Qualitative Analysis of Security Issues using Firewalls

Piyush Kashiyani[1], Karthik Chawda[2]

[1]M.E (CSE), Parul Institute of Technology
Vadodara, India
kashiyanipiyush@gmail.com
[2]Assistant Professor, CSE Department
Parul Institute of Engineering and Technology
Vadodara-India
er.karthikchawda.pg@gmail.com

Abstract: ***Technologies such as service-oriented architecture and cloud computing has enabled us to perform business services more efficiently and effectively. However, we still suffer from unintended security leakages by unauthorized actions in business services. Firewalls are the most widely deployed security mechanism to ensure the security of private networks in most businesses and institutions. The effectiveness of security protection provided by a firewall mainly depends on the quality of policy configured in the firewall. Unfortunately, designing and managing firewall policies are often error prone due to the complex nature of firewall configurations as well as the lack of systematic analysis mechanisms and tools. In this paper, we represent an innovative policy anomaly management framework for firewalls, adopting a rule-based segmentation technique to identify policy anomalies and derive effective anomaly resolutions. In particular, we articulate a grid-based representation technique, providing an intuitive cognitive sense about policy anomaly. In addition, we demonstrate how efficiently our approach can discover and resolve anomalies in firewall policies through rigorous experiments.***

Keywords: access control, multilevel security, network security, security management, visualization tool.

1. Introduction

Growth of Internet-based information sharing across distributed sites has led to the need for improved security. The demand for this increased security is complicated by network scalability, the insatiable curiosity of hackers, and the criminal activity of intruders bent on trying to get something for nothing. Information security is a critical need in literally every computer network domain of significance and value, e.g., with financial information, corporate proprietary information, contractual and legal information, human resource data, medical records, etc. The theme of this paper is to address such diversity of security needs among the different information and resources connected over a secure data network. Applications of such secure data networks include large-scale distributed databases, knowledge-base systems, collaborative workgroup environments, intranet or internet based systems, etc.

2. Software Development Activity

Software development is achieved through a collection of activities that lead to the building of a software product. Software development is traditionally divided into various phases, which we describe briefly in the following sections. For small projects, the phases are carried out in the order shown; for larger projects, the phases are interleaved or even applied repetitively with gradually increasing refinement.

3. Requirement Analysis

Extracting the requirements of a desired software product is the first task creating it. While customers probably believe they know what the software is to do, it may require skill and experience in software engineering to

recognize incomplete, ambiguous or contradictory requirements. Requirements analysis can itself be broken down in sub-activities. This phase is often the topic of processes itself, often referred to as the requirements process, or even requirements engineering process, the latter title being debatable. Requirements analysis methodologies have been created, one of the most popular are the use case driven methodology.

4. Technical Challenges

This problem is technically challenging. First, MSU cannot simply block VPN connections because, otherwise, the IBM representative may fail to perform his duties. Second, MSU cannot share its firewall policy with IBM. Firewall policies are typically kept confidential due to security and privacy concerns. Knowing the firewall policy of a network could allow attackers to easily spot the security holes in the policy and launch corresponding attacks. A firewall policy also reveals the IP addresses of important servers, which are usually kept confidential to reduce the chance of being attacked. Furthermore, from a firewall policy, one may derive the business relationship of the organization with their partners. Third, IBM cannot share the traffic in its VPN tunnel with MSU due to security and privacy concerns. For example, IBM may want to keep the IP address of its customer database server confidential to reduce the likelihood of being attacked. One main purpose of VPN is to achieve such confidentiality.

