2581-4575

International Journal For Recent Developments in Science & Technology

Crossref

A Peer Reviewed Research Journal

DESIGN OF HIGH EFFICIENT NETWORK SECURITY FOR INFORMATION STORAGE

[1]K.BHAVANI, [2]ASIA TASLEEM

[1,2]ASSISTANT PROFESSOR, DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING, NARSIMHA REDDY ENGINEERING COLLEGE, HYDERABAD TS, INDIA

[1]k.bhavani@nrcmec.org,[2]Asia.tasleem01@gmail.com

ABSTRACT

Network security management is one of the most topical concerns of information security (IS) in modern enterprises. Due to great variety and increasing complexity of network security systems (NSSs) there is a challenge to manage them in accordance with IS policies. Incorrect configurations of NSSs lead to outages and appearance of vulnerabilities in networks. Moreover, policy management is a time and resource consuming process, which takes significant amount of manual work. The paper discusses issues of policy management process in its application for NSSs and describes a policy model aimed to facilitate the process by means of specification of IS policies independently on platforms of NSSs, selection of the most effective NSSs aligned with the policies, and implementation of the policies in configurations of the NSSs.

Keywords: Information Security Policy, Policy Management Process, Network Security System, Finite Automaton, Algebra.

1 INTRODUCTION

Network security in most enterprises relies on such network security systems (NSSs) as firewalls and intrusion detection/prevention systems (IDS/IPS) [1]. However, management of NSSs faces challenges tied with time-consuming manual processes, lack of visibility in information security (IS) policies and configuration errors, which lead to network outages and appearance of vulnerabilities [2]. For instance, a policy (hereafter "policy" means "IS policy") for Check Point or Cisco firewalls may consist of thousands of rules and such complexity of policies is the main cause of configuration errors [3,4]. Thus, on the one hand, increasing number of NSSs and their increasing functionality allow to counter more threats and reduce IS risks as a result. On the other hand, complexity of NSSs' management leads to new risks and time-consuming processes, which reduce overall efficiency of NSSs utilization. Therefore, policy management process for NSSs needs simplification in order to reduce probability of errors and efforts on time-consuming tasks. A formal approach to policy modeling presented in the paper is aimed to facilitate the process by means of specification of policies independently on platforms of NSSs, selection of the most effective NSSs,

International Journal For Recent Developments in Science & Technology
2581-4575
Crossref
A Peer Reviewed Research Journal

and translation of the specified policies into configurations of the NSSs. The contributions of the paper are (a) a policy model for NSSs based on a finite automaton representation of an NSS, (b) an approach to classification of NSSs and selection of the most effective NSS, and (c) a policy algebra based on the model.

2 RELATED WORK

Models of policy management process are presented in [5, 6, 7, and 8]. All the models consider policy management as iterative process and include similar operations. The most detailed description of policy management process is presented in [5] and from the standpoint of NSSs management the following operations of the process are important. During Policy Assessment step a request for initial policy creation or update of the existing one is evaluated in order to identify policy conflicts and effects. The requested change should be made in the framework of existing IS maintenance system (ISMS includes IS management system and security tools and measures). Identification of IS threats for assets and a list of appropriate options of NSSs that counter the threats, payroll and non-payroll cost of the options as well as determination of options priority are included in Risk Assessment step. Creation of new policies or update of existing ones proceeds during Policy Development step.

Requirements for ISMS are derived upon Requirements Definition step in order to assure that it is aligned with new policies. In the course of Controls Definition step the requirements to ISMS are transformed into a selection of the best options of NSSs and requirements to them. Upon Controls Implementation step the NSSs are installed and configured in accordance with the policies. Compliance and audit checks carried out during Monitor Operations step in order to ensure that ISMS functions in alignment with the policies. Review Trends and Manage Events step includes identification of events and trends (internal and external in relation to an enterprise) that may indicate a need to make changes in the policies. Further, during the step possible changes are evaluated against any appropriate criteria in order to make sure that the changes are essential and escalated to the beginning of the process [5]. In addition, if during Policy Assessment, Risk Assessment or Policy Development steps it is identified that some policies are not needed any more, and then they must be retired [7]. Note that policy management process is iterative due to continuous changes in technologies, business environment and legal requirements [8].

