Application of the XTT Rule-Based Model for Formal Design and Verification of Internet Security Systems*

Grzegorz J. Nalepa[1]

Institute of Automatics,
AGH University of Science and Technology,
Al. Mickiewicza 30, 30-059 Kraków, Poland
gjn@agh.edu.pl

Abstract. The paper presents a concept of support for the design and analysis of Internet security systems with a rule-based methodology. It considers a web security architecture, including a network and application-level firewall with intrusion detection systems. The XTT methodology allows for hierarchical design, and on-line analysis of rule-based systems. It is applied using the Unified Firewall Model, allowing for implementation-agnostic formal design and verification of firewalls. UFM extension aimed at integration with ModSecurity HTTP firewall are introduced.

1 Introduction

In order to provide security, complex Internet security systems combine number of advanced techniques from different domains [1]. In this heterogeneous environment finding an accurate approach to the problem of the design of such systems remains a challenge. This paper presents a concept of practical support of design and analysis of selected Internet security systems with a formal methodology, based on the classic rule-based programming paradigm [2,3]. The paper considers an extended security architecture for web systems security, including a network-level and application-level firewalls integrated with an intrusion detection system (IDS). The rule-based XTT methodology allows for hierarchical design, and on-line analysis of rule-based security systems [4]. The methodology is applied, using the Unified Firewall Model (UFM) [5,6], allowing for implementation-agnostic formal design and verification of firewalls. The paper presents extensions towards integration with an application-level HTTP firewall.

This paper is organized as follows: in Sect. 2 a short discussion of rule-based security systems is given, and the architecture of integrated web firewalls is discussed. In Sect. 3 important elements of rule-based systems (RBS) formalism are recalled, with the XTT design process briefly discussed. In Sect. 4 the UFM is presented, along with extensions. Elements of the visual UFM design are presented in Sect. 5. Directions for future work are presented in the Sect. 6.

* The paper is supported by the Hekate Project funded from 2007–2009 resources for science as a research project, and from AGH University Grant No.: 11.11.120.44.

2 Rule-Based Computer Security Systems

Network firewalls are the most common component of every network infrastructure. They are in fact optimized real-time rule-based control systems. A natural feedback for firewalls is provided by intrusion detection systems (IDS) that play a critical role in monitoring and surveillance. Network infrastructure based on network firewalls and IDS is often extended with application-level firewall solutions. These are especially important with complex web services, based on the HTTP protocol. Solutions such as *ModSecurity* [7], allow for HTTP traffic monitoring and filtering, with real-time intrusion detection. The practical development of integrated security systems is non trivial. The fact is, that it is usually a complex engineering task, often close to a hand craft activity. Improving this design and analysis process remains an area of active research.

The general idea behind this paper is to consider an integrated hybrid network and web application-level firewall model. A statefull network firewall serves as a gateway to a demilitarized zone (DMZ), where the main web server is located; it is integrated with an application-level firewall, working at the HTTP level. The infrastructure includes an intrusion detection system with several sensors for the subnetworks. In this approach an open implementation, called *ModSecurity* [7] (`www.modsecurity.org`) is considered. *ModSecurity* is an embeddable web application firewall available for the well known opensource *Apache2* webserver. Ultimately, the application of the UFM/XTT approach presented in this paper should develop into an integrated design methodology, combining both network-level and application-level firewall, and the intrusion detection system.

3 Formal Rule-Based Systems Analysis with XTT

Rule-Based Systems (RBS) [2,3] constitute a powerful AI tool for specification of knowledge in design and implementation of systems in many domains. In the formal analysis of RBS important aspects of the design and implementation are identified, such as *rulebase design*, and *inference engine implementation*. In order to design and implement a RBS in a efficient way, the knowledge representation method should support the designer introducing a scalable *visual representation*. As the number of rules exceeds even relatively very low quantities, it is hard to keep the rule-base consistent, complete, and correct. These problems are related to knowledge-base verification, validation, and testing. To meet security requirements a *formal analysis and verification* of RBS should be carried out; it usually takes place after the design. However, the XTT method allows for on-line verification during the design and gradual refinement of the system.

The main goal of the XTT approach is to move the design procedure to a more abstract, logical level. The design begins with the *conceptual design*, which aims at modelling the most important features of the system, i.e. attributes and functional dependencies among them. *Attribute-Relationship Diagrams* [3], allow for specification of functional dependencies of system attributes. An ARD diagram is a hierarchical conceptual system model at a certain abstract level.

The ARD model is the basis for the actual XTT model which allows for the *logical design*. The main idea behind *XTT* [8] knowledge representation and design method aims at providing a hierarchical visual representation of the decision tables linked into tree-like structure, according to the control specification provided. The logical design specification can be automatically translated into a low-level code, including Prolog, so that the designer can focus on logical specification of safety and reliability. The translation is the *physical design*. Selected formal system properties can be automatically *analyzed on-line* during the logical design, so that system characteristics are preserved. In this way XTT provides a clear separation of logical and physical design phases.

