Public/Private/Wireless Information Security
A blue print for safeguarding sensitive information

Mark Rader and Professor J. D. Birdwell
Laboratory for Information Technologies
439 Science & Engineering Research Facility
The University of Tennessee
Knoxville, TN 37996-2100

ABSTRACT

Rapid access to investigative information stored on computer systems by narcotics and other law enforcement agencies is essential to the success of many investigations. Public, private, and wireless networks can provide connectivity among participants in an investigation and substantially enhance productivity. The unwary agency, however, can provide information about ongoing investigations to unauthorized persons, or expose the underlying information database to attack. A typical problem is the interconnection of narcotics units with agency networks with a minimal firewall or router between the narcotics unit and other organizations. This configuration allows sensitive data to exist on portions of the network where it is easy to program a personal computer to "snoop" the network and obtain user password and other text-based information.

This paper describes a way to create an affordable firewall that can be used both to compartmentalize networks (intranets) and to protect their resources from public Internet threats. The firewall uses the Linux operating system on a Pentium PC and the Trusted Information Systems (TIS) firewall toolkit (fwtk). A transparent encryption layer can be added to the firewall to protect sensitive data exchanged with external or internal cooperating agencies across insecure networks such as the Internet. The encryption layer also provides a basis for secure wireless access to network resources by investigators in the field.

By using inexpensive internal firewalls with strong encryption, investigative agencies can segment their internal networks to allow for supervised access control and encryption across sensitive segments. Network activity can be monitored without compromising security to aid the detection and prosecution of unauthorized users either inside or external to the agency.

1. Introduction

This document describes a firewall methodology which can be used to create affordable firewalls at institutions requiring information security for their internal networks or Intranets and the Internet at large. This type of firewall can be produced using the Linux operating system on a fast Pentium PC with either the Trusted Information Systems (TIS) toolkit or several of the publicly available TCP wrappers. All of these wrappers, proxies, and authentication schemes have

This paper was presented at the ONDCP/CTAC International Symposium, in Chicago, IL, August 18-22, 1997 pp 16-21-16-28. For additional information contact L.I.T. Ms. Leanne Ellis, Dept. of Electrical Engineering, Knoxville, TN 37996-2100. Lit@hickory.engr.utk.edu. http://www.lit.net/ondcp/index.html.

individual advantages and disadvantages over the other publicly available toolkits. The choice of toolkit and methodology comes down to two questions. How secure do I need my system, and how much outside access my users really need?

A firewall is described which both acts as a portal to a secure network and as a 10base-T to 100base-TX bridge. The secure network topology is a class C network with 2 subnets. One of these subnets is a 100base-TX leg connected directly to the secure side of the firewall. The second subnet is a 10BaseT segment connected to the secure network through a gateway machine. The firewall machine is secured with the philosophy of no access for all machines and no remote firewall access.

These firewalls can be used both as gateways to the Internet and as segmentation points to separate secure or sensitive data nets from the rest of an organization's network. The records stored on this firewall can be used to log accesses across the firewall and are essential in tracking and prosecuting hackers. A properly configured firewall can also be used to encrypt network traffic from the secure subnet to other secure subnets. This is a useful technique in preventing electronic eavesdropping of file and password information.

2. Philosophy of Firewalling and Security

The design and implementation of an Internet security system first begins at the site policy level. The basic security policies put in place by the site's administrators will determine the effectiveness of the firewall and the degree to which services will be limited to the desktop user. To put an effectively firewall security system in place, a site policy must be adopted which allows the firewall administrator to secure the Intranet/Internet interface and stop security violators. We advocate a philosophy of no access except those which are unless expressly permitted.

Physical as well as network security must be addressed in the design of a your firewall system. It is, in the author's opinion, better to have the firewall physically accessible to a select group of trusted users that to one person as penetration signs, such as excessive drive usage, will often be noticed by users and reported. The philosophy of our system will be to forbid login to the firewall system from other nodes of the network, or to limit network login to a select number of machines using a one time challenge-response system and/or packet level encryption.