The research works on network security analysis and policy specifications can be broadly classified into three categories: (a) firewall analysis algorithms and tools; (b) security policy specification languages; and (c) network security analysis using formal approaches. However, none of these addresses the issue of hidden access path analysis which is germane to the topic of this paper. Though the present work focuses on formal approach, a brief overview of all the categories has been presented in this section. Existing literatures on firewall analysis primarily concentrate on inconsistency and redundancy checks but

International Journal For Recent Developments in Science & Technology
2581-4575
Crossref
A Peer Reviewed Research Journal

most of the works are not formally verified. Tools that allow user queries for the purpose of firewall analysis and management include Firmato [1] and Lumeta [2]. These tools can specify an abstract network access control policy and firewall rules that satisfy that policy but lacks in incorporating temporal constraints and hidden rule analysis. Uribe and Cheung [3] have implemented an expert system for inconsistency detection. AlShaer and Hamed [4] worked on the Firewall Policy Advisor. But both of these tools can handle a specific set of problems and simple set of policy constraints. Liu et al. [6] proposed algorithms specialized for finding redundancies. The work of Guttmann et al. [12] focuses on high level modeling of firewall and network configurations that satisfy a given policy but the policy specifications are more general.

3. CONCEPTS OF NETWORK SECURITY

This chapter discusses security policies in the context of requirements for information security and the circumstances in which those requirements must be met, examines common principles of management control, and reviews typical system vulnerabilities, in order to motivate consideration of the specific sorts of security mechanisms that can be built into computer systems—to complement nontechnical management controls and thus implement policy—and to stress the significance of establishing GSSP. Additional information on privacy issues and detailing the results of an informal survey of commercial security officers is provided in the two chapter appendixes.

Organizations and people that use computers can describe their needs for information security and trust in systems in terms of three major requirements:

- Confidentiality: controlling who gets to read information;
- Integrity: assuring that information and programs are changed only in a specified and authorized manner; and
- Availability: assuring that authorized users have continued access to information and resources.

These three requirements may be emphasized differently in various applications. For a national defense system, the chief concern may be ensuring the confidentiality of classified information, whereas a funds transfer system may require strong integrity controls. The requirements for applications that are connected to external systems will differ from those for applications without such interconnection. Thus the specific requirements and controls for information security can vary.The framework within which an organization strives to meet its needs for information security is codified as security policy. A *security policy* is a concise statement, by those responsible for a system (e.g., senior management), of information values, protection responsibilities, and organizational commitment. One can implement that policy by taking specific actions guided by management control principles and utilizing specific security standards, procedures, and mechanisms. Conversely, the selection of standards,

International Journal For Recent Developments in Science & Technology
2581-4575
Crossref
A Peer Reviewed Research Journal

procedures, and mechanisms should be guided by policy to be most effective.

To be useful, a security policy must not only state the security need (e.g., for confidentiality—that data shall be disclosed only to authorized individuals), but also address the range of circumstances under which that need must be met and the associated operating standards. Without this second part, a security policy is so general as to be useless (although the second part may be realized through procedures and standards set to implement the policy). In any particular circumstance, some threats are more probable than others, and a prudent policy setter must assess the threats, assign a level of concern to each, and state a policy in terms of which threats are to be resisted. For example, until recently most policies for security did not require that security needs be met in the face of a virus attack, because that form of attack was uncommon and not widely understood. As viruses have escalated from a hypothetical to a commonplace threat, it has become necessary to rethink such policies in regard to methods of distribution and acquisition of software. Implicit in this process is management's choice of a level of residual risk that it will live with, a level that varies among organizations.

Management controls are the mechanisms and techniques—administrative, procedural, and technical—those are instituted to implement a security policy. Some management controls are explicitly concerned with protecting information and information systems, but the concept of management controls includes much more than a computer's specific role in enforcing security. Note that management controls not only are used by managers, but also may be exercised by users. An effective program of management controls is needed to cover all aspects of information security, including physical security, classification of information, the means of recovering from breaches of security, and above all training to instill awareness and acceptance by people. There are trade-offs among controls. For example, if technical controls are not available, then procedural controls might be used until a technical solution is found.

4. A POLICY MODE

l A system is called an NSS if it is intended to directly or indirectly secure information transferable through an enterprise's network. Assume that function of an NSS is to form any output in accordance with a policy by means of processing of network traffic that comes to its input. In the general case an output of an NSS is network traffic or messages such as log entries and alerts that sent to other systems or IS administrator's console.

