devise a classification, or set of classifications, that enable the analyst to abstract the information desired from a set of vulnerabilities. This information may be a set of signatures, for intrusion detection; a set of environment conditions necessary for an attacker to exploit the vulnerability; a set of coding characteristics to aid in the scanning of code; or other data.

Government and academic philanthropists, and some companies, offer several widely used and highly valued announcement, alert, and advisory services for free. Each of those organizations referred to the same vulnerability by a different name. Such confusion made it hard to understand what vulnerabilities you faced and which ones each tool was looking for- or not looking for. The MITRE Corporation began designing a method to sort through the confusion. It involved creating a reference list of unique vulnerability and exposure names and mapping them to appropriate items in each tool and database. We use CVE (Common Vulnerabilities and Exposures) names in a way that lets a security system crosslink its information with other security systems [8].

A. Individual Vulnerability and Countermeasure Representation

Representation of vulnerability using AV shows a way to NSO removing vulnerability or finding work-around of the vulnerability. When we analyze the vulnerability, we define the vulnerability expression which consists of AVs and their relation.

Vulnerability expression for CVE-2003-0010 *is “(AI3-203 or AI3-201) and KM1-001 and KM2-001”*. According to the expression, we can figure out that this vulnerability could be exploited by external malicious or unexpected input.

Also, we could figure out that the external input causes overflowing of stack and execution of arbitrary codes. In this way, the vulnerability expression consists of external cause and internal causes and they works together when attacker tried to exploit the vulnerability. In the other hand, the vulnerability information represented by AVs, can be used to extract the countermeasures that disable the effectiveness of vulnerability by remove one or more AVs in vulnerability.

Especially, in our representation scheme, we are making use of the negation operator to represent the countermeasure of vulnerability as shown in below expressions.

¬{(AI3-203 or AI3-201) and KM1-001 and KM2-001} ➔
(¬AI3-203 and ¬AI3-201) or ¬KM1-001 or ¬KM2-001

The above expression shows that if we want to prevent from exploiting vulnerability, we need to protect malformed two kind of external input, or protect the memory overrun using memory protection mechanism, or disable the execution of arbitrary code from updated return address.

Example A. Buffer Overflow Vulnerability
The CVE-2003-0010 can be analyzed as shown in below box.

- ■ CVE-2003-0010's AV based Analysis Result
  - ■ Vulnerability Expression : (AI3-203 or AI3-201) and KM1-001 and KM2-001
    - ■ AI3-203 : no check input_message_HTML
    - ■ AI3-201 : no check input_message_email
    - ■ KM1-001 : can not prevent from altering return address
    - ■ KM2-001 kernel could not check execution from updated memory.Stack Overflow

B. Network-wide Vulnerability and Countermeasure Representation

To illustrate the network-wide case, we will aggregate the vulnerabilities in some systems. For each system, we can generate countermeasure expressions as same way that we see in system-wide case.

System 1:

	CVE ID	Bugtraq ID	Category
1	CVE-2002-0392	5033	Buffer Overflow
2	CAN-2005-1208	13942	Buffer Overflow
3	CAN-2004-0081	9899	Unexpected Condition
4	CVE-2000-0222	990	Configuration
5	CAN-2002-1117		Configuration
6	CAN-2003-0543, CAN-2003-0544, CAN-2003-0545	8732, 13359	Buffer Overflow

System 2:

	CVE ID	Bugtraq ID	Category
1	CVE-2000-0222	990	Configuration
2	CAN-2002-1117		Configuration
3	CAN-2003-0543, CAN-2003-0544, CAN-2003-0545	8732, 13359	Buffer Overflow
4	CAN-2005-1794	13818	Information Disclosure
5	CVE-2001-0540	3099	DoS
6	CAN-1999-0621		Information Disclosure

SVDB has the specific information that can be used by security agents as well as the common information of vulnerability of system. SVDB has four components; vulnerability information, packet information, system information and references information. SVDB also has particular parts for accuracy and efficiency of security agents. The payload size is used to test the packet payload size. The offset modifies the starting search position for the pattern match function from the beginning of the packet payload. The payload size and offset have the added advantage of being a much faster way to test for a buffer overflow than a payload content check. URL contents allow search to be matched against only the URL portion of a request. Table I shows the table of SVDB for the simulation [9].

