TAMPEREEN AMMATTIKORKEAKOULU
Tietotekniikan koulutusohjelma
Tietoliikennetekniikka

Tutkintotyö

Jari Haapa-aho

KNOPPIX BASED FIREWALL/ROUTER

Project work
Jari Haapa-aho
Beng (Hons) Communication Systems Engineering
University of Portsmouth
Supervisor: Chi Nguyen
2006

ABSTRACT

The subject of this project is Knoppix based firewall and router. Knoppix is Linux live CD distribution based on Debian and it was developed by Klaus Knopper.

A firewall's basic task is to prevent forbidden communication between two networks and it is usually situated between internet and internal network. Knoppix live CD makes it possible to do packet filtering in a cost-effective way, because it doesn't need the latest pc technology and it supports very well pc hardware.

This project's aim was to develop firewall program for Knoppix live CD. Firewall is possible to be used as a gateway firewall or a host only firewall. Other essential functions are its easy-of-use and possibility of sharing internet using only one public IP address.

While building a home network or small corporate network it is needed to pay attention for network security. Firewalls are more than needed these days and using those with good configuration, they will increase security of data communication and also improve operational reliability of the network. This work was made by using Knoppix live CD version 4.0 which includes already installed iptables program. In addition I used for programming bash script language and Tk toolkit.

This work's results meet the requirements of the project. Final program is easy-to-use and that makes it possible for the end user to share the internet connection in a safe way. The end users don't need to have much experience about Linux systems or network security.

At present the program makes packet filtering based on its current state. In future it could need a proxy server and there is for example Squid available for this purpose. Also the routing is not fully supported yet and the current version shares the internet using masquerading.

Keywords: knoppix, firewall, iptables

TIIVISTELMÄ

Tämän työn englanninkielinen nimi on Knoppix based firewall and router. Työn aiheena on palomuuri ohjelman toteuttaminen Knoppix live cd:lle, jonka on alun perin kehittänyt Klaus Knopper Linux Debianin pohjalta.

Palomuurin tehtävä on estää luvaton tietoliikenne ja sallia vain käyttäjän tarvitsemat palvelut kahden eri verkon välillä. Useasti palomuuri sijoitetaankin internetin ja kotiverkon välille. Knoppix live cd luo edullisen toimintaympäristön toteuttaa palomuuri, koska se ei vaadi käytettävältä koneelta suuria tehoja ja se tukee erittäin hyvin olemassa olevaa pc laitteistoa.

Työn tarkoituksena on ollut kehittää Linuxiin pohjautuva palomuuriohjelma joka sisällytetään Knoppix live cd:lle. Palomuuri toiminnon lisäksi ohjelman keskeisiä tavoitteita olivat sen helppokäyttöisyys sekä mahdollisuus jakaa internet yhteys useammalle koneelle käyttäen vain yhtä julkista ip osoitetta.

Pystytettäessä kotiverkkoa tai pientä yritysverkkoa on kiinnitettävä tietoturvaan erityistä huomiota. Palomuurit ovatkin erittäin tarpeen nykypäivänä, koska ne edistävät tietoliikenneyhteyden turvallista käyttöä sekä parantavat verkon toimintavarmuutta.
Tässä työssä palomuuriohjelma toteutettiin käyttäen Knoppix live cd 4.0 versiota, joka sisälsi valmiiksi asennettuna iptables komentorivityökalun. Lisäksi työssä käytettiin bash skriptejä ja graafisen käyttöliittymän toteutukseen Tk toolkit ohjelmointikieltä.

Tässä työssä saavutettiin asetetut tavoitteet. Ohjelma on helppokäyttöinen ja sen ansiosta vähemmän tietokoneita sekä tietoturvaa tuntevan henkilön on tarvittaessa mahdollista jakaa internet yhteys turvallisesti. Nykyinen palomuuriohjelma toteuttaa yhteyden tilaan pohjautuvan pakettien suodatuksen. Tulevaisuudessa ohjelmaan voisi liittää välityspalvelimen (proxy) lisäämään entisestään tietoturvaa. Lisäksi reititystoimintojen kehittäminen erilaisia tarpeita vastaaviksi parantaisi ohjelman käyttöastetta entisestään, koska nykyinen versio tukee ainoastaan yhteyden jakamista yhteen verkkokortin porttiin käyttäen masqueradingia.

Hakusanat: knoppix, palomuuri, iptables

TABLE OF CONTENTS

ABSTRACT
TIIVISTELMÄ
TABLE OF CONTENTS 4
1 PROJECT BACKGROUND 6
2 TECHNICAL BACKGROUND 6
2.1 Network models 6
2.1.1 OSI 7 layer model 6
2.1.2 The TCP/IP model 8
2.2 Protocols 8
2.2.1 Internet protocol 8
2.2.2 Transmission Control Protocol 9
2.2.3 User Datagram Protocol 11
2.2.3 Internet Control Message Protocol 12
2.3 Service ports 13
2.4 proc file system 14
2.5 Packet filtering framework 15
2.5.1 netfilter 15
2.5.2 iptables tool 15
2.6 Loading extra modules 20
2.7 Dynamic host configuration protocol 21
2.8 Network analysis tools 21
2.8.1 Monitoring IP traffic 21
2.8.2 Nmap port scanner 22
2.8.3 Nessus vulnerability scanner 23
2.9 Tcl/Tk 23
2.10 Remastering Knoppix CD 26
3 KEY PROJECT CONCEPTS 29
3.1 Firewall 29
3.1.1 What is a firewall? 29
3.1.2 Firewall types 29
3.2 Sharing the internet connection 31
3.2.1 Masquerading 31
3.2.2 Network address translation 31
3.3 Remastering Knoppix 33
4 PRIMARY DEVELOPMENT TOOLS 34
4.1 iptables 34
4.2 Ethereal 34
4.3 Tcl/Tk 35
4.4 Bash 35
5 PROJECT BUILDING PROCESS 35
5.1 Planning firewall 36
5.2 Implementing firewall 37
5.3 Debugging firewall 38
5.4 Remastering Knoppix CD 38

6 DISCUSSION 39
7 CONCLUSION 40
BIBLIOGRAPHY 41
APPENDIX A 43
Screenshot from Knoppix desktop 43

1 PROJECT BACKGROUND

Knoppix is a bootable Linux CD distribution and it got started by Klaus Knopper. Knoppix has large hardware support which has made it very popular live-CD distribution. Knoppix includes a lot of different kind of software like development tools, text editors, graphic programs, internet software, multimedia, software etc. There are also a lot of services to run, such as SSH-server.

Nowadays firewalls are more than needed when you are connected to the internet. There are a lot of threads and if you haven't got a good protection, someone may for example use your computer for illegal use, or steal your personal data.

This project is based on Knoppix live-CD. Firewall is done by using netfilter and iptables tool. Linux kernel offers advantages for routing. These tools are already included in Knoppix Linux live-CD distribution. This project's goal was to build a gateway firewall and make graphical user interface tool to end-user. This project offers great knowledge about Linux, TCP/IP, firewalls, routing, scripting and how to build own Knoppix live-CD.

2 TECHNICAL BACKGROUND

2.1 Network models

Network models define a set of network layers and how they interact. There are several different network models and the most important models are the TCP/IP model and the OSI 7 model.

2.1.1 OSI 7 layer model

The Open System Interconnection 7 (OSI 7) model is defined by International Standards Organisation. OSI 7 model defines seven layers that describe how applications communicate with each other through the network. Each layer offers services to the lower layer and otherwise it will use services which are served from the upper layer. The model is generic and it can be used to all network types. (Mann & Krell, 2002, p. 4)

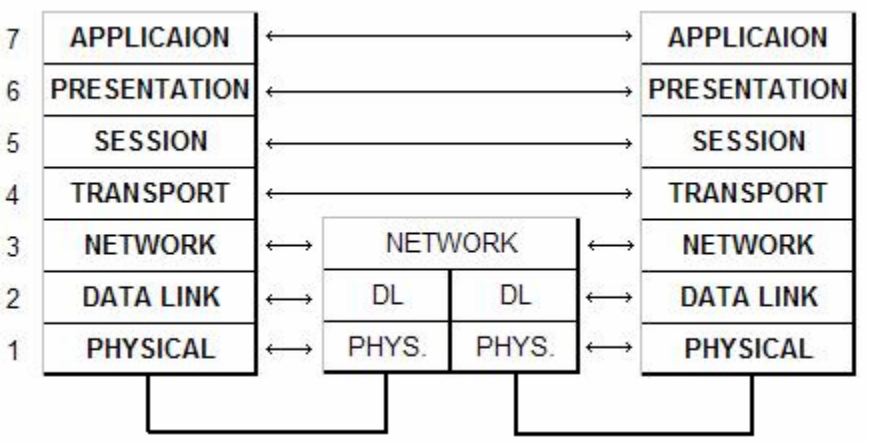

Figure 1. OSI 7 model.