Basic System Configuration

The system configuration is a low to mid performance IBM compatible PC with hardware modifications to enhance the system's performance for firewalling. The operating system is Linux with the 1.2.13 kernel, and the TIS toolkit. A minimal hard drive is used; however, the amount of logging information is generated by network traffic may require additional space. Two network cards are required. The first is a 3com 509 Combo card and will function as the interface to the outside network. The second is a 3Com 595 Vortec 10/100 baseTX Card and will function on the secure side of the network.

Dual Purpose Bastion Machines

Our firewall serves two purposes. Its primary function is that of a firewall and security monitor for our secure subnet. This allows us to monitor all incoming and outgoing traffic. Neither side of the firewall has any direct connection to the other except through the firewall host. This meets the definition of a Bastion Host.

The second purpose is to act as a bridge between the outside 10baseT network and the internal 100baseTX network. Selecting different network speeds and connection types offers several physical security advantages. One of the easiest ways to defeat a firewall security system is to create a physical connection from the secure network to the unsecured network with a simple patch cable. If the two networks are incompatible, except through the firewall bridge machine, security to your site is increased. Other external network adapters can be supported, including T-1 interfaces for point-topoint links.

System Selection

Selection of the hardware and software components of the firewall are critical to the success of the unit and minimization of network user frustration. A limited capacity machine will lead to network congestion and eventual collapse of the firewall due to overloading.

Hardware Selection

One of the major factors in selecting the hardware for the firewall is the network speed and usage of the Inter-

intranet getaway. As a rule of thumb, bigger is better and, too much capacity will not be noticed but too little will bring about user problems and slow down network operations. In particular, if 100Base-TX network connections are being used Pentium™ machines with chip speeds of greater than 100MHz are indicated. Memory can be a limiting factor of many of the off the shelf machine configurations. Practical experience has shown that while not memory intensive as a single process, multiple proxy connections can consume extraordinary amounts of memory particularly when the firewall machine may be acting as primary router and as a gateway between Ethernet segments of unequal speed. A CD-ROM drive is essential to setting up this system.

For our own internal network a Gateway 2000 P5-100 was selected with 48 Megabytes of RAM and a 1-Gigabyte hard drive. The machine was originally configured with 16 Mbytes of RAM, but this proved to be a major impediment to the firewall performance. The TIS firewall code has also been tested on P5-75 machines with satisfactory performance. A third firewall machine with a 486-50 processor was also constructed. This proved to be an unsatisfactory arrangement, as the Linux kernel and firewall code required 10-12 hours to compile.

As Linux was designed to be a poor mans version of UNIX, it has been found through experience that the less expensive and more common the hardware selected the greater the odds of success. Most of the drivers are written for the common denominator hardware components, so use of more exotic components many result in a loss of time while the proper drivers are acquired.

Software Selection

The operating system chosen for the firewall was Red Hat "Official" Linux. This package was chosen for a variety of reasons. These reasons include:

1. Ease of installation.
2. Stability of the Kernel
3. Utilities included with the distribution.
4. Final installation configuration.

Several versions of Linux were tested for ease of installation and the final configuration of the system drives. Significant problems were found with the selection of the proper boot disk images to install the software. This process has been automated with the 3.03 "Picasso" Release of Red Hat so that a CD-ROM is required. This version of Linux can also be installed over the Internet from the site ftp.redhat.com with the instructions provided there.

There are at least two publicly available firewall codes available over the Internet. The first is the Trusted Information Systems "TIS" firewall toolkit available from ftp.tis.com under the directory /pub/firewalls/toolkit; another option is the TAMU Tiger toolkit available from net.tamu.edu under the directory /pub/security/TAMU. The second site also includes some intrusion detection tools as well as a proxy firewall code.

Red Hat Linux also comes with a proto-firewall code in the form of TCPD for allowing and denying access to the machine login prompt. This is insufficient for true firewall purposes, as the code strategy is "what is not denied is permitted". This means that a table denying access to all machines except your own systems must be setup. Any flaws in the denial logic will permit access by unwanted machines into your system. Also, this access logic does not discriminate between services; if access is allowed for one service it is allowed for all supplied services.

Other sites offering useful software include BELLCORE with the S/Key software package at **thumper.bellcore.core** under the directory /pub/nmh/skey, Stanford with the SWATCH Logfile at **sierra.stanford.edu** under the directory /pub/sources, and Cert with the COPS package at _____.

