DOKUZ EYLÜL UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

EMBEDDED FIREWALL SOFTWARE PACKAGE

by
Necati DEMİR

December, 2009
İZMİR

EMBEDDED FIREWALL SOFTWARE PACKAGE

A Thesis Submitted to the
Graduate School of Natural and Applied Sciences of Dokuz Eylül University
In Partial Fulfillment of the Requirements for the Degree of Master of Science in
Computer Engineering, Computer Engineering Program

by
Necati DEMİR

December, 2009
İZMİR

M.Sc THESIS EXAMINATION RESULT FORM

We have read the thesis entitled **“EMBEDDED FIREWALL SOFTWARE PACKAGE”** completed by **NECATİ DEMİR** under supervision of **ASST. PROF. DR. GÖKHAN DALKILIÇ** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Asst. Prof. Dr. Gökhan DALKILIÇ

Supervisor

(Jury Member) (Jury Member)

Prof.Dr. Cahit HELVACI

Director

Graduate School of Natural and Applied Sciences

ACKNOWLEDGMENTS

I would like to thank to my thesis advisor Assist. Prof. Dr. Gökhan DALKILIÇ for his suggestions, comments and guidance. I also would like to thank to my wife Mehtap for her patience and support and to my mother Sebahat and my father Metin for their support. This thesis has been supported by Scientific Research Projects, Dokuz Eylül University as the project number 2007.KB.FEN.001.

Necati DEMİR

EMBEDDED FIREWALL SOFTWARE PACKAGE

ABSTRACT

Firewalls became an indispensable part of computer networks. Corporations buy commercial firewalls or use open source firewall software packages to meet their need. Most of these are Linux or FreeBSD based and few of them are OpenBSD based, which has few features or are command line tools. In this thesis, creating an OpenBSD based firewall software package, which has a graphical user interface, an extensible infrastructure and targets embedded systems, has been aimed.

In this thesis, firstly firewall has been described and then previous studies about firewall software packages have been discussed. Since this study aims OpenBSD operating system and embedded hardware, the basics of OpenBSD and the design decisions for an embedded system have been discussed. Finally the infrastructure of the implementation of this study has been explained.

Keywords: Firewall, OpenBSD, PF, Network Security

GÖMÜLÜ DÜVENLİK DUVARI YAZILIM PAKETİ

ÖZ

Güvenlik duvarları, bilgisayar ağlarının vazgeçilmez bir parçası haline gelmiştir. Kurumlar, güvenlik duvarı ihtiyaçlarını karşılamak için ticari bir ürünler satın alırlar veya açık kaynak güvenlik duvarı yazılım paketleri kullanırlar. Bunların çoğu Linux veya FreeBSD tabanlıdır ve çok azı OpenBSD tabanlıdır ki bunlar daha az özellik içerir veya komut satırı uygulamasıdır. Bu tezde, grafik kullanıcı arabirimine, esnek bir altyapıya sahip ve gömülü sistemleri hedefleyen OpenBSD tabanlı bir güvenlik duvarı paketi oluşturmak hedeflenmiştir.

Bu tezde, ilk olarak güvenlik duvarı anlatılmıştır ve daha sonra güvenlik duvarı yazılım paketleri ile ilgili önceki çalışmalar incelenmiştir. Bu çalışma OpenBSD işletim sistemini ve gömülü donanımı hedeflediğinden, OpenBSD temelleri ve gömülü sistemler için dizayn kararları incelenmiştir. Son olarak bu çalışmanın uygulamasının alt yapısı açıklanmıştır.

Anahtar sözcükler: Güvenlik Duvarı, OpenBSD, PF, Ağ Güvenliği

CONTENTS

CHAPTER ONE
INTRODUCTION

Firewalls became an indispensable part of computer networks. Corporations buy commercial firewalls or use open source firewall software packages to meet their need. Most of these projects are Linux or FreeBSD based firewalls and few of them are OpenBSD based. Some of them only provide managing firewall rules, not other basic functions and some of them are command line tools. Beside of these, OpenBSD is the most secure operating system. Only two remote holes have been found for nearly ten years (OpenBSD, 2009).

There are few OpenBSD based projects and two of them are considerable; routerctl (routerctl, 2009) and PFW (PFW, 2009). routerctl is a command line application, it is not GUI based. Not having a graphical user interface is the lack of routerctl. PFW is GUI application but it provides only managing firewall rules. Not providing more features is the lack of PFW. It is really surprising that there is not a tool to manage basic features of OpenBSD with a graphical interface.

In this study, an embedded firewall software package for OpenBSD, which has a graphical user interface, an extensible infrastructure and targets embedded systems, is aimed. The implementation is called DemirFW. The term *embedded* is used to define the hardware, which has restricted memory and CPU power. The term *firewall software package* is used to define the entire management system including GUI, library, database, etc.

The studies in this thesis are grouped into four chapters:

1. **Firewalls:** The term firewall has been described in this chapter. This includes some basic headlines like firewall characteristics and firewall types. Additionally, previous works about firewall management projects have been discussed in this chapter.

2. **Using OpenBSD:** Since DemirFW is based on OpenBSD, first the basics of OpenBSD have been described. This chapter contains some basic headlines for configuring OpenBSD like configuring network and configuring DHCP server. This chapter shows how to do simple configurations without a management tool.

3. **Justifying Design Decisions of DemirFW:** DemirFW aims to run on embedded hardware. So some design decisions are made according to this. These decisions are needed for hardware performance; for example is chosen application consuming less memory? This chapter describes the reasons of the decisions.

4. **Implementation: DemirFW:** This chapter describes the infrastructure of DemirFW. DemirFW has been implemented in two parts; the server part and client part. The term *server part* describes the implementation of software in OpenBSD and the term *client part* describes the implementation of the software, which manages the server part. Python and Java has been used while implementing server part and client part, respectively. Also, some additional libraries have been used: SQLite and XML-RPC. DemirFW server part and client part communicates via remote procedure call protocol, so that client can be implemented in various programming languages.

CHAPTER TWO
FIREWALLS

2.1 What is Firewall?

Firewall is a system designed to prevent unauthorized access to or from a private internal network. It is typically set up at the periphery of a network, whether it is between the internal and external networks or between different segments of the internal network (for example between departmental LANs) (Genty, Kerouanton & Khor, 2000).

For firewalls to properly do their job, all network traffic must be routed through them. Having alternate network paths defeats the purpose of having a firewall. Dial-up connections within the network to independent ISPs is an example of alternate network path. The total security of a network depends on its weakest link. Having alternate paths may weaken the security posture and secrete an element of the unknown because of additional vulnerabilities opened through the alternate path (Genty, Kerouanton & Khor, 2000).

2.2 Why Need a Firewall?

With the rapid growth of interest in the Internet, network security has become a major concern to companies throughout the world. The fact that the information and tools needed to penetrate the security of corporate networks are widely available has only increased that concern (Cisco Systems, 2009).

Because of this increased focus on network security, network administrators often spend more effort protecting their networks than on actual network setup and administration. New tools that probe for system vulnerabilities assist in these efforts, but these tools only point out areas of weakness instead of providing a means to protect networks. Thus, as a network administrator tries to keep abreast of the wide number of security issues (Cisco Systems, 2009).

2.3 Misconceptions about Firewalls

A big misconception is that firewall provides all the protection required for a network. This is untrue. Firewalls can be primary security enforcer, but must be used in conjunction with other security elements. For example, if a Web server is badly configured that causes security problems, having a firewall will not be sufficient (Genty, Kerouanton & Khor, 2000).

2.4 Firewall Characteristics

The design goals for a firewall are listed: (Stalling, 1999)

- All traffic from inside to outside, and vice versa, must pass though the firewall. This is achieved by physically blocking all access to the local network except firewall.
- Only authorized traffic, as defined by the local security policy, will be allowed to pass.
- The firewall itself is immune to penetration. This implies that use of a trusted system with a secure operating system.

Originally, firewalls focused primarily on service control, but they have evolved to provide four controls: (Stalling, 1999)

<u>Service Control</u>: Determines the types of Internet services that can be accessed, inbound or outbound. The firewall may filter traffic on the basis of IP address and TCP port number; may provide proxy software receives and interprets each service request before passing it on; or may host the server software itself, such as a Web or mail service.

<u>Direction Control</u>: Determines the direction in which particular service requests may be initiated and followed to flow through the firewall.

User Control: Control access to a service according to which user is attempted to access it. This feature is typically applied to users inside the firewall perimeter (local users). It may also be applied to incoming traffic from external users; the latter requires some form of secure authentication technology, such as is provided in IPSec.

Behavior Control: Controls how particular services are sued. For example, the firewall may filter e-mail to eliminate spam, or it may enable external access to only a portion of the information on a local Web server.

2.5 Types of Firewalls

There are four types of firewalls: (Genty, Kerouanton & Khor, 2000)

- Packet Filter Firewalls
- Circuit Level Firewalls
- Application Layer Firewalls
- Dynamic Packet Filter Firewalls

2.5.1 Packet Filter Firewalls

Packet filter firewalls applies a set of rules to each incoming IP packet and then forwards or discards the packet (Figure 2.1 Packet Filtering Firewall (Boyer, 1997)). This type of firewall is typically configured to filter packets going in both directions (from and to the internal network). Filtering rules are based on fields in the IP and transport (e.g., TCP or UDP) header, including source and destination IP address, IP protocol field (which defines the transport protocol), and TCP or UDP port number (which defines an application such as SNMP or TELNET) (Stalling, 1999).

The packet filter is typically set up as a list of rules based on matches to fields in the IP or TCP header. If there is a match to one of the rules, that rule is invoked to determine whether to forward or discard the packet. If there is no match to any rule, than a default action is taken (Stalling, 1999).

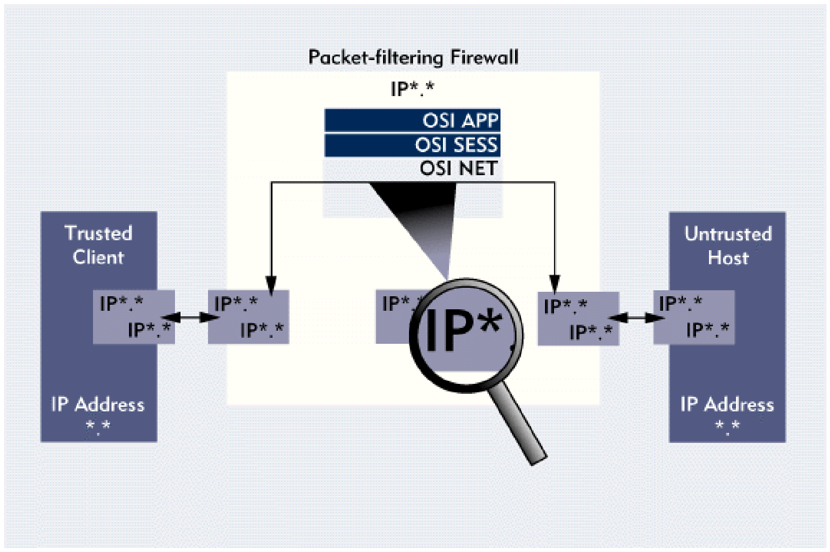


Figure 2.1 Packet Filtering Firewall (Boyer, 1997)

2.5.2 Circuit Level Firewalls

Circuit level firewalls operate at transport layer. They support only TCP protocol because this type of firewall validates that a packet is either a connection request or a data packet belonging to a connection (See Figure 2.2). The firewall forms a type of state table to monitor the handshake of each new connection. After the handshakes are complete, the firewall inspects the packets to determine if the packets are to be allowed through or not, based on a rule-set. Once allowed, all network packets associated with this connection are allowed without further checks (Genty, Kerouanton & Khor, 2000).

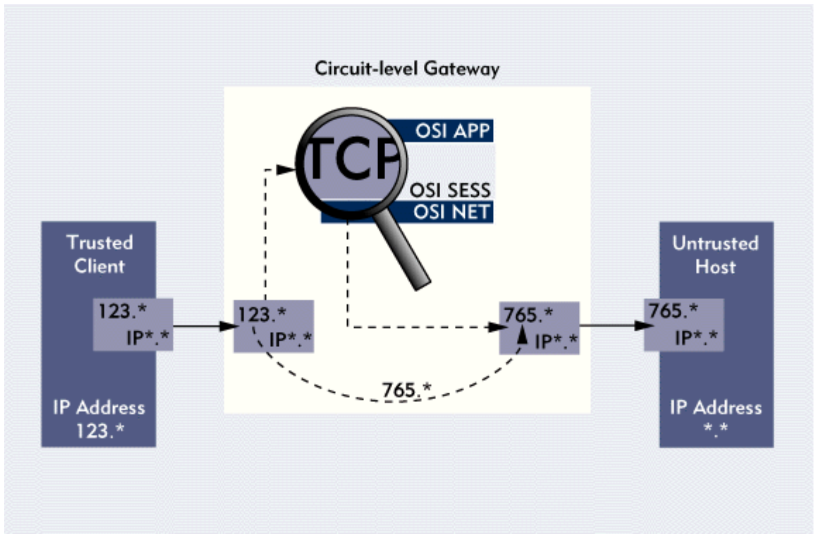


Figure 2.2 Circuit Level Firewall (Boyer, 1997)

2.5.3 Application Layer Firewalls

Application layer firewalls act as a proxy between internal and external users. No direct connections are made between internal clients and external server. (See Figure 2.3) The advantage of these firewalls is that the firewall has better control over what access to grant. For example, an FTP proxy can be used to restrict commands, for example, allowing get but disallowing put commands (Genty, Kerouanton & Khor, 2000).

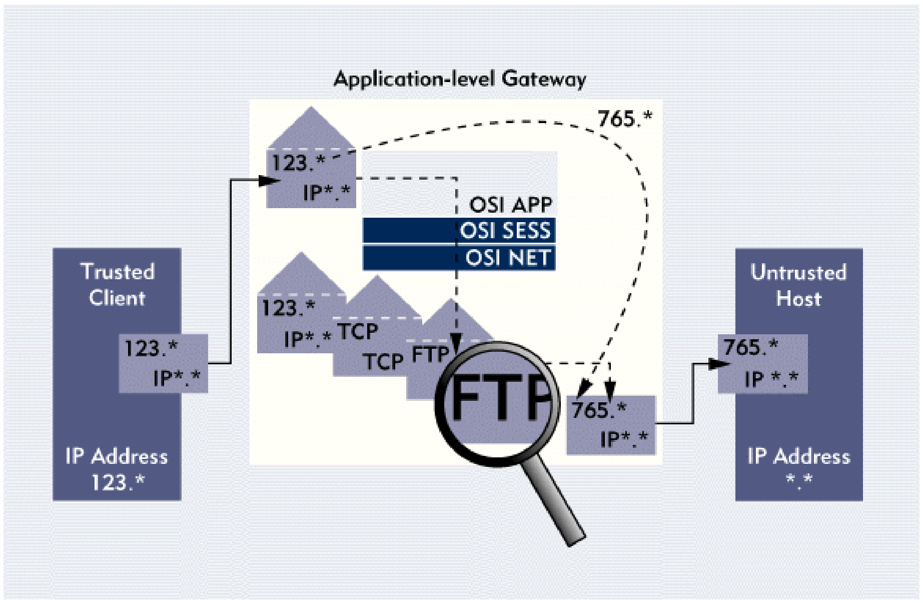


Figure 2.3 Application Level Firewall (Boyer, 1997)

The disadvantage of application layer firewalls is that they are too tied to specific protocols. An FTP proxy is different form a HTTP proxy and so on. Each specific protocol requires its own specific proxy, and when new protocols come out, new proxies have to be developed (Genty, Kerouanton & Khor, 2000).

2.5.4 Dynamic Packet Filter Firewalls

Dynamic packet filter firewalls provide additional protection for the UDP transport protocol by associating the UDP packets with a virtual connection table within the firewall. This connection table keeps track of the connection at both sides of the firewall. (See Figure 2.4) So, for example, if an attacker attempts to simulate a reply to a request from internal host, it will fail. The dynamic packet filter firewall will know the internal host did not initiate the request, and, therefore, will not allow the simulated reply in (Genty, Kerouanton & Khor, 2000).

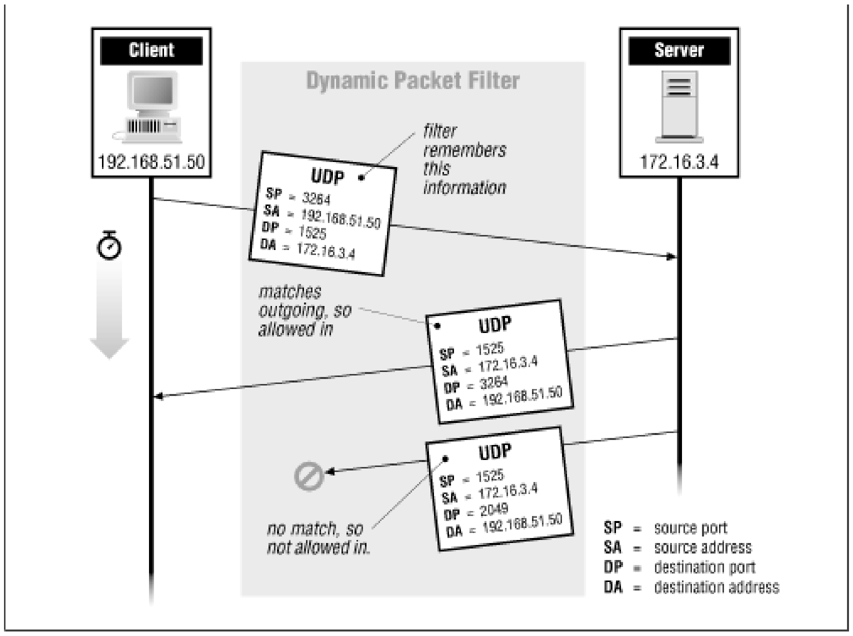


Figure 2.4 Dynamic Packet Filter (Zwicky, Cooper, & Chapman, 2000)

2.6 Open Source Firewall Software Packages and Firewall Management Projects

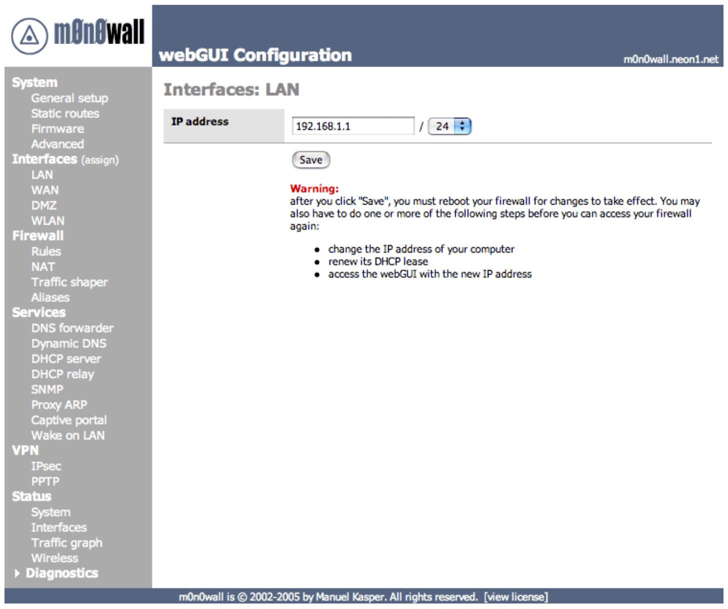
This study aims to develop a firewall software package for OpenBSD. First, some open source variants will be discussed.

2.6.1 m0n0wall

m0n0wall is a project aimed at creating a complete, embedded firewall software package that, when used together with an embedded PC, provides all the important features of commercial firewall boxes (including ease of use) at a fraction of the price. m0n0wall is based on a bare-bones version of FreeBSD, along with a Web server, PHP and a few other utilities. The entire system configuration is stored in one single XML text file to keep things transparent (m0n0wall, 2009). A screenshot of m0n0wall can be seen on Figure 2.5.

2.6.2 pfSense

pfSense is a free, open source customized distribution of FreeBSD tailored for use as a firewall and router. In addition to being a powerful, flexible firewalling and routing platform, it includes a long list of related features and a package system allowing further expandability without adding bloat and potential security vulnerabilities to the base distribution (pfSense, 2009). A screenshot of pfSense can be seen on Figure 2.6.



