Teaching about Firewall Concepts using the iNetwork Simulator

M. Ye and K. Sandrasegaran
Institute for Information and Communication Technologies (IICT)
University of Technology Sydney
P.O. Box 123, Broadway
NSW 2007, Australia
Email: melissa.ye@uts.edu.au, kumbes@eng.uts.edu.au

Abstract

The iNetwork Simulator is a software application created provides a user:friendly graphical interface for building and simulating basic communication networks. Such networks can comprise of devices such as workstations, servers, switches and routers.

This project enhances the iNetwork Simulator application by allowing firewalls to be simulated. Firewalls play in integral part in the security infrastructure of many organisations (and individuals). Being able to interact with a virtual firewall would benefit many students with an interest in the security-related aspects of communication networks.

The firewall component was implemented using the Microsoft .NET Framework and the C# programming language (the same platform used to originally develop the iNetwork Simulator application). The full software development lifecycle was followed during the course of this project.

1 Introduction

The iNetwork Simulator was originally conceived as a convenient and cost-effective method of allowing students to experiment with a realistic networking environment without the need of having to physically assemble and configure actual hardware. Its aim was "to improve the quality of education by supplementing the work done in the classroom with appropriate and realistic laboratory or practical work" (Trieu 2004).

A number of options, including that of adding a firewall component, were identified that would extend and enhance the application's usefulness as an educational tool. With the advent of the internet there has been an increasing focus on security.

"The global Internet security market is expected to grow at an annual 16 percent over the next five years to reach $58.1 billion by 2010...serious Internet security threats will continue to be the key market driver." (Walko 2005).

As a result, firewalls have since become an important component of many networks. Incorporating these devices into the iNetwork Simulator would therefore further benefit students.

2 Background

A firewall is a hardware- or software-based device that is used to allow or block traffic (i.e. packets of data) between different networks. The firewall is placed "between" these networks and examines all data that pass through it. A network administrator configures the rules on the firewall that specifies which types of traffic are allowed to pass through and which types of traffic will be blocked.

Firewalls form an integral part of an organisation's security policy as they can be used to prevent access to inappropriate content or prevent intruders from accessing a company's internal network from the internet.

2.1 Network & Application Layer Firewalls

Network layer firewalls are firewalls that operate in the network layer of the International Standards Organization (ISO) Open Systems Interconnect (OSI) networking model. They act as packet filters – allowing or blocking packets based on their source address, source port, destination addresses and/or destination port.

Application layer firewalls are those that operate in the application layer of the OSI networking model. They can block data such as certain websites, viruses and content deemed inappropriate by the network administrator.

Hybrid firewalls also exist. These are a combination of network layer and application layer firewalls.

2.2 Stateless & Stateful Firewalls

Firewalls can also be classified as *stateless* or *stateful*. Stateless firewalls treat each packet of data separately. Stateful frrewalls, on the other hand, monitor packets of data as part of a network connection. Thus, stateful firewalls can determine whether a packet is part of a legitimate connection. This provides an important advantage over stateless firewalls.

For example, FTP is one of a number of communication protocols that may open one or more random ports during a session. A packet destined for one of these random ports will be blocked by a stateless firewall, thus terminating a valid FTP session. The reason is that a stateless firewall will believe that the packet is for a non-standard service; it cannot tell that the packet is part of a legitimate connection.

3 Firewall Functionality

The firewall implemented for the iNetwork Simulator is a stateless firewall that operates in the networking layer of the OSI model. For consistency with the rest of the application, it behaves in a similar manner to other network devices, in terms of its interface and configuration mechanism.

Users create one or more rules, known as the ruleset or rulebase, to filter incoming packets. Outgoing packets are not filtered. A rule examines packets by protocol, source IP address, destination IP address, source port and destination port.

When the firewall intercepts a packet, each rule in the ruleset is executed until a matching rule is found. Depending on how the rule is defined, a matching packet has one of three possible actions taken on it:

- *Permit.* The packet is permitted to pass through the firewall.
- *Deny.* The packet is not permitted to pass through the firewall and the sender of the packet is notified by an ICMP Destination Unreachable packet.
- *Drop.* The packet is not permitted to pass through the firewall and the sender ofthe packet is not notified.

Packets that do not match any rule in the ruleset are permitted to pass through the firewall.

There are two ways to simulate the firewall component:

- Using the existing iNetwork Simulator utilities such as the DOS *ping* command; or
- Through a new interface specifically created for sending ICMP, TCP or UDP packets.

The simulation results are viewed using the existing Activity Log interface.

4 Development Environment

Due to the fact that the firewall component is a functional extension to the iNetwork Simulator, the development environment was identical to that used in the original project.

Thus, the Microsoft .NET Framework Version **1.1** was used as the development platform. C# was adopted as the programming language and code was written using the Microsoft Visual Studio .NET 2003 Integrated Development Environment (IDE).

