A semantic based tool for firewall configuration*
(Extended abstract)

P. Adão
SQIG–IT, Instituto de Telecomunicações and
Instituto Superior Técnico,
Universidade de Lisboa, Portugal
Email: pedro.adao@ist.utl.pt

C. Bozzato, G. Dei Rossi, R. Focardi and F.L. Luccio
Dipartimento di Scienze Ambientali, Informatica e Statistica,
Università Ca' Foscari Venezia, Italy
Email: {cbozzato,deirossi,focardi,luccio}@dsi.unive.it

Abstract

The management and specification of access control rules that enforce a given policy is a non-trivial, complex, and time consuming task. In this paper we aim at simplifying this task both at specification and verification levels. For that, we propose a formal model of Netfilter, a firewall system integrated in the Linux kernel. We define an abstraction of the concepts of chains, rules, and packets existent in Netfilter configurations, and give a semantics that mimics packet filtering and address translation. We then introduce a simple but powerful language that permits to specify firewall configurations that are unaffected by the relative ordering of rules, and that does not depend on the underlying Netfilter chains. We give a semantics for this language and show that it can be translated into our Netfilter abstraction. We then present Mignis, a publicly available tool that translates abstract firewall specifications into real Netfilter configurations. Mignis is currently used to configure the whole firewall of the DAIS Department of Ca' Foscari University.

Protecting networks from external and internal attacks is a crucial task. System administrators rely on the usage of firewalls that examine the network traffic and enforce policies based on specified rules. However, implementing correct policies is a non-trivial task: if a policy is too weak the system may be attacked by exploiting its weaknesses, while if it is too restrictive legitimate traffic may be filtered out.

*This work has been partially supported by the PRIN 2010 Project *Security Horizons*, by FCT projects ComFormCrypt PTDC/EIA-CCO/113033/2009, and by PEst-OE/EEI/LA0008/2013.

Manually proving that implementations comply with a firewall policy is a too much time-consuming practice given that firewall rules are usually written in low-level, platform-specific languages, thus automatic tools for testing them have been developed [23, 35]. These tools however do not prevent users from introducing new flaws when modifying such policies. Some flaws may derive from the wrong order of firewall rules (consistency problems), and some others from the lack of matching rules for every packet that crosses the firewall (completeness problems). Another approach is to use a firewall design process that passes through different verification stages [27], but this is also time and resource consuming. Policy visualization tools have also been developed [24, 29, 30, 34], but they are not sufficiently helpful in dynamically changing networks where new services are added over time, as these typically impose very articulated firewalls composed of hundreds or even thousands of interacting rules. It is in fact very difficult to keep the number of rules small also because of redundancies (compactness problem).

In our opinion, there is an increasing need for formal and general tools to reason about the security of firewalls. Existing tools are however still far from the intended goal and we propose in this paper one further step in that direction.

Our contribution Netfilter is a firewall system integrated in the Linux kernel [33]. A firewall in Netfilter is implemented as a series of chains, tables and rules that are executed in a precise given order. In this paper we propose a model of Netfilter in which we abstract the concept of chains, rules and packets, and introduce the notion of state that records the information about exchanged packets. We give a semantics for this abstraction, close to the real one, that specifies how packets are dealt by the firewall in a specific state.

The novel features of our model allow us to introduce a new simple declarative language that specifies firewall policies by abstracting both the order in which rules are applied, and the different chains that Netfilter provides. The main advantage of this language is that transitions are defined in a single-step fashion, contrary to the multi-step semantics associated with the evaluation of the different tables of Netfilter. We then show how this language can be translated into our Netfilter abstraction, and we provide sufficient conditions under which a specification given in this language and its translation into Netfilter abstraction have the same effect on packets, both in terms of filtering and network address translation.

It is important to stress that, in our high level setting, any order of rules is acceptable and irrelevant for the semantics, whereas in Netfilter the order in which rules are written is fundamental and in general not interchangeable. Indeed, a well-known difficulty that reduces significantly the usability of Netfilter is that adding/deleting/-modifying rules is context-dependent and might potentially break the whole firewall policy. This makes it painful for system administrators to modify complex Netfilter configurations. Our firewall language, instead, makes it very easy to modify a configuration as the relative order of rules never affects the behavior of the generated

Netfilter rules. This language, in spite of its simplicity, is expressive and powerful enough to specify the most commonly used network security policies.

In order to demonstrate the feasibility and illustrate the simplicity and advantages of this approach we also present MIGNIS [1], a novel publicly available tool that translates, according to the aforementioned results, abstract firewall specifications into real Netfilter configurations. We then show an example of how MIGNIS can be used in a realistic, large scale, and non-trivial setting: MIGNIS is currently used to configure the firewall of the DAIS Department of the Ca' Foscari University of Venice. Using the overlap-detecting capabilities of MIGNIS and its simple syntax we were able to tackle the compactness problem by capturing many redundancies in the initial Netfilter configurations, and we could thus drastically reduce the number of configuration lines. Moreover, we have run some experiments by querying the MIGNIS specification and we were able to extract information such as the rules that affect packets from a certain host or whether a certain rule is already included or not in the specification.

While we were not the first ones presenting a language and a model that simplify firewall specification [1, 2, 3, 5, 6, 8, 12, 14, 18, 20, 22, 25, 36], to the best of our knowledge our model is the first that provides correctness guarantees about the generated configuration.

We believe this work may have impact in several communities. From a practical perspective we allow practitioners to specify firewall configurations in a simple understandable language with single-step semantics, and to generate the list of rules that implements that configuration in Netfilter. For theoreticians we propose a formalization of the behavior of a firewall that is amenable to verification of the intended security properties.

References

[1] High level firewall language. `http://www.hlfl.org`, 2003.

