Computing and Informatics, Vol. 23, 2004, 77–100

MANAGEMENT AND VERIFICATION OF FIREWALL AND ROUTER ACCESS LISTS*

Ahmed AbdAllah Hassan, Ladislav Hudec

Faculty of Informatics and Information Technologies
Slovak University of Technology in Bratislava
Ilkovičova 3
812 19 Bratislava, Slovakia
e-mail: ahmadtdm@yahoo.com, lhudec@dcs.elf.stuba.sk

Manuscript received 8 September 2003; revised 13 February 2004
Communicated by Ladislav Hluchý

Abstract. Security in computer networks is a very complex task especially if it is required to separate a corporate network from public Internet or to divide a company's intranet into multiple zones with different security requirements. The network security policy that describes these security requirements is primarily presented in a high-level form. Also, the security policy is enforced using some low-level security mechanisms, mainly firewall technology. One of the main difficulties faced by the network administrator is how to translate the high-level policy description to the low-level firewall rule-base. This paper presents Role-Based Network Security (RBNS) model that can be used as an intermediary level between high-level policy form and low-level firewall rule-base. We use the Role-Based Access Control (RBAC) model as a framework for our proposed RBNS model. The main concept of RBNS model is that network services are assigned to roles and hosts are made members of appropriate roles thereby acquiring the roles' network services. Also, the paper presents a compilation algorithm that can be used to automatically generate the low-level firewall rule-base from the RBNS intermediary-level. The paper presents a proposed verification algorithm to prove that the high-level policy and the translated low-level firewall rule-base are equivalent. Based on the RBNS model, we design and implement a firewall management toolkit. The paper demonstrates in brief the toolkit's capabilities through an example, thus showing that the using of this model separates the high-level security policy from the underlying enforcement

* This work was supported by Slovak Science Grant Agency No. VG 1/0157/03 "Methods and tools for development of the secure networked and distributed mobile computer systems and their management II".

mechanism. This separation offers easier management and debugging of low-level firewall rule-base at an appropriate level of abstraction.

Keywords: Network security, security modeling, security policy, firewall management, packet filtering, router access lists

1 INTRODUCTION

Security of networks has become one of the primary concerns when it is required to connect a private network to the Internet or even to isolate a particular subnet from the overall corporate network. To provide the required level of protection, an organization needs a security policy to prevent unauthorized users from accessing resources on the private network and to protect against the unauthorized export of private information. Even if an organization is not connected to the Internet, it may still need to establish an internal security policy to manage users' access to portions of the network and protect sensitive or secret information [1, 2]. This security policy is primarily presented in a high-level form. An example of high-level service-specific policy is described in [3].

In a network security policy, the security policy concentrates on controlling network access to hosts and services. In this case, firewall technology is the main security mechanism that can be used to enforce the security policy [4, 5, 6, 7]. The single most important factor of firewall's security is how the network administrator configures it. However, while firewalls themselves have been some impressive technological advances, firewall configuration and management seem to be lagging behind [8].

Let us go briefly over the reasons that make firewall management a difficult task. A firewall is typically placed on a gateway, separating the corporate Intranet from the public Internet. Most of today's firewalls are configured via a rule-base. This rule-base instructs the firewall, which inbound packets to let pass and which to block. Also, it specifies which outbound packets are allowed. The administrator needs to implement the high-level corporate security policy with this low-level rule-base.

Typically, the configuration interface allows the security administrator to define various host-groups and service-groups. A single rule typically includes a source, a destination, a service-group, and an appropriate action [6]. The source and destination are host-groups, while the action is either "pass" or "drop". From this description, it is obvious that the security of whole Intranet depends on the exact content of the rule-base, with no higher level of abstraction available.

This problem becomes worse for a medium- or large-sized company that uses more than a single firewall. These firewalls divide the company's departments into multiple zones. In this case the security policy is typically realized by multiple rule-bases located on multiple gateways that connect the different zones to each other.

Thus the interplay between these rule-bases must be carefully examined so as not to introduce security holes.

To solve these problems, we presented a network security model, which acts as an intermediary level between high-level security policy and low-level firewall rule-base (Figure 1).

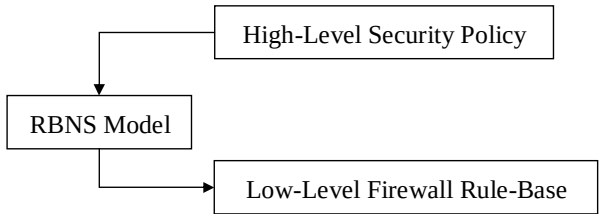

Fig. 1. RBNS model as intermediary level

1.1 Big Picture of the Problem

The problem of firewall management has a similar flavor to access control problem in the sense that the definition of firewall rules can be considered as an assignment process of some services (corresponding to permissions in access control problem) to IP addresses (corresponding to users in access control problem). This similarity led us to choose RBAC model, a generalized approach to access control [9, 10, 11, 12], as a framework to RBNS model. However, we added some foundational work to the RBAC model to tune it to be appropriate as a network security model.

1.2 Advantages of the Model

The main advantages of using the RBNS model are:

1. It allows the management of firewall configuration files in a higher level of abstraction. This decreases the probability of mistaken configured firewalls and provides easier analysis and debugging of firewall configuration files.
2. It allows the separation of the security policy design from the firewall vendor specifics. This allows a security administrator to focus on designing an appropriate policy without worrying about firewall rule complexity, and other low-level configuration issues. It also enables a unified management of network components from different vendors and a much easier transition when a company switches vendors.
3. It allows the separation of the security policy design from the actual network topology. This enables the administrator to maintain a consistent policy in the face of Intranet topology changes. Furthermore, this modularization also allows the administrator to reuse the same policy at multiple corporate sites with

different network details, or to allow smaller companies to use default/exemplary policies designed by experts.

1.3 Related Work

The work related to our problem is probably Firmato and Fang [8, 13]. Firmato is a firewall management toolkit with: (1) an entity-relationship model containing global knowledge of the security policy and topology, (2) a model definition language, which is used as an interface to define an instance of the entity-relationship model, (3) a model compiler, translating global knowledge of the model into firewall-specific configuration files, and (4) a graphical firewall rules illustrator. However, the transition from high-level to its abstracted model level is done through the model definition language which seems complicated and needs hard effort. Also, it used an entity relationship model that suffers from the following weaknesses:

- Mixing the concepts of roles and groups
- Making the roles as an attribute of capability entity whereas the capability entity is assigned to roles. Besides hosts are attached to roles. This leads to difficult and confusing assignment between hosts and services.