Usually layer diagrams are drawn with Layer 1 at the bottom and Layer 7 at the top, as you can see from figure 1. Layer 1 is the physical layer and defines the physical characteristic of the network. This layer is used to send and receive data, example devices could be network interface card (NIC) on your personal computer (PC) or router interfaces. Layer 2 is the data link layer. This layer defines the methods used to transmit and receive data on the network, and it handles data frames between the network and physical layers. It is responsible for error-free transfer of frames to other computer via the physical layer. Layer 3 is the network layer. It translates logical network address and name to the physical address. It is responsible for addressing, determining routes for sending, and managing network problems. There are protocols such as IP, ICMP, RIP and IPX. Layer 4 is the transport layer. This layer offers reliable end-to-end connection service and it does flow control and error-handling. It ensures complete data transfer. There are protocols such as TCP and UDP. Layer 5 is the session layer. This layer establishes, manages and terminates connections between applications. The session layer sets up, coordinates, and terminates conversations, exchanges, and dialogues between the applications at each end. It deals with session and connection coordination. Layer 6 is the presentation layer. The presentation layer works to transform data into the form that the application layer can accept and otherwise application data will be translated for the network format. Protocol conversions, encryption or decryption, data compressing and graphics expansion all takes place here. Layer 7 is the application layer. This layer supports application and end-user processes. Communication partners are identified, quality of service is identified, user authentication and privacy are considered, and any constraints on data syntax are identified. Everything at this layer is

application-specific. This layer provides application services for file transfers, e-mail, and other network software services. Telnet and FTP are applications that exist entirely in the application level. (Mann & Krell, 2002, p. 5)

2.1.2 The TCP/IP model

The TCP/IP model has got its name from two main protocols, transmission control protocol (TCP) and internet protocol (IP). The TCP/IP model doesn't exactly match the OSI model. There are fewer levels than the seven layers of the OSI model. There are defined 4 layers: application layer, transport layer, internet layer and network layer. Figure 2 depicts the relationship between the OSI 7 model and the TCP/IP model. (Mann & Krell, 2002, pp. 6-7)

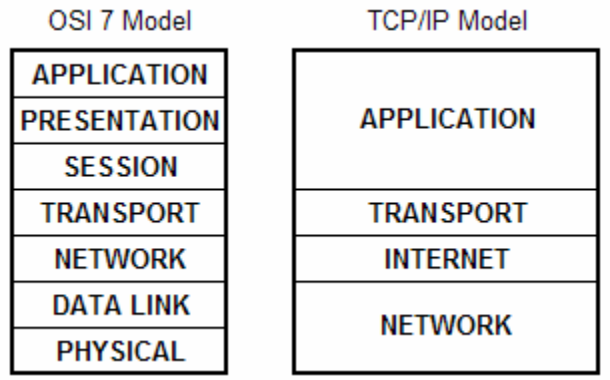

Figure 2. OSI 7 and TCP/IP models.

2.2 Protocols

2.2.1 Internet protocol

Internet Protocol (IP) is a data-oriented protocol used by source and destination hosts for communicating data across a packet-switched internet. Data in an IP network are sent in blocks referred to packets or data grams. Internet protocol provides an unreliable datagram service (also called best effort). IP is the common element in today's public internet. The current and most popular network layer protocol in use today is IPv4. IPv6 is the proposed successor to IPv4. IPv4

uses 32 bits for addressing. IPv6 uses 128 bits for the source and destination addresses. IPv4 header format is shown in figure 3.

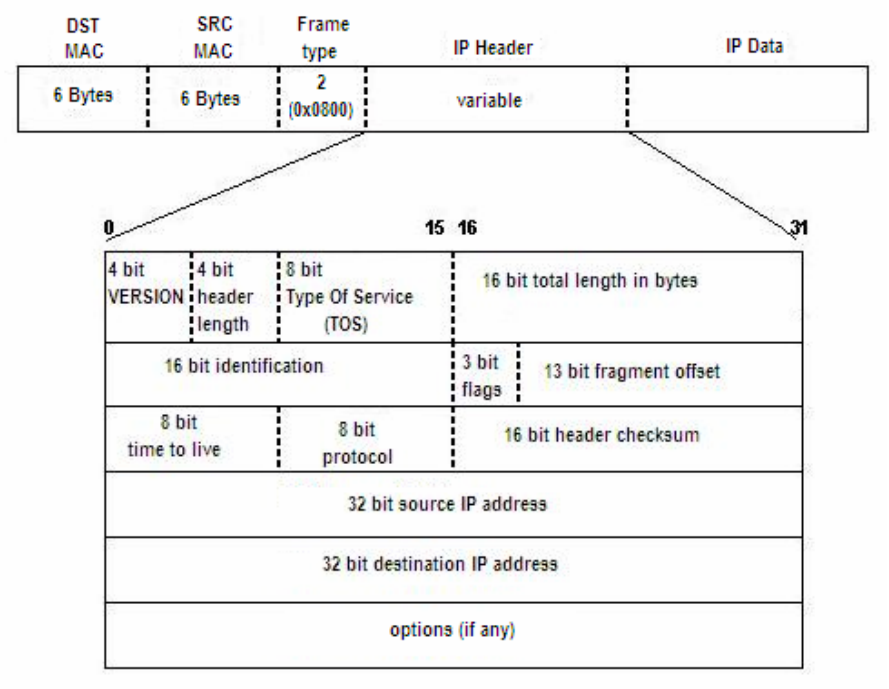

Figure 3. Ethernet frame and IP data gram.

2.2.2 Transmission Control Protocol

Transmission Control Protocol (TCP, IP protocol number 6) is a stateful and connection-oriented protocol. Stateful means that the protocol maintains information about what it has done and what it is going to do next. Connection-oriented means that connection points are attached together. TCP guarantees reliable and in-order delivery of data from sender to receiver. It will detect communication failures and after that it provides error messaging. (Mann & Krell, 2002, p. 241)

TCP connection is always initiated with the 3-way handshake. The client machine which wants to create a new connection will send the first packet, where the SYN-bit is set (TCP-flag). Other machine, which receives this packet, will send back the SYN-ACK-segment, if it allows the connection. Otherwise it will send the SYN-RST-segment to deny connection. When client receives SYN-ACK-segment it responses to this by sending the ACK-segment back to server. After this, the connection is established and data can be sent. A four-way handshake is used for

disconnecting. See the figure 4 where is shown finite state machine for TCP. (Mann & Krell, 2002, pp. 246-247)

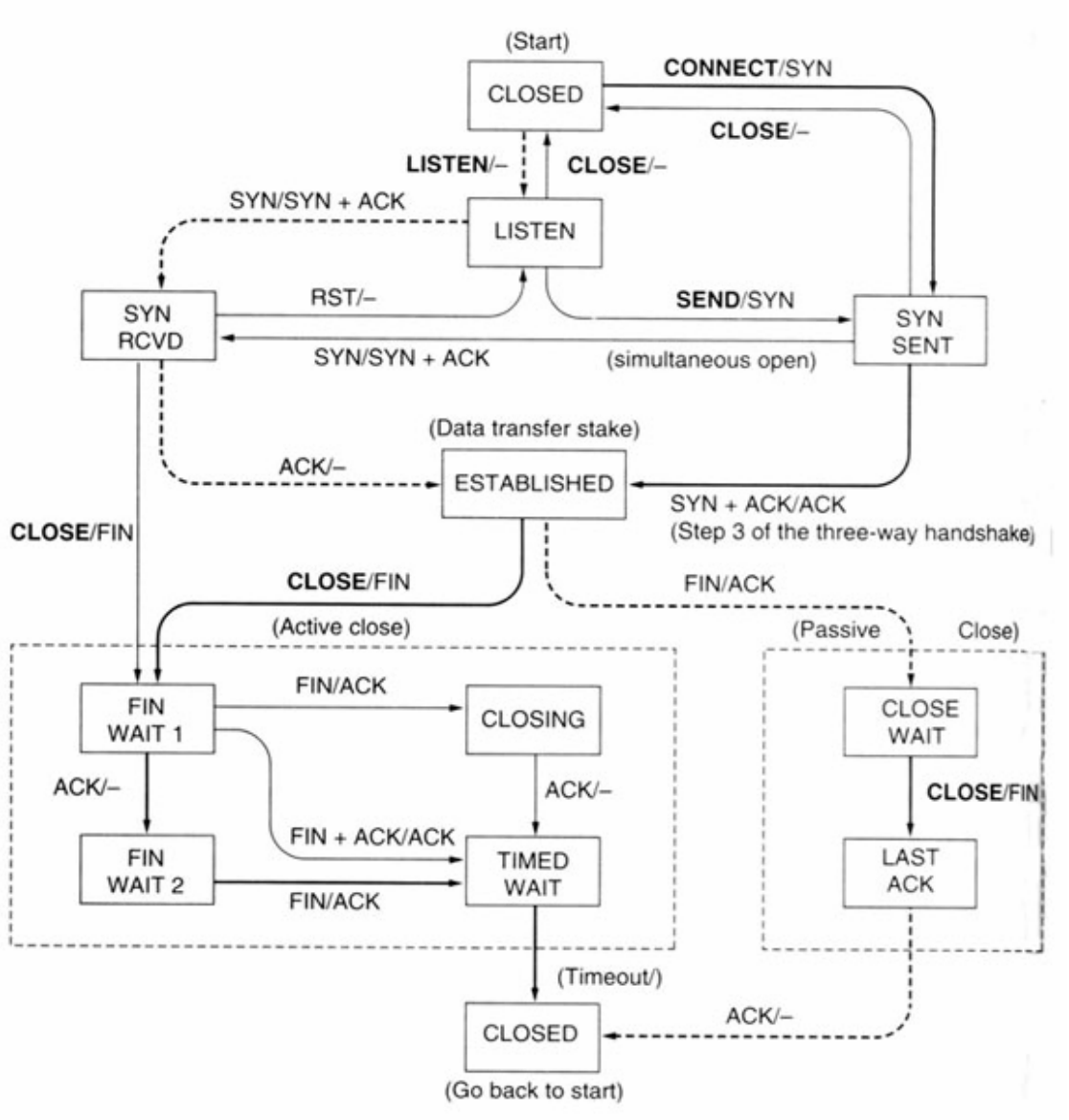

Figure 4. TCP finite state machine. (El-Saddik, n.d.)

