

A Computer Network Defense Policy Refinement Method

Zhao Wei¹, Yanli Lv², Chunhe Xia¹, Yang Luo¹, and Qing Wei¹

¹Key Laboratory of Beijing Network Technology

School of Computer Science and Engineering, Beihang University, Beijing, China

²Information Center of Ministry of Science and Technology, The Ministry of Science and Technology of the People's Republic of China, Beijing, China

wz@cse.buaa.edu.cn, lvy1@most.cn, xch@buaa.edu.cn,
veotax@sae.buaa.edu.cn, wq2012_buaa@163.com

Abstract. The existing methods of policy refinement in computer network defense (CND) can only support the refinement of access control policy, but not the policies of protection, detection, response, and recovery. To solve this problem, we constructed a computer network defense policy refinement model and its formalism specification. An algorithm of defense policy refinement is designed. At last, the effectiveness of our methods was verified through one experiment cases of the composition policies with intrusion detection, vulnerabilities detection, and access control.

Keywords: computer network defense, formalism specifications, policy refinement, semantic consistency.

1 Introduction

The growing network information system and the emergence of new technology such as cloud computing and big data have brought up a huge challenge to the efficiency and accuracy of the network management. It is time-consuming and easy to make mistakes for the traditional manual network security management in the large-scale network system. In order to solve these problems, researchers have proposed policy-based architecture [1], policy-driven management methods [2] to simplify the management for the complicated and distributed network system, such as cloud framework [3]. Administrator may specify the targets and constraints only in the form of policy. A policy can be defined as a set of rules. These rules are used to express how to reach a desired behavior. Policy refinement can complete this process. Policy refinement is a process of transforming high-level abstract policy to low-level concrete ones[4].

Because of the complicated policy refinement process and manual operation for some refinement process, some researchers have proposed automatic policy refinement methods in different fields, such as policy refinement in usage control policies [5]. Reference [6-7] discussed policy refinement methods in the network security management. However, these policy refinement methods only support the refinement of access control policies instead of defense policy, such as detection, recovery policy, et al.

Computer network defenses are actions through the use of computer networks to protect, monitor, analyze, detect and respond to unauthorized activities within Department of Defense information systems and computer networks [8].

It remains unclear about how to expand the policy refinement methods to computer network defense field in order to support the refinement of defense policy including protection, detection, response, and recovery. Based on this problem, we have proposed a computer network defense policy refinement method. A formalism model of policy refinement is provided. This model supports the refinement of four types of defense policies including protection, detection, response, and recovery policies. At last, we designed an algorithm of defense policy refinement and the effectiveness of the methods we proposed is verified through two experiments.

The rest of this paper is organized as follows. Section 2 gives related works of policy refinement. CND policy refinement model and its formalism specification are provided in Section 3. A CND policy refinement algorithm is designed in Section 4. Section 5 gives the experiment analysis and verification for CND policy refinement. Finally, Section 6 concludes the paper.

2 Related Works

Automatic policy refinement methods simplify security service management in complex network environment. Previous researchers have proposed various policy refinement methods for the network security management. These methods are shown as follows:

Reference [9] proposed a policy refinement method that can get action sequence to achieve high-level goal based on event-calculation and abductive reasoning.

Reference [10] proposed a policy refinement method based on MBM model. Reference [11] proposed a model-based refinement of security policy method in the collaborative virtual organization. This model-to-model transformation technique can transform XACML-based VO policy to the resource level. Reference [12] shows a security policy refinement framework in the network environment. This framework includes a three-level model. The top level of RBAC model is used to express security goals. The middle level of the network security tactics model is used to express the constraints of data stream. The bottom of the model is an abstract view oriented towards the technical capacity. At last, the implementation of this model is realized within the framework of CIM/WBEM. Reference [13] proposed a policy refinement method based on event-B. The policy has four levels including user-service level, process-terminal service level, host-port level, and interface-port level. Reference [14] extended OrBAC model and proposed a policy refinement method transforming a high-level security policy into low-level security mechanism. An example is provided to verify the effectiveness of the method.

In conclusion, most of the policy refinement methods [12-14] only support the refinement of access control policy instead of defense policies such as protection and detection policies, et al. Reference [9] can support policy refinement of network management. However, they cannot support defense policy refinement from the perspective of computer network defense.