Let P be a set of all possible policies that NSSs can implement. Actually, P consists of sequences of symbols that form commands for different NSSs. If any NSS uses GUI instead of CLI, its policy can be expressed as a text string. For instance, the first rule of Check Point firewall policy shown on Fig. 1 can be written as "N=1 Source=Any Destination=Web-Server Service=Any …" or in any other way that reflects semantics of the rule. Also, the set P includes the empty sequence ε. Let T be a set of network traffic,

International Journal For Recent Developments in Science & Technology
2581-4575
Crossref
A Peer Reviewed Research Journal

where $t \in T$ is a network packet (i.e., a sequence of bits), or the empty sequence ε, which means absence of traffic. Consider an NSS as a finite automaton:

$$F = < T \times P, S, T \times M, \delta, f >,$$

where $T \times P$ is an input alphabet, S is a finite set of internal states of NSS, $T \times M$ is an output alphabet, δ: $T \times P \times S \rightarrow S$ is a state-transformation function and f: $T \times P \times S \rightarrow T \times M$ is an output function, M is a set of output messages, which also includes ε (similar to the case of the set T). An NSS functions in discrete time τ and transforms an input traffic $(\tau) \in T$ into an output traffic $t'(\tau) \in T$ in accordance with a policy $p(\tau) \in P$. An NSS changes its internal state $(\tau) \in S$ into state $(\tau + 1) \in S$ while it functions (Fig. 2). The set S can include such parameters as time, number and sequence of packets in a session and other parameters essential to model stateful analysis of a network traffic. Such NSS as a stateless packet filter can be considered as an NSS with one state, i.e., $|S| = 1$.

Assume that every policy $p \in P$ is represented as a triple of the following finite vectors:

- $\vec{x} = (x1, x2, \dots)$ is an input vector, describing an input traffic of an NSS, where $xi \in Xi$, while Xi is a set of parameters of any homogenous nature (for instance, Xi can be a set of IP addresses, protocols, port numbers or any other attribute of a network traffic);
- $\vec{y} = (y1, y2, \dots)$ is an output vector, describing an output traffic and/or messages generated by an NSS, were $yi \in Yi$ while Yi is a set of parameters of output network traffic of any homogenous nature (similar to the case of the input vector) or a set of parameters of the messages;
- $\vec{z} = (z1, z2, \dots)$ is a state vector, describing an internal state of an NSS. For instance, $zi \in Zi$ can be a system time of an NSS.

Fig. 1. An example of Check Point firewall policy

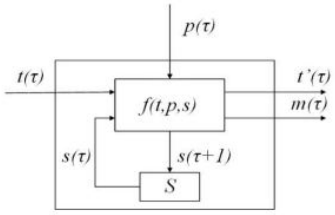

Fig. 2. A finite automaton model of an NSS

when considering Check Point firewall policy (Fig. 1) and network address translation (NAT) policy (Fig. 3), the NSSs can be decomposed to simple NSSs that implement single rules. Fig. 4 shows an example of the composite NSS that implements mentioned firewall and NAT policies. Note that the latter rule of the firewall policy in Fig. 1 is not presented in Fig.4 because each simple NSS Ffw blocks by default any traffic that does not match its policy (i.e., it outputs the empty sequence ε). In contrast with firewall policies, NAT policy accepts any traffic by default, therefore an additional simple NSS $Fnat$ with the default policy (which accepts all traffic that does not match the other NAT rules) is introduced into the model in Fig. 4.

International Journal For Recent Developments in Science & Technology
2581-4575
Crossref
A Peer Reviewed Research Journal

Note also that state blocks S (shown in Fig. 2) of each simple NSS are not depicted in Fig. 4 for the sake of compactness. Two simple NSSs $F1(Def(pgen1))$, $F2(Def(pgen2)) \in \mathcal{F}s$ are called equivalent if and only if they produce equal outputs for equal inputs while implementing policies:

$$F_1\left(Def(p_{gen1})\right) \cong F_2\left(Def(p_{gen2})\right) \leftrightarrow$$

$$\forall p_1 \in Def(p_{gen1})\ \exists p_2 \in Def(p_{gen2}): f_1^*(t, p_1, s_1) = f_2^*(t, p_2, s_2)\ \forall t \in T^* \wedge$$

$$\forall p_2 \in Def(p_{gen2})\ \exists p_1 \in Def(p_{gen1}): f_2^*(t, p_2, s_2) = f_1^*(t, p_1, s_1)\ \forall t \in T^*$$