Besides, its compiler is tailored to the Lucent Managed Firewall. The kit does not generate a general syntax of the rule-base that can be used with several firewalls. Also, the kit focuses only on stateless packet filtering. Moreover, there is no effort done to deal with more complicated security policy that assumes NAT technology or needs checking of ACK bit. There is no effort done to verify the equivalence between the high-level security policy and the corresponding low-level firewall configuration files.

The Guttman's work on filtering [14, 15] is also related to our work in the sense that it introduced a language for expressing the network security policy as an important step toward network security management. However, it does not provide automatic generation of firewall rules or complete separation of the security policy from the network topology.

Scott's work [16] is also related to our work from the view that it wants to maintain and validate firewall rule-base. It presents a new algorithm for representing firewall rule-base as a BDD (binary decision diagram) – a compact method of representing and manipulating Boolean expressions – and then shows how the resulting Boolean expression can be used to analyze rule sets. However, no effort is done for applying this algorithm for automatic generation of firewall rule-base.

Role-based access control is our framework model. However, their focus is mostly on assigning permissions to users (humans) of computing systems, whereas we deal with assigning services to IP and with topology related issues.

On the other hand, our proposed model can be used directly to present the high-level security policy. Our management kit provides a user friendly GUI to accomplish this task. Also, our proposed model introduces the concept of constraints

to allow the presentation of more rigorous security policies that may contain protection against spoofing attack, ACK bit status checking, or even includes NAT technology. In addition, a verification algorithm is proposed to verify the equivalence between high-level policy and corresponding low-level firewall configuration files.

1.4 Content and Structure of the Paper

Section 2 describes in details the RBNS model. With the aid of running example, Section 2 illustrates how the high-level security policy can be represented by the RBNS intermediate-level and how this intermediate-level is compiled to the low-level firewall rule-base. Section 3 introduces an algorithm for verifying the equivalence between high-level policy and the corresponding low-level firewall rule-base. Section 4 explains in brief our firewall management toolkit. Finally, Section 5 concludes the paper and mentions possible extension of the work.

2 RBNS MODEL

This section presents detailed description of the RBNS model. It starts with brief definition of the used terminology. Then, Section 2.2 introduces the concept and the mathematical description of the base-RBNS model. Section 2.3 illustrates how high-level security policy is defined using RBNS model entities. Then, Section 2.4 explains how the RBNS intermediate-level of the security policy is compiled to low-level firewall rule-base. Section 2.5 describes an extension of the Base-RBNS model – the constrained-RBNS model.

2.1 Terminology

Before the description of the model, we define precisely the following terms [5]:

Gateways These are the packets filtering machines, which can be either firewalls or routers.

Interfaces By definition a gateway is multi-homed host, since it has at least two Internet connections. Each connection goes through an interface, which has its own unique IP address. We assume that each interface has a packet filtering the rule-base associated with it. (This is more general than assuming only a single rule-base per gateway).

Zones The gateways partition the IP address space into disjoint zones. Most zones correspond to a corporation's subnet(s), usually with one big "Internet" zone corresponding to the portion of the IP address space that is not used by the corporation.

2.2 Base-RBNS Model Description

The main concept of RBNS model is that services are associated with roles, and hosts are assigned to appropriate roles thereby acquiring the roles' network services. RBNS model has two levels; the base-RBNS level and the constrained-RBNS level. The base level is used to present primary security policy. If the security policy assumes NAT technology, protection against spoofing attacks, ACK bit checking, or other more rigorous security requirements, then the constrained-RBNS level has to be used to present these more rigorous security requirements.

As shown in Figure 2, base-RBNS model, there are three sets of entities called hosts (H), roles (R), and services (S).

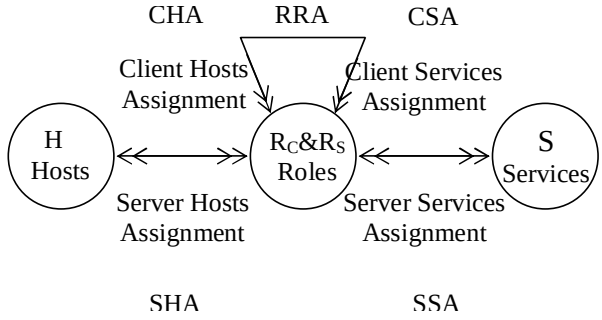

Fig. 2. The RBNS model

A host entity models a machine on the network with its IP address. The host entity represents the topology-related part on the model. It plays double parts in the model: the host may appear as a source IP requesting a service (client-host) or as a destination IP announcing which services can be supplied by it (server-host).

A service entity is the combination of a protocol-base (e.g., TCP, UDP, etc.) and the standard port numbers of that service. For instance, the service telnet is defined as TCP with port number 23. These services are always positive and confer the ability to the holder of the services to perform some actions on the network.

A role is properly viewed as a semantic construct around which network security policy is formulated. We have two types of roles; client-role and server-role. Client-role intermediates a client-host to the network services permitted to that host. Server-role intermediates a server-host to the network services supplied by that host.

Also, Figure 2 shows the following relations:

Client-hosts assignment (CHA) defines the assignment between hosts and client-roles.

Server-hosts assignment (SHA) defines the assignment between hosts and server-roles.

Client-services assignment (CSA) defines the assignment between services and client-roles.

Server-services assignment (SSA) defines the assignment between services and server-roles.

Role-role assignment (RRA) defines the assignment between client-roles and server-roles.

The assignment between client-roles and server-roles enables the model to define a complete communication path between client-hosts that are made members of a client-role (sources) and server-hosts that are made members of a server-role (destinations).

All these relations are many-to-many relations. A host can be a member of many roles and a role can have many hosts. Similarly, a role can have many services and the same service can be assigned to many roles.

The key to RBNS model lies in these relations. The placement of a role as an intermediary to enable a host to exercise a service provides much greater control over rules' configuration and review than does direct relating hosts to services like host-based model.