The TCP header consists of identification and control information. The TCP header is shown in figure 5. Source port and destination port make sure that the TCP segment gets to the right application. Sequence number tells where the segment fits in the overall data stream. Acknowledgement number is the sequence number of the next segment expected by the sender. This field is only used if the ACK-bit is set (equals to 1). The TCP header length field tells the

header length. Reserved field is not used. URG, ACK, PSH, RST, SYN, and FIN are TCP bit flags. These flags are used to control connection. The URG flag (the urgent flag) indicates that urgent pointer field contains data. The ACK flag (the acknowledgement flag) indicates that a valid acknowledgement present in the acknowledgement field. The PSH flag (the push flag) tells to the recipient that gets this data to the application as soon as possible. The RST flag (the reset flag) indicates that connection needs to be terminated. The SYN flag (the synchronize flag) indicates that sequence numbers need to be synchronized. The FIN flag (the finish flag) indicates that the sender has completed sending and is now ready to terminate connection. Window size field contains information about how much data the receiver is willing to accept. Checksum field contains the checksum for the TCP segment. This sum contains header and data. Urgent pointer field is used when URG flag is set. It tells the last sequence number for the last chunk of urgent data. Options field can contain one or more options. Example one option might be timestamp. Data field size is optional. (Mann & Krell, 2002, pp. 242-243)

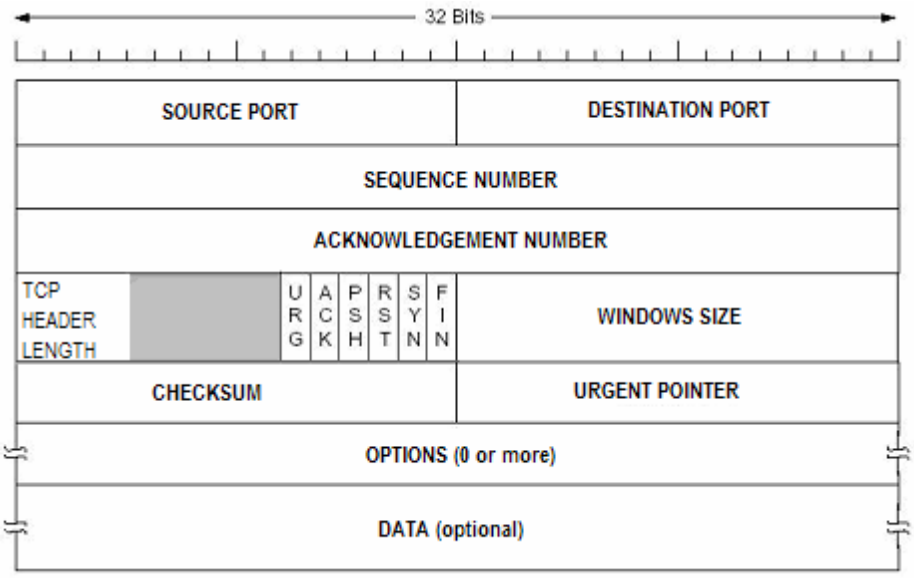

Figure 5. TCP header. (Mann & Krell, 2002, p. 242)

2.2.3 User Datagram Protocol

User Datagram Protocol (UDP, IP protocol number 17) is a connectionless datagram protocol. It doesn’t guarantee the delivery and it is possible that packets arrive in out of order. That makes it

unreliable. If an application requires these characteristics, it must provide them itself, or use TCP. The advantage of UDP is that it reduces processing and consumption of space. UDP is typically used for applications such as streaming media (audio and video) where on-time arrival is more important than reliability. It is also used for simple query/response applications like DNS lookups. (Mann & Krell, 2002, p. 238)

Figure 6 shows the UDP header. There are source and destination port fields. Source port tells the port number from sending process and destination port number is for receiving process. The length field consists of sum of UDP header and application data. UDP checksum field is for error detection. Data field is for application data and it is encapsulated by UDP. (Mann & Krell, 2002, p. 239)

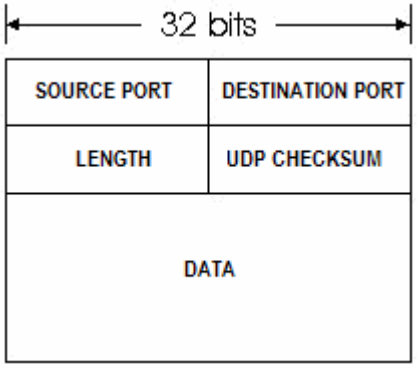

Figure 6. The UDP Header. (Mann & Krell, 2002, p. 240)

2.2.3 Internet Control Message Protocol

Internet Control Message Protocol (ICMP) is the support protocol for IP and both of the protocols live at the same level of the TCP/IP. ICMP messages are used for an error reporting between host to host or host to gateway. The IP protocol doesn't support perfect error handling and ICMP messages are made to solve these kinds of problems. ICMP messages are complicated and difference between two messages might be hard to understand.

All of the ICMP headers contains type, code and checksum field. Type and code dependent data field is not always used. The ICMP header is shown in picture 7. There are three kinds of ICMP

messages: error, request, and replies. The type field specifies what kind of message it is. The code field specifies more information. For example, if the packet type is destination unreachable, there are several possible code values, such as network unreachable and host unreachable. The checksum is a checksum for the whole packet. Data field is used for example, when sending echo request or echo reply packet. Otherwise it is just empty. (Mann & Krell, 2002, pp. 104-105)

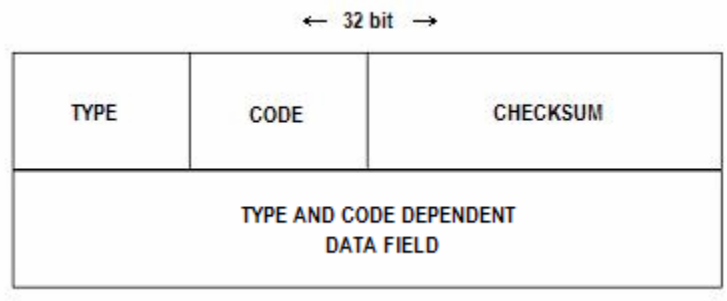

Figure 7. The ICMP header. (Mann & Krell, 2002, p. 105)

2.3 Service ports

Both protocols, TCP and UDP, are used to carry data. When application sends data to another over network, transport layer will carry out that data gets to the right application or process. This is done by using port numbers. Transport layer will write a source and destination port numbers in the transport layer header. These ports are the virtual and they are usually called service ports. Port numbers for server applications are generally associated with a well known port numbers. These well known numbers are officially allocated by the Internet Assigned Numbers Authority (IANA). Port numbers are divided to the three categories as it is described in table 1. Nowadays developers often choose the port numbers themselves, and application software might also allow user to specify port number to use. Client application generally uses a random port number assigned by the operating system. (Mann & Krell, 2002, pp. 267-269)

The Linux operating system associates port numbers with service names in the /etc/services file. Notice, that service can use the same port number for both UDP and TCP. Example dns uses both of the protocols, udp and tcp, and its port number is 53. Other well known port numbers are http (tcp, 80), https (tcp, 443), ftp (tcp, 21), ssh (tcp, 22) and telnet (tcp, 23). More information about

service names with port numbers is shown in IANA website (http://www.iana.org/assignments/port-numbers , 2.1.2006). (Mann & Krell, 2002, p. 269)

Table 1. IANN port numbers. (Mann & Krell, 2002, p. 238)

PORT NUMBERS	NAME	DESCRIPTION
0 to 1023	well known, privileged	Port numbers are assigned to specific services. Need a root privileges to bind these ports.
1024 to 49151	registered ports	Port numbers in that range can be registered to a particular service. Use of these ports may be duration of system up-time or dynamic.
49152 to 65535	dynamic or private ports	Port numbers in this range are generally allocated dynamically for use during the lifespan of a process.