where $s1$ and $s2$ are initial states of $F1$ and $F2$ respectively, $T*$ is a set of finite sequences of network packets, $f1*$ and $f2*$ are extensions of $f1$ and $f2$ to $T*$. By this equivalence relation the set of simple NSSs partitioned to equivalence classes. All simple NSSs inside any equivalence class produce equal outputs while implementing respective policies and processing network traffic. However, their policies from the syntax point of view can be different. In order to demonstrate the equivalence of NSSs consider Check Point and Cisco firewalls along with the following policy: "Hosts from the network 192.168.1.0/24 are allowed to establish connections to 80 TCP-port on server 10.1.1.10. Connection attempts must be logged". Note that the policy consists of two parts: authorization (allows connections to the server) and obligation (requires logging of connection attempts). Each of two selected NSSs are able to implement the policy. It can be represented in Check Point as shown in Fig. 5. In order to implement the policy in Cisco it is necessary to add one rule to an access-list (for instance, access-list 101):

NO.	ORIGINAL PACKET			TRANSLATED PACKET			INSTALL ON	COMMENT
	SOURCE	DESTINATION	SERVICE	SOURCE	DESTINATION	SERVICE		
1	Web-Server	Any	Any	Web-Server	Original	Original	Corporate-gw	Automatic rule (see the network object data).
2	Any	Web-Server	Any	Original	Web-Server	Original	Corporate-gw	Automatic rule (see the network object data).

Fig. 3. An example of Check Point NAT policy

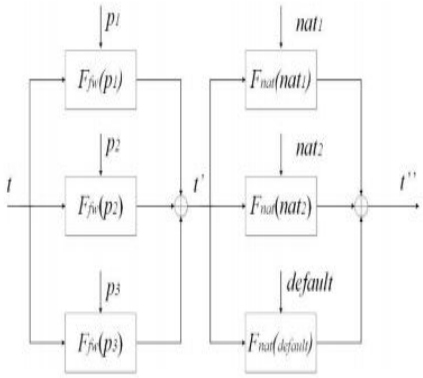

Fig. 4. A composition of simple NSSs

NO.	SOURCE	DESTINATION	VPN	SERVICE	ACTION	TRACK	INSTALL ON	TIME	COMMENT
1	Network-192.168.1.0_24	Server-10.1.1.10	Any Traffic	http	accept	Log	Corporate-gw	Any	

Fig. 5. Check Point firewall policy

5. SECURITY POLICY SPECIFICATION MODEL

The security policy of a network defines a set of parameterized functional rules on flow of packets between different zones in the network. Complexity of the security policy depends on the size of the network, number of controlling parameters and dependency amongst the rules. The specification language must be expressive enough to represent complex security constraints of the network correctly. In the following section, the various constructs of the proposed security policy specification language, SPSL are described.

International Journal For Recent Developments in Science & Technology
2581-4575
Crossref
A Peer Reviewed Research Journal

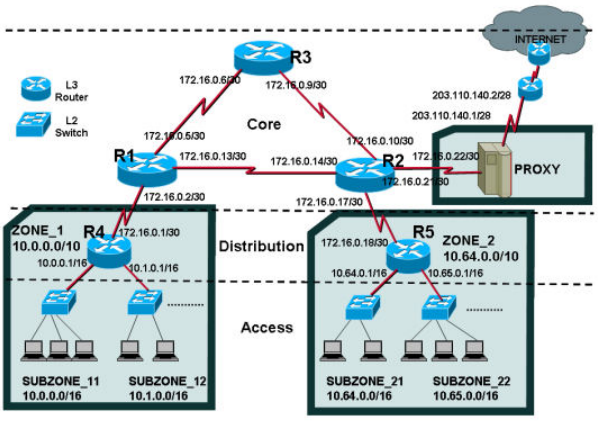

Figure 6.A Typical Enterprise Network.

5.1Security Policy Specification Language (SPSL)

A policy specification language, SPSL (Security Policy Specification Language), has been proposed to model the network topology and the policy rules in an enterprise. The main constructs of the policy specification language can be classified as:

(1) Network topology specification and

(2) Network services and policy rule specification.

The SPSL allows specifying explicit "permit" and "deny" service access rules across the network zones.

1) Network Topology specification: The proposed SPSL language has the following constructs to describe the network topology.