The following definition formalizes the above discussion as:

Definition 1. The base-RBNS model has the following components:

- H, R_C, R_S, and S (hosts, client-roles, server-roles, and services, respectively).
- $CSA \subseteq S \times R_C$, a many-to-many service to client-role assignment relation.
- $SSA \subseteq S \times R_S$, a many-to-many service to server-role assignment relation.
- $CHA \subseteq H \times R_C$, a many-to-many host to client-role assignment relation.
- $SHA \subseteq H \times R_S$, a many-to-many host to server-role assignment relation.
- $RRA \subseteq R_C \times R_S$, a many-to-many client- role to server-role assignment relation.

An important point to mention is that if some client-hosts are made members in a client-role, these client-hosts will have the ability to perform the services owned by this client-role on the network. However, a question arises: which hosts on the network these client-hosts would be permitted to perform these services on them? The answer to this questions lies on the role-to-role assignment relation. If there is a client-role "A" assigned to server-role "B" then, the members of role "A" have the ability to perform the *common services* between roles "A" and "B" on the members of role "B".

2.3 The Definition of High-Level Policy Using RBNS Notions

This section uses an example to illustrate how various entities of the RBNS model can be used to define a concrete policy.

For this purpose we consider a corporation with a two-firewall network configuration as shown in Figure 3. The network is defined as a subnet 172.20.*.*. The corporations' hosts IPs are listed in Figure 3.

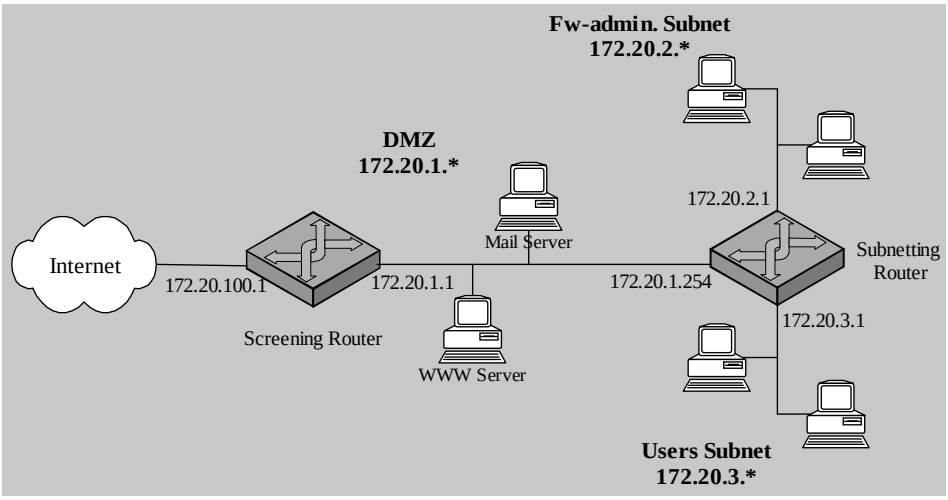

Fig. 3. The network topology of the example

The external firewall (screening router) guards the corporation's Internet connection. Behind it is the DMZ (demilitarized zone), which contains the corporation's externally visible servers. In our case these servers provide web and SMTP services. The corporation uses two hosts to provide these services, one for web and the other for SMTP service. Behind the DMZ is the internal firewall (subnetting router), which guards the corporation's Intranet. This firewall actually has three interfaces: one for the DMZ, one for the corporate users' subnet, and a separate interface connecting to the subnet of firewalls administrators. Securing the administrators subnet is critical to the network's integrity, so it is more appropriate to separate their subnet from the other corporate hosts.

2.3.1 High-Level Policy Definition

We consider a simple high-level security policy which has the following goals:

1. All internal corporate hosts can access all Internet resources.
2. Corporate hosts and external hosts can access web server only with HTTP and HTTPS services.
3. Corporate hosts and internet hosts can access mail server only with SMTP service.
4. The firewall gateway interfaces are only accessible from the fw-admin. hosts.

2.3.2 The Logical Procedure to Define the Model Entities

To present this high-level policy using RBNS model the network administrator needs to define the various RBNS entities in frame of the logical procedure described in Figure 4.

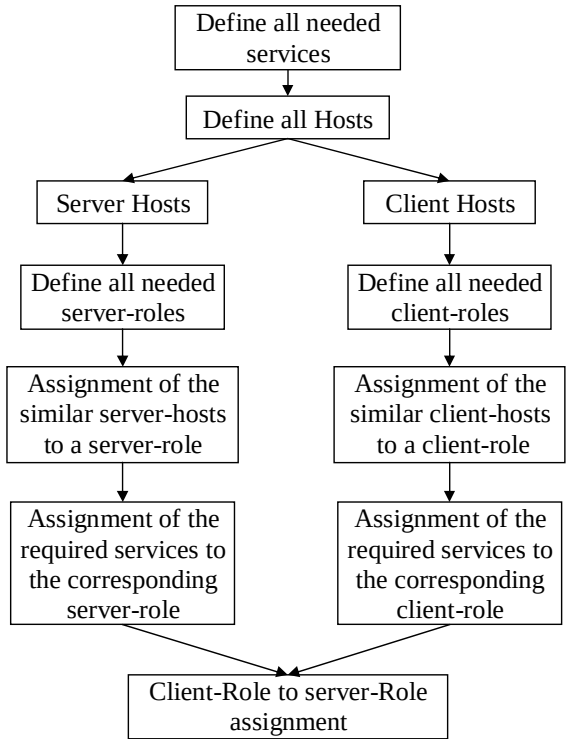

Fig. 4. Logical procedure for RBNS model entities definition

2.3.3 The RBNS Model Entities Definition

Within the above procedure the RBNS model entities corresponding to the running example can be defined in the following steps:

Step 1: The definition of all needed network services

The first step is to define the various services according to the following syntax:

$< service - name >'='< protocol - base >' ['< standard - port - number >']'$

According to the policy described above the following services are needed:

http = TCP [80]
https = TCP [443]
smtp = TCP [25]
ssh = TCP [22]
ping = ICMP [8]
all-tcp = TCP [*]

Step 2: The definition and classification of all network hosts

All hosts listed in Figure 3 are defined and classified in Table 1.

Host name	IP and mask	Type
Fw-admin	172.20.2.*/24	Client
WWW-server	172.20.1.4/32	Server
Mail-server	172.20.1.5/32	Server
Corporate-users	172.20.3.*/24	Client
Fw-interfaces	172.20.1.1/32	Server
	172.20.1.254/32	Server
	172.20.3.1/32	Server
	172.20.2.1/32	Server
External-hosts	*.*.*.*/0	Client & Server

Table 1. Network hosts definition and classification

Step 3: The definition of needed server-roles and client-roles

Based on step 2 and the goals of the policy the needed server-roles could be as follows:

- R-web-server
- R-mail-Server
- R-fw-interfaces
- R-internet-access

Also, the client-roles could be as follows:

- R-fw-admin
- R-corp-users
- R-external-users.