Interfaces: LAN

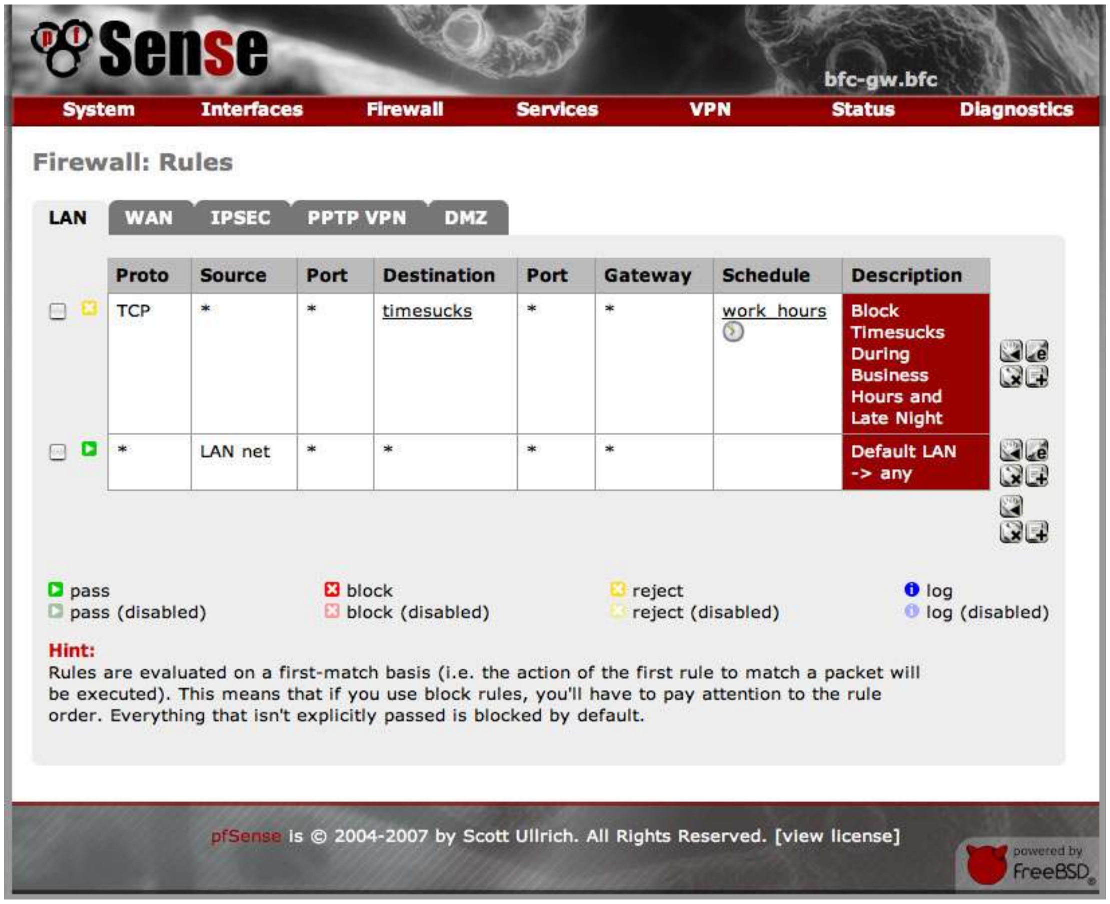
IP address: 192.168.1.1 / 24

Warning:
after you click "Save", you must reboot your firewall for changes to take effect. You may also have to do one or more of the following steps before you can access your firewall again:

- change the IP address of your computer
- renew its DHCP lease
- access the webGUI with the new IP address

m0n0wall is © 2002-2005 by Manuel Kasper. All rights reserved. [view license]

Figure 2.5 Screenshot of m0no0wall (m0n0wall, 2009)



Firewall: Rules

Proto	Source	Port	Destination	Port	Gateway	Schedule	Description
TCP	*	*	timesucks	*	*	work_hours	Block Timesucks During Business Hours and Late Night
*	LAN net	*	*	*	*		Default LAN -> any

🟩 pass	🟥 block	🟨 reject	● log
pass (disabled)	block (disabled)	reject (disabled)	log (disabled)

Hint:
Rules are evaluated on a first-match basis (i.e. the action of the first rule to match a packet will be executed). This means that if you use block rules, you'll have to pay attention to the rule order. Everything that isn't explicitly passed is blocked by default.

pfSense is © 2004-2007 by Scott Ullrich. All Rights Reserved. [view license]

powered by FreeBSD

Figure 2.6 Screenshot of pfSense (pfSense, 2009)

2.6.3 *Firewall Builder*

Firewall Builder is a GUI firewall configuration and management tool that supports iptables (netfilter), ipfilter, pf, ipfw, Cisco PIX (FWSM, ASA) and Cisco routers extended access lists. Both network administrators and hobbyists managing firewalls with policies more complex that is allowed by simple Web based UI can simplify management tasks with the application. The program runs on Linux, FreeBSD, OpenBSD, Windows and Mac OS X and can manage both local and remote firewalls (Firewall Builder, 2009). A screenshot of Firewall Builder can be seen on Figure 2.7.

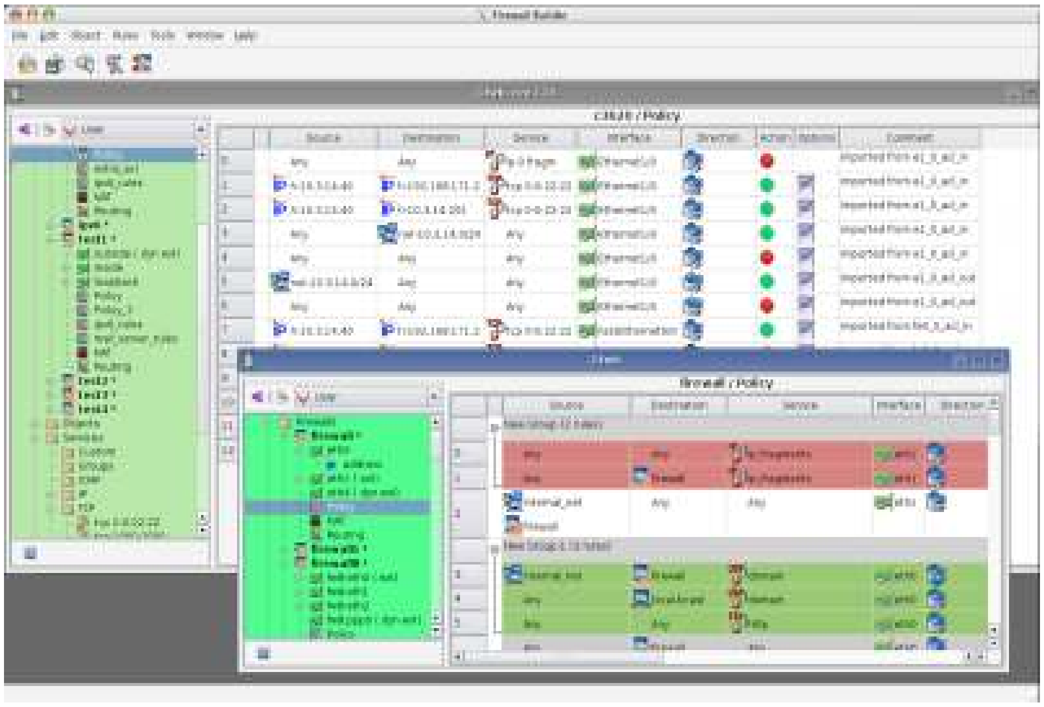
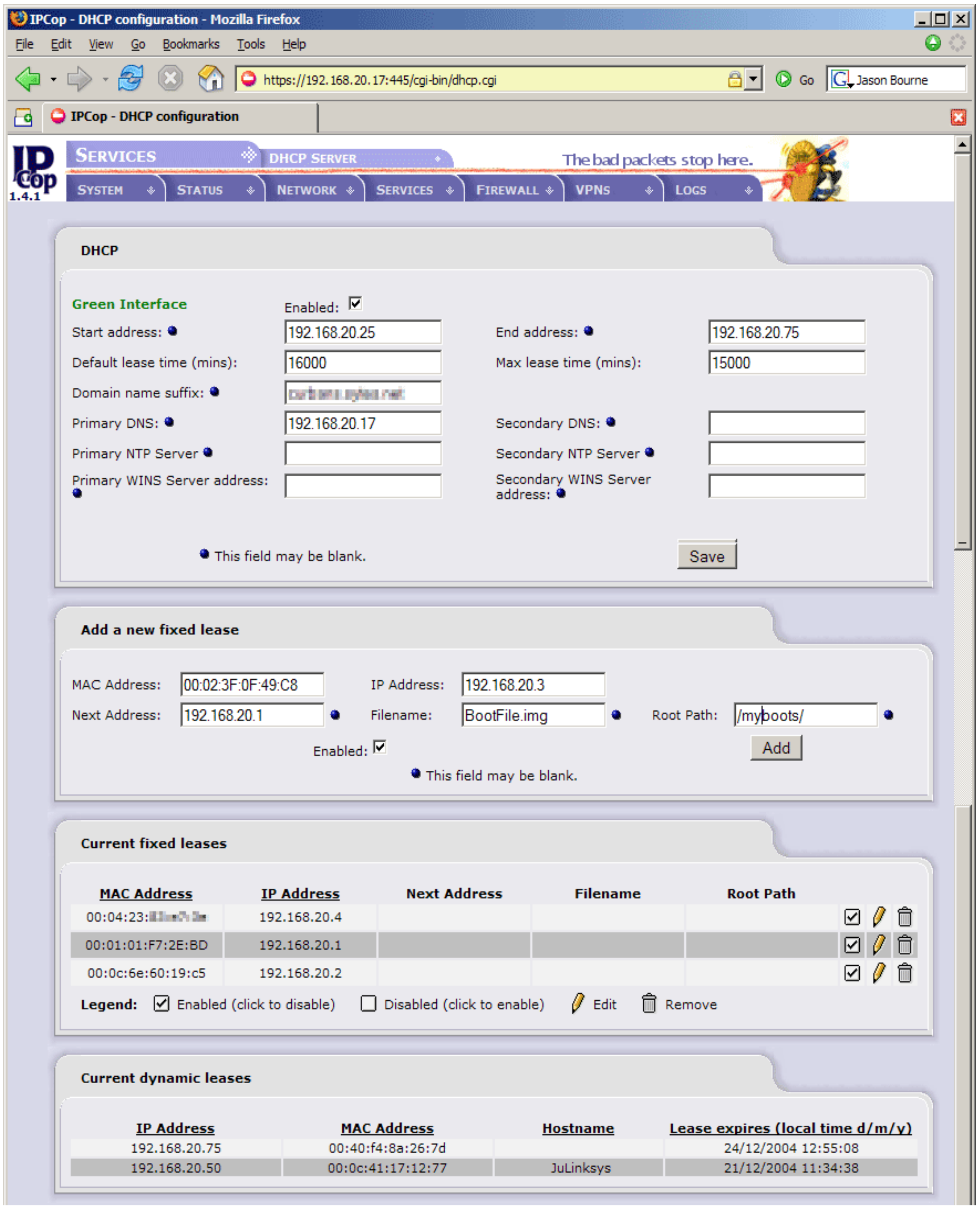


Figure 2.7 Screenshot of Firewall Builder (Firewall Builder, 2009)

2.6.4 IPCop

IPCop Firewall is a Linux firewall distribution geared towards home and SOHO (Small Office/Home Office) users. The IPCop interface is very user-friendly and task-based. IPCop offers the critical functionality of an expensive network appliance using stock, or even obsolete, hardware and Open Source Software (IPCop, 2009). A screenshot of IPCop can be seen on Figure 2.8.


IPCop - DHCP configuration - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
https://192.168.20.17:445/cgi-bin/dhcp.cgi
Go
Jason Bourne
IPCop - DHCP configuration
IPCop 1.4.1
SERVICES
DHCP SERVER
The bad packets stop here.
SYSTEM
STATUS
NETWORK
SERVICES
FIREWALL
VPNS
LOGS

DHCP

Green Interface Enabled: ☑

Start address: ●	192.168.20.25	End address: ●	192.168.20.75
Default lease time (mins):	16000	Max lease time (mins):	15000
Domain name suffix: ●	[redacted]		
Primary DNS: ●	192.168.20.17	Secondary DNS: ●	
Primary NTP Server ●		Secondary NTP Server ●	
Primary WINS Server address: ●		Secondary WINS Server address: ●	

● This field may be blank. Save

Add a new fixed lease

MAC Address:	00:02:3F:0F:49:C8	IP Address:	192.168.20.3		
Next Address:	192.168.20.1 ●	Filename:	BootFile.img ●	Root Path:	/myboots/ ●

Enabled: ☑ Add

● This field may be blank.

Current fixed leases

MAC Address	IP Address	Next Address	Filename	Root Path	
00:04:23:[redacted]	192.168.20.4				☑ ✎ 🗑
00:01:01:F7:2E:BD	192.168.20.1				☑ ✎ 🗑
00:0c:6e:60:19:c5	192.168.20.2				☑ ✎ 🗑

Legend: ☑ Enabled (click to disable) ☐ Disabled (click to enable) ✎ Edit 🗑 Remove

Current dynamic leases

IP Address	MAC Address	Hostname	Lease expires (local time d/m/y)
192.168.20.75	00:40:f4:8a:26:7d		24/12/2004 12:55:08
192.168.20.50	00:0c:41:17:12:77	JuLinksys	21/12/2004 11:34:38

Figure 2.8 Screenshot of IPCop (IPCop, 2009)

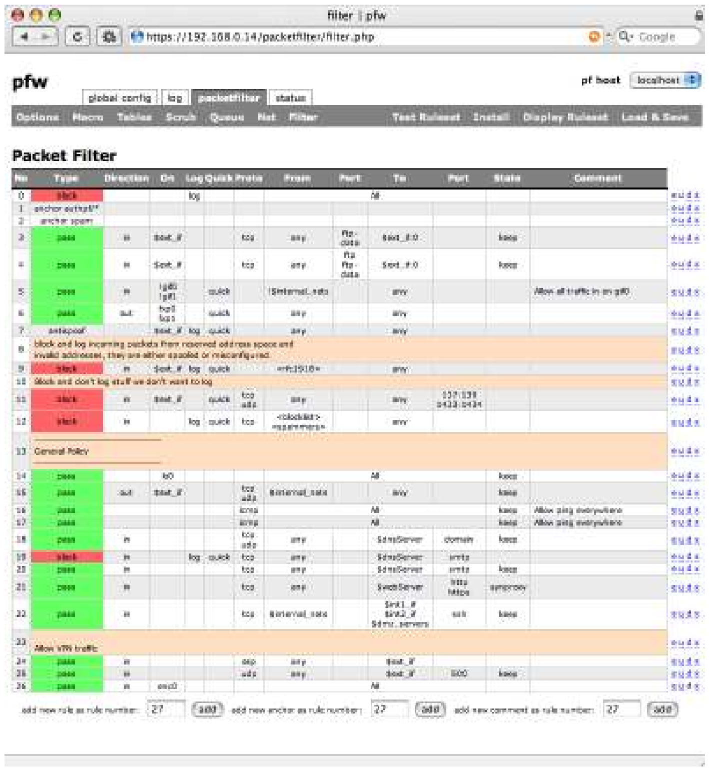
2.6.5 *Previous* Projects *For OpenBSD*

There are two considerable projects for OpenBSD management but one of them is a command line tool and the other has ability to manage firewall rules

2.6.5.1 routerctl

It manages network connectivity, DHCP, NAT, packet filtering and port forwarding via a single high-level configuration file. There is a graphical 'wizard' to setup the network for the first time, and the program provides general diagnostic information via a Web browser (routerctl, 2009).

routerctl is a command line application. It reads a specific configuration file and creates firewall rules or configures DHCP server. Initial configuration file can be created via a Web browser, but it does not have the ability of editing rules.



Packet Filter

No	Type	Direction	On	Log	Quick	Proto	From	Port	To	Port	State	Comment
0	block			log					All			
1	anchor authpf/*											
2	anchor spam											
3	pass	in	$ext_if			tcp	any	ftp-data	$ext_if:0		keep	
4	pass	in	$ext_if			tcp	any	ftp ftp-data	$ext_if:0		keep	
5	pass	in	!gif0 !gif1		quick		!$internal_nets		any			Allow all traffic in on gif0
6	pass	out	!tun0 !tun1		quick		any		any			
7	antispoof		$ext_if	log	quick		any		any			
8	block and log incoming packets from reserved address space and invalid addresses, they are either spoofed or misconfigured.											
9	block	in	$ext_if	log	quick		<rfc1918>		any			
10	block and don't log stuff we don't want to log											
11	block	in	$ext_if		quick	tcp udp	any		any	137:139 1433:1434		
12	block	in		log	quick	tcp	<blocklist> <spammers>		any			
13	General Policy											
14	pass		lo0						All		keep	
15	pass	out	$ext_if			tcp udp	$internal_nets		any		keep	
16	pass					icmp			All		keep	Allow ping everywhere
17	pass					icmp			All		keep	Allow ping everywhere
18	pass	in				tcp udp	any		$dnsServer	domain	keep	
19	block	in		log	quick	tcp	any		$dnsServer	smtp		
20	pass	in				tcp	any		$dnsServer	smtp	keep	
21	pass	in				tcp	any		$webServer	http https	synproxy	
22	pass	in				tcp	$internal_nets		$int1_if $int2_if $dmz_servers	ssh	keep	
23	Allow VPN traffic											
24	pass	in				esp	any		$ext_if			
25	pass	in				udp	any		$ext_if	500	keep	
26	pass	in	enc0						All			

add new rule as rule number: 27 add | add new anchor as rule number: 27 add | add new comment as rule number: 27 add

Figure 2.9 Screenshot of PFW

2.6.5.2 PFW

PFW is a Web based packet filter management tool. It runs on OpenBSD. It has the advantage of editing firewall rules. This project has not the ability to control other services like DHCP. PFW project is no longer maintained (PFW, 2009). Figure 2.9 shows a screenshot of PFW.

CHAPTER THREE
USING OPENBSD

This chapter covers how to configure OpenBSD via command line tools. These contain;

- Networking configuration
- DHCP server configuration
- PF configuration

3.1 Networking

In OpenBSD, network configuration is done by editing some system files and using same core applications.

3.1.1 Identifying and Configuring interfaces

To identify interfaces, ifconfig command is used.

```
$ ifconfig
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 33224
        inet 127.0.0.1 netmask 0xff000000
        inet6 ::1 prefixlen 128
        inet6 fe80::1%lo0 prefixlen 64 scopeid 0x5
fxp0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        address: 00:04:ac:dd:39:6a
        media: Ethernet autoselect (100baseTX full-duplex)
        status: active
        inet 10.0.0.38 netmask 0xffffff00 broadcast 10.0.0.255
        inet6 fe80::204:acff:fedd:396a%fxp0 prefixlen 64 scopeid 0x1
```

3.1.1.1 Configuring Interfaces Manually

Interfaces are configured at boot time by using */etc/hostname.{interface_name}* file. The layout of these files is like that:

```
address_family address netmask broadcast [other options]
```

A simple configuration looks like that:

```
$ cat /etc/hostname.fxp0

inet 10.0.0.38 255.255.255.0 NONE
```

3.1.1.2 Configuring Interfaces via DHCP

/etc/hostname.{interface_name} file must be edited for interface to obtain network settings via DHCP.

```
$cat /etc/hostname.fxp0

echo dhcp > /etc/hostname.fxp0
```

3.1.2 Configuring Default Gateway

At boot time, /etc/mygate file is used to configure the gateway address.

```
$ cat /etc/mygate

10.0.0.2
```

3.1.3 Configuring DNS Servers

DNS resolution is controlled by /etc/resolv.conf file.

```
$ cat /etc/resolv.conf

nameserver 125.2.3.4

nameserver 125.2.3.5
```

3.1.4 Setting Host Name

The hostname is specified in /etc/myname file.

```
$ cat /etc/myname
demirfw.example.com
```

3.1.5 Activating Changes

To activate the settings, the machine can re rebooted or /etc/netstart script can be used.

```
# sh /etc/netstart
writing to routing socket: File exists
add net 127: gateway 127.0.0.1: File exists
writing to routing socket: File exists
add net 224.0.0.0: gateway 127.0.0.1: File exists
```

3.1.6 Setting as a Router

To use OpenBSD as a router, packet forwarding must be enabled. *sysctl* command is used for this.

```
# sysctl net.inet.ip.forwarding=1
net.inet.ip.forwarding: 0 -> 1
```

To make this change permanent, */etc/sysctl.conf* file is used. To do this line must be added.

```
net.inet.ip.forwarding=1
```

Dynamic Host Control Protocol (DHCP) is used to configure network interfaces automatically. A host in network broadcast a DHCP request, if there is a DHCP server in the LAN, the server replies this request.

3.2 DHCP Server

Dynamic Host Control Protocol (DHCP) is used to configure network interfaces automatically. A host in a network broadcast a DHCP request, if there is a DHCP server in the LAN, the server replies this request.

To configure DHCP server in OpenBSD, first it must be determined that which interface will be used. Since OpenBSD is used as a firewall, the internal interface is used for this configuration. In this example, external interface is *fxp0*.

```
# echo 'dhcpd_flags="fxp0"' >>/etc/rc.conf.local
```

DHCP server reads the /etc/rc.conf.local file and uses the dhcpd_flags variable as input. After this configuration, /etc/dhcpd.conf must be edited.

```
Option  domain-name "example.com";
      option  domain-name-servers 192.168.1.3, 192.168.1.5;
      subnet 192.168.1.0 netmask 255.255.255.0 {
              option routers 192.168.1.1;
              range 192.168.1.32 192.168.1.127;
      }
```

3.3 Packet Filter (PF)

Packet Filter (PF) is the firewall implementation of OpenBSD. It is used for filtering network packets and also capable of network address translation.

3.3.1 PF Basics

To enable Packet Filter (PF), *pfctl* command is used.

```
# pfctl –e
# pfctl -d
```

These commands are used to enable and disable, respectively. To enable PF permanent this line must be added to /etc/rc.conf.local:

```
pf=YES
```

/etc/pf.conf is the file which is interpreted by PF and stores PF rules. Following commands are use to demonstrate simple pfctl commands.

```
# pfctl -f /etc/pf.conf     Load the pf.conf file
# pfctl -nf /etc/pf.conf    Parse the file, but don't load it
# pfctl -Nf /etc/pf.conf    Load only the NAT rules from the file
# pfctl -Rf /etc/pf.conf    Load only the filter rules from the file
# pfctl -sn                 Show the current NAT rules
# pfctl -sr                 Show the current filter rules
# pfctl -ss                 Show the current state table
# pfctl -si                 Show filter stats and counters
# pfctl -sa                 Show EVERYTHING it can show
```

(PF: Getting Started, 2009)

3.3.1.1 List and Macros

Lists are used to specify multiple similar criterions within a rule. These are port numbers, address, etc. Lists are defined by specifying items within { } brackets. For example;

```
block out on fxp0 from { 192.168.0.1, 10.5.32.6 } to any
```

gets expanded to:

```
block out on fxp0 from 192.168.0.1 to any
block out on fxp0 from 10.5.32.6 to any
```

Macros are user-defined variables that can hold IP addresses, port numbers, interface names, etc. Example usage of macros:

```
ext_if="fxp0"
block in on $ext_if from any to any
```

3.3.1.2 Tables

Tables are used to hold a group of IP addresses and used to specify multiple similar criterions within a rule. The following attributes may be specified for each table:

- `const` - the contents of the table cannot be changed once the table is created. When this attribute is not specified, pfctl may be used to add or remove addresses from the table at any time. (PF: Tables, 2009)
- `persist` - causes the kernel to keep the table in memory even when no rules refer to it. Without this attribute, the kernel will automatically remove the table when the last rule referencing it is flushed. (PF: Tables, 2009)

Here is an example:

```
table <goodguys> { 192.0.2.0/24 }
table <rfc1918> const { 192.168.0.0/16, 172.16.0.0/12, \
   10.0.0.0/8 }
table <spammers> persist

block in on fxp0 from { <rfc1918>, <spammers> } to any
pass  in on fxp0 from <goodguys> to any
```

3.3.1.3 Packet Filtering

Packet filtering is the selective passing or blocking of data packets as they pass through a network interface. This is the simplified syntax of packet filtering rule:

```
action [proto protocol] [from src_addr [port src_port]] [to dst_addr
[port dst_port]]
```

action

The action to be taken for matching packets, either pass or block. The pass action will pass the packet back to the kernel for further processing while the block action will react based on the setting of the block-policy option. The default reaction may be overridden by specifying either block drop or block return. (PF: Packet Filtering, 2009)

protocol

The Layer 4 protocol of the packet:

- tcp
- udp
- icmp
- icmp6
- A valid protocol name from /etc/protocols
- A protocol number between 0 and 255
- A set of protocols using a list.

src_addr, dst_addr

The source/destination address in the IP header. Addresses can be specified as:

- A single IPv4 or IPv6 address.
- A fully qualified domain name that will be resolved via DNS when the rule set is loaded. All resulting IP addresses will be substituted into the rule.
- The name of a network interface. Any IP addresses assigned to the interface will be substituted into the rule.
- A table
- Any of the above but negated using the ! ("not") modifier.
- A set of addresses using a list
- The keyword all which is short for from any to any.

src_port, dst_port

- The source/destination port in the Layer 4 packet header. Ports can be specified as:
- A number between 1 and 65535
- A valid service name from /etc/services
- A set of ports using a list
- A range:
    - != (not equal)
    - < (less than)
    - (greater than)
    - <= (less than or equal)
    - >= (greater than or equal)
    - >< (range)
    - (inverse range)

Some examples:

```
pass in  on dc0 from 192.168.0.0/24 to 192.168.0.1
pass out on dc0 from 192.168.0.1 to 192.168.0.0/24
```

3.3.1.4 Network Address Translation (NAT)

Network Address Translation (NAT) is a way to map an entire network (or networks) to a single IP address.

