A Framework for Network Security System Design

JOÃO PORTO*AND PAULO LÍCIO DE GEUS
Institute of Computing
University of Campinas
Avenida Albert Einstein, 1251 Caixa Postal 6176
13083-970 Campinas - SP
BRAZIL
{jporto,paulo}@ic.unicamp.br http://www.ic.unicamp.br/∼paulo

Abstract: This work presents a framework for network system development that introduces a new phase in the usual procedure: the *network security design*. The main goal of this phase is to bridge the gap between high-level security requirement analysis and the low-level system implementation through the generation of a model of the network system architecture plus the security policies associated with the components of the model that have to enforce them. For this purpose, the design phase is composed by two complementary steps: an *architectural model* and a set of *design-level security policies*. The main advantages and desired characteristics of these models are analyzed; they are related to existing work in the area; and future research directions are pointed.

Key-words: network; security; security systems; firewalls; security policy; network architecture.

1 Introduction

The utilization of computers and data communication networks are notably growing, thus making them an essential resource to many kinds of organizations, as businesses, academic and governmental entities. This trend to ubiquitousness of computing equipment leads to growing geographic dispersion of users and devices, higher-speed channel needs and to a large degree of heterogeneity among the organizations' elements. These facts pose many new challenges to the traditional approaches for information security. This work focus on *network security systems*, defined in this context as a set of devices, software and technologies that collaborate to implement an organization's security policy.

As the security needs of organizations get more complex, so do the network security systems and the traditional approaches—like *firewalls* [5] [20]—have to go through several changes to get adapted. Incorporation of distributed mechanisms to enforce security [3], decentralized trust management [4], and the the widely spread use of cryptographic techniques (like IPSec [10] and *Virtual Private Networks* [20]), are examples of these changes.

The development process of network security systems usually goes through three phases: i) it starts with documented high-level security policies and controls based on some guideline manual (*e.g.* the ISO/IEC 17799 standard [8]); ii) it passes, preferably, by the formal specification of the security requirements [14], [11]; and then iii) goes to the implementation of the several enforcement mechanisms that composes the system [9]. We can note here a gap between the high-level specification security requirements and the implementation of the mechanisms to enforce them: the security designer goes from a high-level description straight to the implementation of a complex system, with different components that sometimes have completely discrepant idiosyncrasies in their configuration. This process is greatly error prone, and may lead, for instance, to not properly enforcing the required security policy, thereby introducing security holes and a very dangerous false sense of security.

This paper aims to bridge this gap by presenting a framework for the design of network security systems that introduces an additional phase: the *network security design*. This phase consists of the generation of a model of the network system architecture plus the security policies associated with the components of the model that have to enforce them. This is analogous to the software design in software engineering process: technologies that will be used in the implementation are chosen and modeled in a way that gives a holistic view of the system and the responsibilities associated with each mechanism to be implemented.

1.1 Paper Organization

In Section 2 the proposed framework is briefly described, and the *network security design* phase is further analyzed in Section 3. In Section 4 the main char-

*Supported by CAPES.

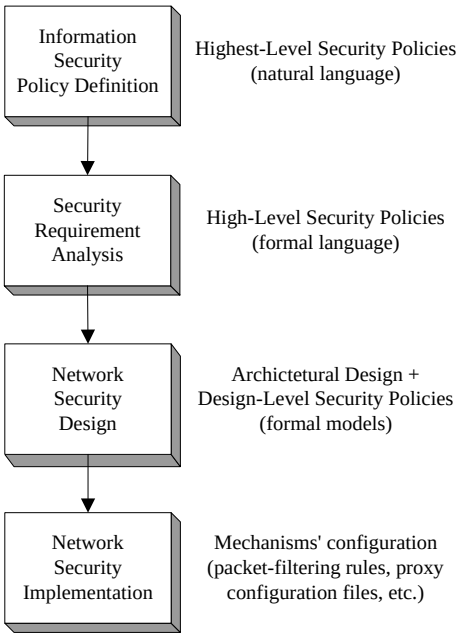

Figure 1: Proposed Framework

acteristics, advantages and limitations of the framework are presented and we also indicate some future work. In Section 5 we present the conclusions of this work.