The meaning of roles is clear from their names.

Step 4: The assignment of hosts and services to the appropriate roles

Table 2 summarizes all roles with their host members and associated services.

Step 5: role-to-role assignment

The last step is to assign each client-role to the peer server-roles. Table 3 shows each client-role with its peer server-roles.

Role	Member-Hosts	Services
R-web-server	WWW-server	http & https
R-mail-Server	Mail-server	smtp
R-fw-interfaces	fw-interfaces	ssh & ping
R-internet-access	external-hosts	all-tcp
R-fw-admin	fw-admin	all-tcp
R-corp-users	Corporate-users	all-tcp
R-external-users	External-hosts	http & https & smtp

Table 2. Roles' member-hosts and services

Client-role	Peer server-roles
R-fw-admin	R-web-server & R-mail-Server & R-fw-interfaces & R-internet-access
R-corp-users	R-web-server & R-mail-Server & R-internet-access
R-external-users	R-web-server & R-mail-Server

Table 3. Role-to-role assignment

By the end of this step the high-level security policy is completely translated to the RBNS intermediate level.

The next section illustrates how the RBNS intermediate level could be automatically transferred to the low-level firewall configuration file.

2.4 RBNS model compiler

This section explains an algorithm to automatically compile the RBNS intermediate-level of the security policy to the low-level firewall rules-base. The following pseudo code defines the compilation algorithm for both stateless and stateful firewalls:

```
DOWHILE i <= no-of-client-roles {
  DOWHILE j <= no-of-host-member-of-client-role-i {
    rule-sourceIP-field = host-member-j-IP
    if stateless filtering is considered add another return-rule with
      return-rule-destinationIP-field = host-member-j-IP
    DOWHILE k <= no-of-peer-server-roles-of-client-role-i {
      Allowed services = common services between client-role-i and server-role-k
      rule-protocol-field = base protocol of the service
      rule-sourceport-field = (> 1023)
      rule-destinationport-field = standard port of the service
      if stateless filtering is considered add the following field to the return-rule {
        return-rule-protocol-field = base protocol of the service
        return-rule-sourceport-field = standard port of the service
        return-rule-destinationport-field = (> 1023) }
      DOWHILE l <= no-of-host-members-of-server-role-k {
        rule-destinationIP-field = host-member-l-IP
```

```
            if stateless filtering is considered add the following field to the return-rule
              return-rule-sourceIP-field = host-member-l-IP }
        END DOWHILE }
      END DOWHILE }
    END DOWHILE }
END DOWHILE
```

Based on Tables 1, 2, and 3, the compilation algorithm could be applied on the running example of Section 2.3 to generate the corresponding low-level firewall rules. Table 4 presents a sample from these rules.

Rule No.	Source IP	Dest. IP	Protocol	Source port	Dest. port
1	172.20.2.*/24	172.20.1.4/32	tcp	> 1023	80
2	172.20.2.*/24	172.20.1.1/32	icmp	> 1023	8
3	172.20.2.*/24	*.*.*.*/0	tcp	> 1023	*
4	172.20.3.*/24	172.20.1.4/32	tcp	> 1023	443
5	172.20.3.*/24	172.20.1.5/32	tcp	> 1023	25
6	*.*.*.*/0	172.20.1.4/32	tcp	> 1023	80
7	*.*.*.*/0	172.20.1.5/32	tcp	> 1023	25

Table 4. Low-level stateful firewall rules

If the compilation algorithm is applied to generate a stateless packet filter rules the following rules would be obtained:

- The same rules defined in Table 4.
- Besides, another rule set corresponding to each rule in Table 4 to define the response direction of the packet. A sample of these required rules is summarized in Table 5.

Rule No.	Source IP	Dest. IP	Protocol	Source port	Dest. port
1'	172.20.1.4/32	172.20.2.*/24	tcp	80	> 1023
2'	172.20.1.1/32	172.20.2.*/24	icmp	8	> 1023
...	...	...	...	...	...

Table 5. Extra low-level rules needed for stateless firewall

2.5 The Constrained-RBNS Model

Constrained-RBNS level introduces the concept of constraints to increase the capability of base-RBNS level to present more rigorous security policies that may contain protection against spoofing attack, ACK bit status checking, or even includes

NAT technology. These constraints apply to all aspects of RBNS as indicated in Figure 5.

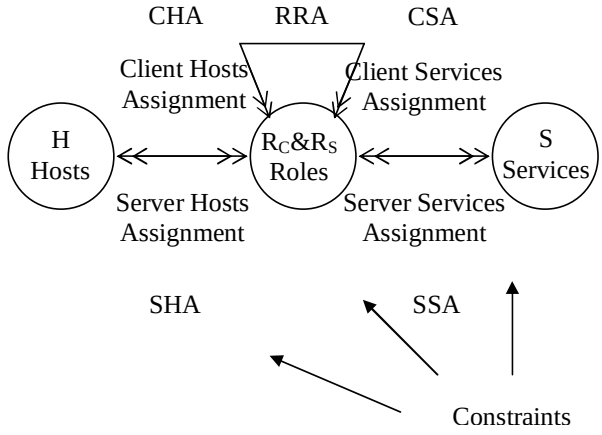

Fig. 5. Constrained RBNS model

The constrained-RBNS is formally defined as:

Definition 2. Constrained-RBNS is unchanged from base-RBNS except for requiring that there be a collection of constraints that enhance the ability of the model to present more complicated high-level security policy.

Security requirements representation in the low-level differs from product to another so, we define the constraints in the low-level by sentences in some appropriate formal language. The remaining part of this section illustrates the anti-spoofing attack constraints as an example.

The base-RBNS model can not present any rule of the high-level policy regarding the protection against spoofing attacks. Spoofed packet is any packet comes on an interface of the gateway claiming that it comes from another host. Without any protection against this type of attack this spoofed packet will take all the services associated to the roles in which the claimed IP is a member.

To reduce the possibility of spoofing attacks the constrained-RBNS model uses some constraints. These constraints define the direction of the packet as shown in Table 6.