5 Software Architecture

The iNetwork Simulator application is based on a "centralised controller architecture" (Trieu 2004). In other words, the application is comprised of a number of subsystems, all of which are driven from the user interface.

The architecture diagram for the application has been reproduced below in Figure **1.**

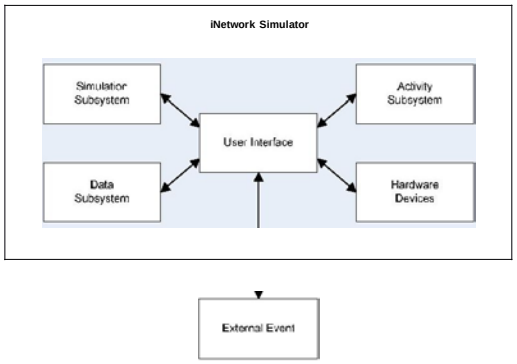

Figure 1 – Software Architecture

6 Design Considerations

Although the original software architecture hinted at discrete subsystems communicating with each other, a review of the application code has revealed significant coupling between these subsystems.

"One of the major problems with the iNetwork Software is its ability to be upgraded. Due to the existing design adopted, external developers will find it difficult adding additional network devices to the iNetwork Software." (Trieu 2004).

For example, classes representing the various hardware devices (such as routers and switches) contain both business logic and presentation logic – the former being associated with the Hardware Devices subsystem and the latter being associated with the User Interface subsystem. Strong coupling between subsystems makes an application difficult to maintain and extend.

This project did not seek to redress these issues, because it would involve restructuring a significant portion of the application. For this reason, different design strategies were adopted for new classes (created specifically for the firewall component) and changes to existing classes.

6.1 New Classes

New classes were designed with two major goals:

- To allow code to be maintained without difficulty.
- To allow the functionality of the firewall component to be easily extended in the future.

As such, oriented techniques and common design patterns were adopted. Design patterns are "descriptions of communicating objects and classes that are customized to solve a general design problem in a particular context" (Gamma, et al1995, p.3).

Figure 2 provides an illustration of this design rationale. **It** is a UML diagram that specifies the design of the protocols used by a firewall rule when examining packets.

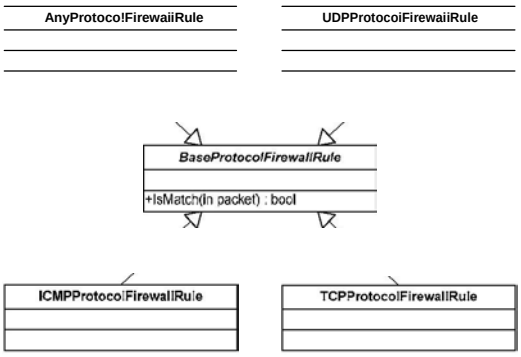

Figure 2 – Firewall Protocol Class Hierarchy

A requirement of a firewall rule is the ability to match a packet on the following protocols: TCP, UDP, ICMP or any protocol. Each class in Figure 2 encapsulates functionality specific to its domain. For example, the *ICMPProtocolFirewallRule* class contains code that deals with matching ICMP packets only.

This makes adding support for other protocols simple – create a new class that inherits from the *BaseProtocolFirewallRule* class. The other classes are not impacted at all, thus reducing the amount of uncertainty and time associated with all code changes.

6.2 Existing Classes

Where possible, changes to the existing classes were minimised for two main reasons:

- A project scope that was limited by time constraints.
- To ensure that the firewall component did not adversely impact the existing functionality of the application.

This choice was taken even if it meant perpetuating code requiring refactoring. Refactoring is time-consuming process but has the benefit of "changing a software system in such a way that it does not alter the external behaviour of the code yet improves its internal structure" (Fowler 1999, p.xvi).

7 Testing Process

The process used to test the firewall component is illustrated in Figure 3. Emphasis was placed on *system testing* and *regression testing.*

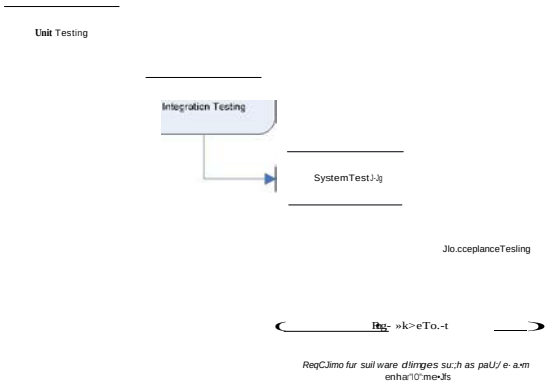

Figure 3 – Testing Process

System testing was used to evaluate whether the firewall component complied with the pre-defined requirements.

Regression testing was required to ensure that the existing functionality of the iNetwork Simulation application still worked. Particular attention was placed on functionality dealing with the transmission of packets of data between the network devices as most of the code changes to the existing classes impacted this area.