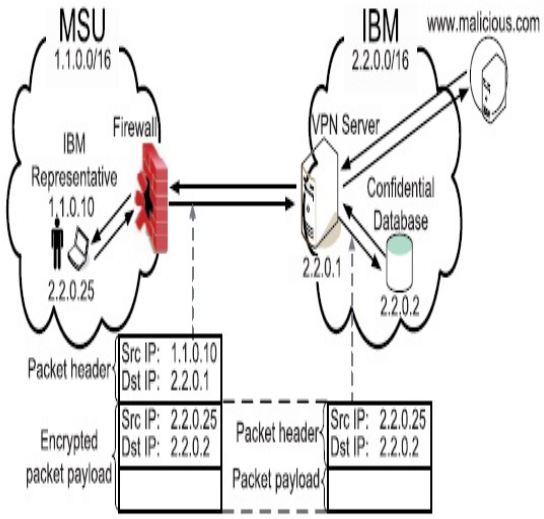


Figure 1: how firewall works in real-time system

5. Specification

Specification is the task of precisely describing the problem description, often in a mathematically rigorous way, in most cases actually building a more or less complete model of the problem to be solved. A wide array of specification techniques exists, and many application areas rely on dedicated Specification techniques. In practice, most successful specifications are written to understand and fine-tune applications that were already well-developed. Specifications are most important for external interfaces that must remain stable. This is often referred to as software interface specifications. Note that internal software interfaces are generally established in the design phase of development, which makes this kind of specification a design artifact. However, some systems have external software interfaces with other existing systems. The specification of these interfaces then becomes something that is very important to specify in the early phases of development, and is thus really considered a specification artifact.

6. Design

Design refers to conceptually determining how the software is to function in a general way without being involved in low-level operational details. Usually this phase is divided into two sub-phases, such as architectural design, detailed design, or algorithmic design. It can also be categorized into different focus such as graphical user interface (GUI) design, or database design, is depending on the nature of the application. Design is an extremely important phase in the development of the software, as it permits the developers to think in relatively abstract terms about the solution. This is in contrast with the implementation phase, where the solution is approached from a very concrete and often short-sighted point of view. Approaching the problem from an abstract point of view permits to 'see the big picture', and develop abstract models of the solution that can be easily modified as details are grafted and the solution become more and more concrete. At a certain point, the designed solution will be concrete enough so that the implementation (i.e. coding) phase can start.

7. Implementation

By Reducing a design to code may be the most obvious part of the software security job, but it is not necessarily the largest portion. A common Point in software development & security practice is that coding should be the main focus in software development. Such a misconception is a popular reason for low software quality or software project failure. Note that effective programmers have to deal with quality factors, and provide qualities such as code readability, maintainability, and often test the code as they are writing it.

8. Testing

Another common fallacy of software development is that testing is mostly about executing the application and making sure that it does not crash before it is made operational. Testing is basically about verifying the quality of the product, and validate that it meets its requirements. In fact, testing" is about the validation and verification of the various artifacts produced during the development of the software. Not only the code. For example, when requirements are established, one has to assess their quality in order to prevent that further phases be implementing a faulty or incomplete solution. The keyword "Testing" is most often referring to quality assurance of the code itself, but real software engineering will care About the quality of all artifacts produced not only the code or the final application.

9. A Firewall Security Model

For firewall architecture considerations, the network node input and output communications threads can be broken into classes (or class-filters) that follow the Open System Interconnection (OSI) stack. For example, the filtering policy deals primarily with accepting or rejecting devices based on their Medium Access Control (MAC) address. Similarly, the IP addresses can be filtered. Addresses are typically filtered by routers that work on the Ethernet source and destination addresses (each 48 bits long and often called the MAC address). Routers are more typically set up to filter addresses at the IP layer rather than the MAC layer. Protocols at OSI layers are also the focus of filtering since the protocol determination is typically in the same packet header along with the address (in the IP header). This has lent itself to providing a mechanism for filtering packets entering the Intranet at key gateways. The protocols are then related to port numbers that have services ready to process the data. Majority the workstations, the IP service has the facilities to reroute packets, to accept the packets for further processing, or to reject the packets. According to the Berkley Sockets architecture, an application service is registered with a port to accept data at the TCP or UDP level. Making the analogy along the lines of the OSI layers enables the use and definition of distinct roles in the end-to-end network security model. It is then natural to have the firewall instituted with such a design charter that aligns along

these boundaries. The different OSI layers could not only augment the understanding of the various charters of a firewall, but also provide a sense of completeness or comprehensiveness in offering the end-to-end security service.