2.4 proc file system

proc file system is used to obtain information about the system and to change kernel parameters at the runtime. You can enable these simple changes to the Linux kernel via proc pseudo files. Some of the files are read-only. proc is not an actual directory on your disk and that's why it doesn't take space from your hard drive. Kernel generates this pseudo file system by itself. There are several network related protections. By configuring kernel flags you can increase your network security. Many configurable proc entries have either a 0 or a 1 value. For example the /proc/sys/net/ipv4/tcp_syncookies can only be turned on or off. More information about configurable kernel flags is available in /usr/share/sitar/proc.txt (Knoppix Linux).

2.5 Packet filtering framework

2.5.1 netfilter

netfilter is the system compiled into the kernel which provides access into the IP stack. There is loadable modules (e.g. iptables and ipchains), which are used to perform operations for packets. netfilter is there all the time, as long it is compiled to the kernel and user makes choices, which modules to load and use. (“The netfilter.org project”, n.d.)

2.5.2 iptables tool

iptables is command line tool and it is used to set up the rules to the modules.

General syntax for iptables command is

iptables [-t table] command [match] [target/jump]

First option is for table selection. If you don’t specify table, the rules will be automatically implement to the filter table. Command tells to the program what it should do, for example to insert a new rule or delete rule. Match tells to the kernel what kind of packet we are looking for. Match could be for example source or destination IP address, network interface, protocol etc. There are a lot of possibilities. Finally we have a target or jump. If all the criteria match to the packet, we will tell to the kernel what to do next. Target could for example drop the packet or allow it. It is also possible to tell the kernel to send the packet to another chain, which we have created. **(**Eychenne, 2002)

Command

iptables command makes possible to insert, delete or modify rule in the specific table. It could also be used to view all the rules or just rules from the specific chain, set default policies to the specific chain, flush all rules etc. All the commands are shown in table 2. More information about iptables and its commands are available in the Linux man-pages. **(**Eychenne, 2002)

Table 2. Commands. (Eychenne, 2002)

command	Explanation
-A, --append	Append rule to the end of the selected chain
-D, --delete	Delete rule from the specific chain
-I, --insert	Insert rule in the selected chain as the given rule number
-R, --replace	Replace a rule in the selected chain as the given rule number
-L, --list	List all rules in the selected chain (filter is default).
-F, --flush	Flush the selected chain (all the chains in the table if none is given).
-Z, --zero	Zero the packet and byte counters in all chains
-N, --new-chain	Create a new user defined chain.
-X, --delete-chain	Delete the user defined chain. There must be no references to the chain.
-P, --policy	Set the default policy for the chain to the given target.
-E, --rename-chain	Rename the user specified chain to the user supplied name.
-h, Help	Give a description of the command.

Tables

The -t option specifies which table to use. There are three tables defined: nat, filter and mangle. Each table includes predefined chains for different purposes. Tables and their chains are shown in the figure 8. **(**Eychenne, 2002)

nat table

The nat table should only be used for network address translation on different packets. Only the first packet in a stream will hit this table and all the rest of the packets will automatically have the same action taken on them as the first packet. This is the main reason why this table shouldn't use any filtering. In the nat table it is possible to use targets as DNAT, SNAT, MASQUERADE and REDIRECT.

The DNAT target means that it will change the destination address of packet. Address translation occurs before routing and that's why it is used for redirecting traffic to the other host or network,

for example demilitarized zone (DMZ). The SNAT is used for changing the source address of packets. It is mainly used to hide your local networks or DMZ. The MASQUERADE target is used in exactly the same way as SNAT. The difference between these two mechanisms is that masquerade will check each time automatically which IP address to use when it gets hit by a packet. The MASQUERADE target is planned to use network, when you get your IP address dynamically from DHCP server. If you are using nat, then the maximum user capacity depends on how many public IP addresses you got.

In the nat table there are three defined chains: PREROUTING, POSTROUTING and OUTPUT. The PREROUTING chain is used for modifying packets first when they get in to the firewall. The POSTROUTING chain modifies packets just before packets are leaving from the firewall, and the latest chain is the OUTPUT which is used for modifying locally generated packets. (Mann & Krell, 2002, pp. 729-730)

filter table

The filter table is especially made for filtering packets. There are a lot of possibilities to make different rules to match to the packet. If packet will match to one of the rules, then we could for example DROP, LOG, ACCEPT or REJECT that packet.

The filter table consists of three defined chains: INPUT, OUTPUT and FORWARD. If packet is destined to the local host, it will come to the INPUT chain after the first routing decision. When packet leaves from local host, it will traverse over the OUTPUT chain after routing decision. The FORWARD chain is for packets which are not generated locally and are not destined to the local host. (Mann & Krell, 2002, pp. 728-729)

mangle table

The mangle table is used for modifying IP header. In this table it is possible to change the time to live (TTL), type of service (TOS), or MARK value. TOS field could be used for advanced routing. TTL field tells how long packet will live, before it will die. MARK is used for marking

the packets and it is for example used for advanced routing. The mangle table consists of five built-in chains, PREROUTING, POSTROUTING, OUTPUT, INPUT, and FORWARD chains. PREROUTING is used for modifying packets when they enter the firewall. POSTROUTING is used for mangling packets after the latest routing decision. OUTPUT chain is for locally generated packets, before routing decision. INPUT chain is used for modifying packets after the first routing decision when packets are routed to the local machine itself. FORWARD chain is for forwarded packets and it will modify packets after the first routing decision. (Mann & Krell, 2002, p. 730)

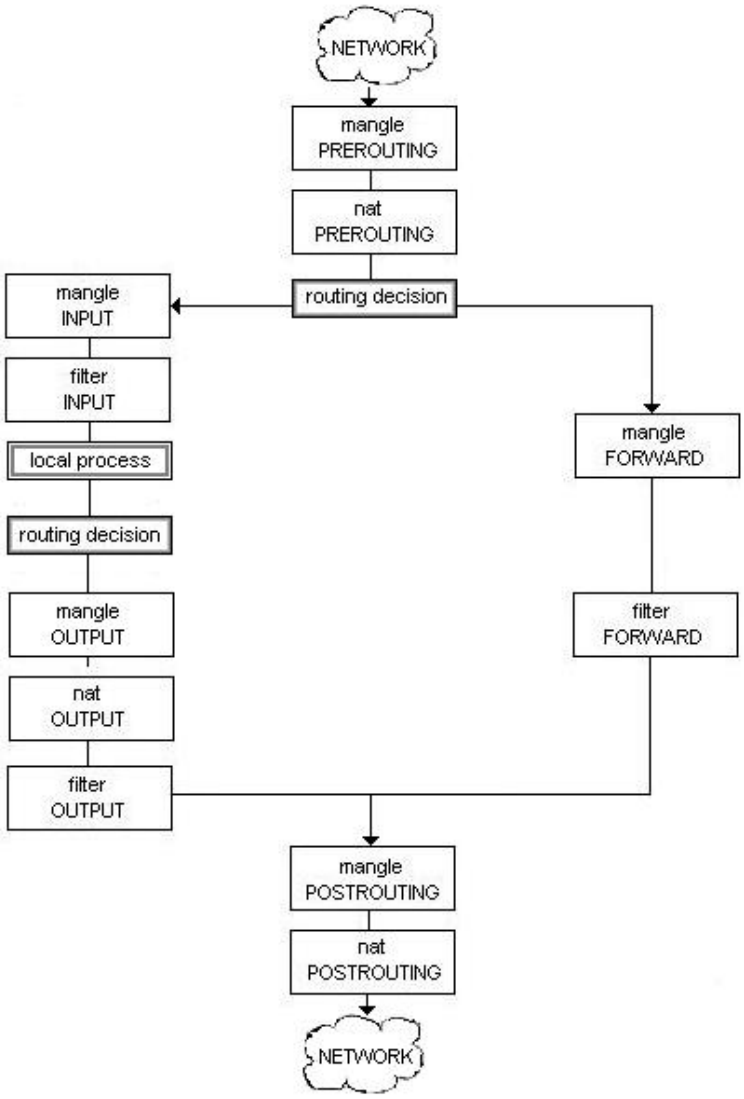

Figure 8. netfilter flowchart. (F. M. & J. A., "Firewalls – iptables", January 12, 2004)

Match

Match includes the special information about the packet. Firewall compares this information for the content of packets, which traverse over the specific chain where the rule is inserted. A single rule can contain several matches and that makes the firewall rule more specific. For example, it is possible to do matches from used protocol (TCP, UDP, ICMP or all), source IP address, destination IP address, hardware address (MAC), network interface, service port number, TCP options, ICMP type, and packet length. There are also some special matches like connection state match, limit match and owner match. (Mann & Krell, 2002, pp. 738-743)

Targets and jumps

The target or jump tells to the firewall what to do when packet is matched to the rule. The mostly used targets are DROP, ACCEPT, REJECT, and LOG. DROP means that packet will be just dropped dead and then firewall doesn't do anything else. REJECT is another way to deny packet traversing over the firewall, but it will send an error massage back to the packet sender. ACCEPT allows packet to continue traversing for another table or out from the firewall. The LOG target is designed for logging detailed information about packets. There are lots of other targets like RETURN, DNAT, SNAT, MARK, and REDIRECT. (Mann & Krell, 2002, pp. 744-747)

The jump is used at the same way as target, except it requires also the name of the destination chain where to jump. This chain must be in the same table and must be created already. It is very useful to create new chains, because it makes the firewall structure well ordered.