3. Setting up the System

Once the system has been selected and purchased, it must be prepared for use as a firewall by making numerous changes both to the system software and the hardware attached to the computer system. These changes include removal of most or all of the Microsoft operating system components, addition of the network cards, and installation of the Linux operating system with all necessary system patches to ensure that the network cards and system hardware operates we suggest leaving the case off of the system during installation. This allows ready access to information about the system while the software is being installed.

Software Installation

The first step in installing the Linux OS is to repartition the hard drive into two parts using the `fips` software tool provided with most versions of Linux. This step

creates two logical disk drivers on a single physical device. The first partition is a DOS/Windows partition and is useful for some system maintenance task such as configuring network cards and other pieces of hardware. The second partition houses the Linux operating system. This partition will yet be subdivided into swap space and additional Linux partitions as required. In order to do this successfully a de-fragmentation utility must be run in order to compress all of the DOS/Windows software into the lower segments of the drive; otherwise, this software may be lost in the partitioning process.

The next installation step is to run the program called **RedHat.exe** located on the distribution CD-ROM. This starts the installation procedure and will ask several questions about system configuration and the installed hardware. The installer can usually easily answer these questions. In our case with the two 3Com network cards, we will select the driver for 509 3Com card. As there are no included drivers for the 595 Ethernet card, in the initial installation the 509 card will become eth0.

The Red Hat installation program will now boot the system into Linux to continue installation. In this portion of the installation process, the hard drive can be subdivided into additional partitions using the Linux analog of FDISK. A swap partition of at least 50MB should be defined, and we recommend that the remaining portion of the hard drive be split into two other partitions. This provides a total of 4 primary partitions: One for DOS, one for swap, and two for Linux. The reason for this extra partition is to allow the administrator to put dangerous applications into this area and dismount the partition during normal firewall operation. It is also recommended that all of the applications be installed. The Ethernet device for the insecure IP address is also configured during this step, as you access to several sites on the Internet is needed for later steps in the installation procedure.

Hardware Configuration

Configuring the hardware for the Linux operating system is fairly straight forward but is easier to do under the dos/windows environment. This requires the setup disks supplied by the manufacture of the various components. Very few of the components require any hardware setting modifications except for the two network interface cards for use on the secure and insecure side of the firewall. These two cards require some change to the IRQ and I/O address settings to function properly.

In particular, the high-speed interface card requires the transmission speed setting to be locked to the high-speed setting. Under the Linux 1.2.13 Kernel the driver available under Linux does not support the automatic mode sensing. The second card should require no modifications to the hardware settings, but knowledge of the settings is in order for the Linux loader (LILO) configuration to allow for the use of both network cards. For proper setup, both the IRQ and the I/O address are needed.

Required Patches

Most of the major security breaches in the UNIX operating system come from not installing required supplemental security patches. This leaves known security holes in the operating system. Known breaches can aid a potential intruder, once it is determined what operating system is being used by the secure system. RedHat Inc. maintains a series of security patches and alerts on their web site at **www.redhat.com**.

A second patch is required to provide Linux support for the fast network interface installed in the firewall machine. This patch is available at **http://cesdis.gsfc.nasa.gov/linux/drivers/vortex.html** and works for the Picasso edition of RedHat. This patch is required for our system. It is not required if two supported cards are used. Newer versions of the RedHat Linux support all of these cards, and not require the patches.

Configuring LILO

In order to use the second network interface card, the system must be made aware of the card and its associated IRQ and I/O address. This can be done manually at boot up but can be done automatically with changes to LILO or the **Li**nux **Lo**ader (LILO). Automation of the network card configuration is easily accomplished using the LILO method.

Automatic network configuration is added by editing the file lilo.conf under the /etc directory. The file listing below is typical of the machines with two Ethernet adapters:

```
boot=/dev/hda
map=/boot/map
install=/boot/boot.b
prompt
timeout=50
```

```
image=/vmlinuz
        label=linux
        append = "ether=3,0x280,0,1,eth1"
        root=/dev/hda3
        read-only
#       append = ""
other=/dev/hda1
        label=dosc
        table=/dev/hda
```

The line = "ether=3,0x280,0,1,eth1" must be added to this file to allow the slow speed card to be recognized. In this command line the first parameter is the IRQ setting of the 10baseT card. The second option is the memory I/O address of the 10baseT-network card. The next two parameters are constant for all network cards and the final parameter indicates the network card id. For other systems these parameters will change. One of the features of the network driver is that the high speed Ethernet device must be eth0.