Based on the existing policy refinement methods and the characteristics of computer network defense, we proposed a computer network defense policy refinement method. A formalism policy refinement model is provided. Compared with other policy

refinement methods, our method not only supports the refinement of access control policy, but also the defense policy.

3 Computer Network Defense Policy Refinement Model

Computer network defense policy refinement is a process transforming goal-level (high-level) defense policy goals to operational-level (low-level) defense policies.

Defense policy refinement model has two levels: Goals level and Operational level. The elements of operational level are refined by goal level. So it forms a hierarchical structure form high-level to low-level. The policies of goal level express the high-level security requirements and defense goals. The policies of operational level express the operational actions related to concrete network environment.

Definition Defense Policy Refinement Model: The defense policy refinement model consists of elements at both goal level and operation level as well as defense policy goals, operational-level defense policies and refinement relations among elements at these two levels. We can conclude that the elements of operational-level are refined by goal-level. So it forms a hierarchical structure form high-level to low-level. The formalism of this model is shown as follows:

$$\left\{ \begin{array}{l} M ::= (G, O, R), \\ G = \{ \text{Domain, Role, Target, Activity, Means, ContextType, MeansConstraints} \} \\ O = \{ \text{SNode, User, TNode, Resource, Action, DefenseAction,} \\ \quad \text{DefenseEntity, Context, PolicyRelation} \} \\ R \subseteq G \times O; HPR \subseteq G \times G; LPR \subseteq O \times O; \\ HPGOAL ::= (G, HPR); LPOPERATION ::= (O, LPR); \end{array} \right.$$

Wherein, G denotes the set of elements of the goal level, O the set of the operational level, R the set of the refinement relations between the elements of goal level and operational level. HPGOAL the set of the policy goals which consists of goal-level elements and theirs relations, LPOPERATION the set of the operational-level policy which consists of operational-level elements and theirs relations

The meaning of the elements of the goal level are explained as follow:

Domain: It denotes a scope or area. Domain can be divided depending on the environment of network such as organization structure, geographical boundary, security level, and management responsibility. It is shown as a hierarchical structure.

Role: It is a set of users who share common characteristics.

Target: It is a set of resources with common characteristics. Target is divided into four classes such as data, operation system, application programs, and services.

Activity: It is a set of actions with common characteristics. It is divided two classes including activities of local process with configuring, acquiring and operating and activities of interact process with accessing and transferring.

Means: Means is a set of defense activities. According to the model of PDRR, means are divided into four classes such as protection (including the permission access control and the denying access control, user authentication, encryption communication, backup), detection (including intrusion detection, vulnerabilities detection), response (including access control, system rebooting and system shutdown), and recovery (including rebuild and making patch).

MeansConstraints: Meansconstraints means time series and logic relations between defense means including sequence and, sequence or, parallel and, parallel or. They are shown as follow:

$$R_{means} = \{r_{seq_and}, r_{seq_or}, r_{concu_and}, r_{concu_or}, r_{xor}\}$$

In brief, we assume that there are only two means in one composition policy goal. $Means = \{mean_1, mean_2\}$. Each relation is explained separately as follows:

r_{seq_and} : If $seq_and(mean_1, mean_2)$, it denotes that the $mean_1$ is executed first. If the executing effect of $mean_1$ is true, the $mean_2$ is executed. Only if both means are successfully completed can we say the policy goal is completed successfully.

r_{seq_or} : If $seq_or(mean_1, mean_2)$, it denotes that the $mean_1$ is executed first. If the executing effect of $mean_1$ is true, the $mean_2$ does not need to be executed. If the executing effect of $mean_1$ is false, the $mean_2$ must be executed. Whether the policy goal is completed successfully or not depends on the success of $mean_1$ or $mean_2$.

r_{concu_and} : If $concu_and(mean_1, mean_2)$, it denotes that both $mean_1$ and $mean_2$ are executed at the same time. If the effects of $mean_1$ and $mean_2$ are true, we can say that the policy goal is successfully accomplished.

r_{concu_or} : If $concu_or(mean_1, mean_2)$, it denotes that both $mean_1$ and $mean_2$ are executed at the same time. Only if there is a true executed effect between $mean_1$ and $mean_2$, can we say that the policy goal is successfully accomplished.

r_{xor} : If $xor(mean_1, mean_2)$, it denotes that there exists one executing means between $mean_1$ and $mean_2$. Whether the policy goal is completed successfully depends on the true effect of $mean_1$ or $mean_2$.

