A Pattern for Whitelisting Firewalls (WLF)

Isaura Nathaly Bonilla Villarreal
Florida Atlantic University, Boca Raton, FL, USA, ibonill1@fau.edu

Eduardo B. Fernandez
Florida Atlantic University, Boca Raton, FL, USA, ed@cse.fau.edu

Maria M. Larrondo-Petrie
Florida Atlantic University, Boca Raton, FL, USA, petrie@fau.edu

Keiko Hashizume
Florida Atlantic University, Boca Raton, FL, USA, ahashizu@fau.edu

ABSTRACT

Firewalls are a useful defense for web sites and they are used in all kinds of systems, from laptops to large multiprocessors. Firewalls filter traffic to/from other web sites to prevent communication with potentially harmful locations. Firewalls typically use an open policy, where all sites are considered acceptable unless we explicitly blacklist them. To improve security we can communicate only with trusted sites using a whitelist (list of trusted sites). We present here a pattern for whitelisting firewalls that complements existing patterns for blacklisting firewalls. This pattern adds to our catalog of security patterns.

Keywords: Firewall, Whitelisting, pattern, security patterns, threat prevention.

1. INTRODUCTION

Firewalls are a useful defense for web sites and they are used in all kinds of systems, from laptops to large multiprocessors. Firewalls filter traffic to/from other web sites to prevent communication with potentially harmful locations. Firewalls typically use an open policy, where all sites are considered acceptable unless we explicitly blacklist them (classify them as not trusted). This is a useful policy for sites which, because of their purpose, need to reach the widest possible audience. If we want to deal with business partners, we can improve security by communicating only with them; this is the idea of the whitelisting firewall. We describe it as a security pattern. According to [Gam94], cited by [Fer08] a *pattern* is a packaged solution to a recurrent problem. A security pattern is a solution to a recurrent threat in computer systems described through a template that includes several sections [Fer13]. The sections include the threat being controlled, how the threat is handled using a solution described by a UML diagram, the consequences of using the pattern, and the real systems where the pattern has been used. The Whitelisting Firewall pattern adds to our catalog of security patterns [Fer13], and complements our patterns for blacklisting firewalls [Sch06]. The following sections describe the pattern and some conclusions.

2. WHITELISTING FIREWALL

2.1. INTENT

Define a list of sites with which we want to communicate. We want to prevent the client to get access to an external site that is considered untrustworthy when it is not in the list, or to stop traffic from an untrusted site.

2.2. CONTEXT

Users who connect to the Internet and to other networks who need to enforce their policies to all of the sites as one layer in a defense in depth strategy in their site.

2.3. PROBLEM

Some sites are insecure and may contain malware, which will attack our host or download further malware if we visit them. We need to establish a policy to decide which sites we want to communicate with and enforce this policy. The solution is affected by the following forces: Security: We only want to communicate with sites that are known and are trusted, this will increase security. Transparency: The users of the system should not be aware of this filtering. Overhead: Filtering should not reduce performance significantly. Usability: It should be easy to apply and manage filtering policies.

The solution is to implement a whitelisting firewall, which is a filtering mechanism that can enforce communication with only approved sites by keeping a list of acceptable interlocutors.

2.4. STRUCTURE

The class diagram for the Whitelisting Firewall will be published in the full paper in PLOP 2013. The **Whiteliting (WL) Firewall** intercepts the traffic between a **LocalHost** and a set of **ExternalHosts.** WLFirewall includes a **Whitelist,** which is composed of a set of ordered rules. Some of the rules may be **ExplicitRules** or may be implicit (default) rules. The complete description of the structure is described in the full paper.

2.5. DYNAMICS

We describe the dynamic aspects of the Whitelisting Firewall using a sequence diagram for one of its basic use cases. Basically, the whitelisting firewall applies the policy of a closed world, where every address is denied access unless explicitly permitted [EMA12, Fer13]. Figure 2 shows a sequence diagram for the Use Case "Filter an incoming request". A **node** requests to run some **service** or a**pplication**. The sequence diagram will be shown in the full paper. The **Whitelisting Firewall** receives this request and using a list of known and trusted sites checks if the request comes from or goes to a trusted site and allows the connection or denies the request.

2.6. IMPLEMENTATION

The Whitelisting Firewall should reside on a host computer and maintain a local set of rules. The institution must define these rules according to their policies. The whitelist is usually rather short and there is no need for hardware assistance to search it and the filtering should happen at the IP layer. It is possible to filter also at the application layer and stateful firewalls could accelerate filtering. There are specific approaches to use with the Whitelisting Firewall such as Gold Image and Digital Certificates. With the Gold Image for static systems, the list is created by first hashing a standard workstation image. After using an image to build the first whitelist, keeping it up to date will be the biggest challenge facing most groups. On the other hand, digital certificates are one of the most effective techniques to trust certain publishers of software. By their signature the certificates are automatically in the whitelist and are considered trusted [San12].

2.7. VARIANTS AND RELATED PATTERNS

We can combine Whitelisting and Blacklisting. For some addresses, we can apply Blacklisting, then Whitelisting for the remaining addresses. Another possibility is to apply first whitelisting and then blacklisting. The related patterns are presented in the full paper to be published in PLOP 2013.

3. CONCLUSIONS

Patterns solve specific problems in a context, the architectural patterns can make software architectures more extensible and efficiently. Thus the design and implementation of patterns in security has become important in current systems. Consequently, security has to meet the objective of: Confidentiality, integrity, and availability. For this reason, the use of patterns in security has been implemented in this paper and the whitelisting firewall pattern complements the three blacklisting firewall patterns of [Sch06]. Together with the IDS and VPN patterns that have a catalog to handle some aspects of security in networks [Fer13]. Finally, other patterns useful for network security are the cryptographic patterns that are shown in [Fer13].

REFERENCES

[Bit12] Bit9 products. Bit9 Security Platform.Retrieved from: http://www.bit9.com, Last accessed: November 17, 2012.

[Cor12] Coretrace Corp. Coretrace products. Retrieved from: http://www.coretrace.com/products-2/bouncer-overview#application. Last accessed: November 17, 2012.

[EMA12] EMA Corporation. Realistic Security, Realistically Deployed: Today's Application Control and Whitelisting. October 2012. Retrieved from: http://www.bit9.com. Last accessed: February 17, 2012.

[Fer08] E.B.Fernandez. Security patterns and a methodology to apply them.

[Fer13] E.B.Fernandez, Security patterns in practice: Building secure architectures using software patterns, to appear in the Wiley Series on Software Design Patterns.

[McA13] McAfee Products. McAfee Application Control. Retrieved from: http://www.mcafee.com/us/products/application-control.aspx. Last accessed: March 21, 2012.

[SAN] Products. Retrieved from: http://www.sans.org/reading_room/whitepapers/application/application-whitelisting-panacea-propaganda_33599. Last accessed: November 17, 2012.

[Sch06] M. Schumacher, E. B.Fernandez, D. Hybertson, F. Buschmann, and P. Sommerlad, Security Patterns: Integrating security and systems engineering . Wiley Series on Software Design Patterns, 2006.

Authorization and Disclaimer