After this file has been saved, the command “lilo” is run to reconfigure the master boot record (MBR) of the computer. This will add the command line to the boot up system. Otherwise the command ether=3,0x280,0,1,eth1 must be typed at boot-up each time.

Configuring System Software for Gateway Transport and Compiling the Kernal for 100baseTX operation

The final step in the initial configuration is to recompile the Kernel. This is accomplished by going to the directory /usr/src/linux and running the command make config. This will run the Kernel configuration routine. Most of the default options will be sufficient, but changes must be made to 3 of the default options. The changes are:

1. Answer “Yes” to Do you want to see Alpha Drivers.
2. Answer “Yes” to Enable IP forwarding and gatewaying
3. Answer “Yes” when offered the 595 driver.

This will enable the machine to act as a router for first testing of the network. It will next be necessary to run the command string make dep; make clean; make zlilo. This will rebuild the kernel and load it for use. The final step in this rebuild is to reboot the machine.

The final stage of the preparation process is to configure the IP address and network of each card. This is easily done as root under X-Windows, and is done from the Red Hat Package Manager control panel under network configuration. The 10baseT card will be eth1 with the 100baseTX card being eth0. Rebooting may be necessary after this step. It would be advisable to test the network configuration at this point and ensure that, not only does the firewall machine see both sides of the network, but also the computers inside see external systems.

4. Firewall Construction

The construction of the actual firewall consists of 3 steps. These steps include:

1. Recompiling the Kernel to break the network connection between the two network segments.
2. Obtaining compiling and installing the TIS tool kit.
3. Removal of Extra Services not needed by or dangerous to the firewall machine.

These steps are extremely time consuming and unless proper care is taken the firewall will have significant security holes in it.

Breaking the Gateway Connections

The first step in firewall construction is to eliminate the network connection formed during the Linux installation. Recompiling the Kernel as described above with the following options changed from their default does this:

1. Answer “No” to enable IP forwarding and gatewaying.
2. Answer “Yes” to enable IP firewalling.
3. Answer “Yes” to enable IP logging.

and recompile the Kernel.

This will disable the network forwarding and enable the native firewall protection under Linux. At this point, after rebooting the system both the Inter and Intranets will be physically and electronically separate.

TIS Toolkit

The next step in the process is to obtain, compile and install the TIS tool kit. The tool kit can be obtained from Trusted Information Systems using ftp at

www.tis.com under the directory /pub/firewalls. TIS has instituted an e-mail policy that in order to obtain the tool kit, an e-mail must be sent to their server and the exact location of the took kit will be disclosed. Details can be obtained under /pub/firewalls. In addition, the e-mail logs of frequently asked questions and reports of problems can be obtained at this address.

In addition to the TIS tool kits there are several independent patches developed for the tool kit and are available at the **site ftp://ftp.optimation.co.nz/pub/fwtk/fwtk-2.0alpha-Optimation1.tar.gz**. These files replace several of the tool kit files to remove several bugs. Namely they make changes to HTTP-GW to allow it to function properly under Linux.

Next it will be necessary to uncompress both the TIS tool kit and the Optimization patches, in that order, to a specific sub directory in the root home area. . For simplicity sake, the directory created will be called fwtk2alpha. The decompression of the tool kit creates a directory called fwtk in this sub directory, and the patches will have to be uncompressed into the fwtk directory. It will next be necessary to edit the file ***Makefile.config***, and change the following lines.

```
#FWTKSRCDIR = /u/b/mbr/firewalls/fwtk
FWTKSRCDIR = /usr/local/src/fwtk
```

to reflect the absolute path of the kit. In our case it will be

```
FWTKSRCDIR = /root/fwtk2alpha/fwtk
#FWTKSRCDIR = /usr/local/src/fwtk
```

changes are also necessary for auto install of the Manuel pages but are not required for the installation. Next run the command fix/make to create a customized make file for your system. Then as root run the command make. This will create the tool kit binaries. To install these binaries in the directory /usr/local/etc run the command make install. This installation directory can be configured in the file Makefile.config. This will complete the installation of the actual toolkit. The final two steps are required to configure the Linux OS to use the tool kit for IP proxy services.