This is the simplified syntax of NAT rule:

```
nat [pass] on interface from src_addr [port src_port] to \
   dst_addr [port dst_port] -> ext_addr
```

nat

The keyword that begins a NAT rule.

pass

Causes translated packets to completely bypass the filter rules.

interface

The name or group of the network interface to translate packets on.

src_addr

The address family, either inet for IPv4 or inet6 for IPv6. PF is usually able to determine this parameter based on the source/destination address(es).

src_port

The source port in the Layer 4 packet header. Ports can be specified as:

- A number between 1 and 65535
- A valid service name from /etc/services
- A set of ports using a list
- A range:
  - != (not equal)
  - < (less than)
  - (greater than)
  - <= (less than or equal)
  - >= (greater than or equal)
  - >< (range)
  - (inverse range)

The port option is not usually used in nat rules because the goal is usually to NAT all traffic regardless of the port(s) being used.

dst_addr

The destination address of packets to be translated. The destination address is specified in the same way as the source address.

dst_port

The destination port in the Layer 4 packet header. This port is specified in the same way as the source port.

ext_addr

The external (translation) address on the NAT gateway that packets will be translated to. The external address can be specified as:

- A single IPv4 or IPv6 address.
- A fully qualified domain name that will be resolved via DNS when the rule set is loaded.
- The name of the external network interface. Any IP addresses assigned to the interface will be substituted into the rule at load time.

Some Examples:

```
nat on tl0 from 192.168.1.0/24 to any -> 24.5.0.5
```

3.3.1.5 Port Forwarding (Redirection)

When NAT is used all machines have access to Internet. But if a machine should be accessed from outside, port forwarding is used.

For example; the following rules redirects TCP port 80 to the machine which has IP address 192.168.1.240.

```
rdr on tl0 proto tcp from any to any port 80 -> 192.168.1.20
```

3.3.2 Simple Scenario - Using OpenBSD as Firewall

In this scenario, there is a network with subnet 192.168.0.0/24 which connects to Internet using a modem. Figure 3.1 shows the diagram after inserting OpenBSD as a firewall between modem and local area network. The external interface of OpenBSD is fxp0 and internal interface is rx0. Modem has 10.0.0.2 IP address and has a DHCP server, so fxp0 interface will obtain IP via DHCP. rx0 interface will be 192.168.1.1.

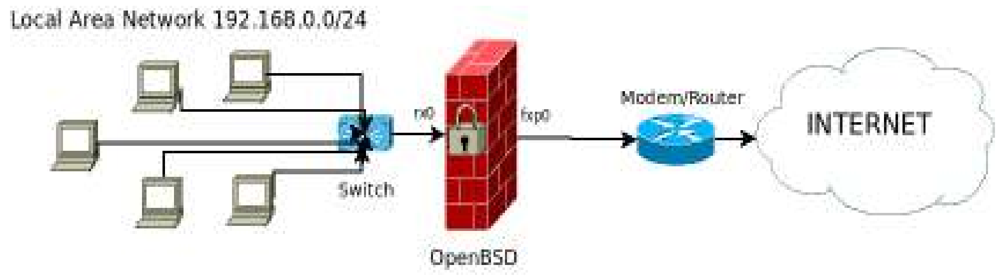


Figure 3.1 Simple Home/Office Network with OpenBSD

Configuring interfaces will be done with these commands:

```
# echo dhcp > /etc/hostname.fxp0

# echo inet 192.168.0.1 255.255.255.0 NONE > /etc/hostname.rx0

# sysctl net.inet.ip.forwarding=1

# echo net.inet.ip.forwarding=1 >> /etc/sysctl.conf

# /etc/netstart
```

Configuring DHCP server on OpenBSD will be done with these commands:

```
# echo rx0 >/etc/dhcpd.interfaces

# cat >> /etc/dhcpd.conf << EOF

        option  domain-name-servers 10.0.0.2;

        subnet 192.168.0.0 netmask 255.255.255.0 {

                option routers 192.168.0.1;

                range 192.168.0.50 192.168.0.70;

        }

EOF

# echo dhcpd_flags="" >> /etc/rc.conf.local

#  dhcpd xl0
```

Configuring PF for NAT will be done with these commands:

```
# cat >> /etc/pf.conf << EOF

       ext_if="fxp0"

   int_if="rx0"

   nat on $ext_if from !($ext_if) to any -> ($ext_if)

EOF

# echo pf_flags=”” >> /etc/rc.conf.local

# pfctl –e
```

CHAPTER FOUR
JUSTIFYING DESIGN DECISIONS OF DEMIRFW

DemirFW implementation aims to run on embedded systems. OpenBSD operating system needs low performance of hardware and can be run on embedded systems. Since DemirFW is based on OpenBSD, only some design decisions is needed for a better performance to run on embedded hardware.

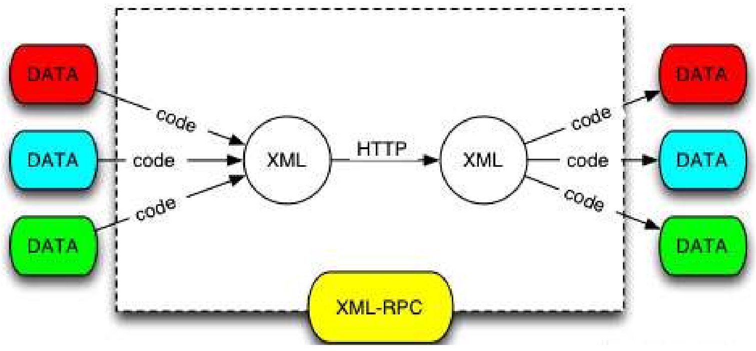
4.1 Remote Procedure Call Protocol

Remote Procedure Call Protocol is used for communication between DemirFW client and server part. There are two considerable alternatives. These are XML-RPC protocol and SOAP protocol.

4.1.1 XML-RPC

XML-RPC is a specification and a set of implementations that allow software running on different operating systems, running in different environments to make procedure calls over the Internet (XML-RPC, 2009).

XML-RPC uses HTTP for transport and XML for encoding. XML-RPC is designed to be as simple as possible, while allowing complex data structures to be transmitted, processed and returned. Figure 4.1 shows the process of XML-RPC.



.

Figure 4.1 XML-RPC Structure (XML-RPC, 2009)

An example of a typical XML-RPC request would be:

```
<?xml version="1.0"?>
<methodCall>
  <methodName>examples.getStateName</methodName>
  <params>
    <param>
        <value><i4>40</i4></value>
    </param>
  </params>
</methodCall>
```

An example of a typical XML-RPC response would be:

```
<?xml version="1.0"?>
<methodResponse>
  <params>
    <param>
        <value><string>South Dakota</string></value>
    </param>
  </params>
</methodResponse>
```

4.1.2 SOAP

SOAP is a simple XML based protocol to let applications exchange information over HTTP. SOAP once stood for 'Simple Object Access Protocol' but this acronym was dropped with Version (SOAP Version 1.2, 2009). Version 1.2 became a W3C recommendation on June 24, 2003.

An example of a typical SOAP request would be:

```
<soapenv:Envelope

    xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"

    xmlns:xsd="http://www.w3.org/2001/XMLSchema"

    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

    xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
http://schemas.xmlsoap.org/soap/envelope/">

  <soapenv:Body>

    <req:echo
xmlns:req="http://localhost:8080/axis2/services/MyService/">

      <req:category>classifieds</req:category>

    </req:echo>

  </soapenv:Body>

</soapenv:Envelope>
```

An example of a typical XML-RPC response would be:

```
<soapenv:Envelope

    xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"

    xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"

    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

    xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
http://schemas.xmlsoap.org/soap/envelope/">

  <soapenv:Header>

    <wsa:ReplyTo>


<wsa:Address>http://schemas.xmlsoap.org/ws/2004/08/addressing/role/a
nonymous</wsa:Address>

    </wsa:ReplyTo>

    <wsa:From>


<wsa:Address>http://localhost:8080/axis2/services/MyService</wsa:Add
ress>

    </wsa:From>


<wsa:MessageID>ECE5B3F187F29D28BC11433905662036</wsa:MessageID>

  </soapenv:Header>
```

```
    <soapenv:Body>

      <req:echo
xmlns:req="http://localhost:8080/axis2/services/MyService/">

        <req:category>classifieds</req:category>

      </req:echo>

    </soapenv:Body>

  </soapenv:Envelope>
```

4.1.3 SOAP vs. XML-RPC

SOAP and XML-RPC are platform independent and language independent protocols. So it seems that both protocols can be used with DemirFW. According to the paper which discusses XML-RPC and SOAP on embedded systems (Dissanaike, S., Wijkman, P. & Wijkman, M., 2004), XML-RPC is better suited for an embedded system because the protocol specifications and implementation of XML-RPC is significantly smaller than SOAP's. The complexity of SOAP can be seen on Table 4.1. This table shows the comparison of features of SOAP and XML-RPC. SOAP has more features but XML-RPC specifications are enough for DemirFW. More specifications mean more processing time and more memory space. Since an embedded system has constrained memory and CPU power, the implementation that costs less resource is important. With considering this information, XML-RPC is used in DemirFW.

Table 4.1 Comparison of XML-RPC and SOAP (Dissanaike, S., Wijkman, P. & Wijkman, M., 2004)

Feature	**XML-RPC**	**SOAP**
Embedded Implementation	YES	NO
Basic Scalars, Structs, Arrays	YES	YES
Named structs and arrays	NO	YES
Detailed Fault Handling	YES	YES
Short Learning Curver	YES	NO
Developers Specified Character Set	NO	YES
Developer Defined Data Types	NO	YES
Can Specify Recipient	NO	YES
Require Client Understanding	NO	YES
Message Specific Processing Instructions	NO	YES

4.2 Web Server

XML-RPC protocol implementation runs on a Web server in DemirFW. There are two widely used options for Web server. These are Apache and Lighttpd.

4.2.1 Apache

Apache is widely used Web server in the world. Apache became the first Web server to surpass the 100 million Web site milestone (February 2009 Web Server Survey, 2009). Apache is developed and maintained by an open community of developers under the auspices of the Apache Software Foundation. The application is available for a wide variety of operating systems, including Unix, GNU, FreeBSD, Linux, Solaris, Novell NetWare, Mac OS X, Microsoft Windows, OS/2.

4.2.2 Lighttpd

Lighttpd is designed to be fast. Some known websites like YouTube and Wikipedia are using lighttpd (PHP Magazine, 2009). It has smaller footprint compared to other Web servers. With this feature, it consumes less memory. Since it consumes less memory, it can be used in memory restricted hardware.

4.2.3 Apache vs. Lighttpd

Since DemirFW aims to run on embedded systems, it is important for applications to consume less memory. Lighttpd has small footprint and consumes less memory and one of the advantages of lighttpd is to serve more transactions as seen on Figure 4.2 and Figure 4.3. The x axis and y axis of Figure 4.2 show average number of requests per second and Web servers, respectively. As seen in Figure 4.2, Lighttpd is the Web server which can handle more requests than others. Figure 4.2 shows the comparision of Apache and Lighttpd with a content management system installed. The x axis and y axis of Figure 4.3 show number of requests per second and configuration types, respectively. Configuration types are default, apc, wp-cache which are default configuration, php cache configuration enabled, wp-cache (which is the cache of content management system) enabled, respectively. With all different configurations, Lighttpd can handle more requests than Apache.

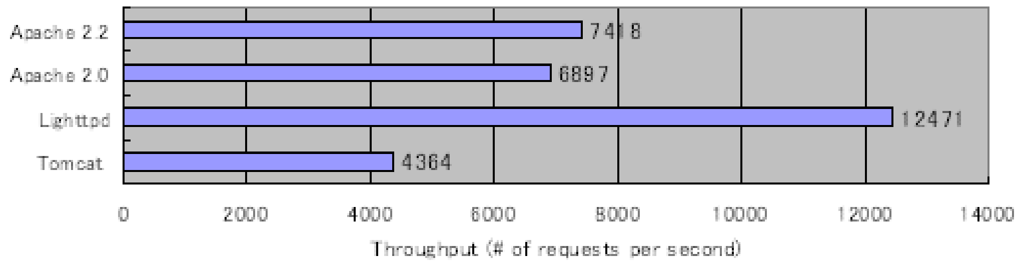


Figure 4.2 Apache vs Lighttpd (Performance Comparison of Web Service Engines, 2009)

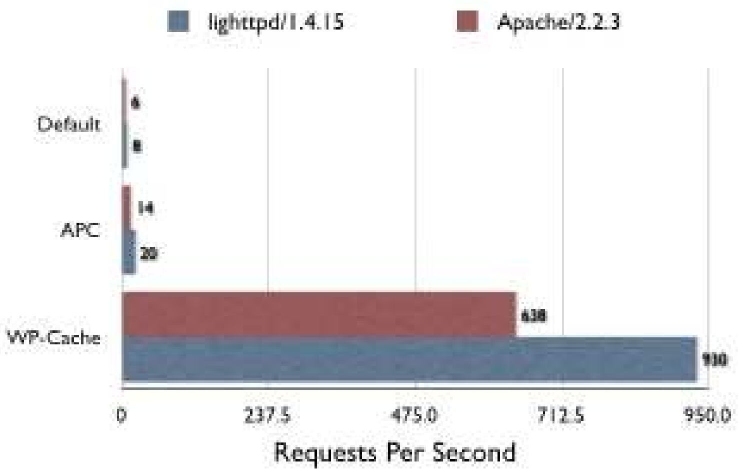


Figure 4.3 Apache vs. Lighttpd (Apache vs. Lighttpd, 2009)

4.3 Database

DemirFW uses database to store information. There are some widely used database servers like MySQL and some embedded databases like SQLite. OpenBSD package repository has two of them.

4.3.1 MySQL

MySQL is the most used database server which has more than 6 million installations (MySQL Downloads, 2009). MySQL runs as a background server and provides multi user access. Several major companies such as Google (Google Releases Improved MySQL Code, 2009), Facebook (Facebook Blog, 2009), Wikipedia (Wikimedia Servers, 2009) uses MySQL.

4.3.2 SQLite

SQLite is a software library that implements a self-contained, serverless, zero-configuration, transactional SQL database engine. SQLite is the most widely deployed SQL database engine in the world (Most Widely Deployed SQL Database, 2009). SQLite is deployed in Mozilla Firefox, Skype, Symbian and iPhone (Most Widely Deployed SQL Database, 2009).

4.3.3 MySQL vs. SQLite

DemirFW focuses on embedded hardware. The applications that run on embedded hardware should consume less memory and CPU. MySQL is a database server and always run in background and consumes memory. SQLite is a software library, so it consumes memory while a query is in process. Considering this information, SQLite is used in DemirFW.

CHAPTER FIVE
IMPLEMENTATION: DEMIRFW

DemirFW is a firewall software package implemented for this thesis. DemirFW consists of two parts: server part and client part. Server part and client part implementation is nearly 5500 lines of code with comments. Java and Python programming languages have been used for this study.

Appendix A and Appendix B contain only source code of modules and module clients, not other implementations like which parses XML strings.

5.1 Server Part

Server Part has modules that XML-RPC server serves. These modules access corresponding files or services to process the order coming from client. Figure 5.1 shows the internal structure of server part.

5.1.1 Packet Filter (PF)

Packet Filter is the blocking/passing packet mechanism of OpenBSD. DemirFW is used to control this service. More details on PF were discussed in Chapter Three.

5.1.2 DHCP Server

DHCP server is used to distribute IP addresses to hosts on local area network. DemirFW is used to control this service. More details on DHCP server were discussed in Chapter Three.

5.1.3 System Files

Some files in OpenBSD are used to hold IP address of the hardware or nameserver to resolve domains. DemirFW is used to control these system files.

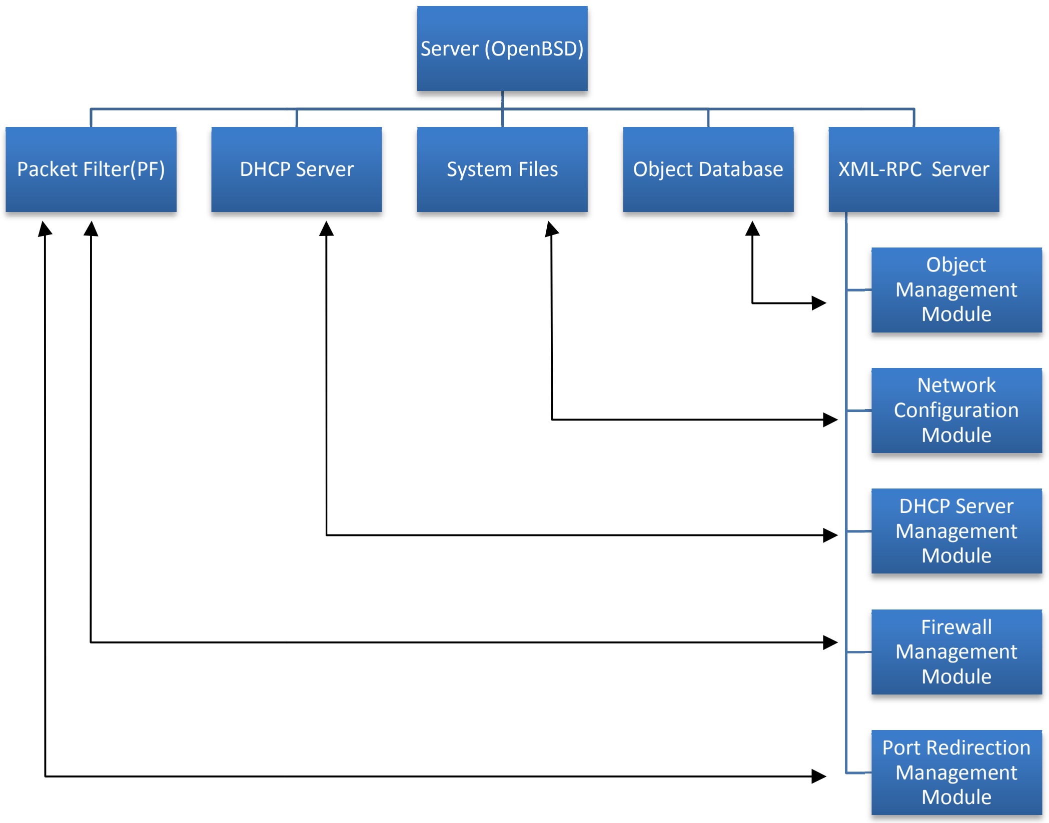

Server (OpenBSD)
Packet Filter(PF)
DHCP Server
System Files
Object Database
XML-RPC Server
Object Management Module
Network Configuration Module
DHCP Server Management Module
Firewall Management Module
Port Redirection Management Module

Figure 5.1 Internal Structure of DemirFW

5.1.4 Object Database

Object database is used to store some information like IP addresses, port numbers, network addresses. SQLite is used for object database. Justifying of using SQLite has been discussed in Chapter Four.

5.1.5 XML-RPC Server

XML-RPC is a remote procedure call protocol implementation. It provides to access modules inside server part. These modules are;

1. Object Management Module
2. Network Configuration Module
3. DHCP Server Management Module

4. Firewall Management Module
5. Port Redirection Management Module

5.1.5.1 Object Management Module

Every IP address, port, network is an object. Before using objects, objects must be created. For example, the rule "*block TCP packets from 10.0.0.2 to 10.0.0.5 going to port 80*" is created with firewall management service but 10.0.0.2, 10.0.0.5, and TCP port 80 are objects and must be created with object management module.

Objects are saved in a database. This database is access via with SQLite libraries. Table 5.1 shows the schema of the database table used for object management.

Table 5.1 Schema of the Object Database

```
CREATE TABLE hosts ( id INTEGER  PRIMARY KEY AUTOINCREMENT, name
VARCHAR(32) NOT NULL UNIQUE, ip VARCHAR(512) NOT NULL UNIQUE);

CREATE TABLE ipgroups ( id INTEGER  PRIMARY KEY AUTOINCREMENT, name
VARCHAR(32) NOT NULL UNIQUE, ip VARCHAR(512) NOT NULL UNIQUE);

CREATE TABLE ipranges ( id INTEGER  PRIMARY KEY AUTOINCREMENT, name
VARCHAR(32) NOT NULL UNIQUE, ip VARCHAR(512) NOT NULL UNIQUE);

CREATE TABLE networks ( id INTEGER  PRIMARY KEY AUTOINCREMENT, name
VARCHAR(32) NOT NULL UNIQUE, ip VARCHAR(512) NOT NULL UNIQUE);

CREATE TABLE servicegroups ( id INTEGER  PRIMARY KEY AUTOINCREMENT,
name VARCHAR(32) NOT NULL UNIQUE, ip VARCHAR(512) NOT NULL UNIQUE);

CREATE TABLE tcpservices ( id INTEGER  PRIMARY KEY AUTOINCREMENT,
name VARCHAR(32) NOT NULL UNIQUE, ip VARCHAR(512) NOT NULL UNIQUE);

CREATE TABLE udpservices ( id INTEGER  PRIMARY KEY AUTOINCREMENT,
name VARCHAR(32) NOT NULL UNIQUE, ip VARCHAR(512) NOT NULL UNIQUE);
```

<u>submitObjects function</u>

This function is used to create new object or edit an existing objects. Input parameters of this function are type and request. type parameter is used to identify the type of objects which the request parameter contains. *type* parameter can be one of the followings:

- host

- network
- iprange
- ipgroup
- tcpservice
- udpservice
- servicegroup

request parameter is a set of commands formatted in XML and describes to create or edit objects. Return value indicates if the operation is successful or not. Table 5.2 shows example request parameters.