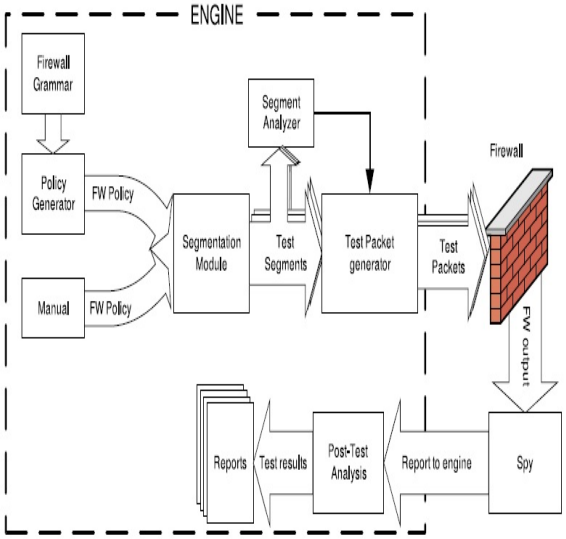


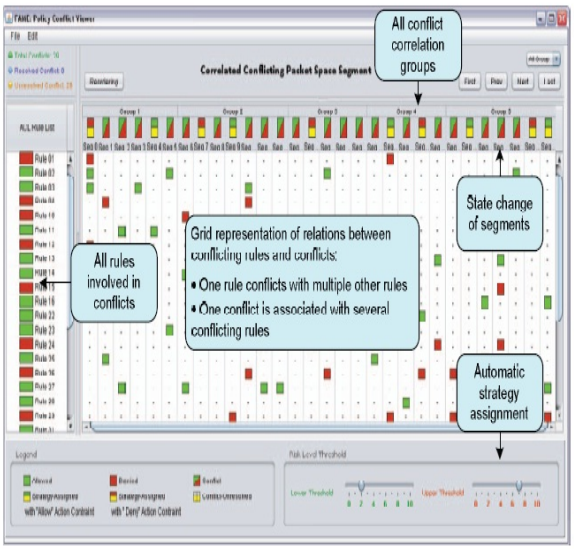
Figure2: Flowchart for Firewall

10. Design Multiple Firewalls

In Fig. 1 This network has a gateway router with two interfaces: interface 0, which connects the gateway router to the outside Internet, and interface 1, which connects the gateway router to the inside local network. The firewall for this local network resides in the gateway router. Suppose that the requirement specification for this firewall is given as follows: The mail server with IP address 192.168.0.1 can receive e-mail packets. The packets from an outside malicious domain 224.168.0.0/16 should be blocked. Other packets should be accepted and allowed to proceed. Suppose that we give this specification to two teams —Team A and Team B—which design the firewalls.

Implementation of Anomaly Management Framework in Framework Anomaly Management Environment (FAME)

Based on our policy anomaly management framework, it consists of six components: segmentation module, correlation module, risk. FAME was implemented in Java. Assessment modules, action constraint generation module, rule reordering module, and property assignment module. The segmentation module takes firewall policies as an input and identifies the packet space segments by partitioning the packet space into disjoint subspaces. FAME utilizes Ordered Binary Decision Diagrams5 to represent firewall rules and perform various set operations, such as unions ([), intersections (\), and set differences (n), required by the segmentation algorithm. A BDD library called JavaBDD, which is based on BuDDy package [25], is employed by FAME. Once the segmentation of packet space is identified, FAME further identifies different kinds of segments and corresponding correlation groups. In risk assessment module, Nessus [26] is utilized as a vulnerability scanner to identify the vulnerabilities within a conflicting segment. Network address space of each conflicting segment is fed into Nessus to get the vulnerability information of a given address space. Nessus produces the vulnerability information in a “nbe” format. The risk assessment module utilizes tissyn be script [27] to parse the Nessus results and store the vulnerability information to a vulnerability database. A risk calculator retrieves vulnerability information, such as CVSS base score and asset importance value, to calculate the risk level of each conflicting segment.



(a) Conflict Viewer for all conflicts

Figure 3: Implementation in FAME Model

11. Problem Regarding Firewalls

Higher security: If any hacker or higher level of threats attacks on system then it needs higher security for system.

Require high bandwidth for network transmission: UML diagram is stored/saved in different binary format. In a large enterprise application this diagram becomes huge in size.

Filtering of packets: it doesn’t need that each packet may filter every time so that secure data transmission can be varied.