ContextType: It is a set of contexts with common characteristics. It is divided into two classes including vulnerability ct_{vul} and event ct_{event} .

The meaning of the elements at the operational level are explained as follow:

SNode: It denotes a host node in which a user initiates an operation to resource.

TNode: It denotes a host node in which the resource exists.

User: It denotes people who can initiate an operation.

Resource: It denotes an entity that needs protection, such as the instance of the data, operation system, service, application program, and data.

Action: It denotes a change that cannot be subdivided, such as the actions of adding, deleting, and changing corresponding to operating activity; the actions of sending, receiving, requesting and replying corresponding to transferring activity.

DefenseEntity: It means security device that can executed as defense action. It denotes in device number. Defense entity includes firewall $de_{firewall}$, IPsec VPN de_{ipsec_vpn} , backup server de_{backup_server} , system management server $de_{sysmanage_server}$, IDS $de_{intrude_detect}$, vulnerabilities scan server de_{vul_server} .

DefenseAction: It is an atomic defense action leading to state change. Defense actions include permit action da_{permit} and deny action da_{deny} of firewall, the permission Encryption action $da_{\text{permit_crypt}}$ of IPsec VPN, backup action da_{backup} and rebuild action da_{rebuild} of backup server, user authentication action $da_{\text{authenticate}}$, making patch action $da_{\text{makepatch}}$ and rebooting and shutdown action of system management server, alerting action da_{alert} of IDS, scan action da_{vulscan} of vulnerability scan server.

PolicyRelation: It denotes time series and logic relations among operational-level policies including sequence and, sequence or, parallel and, parallel or. It is equivalent to the means relation.

Context: It means a concrete environment in which we can deploy some means in domain, such as a concrete vulnerability (its vulnerability number is CVE-2002-0073), intruding event (DoS attacking).

The refinement relations of elements between goal-level and operational-level are defined as follows:

$$R = \{RS, RU, RT, RR, RA, RDD, RC, RP\}$$

$RS \subseteq \text{Role} \times \text{Domain} \times \text{SNode}$, representing refinement from role that belongs to a domain of source node;

$$RU \subseteq \text{Role} \times \text{User}$$
, representing refinement from the role of user;

$RT \subseteq \text{Target} \times \text{Domain} \times \text{TNode}$, representing refinement from target that belongs to a domain of target node;

$$RR \subseteq \text{Target} \times \text{Resource}$$
, representing refinement from target to resource;

$$RA \subseteq \text{Activity} \times \text{Action}$$
, representing refinement from activity to action;

$RDD \subseteq \text{Means} \times \text{DefenseAction} \times \text{DefenseEntity}$, representing refinement from defense means to defense action and defense entity;

$RC \subseteq \text{ContextType} \times \text{Context}$, representing refinement from context type to context ;

$RP \subseteq \text{MeansConstraint} \times \text{PolicyRelation}$, representing refinement from MeansConstraints to PolicyRelation; In this paper, PolicyRelation is equivalent to MeansConstraints.

4 The Algorithm of Computer Network Defense Policy Refinement

Based on our computer network defense policy refinement model, we first constructed a defense policy refinement repository that includes network situation information and refinement rules. Then, we designed a CND policy refinement algorithm combined with defense policy refinement repository.

1. Repository of CND policy refinement. The repository includes network situation information and policy refinement rules. They are created with MySQL database.

Network situation information includes domain information that divides organization and forms a hierarchy structure; it also includes nodes information that describes the

name, ID of nodes, user, and resources; linking relations among nodes that is constructed with adjacent matrix; roles information that describes the name, ID of roles and the domain of role; targets information that describes the name, ID of targets and the domain of target; defense entities information that describes the name, ID of defense entities, defense action; defense means that describes the name, ID of defense means; means relations among means; context type that describes the vulnerabilities and events.

Refinement rules describe the refinement relations between the elements of goal-level and operational-level. Refinement rules include the role-user rules that specify the refinement relations between role and user; the domain-node rules that specify the refinement relations between domain and node; the activity-action rules that specify the refinement relations between activity and action; context rules that specify the refinement relations between context type and context; means-defense entity rules that specify the refinement relations between means and defense entity.

2. The process of CND policy refinement algorithm.

We have designed a description language CNDIDL [15] for the CND policy goal. A scanning method was devised based on the lexical and syntax rules to decompose the defense policy goal described by CNDIDL and stored into the memory data structure. After the decomposition, we can transform a CND policy goal to one or more operational-level defense policies through policy refinement repository.