4 The Extension of the Unified Firewall Model

The *Unified Firewall Model* (UFM) [5] is a formal, implementation-free firewall model, build on top of the XTT methodology, providing a unified attribute specification for representing network firewalls. It is introduced as a middle-layer in firewall design process, enabling formal analysis of the created firewall. Generation of target language code for specific firewall implementation is achieved by defining translation rules from the abstract model into a specific implementation. In order to apply the UFM-based approach to firewall system design it is necessary to define: formal firewall system attributes, attributes domains, and syntax for expressing firewall policy in different implementations A full list of conditional firewall attributes is specified; they correspond to information found in network packets header. The specification is given in the Table 1, where each attribute is specified with: *Name*, *Symbol*, *Subset* (the position in inference process, specifying whether attribute is *input*, *output* or its value is defined during inference process – *middle*), and *Atomicity* (specifying whether attribute takes only *atomic* values from specified domain or also *sets* or *ranges* of these values).

Name	**Symbol**	**Subset**	**Domain**	**Atomic**
Source/Destination IP	aSIP/aDIP	input	Ipaddr	*set*
Protocol	aPROTO	input	Protocol	*set*
Destination port	aPORT	input	Port	*atomic*
Input/Output interface	aIINT/aOINT	input	Interface	*atomic*
ICMP type	aICMPT	input	Icmptype	*atomic*
ICMP error code	aICMPC	input	Icmperrcode	*atomic*
TCP flags	aTCPF	input	Tcpflags	*atomic*
Service	aSERV	middle	Service	*set*
Source/Destination group	aSGR/aDGR	middle	Group	*set*
Action	aACT	output	Action	*atomic*

Table 1. UFM Attribute Specification

In order to construct a practical firewall implementation, it is necessary to provide a formal translation from the unified model to particular implementa-

tions. Two open firewall implementations have been considered so far: Linux NetFilter and OpenBSD PacketFilter (PF). Full translation contained in [5] is long and detailed, and is out of scope of this paper. The goal of this research is to extend the UFM with attributes needed to describe an application-level HTTP firewall. Some natural specification restrictions are considered here, such as the fact that HTTP packets are contained only in the TCP packets, so the specification restricts HTTP-related rules only with use of the TCP. It can be observed, that traffic restrictions are in practise down-to-top; that is when designing the system the packet traversal in the network stack must be taken into account. A packet blocked at the network firewall level, basing on the IP/TCP attributes does not reach the HTTP level application firewall. So the translation to the target languages is not trivial, e.g. a rule: "block the traffic from the IP w.x.y.z" can be carried out at the IP level, so the HTTP level firewall never sees the packet. The traffic could also be allowed at the IP level, with the packet blocked at the HTTP level. Practically, at the unified level more detailed rules are considered. Let's consider three general firewall rules, each one more specific:

```
1) block the traffic from the IP w.x.y.z
2) block the traffic from the IP w.x.y.z to destination port HTTP
3) block the traffic from the IP w.x.y.z to destination port HTTP
                                        with BODY containg "cmd.exe"
```

In the extended UFM, the first rule is translated into 1 network level firewall rule; the second rule generates 2 rules, the second one for the HTTP level firewall (the fact is, that the rules are redundant, but both security policies are consistent); the third rule also generates 2 target rules, the first one identical to the previous ones, and the second for the HTTP level firewall which is more specific. In this way, there is a more fine-grained control over the security policy.

ModSecurity supports number of complex rule cases, the most important and common is the basic: `SecRule VARIABLES OPERATOR [ACTIONS]`. Where `VARIABLES` together with the `OPERATOR` part specifies rule precondition, and the `ACTIONS` specifies the decision (if omitted the default decision is made). In order to extend the UFM to support HTTP-level firewall, ModSecurity-specific attributes have been introduced. The translation procedure had to be extended, taking into account a two level firewall. The ModSecurity-specific part is trigerred when the destination port 80 (service www) is encountered. Let us now move to the design and verification procedure, using the extended UFM.

5 Visual UFM Design with XTT

The UFM has been developed for the integration with the XTT design process, which in case of generic RBS consists of several basic stages. On top of this process a Unified Firewall Model is built. In this case, the RBS designed is an *abstract firewall*. The UFM model provides a well-defined attribute specification for the 1st phase. Using this specification, in the 2nd phase, a general ARD model capturing functional relationships between UFM attributes, has been built [5]. Its simplified version is presented in Fig. 1. Basing on the conceptual design,

XTT tables are built. Every row of an XTT table corresponds to a firewall rule in the UFM. Having the XTT structure partially designed, it possible to conduct a verification of the firewall structure. The last stage is the physical design, where the XTT rules are translated into a given firewall implementation language, using formally predefined translation provided by the UFM.