Zone: A zone is a logical unit consisting of workstations, servers or other systems in the network, usually refers to a particular section of an organization. It is represented by IP address block(s). Further, a zone can be partitioned into multiple disjoint sub-zones. Few such zone definitions (refer figure 1 are given in the following example.

Example 1:

Zone ZONE_11 [10.0.0.0-10.0.255.255];
Zone ZONE_12 [10.1.0.0-10.1.255.255];
Zone ZONE_1 [ZONE_11, ZONE_12];

Router: A router is interconnecting Layer-3 switch for connecting various sub-networks. A router can connect multiple network zones. It consists of set of interfaces.

Interface: An interface is the connecting link between a zone and a router or multiple routers. Each interface is identified by an unique IP address

6.A POLICY ALGEBRA

All NSSs inside any equivalence class produce equal outputs while implementing policies. However, in the general case their policies from the syntax point of view are different. As can be seen from the above examples, Check Point does not use CLI and it is not possible to compare its policy with the analogue in Cisco from the syntax point of view; however, the policies have equal semantics. Thus, in the general case there are multiple policies that describe the same simple NSS inside an equivalence class. In order to reduce redundancy of policies for NSSs the set of simple policies Ps needs to be substituted for a set of simple unified policies Pu and $|Pu|$ must be minimal. Consequently, only one generalized policy needs to be assigned for every single equivalence class. In other words, if the set $\mathcal{F}s$ consists of N classes then it is required to have N generalized policies in order to specify policies for all NSSs.

Let $\Omega = \{S, \Phi, \sigma\}$ be a many-sorted signature, where $S = \{st, sm, s1, s2, \ldots, sl\}$ is a set of sorts while st and sm are sorts of network traffic and messages

International Journal For Recent Developments in Science & Technology
2581-4575
Crossref
A Peer Reviewed Research Journal

respectively, $\Phi = \{+, \varphi 1, \ldots, \varphi N\}$ is a set of functional symbols, $\sigma(\varphi i) =< st, si1, \ldots, sik, sm, st >$ is a sort function that defines sorts of arguments $st, si1, si2, \ldots, sik$ and sorts of values sm, st for every functional symbol $\varphi i \in \Phi$. In addition, suppose that $(+) =< , st, st >$, i.e., function "+" has two arguments of the sort of network traffic and a value of the same sort (two examples of the function are shown in Fig. 4).

Let $V = Vst \cup Vsm \cup Vs1 \cup \ldots \cup Vsl$ be a set of variables, where Vsi is a set of variables of a certain sort $si \in S$. Suppose that Θ is the set of Ω-terms that is recursively defined as follows:

- any variable Vsi is a Ω-term θsi of the sort si ;
- any finite expression $\varphi i(\theta st, \theta si1, \theta si2, \ldots, \theta sik, \theta sm, \theta st)$ such that $\sigma(\varphi i) =< st, si1, si2, \ldots, sik, sm, st >$ is a Ω-term of the sort st, where θst, θsm and θsij are any variables of corresponding sorts st, sm and sij.

Suppose that any $si \in S$ is a name of a certain parameter of generalized policy, any $\varphi i \in \Phi$ is a designation of a certain equivalence class of NSSs, then $(\theta st, \theta si1, \theta si2, \ldots, \theta sik, \theta sm, \theta st)$ is a representation of the output function of an NSS from equivalence class designated by φi that includes representation of respective simple policy. By construction, Θ includes representations of all generalized policies for simple and composite NSSs. For instance, the NSS shown in Fig. 4 can be represented as follows:

$$\begin{cases} F_{fw}(t, p_1, s) + F_{fw}(t, p_2, s) + F_{fw}(t, p_3, s) = t' \\ F_{nat}(t', nat_1, s) + F_{nat}(t', nat_2, s) + F_{nat}(t', default, s) = t'' \end{cases}$$