The process of transformation algorithm is shown as follow:

(1) At first, we used each defense means in the list of defense to estimate which type of defense policy goal it is. Based on the goals of protection (access control, user authentication, encryption communication, backup, patch making), detection (intrusion detection and vulnerabilities detection), response (rebooting, shutdown and the adding of access control rules) and recovery (rebuild), we completed the CND policy refinement with corresponding refinement algorithm. Now, we take the policy goal refinement of access control for example.

(2) According to defense means, we derived a type of defense entity-firewall through looking into the table of means-defense entity. In addition, in order to get the instance of defense entity to execute operational-level access control policy, we would first find a set of simple paths from source node to destination node. Simple path is a node sequence in which there is not a same node. For the permission policy, we would choose all firewalls in these paths. For the denial policy, we would choose the nearest firewalls from the source node in these paths. The pseudocode of algorithm of getting simple path set between source node and target node is shown as follow:

```

1 Algorithm GETSimplePathSet
2 INPUT : InitialNode : u, TargetNode : v, d = -1
3 OUTPUT : The set of simple path between node of u and v : PathSet
4 Procedure GetSimplePathSet(u, v, d)
5   d + 1 ← d;
6   visited[u] = true;
7   path[d] = u;
8   IF (u == v) THEN
9     FOR (i = 0) TO (d + 1) DO
10    PATH ← path[i]; // put all nodes into set PATH between u and v
11  REPEAT
12 PathSet ← PathSet ∪ PATH; // get the set of all simple paths

```

```

13 ELSE THEN
14   N = getAdjVetxSet(u); // get adjacent node set of u
15   WHILE(n = GetFirstNode(N))
16     IF(!visited[n]) THEN
17       GetSimplePath(n,v,d);
18     END IF
19     n ← GetNextNode(N);
20   END WHILE
21 ENDIF
22END GetSimplePathSet

```

In this algorithm, we used an undirected graph to express the connecting relation between nodes for network topology and used an adjacency list to store an undirected graph. The time complexity of the algorithm is $O(n+e)$. wherein, n denotes the number of vertex in undirected graph, e denotes the number of edge in undirected graph.

In choosing instance of other defense entity (such as IDS, system management server...et.al), we would choose the nearest defense entity for the protection resource.

(3)According to the role, we can derive a set of users by looking into the table of role-user. Then we can get the set A of nodes by looking into table of node information. According to the domain, we can derive a set B of nodes by looking into the table of domain-node. At last, we can get the set C of node by operation of $A \cap B$ and get the corresponding IP address for these nodes by looking into the table of node information. In the same way, according to the target, we can derive a set of resources by looking into the table of target-resource. Then we can get the set A of nodes by looking into table of node information. According to the domain, we can derive a set B of nodes by looking into the table of domain-node. At last, we can get the set C of node by operation of $A \cap B$ and get the corresponding IP address and port number for these nodes by looking into the table of node information.

(4)According to the activity, we can derive a set of actions by looking into the table of activity-action.

(5)We can get an operational-level policy for firewall through the composition of the source IP address, the target IP address, port number, the set of actions and defense action.

(6)If more than two defense means exist in the list of defense means, we would get the next defense means and repeat the operation of (3)~(6) until all the means were processed. Then we get the means constraints and transform them to the relations of operational-level policies.

The pseudocode of transformation algorithm is shown as follows:

```

1Algorithm CNDPolicyTransformation
2INPUT : CNDPolicyGoal : G = {g1, g2, ..., gn}; Re finementRule : R
3OUTPUT : CND Operational Policy and Policy relations Set : O
4Procedure PolicyTransform(G,R)
5 // find the set of means gi in a policy goal G and
6 assign all elements of the gi to the set M //
7 M = findelement(G, Means, gi);

```

```

8 // find the set of means constraint  $s_{g_i}$  in a policy goal  $G$  and
9 assign all elements of the  $g_i$  to the set  $MC$  //
10  $MC = \text{findelement}(G, \text{MeansConstraint } s, g_i);$ 
11  $G = G - g_i - s_{g_i};$ 
12  $1 \leftarrow x;$ 
13 WHILE( $m_x \in M$ )
14   FOR ( $j=1$ ) TO ( $n-2$ ) DO
15     // Base on  $R$ , get the elements of operational level corresponding
16     to the elements of goal level //
17      $O^x \leftarrow g_j$ 
18   REPEAT
19      $O^x = \sum_j^1 \cup O_j^x$  // the corresponding operational level policy for the  $x$ th means //
20      $x \leftarrow x+1;$ 
21   REPEAT
22 // refinement from means constraint  $s$  "MC" to operational level policy relation "PR" //
23  $PR \leftarrow MC$ 
24  $O = \sum_{k=1}^{|M|} \cup O^k \cup PR$ 
25 END Policy Re fine

```

The time complexity of the algorithm is $O(s \bullet m)$. wherein, n denotes the number of means in one policy goal, m denotes the number of elements in one policy goal.