Example rules

These example rules will allow new http and dns connections to internet from our local host. It also allows ICMP echo request to internet (ping) and ICMP echo reply (pong) messages back to our local host. At first we flush all the filter table chains and also delete user made chains. After that we set the default policy to drop traffic for all chains in the filter table.

iptables -F

iptables -X

iptables -P INPUT DROP

iptables -P OUTPUT DROP

iptables -P FORWARD DROP

iptables -A OUTPUT -p tcp --sport 1024:65535 --dport 80 -m state --state NEW -j ACCEPT

iptables -A OUTPUT -p tcp --sport 0:65535 --dport 53 -m state --state NEW -j ACCEPT

iptables -A OUTPUT -p udp --sport 0:65535 --dport 53 -j ACCEPT

iptables -A OUTPUT -p icmp --icmp-type echo-request -j ACCEPT

iptables -A INPUT -p icmp --icmp-type echo-reply -j ACCEPT

Finally we inserted new rules to the beginning of the INPUT and OUTPUT chains. Following rules allow all already established and related connections.

iptables -I INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

iptables -I OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

2.6 Loading extra modules

If all the modules are not already compiled statically into your Linux kernel and there is need to get support example LOG target, it is possible to load modules by following command:

modprobe ipt_LOG

Modprobe is an intelligent module loader, because it also loads all the other needed modules and a loaded module will work appropriately. If you want to view already loaded modules, type:

lsmod

If you need information about some module, type:

modinfo [module]

There are also helper modules different protocols and other services, example for NAT, FTP and IRC. If you want to get FTP traffic work a properly, you need to load ip_conntrack_ftp module. (Henderson, 2005)

2.7 Dynamic host configuration protocol

The dynamic host configuration protocol (DHCP) provides configuration parameters to hosts in a client-server model. DHCP server allocates network addresses and delivers configuration parameters to hosts. DHCP supports three mechanisms for IP address allocation: automatic allocation, dynamic allocation, and manual allocation. In automatic allocation a permanent IP address is assigned to the client. Using dynamic allocation, the address is assigned for a limited period of time (lease time). The latest mechanism is manual allocation, where the address is assigned manually. (Vuksan, 2000)

In the Linux world, there are DHCP server and client daemons with configuration files. Server daemon is for sharing information for hosts and client daemon is used for receiving this information. This configuration is saved to */etc/dhcp3/dhcpd.conf* file in the Knoppix distribution.

Currently there are three different DHCP client programs for Linux, such as dhcpcd, pump and dhclient. If you are using Knoppix, you may find client module from */sbin/pump* and its configuration file from */etc/pump.conf*.

2.8 Network analysis tools

2.8.1 Monitoring IP traffic

There are many useful tools for monitoring IP traffic. These tools are often called packet sniffers or network analyzers. They can for example be used for trouble-shooting network problems and helping tool for firewall configuration, or education. Some of the well-known packet sniffers are tcpdump and Ethereal. Both of these programs need super user privileges to use. (Mancill, 2001, p. 113-115)

tcpdump is a common packet sniffer. It is a command line tool and it can be started without any arguments (tcpdump). It allows the user to intercept and display packets which have been transmitted or received over a network. More information about possible arguments is available in man pages. (Mancill, 2001, p. 115)

Ethereal functionality is very similar to tcpdump. Using Ethereal, at first you need to capture data and after that it is possible to explore captured data. Ethereal has a graphical user interface (GUI) and it makes it easy to use. When starting to capture traffic, click the Capture menu→!nterfaces. Then select the interface you want to capture by clicking Capture. After you think that you have got enough information captured, stop capturing by clicking Stop button. After that Ethereal will show captured data in main window. It is possible to see different statistics from captured data. (Mancill, 2001, pp. 122-123)

2.8.2 Nmap port scanner

Nmap is free port scanning software and it is used in command line. It is designed to detect open ports on a target computer and it supports many different scanning techniques. It can determine which services are running on those ports and which operating system the computer is running. Nmap is often confused with host vulnerability assessment tools such as Nessus. Nmap is built to evade intrusion detection systems. (“Nmap Reference Guide (Man Page)”, n.d.)

General syntax for nmap is
nmap [Scan Type] [Options] { target specification }

Typical Nmap scans are shown in Figure 10. First command checks available hosts to scan. It will do a ping scan for IP range 192.168.0.1-10 and option –T4 is for faster execution. Next command is for scanning open ports. Argument –A is used for enabling OS and version detection, -p1-1024 tells for the program that to scan the port range 1-1024. The target specification could be an IP address or host name. (“Nmap Reference Guide (Man Page)”, n.d.)

```
root@0[~]# nmap -sP -T4 192.168.0.1-10

Starting nmap 3.81 ( http://www.insecure.org/nmap/ ) at 2006-02-09 00:26 EET
Host 192.168.0.1 appears to be up.
MAC Address: 00:30:AB:1B:B6:B8 (Delta Networks)
Host 192.168.0.2 appears to be up.
MAC Address: 00:11:50:0B:6F:C1 (Belkin)
Host 192.168.0.5 appears to be up.
Nmap finished: 10 IP addresses (3 hosts up) scanned in 21.119 seconds
root@0[~]# nmap -A -p1-1024 -T5 192.168.0.2

Starting nmap 3.81 ( http://www.insecure.org/nmap/ ) at 2006-02-09 00:27 EET
Interesting ports on 192.168.0.2:
(The 1021 ports scanned but not shown below are in state: closed)
PORT    STATE SERVICE      VERSION
135/tcp open  msrpc        Microsoft Windows msrpc
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds Microsoft Windows XP microsoft-ds
MAC Address: 00:11:50:0B:6F:C1 (Belkin)
Device type: general purpose
Running: Microsoft Windows 2003/.NET|NT/2K/XP
OS details: Microsoft Windows 2003 Server or XP SP2

Nmap finished: 1 IP address (1 host up) scanned in 24.259 seconds
root@0[~]#
```

Figure 10. Using Nmap port scanner.

2.8.3 Nessus vulnerability scanner

Nessus is a comprehensive vulnerability scanning program. It is the world's most popular vulnerability scanner at the moment (12.2.2006) (“About”, n.d.). It consists of Nessus daemon and client. The nessusd does the scanning and the client presents the results to the user. In typical operation, Nessus begins by doing a port scan to determine which ports are open on the target and then tries various exploits on the open ports. The vulnerability tests, available as a large body of plug-ins, are written in nessus attack scripting language (NASL), a scripting language optimised for custom network interaction. (Rankin, 2004, p. 144)

2.9 Tcl/Tk

Tcl/Tk was created by John Ousterhout, a professor at the University of California at Berkeley, in the late 1980's. Tool Command Language (Tcl) is a very powerful dynamic programming language, suitable for a very wide range of uses, such as web and desktop applications,

networking, administration, and testing. It is open source, business-friendly and easy to learn. Tcl is language that is truly cross platform. Tk is a graphical user interface (GUI) toolkit for Tcl, but it is also shared for other dynamic languages. Tk can produce applications that run unchanged across Windows, Mac OS X and Linux and more. Tk's basic abstraction is a widget. The widget serves some purpose and it is usually inserted in the main window. Each widget has a default way how to use it. (Peralta, 2003)

Wish is a simple program. It is build using Tcl command language, Tk and a main program that reads commands from a standard input or from user selected file. At first it will create a main window and after that process Tcl commands. If you run wish command without arguments, then it will read Tcl commands from standard input. If user home directory includes a .wishrc file, the wish evaluates the file as a Tcl script just before reading the first command from standard input. If wish is started with an initial filename argument, then it reads commands from that file. It won't read therefore the standard input and there is no automatic evaluation for .wishrc file. ("wish man pages", n.d.)

Commands

Create a widget by giving widget type, for example, text, button, label or frame.

frame .frame1 -border 0 -relief groove
button .frame1.nbutton -text "Next"

.frame1 is name for a widget. There are also options available for each widget. Border specifies the border width for frame and relief specifies the 3-D effect desired for the widget. Second command tells that nbutton is a sub window for .frame1.

You can alter characteristics of a widget that has already been created. There are widget commands available for each widget. Example, there are available following commands for button: cget, configure, flash, and invoke ("Tk built-in commands – button manual page", n.d.). Next is used configure command for changing the value of the -text option.
.frame1.nbutton configure -text "Close"

Pack command is used for setting up a size and a position of a widget. Geometry manager means any procedure that helps you fit a widget to another widget. Without a geometry manager, your widgets cannot be seen on the screen. The simplest way to put the .frame1 widget under geometry management is to use following command. (Peralta, 2003)

pack .frame1

When you need to destroy a widget, it will be done by using a destroy command. For example destroy .frame1 as following command.

destroy .frame1

Example program

Next is presented simple program which is shown in figure 11. This example presents how easy it is to create GUI using Tk widgets. It could be launch in Linux using command *wish [filename]*.

```
frame .frame1 -border 0 -relief groove
pack .frame1
label .frame1.lab -width 30 -height 6 -background green
.frame1.lab configure -text "TESTING"
pack .frame1.lab -side top -fill both
button .frame1.exitb -text "Close" -command {exit}
pack .frame1.exitb -side left
button .frame1.clearb -text "Clear" -command { destroy .frame1}
pack .frame1.clearb -side right
```

Figure 11. The program looks like that when it has launched in Linux system.