2 Proposed Framework

The framework proposed in this paper is represented in Figure 1.

The first phase is the Information Security Requirements Definition, that consists of the elaboration of the Information Security Policy document with the goal of providing "management direction and support for information security" [8]. The network security system is just a small part (although an important one) of an organization's information security infrastructure and must be considered together with "several other fields, such as physical security, personnel security, operations security, communication security, and social mechanisms" (Icove, cited by Schuba and Spafford [15]). This is usually done by performing an enterprise business risk analysis following a guideline manual such as the ISO/IEC *Code of practice for information security management* [8]. The final product of this step is a document in natural language describing a set of information security policies and controls.

The term *security policy* is very overloaded—as noted by Sterne [18]— embodying several different levels of security policies. We shall name the policy generated in the first phase *highest-level security policies*.

The next step, Information Security Requirement Analysis consists of the formal representation of the highest-level security policies, which achieves a *high-level formal security policy base*. This phase is not always done, but there are benefits that makes them worth it, *e. g.* a formal model can be *analyzed* to detect conflicts between policies, and its formality eliminates ambiguities that may be present in the natural language highest-level policies. Despite of the several research efforts that have been done on formal specification of security requirements since the classical work of Bell and LaPadula [2], there still are important technical challenges in this field—as pointed by Rushby [14]—making this analysis not always easy.

Leiwo and Zheng [11] present a framework for dealing with high level policies with a formal approach, allowing conflict detection and harmonization in a layered fashion. Either using formal representation or not, the product of the analysis phase is a set of security policies that we will thereafter call *high level security policies*. Within the analysis process several levels of policies may co-exist [11], the referred set being the final refined and presumably consistent one.

From this second phase on we will be concerned only with technical security policies, more specifically *network domain security policies*, following the definition of Schuba and Spafford [15]: "a subset of a security policy, addressing requirements for authenticity and integrity of communication traffic (...), authorization requirements for access requests (...), and auditing requirements". These three types of security policies, namely *authenticity and integrity*, *confidentiality*, *access control*, and *auditing*, are the ones that will be enforced by network security mechanisms such as packet filters, proxy agents, cryptographic associations, and logging agents.

3 Network Security Design

The main objective of the Network Security Design is to transform the *high level formal security policy base* in a model of the network security system that will be used to enforce those policies. This model will represent: each technology that will be used; the interaction among different technologies; and the link between each high-level security policy and the correspondent components that will enforce it.

For this purpose, this phase is subdivided in two steps: the *Architectural Design* and the *Security Policy Design*, analyzed in the next sections.

3.1 Architectural Design

The Architectural Design goal is to establish the overall structure of the network security system, by represent-

ing the several components and technologies that will be used to build it. The designer will then choose the most appropriate technologies to compose the system, such as packet filtering routers, proxy agents, black-box firewall products, and cryptographic protocols.

All these system's components and their communication will be represented in an *Architectural Model* of the system. Along with a better understanding of the system that will be implemented, the Architectural Model also provides the means for establishing critical components and the impacts on the whole system generated by faults on each of these components.

Much research has been done on architectural design for software products and it also has been shown that it really works in practice. The architectural design is generally associated with *quality attributes* as performance, reliability, modifiability, maintainability and it and has been increasingly getting attention with the widely spread use of *components off-the-shelf* (COTS)[1]. It is clear that a network security system relies on several types of COTS that implement different security technologies (as mentioned above), so—as in software development—its architecture becomes an extremely relevant matter.