5 The Experiment

In this section, we provide some examples to illustrate the effectiveness of policy refinement through our experiments.

5.1 Experimental Environment

Experiment goal: In order to test the validity of our refinement methods, we use CNDIDL[15] to describe one or more high-level defense policy goals. These high-level defense policy goals can be transformed to operational-level defense policies automatically with ours policy refinement method and the generated operational-level policies would be simulated in simulation platform GTNeTs in which the defense effect can be observed.

Network topology environment is shown in Fig. 1. The whole network is divided into three main parts: external network, DMZ, and internal network. DMZ includes Web server, DNS server, FTP server, and SMTP server (Corresponding IP addresses are 192.168.1.4/24, 192.168.1.5/24, 192.168.1.3/24, and 192.168.1.2/24.). The internal network is partitioned into two segments by switcher, i.e. Net 1 and Net 2. There are three hosts and one system management server (IP:192.168.2.2/24)in Net 1; one host and one Database Server (IP :192.168.3.2/24) in Net 2. There exist vulnerabilities in hosts and servers of DMZ and internal network (They are shown in Table 2). By exploiting these vulnerabilities, the attacker gains root access and brings about DoS attack.

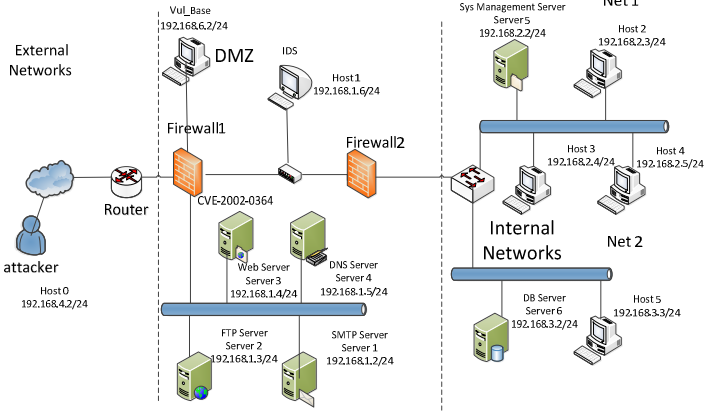


Fig. 1. Network topology

5.2 Experiment Verification and Analysis

The refinement of composition policies including intrusion detection, vulnerabilities detection, and access control.

Scenario: It was assumed that the attacker Host0 (IP Address 192.168.4.2) can access FTP server in DMZ, and bypass the Firewall2 and access DB server6 to conduct a DoS attack according to the configuration vulnerability of firewall. We deployed IDS, Vulnerability scan and Firewall1 to protect database server that provides services, when Dos attack was detected. The high-level policy goals are described by CNDIDL as follow:

```
PolicyGoal1{Extranet,Unauthorized(user),{DMZ,Net2},{FTPservice,DBservice},
TCP access,{int rusion _ detect,vul _ scan,-access _ control},
{seq _ and(int rusion _ detect,vul _ scan),seq _ and(vul _ scan,-access _ control)}}
```

The description text for high-level policy goals includes a composition policy goal that describes intrusion detection and access control for FTP server2 and DB server6. Our policy refinement methods were used to transform high-level defense policy goals to operational-level defense policies. The operational-level defense policies are shown as follow:

```
PolicyOperation1{
IDS1(alert TCP 192.168.4.2/24 192.168.1.0/24 21,"bufferoverflow";
alert TCP 192.168.4.2/24 192.168.3.0/24 1521,"dos");},
PolicyOperation2{Vul _ Base(scan 192.168.1.0/24;scan 192.168.3.0/24);},
PolicyOperation3{firewall 1
(deny TCP 192.168.4.2/24 192.168.1.3/24 21;
deny TCP 192.168.4.2/24 192.168.3.2/24 1521;);inpara : {int erface : 4}},
policy _ relations : seq _ and(1,2);seq _ and(2,3)
```