TABLE I
TABLE OF SVDB

Table	Field
Vulnerability Information	Vulnerability Name(CVE), Summary, Published, Vulnerability Type, Exploitable Range, Loss Type, Vulnerable Software and Versions
Packet Information	IP flags, TTL, Protocol, Source IP, Destination IP, IP options, ICMP code, ICMP type, Source port, Destination port, Sequence number, Acknowledgement number, TCP flag, Offset, Payload size, URL contents, Contents, CVE Name
System Information	Vulnerable Software and Versions, Vendor, Name, Version
References Information	Source, Type, Name, Link

IV. System Modeling

Policy Framework model is divided into PMT model and PDP model. PMT model is composed of Resource Discovery model and Validity Check model. Resource Discovery model determines the topology of the network, the users, and applications operational in the network. In order to generate the configuration for the various devices in the network, the capabilities and topology of the network must be known. Validity Check model consists of various types of checks: Bounds checks, Consistency checks, Feasibility checks.

PDP model is composed of Supervision model, Policy Transformation model and Policy Distributor model. Supervision model receives events from network devices and monitors network usage. The PDP can use this information about the network to invoke new policy-based decisions. Policy Transformation model translates the business-level policies into technology-level policies that can be distributed to the different devices in the network. Policy Distributor model is responsible for ensuring that the technology-level policies are distributed to the various devices in the network.

IDS model is divided into Detector model, Response Generator model and Logger model. Detector model is further decomposed into Pattern Matcher model and Analyzer model. Pattern Matcher model is a rule-based expert system that detects intrusions through pattern matching procedure with packet data and rules. Analyzer model is a statistical detection engine that detects intrusions by analyzing system log and audit. Response Generator model determines a response according to the detection result of Detector model and sends a message. Logger model records all information of detection procedure in the log file.

Policy server reports the information which is defined by the policy-based framework to other network components. Firewall uses this information to prevent harmful packets from external network. Intrusion detection agent of each subnet detects the intrusion and reports the intrusion information to the policy-based framework. The policy management tool in the common policy-based framework is used by the network administrator but we have appended the a few interface modules for the more automatic control. The policy management tool in the proposed system is accessed by the administrator, vulnerability database, and intrusion detection system. Vulnerability database helps the system manager to find bugs that enable users to violate the network security policies. The policy server reports to the firewall for the prevention from damaging the network.

V. Conclusion

We presented a policy-based network simulation environment for the security system. The security system makes a various network situations-the policies should be applied to change the network states. These situations include the response of intrusion detection system and policy change by the firewall, etc. Policy-based network management provides a means by which the administration process can be simplified and largely automated. The proposed system has an advantage of the management. The administrator can easily apply the policies to the network components with the policy-based framework. The security system makes a various network situations-the policies should be applied to change the network states. These situations include the response of intrusion detection system and policy change by the firewall, etc. Policy-based framework supports the automatic and flexible environment for changing the network situation. We also proposed the structure for policy rule induction form vulnerabilities stored in SVDB.

References

[1] C. M. King, C. E. Dalton, T. E. Osmanoglu, Security Architecture, RSA press, 2001.
[2] B. P. Zeigler, H. Praehofer and T.G. Kim, Theory of Modeling and Simulation, Academic Press, 2000.
[3] Seo, Hee Suk and Cho, Tae Ho, "An application of blackboard architecture for the coordination among the security systems," Simulation Modelling Practice and Theory, Elsevier Science B.V., Vol. 11, Issues 3-4, pp. 269-284, Jul. 2003.
[4] R. Bace, Intrusion Detection, Macmillan Technical Publishing, 2000.
[5] F. Cohen, "Simulating Cyber Attacks, Defences, and Consequences," Computer & Security, Vol.18, pp. 479-518, 1999.
[6] Dinesh C. Verna. Policy-Based Networking: Architecture and Algorithm, New Rider, 2001.
[7] Dave Kosiur. Understanding Policy-Based Networking, John Wiley & Sons, Inc. 2001.
[8] B. Moore, et al., "Policy Core Information Model-Version 1 Specification," IETF RFC 3060, Feb 2000.
[9] E. D. Zwicky, S. Cooper and D. B. Chapman, Building Internet Firewalls second edition, O'reilly & Associates, 2000.