Interface	**Source IP**	**Action**
Internal	Internal	Pass
Internal	External	Dropp
External	Internal	Drop
External	External	Pass

Table 6. Anti-spoofing constraints

The corresponding low-level rules to these constraints will be:

Source	Destination	Service	Direction	Action
Internal	Any	Any	Outbound	Pass
External	Any	Any	Outbound	Drop
Internal	Any	Any	Inbound	Drop
External	Any	Any	Inbound	Pass

These rules can be used in general case when the security policy deals with spoofed packets coming on both sides of the firewall (subnetting firewall). However, practically, spoofed packet is one which is coming from the internet into the corporate network with its source address belongs to one of the corporate addresses. In this case the firewall uses one anti-spoofing rule that can be defined as:

Source	Destination	Service	Direction	Action
Internal	Any	Any	Inbound	Drop

Definition 3. The formal language presentation of the anti-spoofing constraints can be stated as:

For all rules, accept packets iff (source IP and Interface IP are on the same subnet).

3 VERIFICATION ALGORITHM

Verification involves developing formal description of the high-level network security policy and proving that the implementation of this policy into low-level is equivalent to the high-level one. This section starts with the explanation of a formal model that describes the high-level security policy. Section 3.2 describes how rules can be presented by Boolean expression. Section 3.3 illustrates the proposed verification algorithm.

3.1 Formal Description of High-Level Network Security Policy

The high-level security policy can be formally described using the access control matrix (ACM) [17]. In this case, the subjects of the ACM matrix correspond to IPs of hosts requiring services on the network (client hosts) and the objects of the matrix correspond to IPs of other hosts providing services on the network (server hosts). The matrix decision rules have to specify the permitted services between the subject IP and the Object IP.

The ACM matrix that describes the high-level policy defined in Section 2.3.1 could be as follows:

		Server-hosts			
		WWW-server	Multi-server	Fw-interfaces	External-hosts
Client-hosts	Fw-admin	http https	smtp ftp	ssh ping	all-tcp
	Corporate-users	http https	smtp ftp	-	all-tcp
	External-Hosts	http https	smtp ftp	-	-

3.2 Converting Rule Sets into Boolean Expressions

The key technique used is that numbers can be represented as bit vectors. For example, an address segment is a number between 0 and 255. At a lower level, the address segment is just a vector of 8 bits. For example, to represent the 8-bit number x symbolically, we introduce the bit-vector $<x_7, \ldots, x_0>$, where each of the x_is are Boolean variables. The condition that the vector of x's is equal to 3, is just $<x_7, \ldots, x_0> = <f, f, f, f, f, f, t, t>$. Using the definition of equality of vectors yields the Boolean expression $x_7', x_6', x_5', x_4', x_3', x_2', x_1, x_0$, (bits 0 and 1 are high, the others low).

The key components in a rule are:

- The protocol of the packet: TCP, UDP or ICMP.
- The range of port numbers.
- The source address: four segments, each a number in the range $0 \ldots 255$.
- The source address's mask.
- The destination address (in the same format as the source).
- The destination address's mask.

The remaining part of this section introduces a number of Boolean variables and expressions to represent the information in the rule.

Each protocol is assigned a number $0, \ldots, n-1$. These numbers can be represented in $m_n = log_2 n$ bits, and so we introduce p_m variables $< p_{m-1}, \ldots, p_0 >$ to encode the protocol used. In the examples given below, the protocols can be represented in 3 bits. For example, if the rule refers to a tcp protocol (protocol 3) packet, then this is represented by the expression $<p_2, p_1, p_0> = <f, t, t>$, or just $<p_2', p_1, p_0>$.

For each segment of the source address we introduce 8 variables of the form $sax_7, \ldots, sax_0$, where x is the segment number. For example, if segment 2 of the source address refers to the number 141, this is encoded as:

$$sa2_7, sa2_6', sa2_5', sa2_4, sa2_3, sa2_2, sa2_1', sa2_0$$

For each segment of the destination address we introduce 8 variables of the form $dax_7, \ldots, dax_0$, where x is the segment number. The encoding of destination addresses is similar to the encoding of the source address.

As there can be up to 64000 ports specified, port numbers can be represented in 16 bits, so we introduce the Boolean vector $< po_{15}, \ldots, po_0 >$ (with po_{15} being the most significant bit) which encodes the port number. Using these variables it is possible to represent individual ports as well as ranges of ports.

The main function of mask is to define the bits of the base address that should be matched. To include the effect of masks in our proposed mechanism we can add a simple function that extracts just the IP bits of the base address that should be matched.

For example, consider the following rule:

Rule No.	Source IP	Destination IP	Protocol	Port
100	20.9.17.8/16	40.8.16.20/32	tcp	80

This rule would be encoded by the Boolean expression:

Exp_ 100

$$
\begin{aligned}
&\{\\
&(sa1_7^{/}, sa1_6^{/}, sa1_5^{/}, sa1_4, sa1_3^{/}, sa1_2, sa1_1^{/}, sa1_0 \ \hat{}\\
&sa2_7^{/}, sa2_6^{/}, sa2_5^{/}, sa2_4^{/}, sa2_3, sa2_2^{/}, sa2_1^{/}, sa2_0) \ \hat{}\\
&(da1_7^{/}, da1_6^{/}, da1_5, da1_4^{/}, da1_3, da1_2^{/}, da1_1^{/}, da1_0^{/} \ \hat{}\\
&da2_7^{/}, da2_6^{/}, da2_5^{/}, da2_4^{/}, da2_3, da2_2^{/}, da2_1^{/}, da2_0^{/} \ \hat{}\\
&da3_7^{/}, da3_6^{/}, da3_5^{/}, da3_4, da3_3^{/}, da3_2^{/}, da3_1^{/}, da3_0^{/} \ \hat{}\\
&da4_7^{/}, da4_6^{/}, da4_5^{/}, da4_4, da4_3^{/}, da4_2, da4_1^{/}, da4_0^{/}) \ \hat{}\\
&(p_2^{/}, p_1, p_0) \ \hat{}\\
&(po_{15}^{/}, po_{14}^{/}, po_{13}^{/}, po_{12}^{/}, po_{11}^{/}, po_{10}^{/}, po_9^{/}, po_8^{/}, po_7^{/}, po_6, po_5^{/}, po_4, po_3^{/}, po_2^{/}, po_1^{/}, po_0^{/})\\
&\}
\end{aligned}
$$

It is also useful to note that the appearance of "all" in certain field is encoded in the expression by the absence of that field. For example, if we want to allow all protocols then the vector $< p_2, p_1, p_0 >$ must be replaced by 1. This means that this part is eliminated from the expression.