The description text for operational-level policy includes three operational policies and relation of “sequence and”. These operational policies specify the availability requirements that guarantee that the server can provide services under the attacks of

“bufferflow” and “dos”. Wherein, when IDS1 detects the attacks including “bufferflow” and “dos”, IDS1 sends information to the vulnerability server where the distributed vulnerabilities detection software is installed. The vulnerability server is informed of vulnerability CVE-2002-0509 and CVE-2002-0037 and it sends alert information subsequently. Then, the firewall interface of IP 192.168.1.5 will add a denying rule. The running effectiveness in simulation platform GTNetS is shown in the Fig.2. The yellow packet in the circle denotes enquiring packet from IDS to vulnerability library.

The vulnerability server queries the database and affirms this attack. Then it sends affirmed information to IDS. In Fig. 3, the gray packet in the circle denotes affirmed packet from vulnerability library to IDS.

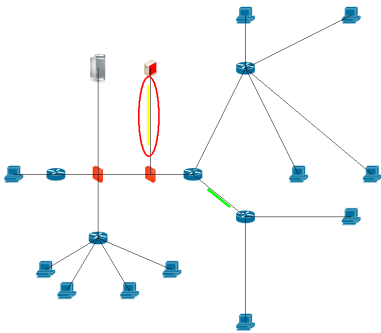


Fig. 2. Inquiring message

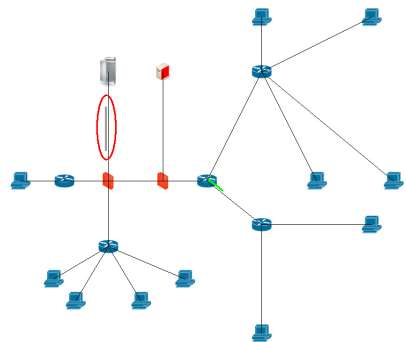


Fig. 3. Affirming message

In Fig. 4, the red packet in the circle denotes that IDS informs the firewall1 to forbid the unlawful access after receiving the vulnerability affirmed information.

In this way, the packet of attacker cannot bypass the firewall1. It is shown in Fig. 5. The control platform results of the packet denying from attacker are shown in Fig. 6.

Based on this experiment, we find that IDS, Firewall, and vulnerability server execute policy correctly. When there is some vulnerability in the network, the linkage of security equipment can complete network defense effectively. And there is no need for human interference.

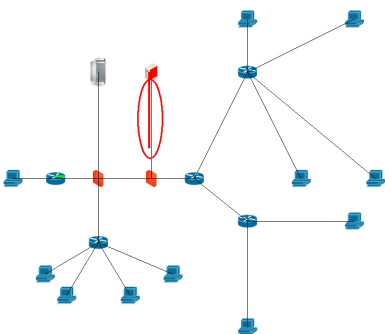


Fig. 4. Denying packet message

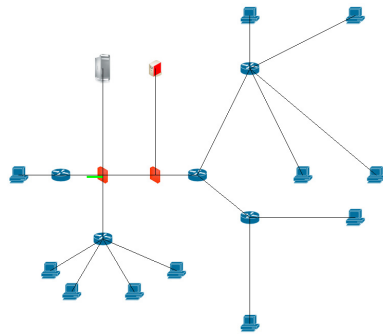



Fig. 5. The packet denied from attacker



```
IP access list 4
the packet from 192.168.4.2 has been blocked
```

Fig. 6. The control platform results of the packet denying from attacker

6 Conclusions

The existing security policy refinement methods and model are not constructed on the grounds of computer network defense. Thus they can only support the refinement of access control policy, but not the refinement of defense policy, such as the policies of IDS, backup, and recovery. For this reason, we proposed a method for computer network defense policy refinement. This method extends the existing security policy refinement model and supports the refinement of defense policy such as protection, detection, response and recovery. We constructed a defense policy refinement model and its formalism description. Based on the policy refinement model, we designed an algorithm of computer network defense policy refinement. We conducted two experiments and verified the effectiveness of this method. Compared with other policy refinement methods in reference [12-14], our method not only supports the refinement of access control policy, but also the defense policy including protection (i.e., access control, user authentication, encryption communication, backup), detection (i.e., intrusion detection, vulnerabilities detection), response (i.e., system rebooting, shutdown) and recovery (i.e., rebuild, patch making).