[2] Firestarter. `http://www.fs-security.com/`, 2007.

[3] Kmyfirewall. `http://www.kmyfirewall.org/`, 2008.

[4] Ipfilter. `http://coombs.anu.edu.au/~avalon/`, 2009.

[5] Netspoc: A network security policy compiler. `http://netspoc.berlios.de`, 2011.

[6] Pyroman. `http://pyroman.alioth.debian.org/`, 2011.

[7] Rule markup language. `http://www.ruleml.org/`, 2011.

[8] Firewall builder. `http://www.fwbuilder.org/`, 2012.

[1] Available for download at the address `https://github.com/secgroup/Mignis`.

[9] Frenetic, a family of network programming languages. http://www.frenetic-lang.org/, 2013.

[10] Oasis extensible access control markup language. http://xacmlinfo.org/category/xacml-3-0/, 2013.

[11] Packet filtering. http://www.openbsd.org/faq/pf/filter.html, 2013.

[12] Uncomplicated firewall. https://help.ubuntu.com/community/UFW, 2013.

[13] Chef. http://www.getchef.com/chef/, 2014.

[14] Iptables made easy, shorewall. http://www.shorewall.net/, 2014.

[15] LCFG large scale unix configuration system. http://www.lcfg.org/, 2014.

[16] pfSense, a proven open source firewall. http://www.pfsense.org/, 2014.

[17] With puppet enterprise, you pull the strings. http://puppetlabs.com/, 2014.

[18] C. J. Anderson, N. Foster, A. Guha, J.-B. Jeannin, D. Kozen, C. Schlesinger, and D. Walke. Netkat: Semantic foundations for networks. In *Proc. of the 41st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL 2014), to appear*. ACM, 2014.

[19] Anonymous. A semantic based tool for firewall configuration. http://dl.dropboxusercontent.com/u/7143532/iptables/ccs.pdf, 2013.

[20] Y. Bartal, A. Mayer, Nissim, and A. Wool: Firmato. A Novel Firewall Management Toolkit. *ACM Transactions on Computer Systems*, 22(4):1237–1251, 2002.

[21] F. Cuppens, N. Cuppens-Boulahia, J. Garca-Alfaro, T. Moataz, and X. Rimasson. Handling stateful firewall anomalies. In *SEC*, volume 376 of *IFIP Advances in Information and Communication Technology*, pages 174–186. Springer, 2012.

[22] F. Cuppens, N. Cuppens-Boulahia, T. Sans, and A. Miège. A formal approach to specify and deploy a network security policy. In *Formal Aspects in Security and Trust (FAST'04)*, pages 203–218, 2004.

[23] A. El-Atawy, T. Samak, Z. Wali, E. Al-Shaer, F. Lin, C. Pham, and S. Li. An automated framework for validating firewall policy enforcement. In *Proc. of the Eighth IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY'07)*, pages 151–160. IEEE, 2007.

[24] T. Göbel F. Mansmann and W. Cheswick. Visual analysis of complex firewall configurations. In *Proc. of the Ninth International Symposium on Visualization for Cyber Security*, VizSec'12, pages 1–8. ACM, 2012.

[25] M.G. Gouda and A.X. Liu. Structured firewall design. *Comput. Netw.*, 51(4):1106–1120, March 2007.

[26] A. Jeffrey and T. Samak. Model checking firewall policy configurations. In *Proc. of the 2009 IEEE International Symposium on Policies for Distributed Systems and Networks (POLICY '09)*, pages 60–67. IEEE Computer Society, 2009.

[27] A.X. Liu and M.G. Gouda. Diverse Firewall Design. *IEEE Transactions on Parallel and Distributed Systems*, 19(9):1237–1251, 2008.

[28] S. Martínez, J. Cabot, J. Garcia-Alfaro, F. Cuppens, and N. Cuppens-Boulahia. A model-driven approach for the extraction of network access-control policies. In *Proc. of the Workshop on Model-Driven Security*, MDsec '12, pages 5:1–5:6. ACM, 2012.

[29] S. Morrissey and G. Grinstein. Visualizing firewall configurations using created voids. In *Proc. of the Int. Workshop on Visualization for Cyber Security.* ACM, 2009.

[30] S. Morrissey, G. Grinstein, and B. Keyes. Developing multidimensional firewall configuration visualizations. In *Proc. of the 2010 International Conference on Information Security and Privacy.* ISRT, 2010.

[31] S. Pozo, R. Ceballos, and R. M. Gasca. Afpl, an abstract language model for firewall acls. In *Proc. of the international conference on Computational Science and Its Applications, Part II*, ICCSA '08, pages 468–483. Springer-Verlag, 2008.

[32] R. M. Marmorstein. *Formal Analysis of Firewall Policies.* PhD thesis, College of William and Mary, Williamsburg, VA, May 2008.

[33] R. Russell. Linux 2.4 packet filtering howto. `http://www.netfilter.org/documentation/HOWTO/packet-filtering-HOWTO.html`, 2002.

[34] T. Tran, E. Al-Shaer, and R. Boutaba. Policyvis: Firewall security policy visualization and inspection. In *Proc. of the 21st Large Installation System Administration Conference (LISA '07)*, pages 1–16. Usenix association, 2007.

[35] J. Walsh. Icsa labs firewall testing: An in depth analysis. `http://bandwidthco.com/whitepapers/netforensics/penetration/Firewall%20Testing.pdf`, 2004.

[36] B. Zhang, E. Al-Shaer, R. Jagadeesan, J. Riely, and C. Pitcher. Specifications of a high-level conflict-free firewall policy language for multi-domain networks. In *Proc. of ACM Symposium on Access Control Models and Technologies (SACMAT 2007).* ACM, 2007.