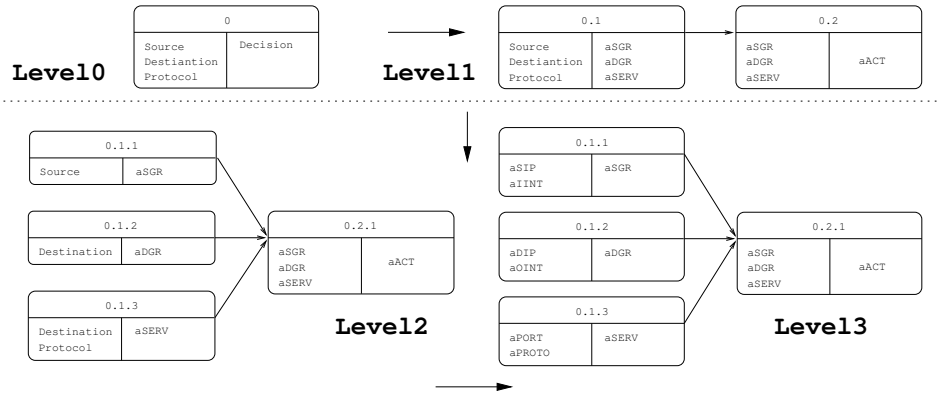

Fig. 1. The ARD Diagram for the UFM

Let us show how rules translation to the target implementation language is performed for the an example XTT rule.

```
Precondition(aSGR=inet, aDGR=fw_inet, aSERV=www),
Retract(aDIP=f_inet, aPort=80), Assert(aDIP=d_3w, aPort=8080),
Decision(aACT=dnat)
```

This is a rule for web proxy address translation. During the system attribute specification symbolic names of given IP networks and addresses are defined, in this case these are: `f_inet`, `fw_inet`, `d_3w`. The firewall rule for the NetFilter is:

```
iptables -t nat -A PREROUTING -s 0/0 -i eth0 -d 10.10.22.129
   -p tcp --dport 80 -j DNAT --to-destination 192.168.2.2:8080
```

The full translation is discussed in [5]. It is important to point out that the expressiveness of the UFM is as high as possible, so it is closer to the more expressive target language, e.g. OpenBSD PF. However, all of the UFM syntactic structures can be translated to any firewall language, provided that the the implementation has the features represented by the UFM. The whole design process proposed in this paper is presented in Fig. 2. An important part of this process is the analysis and verification framework. The XTT approach offers a possibility of automatic, on-line formal analysis of the firewall structure *during* the logical design. The analysis is accomplished by an automatic transformation of the XTT model into a corresponding code in Prolog. An extensible Prolog-based inference engine is provided, with number of analysis modules available, for important firewall features, including completeness, or determinism.

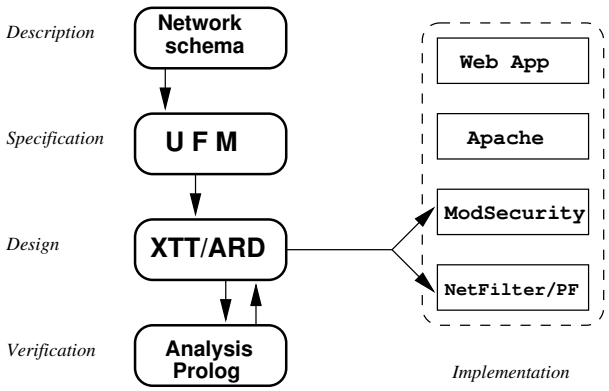

Fig. 2. UFM-based Design and Verification Process

6 Future Research

The original contribution of this paper is the extension of the formalization of the *Unified Firewall Model*, aimed at the application-level HTTP firewall, *ModSecurity* for the Apache webserver. This research should still be considered a work in progress. Future work includes: UFM application to intrusion detection systems, e.g. *Snort*, and improved verification with firewall-specific plugins. The XTT-backed UFM approach for security systems design and analysis allows for improving system quality, by introducing the formal verification during the visual design, while offering an abstract layer over common firewall implementations.

References

1. Garfinkel, S., Spafford, G., Schwartz, A.: Practical Unix and Internet Security. 3rd edn. O'Reilly and Associates (2003)
2. Jackson, P.: Introduction to Expert Systems. 3rd edn. Addison–Wesley (1999) ISBN 0-201-87686-8.
3. Ligęza, A.: Logical Foundations for Rule-Based Systems. Springer-Verlag, Berlin, Heidelberg (2006)
4. Nalepa, G.J., Ligęza, A.: Security systems design and analysis using an integrated rule-based systems approach. In Szczepaniak, P., Kacprzyk, J., Niewiadomski, A., eds.: Advances in Web Intelligence: AWIC2005. Volume LNAI 3528., Springer-Verlag (2005)
5. Budzowski, M.: Analysis of rule-based mechanisms in computer security systems. formulation of generalized model for firewall systems. Master's thesis, AGH-UST (2006) Supervisor: G. J. Nalepa, Ph. D.
6. Nalepa, G.J.: A unified firewall model for web security. In: accpeted for: the 5th Atlantic Web Intelligence Conference, Fontainebleau, France (2007)
7. Breach Security, I.: ModSecurity Reference Manual v2.1. www.breach.com. (2007)
8. Nalepa, G.J., Ligęza, A.: A graphical tabular model for rule-based logic programming and verification. Systems Science **31**(2) (2005) 89–95