Table 5.2 Example or request parameter for submitObjects function

`<demir-fw>`	`<demir-fw>`
` <objects>`	` <objects>`
`  <object>`	`  <object>`
`   <id>-1</id>`	`   <id>15</id>`
`   <name>host23</name>`	`   <name>host24</name>`
`   <value>192.168.1.23</value>`	`   <value>192.168.1.24</value>`
`  </object>`	`  </object>`
` </objects>`	` </objects>`
`</demir-fw>`	`</demir-fw>`

<u>getObjects function</u>

This function is used to retrieve stored objects. Input parameter is used to identify the type of object which will be retrieved. *type* parameter can be one of the followings:

- host
- network
- iprange
- ipgroup
- tcpservice
- udpservice

- servicegroup

Table 5.3 Example of return value for getObjects function

`<demir-fw>`	`<demir-fw>`
`<objects type='udpservice'>`	`<objects type='network'>`
` <object>`	` <object>`
`  <id>2</id>`	`  <id>1</id>`
`  <name>deneme</name>`	`  <name>network1</name>`
`  <value>12:12-43:98</value>`	`  <value>172.16.0.0/8</value>`
` </object>`	` </object>`
` <object>`	` <object>`
`  <id>3</id>`	`  <id>2</id>`
`  <name>test</name>`	`  <name>network5</name>`
`  <value>1:1-1:1</value>`	`  <value>172.16.1.0/23</value>`
` </object>`	` </object>`
`</objects>`	`</objects>`
`</demir-fw>`	`</demir-fw>`

Return value is an XML string which contains objects' information.

Table 5.3 shows two examples. The first example is the return of the request with the *type* parameter value udpservice and the second example is the return of the request with the *type* parameter value network.

5.1.5.2 Network Configuration Module

A firewall should be accessed in network so the network configuration must be done for access. Network Configuration Module provides these settings to be done. With Network Configuration Module, the external and internal interfaces of firewall can be configured. External interface can be configured manually or to obtaion IP address via DHCP, so this should be defined in request.

<u>submitNetworkConfiguration function</u>

This function is used to configure network settings of the firewall. Input parameter is a set of commands formatted in XML and contains network settings for the firewall. Table 5.4 shows examples of input parameter. Using first example, external interface will obtain IP via DHCP and internal interface IP will be 10.0.0.2 with netmask 255.0.0.0. In the second example; external interface IP is given manually. Hence so, gateway and the DNS settings are given manually. There is no return value because DemirFW will restart immediately to apply settings.

Table 5.4 Examples of input parameter for submitNetworkConfiguration function

```
<demir-fw>
 <network>
  <interfaces>
   <ext0>dhcp</ext0>
   <int0>10.0.0.2/255.0.0.0</int0>
  </interfaces>
 </network>
</demir-fw>
```

```
<demir-fw>
 <network>
  <interfaces>
   <ext0>192.168.1.15/255.255.255.0</ext0>
   <int0>10.0.0.2/255.0.0.0</int0>
  </interfaces>
  <gateway>192.168.1.1</gateway>
  <dns>192.168.1.1</dns>
  </network>
</demir-fw>
```

getNetworkConfiguration function

This function is used to retrieve network configuration. This function does not get any parameters. Return value is an XML string, which contains network configuration. This XML string is the value of the request parameter given for submitNetworkConfiguration function.

5.1.5.3 DHCP Server Management Module

Since a firewall is in the entry point of the network, it should have more than “packet blocking/passing ability”. One of them is DHCP server.

DHCP server is used for the hosts in the network to obtain IP address automatically. When the host in the network requests a DHCP request, DHCPserver sends a network setting. This network setting contains IP address, gateway address and the DNS server address.

submitDHCPDConfiguration function

This function is used to configure DHCP server settings used in the firewall. Input parameter is a set of commands formatted in XML and contains DHCP server settings. Table 5.5 shows an example input parameter. Return value indicates if the operation is successful or not.

Table 5.5 Example input parameter for submitDHCPConfiguration function

```
<demir-fw>
 <dhcpd>
  <router>10.0.0.2</router>
  <subnet>10.0.0.0</subnet>
  <netmask>255.0.0.0</netmask>
  <range>10.0.0.10-10.0.0.20</range>
  <dns>192.168.1.1</dns>
 </dhcpd>
</demir-fw>
```

getDHCPDConfiguration function

This function is used to retrieve DHCP server configuration. This function does not get any parameters. Return value is an XML string which contains network configuration. This XML string is the value of the input parameter of submitDHCPDConfiguration function.

5.1.5.4 Firewall Management Module

The most important part of a firewall is the function of blocking/passing packets. With given rules, firewall blocks or allows packets. The rules are given to firewall management module. There are two actions; block and pass.

submitFirewallRules function

This function is used to set firewall rules. Input parameter is a set of commands formatted in XML and contains firewall rules. Table 5.6 shows examples of input parameter. Return value indicates if the operation is successful or not.

Table 5.6 Examples of input parameter for submitFirewallRules function

```
<demir-fw>
 <firewallRules>
  <firewallRule>
   <action>pass</action>
   <source> any </source>
   <target> any </target>
   <port> any </port>
  </firewallRule>
 </firewallRules>
</demir-fw>
```

```
<demir-fw>
 <firewallRules>
  <firewallRule>
   <action>pass</action>
   <source> any </source>
   <target> any </target>
   <port> any </port>
  </firewallRule>
  <firewallRule>
   <action>block</action>
   <source>
    <host>16</host>
   </source>
   <target> any </target>
   <port> any </port>
  </firewallRule>
 </firewallRules>
</demir-fw>
```

getFirewallRules function

This function is used to retrieve firewall rules. This function does not get any parameters. Return value is an XML string, which contains firewall rules. This XML string is the value of the input parameter of submitFirewallRules function.

5.1.5.5 Port Redirection Management Module

To access to a local service (e.g. Apache Web Server) from outside the LAN, port redirection is needed. With Port Redirection Management module, these port redirections can be done.

submitRdrRules function

This function is used to set firewall rules. Input parameter is a set of commands formatted in XML and contains firewall rules. Return value indicates if the operation is successful or not.

Table 5.7 Example request parameters for submitRdrRules function

```
<demir-fw>
 <redirectionRules>
  <redirectionRule>
   <protocol>tcp</protocol>
   <hostId>16</hostId>
   <externalPortId>5</externalPortId>
   <internalPortId>4</internalPortId>
  </redirectionRule>
 </redirectionRules>
</demir-fw>
```

getRdrRules function

This function is used to retrieve port redirection rules. This function does not get any parameters. Return value is an XML string, which contains port redirection rules. This XML string is the value of the input parameter of submitRdrRules function.

5.2 Client Part

DemirFW client part can be implemented in any programming language with the advantage of XML-RPC. In this study, DemirFW client part is implemented in Java so that can run on all operating systems that have Java Virtual Machine. Also, it can run on Web browsers using Java Applet (Appendix C).

For each server module, client part has a module client. These module clients connect to XML-RPC server and XML-RPC server redirects requests to the corresponding modules. Each server module has a corresponding module client on client part. These are;

1. Object Management Module Client
2. Network Configuration Module Client
3. DHCP Server Management Module Client
4. Firewall Management Module Client
5. Port Redirection Management Module Client

Each module client is a part of whole client application. For each module client, a frame is used. In Figure 5.2, the menu of main window is shown. Clicking on menu will open a frame.

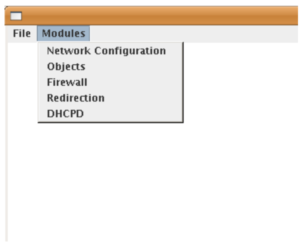
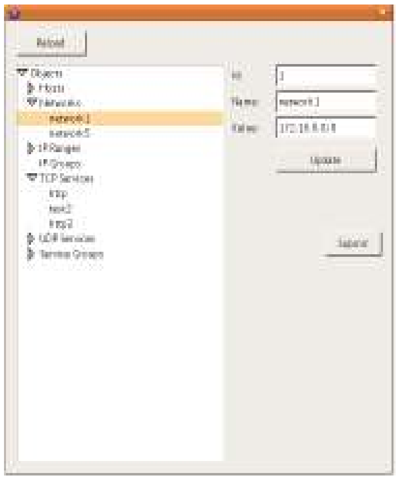


Figure 5.2 DemirFW Client Main Menu

5.2.1 *Object Management Module Client*

Object Management Module Client is used to access Object Management Module. Figure 5.3 shows a screenshot of Object Management Module Client.

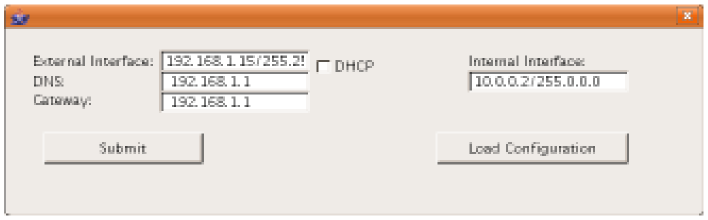


Id: 1
Name: network1
Value: 172.16.0.0/8

Figure 5.3 Object Management Module Client

5.2.2 *Network Configuration Module Client*

Network Configuration Module Client is used to access Network Configuration Module of XML-RPC server. Figure 5.4 shows a screenshot of Network Configuration Module Client.

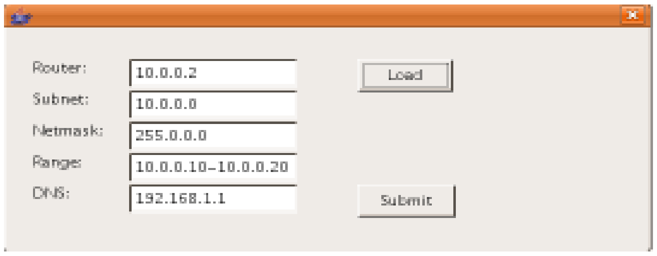


External Interface: 192.168.1.15/255.2!
DNS: 192.168.1.1
Gateway: 192.168.1.1
☐ DHCP
Internal Interface: 10.0.0.2/255.0.0.0

Figure 5.4 Network Configuration Module Client

5.2.3 *DHCP Server Management Module Client*

DHCP Server Management Module Client is used to access DHCP Server Management Module. Figure 5.5 shows a screenshot of DHCP Server Management Module Client.



Router: 10.0.0.2
Subnet: 10.0.0.0
Netmask: 255.0.0.0
Range: 10.0.0.10-10.0.0.20
DNS: 192.168.1.1

Figure 5.5 DHCP Server Management Module Client

5.2.4 Firewall Management Module Client

Firewall Management Module Client is used to access Firewall Management Module. Figure 5.6 shows a screenshot of Firewall Management Module Client.

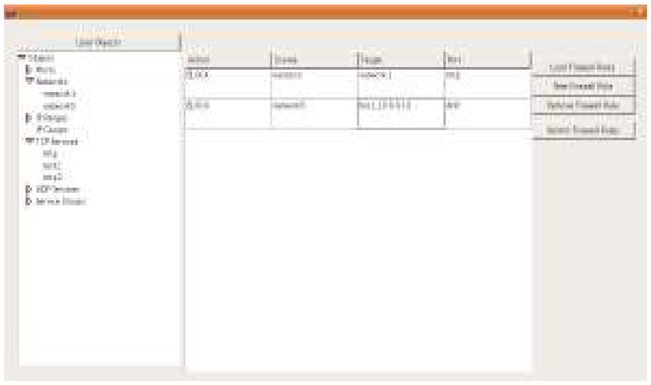
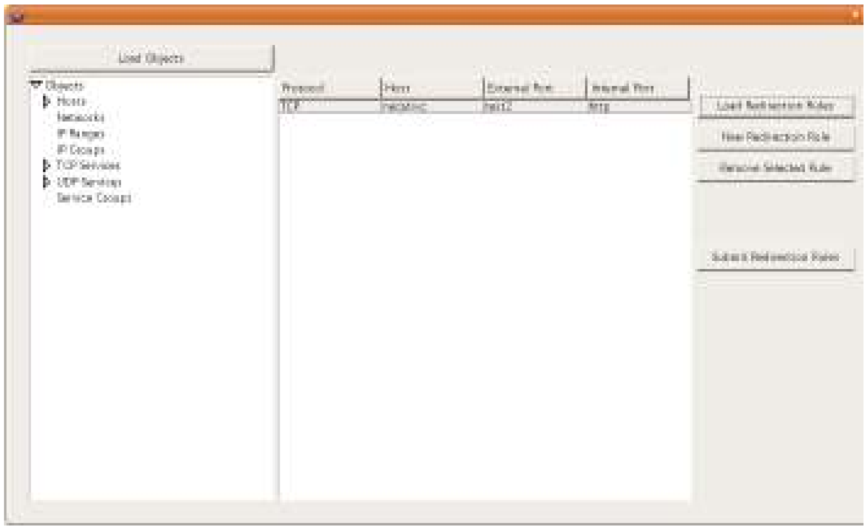


Figure 5.6 Firewall Management Module Client



Protocol	Host	External Port	Internal Port
TCP	[illegible]	host2	http

Figure 5.7 Port Redirection Management Module Client

5.2.5 Port Redirection Management Module Client

Port Redirection Management Module Client is used to access Port Redirection Management Module. Figure 5.7 shows a screenshot of Port Redirection Management Module Client.

5.3 Hardware

Although OpenBSD can be installed on various hardware, a small hardware was used in this thesis project. This hardware is eBox-4800 (Figure 5.8). Table 5.8 shows the eBox-4300 specifications.

Figure 5.8 eBox-4800 (eboxIV, 2009)

Table 5.8 Specifications of eBox-4800 (eBox, 2009)

CPU	VIA Esther 1.2 GHz
BIOS	AMI BIOS
System Chip	VIA CX700M
I/O Chipset	Winbond W83697
System Memory	Onboard 512 MB
Ethernet Chipset	Realtek 8100B 10/100 Base-T
Size (W x H x D)	170 x 123 x 38 mm

5.4 DemirFW in the Real World

DemirFW has been used in a small corporation with 10 computers and one server. Computer users are ordinary users and the server is a Web server. Figure 5.9 and Figure 5.10 show the corporation network before DemirFW and after DemirFW.

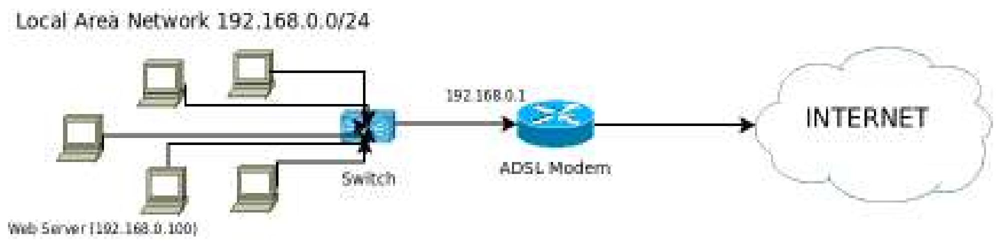


Figure 5.9 Corporation network before DemirFW

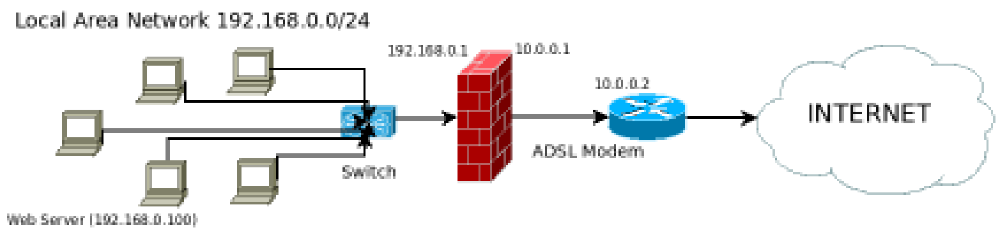


Figure 5.10 Corporation network after DemirFW

Before DemirFW, computers were using ADSL modem as a gateway. After DemirFW, computers' network configuration is not changed, but they were using DemirFW as a gateway. All network traffic has been forwarded from ADSL modem to DemirFW, so DemirFW would have the control for all traffic. DemirFW has been configured for DHCP server as well, since it is taking the role of DHCP server from ADSL modem. For two weeks, all the traffic has been saved with tcpdump, which is a network monitoring tool (tcpdump, 2009), to analyze later.

First week, DemirFW has been configured for DHCP server and to forward incoming connections to ports 80 and 22 to the server inside. Second week, DemirFW has been configured to allow only TCP ports 25, 80, 110, 443 and UDP port 53 from inside to outside. In addition, TCP port 22 connection has been allowed for only one specific address. The other ports for outgoing and incoming connections have been blocked. TCP ports 25 and 110 are used to send and receive mail, respectively. TCP port 80 and 443 are HTTP and HTTPS ports, so that users can browse Web pages. UDP port 53 is allowed for DNS queries.

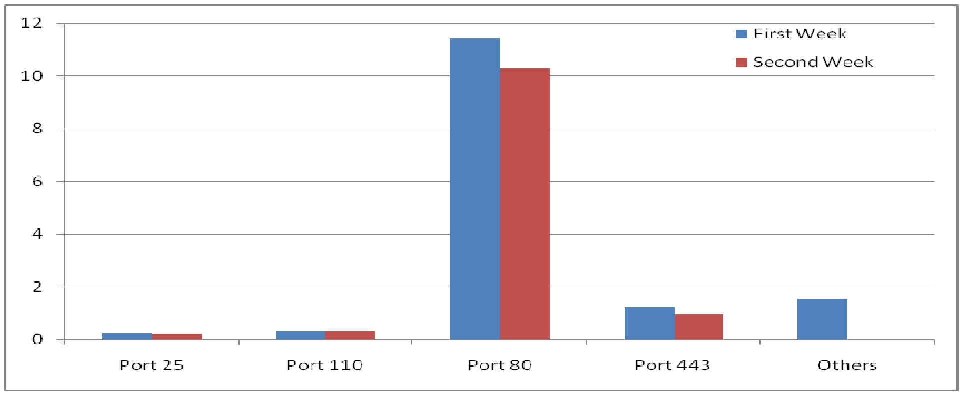


Figure 5.11 Analysis of TCP packets for two weeks

After two weeks, saved packets have been analyzed. Figure 5.11 shows the analysis of TCP packets. The x axis and y axis show TCP ports and gigabyte values that passed through DemirFW, respectively. The analyze has been done by splitting the packets to first week and second week. First week, no firewall rules were active and second week, firewall rules were active. As seen in Figure 5.11, the traffic passed in first week and second week are similar but it can be understood that second week there was less HTTP and HTTPS traffic. The column “others” shows the ports, except TCP port 25, 110, 80 and 443 like RTSP port, IRC port. It is evident that after activating firewall rules in DemirFW, connections from inside to “other” ports have been blocked.

After looking deeper to“other” ports passed in first week, it has been observed that some connections have been tried from local network to IRC ports of outside servers. After analyzing the computers in the network, two computers have been found with update disabled and several worms. One of the worms was “W32.Spybot.Worm”. This is an old worm but since updates have been disabled, it was still active and has tried to connect to IRC servers to send private information (W32.Spybot.Worm, 2009). In these illegal attacks, DemirFW has blocked these packets, so that it saved bandwidth and provided security.

Also, saved packets have been analyzed for incoming SSH connections. Figure 5.12 shows the results of this analysis. X axis and y axis of Figure 5.12 show status of connections and number of connection attempts. In first week, there were 283 connection attempts and all has been allowed. In second week, there were 39 connection attempts where 25 of them have been blocked and 14 of them have been allowed. The allowed connections were from allowed specific address. The reason of decreasing illegal connection attemps for second week can be thought as that attackers did not try to attempt to connect after trying 2 or 3 times because of the blocked connections. 9 IP addresses have tried SSH connections in second week. However, in first week, the illegal SSH connection attemps have been done from only 2 different IP addresses which tried to connect more than 100 times with different username and password combinations.

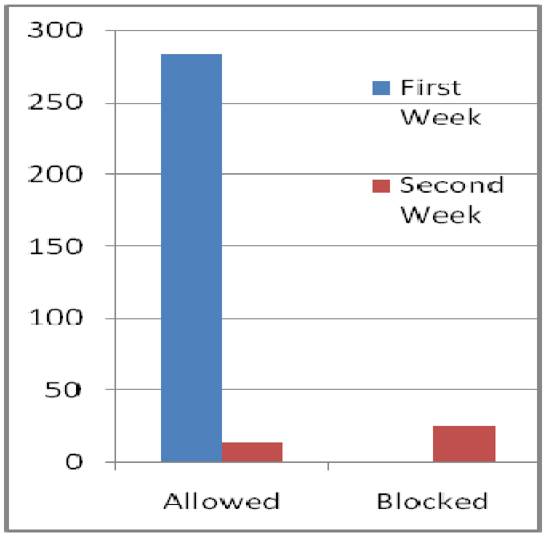


Figure 5.12 Analysis of incoming SSH connections

It is evident that firewalls are a must for a network. Not only for incoming connections, also outgoing connections should be filtered. In this field test, DemirFW has been configured to allow restricted outgoing and incoming connections. So attacks from outside has been blocked, also with filtering outgoing connections, malicious usage of the computers in the network has been blocked.

CHAPTER SIX
CONCLUSION & FUTURE WORK

There are many open source firewall management projects. Some of them are used to create only firewall rules and some of them provide more features. These ones, which provide more features, are mostly Linux based and few of them are FreeBSD based. There are only two considerable projects for OpenBSD; one of them is a command line tool and the other provides only one feature. In this study, an OpenBSD based embedded firewall software package, which has a graphical user interface, project has been aimed.

The implementation of this thesis is called DemirFW. DemirFW focuses on embedded systems. In this thesis, design decisions have been discussed for a better performance. Justification of chosen platforms can be listed as below.

- XML-RPC has been used as Remote Procedure Call protocol. It is lighter than the other option.
- Lighttpd has been used as Web server. It consumes less memory. This is main reason to use Lighttpd.
- SQLite has been used as database. SQLite is designed for embedded applications and used by mobile phone manufacturers like Apple. SQLite is not a database server, it is a software library. It only consumes memory while accessing database but the other option runs as a server in background.

DemirFW implementation has been discussed in two parts; server part and client part. Client part connects to server part via XML-RPC protocol. The XML-RPC server processes the data and sends it to the corresponding module, which is used to access the services of OpenBSD. Client side is an independent system and can be implemented in any language. The comparison of DemirFW and other OpenBSD based management projects can be seen on Table 6.1.

Table 6.1 shows the features of DemirFW, routerctl and PFW. routerctl has the basic features for managing OpenBSD, but it does not have a graphical user interface. The configuaration is done via a configuration file. PFW only provides management of PF, it does not provide other features like managing DHCP server or configuring network interfaces. Also two of them do not use a remote prodecure call protocol infrastructure. DemirFW provides managing basic features of OpenBSD to use it as a firewall on a network. Also DemirFW has a remote prodecure call protocol infrastructure, which provides extensibility.

Advantages of DemirFW can be listed as:

- DemirFW provides a simple interface to manage basic functions to use OpenBSD as a firewall. Previously discussed applications, focused on OpenBSD, are command line tools or provide less control.
- DemirFW uses the advantage of remote procedure call protocol. This provides extensibility for communication. With this feature, a new interface can be built easily or a part of DemirFW can be controlled via a totally different application.