where Ffw and $Fnat$ are designations of equivalence classes; pi, $natj$, $default$, and s are collections of parameters of respective policies and states (parameters of policies consist of input, output and state vectors). Note that notations $+(\theta st, \theta st)$ and $\theta st + \theta st$ are equivalent. Let $(A,)$ be a many-sorted algebra, where A is a carrier set of the algebra and I is an interpretation of the signature Ω. For every sort $si \in S$ the interpretation associates a subset $Asi \subseteq A$ and for every functional symbol $\varphi i \in \Phi$ it associates the function: $Ast \times Asi1 \times Asi2 \times Asik \rightarrow Asm \times Ast$ that defines the output function of an NSS of the respective equivalence class . Thus, the algebra models all definite policies for simple NSSs of each equivalence class as well as definite policies for composite NSSs. In order to translate the policies into the native policy formats of concrete NSSs' platforms described many-sorted algebra can be represented as a formal language with a generative grammar $G = (\mathcal{T}, \mathcal{N}, \mathcal{S}, R)$, where $\mathcal{T}$ is a terminal vocabulary that reflects carrier set A, $\mathcal{N}$ is a non-terminal vocabulary that includes terms, $\mathcal{S} \in \mathcal{N}$ is a starting non-terminal symbol of every policy, and R is a set productions. Representation of the policies in the form of a formal language allows application of the existing parsing algorithms.

7. CONCLUSION

The approach to policy modeling presented in the paper is based on finite automaton model of an NSS. Decomposition of an NSS

International Journal For Recent Developments in Science & Technology
2581-4575
Crossref
A Peer Reviewed Research Journal

to simple NSSs and their classification facilitates composition of policies and selection of the most effective NSSs aligned with them. Policies are considered independently on platforms of NSSs and can be specified identically for equivalent NSSs. Application of translation methods for unified policies allows implementation of policies in concrete NSSs platforms. The future challenge for the approach is development of derivation method from RBAC policies into described NSSs policies and building of conflict resolution methods. Construction of an algorithm for classification of NSSs is also a future work.

REFERENCES

1. 2014 Cyberthreat Defense Report North America & Europe, Cyber Edge Group (2014)

2. The State of Network Security 2013: Attitudes and Opinions, AlgoSec (2013)

3. Chappell, M.J., D'Arcy, J., Striegel, A.: An Analysis of Firewall Rulebase (Mis)Management Practices, Journal of Information System Security Association (ISSA) 7, 12–18 (2009)

4. Wool, A.: Trends in Firewall Configuration Errors: Measuring the Holes in Swiss Cheese. IEEE Internet Computing 14 (4), 58–65 (2010)

5. Rees, J., Bandyopadhyay, S., Stafford, E.H.: PFIRES: a Policy Framework for Information Security. Communications of the ACM 46 (7), 101-106 (2003)

6. Wahsheh, L.A., Alves-Foss, J.: Security Policy Development: Towards a Life-Cycle and Logic-Based Verification Model. American Journal of Applied Sciences 5 (9), 1117–1126 (2008)

7. Knapp, K.J., Morris, R.F. Jr., Marshall, T.E., Byrd, T.A.: Information security policy: An Organizational-level Process Model. Computers & Security 28 (7), 493–508 (2009)

8. Tuyikeze, T., Potts, D.: An Information Security Policy Development Life Cycle. In: South African Information Security Multi-Conference, pp. 165–176. (2010)

9. Labored, R., Kamel, M., Barrera, F., Benzekri, A.: Implementation of a Formal Security Policy Refinement Process in WBEM Architecture. In: 4th Latin American Network Operations and Management Symposium, pp. 65–76. (2005)

10. Cuppens, F., Cuppens-Boulahia, N., Sans, T., Miège, A.: A Formal Approach to Specify and Deploy a Network Security Policy. In: 2nd Workshop on Formal Aspects in Security and Trust, pp. 203–218. (2004)

11. Preda, S., Cuppens-Boulahia, N., Cuppens, F., Garcia-Alfaro, J., Touting, L.: Model-Driven Security Policy Deployment: Property Oriented Approach. In: 2nd International Symposium on Engineering Secure Software and Systems, pp. 123–139. (2010)

12. Preda, S., Cuppens-Boulahia, N., Cuppens, F., Touting, L.: Architecture-Aware Adaptive Deployment of Contextual Security Policies. In: 5th International Conference on Availability, Reliability and Security, pp. 87–95. (2010)

13. Garcia-Alfaro, J., Cuppens, F., Cuppens-Boulahia, N., Preda, S.: MIRAGE: A Management Tool for the Analysis and Deployment of Network Security Policies. In: 5th international Workshop on Data

2581-4575
International Journal For Recent Developments in Science & Technology
Crossref
A Peer Reviewed Research Journal


Privacy Management and 3rd International Conference on Autonomous Spontaneous Security, pp. 203–215. (2010)

14. Rahman, M. A., Al-Shaer, E.: A Formal Framework for Network Security Design Synthesis. In: 33rd International Conference on Distributed Computing Systems, pp. 560–570. (2013)