Other research efforts toward this direction are the *survivability* studies[2]. The easel language [7], for instance, is an approach for modeling the architecture of a system and evaluate impacts of failures of individual components on the system's goal. Although the survivability architecture approach overlaps to some extent the security architectural design as proposed in this work, they also differ significantly. As claimed by Fischer [7], "survivability is concerned primarily with system availability and mission fulfillment", while network security design is primarily concerned with *authenticity and integrity*, *confidentiality*, *access control*, and *auditing* requirements, as previously mentioned (Section 2).

Some of the characteristics of an efficient network security architectural design model can be learned from these and other studies. A highly desired characteristic is that the model generated be formal. The long-term research on *formal methods* have shown that a formal model has many advantages over an informal or semi-formal one. For instance, the model can be automatically *analyzed* instead of manually reviewed [14], and and it also allows code to be directly synthesized from the model. Ideas from the previously mentioned Bell and LaPadula model [2] and from other formal techniques such as Petri Nets [13] are very insightful for the conception of the new *network security architectural model*.

Another characteristic that can be borrowed from formal modeling techniques is the hierarchical decomposition, that is, the ability of the model to represent lower-level subsystems yet to be developed as a black-box component in a high-level model. Those subsystems could then be independently developed, in a *top-down* fashion development process. A *bottom-up* strategy may also be used: the development may begin with lower-level subsystems and then go to the model of higher-level functionalities using the predefined lower-level black-boxes. The hierarchical decomposition makes the model both more understandable and more scalable.

In the network security context, a *firewall*, for instance, could be modeled by relying on packet filtering black-boxes, and proxy agent black-boxes (these could also be expanded by relying on protocol-specific applications). Then, a firewall black-box could in turn be used as a component of the higher-level model of a network security system, together with other elements, such as decentralized trust management [4] and cryptographic associations management [16].

A complementary desirable characteristic, also gathered from pre-existing formal models, is the layered approach, that is, organizing the model in different layers according to some criteria (as in operating system design, for instance). The criteria in our case could be the level of sensitivity of information that is being protected. This is a common approach on information security, present in several techniques, such as the Bell and LaPadula multi-level security model [2], that allows for maintaining critical assets in inner layers, with higher degree of protection.

With these considerations, we can enumerate the basic components of the Architectural Model:

Network Entities: these are software and devices—such as routers, workstations, proxy agents, packet filters, VPN gateways—that constitute the system;

Communication Flows: they represent the data communication between entities of the model.

3.2 Security Policy Design

The second step in network security design is the *security policy design*, whose aim, as the name suggests, is to define a set of *design-level security policies*. These are the lower-abstraction-level policies in our framework and they are characterized as being near enough to technical implementation, but vendor- and device-independent.

[1] See Sommerville [17] for further reference.

[2] "*Survivability* is defined as the ability of a system to fulfill its mission in a timely manner in the presence of attacks, failures or accidents", by Ellison *et al.* [6].

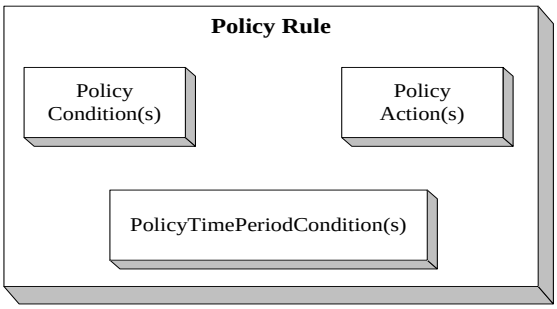

Figure 2: Overview of a Policy Rule

This definition converges with the policies' definition in the Policy Core Information Model (PCIM) from IETF's RFC 3060 [12], and as a matter of fact the objective of the PCIM is: "to serve as an extensible class hierarchy (through specialization) for defining policy objects that enable application developers, network administrators, and policy administrators to represent policies of different types" [12]. This indicates that besides being the basis for the so-called policy-based networking, involving notably QoS[3] parameters, the PCIM also can be a valuable reference for security systems' configuration.