Table 6.1 Comparison of OpenBSD Based Firewall Management Projects

	DemirFW	routerctl	PFW
Graphical Interface	YES	NO	YES
Managing with Objects	YES	YES	YES
Managing Network Interfaces	YES	YES	NO
Managing DHCP Server	YES	YES	NO
Managing PF	YES	YES	YES
Managing Port Redirection	YES	YES	YES
Remote Procedure Call Protocol Infrastructure	YES	NO	NO
NAT Functionality	YES	YES	YES

As a future work, a wireless function and a Web filtering function can be added. But with adding Web filtering function, the hardware requirements will increase. For these features, server and client communication infrastructure is ready so it will be easier to implement these management modules.

REFERENCES

Apache vs Lighttpd, Retrieved October 15, 2009, from http://immike.net/blog/2007/05/07/the-showdown-apache-vs-lighttpd/

Boyer, L. (1997). *Great Walls of Fire*. Retrieved March 15, 2009 from http://support.novell.com/techcenter/articles/nc1997_01a.html.

Dissanaike, S., Wijkman, P. & Wijkman, M. (2004). Utilizing XML-RPC or SOAP on an Embedded System. *Distributed System Workshops, 24,* 438-440. Retrieved October 29, 2009, from IEEE database.

Cisco Systems (2009). *Why you need a firewall*. Retrieved March 15, 2009 from http://www.cisco.com/univercd/cc/td/doc/product/iaabu/centri4/user/scf4ch2.htm.

eBox. Retrieved June 24, 2009, from http://www.compactpc.com.tw/ebox-4850.htm.

eboxIV. Retrieved June 24, 2009, from http://www.dsl-ltd.co.uk/images/eboxIV.gif

Facebook Blog, Retrieved November 5, 2009, from http://blog.facebook.com/blog.php?post=7899307130

February 2009 Web Server Survey. Retrieved October 14, 2009, from http://news.netcraft.com/archives/2009/02/18/february_2009_web_server_survey.html

Firewall Builder. Retrieved May 30, 2009, from http://www.fwbuilder.org/

Google Releases Improved MySQL Code, Retrieved October 5, 2009, from http://www.informationweek.com/news/internet/showArticle.jhtml?articleID=199201237

Genty, M., Kerouanton, B. & Khor, C. K. (2000). *Additional AIX Security Tools on IBM eServer pSeries, IBM RS/6000, and SP/Cluster*. IBM Redbooks, 13, 15, 16

Performance Comparison of Web Service Engines. Retrieved October 15, 2009, from https://researchweb.watson.ibm.com/trl/people/mich/pub/200809_icws2008wsperf.pdf

PF: Getting Started. Retrieved May 30, 2009, from http://www.openbsd.org/faq/pf/config.html

PF: Packet Filtering. Retrieved May 30, 2009, from http://www.openbsd.org/faq/pf/filter.html

PF: Tables. Retrieved May 30, 2009, from http://www.openbsd.org/faq/pf/tables.html

pfSense. Retrieved May 30, 2009, from http://www.pfsense.com/

PHP Magazine. Retrieved October 15, 2009, from http://www.phpmagazine.net/2007/02/lighttpd_vs_apache.html

IPCop. Retrieved May 30, 2009, from http://www.ipcop.org/index.php?name=FAQ&id_cat=2

m0n0wall. Retrieved May 30, 2009, from http://m0n0.ch/wall/

Most Widely Deployed Database. Retrieved October 16, 2009, from http://www.sqlite.org/mostdeployed.html

MySQL Downloads. Retrieved October 16, 2009, from http://www.mysql.com/news-and-events/generate-article.php?id=2005_42

OpenBSD: Retrieved July 9, 2009, from http://www.openbsd.org

PFW: Retrieved November 4, 2009, from http://www.allard.nu/pfw/

The Netperf: Retrieved June 28, 2009, from http://www.netperf.org/netperf/

SOAP Version 1.2. Retrieved October 13 2009, from http://www.w3.org/TR/soap12-part1/#intro

Stalling, W (1999). *Cryptography and Network Security: Principles and Practice* (2nd Edition). New Jersey: Prentice Hall, 518-521

routerctl: Retrieved November 4, 2009, from http://sofi-firewall.sourceforge.net/routerctl/

tcpdump, Retrieved November 5, 2009, from http://www.tcpdump.org/

W32.Spybot.Worm, Retrieved November 5, 2009, from http://www.symantec.com/security_response/writeup.jsp?docid=2003-053013-5943-99

Wikimedia Servers, Retrieved November 5, 2009, from http://meta.wikimedia.org/wiki/Wikimedia_servers

XML-RPC: Retrieved May 30, 2009, from http://www.xmlrpc.com/

Zwicky, E. D., Cooper, S., & Chapman, D. B. (2000). *Building Internet Firewalls* (2nd Edition). O'Reilly Media, Inc., 111

APPENDIX A
SOURCE CODE OF MODULES

A.1. Object Manager.py (Object Management Module)

```
#DemirFW Implementation for M.Sc. Thesis
#Necati Demir, 2009

from pysqlite2 import dbapi2 as sqlite3
from errors import *

class ObjectManager:

        def __createSQLQuery(self, command, type, id, name, value):
                if command=="update":
                        if type=="host":
                                query = "UPDATE hosts SET name='%s', 
ip='%s' WHERE id='%s'" % (name, value, id)
                        elif type=="network":
                                query = "UPDATE networks SET 
name='%s', ip='%s' WHERE id='%s'" % (name, value, id)
                        elif type=="iprange":
                                query = "UPDATE ipranges SET 
name='%s', ip='%s' WHERE id='%s'" % (name, value, id)
                        elif type=="ipgroup":
                                #h:1,2;n:1,2;i:1,2
                                query = "UPDATE ipgroups SET 
name='%s', ip='%s' WHERE id='%s'" % (name, value, id)
                        elif type=="tcpservice":
                                query = "UPDATE tcpservices SET 
name='%s', ip='%s' WHERE id='%s'" % (name, value, id)
                        elif type=="udpservice":
                                query = "UPDATE udpservices SET 
name='%s', ip='%s' WHERE id='%s'" % (name, value, id)
                        elif type=="servicegroup":
                                #t:1,2;u:1,2
                                query = "UPDATE servicegroups SET 
name='%s', ip='%s' WHERE id='%s'" % (name, value, id)

                elif command=="insert":
                        if type=="host":
                                query = "INSERT INTO hosts VALUES 
(NULL,'" + name + "','" + value + "')";
                        elif type=="network":
                                query = "INSERT INTO networks VALUES 
(NULL,'" + name + "','" + value + "')"
                        elif type=="iprange":
                                query = "INSERT INTO ipranges VALUES 
(NULL,'" + name + "','" + value + "')"
                        elif type=="ipgroup":
                                #h:1,2;n:1,2;i:1,2
                                query = "INSERT INTO ipgroups VALUES 
(NULL,'" + name + "','" + value + "')"
                        elif type=="tcpservice":
                                query = "INSERT INTO tcpservices 
VALUES (NULL,'" + name + "','" + value + "')"
                        elif type=="udpservice":
                                query = "INSERT INTO udpservices
VALUES (NULL,'" + name + "','" + value + "')"
                        elif type=="servicegroup":
                                #t:1,2;u:1,2
                                query = "INSERT INTO servicegroups
VALUES (NULL,'" + name + "','" + value + "')"

                return query+";"

        def updateObject(self, type, id, name, value):
                query = self.__createSQLQuery("update", type, id,
name, value)
                print query
                conn = sqlite3.connect('/opt/demir-
fw/var/firewall.db')
                c = conn.cursor()
                try:
                        c.execute(query)
                except:
                        raise UpdateObjectError

                conn.commit()
                c.close()

        def createObject(self, type, name, value):
                query = self.__createSQLQuery("insert", type, '-1',
name, value)

                conn = sqlite3.connect('/opt/demir-
fw/var/firewall.db')
                c = conn.cursor()
                try:
                        c.execute(query)
                except:
                        raise CreateObjectError

                conn.commit()
                c.close()

        def getObjById(self, obj, id):
                if obj=="host":
                        query = "SELECT * FROM hosts WHERE id = " +
str(id)
                elif obj=="network":
                        query = "SELECT * FROM networks WHERE id = "
+ str(id)
                elif obj=="iprange":
                        query = "SELECT * FROM ipranges WHERE id = "
+ str(id)
                elif obj=="ipgroup":
                        query = "SELECT * FROM ipgroups WHERE id = "
+ str(id)
                elif obj=="tcpservice":
                        query = "SELECT * FROM tcpservices WHERE id =
" + str(id)
                elif obj=="udpservice":
                        query = "SELECT * FROM udpservices WHERE id =
" + str(id)
```

```
                conn = sqlite3.connect('/opt/demir-
fw/var/firewall.db')
                c = conn.cursor()
                c.execute(query)
                x=c.fetchall()
                c.close()
                return x

        def getObjects(self, obj):
                if obj=="host":
                        query = "SELECT * FROM hosts"
                elif obj=="network":
                        query = "SELECT * FROM networks"
                elif obj=="iprange":
                        query = "SELECT * FROM ipranges"
                elif obj=="ipgroup":
                        query = "SELECT * FROM ipgroups"
                elif obj=="tcpservice":
                        query = "SELECT * FROM tcpservices"
                elif obj=="udpservice":
                        query = "SELECT * FROM udpservices"
                elif obj=="servicegroup":
                        query = "SELECT * FROM servicegroups"

                conn = sqlite3.connect('/opt/demir-
fw/var/firewall.db')
                c = conn.cursor()
                c.execute(query)
                x=c.fetchall()
                c.close()
                return x

        def submitObjects(self, type, objects):
                for object in objects:
                        id = int(object['id']); name =
object['name']; value = object['value']
                        if id == -1: self.createObject(type, name,
value)
                        else: self.updateObject(type, id, name,
value)
```

A.2. NetworkManager.py (Network Management Module)

```
#DemirFW Implementation for M.Sc. Thesis
#Necati Demir, 2009

from errors import *
from globals import *
from pfmanager import *
import os

class NetworkManager:

        def setConf(self, nconf):
                self.__nconf=nconf

        def __cleanNetworkFiles(self):
                hfiles = os.popen("ls /etc/hostname* 2>/dev/null ") #
hostname files
                for h in hfiles:
                        os.remove(h.replace('\n',''))
                try:
                        os.remove("/etc/mygate");
                        os.remove("/etc/resolv.conf");
                except: pass

        def submit(self,r=1):
                self.__cleanNetworkFiles()
                interfaces=getInterfaces()

                #Interface settings
                for i in self.__nconf['interfaces']:
                        f=open('/etc/hostname.'+interfaces[i], 'w')
                        if self.__nconf['interfaces'][i] == 'dhcp':
                                f.write('dhcp\n')
                                f.write('up\n')
                                f.close()
                                continue
                        #else if not dhcp

        ip=self.__nconf['interfaces'][i].split('/')[0]

        netmask=self.__nconf['interfaces'][i].split('/')[1]
                        f.write('inet '+ip+' '+netmask+' NONE\n')
                        f.write('up\n')
                        f.close()

                #Gateway and DNS settings
                if self.__nconf.has_key("gateway"):
                        f = open("/etc/mygate", "w")
                        f.write(self.__nconf['gateway']+"\n")
                        f.close()
                if self.__nconf.has_key("dns"):
                        f=open("/etc/resolv.conf","w")
                        dns=self.__nconf['dns'].split(',')
                        for d in dns: f.write("nameserver "+d+"\n")
                        f.close()

                if r==1: os.system("reboot")
                else: os.system("sh /etc/netstart")
```

A.3. DHCPD.py (DHCP Server Management Module)

```
#DemirFW Implementation for M.Sc. Thesis
#Necati Demir, 2009

import os, popen2
from errors import DHCPDManagerError

class DHCPDManager:

        def setConf(self, conf):
                self.conf = conf

        def submit(self):
                configFile=""
                f=open("/opt/demir-fw/etc/dhcpd.conf")
                for line in f:
                        configFile=configFile+line
                f.close()
                configFile=configFile.replace("#DNS#",
self.conf['dns'].replace(',', ','))
                configFile=configFile.replace("#SUBNET#",
self.conf['subnet'])
                configFile=configFile.replace("#NETMASK#",
self.conf['netmask'])
                configFile=configFile.replace("#ROUTER#",
self.conf['router'])
                configFile=configFile.replace("#RANGE#",
self.conf['range'].replace('-', ' '))

                f=open("/etc/dhcpd.interfaces", "w")
                f.write(getInterfaces()['int0'])
                f.close()

                f=open("/etc/dhcpd.conf.tmp", "w")
                f.write(configFile)
                f.close()

                os.system("ps aux | grep dhcpd| grep _dhcp| awk
'{print $2}'| xargs -n1 kill")
                os.system("touch /var/db/dhcpd.leases")
                child = popen2.Popen3("dhcpd -c /etc/dhcpd.conf.tmp")
                result = child.wait()
                if (result != 0):
                        os.system("dhcpd")
                        raise DHCPDManagerError
                else:
                        f=open("/etc/dhcpd.conf", "w")
                        f.write(configFile)
                        f.close()
```

A.4. FirewallManager.py (Firewall Management Module)

```
#DemirFW Implementation for M.Sc. Thesis
#Necati Demir, 2009

from globals import *
from pysqlite2 import dbapi2 as sqlite3
from errors import *
from ObjectManager import ObjectManager
import random, os, popen2

class FirewallManager:
        __rules = []
        __tables = []
        __conf = []

        def _createTable(self, name, table):
                objmgr = ObjectManager()

                self.__tables.append("table <" + name + "> persist
file '/opt/demir-fw/etc/pf/" + name + "'")
                f = open("/opt/demir-fw/etc/pf.temp/" + name, "w")
                if table.has_key("host"):
                        for id in table["host"]:
f.write(objmgr.getObjById("host", id)[0][2] + "\n")
                if table.has_key("network"):
                        for id in table["network"]:
f.write(objmgr.getObjById("network", id)[0][2] + "\n")
                if table.has_key("iprange"):
                        hosts =
self.convertIPRangesToHosts(table["iprange"])
                        for host in hosts: f.write(host + "\n")
f.write(objmgr.getObjById("iprange", id)[0][2] + "\n")
                f.close()

        def __createXRule(self, action, proto, source, sport,
target, tport):

                rule = action
                if proto != "":
                        rule = rule + " proto " + proto

                if source != "any":
                        table_name=str(random.random()).split('.')[1]
                        self._createTable(table_name, source)
                        rule += " from <" + table_name + "> "
                else:
                        rule += " from any "

                if sport != "": rule += " port " + sport

                if target != "any":
                        table_name=str(random.random()).split('.')[1]
                        self._createTable(table_name, target)
                        rule += " to <" + table_name + "> "
                else:
                        rule += " to any "

                if tport != "": rule += " port " + tport
                self.__rules.append(rule)

        def __createRule(self, action, source, target, port):
                objmgr = ObjectManager()
                if port == "any":
                        self.__createXRule(action, "", source, "",
target, "")
                else:
                        if port.has_key("tcpservice"):
                                for id in port["tcpservice"]:
                                        service =
objmgr.getObjById("tcpservice", id)[0][2]
                                        sport = service.split('-')[0]
                                        tport = service.split('-')[1]
                                        self.__createXRule(action,
"tcp", source, sport, target, tport)
                        if port.has_key("udpservice"):
                                for id in port["udpservice"]:
                                        service =
objmgr.getObjById("udpservice", id)[0][2]
                                        sport = service.split('-')[0]
                                        tport = service.split('-')[1]
                                        self.__createXRule(action,
"udp", source, sport, target, tport)

        def __newRule(self, rule):
                action = rule["action"]
                source = rule["source"]
                target = rule["target"]
                port = rule["port"]

                self.__createRule(action, source, target, port)


        def __submitRules(self):
                lines = open("/etc/pf.conf").readlines()
                linesLen = len(lines)

                for i in range(0, linesLen):
                        if (lines[i].strip().find("#RDR Rules
Start#") == 0): startIndex = i
                        if (lines[i].strip().find("#RDR Rules End#")
== 0): endIndex = i

                for i in range(startIndex+1, endIndex):
                        lines.pop(startIndex+1)

                i=0
                for rule in self.__rules:
                        i += 1
                        lines.insert(startIndex+i, rule + '\n')

                open("/etc/pf.conf","w").writelines(lines)

        def __createSimpleNAT(self):
                interfaces=getInterfaces()

                for i in interfaces:
```

```
        self.__conf.append(i+"='"+interfaces[i]+"'\n")

                fsn=open("/opt/demir-fw/etc/pf.simple.nat") #file
simple nat
                for line in fsn:
                        self.__conf.append(line)
                fsn.close()

        def __submitRules(self):
                os.system("rm -rf /tmp/pf; cp -rf /opt/demir-
fw/etc/pf/ /tmp/; rm -f /opt/demir-fw/etc/pf/*")
                os.system("mv -f /opt/demir-fw/etc/pf.temp/*
/opt/demir-fw/etc/pf/")

                self.__createSimpleNAT()

                f=open("/etc/pf.conf.test", "w")
                for line in self.__conf:
                        f.write(line)
                for line in self.__tables:
                        f.write(line + "\n")
                for line in self.__rules:
                        f.write(line + "\n")
                f.close()

                child = popen2.Popen3("pfctl -nf /etc/pf.conf.test");
                result = child.wait()
                if (result != 0): # if not successfull, rollback and
raise exception
                        os.system("rm -f /opt/demir-fw/etc/pf/*; mv -
f /tmp/pf/* /opt/demir-fw/etc/pf/; rm /etc/pf.conf.test")
                        raise PFManagerError
                else: #if successfull
                        os.system("mv /etc/pf.conf.test
/etc/pf.conf");
                        child = popen2.Popen3("pfctl -d; pfctl -e -Fa
-f /etc/pf.conf")
                        child.wait()

        def     submit(self, rules):
                print "submit"
                for rule in rules: self.__newRule(rule)
                self.__submitRules()
```

A.5. RedirectionManager.py (Port Redirection Manager Module)

```
#DemirFW Implementation for M.Sc. Thesis
#Necati Demir, 2009

from pysqlite2 import dbapi2 as sqlite3
from errors import *
from ObjectManager import ObjectManager
import random, os, popen2

class RedirectionManager:
        __rules = []

        def __newRule(self, rule):
                protocol = rule["protocol"]
                hostId = rule["hostId"]
                externalPortId = rule["externalPortId"]
                internalPortId = rule["internalPortId"]
                objmgr = ObjectManager()

                if protocol == "tcp":
                        service = objmgr.getObjById("tcpservice",
externalPortId)[0][2]
                        externalTPort = service.split('-')[1]

                        service = objmgr.getObjById("tcpservice",
internalPortId)[0][2]
                        internalTPort = service.split('-')[1]
                else:
                        service = objmgr.getObjById("udpservice",
externalPortId)[0][2]
                        externalTPort = service.split('-')[1]

                        service = objmgr.getObjById("udpservice",
internalPortId)[0][2]
                        internalTPort = service.split('-')[1]

                ip = objmgr.getObjById("host", hostId)[0][2]
                self.__createRule(protocol, ip, externalTPort,
internalTPort)

        def __createRule(self, proto, ip, externalTPort,
internalTPort):
                rule = """rdr on $ext0 proto %s from any to any port
%s -> %s port %s """ % (proto, externalTPort, ip, internalTPort)
                self.__rules.append(rule)

        def __submitRules(self):
                lines = open("/etc/pf.conf").readlines()
                linesLen = len(lines)

                for i in range(0, linesLen):
                        if (lines[i].strip().find("#RDR Rules
Start#") == 0): startIndex = i
                        if (lines[i].strip().find("#RDR Rules End#")
== 0): endIndex = i

                for i in range(startIndex+1, endIndex):
                        lines.pop(startIndex+1)

                i=0
                for rule in self.__rules:
                        i += 1
                        lines.insert(startIndex+i, rule + '\n')

                open("/etc/pf.conf","w").writelines(lines)

                child = popen2.Popen3("pfctl -d; pfctl -e -Fa -f
/etc/pf.conf")
                child.wait()


        def     submit(self, rules):
                for rule in rules: self.__newRule(rule)
                self.__submitRules()
```