Using the methods described above the entire rule set can in principle be represented by a Boolean expression. The following algorithm [16] illustrates this:

- If the rule set is empty then no packets can be accepted and so the corresponding Boolean expression is "f".
- If the first rule is an accept rule then a packet will be accepted if it matches the rule or if accepted by the rest of the rule set. So the corresponding Boolean expression is the disjunction of the Boolean expression representing the first rule and the Boolean expression representing the rest of the rules.

- If the first rule is a reject rule then a packet will be accepted if it does not match the first rule and it is accepted by the rest of the rule set. So the corresponding Boolean expression is the conjunction of the negation of the Boolean expression representing the first rule and the Boolean expression representing the rest of the rules.

However, the RBNS model assumes the security policy (default == deny). In this case all the rules should be of type ACCEPT. In this case the Boolean expression that presents all the entire rule set is:

$$\text{Exp\_1 or (Exp\_2 or (Exp\_3} \ldots \text{ or (Exp\_}i \ldots))\text{)}$$

where Exp_i is the Boolean expression of the i^{th} rule.

3.3 The Verification Algorithm

Our proposed verification algorithm, Figure 6, has three phases; the first phase is to construct the ACM matrix that describes the high-level policy. Then, the matrix entities are encoded to binary format. The second phase is to convert the low-level entire rule set generated from the RBNS model to Boolean expression. In the third phase all possible communication paths are extracted from the ACM matrix and are substituted with their binary data into the Boolean expression of the second phase. The verification decision is taken according to the following criteria:

1. If the Boolean expression results ONEs for all extracted communication paths then the high-level policy is successfully implemented (regarding the accepted traffic) in the low-level.
2. If the Boolean expression results ZEROs for some communication paths then these paths are reported as absent paths. The low-level firewall rules have to be changed in someway to accept these packets.

To ensure that there are no other communication paths without their definition in the high-level policy the algorithm has to check all the remaining possibilities of the Boolean expression (remaining 2^{83} cases as we have 83 variables constituting the Boolean expression). This number seems to be large; thus, the algorithm chooses some random cases to check. In this case the verification decision is taken according to the following criteria:

1. If the Boolean expression results ZEROs for all these paths then the high-level policy is successfully implemented (regarding the denied traffic) in the low-level.
2. If the Boolean expression results ONEs for some communication paths then these paths are reported as violation paths. The low-level firewall rules have to be changed in some way to deny these packets.

The algorithm described above has been implemented in a prototype tool using MatLab version 6.5. The algorithm has been tested on the same running example

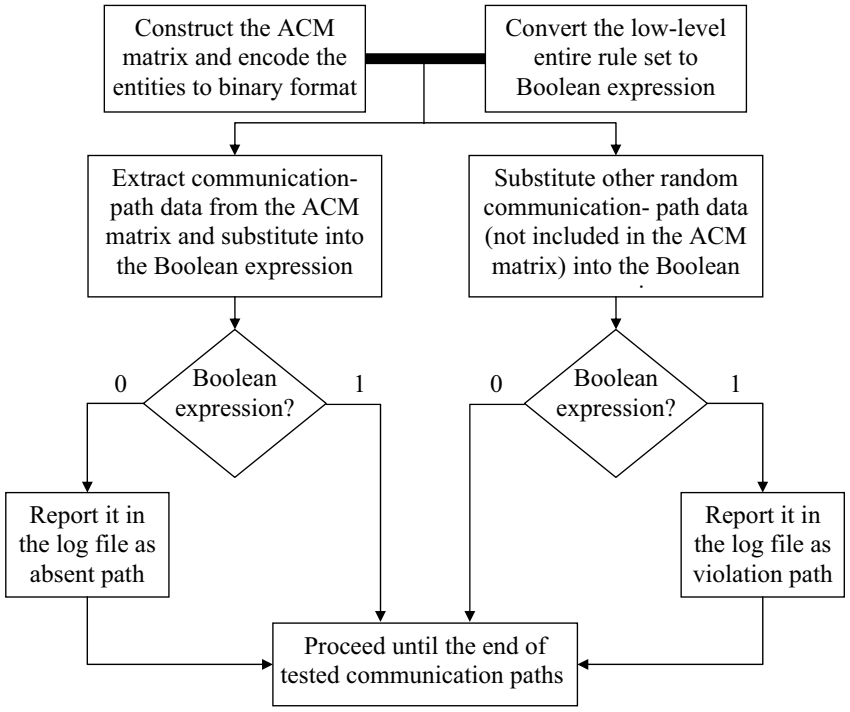

Fig. 6. Verification algorithm flow chart

described in Section 2.3. A set of just 22 rules was converted into a Boolean expression. The all possible communication paths are extracted from the ACM matrix and then substituted in the Boolean expression. Also, some other random communication paths are substituted in the Boolean expression. The log file of the tool reported ONEs for all communication paths extracted from the ACM matrix verifying the high-level policy is successfully implemented in the low-level. The log file reported ZEROs for all other random communication paths verifying no violation in the low-level rules.

4 CONFIGMAKER: A FIREWALL MANAGEMENT TOOLKIT

Based on RBNS model we have designed and implemented a firewall management toolkit. ConfigMaker stands for configuration maker as the toolkit is used to automatically generate the low-level firewall configuration rule-base from the high-level security policy.

This section provides a brief description of the toolkit and its using. However, the section concentrates mainly on how the RBNS model concepts are implemented and

how the automatic translation from high-level security policy to low-level firewall configuration rule-base is accomplished. The toolkit supports both stateless and stateful firewalls. It supports ACK bit checking and NAT technology. Detailed description of the toolkit is provided in [18, 19].