This fact means that the *network security policy design* hereby proposed can use the PCIM classes to represent the design-level network security policies. The definition of policy that will be used within this phase is therefore conformant to RFC 3198: "a set of rules to administer, manage, and control access to network resources" [19], and the policy rule is illustrated conceptually in Figure 2.

A *Policy Rule* is a class that basically associates a set of conditions with a set of actions (Figure 2), representing the semantics "If Condition then Action" [12]. Figure 2 also shows that a policy rule "may be also associated with one or more policy time periods, indicating the schedule according to which the policy rule is active and inactive". The details of these classes are not relevant for this paper, see RFC 3060 [12] for a comprehensive explanation.

The PCIM matches the needs for the four types of network security requirements mentioned in Section 2: *authenticity and integrity*, *confidentiality*, *access control* and *auditing*. It is therefore naturally suitable to represent policies related to several security enforcement mechanisms: filtering rules, proxy agent configuration, cryptographic associations (like IPSec policies), keynote credentials, and logging requirements can be uniformly and conveniently addressed.

The link between the policy rules of this step and the elements of the architectural model, described in the previous section, can be done using another concept of PCIM, the *roles*. A role is defined as "a type of attribute that is used to select one or more policies for a set of entities and/or components from among a much larger set of available policies" [12]. So, the *network entities* of the architectural model are assigned roles that in turn are linked to policy rules.

So, at the end of this phase the overall structure of the network security system is precisely defined by the architectural model, and the functional responsibility of each system's component is determined by a set of associated security policies. By defining these policies in conformity with PCIM we get the additional advantage of representing both security policies and other types of policies (differentiated and integrated services) under a common standard.

4 Analysis of the framework and related work

The main advantage of the network security design is to give a uniform and concise view of the system to be implemented, as well as to make the transition from high-level policies to enforcement mechanisms smoother. It intends to supply the lack of a unified higher-level view of the several technologies and devices utilized in real-world network security systems. *Firmato*, from Bartal *et al.* [1] is also an effort toward this direction, but is restricted to *firewalls*, letting out other important components of present systems like cryptographic associations and trust management.

The firewall reference model, from Schuba and Spafford [15], also tries to unify the several different technologies used to implement firewalls in a conceptual model. It is not intended though as a tool to be used in practical development but rather for educational purposes.

In that way, a consistent and comprehensive methodology for network security design, as defined in this context, is still lacking at present scenario. This paper intends to be a conceptual seed for future development in the area.

While there are already defined standards for representing network policies in PCIM [12], as much work has to be done for translating the configuration of current network security technologies into the policy rule's formalism. To fit in the framework proposed in this paper the PCIM classes have also to be lightly adapted to include references to the higher-level policies that have originated a specific design-level policy. This would bring forth traceability of the policies actually implemented by enforcement mechanisms, that is, it would

[3] QoS is the acronym for *Quality of Service*, for further reference see RFC 3198 [19].

be possible to verify for each component of the security system what business need has originated it, and conversely for each business security requirement what technologies were used to implement it.

Besides this, unifying the representation of security policies to be implemented by several different and heterogeneous technologies offers the major advantage of improving the understandability of the system. It could also be used to partially automate the implementation of the system through a kind of "compiler" that takes as input the final design-level security policies, the system's architectural model and some vendor- and device-dependent information, and then generates the lowest-level configuration files (last box in Figure 1). This seems to be another interesting area for future research.

On the other side, the architectural model proposed here is only a conceptual project and, as opposed to the security policy design, it does not have any yet-established standard to be based on. But there are several architectural models from other areas (such as software engineering [17]) that can help in the task of building an open standard for architectural design of security systems. The studies in survivability architectures [7] can also be a valuable resource.

Along with the advantages gathered from architectural design in other areas previously mentioned (Section 3.1), another interesting one is to establish generic styles for system architecture, and then to reutilize it in other systems' development. It would aid in the development of an organization's security system, that could rely in models already defined by experts that are proved to be good. This kind of reutilization already happens in a certain way, but with a formal and precise modeling technique, as proposed in our framework, the exchange of information would be much more effective.