APPENDIX B
SOURCE CODE OF MODULE CLIENTS

B.1. ObjectFrame.java (Object Management Module Client)

```
/*DemirFW Implementation for M.Sc. Thesis
Necati Demir, 2009*/

package demirfwclient;
import demirfwclient.ObjectsTreePanel.treeRootObjects;
import javax.swing.JTree;
import javax.swing.event.TreeSelectionEvent;
import javax.swing.event.TreeSelectionListener;
import javax.swing.tree.DefaultMutableTreeNode;

public class ObjectsFrame extends javax.swing.JInternalFrame {

    /** Creates new form ObjectsFrame */
    private JTree objectsTree;

    public void customInit(){
        objectsTree = objectsTreePanel.getObjectsTree();

        objectsTree.addTreeSelectionListener(new
TreeSelectionListener(){

            public void valueChanged(TreeSelectionEvent e) {
                try{
                FirewallObject fwObj =
((FirewallObject)(((DefaultMutableTreeNode)(e.getPath().getPath()[2]
)).getUserObject()));
                objectIdField.setText( String.valueOf(fwObj.id) );
                objectNameField.setText(fwObj.name);
                objectValueField.setText(fwObj.value);
                saveupdateButton.setText("Update");
                }catch(Exception ex){
                saveupdateButton.setText("Save");
                objectIdField.setText("-1");
                objectNameField.setText("");
                objectValueField.setText("");
                }

            }
        });
    }

    public ObjectsFrame() {
        initComponents();
        customInit();
    }

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated
Code">//GEN-BEGIN:initComponents
    private void initComponents() {
        objectsTreePanel = new demirfwclient.ObjectsTreePanel();
        reloadButton = new javax.swing.JButton();
        objectIdLabel = new javax.swing.JLabel();
        objectIdField = new javax.swing.JTextField();
        objectNameLabel = new javax.swing.JLabel();
        objectNameField = new javax.swing.JTextField();
        objectValueField = new javax.swing.JTextField();
        objectValueLabel = new javax.swing.JLabel();
        saveupdateButton = new javax.swing.JButton();
        submitButton = new javax.swing.JButton();

        setClosable(true);
        setName("Form"); // NOI18N

        objectsTreePanel.setName("objectsTreePanel"); // NOI18N

        org.jdesktop.application.ResourceMap resourceMap =
org.jdesktop.application.Application.getInstance(demirfwclient.Demir
FWClientApp.class).getContext().getResourceMap(ObjectsFrame.class);

reloadButton.setText(resourceMap.getString("reloadButton.text")); //
NOI18N
        reloadButton.setName("reloadButton"); // NOI18N
        reloadButton.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                reloadButtonActionPerformed(evt);
            }
        });
        reloadButton.addMouseListener(new
java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt)
{
                reloadButtonMouseClicked(evt);
            }
        });


objectIdLabel.setText(resourceMap.getString("objectIdLabel.text"));
// NOI18N
        objectIdLabel.setName("objectIdLabel"); // NOI18N


objectIdField.setText(resourceMap.getString("objectIdField.text"));
// NOI18N
        objectIdField.setName("objectIdField"); // NOI18N


objectNameLabel.setText(resourceMap.getString("objectNameLabel.text"
)); // NOI18N
        objectNameLabel.setName("objectNameLabel"); // NOI18N

        objectNameField.setName("objectNameField"); // NOI18N

        objectValueField.setName("objectValueField"); // NOI18N


objectValueLabel.setText(resourceMap.getString("objectValueLabel.tex
t")); // NOI18N
```

```
        objectValueLabel.setName("objectValueLabel"); // NOI18N


saveupdateButton.setText(resourceMap.getString("saveupdateButton.tex
t")); // NOI18N
        saveupdateButton.setName("saveupdateButton"); // NOI18N
        saveupdateButton.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                saveupdateButtonActionPerformed(evt);
            }
        });


submitButton.setText(resourceMap.getString("submitButton.text")); //
NOI18N
        submitButton.setName("submitButton"); // NOI18N
        submitButton.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                submitButtonActionPerformed(evt);
            }
        });

        org.jdesktop.layout.GroupLayout layout = new
org.jdesktop.layout.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .addContainerGap()

.add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEAD
ING)
                    .add(layout.createSequentialGroup()

.add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEAD
ING)
                            .add(reloadButton,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 101,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                            .add(layout.createSequentialGroup()
                                .add(objectsTreePanel,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)

.add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.TRAI
LING, false)
                                    .add(objectIdLabel,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

.add(org.jdesktop.layout.GroupLayout.LEADING, objectNameLabel,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

.add(org.jdesktop.layout.GroupLayout.LEADING, objectValueLabel,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 51, Short.MAX_VALUE))

.addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)

.add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEAD
ING)
                                    .add(saveupdateButton,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 146, Short.MAX_VALUE)

.add(org.jdesktop.layout.GroupLayout.TRAILING, objectIdField,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 146, Short.MAX_VALUE)
                                    .add(objectNameField,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 146, Short.MAX_VALUE)
                                    .add(objectValueField,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 146,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))))
                        .addContainerGap(20,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                    .add(org.jdesktop.layout.GroupLayout.TRAILING,
layout.createSequentialGroup()
                        .add(submitButton)
                        .addContainerGap())))
        );
        layout.setVerticalGroup(

layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .addContainerGap()
                .add(reloadButton)

.addPreferredGap(org.jdesktop.layout.LayoutStyle.UNRELATED)

.add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEAD
ING)
                    .add(layout.createSequentialGroup()

.add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASE
LINE)
                            .add(objectIdLabel)
                            .add(objectIdField,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)

.add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASE
LINE)
                            .add(objectNameLabel)
                            .add(objectNameField,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
```

```
.add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASE
LINE)
                            .add(objectValueLabel)
                            .add(objectValueField,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(org.jdesktop.layout.LayoutStyle.UNRELATED)
                        .add(saveupdateButton,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 30,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                        .add(75, 75, 75)
                        .add(submitButton))
                    .add(objectsTreePanel,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                .addContainerGap())
        );

        pack();
    }// </editor-fold>//GEN-END:initComponents

private void reloadButtonMouseClicked(java.awt.event.MouseEvent evt)
{//GEN-FIRST:event_reloadButtonMouseClicked

}//GEN-LAST:event_reloadButtonMouseClicked

private void
saveupdateButtonActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_saveupdateButtonActionPerformed
    treeRootObjects selectedType =
objectsTreePanel.getSelectedType();

    if (objectIdField.getText().equals("-1")){
        //New object
        FirewallObject fwObj = new FirewallObject();
        fwObj.type = FirewallObject.TYPE.values()[
selectedType.ordinal() ];
        fwObj.id = -1;
        fwObj.name = objectNameField.getText();
        fwObj.value = objectValueField.getText();
        objectsTreePanel.addTreeObject(selectedType, fwObj);
    }else{
        objectsTreePanel.updateObjectInTree(selectedType,
Integer.parseInt(objectIdField.getText()) ,
objectNameField.getText() , objectValueField.getText());
    }
}//GEN-LAST:event_saveupdateButtonActionPerformed

private void submitButtonActionPerformed(java.awt.event.ActionEvent
evt) {//GEN-FIRST:event_submitButtonActionPerformed
    objectsTreePanel.submitObjects();
}//GEN-LAST:event_submitButtonActionPerformed

private void reloadButtonActionPerformed(java.awt.event.ActionEvent
evt) {//GEN-FIRST:event_reloadButtonActionPerformed
```

```
//objectsTreePanel.setServerURL("http://10.1.1.2:81/xmlrpcserver/ind
ex.cgi");
    objectsTreePanel.loadObjectsTree();
}//GEN-LAST:event_reloadButtonActionPerformed

    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JTextField objectIdField;
    private javax.swing.JLabel objectIdLabel;
    private javax.swing.JTextField objectNameField;
    private javax.swing.JLabel objectNameLabel;
    private javax.swing.JTextField objectValueField;
    private javax.swing.JLabel objectValueLabel;
    private demirfwclient.ObjectsTreePanel objectsTreePanel;
    private javax.swing.JButton reloadButton;
    private javax.swing.JButton saveupdateButton;
    private javax.swing.JButton submitButton;
    // End of variables declaration//GEN-END:variables
}
```

B.2. NetworkFrame.java (Network Management Module Client)

```
/*DemirFW Implementation for M.Sc. Thesis
Necati Demir, 2009*/

package demirfwclient;
import java.io.ByteArrayInputStream;
import java.net.MalformedURLException;
import java.net.URL;
import java.util.ArrayList;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import org.apache.xmlrpc.client.XmlRpcClient;
import org.apache.xmlrpc.client.XmlRpcClientConfigImpl;
import org.w3c.dom.CharacterData;
import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;

public class NetworkConfigurationFrame extends
javax.swing.JInternalFrame {

    /** Creates new form NetworkConfigurationFrame */
    public NetworkConfigurationFrame() {
        initComponents();
    }

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated
Code">//GEN-BEGIN:initComponents
    private void initComponents() {

        jPanel2 = new javax.swing.JPanel();
        jLabel1 = new javax.swing.JLabel();
        jLabel2 = new javax.swing.JLabel();
        externalIPInput = new javax.swing.JTextField();
        InternalIPInput = new javax.swing.JTextField();
        jLabel3 = new javax.swing.JLabel();
        dnsInput = new javax.swing.JTextField();
        jLabel4 = new javax.swing.JLabel();
        gatewayInput = new javax.swing.JTextField();
        submitNetworkConfigurationButton = new
javax.swing.JButton();
        loadNetworkConfigurationButton = new javax.swing.JButton();
        externalDHCPCheckBox = new javax.swing.JCheckBox();

        setClosable(true);
        setName("Form"); // NOI18N

        jPanel2.setName("jPanel2"); // NOI18N
        jPanel2.setLayout(new
org.netbeans.lib.awtextra.AbsoluteLayout());

        org.jdesktop.application.ResourceMap resourceMap =
org.jdesktop.application.Application.getInstance(demirfwclient.Demir
FWClientApp.class).getContext().getResourceMap(NetworkConfigurationF
rame.class);
        jLabel1.setText(resourceMap.getString("jLabel1.text")); //
NOI18N
        jLabel1.setName("jLabel1"); // NOI18N
        jPanel2.add(jLabel1, new
org.netbeans.lib.awtextra.AbsoluteConstraints(10, 10, 120, 20));

        jLabel2.setText(resourceMap.getString("jLabel2.text")); //
NOI18N
        jLabel2.setName("jLabel2"); // NOI18N
        jPanel2.add(jLabel2, new
org.netbeans.lib.awtextra.AbsoluteConstraints(420, 10, 120, 20));


externalIPInput.setText(resourceMap.getString("externalIPInput.text"
)); // NOI18N
        externalIPInput.setName("externalIPInput"); // NOI18N
        externalIPInput.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                externalIPInputActionPerformed(evt);
            }
        });
        jPanel2.add(externalIPInput, new
org.netbeans.lib.awtextra.AbsoluteConstraints(130, 10, 140, 20));


InternalIPInput.setText(resourceMap.getString("InternalIPInput.text"
)); // NOI18N
        InternalIPInput.setName("InternalIPInput"); // NOI18N
        InternalIPInput.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                InternalIPInputActionPerformed(evt);
            }
        });
        jPanel2.add(InternalIPInput, new
org.netbeans.lib.awtextra.AbsoluteConstraints(420, 30, 150, 20));

        jLabel3.setText(resourceMap.getString("jLabel3.text")); //
NOI18N
        jLabel3.setName("jLabel3"); // NOI18N
        jPanel2.add(jLabel3, new
org.netbeans.lib.awtextra.AbsoluteConstraints(10, 30, 110, 20));

        dnsInput.setName("dnsInput"); // NOI18N
        dnsInput.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                dnsInputActionPerformed(evt);
            }
        });
        jPanel2.add(dnsInput, new
org.netbeans.lib.awtextra.AbsoluteConstraints(130, 30, 140, 20));
```

```
        jLabel4.setText(resourceMap.getString("jLabel4.text")); //
NOI18N
        jLabel4.setName("jLabel4"); // NOI18N
        jPanel2.add(jLabel4, new
org.netbeans.lib.awtextra.AbsoluteConstraints(10, 50, 110, -1));

        gatewayInput.setName("gatewayInput"); // NOI18N
        gatewayInput.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                gatewayInputActionPerformed(evt);
            }
        });
        jPanel2.add(gatewayInput, new
org.netbeans.lib.awtextra.AbsoluteConstraints(130, 50, 140, 20));


submitNetworkConfigurationButton.setText(resourceMap.getString("subm
itNetworkConfigurationButton.text")); // NOI18N

submitNetworkConfigurationButton.setName("submitNetworkConfiguration
Button"); // NOI18N
        submitNetworkConfigurationButton.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {

submitNetworkConfigurationButtonActionPerformed(evt);
            }
        });
        jPanel2.add(submitNetworkConfigurationButton, new
org.netbeans.lib.awtextra.AbsoluteConstraints(20, 90, 150, 30));


loadNetworkConfigurationButton.setText(resourceMap.getString("loadNe
tworkConfigurationButton.text")); // NOI18N

loadNetworkConfigurationButton.setName("loadNetworkConfigurationButt
on"); // NOI18N
        loadNetworkConfigurationButton.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                loadNetworkConfigurationButtonActionPerformed(evt);
            }
        });
        jPanel2.add(loadNetworkConfigurationButton, new
org.netbeans.lib.awtextra.AbsoluteConstraints(390, 90, 180, 30));


externalDHCPCheckBox.setText(resourceMap.getString("externalDHCPChec
kBox.text")); // NOI18N
        externalDHCPCheckBox.setName("externalDHCPCheckBox"); //
NOI18N
        externalDHCPCheckBox.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                externalDHCPCheckBoxActionPerformed(evt);
            }
        });
        jPanel2.add(externalDHCPCheckBox, new
org.netbeans.lib.awtextra.AbsoluteConstraints(270, 10, -1, -1));

        org.jdesktop.layout.GroupLayout layout = new
org.jdesktop.layout.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .addContainerGap()
                .add(jPanel2,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(67, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(

layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .addContainerGap()
                .add(jPanel2,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 154,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)

.addContainerGap(org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );

        pack();
    }// </editor-fold>//GEN-END:initComponents

private void
externalIPInputActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_externalIPInputActionPerformed
// TODO add your handling code here:
}//GEN-LAST:event_externalIPInputActionPerformed

private void
InternalIPInputActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_InternalIPInputActionPerformed
// TODO add your handling code here:
}//GEN-LAST:event_InternalIPInputActionPerformed

private void dnsInputActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_dnsInputActionPerformed
// TODO add your handling code here:
}//GEN-LAST:event_dnsInputActionPerformed

private void gatewayInputActionPerformed(java.awt.event.ActionEvent
evt) {//GEN-FIRST:event_gatewayInputActionPerformed
// TODO add your handling code here:
}//GEN-LAST:event_gatewayInputActionPerformed

private void
submitNetworkConfigurationButtonActionPerformed(java.awt.event.Actio
nEvent evt) {//GEN-
FIRST:event_submitNetworkConfigurationButtonActionPerformed
    StringBuffer strXML = new StringBuffer();
    strXML.append("<demir-fw>\n");
    strXML.append("<network>\n");

    strXML.append("<interfaces>\n");
    strXML.append("<ext0>");
    if (externalDHCPCheckBox.isSelected())
        strXML.append("dhcp");
    else
        strXML.append(externalIPInput.getText());
    strXML.append("</ext0>\n");
    strXML.append("<int0>");
    strXML.append(InternalIPInput.getText());
    strXML.append("</int0>\n");
    strXML.append("</interfaces>\n");

    if (! externalDHCPCheckBox.isSelected()){
        strXML.append("<gateway>\n");
        strXML.append(gatewayInput.getText());
        strXML.append("</gateway>\n");

        strXML.append("<dns>\n");
        strXML.append(dnsInput.getText());
        strXML.append("</dns>\n");
    }
    strXML.append("</network>\n");
    strXML.append("</demir-fw>\n");

    ArrayList<String> parameters = new ArrayList<String>();
    parameters.add(strXML.toString());
    String returnXML =
ServerConnection.callMethod("submitNetworkConfiguration",
parameters);



}//GEN-LAST:event_submitNetworkConfigurationButtonActionPerformed

private String getCharacterDataFromElement(Element e) {
        try {
                Node child = e.getFirstChild();
                if (child instanceof CharacterData) {
                        CharacterData cd = (CharacterData) child;
                        return cd.getData();
                }
        } catch (Exception ex) {
        }
        return "";
    }

    protected String getElementValue(Element parent, String label) {
            return getCharacterDataFromElement((Element) parent
                            .getElementsByTagName(label).item(0));
    }

private void
loadNetworkConfigurationButtonActionPerformed(java.awt.event.ActionE
vent evt) {//GEN-
FIRST:event_loadNetworkConfigurationButtonActionPerformed
        try {

            String XMLString =
ServerConnection.callMethod("getNetworkConfiguration", null);


            XMLParser xmlParser = new XMLParser(XMLString);
            InternalIPInput.setText(
xmlParser.getValue("interfaces", "int0") );

            String externalIP = xmlParser.getValue("interfaces",
"ext0");
            if (externalIP.equals("dhcp")){
                externalDHCPCheckBox.setSelected(true);
                externalDHCPCheckBoxActionPerformed();
            }else{
                externalDHCPCheckBox.setSelected(false);
                externalDHCPCheckBoxActionPerformed();
                externalIPInput.setText(externalIP);
                gatewayInput.setText( xmlParser.getValue("network",
"gateway") );
                dnsInput.setText( xmlParser.getValue("network",
"dns") );

            }


            /*DocumentBuilder builder =
DocumentBuilderFactory.newInstance()
                    .newDocumentBuilder();
            Document doc = builder.parse(new
ByteArrayInputStream(resultXML.getBytes()));
            NodeList nodes = doc.getElementsByTagName("interfaces");

            Element element = (Element) nodes.item(0);
            externalIPInput.setText( getElementValue(element,
"ext0") );
            InternalIPInput.setText( getElementValue(element,
"int0") );

            nodes =
doc.getElementsByTagName("networkConfiguration");
            element = (Element) nodes.item(0);
            dnsInput.setText( getElementValue(element, "dns") );
            gatewayInput.setText( getElementValue(element,
"gateway") );

                    } catch (Exception ex) {

Logger.getLogger(NetworkConfigurationFrame.class.getName()).log(Leve
l.SEVERE, null, ex);
        }

}//GEN-LAST:event_loadNetworkConfigurationButtonActionPerformed

private void externalDHCPCheckBoxActionPerformed() {
        if(externalDHCPCheckBox.isSelected()){
        externalIPInput.setEnabled(false);
```

```
            dnsInput.setEditable(false);
            gatewayInput.setEditable(false);
        }else{
            externalIPInput.setEnabled(true);
            dnsInput.setEditable(true);
            gatewayInput.setEditable(true);
        }
    }


    private void
    externalDHCPCheckBoxActionPerformed(java.awt.event.ActionEvent evt)
    {//GEN-FIRST:event_externalDHCPCheckBoxActionPerformed
        externalDHCPCheckBoxActionPerformed();
    }//GEN-LAST:event_externalDHCPCheckBoxActionPerformed


        // Variables declaration - do not modify//GEN-BEGIN:variables
        private javax.swing.JTextField InternalIPInput;
        private javax.swing.JTextField dnsInput;
        private javax.swing.JCheckBox externalDHCPCheckBox;
        private javax.swing.JTextField externalIPInput;
        private javax.swing.JTextField gatewayInput;
        private javax.swing.JLabel jLabel1;
        private javax.swing.JLabel jLabel2;
        private javax.swing.JLabel jLabel3;
        private javax.swing.JLabel jLabel4;
        private javax.swing.JPanel jPanel2;
        private javax.swing.JButton loadNetworkConfigurationButton;
        private javax.swing.JButton submitNetworkConfigurationButton;
        // End of variables declaration//GEN-END:variables

    }
```

B.3. DHCPDFrame.java (DHCP Server Management Module Client)

```
#DemirFW Implementation for M.Sc. Thesis
#Necati Demir, 2009

package demirfwclient;
import java.io.ByteArrayInputStream;
import java.net.URL;
import java.util.ArrayList;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JOptionPane;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import org.apache.xmlrpc.client.XmlRpcClient;
import org.apache.xmlrpc.client.XmlRpcClientConfigImpl;
import org.foafscape.security.util.Convert;
import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.NodeList;


public class DHCPDFrame extends javax.swing.JInternalFrame {

    /** Creates new form DHCPDFrame */
    public DHCPDFrame() {
        initComponents();
    }

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated
Code">//GEN-BEGIN:initComponents
    private void initComponents() {

        jPanel1 = new javax.swing.JPanel();
        jLabel1 = new javax.swing.JLabel();
        routerInput = new javax.swing.JTextField();
        jLabel2 = new javax.swing.JLabel();
        subnetInput = new javax.swing.JTextField();
        jLabel3 = new javax.swing.JLabel();
        netmaskInput = new javax.swing.JTextField();
        jLabel4 = new javax.swing.JLabel();
        rangeInput = new javax.swing.JTextField();
        jLabel5 = new javax.swing.JLabel();
        dnsInput = new javax.swing.JTextField();
        submitButton = new javax.swing.JButton();
        loadButton = new javax.swing.JButton();

        setClosable(true);
        setName("Form"); // NOI18N
        getContentPane().setLayout(new
org.netbeans.lib.awtextra.AbsoluteLayout());

        jPanel1.setName("jPanel1"); // NOI18N
        jPanel1.setLayout(new
org.netbeans.lib.awtextra.AbsoluteLayout());
```

```
        org.jdesktop.application.ResourceMap resourceMap =
org.jdesktop.application.Application.getInstance(demirfwclient.Demir
FWClientApp.class).getContext().getResourceMap(DHCPDFrame.class);
        jLabel1.setText(resourceMap.getString("jLabel1.text")); //
NOI18N
        jLabel1.setName("jLabel1"); // NOI18N
        jPanel1.add(jLabel1, new
org.netbeans.lib.awtextra.AbsoluteConstraints(10, 20, -1, -1));


routerInput.setText(resourceMap.getString("routerInput.text")); //
NOI18N
        routerInput.setName("routerInput"); // NOI18N
        jPanel1.add(routerInput, new
org.netbeans.lib.awtextra.AbsoluteConstraints(90, 20, 140, -1));

        jLabel2.setText(resourceMap.getString("jLabel2.text")); //
NOI18N
        jLabel2.setName("jLabel2"); // NOI18N
        jPanel1.add(jLabel2, new
org.netbeans.lib.awtextra.AbsoluteConstraints(10, 50, -1, -1));

        subnetInput.setName("subnetInput"); // NOI18N
        jPanel1.add(subnetInput, new
org.netbeans.lib.awtextra.AbsoluteConstraints(90, 50, 140, -1));

        jLabel3.setText(resourceMap.getString("jLabel3.text")); //
NOI18N
        jLabel3.setName("jLabel3"); // NOI18N
        jPanel1.add(jLabel3, new
org.netbeans.lib.awtextra.AbsoluteConstraints(10, 80, -1, -1));

        netmaskInput.setName("netmaskInput"); // NOI18N
        jPanel1.add(netmaskInput, new
org.netbeans.lib.awtextra.AbsoluteConstraints(90, 80, 140, -1));

        jLabel4.setText(resourceMap.getString("jLabel4.text")); //
NOI18N
        jLabel4.setName("jLabel4"); // NOI18N
        jPanel1.add(jLabel4, new
org.netbeans.lib.awtextra.AbsoluteConstraints(10, 110, -1, -1));

        rangeInput.setName("rangeInput"); // NOI18N
        jPanel1.add(rangeInput, new
org.netbeans.lib.awtextra.AbsoluteConstraints(90, 110, 140, -1));

        jLabel5.setText(resourceMap.getString("jLabel5.text")); //
NOI18N
        jLabel5.setName("jLabel5"); // NOI18N
        jPanel1.add(jLabel5, new
org.netbeans.lib.awtextra.AbsoluteConstraints(10, 140, -1, -1));

        dnsInput.setName("dnsInput"); // NOI18N
        jPanel1.add(dnsInput, new
org.netbeans.lib.awtextra.AbsoluteConstraints(90, 140, 140, -1));


submitButton.setText(resourceMap.getString("submitButton.text")); //
NOI18N
        submitButton.setName("submitButton"); // NOI18N
        submitButton.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                submitButtonActionPerformed(evt);
            }
        });
        jPanel1.add(submitButton, new
org.netbeans.lib.awtextra.AbsoluteConstraints(280, 140, -1, -1));


loadButton.setText(resourceMap.getString("loadButton.text")); //
NOI18N
        loadButton.setName("loadButton"); // NOI18N
        loadButton.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                loadButtonActionPerformed(evt);
            }
        });
        jPanel1.add(loadButton, new
org.netbeans.lib.awtextra.AbsoluteConstraints(280, 20, 80, -1));

        getContentPane().add(jPanel1, new
org.netbeans.lib.awtextra.AbsoluteConstraints(10, 10, 520, 200));

        pack();
    }// </editor-fold>//GEN-END:initComponents

    private void
loadButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_loadButtonActionPerformed
    try{

        String XMLString =
ServerConnection.callMethod("getDHCPDConfiguration", null);
        //String XMLString =
Convert.base64ToString(encodedXMLString);

        XMLParser xmlParser = new XMLParser(XMLString);
        routerInput.setText( xmlParser.getValue("dhcpd", "router")
);
        subnetInput.setText(xmlParser.getValue("dhcpd", "subnet"));
        netmaskInput.setText(xmlParser.getValue("dhcpd",
"netmask"));
        rangeInput.setText(xmlParser.getValue("dhcpd", "range"));
        dnsInput.setText(xmlParser.getValue("dhcpd", "dns"));

    }catch(Exception ex){

Logger.getLogger(DemirFWClientView.class.getName()).log(Level.SEVERE
, null, ex);

    }

    }//GEN-LAST:event_loadButtonActionPerformed
    private void
submitButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_submitButtonActionPerformed
        StringBuffer XMLString =  new StringBuffer();
        XMLString.append("<demir-fw>\n");
        XMLString.append(" <dhcpd>\n");
        XMLString.append("
<router>").append(routerInput.getText()).append("</router>\n");
        XMLString.append("
<subnet>").append(subnetInput.getText()).append("</subnet>\n");
        XMLString.append("
<netmask>").append(netmaskInput.getText()).append("</netmask>\n");
        XMLString.append("
<range>").append(rangeInput.getText()).append("</range>\n");
        XMLString.append("
<dns>").append(dnsInput.getText()).append("</dns>\n");
        XMLString.append(" </dhcpd>\n");
        XMLString.append("</demir-fw>\n");
        //

        ArrayList<String> parameters = new ArrayList<String>();
        parameters.add(XMLString.toString());
        String returnXML =
ServerConnection.callMethod("submitDHCPDConfiguration", parameters);
        JOptionPane.showMessageDialog(null, new
XMLParser(returnXML).getValue("demir-fw", "response"));

    }//GEN-LAST:event_submitButtonActionPerformed


    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JTextField dnsInput;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JButton loadButton;
    private javax.swing.JTextField netmaskInput;
    private javax.swing.JTextField rangeInput;
    private javax.swing.JTextField routerInput;
    private javax.swing.JButton submitButton;
    private javax.swing.JTextField subnetInput;
    // End of variables declaration//GEN-END:variables

}
```