2.10 Remastering Knoppix CD

When Knoppix CD doesn't include software you want to use or there is a need to up-to-date software, you need to customize Knoppix CD. Creating customized Knoppix CD isn't that difficult. There are different ways to go through this project and next is presented one of these methods.

Remastering Knoppix CD requires use of a hard drive and that used space must be formatted with a Linux file system such as EXT3. Remastering process needs at least 3 GB of hard disk space when you got 1 GB or more of RAM. When you have less than 1 GB of RAM, you need to create swap file. It is recommended that you have approx 4.5 GB of free hard disk space. If there is need to edit and/or format partitions, there is available QTParted program for that. (Rankin, 2004, p. 280)

Mount partition

When you got partition which meets these requirements, mount it with read and write permissions. Type as root following command or insert prefix sudo if you are Knoppix user. (Rankin, 2004, p. 281)

mount –o rw /dev/sda1 /mnt/sda1/

Generate a swap file

There is a need to have at least 1 GB of memory. Size of a swap file depends on how much you have RAM. Next command will create a file with size of 800 MB. Run all these commands from mounted partition.
dd if=/dev/zero of=swapfile bs=1M count=800

Then format swapfile using mkswap command.
mkswap swapfile

After that start using swapfile following command.

swapon swapfile

After that you can check available memory using *free* command. (Rankin, 2004, p. 282)

Copy the Knoppix file system

At first you need to copy a complete file system from Knoppix CD to the mounted partition. Create folders for these files and copy files by following commands.

mkdir source

mkdir source/KNOPPIX

cp –Rp /KNOPPIX/ source/KNOPPIX*

Then check that all the files have been copied by typing *ls source/KNOPPIX*. (Rankin, 2004, p. 282)

Install or update packages

Chroot command makes possible to use *source/KNOPPIX* directory for effective root file system. After you have chrooted the directory, there is possible for example to use command apt-get to install or update packages. If you need to get access to the internet, you have to copy */etc/dhcpc/resolv.conf* file to */source/KNOPPIX/etc/dhcpc/resolv.conf*. Chroot the *source/KNOPPIX* directory by following command.

chroot source/KNOPPIX

Then mount the proc file system, which provides access to the network and other interfaces.

mount –t proc /proc proc

Update the list of packages by using command

apt-get update

When you got a list of packages, you can install one of the packages by following command.

apt-get install [package]

If you only want update already installed packages, use the following command.

apt-get upgrade

After all, run the following command to delete all the cached packages.

apt-get clean

When you have finished, unmount the proc file system by typing

umount /proc

Then exit from chrooted environment by pressing *Ctrl-D*. (Rankin, 2004, pp. 283-284)

Master CD file system

After you have made all changes you need to create a master CD file system. This is done by copying all the files from the Knoppix CD, except the highly compressed KNOPPIX file. At first we will create folder for the new Knoppix CD file system and then copy files to it.

mkdir master

rsync –a --exclude "/KNOPPIX/KNOPPIX" /cdrom master/

Next generate highly compressed KNOPPIX file from the *source/KNOPPIX* directory. Following command takes quite a lot of time and it depends on used hardware.

mkisofs -R -U -V "Knoppix fs" -P "Knoppix" -hide-rr-moved -cache-inodes -no-bak -pad source/KNOPPIX | nice -5 /usr/bin/create_compressed_fs - 65536 > master/KNOPPIX/KNOPPIX

When the master CD file system is ready, you can generate md5sum for files you have edited. This is done by using an *md5sum* program. After that you need to create a CD image file that you can then test using Knoppix cheat codes and burn to the cd. Generate cd image by following command. (Rankin, 2004, p. 284)

mkisofs -pad -l -r -J -v -V "KNOPPIX" -no-emul-boot -boot-load-size 4 -boot-info-table –b boot/isolinux/isolinux.bin -c boot/isolinux/boot.cat -hide-rr-moved -o knoppix.iso master/

Test CD image file

Test the generated CD image file by using Knoppix cheat code. Reboot the machine and type the following command for a boot prompt. (Rankin, 2004, p. 285)

knoppix bootfrom=/dev/sda1/knoppix.iso

3 KEY PROJECT CONCEPTS

3.1 Firewall

3.1.1 What is a firewall?

Firewalls are needed to protect your internal network or host, and personal data. Firewall is a piece of hardware and/or software and its functions prevent forbidden communications. A basic task for firewall is to control traffic between different zones and then it will deny or allow different services, such as http and ftp. Typical zones are the internet and internal network. The internet network is zone with no trust and the internal zone has high trust. (Strebe, 2004, p. 74)

A firewall checks all the traffic between the two networks. If traffic is allowed, it will be route between the networks. Otherwise it will block the traffic. Firewall filters inbound and outbound traffic and it is possible to use different settings to each network cards, e.g. eth0 and eth1. Firewall can filter packet based on source and destination address, port numbers, specific type of traffic (different protocols), packet attribute or state. It is possible to keep logs, so you can see later what has happened in your network.

3.1.2 Firewall types

There are two types of firewalls: packet filters and application level gateways (proxy). Packet filters are divided for two categories: stateless packet filters and stateful packet filters. Firewalls supports often network address translation.

Stateless packet filtering firewalls work at the network level IP layer of TCP/IP and that's why it is usually used in routers. Stateless packet filtering system is designed to allow in only packets for the services you want. Every packet has a set of headers containing certain information. Filtering rules consist of source and destination IP address, source and destination port number, network interface, and used protocol. Checking the packet size, you can make sure that packet is valid. This helps to catch a number of denials of service attacks based on malformed packets. (Strebe, 2004, pp. 76-77)

Stateful packet filters work at the TCP layer of TCP/IP. Stateful filters use state information, information of TCP flags, and information of used port number when doing filtering. They monitor TCP traffic through the session. Sessions starts by handshaking between two devices. Intelligent decisions will control whole connection, so it isn't filtering individual packets. On the other hand, it won't allow any of the packets traverse through firewall, if packets don't belong to permitted session. It keeps track of packets it has seen, such as: how many other packets have recently been seen to or from the same host, is packet identical to a recently seen packet, is packet a part of larger packet (fragmented) and is the packet which appears to be a response to another packet. (Strebe, 2004, pp. 76-77)
Application level filters, proxies, are application specific. They can filter packets at the application layer of the TCP/IP. It is implemented through software. All of the connections will be always created to the proxy, which will make decision to allow or deny connection. Incoming or outgoing packets cannot access services for which there is no proxy. This means that if you got configured http-proxy server, it won't allow any ftp, telnet or other traffic through. That's why you need to configure all of the services you want to run your host or network. Examining packets at application layer, they can filter application specific commands. They offer a high level of security. (Strebe, 2004, pp. 80-82)

Firewalls often support network address translation (NAT) functionality. The process of NAT involves re-writing the source and/or destination address of IP packets. Using NAT, it hides the hosts IP addresses behind the firewall. This will improve the security. (Strebe, 2004, p. 77)

3.2 Sharing the internet connection

Linux has a very powerful IP stack built into the operating system. Linux is capable to do routing of your LAN to the internet using NAT or masquerading. These both options are made possible with iptables. Network address translation will also improve LAN security.

The main difference between these two methods is that masquerading needs only one public IP address, unlike NAT needs as many hosts as are used at the same time. (Hasenstein, 1997)

3.2.1 Masquerading

With masquerading you can map all private IP addresses on your LAN to the same public IP address. Masquerading helps to cut down on the number of public IP addresses you need. When all data traffic from your LAN to the Internet is mapped to the same, single public IP address, you will have no need for several public IP addresses. It is mainly used when host gets its IP address dynamically. (Ranch, 2005)

3.2.2 Network address translation

Network address translation (NAT) also allows you to use two sets of IP addresses. NAT differs from masquerading in the way that addresses are mapped. NAT is based on a one-to-one principle. This means that a particular single IP address is mapped to another single IP address. NAT can handle mapping of both single and groups of IP addresses. (Hasenstein, 1997)

NAT mapping can be static or dynamic. The static address translation entails that the original IP address is always tied to the same NAT IP address. With dynamic address translation the different private IP addresses are mapped to a pool of IP addresses and the NAT IP address may be a different one for each single connection. (Hasenstein, 1997)

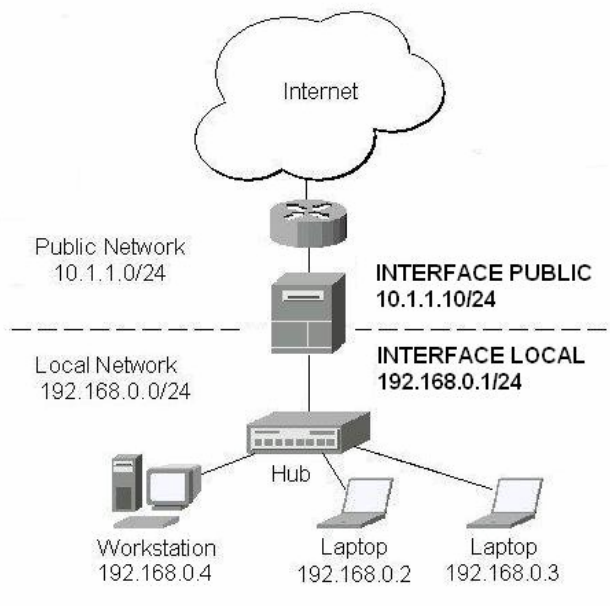

Figure 9. Network address translation.