5 Conclusion

The framework presented in this paper (Figure 1) tries to unify the view of the network system's development cycle throughout the different levels of abstraction. The objective here is to give a conceptual basis for the network security design, the major focus of this work.

The network security design introduced here is composed by two complementary models: an *architectural model* and a set of *design-level security policies*. We have presented here some desired characteristics and main advantages of these models and pointed some directions for future research on a comprehensive network security design methodology.

References:

[1] Y. Bartal, A. J. Mayer, K. Nissim, and A. Wool. Firmato: A novel firewall management toolkit. In *IEEE Symposium on Security and Privacy*, Oakland, California, 1999.

[2] D. E. Bell and L. J. LaPadula. Secure computer systems: Mathematical foundations and model. Technical Report M74–244, MITRE Corporation, Bedford, MA, USA, 1975.

[3] S. M. Bellovin. Distributed firewalls. *;login: magazine, special issue on security*, November 1999.

[4] M. Blaze, J. Feigbaum, J. Ioannidis, and A. Keromytis. The keynote trust management system version 2. RFC 2704. Internet Engineering Task Force, September 1999.

[5] W. Cheswick and S. Bellovin. *Firewalls and Internet Security: Repelling the Wily Hacker*. Addison-Wesley, 1994.

[6] R. J. Ellison, D. A. Fischer, R. C. Linger, H. F. Lipson, T. A. Longstaff, and N. R. Mead. Survivable network systems: An emerging discipline. Technical Report CMU/SEI-97-TR013, Software Engineering Institute, November 1997.

[7] D. A. Fisher. Design and implementation of easel–a language for simulating highly distributed systems. In *Proceedings of MacHack 14, the 14th Annual Conference for Leading Edge Developers*, Deerborn, MI, USA, June 1999.

[8] International Organization for Standardization. Information technology – code of practice for information security management. ISO/IEC 17799:2000, 2000.

[9] S. Garfinkel and G. Spafford. *Practical Unix & Internet Security*. O'Reilly & Associates, 2nd edition, 1996.

[10] S. Kent and R. Atkinson. Security architecture for the internet protocol. RFC 2401. Internet Engineering Task Force, 1998.

[11] J. Leiwo and Y. Zheng. A framework for the management of information security. In *Information Security – Proceedings of the First International Workshop*, number 1396 in Lecture Notes in Computer Science. Springer–Verlag, 1997.

[12] B. Moore, E. Ellesson, J. Strassner, and A. Westerinen. Policy core information model—version

1 specification. RFC 3060. Internet Engineering Task Force, February 2001.

[13] T. Murata. Petri nets: Proprierties, analysis and applications. In *Proceedings of IEEE*, volume 77, no.4, April 1989.

[14] John Rushby. Security requirements specifications: How and what? In *Symposium on Requirements Engineering for Information Security (SREIS)*, March 2001.

[15] C. Schuba and Eugene H. Spafford. A reference model for firewall technology. In *Proceedings of the Thirteenth Annual Computer Security Applications Conference*, 1997.

[16] J. C. Sena. Um modelo para proteção do tráfego de serviços baseado em níveis de segurança. Master's thesis, University of Campinas, Campinas-SP, Brazil, 2002.

[17] I. Sommerville. *Software Engineering*. Addison-Wesley, 6th edition, 2000.

[18] D. F. Sterne. On the buzzword security policy. In *IEEE Symposium on Security and Privacy*, 1991.

[19] A. Westerinen, J. Schnizlein, J. Strassner, M. Scherling, B. Quinn, S. Herzog, A. Huynh, M. Carlson, J. Perry, and S. Waldbusser. Terminology for policy-based management. RFC 3198. Internet Engineering Task Force, November 2001.

[20] E. D. Zwicky, S. Cooper, and D. B. Chapman. *Building Internet Firewalls*. O'Reilly and Associates, 2nd edition, 2000.