B.4. FirewallFrame.java (Firewall Management Module Client)

```
#DemirFW Implementation for M.Sc. Thesis
#Necati Demir, 2009

package demirfwclient;
import java.awt.Point;
import java.util.logging.Level;
import java.util.logging.Logger;
import java.io.ByteArrayInputStream;
import java.net.URL;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.Vector;
import javax.swing.JMenu;
import javax.swing.JMenuItem;
import javax.swing.JPopupMenu;
import javax.swing.SwingUtilities;
import javax.swing.table.DefaultTableModel;
import javax.swing.tree.DefaultMutableTreeNode;
import javax.swing.tree.DefaultTreeModel;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import org.apache.xmlrpc.client.XmlRpcClient;
import org.apache.xmlrpc.client.XmlRpcClientConfigImpl;
import org.w3c.dom.CharacterData;
import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;
import org.foafscape.security.util.Convert;


public class FirewallFrame2 extends javax.swing.JInternalFrame {

    HashMap firewallObjectArray = new HashMap();
    private Thread reloadTreeThread = null;
    enum treeRootObjects { HOSTS, NETWORKS, IPRANGES, IPGROUPS,
TCPSERVICES, UDPSERVICES, SERVICEGROUPS }
    treeRootObjects type;
    enum ruleTableElements {ACTION, SOURCE, TARGET, PORT}

    /** Creates new form FirewallFrame2 */
    public FirewallFrame2() {
        initComponents();
    }


    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated
Code">//GEN-BEGIN:initComponents
    private void initComponents() {

        jPanel1 = new javax.swing.JPanel();
        objectsTreePanel = new demirfwclient.ObjectsTreePanel();
        reloadButton = new javax.swing.JButton();
        ruleTableScrollPane = new javax.swing.JScrollPane();
        ruleTable = new javax.swing.JTable();
        loadRedirectionRulesButton = new javax.swing.JButton();
        newFirewallRuleButton = new javax.swing.JButton();
        removeFirewallButton = new javax.swing.JButton();
        submitFirewallButton = new javax.swing.JButton();

        setClosable(true);
        setName("Form"); // NOI18N

        jPanel1.setName("jPanel1"); // NOI18N
        jPanel1.setLayout(new
org.netbeans.lib.awtextra.AbsoluteLayout());

        objectsTreePanel.setName("objectsTreePanel"); // NOI18N
        jPanel1.add(objectsTreePanel, new
org.netbeans.lib.awtextra.AbsoluteConstraints(10, 40, -1, -1));

        org.jdesktop.application.ResourceMap resourceMap =
org.jdesktop.application.Application.getInstance(demirfwclient.Demir
FWClientApp.class).getContext().getResourceMap(FirewallFrame2.class)
;

reloadButton.setText(resourceMap.getString("reloadButton.text")); //
NOI18N
        reloadButton.setName("reloadButton"); // NOI18N
        reloadButton.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                reloadButtonActionPerformed(evt);
            }
        });
        reloadButton.addMouseListener(new
java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt)
{
                reloadButtonMouseClicked(evt);
            }
        });
        jPanel1.add(reloadButton, new
org.netbeans.lib.awtextra.AbsoluteConstraints(10, 10, 300, -1));

        ruleTableScrollPane.setName("ruleTableScrollPane"); //
NOI18N

        String col[] = {"Action","Source","Target","Port"};
        DefaultTableModel dm = new DefaultTableModel(null, col) {
            public Class getColumnClass(int columnIndex) {
                return String.class;
            }
            public boolean isCellEditable(int row, int column) {
                return false; // Super returns true.
            }
        };
        ruleTable.setModel(dm);
        ruleTable.setName("ruleTable"); // NOI18N
        ruleTable.addMouseListener(new java.awt.event.MouseAdapter()
{
            public void mouseClicked(java.awt.event.MouseEvent evt)
{
                ruleTableMouseClicked(evt);
            }
        });
        ruleTableScrollPane.setViewportView(ruleTable);
        ruleTable.setDefaultRenderer(String.class, new
MultiLineCellRenderer());
        ruleTable.setRowHeight(ruleTable.getRowHeight() * 3);

        jPanel1.add(ruleTableScrollPane, new
org.netbeans.lib.awtextra.AbsoluteConstraints(320, 40, 640, 510));


loadRedirectionRulesButton.setText(resourceMap.getString("loadRedire
ctionRulesButton.text")); // NOI18N

loadRedirectionRulesButton.setName("loadRedirectionRulesButton"); //
NOI18N
        loadRedirectionRulesButton.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                loadRedirectionRulesButtonActionPerformed(evt);
            }
        });
        jPanel1.add(loadRedirectionRulesButton, new
org.netbeans.lib.awtextra.AbsoluteConstraints(960, 50, 190, -1));


newFirewallRuleButton.setText(resourceMap.getString("newFirewallRule
Button.text")); // NOI18N
        newFirewallRuleButton.setName("newFirewallRuleButton"); //
NOI18N
        newFirewallRuleButton.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                newFirewallRuleButtonActionPerformed(evt);
            }
        });
        jPanel1.add(newFirewallRuleButton, new
org.netbeans.lib.awtextra.AbsoluteConstraints(960, 80, 190, -1));


removeFirewallButton.setText(resourceMap.getString("removeFirewallBu
tton.text")); // NOI18N
        removeFirewallButton.setName("removeFirewallButton"); //
NOI18N
        removeFirewallButton.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                removeFirewallButtonActionPerformed(evt);
            }
        });
        jPanel1.add(removeFirewallButton, new
org.netbeans.lib.awtextra.AbsoluteConstraints(960, 110, 190, -1));


submitFirewallButton.setText(resourceMap.getString("submitFirewallBu
tton.text")); // NOI18N
```

```
        submitFirewallButton.setName("submitFirewallButton"); //
NOI18N
        submitFirewallButton.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                submitFirewallButtonActionPerformed(evt);
            }
        });
        jPanel1.add(submitFirewallButton, new
org.netbeans.lib.awtextra.AbsoluteConstraints(960, 150, 190, -1));

        org.jdesktop.layout.GroupLayout layout = new
org.jdesktop.layout.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .addContainerGap()
                .add(jPanel1,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 1160, Short.MAX_VALUE)
                .addContainerGap())
        );
        layout.setVerticalGroup(

layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(org.jdesktop.layout.GroupLayout.TRAILING,
layout.createSequentialGroup()
                .addContainerGap()
                .add(jPanel1,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 550, Short.MAX_VALUE)
                .addContainerGap())
        );

        pack();
    }// </editor-fold>//GEN-END:initComponents

    private void
reloadButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_reloadButtonActionPerformed

//objectsTreePanel.setServerURL("http://10.1.1.2:81/xmlrpcserver/ind
ex.cgi");
        objectsTreePanel.loadObjectsTree(new
ObjectsTreePanel.treeRootObjects[]{
            ObjectsTreePanel.treeRootObjects.HOSTS,
            ObjectsTreePanel.treeRootObjects.NETWORKS,
            ObjectsTreePanel.treeRootObjects.IPRANGES,
            ObjectsTreePanel.treeRootObjects.IPGROUPS,
            ObjectsTreePanel.treeRootObjects.TCPSERVICES,
            ObjectsTreePanel.treeRootObjects.UDPSERVICES,
            ObjectsTreePanel.treeRootObjects.SERVICEGROUPS,});



    }//GEN-LAST:event_reloadButtonActionPerformed

    private void reloadButtonMouseClicked(java.awt.event.MouseEvent
evt) {//GEN-FIRST:event_reloadButtonMouseClicked
```

```
}//GEN-LAST:event_reloadButtonMouseClicked

    private void ruleTableMouseClicked(java.awt.event.MouseEvent
evt) {//GEN-FIRST:event_ruleTableMouseClicked
        Point p = evt.getPoint();
        int rowNumber = ruleTable.rowAtPoint(p);
        int columnNumber = ruleTable.columnAtPoint(p);
        if ( SwingUtilities.isLeftMouseButton( evt ) ){
            if (evt.getClickCount() == 2){ /*double click*/
                if (columnNumber ==
ruleTableElements.ACTION.ordinal()){
                    if (ruleTable.getValueAt(rowNumber,
columnNumber) == ActionObject.ACTION.PASS){

ruleTable.setValueAt(ActionObject.ACTION.BLOCK, rowNumber,
columnNumber);
                    }else if (ruleTable.getValueAt(rowNumber,
columnNumber) == ActionObject.ACTION.BLOCK){

ruleTable.setValueAt(ActionObject.ACTION.PASS, rowNumber,
columnNumber);
                    }else{

ruleTable.setValueAt(ActionObject.ACTION.PASS, rowNumber,
columnNumber);
                    }
                }
            }

        }else if ( SwingUtilities.isRightMouseButton( evt )){

            if (columnNumber != ruleTableElements.ACTION.ordinal()){
                JPopupMenu popupMenu = new JPopupMenu();

                JMenuItem menuItem0 = new JMenuItem("Add Selected
Items");
                menuItem0.addActionListener( new
AddSelectedItems(ruleTable, rowNumber, columnNumber,
objectsTreePanel.getObjectsTree()));
                popupMenu.add(menuItem0);

                JMenu menuItem1 = new JMenu("Remove");
                FirewallObjectGroup fwObjGrp = (FirewallObjectGroup)
ruleTable.getValueAt(rowNumber, columnNumber);
                for (int i=0;i<fwObjGrp.getSize();i++){
                    JMenuItem removeItemSubMenu = new
JMenuItem(fwObjGrp.getObject(i).toString());
                    removeItemSubMenu.addActionListener(new
RemoveSelectedItem(ruleTable, rowNumber, columnNumber,
fwObjGrp.getObject(i)));
                    menuItem1.add(removeItemSubMenu);
                }
                popupMenu.add(menuItem1);

                popupMenu.show(evt.getComponent(), evt.getX(),
evt.getY());
            }
        }
}//GEN-LAST:event_ruleTableMouseClicked
```

```
    private void addReceivedItems(int row, ruleTableElements
ruleTableElement, String action) {
        //ActionObject actionObj = (ActionObject)
ruleTable.getValueAt(row, ruleTableElement.ordinal());
        if (action.equals("pass")){
            ruleTable.setValueAt(ActionObject.ACTION.PASS, row,
ruleTableElement.ordinal());
        }else{
            ruleTable.setValueAt(ActionObject.ACTION.BLOCK, row,
ruleTableElement.ordinal());
        }
    }

    private void addReceivedItems(int row, ruleTableElements
ruleTableElement, String type, int id) {
        FirewallObjectGroup firewallObjGroup = (FirewallObjectGroup)
ruleTable.getValueAt(row, ruleTableElement.ordinal());
        FirewallObject fwObj = null;


        fwObj = objectsTreePanel.searchObject(type, id);
        if (fwObj == null) return;
        firewallObjGroup.add(fwObj);
    }

    private void
loadRedirectionRulesButtonActionPerformed(java.awt.event.ActionEvent
evt) {//GEN-FIRST:event_loadRedirectionRulesButtonActionPerformed
        String XMLString =
ServerConnection.callMethod("getFirewallRules", null);


        XMLParser xmlParser = new XMLParser(XMLString);
        NodeList firewallRulesList =
xmlParser.getByTagName("firewallRule");
        int l = firewallRulesList.getLength();

        for (int i=0;i<l;i++){
            newFirewallRuleButtonActionPerformed(null);


            //action
            String action = xmlParser.getValue(firewallRulesList, i,
"action");
            addReceivedItems(i, ruleTableElements.ACTION, action);

            //source
            Node node =
xmlParser.getByTagName(firewallRulesList.item(i), "source").item(0);
            NodeList childNodes = node.getChildNodes();
            int k = childNodes.getLength();
            for (int j=0;j<k;j++){
                if (childNodes.item(j).getNodeType() ==
Node.ELEMENT_NODE){
                    String _type = childNodes.item(j).getNodeName();
                    int id = Integer.parseInt(
xmlParser.getCharacterDataFromElement( (Element) childNodes.item(j)
) );
                    addReceivedItems(i, ruleTableElements.SOURCE,
_type, id);
```

```
                }
            }

            //target
            node = xmlParser.getByTagName(firewallRulesList.item(i),
"target").item(0);
            childNodes = node.getChildNodes();
            k = childNodes.getLength();
            for (int j=0;j<k;j++){
                if (childNodes.item(j).getNodeType() ==
Node.ELEMENT_NODE){
                    String _type = childNodes.item(j).getNodeName();
                    int id = Integer.parseInt(
xmlParser.getCharacterDataFromElement( (Element) childNodes.item(j)
) );
                    addReceivedItems(i, ruleTableElements.TARGET,
_type, id);
                }
            }

            //port
            node = xmlParser.getByTagName(firewallRulesList.item(i),
"port").item(0);
            childNodes = node.getChildNodes();
            k = childNodes.getLength();
            for (int j=0;j<k;j++){
                if (childNodes.item(j).getNodeType() ==
Node.ELEMENT_NODE){
                    String _type = childNodes.item(j).getNodeName();
                    int id = Integer.parseInt(
xmlParser.getCharacterDataFromElement( (Element) childNodes.item(j)
) );
                    addReceivedItems(i, ruleTableElements.PORT,
_type, id);
                }
            }

        }
    }//GEN-LAST:event_loadRedirectionRulesButtonActionPerformed

    private void
newFirewallRuleButtonActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_newFirewallRuleButtonActionPerformed
        DefaultTableModel model = (DefaultTableModel)
ruleTable.getModel();
        model.addRow(new Object[]{ActionObject.ACTION.PASS, new
FirewallObjectGroup(), new FirewallObjectGroup(), new
FirewallObjectGroup()});
}//GEN-LAST:event_newFirewallRuleButtonActionPerformed

    private void
removeFirewallButtonActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_removeFirewallButtonActionPerformed
        int rowNumber = ruleTable.getSelectedRow();
        ( (DefaultTableModel)(ruleTable.getModel())
).removeRow(rowNumber);
}//GEN-LAST:event_removeFirewallButtonActionPerformed
```

```
    private void
submitFirewallButtonActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_submitFirewallButtonActionPerformed
         StringBuffer strXML = new StringBuffer("<?xml version='1.0'
encoding='utf-8'?>\n");
        strXML.append("<demir-fw>\n");
        strXML.append("<firewallRules>\n");
        Vector allData = ( (DefaultTableModel)(ruleTable.getModel())
).getDataVector();
        //strXML.append(allData.size()+"\">\n");
        Vector rowData = null;
        for (int i=0;i<allData.size();i++){
            strXML.append("<firewallRule>\n");
            rowData = (Vector)allData.elementAt(i);

            //action
            String action =
rowData.elementAt(ruleTableElements.ACTION.ordinal()).toString().toL
owerCase();
            strXML.append("<action>" + action + "</action>\n");

            //source
            strXML.append("<source>\n");
            FirewallObjectGroup fwObjects = (FirewallObjectGroup)
rowData.elementAt(ruleTableElements.SOURCE.ordinal());
            int s = fwObjects.getSize();
            if (s == 0){
                strXML.append("any\n");
            }else{
                for (int j=0;j<s;j++){
                    String _type =
fwObjects.getObject(j).type.toString().toLowerCase();
                    int id = fwObjects.getObject(j).id;

strXML.append("<"+_type+">"+id+"</"+_type+">\n");
                }
            }
            strXML.append("</source>\n");

            //target
            strXML.append("<target>\n");
            fwObjects = (FirewallObjectGroup)
rowData.elementAt(ruleTableElements.TARGET.ordinal());
            s = fwObjects.getSize();
            if (s == 0){
                strXML.append("any\n");
            }else{
                for (int j=0;j<s;j++){
                    String _type =
fwObjects.getObject(j).type.toString().toLowerCase();
                    int id = fwObjects.getObject(j).id;

strXML.append("<"+_type+">"+id+"</"+_type+">\n");
                }
            }
            strXML.append("</target>\n");

            //port
            strXML.append("<port>\n");
            fwObjects = (FirewallObjectGroup)
rowData.elementAt(ruleTableElements.PORT.ordinal());
            s = fwObjects.getSize();
            if (s == 0){
                strXML.append("any\n");
            }else{
                for (int j=0;j<s;j++){
                    String _type =
fwObjects.getObject(j).type.toString().toLowerCase();
                    int id = fwObjects.getObject(j).id;

strXML.append("<"+_type+">"+id+"</"+_type+">\n");
                }
            }
            strXML.append("</port>\n");


            strXML.append("</firewallRule>\n");

        }
        strXML.append("</firewallRules>\n");
        strXML.append("</demir-fw>\n");



        try{
            ArrayList<String> parameters = new ArrayList<String>();
            parameters.add(strXML.toString());

            String returnXML =
ServerConnection.callMethod("submitFirewallRules", parameters);


       }catch(Exception ex){

Logger.getLogger(DemirFWClientView.class.getName()).log(Level.SEVERE
, null, ex);
        }
}//GEN-LAST:event_submitFirewallButtonActionPerformed


    // Variables declaration
    private javax.swing.JPanel jPanel1;
    private javax.swing.JButton loadRedirectionRulesButton;
    private javax.swing.JButton newFirewallRuleButton;
    private demirfwclient.ObjectsTreePanel objectsTreePanel;
    private javax.swing.JButton reloadButton;
    private javax.swing.JButton removeFirewallButton;
    private javax.swing.JTable ruleTable;
    private javax.swing.JScrollPane ruleTableScrollPane;
    private javax.swing.JButton submitFirewallButton;
    // End of variables declaration//
}
```