In a typical configuration, a local network uses one of the private IP address subnets, for example 192.168.0.0/24. See private addresses on the table 3. A router on that network has a private address, such as 192.168.0.1. The router is also connected to the internet and it has at least one public IP address which is assigned by an internet service provider (ISP). When a host sends IP datagram to the public network from the local network, the source address in that datagram will be translated on the fly from the private address to the public address. The router keeps track about the basic data, such as destination address and port. When other machine replies to the router, it uses the connection tracking data it has stored and then determine where to forward the reply.

Table 3. Private IP addresses. (Mancill, 2001, p. 34)

Name	IP address range	number of IPs	network/mask
24-bit block	10.0.0.0 – 10.255.255.255	16,777,216	10.0.0.0/8
20-bit block	172.16.0.0 – 172.31.255.255	1,048,576	172.16.0.0/12
16-bit block	192.168.0.0 – 192.168.255.255	65,536	192.168.0.0/16

3.3 Remastering Knoppix

There is sometimes need for install new programs, remove some unnecessary programs, or just update already installed programs. This operation is called remastering process. Remastering process starts with arrangement to the needed hard disk space. You need to have about 2 GB free disk space for Knoppix files from *KNOPPIX* directory, about 700 MB for new highly compressed *KNOPPIX* image file, about 700 MB for new Knoppix iso image file, and if you have less than 1 GB RAM, you have to create swap file as well. In average you may need 4.5 GB for go through the remastering process.

When you have done the disk arrangements, then create directories for the Knoppix files and for the master CD files. Then copy all the files from the *KNOPPIX* directory in your hard disk and then copy all the files from Knoppix CD, except the big */KNOPPIX/KNOPPIX* file, to the master CD directory. These files are from highly compressed *KNOPPIX* file. Then you are capable to insert new files there, for example own script files, image files, or music files. If you want to for example install new packages, you first use chroot command for that. This command turns the copied files to the effective root file system. After that you are capable to use commands such as apt-get. When you have in chrooted environment you need to mount proc file system which provides access to the network and other interfaces. All of the dynamically created files are not copied when you go to the chrooted environment. That’s why you have to for example set up name server settings before you can use the internet.

After you have done all the changes you will create new highly compressed *KNOPPIX* file using mkisofs command. This will take for a while and you will save this file to the master CD directory. After that regenerate md5sums file for a list of checksums. Then you will have the master CD file directory which contains all the needed files to generate new Knoppix CD iso-image file. New CD iso-image file will be generated using mkisofs command. Once you have done all this, you will have new knoppix.iso file in your hard disk ready for testing. It is recommended that you will test it before burning process using Knoppix cheat codes.

4 PRIMARY DEVELOPMENT TOOLS

4.1 iptables

iptables consist of two parts: a user-space tool and kernel-space modules. The user-space tool is own program and it is called iptables. It is used for setting up the rules to the modules. When people are talking about iptables, they often mean a user-space tool. The kernel-space modules are distributed with the main kernel, and you compile them as you would any other module. ip_tables is a main module. There are other modules specifically for NAT, logging, connection tracking and so on. These modules perform appropriate function on the packets if there is set rules in the chain. (Coulson, 2001)

iptables is command line tool and it is supported in the Linux 2.4.x and 2.6.x kernels for IPv4. It talks to the kernel modules and tells it what to do with packets. It allows user to insert, modify, and delete rules from the kernel's packet filtering table. It is also possible do network address translation and masquerading using iptables. The iptables package also includes ip6tables. ip6tables is used for configuring the IPv6 packet filter. iptables was mostly written by the netfilter core team, but has received numerous contributions from lots of individuals. (“The netfilter.org "iptables" project”, n.d.)

iptables main features are stateless packet filtering for IPv4 and IPv6, stateful packet filter for IPv4, network address and port translation and large number of modules kept in ‘patch-o-matic’ repository. Using netfilter and iptables, it is possible to build firewalls, share internet access using NAT or masquerading, use NAT to implement transparent proxies, build quality of service (QoS) and policy routers, and do packet manipulation (mangling). (“The netfilter.org "iptables" project”, n.d.)

4.2 Ethereal

Ethereal is a free network protocol analyzer for Linux and Windows. Ethereal is useful tool for monitoring and trouble-shooting network traffic. It allows you to examine data from a live network. You can browse the captured data, viewing summary and detail information for each

packet. Ethereal has a nice graphical user interface which makes it easy to use. (Mann & Krell, 2002, pp. 80-81)

4.3 Tcl/Tk

Tool Command Language (Tcl) is a dynamic programming language, suitable for a very wide range of uses, such as web and desktop applications, networking, administration, and testing. It is open source, business-friendly and easy to learn. Tcl is a language that is truly cross platform. Toolkit (Tk) is a graphical user interface toolkit for Tcl, but it is also shared for other dynamic languages. Tk can produce applications that run unchanged across Windows, Mac OS X and Linux and many more. Tk's basic abstraction is the widget. A widget serves some purpose and it is usually inserted in main window. Each widget has default a way how to use. (Peralta, 2003)

4.4 Bash

The Bourne Again Shell (Bash) is the default shell on most Linux systems. A shell is simply a macro processor that executes commands. A shell allows execution of GNU commands those commands can be input by typing from the keyboard or from a file. A shell program is called a script. A Bash script is a file which contains commands. Bash shell reads each line of the script file and executes commands. Writing shell scripts is not hard to learn and the syntax is simple and straightforward. ("Bash", n.d.)

5 PROJECT BUILDING PROCESS

Project building process consists of different parts. First it is needed to make plans how to build firewall. After that you need to choose methods to implement these plans. Before starting to remaster new CD with new firewall program, it is necessary to test it by using debugging tools and other programs, such as internet browser, messenger programs and so on. At the end you will burn new Knoppix CD image file to the CD.

5.1 Planning firewall

Firewall can possibly act like a gateway machine or a standalone machine. User makes selection for external device which might be all possible modems (ppp+), all possible ISDN devices (ippp+) or one Ethernet device (e.g. eth0). User will also choose which Ethernet device is used for sharing the internet connection. If firewall is used as gateway machine, there are six possible directions for data traversing. Otherwise there are just two directions for data traverse. These possible directions are presented in figure 12. Figure 12 also shows the names of each chain which are created for filtering traffic between different zones. When packet comes to firewall, after the routing decision, packet will traverse to the right chain. When packet comes from eth0 and it is going to internal network (eth1) there are used rules from inttolo chain for filtering traffic (see figure 12). Jump for right chain is done by using information where interface packet comes from and where it is going to.

Linux has also very useful tools build-in kernel. proc file system offers possibilities to increase network security. For example activating source address verification is done by using rp_filter.

Nowadays there are different kinds of port scanning methods and these packets must be logged and dropped. There are also rules for allowing important icmp packets, such as destination-unreachable, time exceeded and parameter problem. There are also allowed echo-request (ping) and echo-reply (pong). All icmp packet rules include limit option. Limiting these packets increase security. All packets will traverse over badpacket chain.

End user has three choices. The first configuration is for standalone machine. It will allow all traffic from local host to internet and drop all traffic to other direction. Established and related connections are allowed to all directions. It is also possible to keep logs. Traffic which isn’t allowed or doesn’t match any of rules will be logged. Decreasing log file size is done using limit match. The second configuration is used for sharing internet connection. It will allow automatically all the traffic from local host and from internal network to internet and it will block all the incoming traffic from internet. The traffic between local host and internal network is allowed. Established and related connections are allowed to all directions. User can log packets

the same way as configuration 1, but there is also possible start dhcp server for selected interface. Internal network address will be 192.168.240.0/24. It will configure automatically selected interface for sharing. The third configuration allows user to select which ports are open to each direction. There is possible logging traffic the same way as other configuration methods and use dhcp server. User will select, is firewall acting as router or standalone machine. When the user has selected one of these configurations and activated a firewall, it is possible to use programs from menu bar. There is *open new port* and *redirect traffic.* Open port is used for opening new tcp or udp port for specific destination port number or range, and specific source port number or range. User also selects which chain adds this rule. Redirect traffic is used for redirecting specific port number for user selected IP address.