Internal attackers: internal corrupted data may affect the system or information.

Lack of templatization at method/function level. Unable to generate fix syntax code.

12. Solution

1. Shadowing: A rule can be shadowed by one or a set of preceding rules that match all the packets which also match the shadowed rule, while they perform a different action.
2. Generalization: A rule is a generalization of one or a set of previous rules if a subset of the packets matched by this rule is also matched by the preceding rule(s) but taking a different action.

3. Correlation: One rule is correlated with other rules, if a rule intersects with others but defines a different action. In this case, the packets matched by the intersection of those rules may be permitted by one rule, but denied by others.
4. Redundancy: A rule is redundant if there is another same or more general rule available that has the same effect.

Why Firewalls?

For New Emerging computing technologies such as service-oriented architecture and cloud computing has enabled us to perform business services more efficiently and effectively. However, we still suffer from unintended security leakages by unauthorized actions in business services. Firewalls are the most widely deployed security mechanism to ensure the security of private networks in most businesses and institutions. The effectiveness of security protection provided by a firewall mainly depends on the quality of policy configured in the firewall. Network firewalls act as the first line of defense against unwanted and malicious traffic targeting Internet servers. Predicting the overall firewall performance is crucial to network security engineers and designers in assessing the effectiveness and resiliency of network firewalls against DDoS (Distributed Denial of Service) attacks.

13. Conclusion and Future work

Here, we have presented and validated an analytical model to study and analyze the performance of rule-based network firewalls. From the model, we have derived key features and performance measures of engineering and design significance. These key features and measures include throughput, packet loss, packet delay, and CPU utilization. The model can be used to measure the performance when the firewall is subjected to normal traffic flows as well as DoS attack flows targeting different rule positions.

Currently, research is in progress to enhance the policy generator to incorporate more options and capabilities to generate human like policies, and to target specific operational problems. Moreover, generating independent and orthogonal policies (with respect to the filtering algorithm) renders consecutive policy generation an increasingly hard problem. Studying the segmentation behavior for several policy styles needs further investigation.

References

[1] Privacy Preserving Collaborative Enforcement of Firewall Policies in virtual private network Publisher IEEE VOL. 22, NO.5
[2] Khalid Salah, Khalid Elbadavi, Raouf Boutaba: Performance Modelling and Analysis of Network Firewalls, IEEE VOL.9 N0.1 Mar-2012
[3] I. Herman, G. Melanc¸on, and M. Marshall, “Graph Visualization and Navigation in Information Visualization: A Survey,” IEEE Trans. Visualization and Computer Graphics, vol. 6, no. 1, pp. 24-43, Jan.-Mar. 2000.
[4] J. Burch, E. Clarke, K. McMillan, D. Dill, and J. Hwang. Symbolic model checking: 1020 states and beyond. J. Information and Computation, 98-2, 1992.
[5] W.R. Cheswick and S.M. Bellov, Firewalls and Internet Security, Repelling the Wily Hacker, Addison-Wesley, 1994
[6] D.B. Parker, “Information Security in a Nutshell,” Information Systems Security, spring 1997
[7] Nessus, http://www.nessus.org/, Mar. 2004
[8] World Wide Web consortium: http://www.w3.org,
[9] R. Haeni. Firewall penetration testing. Technical report, The George Washington University Cyberspace Policy Institute, 2033 K St, Suite 340N, Washington, DC, 20006, US, January 1997
[10] G. Shwed, System for Securing Inbound and Outbound Data Packet Flow in a Computer Network, US patent number 5,606,668, Feb. 1997. [11] Http: //www.cisco.com/warp/public/729/c3000/c3000_an. html, 14 Apr. 1997.
[11] http://en.wikipedia.org/wiki/Firewall
[12] A. Wool, “A Quantitative Study of Firewall Configuration Errors,” Computer, vol. 37, no. 6, pp. 62-67, June 2004.
[13] http://www.w3.org/FIREWALL/

Author Profile

Piyush Kashiyani Received the B.E. Degree in Computer Engineering from GEC, Gandhinagar in 2011 and M.E. degree in Computer Science and Engineering from Parul Institute of Engineering, Vadodara during 2011-2013.