8 Screenshots

The screenshots below provide an example of a network with a firewall being simulated.

Figure 4 below provides an example of a network with a firewall together with a couple of windows that are used to configure the firewall's ruleset.

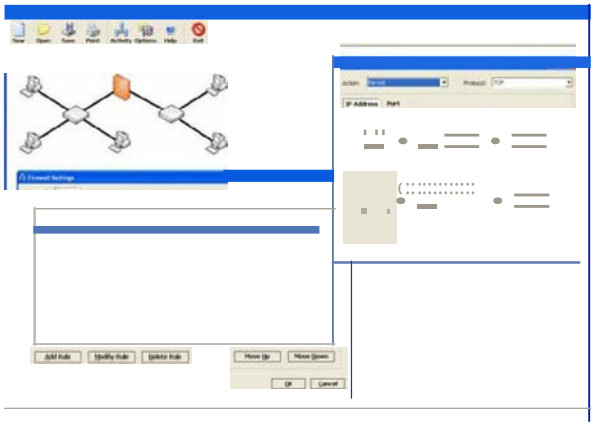

Figure 4 -Configuring the Firewall

Figure 5 below shows the results of *pinging* a workstation protected by the firewall from a workstation outside the firewall.

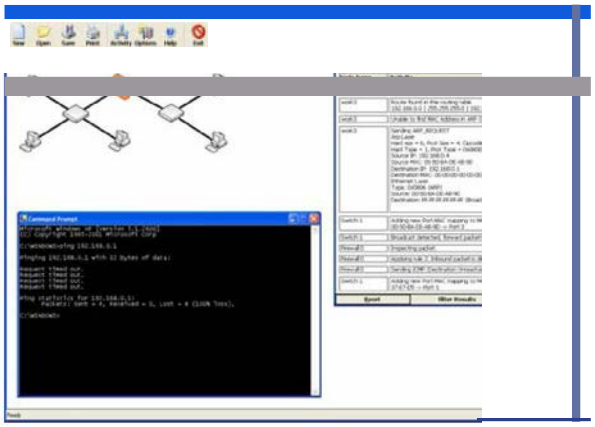

Figure 5 – Testing the Firewall Using the DOS *Ping* Command

Figure 6 below shows the results of using of the *Firewall Simulation* tool to send a TCP packet from a workstation outside the firewall to a workstation protected by the firewall.

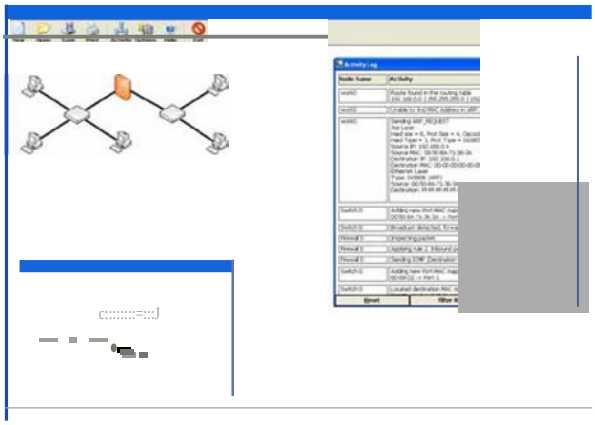

Figure 6 – Testing the Firewall from the *Firewall Simulation* Tool

9 Conclusion

The project has successfully evolved from the project plan to a functioning component of the iNetwork Simulator application.

By following a set of user guides (that have been documented as part of this project), students can immediately "jump in" and experiment with the firewall component. Most benefit, however, would be gained by students having some prior knowledge of network security.

Based on the experiences of this project, possible future enhancements to the firewall component and the iNetwork Simulator have been identified. These include:

- Refactoring the existing code. This would reduce the amount of complexity involved when adding new functionality to the application.
- Providing filtering capabilities for outgoing traffic.
- Building Network Address Translation (NAT) functionality into the firewall component.

10 References

Fowler, M. 1999, *Refactoring: Improving the Design of Existing Code*, Addison-Wesley, Indianapolis.

Gamma, E., Helm, R., Johnson R. & Vlissides, J. 1995, *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley, Indianapolis.

Larman, C. 2005, *Applying UMLand Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development*, 3rd edn, Prentice Hall **PTR,** Upper Saddle River.

Strassberg, K.E., Gondek, R.J. & Rollie, G. 2002, *Firewalls: The Complete Reference*, McGraw-Hill/Osborne, Berkeley.

Trieu, M.H. 2004, 'Interactive Learning Tool for Communication Networks', Capstone Project, University ofTechnology, Sydney.

Walko, J. 2005, 'Internet Security Market To Reach $58 Billion By 2010', *InternetWeek*, 15 November, viewed 11 December 2005, <http://internetweek.cmp.com/trends/173603249>.

Wikipedia, *Firewall*, Answers.com, viewed 31 July 2005, <http://www.answers.com/topic/packet-filter?method=6>.