B.5. RedirectionFrame.java (Port Redirection Management Module Client)

```
/*DemirFW Implementation for M.Sc. Thesis
Necati Demir, 2009*/

package demirfwclient;
import demirfwclient.FirewallObjectGroup;
import demirfwclient.ObjectsTreePanel.treeRootObjects;
import demirfwclient.ProtocolObject.PROTOCOL;
import java.awt.Point;
import java.net.URL;
import java.util.ArrayList;
import java.util.Vector;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JMenu;
import javax.swing.JMenuItem;
import javax.swing.JPopupMenu;
import javax.swing.SwingUtilities;
import javax.swing.table.DefaultTableModel;
import org.apache.xmlrpc.client.XmlRpcClient;
import org.apache.xmlrpc.client.XmlRpcClientConfigImpl;
import org.w3c.dom.NodeList;

public class RedirectionFrame extends javax.swing.JInternalFrame {

    private void addReceivedItems(int i, rdrRuleTableElements
rdrRuleTableElement, int id) {
        FirewallObjectGroup firewallObjGroup = (FirewallObjectGroup)
rdrRuleTable.getValueAt(i, rdrRuleTableElement.ordinal());
        FirewallObject fwObj = null;
        if (rdrRuleTableElement == rdrRuleTableElements.HOST){
            fwObj = objectsTreePanel.searchObject("host", id);
        }
        firewallObjGroup.add(fwObj);

    }

    private void addReceivedItems(int i, PROTOCOL proto,
rdrRuleTableElements rdrRuleTableElement, int id) {
        FirewallObjectGroup firewallObjGroup = (FirewallObjectGroup)
rdrRuleTable.getValueAt(i, rdrRuleTableElement.ordinal());
        FirewallObject fwObj = null;
        if (proto == PROTOCOL.TCP){
            fwObj = objectsTreePanel.searchObject("tcpservice", id);
        }else if(proto == PROTOCOL.UDP){
            fwObj = objectsTreePanel.searchObject("udpservice", id);
        }
        firewallObjGroup.add(fwObj);

    }

    /** Creates new form RedirectionFrame */

    enum rdrRuleTableElements {PROTOCOL, HOST, EXTERNALPORT,
INTERNALPORT}

    public RedirectionFrame() {
        initComponents();
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method
is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated
Code">//GEN-BEGIN:initComponents
    private void initComponents() {

        jPanel1 = new javax.swing.JPanel();
        objectsTreePanel = new demirfwclient.ObjectsTreePanel();
        reloadButton = new javax.swing.JButton();
        jScrollPane1 = new javax.swing.JScrollPane();
        rdrRuleTable = new javax.swing.JTable();
        newRedirectionRuleButton = new javax.swing.JButton();
        jButton2 = new javax.swing.JButton();
        jButton3 = new javax.swing.JButton();
        loadRedirectionRulesButton = new javax.swing.JButton();

        setClosable(true);
        setName("Form"); // NOI18N

        jPanel1.setName("jPanel1"); // NOI18N

        objectsTreePanel.setName("objectsTreePanel"); // NOI18N

        org.jdesktop.application.ResourceMap resourceMap =
org.jdesktop.application.Application.getInstance(demirfwclient.Demir
FWClientApp.class).getContext().getResourceMap(RedirectionFrame.clas
s);

reloadButton.setText(resourceMap.getString("reloadButton.text")); //
NOI18N
        reloadButton.setName("reloadButton"); // NOI18N
        reloadButton.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                reloadButtonActionPerformed(evt);
            }
        });
        reloadButton.addMouseListener(new
java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt)
{
                reloadButtonMouseClicked(evt);
            }
        });

        jScrollPane1.setName("jScrollPane1"); // NOI18N

        String col[] = {"Protocol","Host","External Port","Internal
Port"};
        DefaultTableModel dm = new DefaultTableModel(null, col) {
            public Class getColumnClass(int columnIndex) {
                return String.class;
            }
            public boolean isCellEditable(int row, int column) {
                return false; // Super returns true.
            }
        };
        rdrRuleTable.setModel(dm);
        rdrRuleTable.setName("rdrRuleTable"); // NOI18N
        rdrRuleTable.addMouseListener(new
java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt)
{
                rdrRuleTableMouseClicked(evt);
            }
        });
        jScrollPane1.setViewportView(rdrRuleTable);


newRedirectionRuleButton.setText(resourceMap.getString("newRedirecti
onRuleButton.text")); // NOI18N

newRedirectionRuleButton.setName("newRedirectionRuleButton"); //
NOI18N
        newRedirectionRuleButton.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                newRedirectionRuleButtonActionPerformed(evt);
            }
        });

        jButton2.setText(resourceMap.getString("jButton2.text")); //
NOI18N
        jButton2.setName("jButton2"); // NOI18N
        jButton2.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                jButton2ActionPerformed(evt);
            }
        });

        jButton3.setText(resourceMap.getString("jButton3.text")); //
NOI18N
        jButton3.setName("jButton3"); // NOI18N
        jButton3.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                jButton3ActionPerformed(evt);
            }
        });


loadRedirectionRulesButton.setText(resourceMap.getString("loadRedire
ctionRulesButton.text")); // NOI18N

loadRedirectionRulesButton.setName("loadRedirectionRulesButton"); //
NOI18N
```

```
        loadRedirectionRulesButton.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                loadRedirectionRulesButtonActionPerformed(evt);
            }
        });

        org.jdesktop.layout.GroupLayout jPanel1Layout = new
org.jdesktop.layout.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LE
ADING)
            .add(jPanel1Layout.createSequentialGroup()
                .addContainerGap()

.add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayo
ut.TRAILING, false)
                    .add(org.jdesktop.layout.GroupLayout.LEADING,
reloadButton, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .add(org.jdesktop.layout.GroupLayout.LEADING,
objectsTreePanel, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))

.addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(jScrollPane1,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 508,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)

.add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayo
ut.LEADING)
                    .add(jButton2,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 190, Short.MAX_VALUE)
                    .add(newRedirectionRuleButton,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 190, Short.MAX_VALUE)
                    .add(org.jdesktop.layout.GroupLayout.TRAILING,
jButton3, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .add(loadRedirectionRulesButton,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 190, Short.MAX_VALUE))
                .addContainerGap())
        );
        jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LE
ADING)
            .add(jPanel1Layout.createSequentialGroup()

.add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayo
ut.LEADING)
                    .add(jPanel1Layout.createSequentialGroup()
                        .addContainerGap()
                        .add(reloadButton)

.addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
.add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayo
ut.LEADING)

.add(org.jdesktop.layout.GroupLayout.TRAILING, jScrollPane1,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 500, Short.MAX_VALUE)

.add(org.jdesktop.layout.GroupLayout.TRAILING, objectsTreePanel,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)))
                    .add(jPanel1Layout.createSequentialGroup()
                        .add(68, 68, 68)
                        .add(loadRedirectionRulesButton)

.addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                        .add(newRedirectionRuleButton)

.addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                        .add(jButton2)
                        .add(74, 74, 74)
                        .add(jButton3)))
                .addContainerGap())
        );

        org.jdesktop.layout.GroupLayout layout = new
org.jdesktop.layout.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .addContainerGap()
                .add(jPanel1,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 1026,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)

.addContainerGap(org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );
        layout.setVerticalGroup(

layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .addContainerGap()
                .add(jPanel1,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addContainerGap())
        );

        pack();
    }// </editor-fold>//GEN-END:initComponents

private void reloadButtonMouseClicked(java.awt.event.MouseEvent evt)
{//GEN-FIRST:event_reloadButtonMouseClicked

}//GEN-LAST:event_reloadButtonMouseClicked
```

```
private void
newRedirectionRuleButtonActionPerformed(java.awt.event.ActionEvent
evt) {//GEN-FIRST:event_newRedirectionRuleButtonActionPerformed
DefaultTableModel model = (DefaultTableModel)
rdrRuleTable.getModel();
     model.addRow(new Object[]{ProtocolObject.PROTOCOL.TCP, new
FirewallObjectGroup(), new FirewallObjectGroup(), new
FirewallObjectGroup()});
}//GEN-LAST:event_newRedirectionRuleButtonActionPerformed

private void rdrRuleTableMouseClicked(java.awt.event.MouseEvent evt)
{//GEN-FIRST:event_rdrRuleTableMouseClicked

    Point p = evt.getPoint();
    int rowNumber = rdrRuleTable.rowAtPoint(p);
    int columnNumber = rdrRuleTable.columnAtPoint(p);
    if ( SwingUtilities.isLeftMouseButton( evt ) ){
        if (evt.getClickCount() == 2){ /*double click*/

            if (columnNumber ==
rdrRuleTableElements.PROTOCOL.ordinal()){
                if (rdrRuleTable.getValueAt(rowNumber, columnNumber)
== ProtocolObject.PROTOCOL.TCP){

rdrRuleTable.setValueAt(ProtocolObject.PROTOCOL.UDP, rowNumber,
columnNumber);
                }else if (rdrRuleTable.getValueAt(rowNumber,
columnNumber) == ProtocolObject.PROTOCOL.UDP){

rdrRuleTable.setValueAt(ProtocolObject.PROTOCOL.TCP, rowNumber,
columnNumber);
                }else{

rdrRuleTable.setValueAt(ProtocolObject.PROTOCOL.TCP, rowNumber,
columnNumber);
                }
            }
        }
    }else if(SwingUtilities.isRightMouseButton(evt)){

        if (columnNumber !=
rdrRuleTableElements.PROTOCOL.ordinal()){
                JPopupMenu popupMenu = new JPopupMenu();

                JMenuItem menuItem0 = new JMenuItem("Add Selected
Items");
                menuItem0.addActionListener( new
AddSelectedItems(rdrRuleTable, rowNumber, columnNumber,
objectsTreePanel.getObjectsTree(),true));
                popupMenu.add(menuItem0);

                JMenu menuItem1 = new JMenu("Remove");
                FirewallObjectGroup fwObjGrp = (FirewallObjectGroup)
rdrRuleTable.getValueAt(rowNumber, columnNumber);
                for (int i=0;i<fwObjGrp.getSize();i++){
                        JMenuItem removeItemSubMenu = new
JMenuItem(fwObjGrp.getObject(i).toString());
                        removeItemSubMenu.addActionListener(new
RemoveSelectedItem(rdrRuleTable, rowNumber, columnNumber,
fwObjGrp.getObject(i)));
```

```
                        menuItem1.add(removeItemSubMenu);
                }
                popupMenu.add(menuItem1);



                popupMenu.show(evt.getComponent(), evt.getX(),
evt.getY());
            }

    }


}//GEN-LAST:event_rdrRuleTableMouseClicked

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_jButton2ActionPerformed
int rowNumber = rdrRuleTable.getSelectedRow();
    ( (DefaultTableModel)(rdrRuleTable.getModel())
).removeRow(rowNumber);
}//GEN-LAST:event_jButton2ActionPerformed

private void jButton3ActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_jButton3ActionPerformed
    StringBuffer strXML = new StringBuffer("<?xml version='1.0'
encoding='utf-8'?>");
    // strXML.append("<?xml version='1.0' encoding='utf-8'?>\n");
    strXML.append("<demir-fw>\n");
    strXML.append("<redirectionRules>\n");
    Vector allData = ( (DefaultTableModel)(rdrRuleTable.getModel())
).getDataVector();
    //strXML.append(allData.size()+"\">\n");
    Vector rowData = null;
    for (int i=0;i<allData.size();i++){
        strXML.append("<redirectionRule>\n");
        rowData = (Vector)allData.elementAt(i);

        //tell me the action

strXML.append("<protocol>"+rowData.elementAt(0).toString().toLowerCa
se()+"</protocol>\n");

        //who are the sources?
        strXML.append("<hostId>");
        FirewallObjectGroup hostObjects = (FirewallObjectGroup)
rowData.elementAt(1);
        strXML.append(hostObjects.getObject(0).id);
        strXML.append("</hostId>\n");

        strXML.append("<externalPortId>");
        FirewallObjectGroup externalPortObjects =
(FirewallObjectGroup) rowData.elementAt(2);
        strXML.append(externalPortObjects.getObject(0).id);
        strXML.append("</externalPortId>\n");

        strXML.append("<internalPortId>");
        FirewallObjectGroup internalPortObjects =
(FirewallObjectGroup) rowData.elementAt(3);
        strXML.append(internalPortObjects.getObject(0).id);
        strXML.append("</internalPortId>\n");
        strXML.append("</redirectionRule>\n");

    }
    strXML.append("</redirectionRules>\n");
    strXML.append("</demir-fw>\n");

    try{
        ArrayList<String> parameters = new ArrayList<String>();
        parameters.add(strXML.toString());

        String returnXML =
ServerConnection.callMethod("submitRedirectionRules", parameters);

    }catch(Exception ex){

Logger.getLogger(DemirFWClientView.class.getName()).log(Level.SEVERE
, null, ex);
    }

}//GEN-LAST:event_jButton3ActionPerformed


private void reloadButtonActionPerformed(java.awt.event.ActionEvent
evt) {//GEN-FIRST:event_reloadButtonActionPerformed

//objectsTreePanel.setServerURL("http://10.1.1.2:81/xmlrpcserver/ind
ex.cgi");
    objectsTreePanel.loadObjectsTree(new
ObjectsTreePanel.treeRootObjects[]{
    ObjectsTreePanel.treeRootObjects.HOSTS,
    ObjectsTreePanel.treeRootObjects.TCPSERVICES,
    ObjectsTreePanel.treeRootObjects.UDPSERVICES});




}//GEN-LAST:event_reloadButtonActionPerformed

private void
loadRedirectionRulesButtonActionPerformed(java.awt.event.ActionEvent
evt) {//GEN-FIRST:event_loadRedirectionRulesButtonActionPerformed
    String XMLString =
ServerConnection.callMethod("getRedirectionRules", null);

    XMLParser xmlParser = new XMLParser(XMLString);
    NodeList redirectionRulesList =
xmlParser.getByTagName("redirectionRule");
    int l = redirectionRulesList.getLength();

    for (int i=0;i<l;i++){
        newRedirectionRuleButtonActionPerformed(null);
        String protocol = xmlParser.getValue(redirectionRulesList,
i, "protocol");

        ProtocolObject.PROTOCOL proto;
        if (protocol.equals("tcp")){
            rdrRuleTable.setValueAt(ProtocolObject.PROTOCOL.TCP, i,
rdrRuleTableElements.PROTOCOL.ordinal());
            proto = ProtocolObject.PROTOCOL.TCP;
        }else{
            rdrRuleTable.setValueAt(ProtocolObject.PROTOCOL.UDP, i, 
rdrRuleTableElements.PROTOCOL.ordinal());
            proto = ProtocolObject.PROTOCOL.UDP;
        }

        addReceivedItems(i, rdrRuleTableElements.HOST, 
Integer.parseInt( xmlParser.getValue(redirectionRulesList, i, 
"hostId") ));
        addReceivedItems(i, proto, 
rdrRuleTableElements.EXTERNALPORT, Integer.parseInt( 
xmlParser.getValue(redirectionRulesList, i, "externalPortId") ));
        addReceivedItems(i, proto, 
rdrRuleTableElements.INTERNALPORT, Integer.parseInt( 
xmlParser.getValue(redirectionRulesList, i, "internalPortId") ));

    }

}//GEN-LAST:event_loadRedirectionRulesButtonActionPerformed


    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JButton jButton2;
    private javax.swing.JButton jButton3;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JButton loadRedirectionRulesButton;
    private javax.swing.JButton newRedirectionRuleButton;
    private demirfwclient.ObjectsTreePanel objectsTreePanel;
    private javax.swing.JTable rdrRuleTable;
    private javax.swing.JButton reloadButton;
    // End of variables declaration//GEN-END:variables

}
```

APPENDIX C
DEMIRFW CLIENT AS A JAVA APPLET

To use DemirFW Java client as a java client, main class should be converted to JApplet class. Main class is MainFrame.java.

MainFrame.java

```
/*DemirFW Implementation for M.Sc. Thesis
Necati Demir, 2009*/

package demirfwclient;
import java.awt.Toolkit;

public class MainFrame extends javax.swing.JFrame {

    private void customInit() {
        Toolkit tk = Toolkit.getDefaultToolkit();
        int xSize = ((int) tk.getScreenSize().getWidth());
        int ySize = ((int) tk.getScreenSize().getHeight());
        setSize(xSize,ySize);
    }

    /** Creates new form MainFrame */
    public MainFrame() {
        initComponents();
        customInit();
    }

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated
Code">//GEN-BEGIN:initComponents
    private void initComponents() {

        jDesktopPane1 = new javax.swing.JDesktopPane();
        jMenuBar1 = new javax.swing.JMenuBar();
        fileMenu = new javax.swing.JMenu();
        exitMenuItem = new javax.swing.JMenuItem();
        windowsMenu = new javax.swing.JMenu();
        networkConfigurationMenuItem = new javax.swing.JMenuItem();
        objectsMenuItem = new javax.swing.JMenuItem();
        firewallMenuItem = new javax.swing.JMenuItem();
        jMenuItem2 = new javax.swing.JMenuItem();
        redirectionMenuItem = new javax.swing.JMenuItem();
        jMenuItem1 = new javax.swing.JMenuItem();


setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setName("Form"); // NOI18N

        jDesktopPane1.setName("jDesktopPane1"); // NOI18N

        jMenuBar1.setName("jMenuBar1"); // NOI18N
        org.jdesktop.application.ResourceMap resourceMap =
org.jdesktop.application.Application.getInstance(demirfwclient.Demir
FWClientApp.class).getContext().getResourceMap(MainFrame.class);
        fileMenu.setText(resourceMap.getString("fileMenu.text")); //
NOI18N
        fileMenu.setName("fileMenu"); // NOI18N


exitMenuItem.setText(resourceMap.getString("exitMenuItem.text")); //
NOI18N
        exitMenuItem.setName("exitMenuItem"); // NOI18N
        fileMenu.add(exitMenuItem);

        jMenuBar1.add(fileMenu);


windowsMenu.setText(resourceMap.getString("windowsMenu.text")); //
NOI18N
        windowsMenu.setName("windowsMenu"); // NOI18N


networkConfigurationMenuItem.setText(resourceMap.getString("networkC
onfigurationMenuItem.text")); // NOI18N

networkConfigurationMenuItem.setName("networkConfigurationMenuItem")
; // NOI18N
        networkConfigurationMenuItem.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                networkConfigurationMenuItemActionPerformed(evt);
            }
        });
        windowsMenu.add(networkConfigurationMenuItem);


objectsMenuItem.setText(resourceMap.getString("objectsMenuItem.text"
)); // NOI18N
        objectsMenuItem.setName("objectsMenuItem"); // NOI18N
        objectsMenuItem.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                objectsMenuItemActionPerformed(evt);
            }
        });
        objectsMenuItem.addMouseListener(new
java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt)
{
                objectsMenuItemMouseClicked(evt);
            }
        });
        windowsMenu.add(objectsMenuItem);


firewallMenuItem.setText(resourceMap.getString("firewallMenuItem.tex
t")); // NOI18N
        firewallMenuItem.setName("firewallMenuItem"); // NOI18N
        firewallMenuItem.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                firewallMenuItemActionPerformed(evt);
            }
        });
        firewallMenuItem.addMouseListener(new
java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt)
{
                firewallMenuItemMouseClicked(evt);
            }
        });
        windowsMenu.add(firewallMenuItem);


jMenuItem2.setText(resourceMap.getString("jMenuItem2.text")); //
NOI18N
        jMenuItem2.setName("jMenuItem2"); // NOI18N
        jMenuItem2.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                jMenuItem2ActionPerformed(evt);
            }
        });
        windowsMenu.add(jMenuItem2);


redirectionMenuItem.setText(resourceMap.getString("redirectionMenuIt
em.text")); // NOI18N
        redirectionMenuItem.setName("redirectionMenuItem"); //
NOI18N
        redirectionMenuItem.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                redirectionMenuItemActionPerformed(evt);
            }
        });
        windowsMenu.add(redirectionMenuItem);


jMenuItem1.setText(resourceMap.getString("jMenuItem1.text")); //
NOI18N
        jMenuItem1.setName("jMenuItem1"); // NOI18N
        jMenuItem1.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                jMenuItem1ActionPerformed(evt);
            }
        });
        windowsMenu.add(jMenuItem1);

        jMenuBar1.add(windowsMenu);

        setJMenuBar(jMenuBar1);
        org.jdesktop.layout.GroupLayout layout = new
org.jdesktop.layout.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(jDesktopPane1,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 540, Short.MAX_VALUE)
        );
        layout.setVerticalGroup(

layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(jDesktopPane1,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 351, Short.MAX_VALUE)
        );

        pack();
    }// </editor-fold>//GEN-END:initComponents

private void firewallMenuItemMouseClicked(java.awt.event.MouseEvent
evt) {//GEN-FIRST:event_firewallMenuItemMouseClicked
}//GEN-LAST:event_firewallMenuItemMouseClicked

private void
firewallMenuItemActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_firewallMenuItemActionPerformed

    FirewallFrame fwFr = new FirewallFrame();
    jDesktopPane1.add( fwFr );
    fwFr.setVisible(true);
    jDesktopPane1.updateUI();
}//GEN-LAST:event_firewallMenuItemActionPerformed

private void objectsMenuItemMouseClicked(java.awt.event.MouseEvent
evt) {//GEN-FIRST:event_objectsMenuItemMouseClicked

}//GEN-LAST:event_objectsMenuItemMouseClicked

private void
objectsMenuItemActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_objectsMenuItemActionPerformed
    ObjectsFrame objFr = new ObjectsFrame();
    jDesktopPane1.add( objFr );
    objFr.setVisible(true);
    jDesktopPane1.updateUI();
}//GEN-LAST:event_objectsMenuItemActionPerformed

private void
redirectionMenuItemActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_redirectionMenuItemActionPerformed
    RedirectionFrame rdrFr = new RedirectionFrame();
    jDesktopPane1.add( rdrFr );
    rdrFr.setVisible(true);
    jDesktopPane1.updateUI();
}//GEN-LAST:event_redirectionMenuItemActionPerformed

private void
networkConfigurationMenuItemActionPerformed(java.awt.event.ActionEve
nt evt) {//GEN-
FIRST:event_networkConfigurationMenuItemActionPerformed
```

```
    NetworkConfigurationFrame ncFr = new
NetworkConfigurationFrame();
    jDesktopPane1.add( ncFr );
    ncFr.setVisible(true);
    jDesktopPane1.updateUI();
}//GEN-LAST:event_networkConfigurationMenuItemActionPerformed

private void jMenuItem1ActionPerformed(java.awt.event.ActionEvent
evt) {//GEN-FIRST:event_jMenuItem1ActionPerformed
    DHCPDFrame dhcpdFr = new DHCPDFrame();
    jDesktopPane1.add( dhcpdFr );
    dhcpdFr.setVisible(true);
    jDesktopPane1.updateUI();
}//GEN-LAST:event_jMenuItem1ActionPerformed

private void jMenuItem2ActionPerformed(java.awt.event.ActionEvent
evt) {//GEN-FIRST:event_jMenuItem2ActionPerformed

    FirewallFrame2 fwFr = new FirewallFrame2();
    jDesktopPane1.add( fwFr );
    fwFr.setVisible(true);
    jDesktopPane1.updateUI();
}//GEN-LAST:event_jMenuItem2ActionPerformed

    /**
    * @param args the command line arguments
    */
    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new MainFrame().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JMenuItem exitMenuItem;
    private javax.swing.JMenu fileMenu;
    private javax.swing.JMenuItem firewallMenuItem;
    private javax.swing.JDesktopPane jDesktopPane1;
    private javax.swing.JMenuBar jMenuBar1;
    private javax.swing.JMenuItem jMenuItem1;
    private javax.swing.JMenuItem jMenuItem2;
    private javax.swing.JMenuItem networkConfigurationMenuItem;
    private javax.swing.JMenuItem objectsMenuItem;
    private javax.swing.JMenuItem redirectionMenuItem;
    private javax.swing.JMenu windowsMenu;
    // End of variables declaration//GEN-END:variables

}
```

DemirFWClientFrame2Applet.patch is the patch whis is used to convert DemirFW Java Application to Java Applet.

DemirFWClientFrame2Applet.patch

```
--- MainFrame.java      (revision 11)
+++ MainFrame.java      (working copy)
@@ -12,7 +12,7 @@

-public class MainFrame extends javax.swing.JFrame {
+public class MainFrame extends javax.swing.JApplet {

     private void customInit() {
         Toolkit tk = Toolkit.getDefaultToolkit();
@@ -22,7 +22,8 @@
     }

     /** Creates new form MainFrame */
-    public MainFrame() {
+    //public MainFrame() {
+    public void init() {
         initComponents();
         customInit();
     }
@@ -48,7 +49,7 @@
         redirectionMenuItem = new javax.swing.JMenuItem();
         jMenuItem1 = new javax.swing.JMenuItem();

-
setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
+
//setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE
);
         setName("Form"); // NOI18N

         jDesktopPane1.setName("jDesktopPane1"); // NOI18N
@@ -147,7 +148,7 @@
             .add(jDesktopPane1,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 351, Short.MAX_VALUE)
         );

-        pack();
+        //pack();
     }// </editor-fold>//GEN-END:initComponents

 private void firewallMenuItemMouseClicked(java.awt.event.MouseEvent
evt) {//GEN-FIRST:event_firewallMenuItemMouseClicked
@@ -204,13 +205,13 @@
     /**
     * @param args the command line arguments
     */
-    public static void main(String args[]) {
+    /*public static void main(String args[]) {
         java.awt.EventQueue.invokeLater(new Runnable() {
             public void run() {
                 new MainFrame().setVisible(true);
             }
         });
-    }
+    }*/

     private javax.swing.JMenuItem exitMenuItem;
```