As shown in Figure 7, ConfigMaker architecture is composed of:

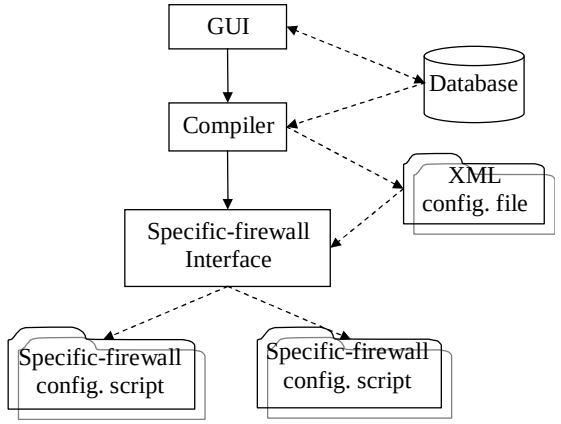

Fig. 7. Kit architecture

- GUI: through this graphical interface the administrator defines the RBNS entities and other topology related data, like firewalls' interfaces IPs, that have to be used to completely define the high-level security policy.
- Database: The database stores all entities data so that the administrator can later modify any entity of the model.
- Compiler: The compiler interacts with the data base to automatically generate the required security rules which are stored in output XML configuration file.
- XML configuration file: This file describes the firewall configuration that enforces the high-level security policy in a general syntax. General syntax means that this syntax is not tailored for specific commercial firewall. However, the XML file contains all the data required by most commercial firewall products.
- Specific-firewall interface: This interface module converts the XML configuration file to specific commercial firewall configuration script. Currently the toolkit supports two commercial firewall products – CISCO router and Linux kernel firewall.

The dashed arrows clarify the dataflow in the toolkit.

4.1 Using the ConfigMaker

This section uses the same running example defined in Section 2.3 to illustrate with the aid of some screen shots how the ConfigMaker toolkit can be used to trans-

late high-level security policy into intermediate-level that can be compiled easily to generate low-level firewall configuration file.

The toolkit user has to go through the following steps to define the RBNS entities:

- *Step 1*: hosts entity definition Figure 8.
- *Step 2*: services entity definition.
- *Step 3*: roles entity definition, Figure 9.

  The assignment of hosts and services to roles are also accomplished through this screen; Figure 9.
- *Step 4*: role-to-role assignment

4.1.1 RBNS Model Compilation

To generate the XML configuration file the administrator chooses "compile DB to XML" from Tools menu. A sample of the generated XML file is:

<communication_ desc>
<source>172.20.2.0</source>
<source_ mask>255.255.255.0</source_ mask>
<source_ port>gt.1023</source_ port>
<protocol>tcp</protocol>
<destination>0.0.0.0</destination>
<destination_ mask>0.0.0.0</destination_ mask>
<destination_ port>*</destination_ port>
</communication_ desc>
<communication_ desc>
<source>172.20.3.0</source>
<source_ mask>255.255.255.0</source_ mask>
<source_ port>gt.1023</source_ port>
<protocol>tcp</protocol>
<destination>0.0.0.0</destination>
<destination_ mask>0.0.0.0</destination_ mask>
<destination_ port>*</destination_ port>
</communication_ desc>

4.1.2 Firewall-Specific Compilation

In this step the administrator can choose between various commercial firewalls to generate its script configuration file. Actually, till now, our kit supports Cisco routers and Linux kernel firewall. A sample of the generated script for Cisco routers is:

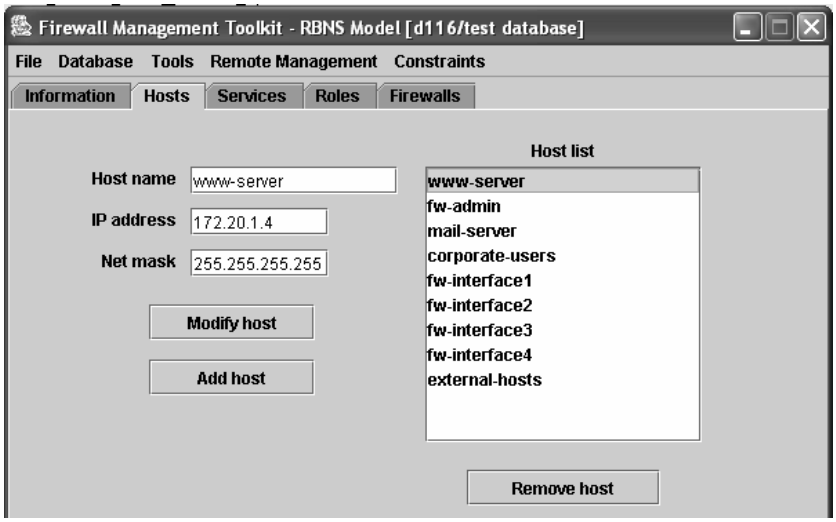

Fig. 8. Hosts definition

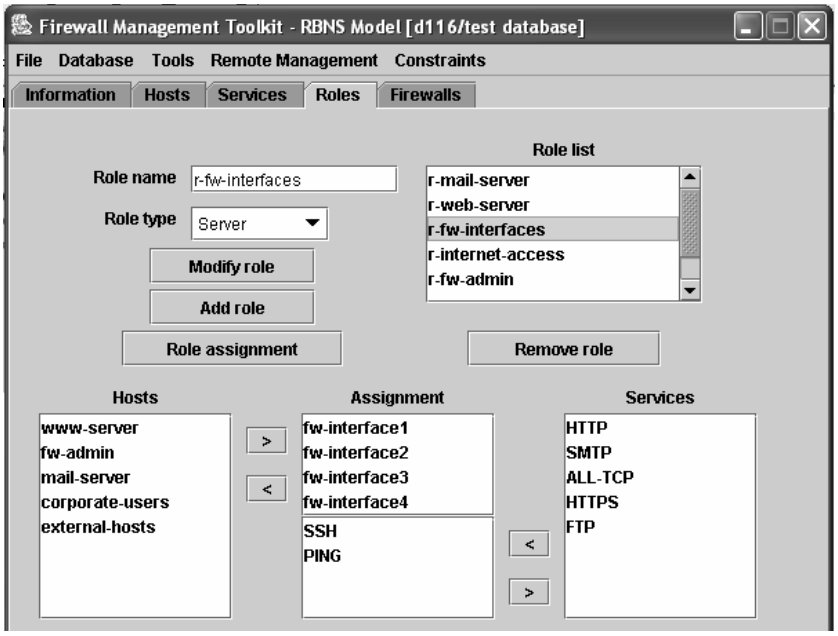

Fig. 9. Roles definition and assignment

```
interface external
    ip address 172.20.100.1 255.255.255.255
    ip access-group 100 in
    ip access-group 101 out
access-list 101 permit tcp host 172.20.2.0 0.0.0.255 host 0.0.0.0 255.255.255.255
      range 0 65535 log
access-list 101 permit tcp host 172.20.3.0 0.0.0.255 host 0.0.0.0 255.255.255.255
      range 0 65535 log
access-list 100 permit tcp host 0.0.0.0 255.255.255.255 host 172.20.1.4 0.0.0.0
      eq 80 log
access-list 100 permit tcp host 0.0.0.0 255.255.255.255 host 172.20.1.4 0.0.0.0
      eq 443 log
access-list 100 permit tcp host 0.0.0.0 255.255.255.255 host 172.20.1.5 0.0.0.0
      eq 25 log
```