Acknowledgment. This work is supported by the following funding sources: the National Nature Science Foundation of China under Grant No. 61170295, the Project of National ministry under Grant No.A2120110006, the Co-Funding Project of Beijing Municipal education Commission under Grant No.JD100060630 and the Project of BUAA Basal Research Fund under Grant No.YWF-11-03-Q-001.

References

1. Zeng, H., Ma, D.F., Li, Z.Q., Zhao, Y.W.: A Policy-Based Architecture for Web Services Security Processing. In: 2012 IEEE Ninth International Conference on e-Business Engineering (ICEBE), pp. 163–169. IEEE Press (September 2012), doi:10.1109/ICEBE.2012.35
2. Loyall, J.P., Gillen, M., Paulos, A., et al.: Dynamic policy-driven quality of service in service-oriented information management systems. *Software-Practice & Experience* 41(12), 1459–1489 (2010), doi:10.1109/ISORC.2010.13
3. Luo, X., Song, M., Song, J.: Research on service-oriented policy-driven IAAS management. *The Journal of China Universities of Posts and Telecommunications* 18, 64–70 (2011), doi:10.1016/S1005-8885(10)60208-7
4. Moffett, J.D., Sloman, M.S.: Policy hierarchies for distributed systems management. *IEEE Journal on Selected Areas in Communications* 11(9), 1404–1414 (1993), doi:10.1109/49.257932

5. Kumari, P., Pretschner, A.: Deriving implementation-level policies for usage control enforcement. In: Proceedings of the Second ACM Conference on Data and Application Security and Privacy, pp. 83–94. ACM Press (2012), doi:10.1145/2133601.2133612
6. Basile, C., Liroy, A., Vallini, M.: Towards a Network-Independent Policy Specification. In: 2010 18th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP 2010), pp. 649–653. IEEE Press (February 2010), doi:10.1109/PDP.2010.45
7. Maity, S., Ghosh, S.K.: Enforcement of access control policy for mobile ad hoc networks. In: Proceedings of the Fifth International Conference on Security of Information and Networks (SIN 2012), pp. 47–52. ACM Press (2012), doi:10.1145/2388576.2388582
8. Department of Defense. JP3-13: Information Operations. US Government printing, Washington, DC (February 2006)
9. Bandara, A.K., Lupu, E.C., Moffett, J., Russo, A.: A goal-based approach to policy refinement. In: Fifth IEEE International Workshop on Policies for Distributed Systems and Networks, pp. 229–239. IEEE Press (June 2004), doi:10.1109/POLICY.2004.1309175
10. de Albuquerque, J.P., Krumm, H., de Geus, P.L., Jeruschkat, R.: Scalable model-based configuration management of security services in complex enterprise networks. *Software: Practice and Experience* 41(3), 307–338 (2011), doi:10.1002/spe.1014
11. Bryans, J.W., Fitzgerald, J.S., McCutcheon, T.: Refinement-Based Techniques in the Analysis of Information Flow Policies for Dynamic Virtual Organisations. In: Camarinha-Matos, L.M., Pereira-Klen, A., Afsarmanesh, H. (eds.) PRO-VE 2011. IFIP AICT, vol. 362, pp. 314–321. Springer, Heidelberg (2011)
12. Laborde, R., Kamel, M., Barrere, F., Benzekri, A.: Implementation of a formal security policy refinement process in WBEM architecture. *Journal of Network and Systems Management* 15(2), 241–266 (2007), doi:10.1007/s10922-007-9063-z
13. Stouls, N., Potet, M.-L.: Security policy enforcement through refinement process. In: Julliand, J., Kouchnarenko, O. (eds.) B 2007. LNCS, vol. 4355, pp. 216–231. Springer, Heidelberg (2006)
14. Hassan, A.A., Bahgat, W.M.: A Framework for Translating a High Level Security Policy into Low Level Security Mechanisms. In: 2009 IEEE/ACS International Conference on Computer Systems and Applications, pp. 504–511. IEEE Press (2009), doi:10.1109/AICCSA.2009.5069371
15. Wei, Q., Lü, L.S., Wei, Z., Wu, W.K., Xia, C.H.: CNDIDL: A CND Intention Description Language for CND Decision. In: 2012 World Congress on Information and Communication Technologies (WICT 2012), pp. 1142–1147. IEEE Press (November 2012), doi:10.1109/WICT.2012.6409246