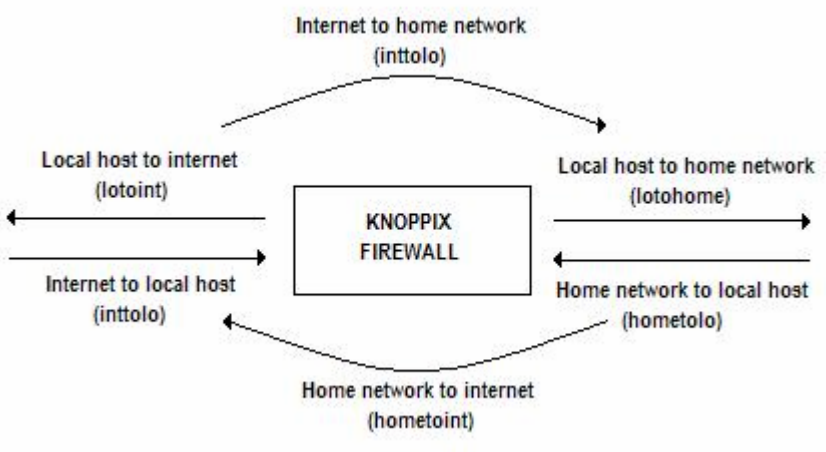

Figure 12. New chain names.

5.2 Implementing firewall

Firewall is build using bash scripting language, Tk toolkit and iptables command line tool. Bash scripting is used for reading configuration file and executes firewall rules based on configuration file. It will also run other necessary commands, example it is responsible for deactivating firewall when user clicks stop button in GUI. Tk is used for creating GUI. It will write configuration file and other help files. iptables command line tool is used as part of bash a script file. It is used for writing filtering rules and setting for kernel modules. Firewall program's operational chart is shown in figure 13.

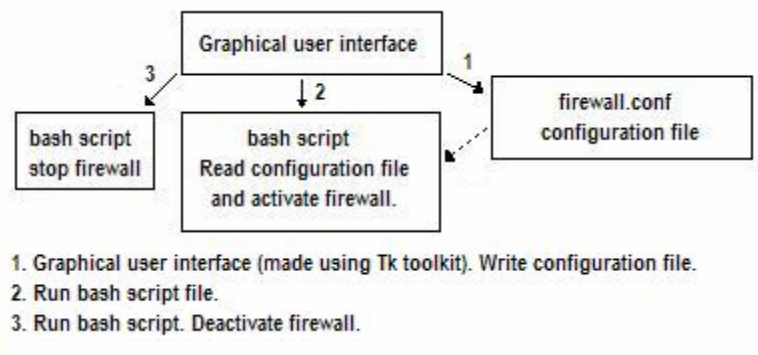

Figure 13. Firewall program's operational chart.

5.3 Debugging firewall

Debugging the firewall is important part of the process, because mistakes in code or in packet filtering settings might do the firewall worthless. Debugging is done by using iptables, Ethereal for monitoring traffic, nmap port scanner and Nessus vulnerability scanner. It is also good to test several different programs that they work correctly. iptables program is used for printing the rule sets for each chain. In this work firewall was tested by using VMware software. There were three virtual machines used for testing. One for Knoppix, which was used for sharing internet connection for vmnet1, and two for virtual machines (Linux, Windows XP) which was connected to vmnet1.

5.4 Remastering Knoppix CD

This part is for making new Knoppix CD with custom settings and programs. Inserting new item for K Menu, making changes to desktop, inserting new keyboard layout and many other things is possible do using KDE control center. This part I changed background image, inserted Finnish keyboard layout for default and inserted new item for *K Menu/KNOPPIX/Services* folder which is named Firewall. When generated new CD image file, then tested it using Knoppix cheat codes before CD burning. If image file is working well and everything is how planned, it is ready for put into CD. At this point it is important that image file is small enough to fit on CD. After all boot Knoppix using new CD and make latest checks that everything is still working correctly.

6 DISCUSSION

Project's aim was to build a Knoppix based firewall and router. Finished product implements this objective. It offers stateful packet filtering and uses other methods which Linux kernel itself offers to increase security. In this work masquerading is used for sharing internet and it is very useful if you are getting your IP address dynamically, just like most people are. There are no possibilities to choose internal network address, or use public IP addresses in your internal network. These new functionalities might be needed for future. One of the purposes in this work was to design the program easy to use for an end user, but it also offers features for advanced users. This was done by making simple graphical user interface. I think this new firewall is comprehensive to other firewall programs, such as Knoppix Firewall. Both of firewalls offer almost the same services, but this new also some extra services, such as DHCP server. User can also open just needed ports and no more.

Knoppix is very popular in these days and now I understand why it is. It is very easy to customize it for your needs. It offers opportunity to use many powerful programs quickly. Just insert CD on the drive and boot it up and you can do almost everything. Knoppix is great, but I wouldn't use it everyday and I think it is best for routing and packet filtering, for testing Linux, or for emergency use.

Reason why I chose to do this project was that in this work I can get good knowledge about Linux and firewalls. I am happy now, because this has been good starting point for using Linux in everyday and also get information where it's capable to use. It's also good to know how to use scripting languages, such as bash and Tcl/Tk. Only thing what I didn't like so much is the remastering process. I used laptop (Intel Celeron 1.5GHz, 512MB RAM) and external USB hard drive. It takes a very long time to have new remastered ISO-image file. If you weren't happy or had forgotten something from a new version of Knoppix, it takes a long time do another fixed version.

7 CONCLUSION

This project's aim was to build a packet filtering firewall and router for a home network or a small business network and include it into a Knoppix live-CD. In addition the firewall should be easy to use. This project was implemented by using iptables command line tool, bash script language, Tcl/Tk script language, and Knoppix live-CD version 4.0. Project's implementation demanded accurate planning and use of debugging tools such as Ethereal. The final program was tested by using several different network programmes and vulnerability tester. After testing the program was included into a Knoppix live-CD as a result of remastering process. Testing process showed that the program is working fine. The final program makes stateful packet filtering, port forwarding to the specific IP address, keeping the logs, and the use of dhcp server possible.

This project progressed fast and without any bigger problems, and this work's results meet the requirements of the project. It is, however more difficult to evaluate the program because end users will have different talents. I think this program's graphical user interface and text information will help user in every step. This program is aimed at people who want to share internet connection in a safe way. This project work can also be useful for those who want to create their own packet filtering firewall or to customize own Knoppix CD.

BIBLIOGRAPHY

Books

Mancill, T. (2001). *Linux routers: A primer for network administrators.* Upper Saddle River, N.J.: Prentice Hall PTR.

Mann, S. & Krell, M. (2002). *Linux TCP/IP network administration.* Upper Saddle River, N.J.: Prentice Hall PTR.

Rankin K. (2004). *Knoppix hacks.* Sebastopol, CA.: O'Reilly Media Inc.

Strebe M. (2004). *Network Security Foundations: Technology Fundamentals for IT Success.* Alameda, CA.: SYBEX Inc.

Internet documents

About (n.d.). Retrieved February 21, 2006, from Nessus, http://www.nessus.org/about/

Bash (n.d.). Retrieved March 23, 2006, from http://en.wikipedia.org/wiki/Bash

Coulson, D. (2001, May). *Internet security mastering iptables*. Retrieved November 22, 2005, from http://davidcoulson.net/writing/lxf/14/iptables.pdf

El-Saddik, A. (n.d.). Retrieved January 22, 2006, *TCP Finite State Machine*, from http://www.kom.e-technik.tu-darmstadt.de/projects/iteach/itbeankit/Applets/TCP/tcp.html#fsm

Eychenne, H. (2002, September). iptables man page from Linux.

Hasenstein, M. (1997). *IP network address translation.* Retrieved February 8, 2006, from http://www.hasenstein.com/linux-ip-nat/diplom/nat.html

Henderson, B. (2005, July). *Linux Loadable Kernel Module HOWTO.* Retrieved February 7, 2006, from http://www.ibiblio.org/pub/Linux/docs/HOWTO/other-formats/pdf/Module-HOWTO.pdf

Nmap Reference Guide (Man Page) (n.d.). Retrieved February 9, 2006, from Insecure.org, http://www.insecure.org/nmap/man/index.html#man-description

Peralta, S. J. (2003, August). *Scripting Graphical Commands with Tcl/Tk Mini−HOWTO.* Retrieved January, 2006, from http://www.ibiblio.org/pub/Linux/docs/HOWTO/other-formats/html_single/Scripting-GUI-TclTk.html

Ranch, D. A. (2005, November). *Linux IP Masquerade HOWTO.* Retrieved December 11, 2005, from http://www.ecst.csuchico.edu/~dranch/LINUX/ipmasq/pdf/ipmasq-HOWTO.pdf

The netfilter.org "iptables" project (n.d.). Retrieved January 25, 2006, from The netfilter.org project, http://www.netfilter.org/projects/iptables/index.html

The netfilter.org project (n.d.). Retrieved January 25, 2006, from The netfilter.org project, http://www.netfilter.org

Tk built-in commands – button manual page (n.d.). Retrieved February 16, 2006, from Tcl Developer Xchange, http://www.tcl.tk/man/tcl8.4/TkCmd/button.htm#M13

Wish man pages (n.d.). Wish man pages from Linux.

Vuksan, V. (2000, October). *DHCP mini-HOWTO.* Retrieved February 8, 2006, from http://www.ibiblio.org/pub/Linux/docs/HOWTO/other-formats/pdf/DHCP.pdf

APPENDIX A

Screenshot from Knoppix desktop