5 CONCLUSION

We have presented a role-based network security model. We discussed in detail the various components of the model. Also, we introduced a mathematical formalization of the model. We have presented the constrained-level of the RBNS model. Constrained- RBNS level introduces the concept of constraints to represent a more complicated security policy. A detailed running example is used throughout the paper to illustrate the ability of our proposed model to be used as an intermediary-level between high-level security policy and low-level firewall rule-base. We have introduced a verification algorithm to verify the equivalence between high-level security policy and the corresponding low-level firewall configuration files. Finally we have presented a design and implementation of ConfigMaker, a firewall management toolkit based on RBNS model.

A more substantial extension to our work is to extend our model to upper layers of network OSI model. Another important direction is to generalize the verification algorithm to be used in the analysis of firewalls and routers access lists.

REFERENCES

[1] Pfleeger, C. P.: Security in Computing. Prentice-Hall Inc., 2nd Edition, 1997.

[2] Golmann, D.: Computer Security. John Wiley and Sons, 1999.

[3] Guttman, B.—Bagwill, R.: Implementing Internet Firewall Security Policy. NIST Special Publication, 800-XX, 1998.

[4] Stallings, W.: Cryptography and Network Security: Principles and Practice. Prentice-Hall Inc., Second Edition, 1999.

[5] Schuba, C. L.: On the Modeling, Design, and Implementation of Firewall Technology. Ph.D. Thesis, Purdue University, Dec. 97.

[6] Zwicky E. D.—Cooper S.—Chapman D. B.: Building Internet Firewalls. O'Reilly & Associates Inc., Second Edition, June 2000.

[7] Strebe, M.—Perkins, C.: Firewalls 24seven. Sybex Inc., 1999.

[8] Bartal, Y.—Mayer, A.—Nissim, K.—Wool, A.: Firmato: A Novel Firewall Management Toolkit. IEEE Symposium on Security and Privacy, May 9–12, 1999.

[9] Sandhu, R. S.: Role-Based Access Control. In Zelkowitz, editor, Advances in Computers, Volume 46, Academic Press, 1998.

[10] Ferraiolo, D. F.—Sandhu, R.—Gavrila. S.: Proposed NIST Standard for Role-Based Access Control. ACM Transactions on Information and Systems Security, Vol. 4, 2001, No. 3.

[11] Sandhu, R. S.—Ferraiolo, D. F.—Kuhn, R.: The NIST Model for Role-Based Access Control: Towards A Unified Standard. ACM, in Proceedings of 5th ACM Workshop on Role-Based Access Control, Berlin, Germany, July 2000.

[12] Sandhu, R. S.—Coyne, E. J.—Feinstein, H. L.—Youman, C. E.: Role-Based Access Control Models. IEEE Computer, Vol. 29, 1996, No. 2, pp. 38–47.

[13] Mayer, A.—Wool, A.–Ziskind, E.: Fang: A Firewall Analysis Engine. IEEE Symposium on Security and Privacy 2000, pp. 177–187.

[14] Guttman, J. D.: Security Goals: Packet Trajectories and Strand Spaces. FOSAD 2000, pp. 197–261.

[15] Guttman, J. D.: Filtering Postures: Local Enforcement for Global Policies. 1997 IEEE Symposium on Security and Privacy, Oakland, CA, pp. 120–129.

[16] Hazelhurst, S.: Algorithms for Analysing Firewall and Router Access Lists. Technical Report TR-Wits-CS-1999-5, July 1999.

[17] Stinson, D. R.: Cryptography: Theory and Practice. CRC Press, Boca Raton, FL, 1995.

[18] Rak, D.: Implementation of Security Model in Common Firewalls. Master Thesis, Slovak University of Technology, Faculty of Electrical Engineering and Information Technology, Department of Computer Science and Engineering, May 2003.

[19] Lehuta, M.: Security Policy of a Company. Diploma Thesis, Slovak University of Technology, Faculty of Electrical Engineering and Information Technology, Department of Computer Science and Engineering, May 2003.

Ahmed AbdAllah Hassan was born in Mansoura, Egypt in 1970. He received a BSc. in electronics engineering and a MSc. in electrical communications from Mansoura University in Mansoura, Egypt, in 1992 and 1995, respectively. He was working at Electronics and Communications department, Mansoura University, Egypt, as a teaching assistant (September 1992–August 1995) and as a lecturer assistant (August 1995 till now). In 2001, he received a doctoral scholarship from Slovak government to continue his PhD. studies. In September 2001, he enrolled in the department of computer science and engineering, Faculty of Electrical Engineering and Information Technology, Slovak University of Technology in Bratislava. Currently he is working as assistant professor at Electronics and Communications department, Mansoura University, Egypt. His research interests include network security and firewall management.

Ladislav Hudec, currently Associate Professor of computer science and engineering and Deputy Director of the Institute of Computer Systems and Networks, Faculty of Informatics and Information Technology, Slovak Technical University. In 1974 he received the Ing. diploma with summa cum laude in electronics from the Faculty of Nuclear Sciences and Physical Engineering, Czech Technical University, Prague; in 1985 he received the CSc. (Ph.D.) degree in computer machinery from the Faculty of Electrical Engineering, Slovak Technical University, Bratislava; in 1989 he was appointed Associate Professor. Since 1974 he is with the Slovak Technical University. During the period 1992–1993 he served as Director of the SARC – Centre for Advancement, Science and Technology, Bratislava. He is author or co-author over 30 scientific papers published in journals and proceedings of the conferences and over 50 technical papers in the field of fault tolerant computing, embedded systems, parallel computing and computer security. He led over 20 research grants and industrial contracts. He lectures on computer architecture and computer security. Dr. Hudec is founding member of the Slovak Centre of the IEE, he is member and President of the Slovak Association for Information Security (SASIB), member of the Information System Audit and Control Association (ISACA), holder of the CISA license. Since 1993 he serves as National Coordinator at the European Cooperation in Science and Technology (COST).