Removal of Extra Services

To activate the tool kit proxy services it will be necessary to edit two files. They are /etc/services and /etc/inetd.conf. These will be configured to use the IP tool kit as opposed to the telnet.d and other services normally offered under Linux. It will also be necessary to remove extra services such as httpd and sendmail from the rc3.d and rc2.d directory to prevent their use. Sendmail in particular is known as being the largest security breech in the UNIX operating system. The listing, shown below, is typical of the services in a firewall stripped machine:

```
USER PID %CPU %MEM SIZE RSS TTY STAT START TIME COMMAND
bin 79 0.0 0.7 68 372 ? S Aug 14 0:01 rpc.portmap
root 1 0.0 0.7 39 368 ? S Aug 14 0:06 init [3]
root 48 0.2 0.8 73 424 ? S Aug 14 66:35 syslogd
root 57 0.0 0.7 52 360 ? S Aug 14 0:00 klogd
root 68 0.0 0.9 62 432 ? S Aug 14 0:00 crond
root 88 0.0 0.8 58 416 ? S Aug 14 0:48 inetd
root 99 0.0 0.7 84 368 ? S Aug 14 0:01 lpd
root 103 0.0 0.7 76 380 v02 S Aug 14 0:00 /sbin/getty tty2 VC lin
root 104 0.0 0.7 76 380 v03 S Aug 14 0:00 /sbin/getty tty3 VC lin
root 105 0.0 0.7 76 380 v04 S Aug 14 0:00 /sbin/getty tty4 VC lin
root 106 0.0 0.7 76 380 v05 S Aug 14 0:00 /sbin/getty tty5 VC lin
root 107 0.0 0.7 76 380 v06 S Aug 14 0:00 /sbin/getty tty6 VC lin
root 108 0.0 0.6 35 296 ? S Aug 14 0:00 bdflush (daemon)
root 109 0.0 0.6 35 304 ? S Aug 14 0:01 update (bdflush)
```

It can be noted that there is very little background process active on this machine to give an active security hole for a hacker to exploit.

The listing below shows typical services file prepared for firewall use:

```
tcpmux 1/tcp
# rfc-1078
#echo 7/tcp
#echo 7/udp
#discard 9/tcp sink null
#discard 9/udp sink null
systat 11/tcp users
#daytime 13/tcp
```

```
#daytime                13/udp
#netstat        15/tcp
qotd            17/tcp          quote
chargen         19/tcp          ttytst
source
chargen         19/udp          ttytst
source
ftp-data  20/tcp
ftp-gw          21/tcp
#ftp            22/tcp
telnet          23/tcp
#smtp           25/tcp          mail
#time           37/tcp          timserver
#time           37/udp          timserver
rlp             39/udp          resource #
resource location
name            42/udp
        nameserver
#whois          43/tcp          nicname
        # usually to sri-nic
domain          53/tcp
domain          53/udp
mtp       57/tcp                # deprecated
bootps          67/udp
        # bootp server
bootpc          68/udp
        # bootp client
#tftp           69/udp
gopher          70/tcp
        # gopher server
rje             77/tcp
#finger         79/tcp
httpd           80/tcp
ssl-gw          81/tcp
link            87/tcp          ttylink
#pop-2          109/tcp
        # PostOffice V.2
#pop-3          110/tcp
        # PostOffice V.3
#sunrpc         111/tcp
#sunrpc         111/tcp
        portmapper      #       RPC     4.0
portmapper UDP
#sunrpc         111/udp
#sunrpc         111/udp
        portmapper      #       RPC     4.0
portmapper TCP
auth            113/tcp
        authentication
sftp            115/tcp
 # End of services.
```

This listing is incomplete but it can be noted that several services such as pop-2 and pop-3 have been eliminated. Services that have still been provided are now shifted to use by the kit. The next listing is most important as it redirects the original functions to the FWTK. This is the innetd.conf file and is show below:

```
# Time service is used for clock
syncronization by folks to lazy to use NTP
#time           stream  tcp     nowait
        root    internal
#time           dgram   udp     wait
        root    internal
#
# Echo, discard, daytime, and chargen are used
primarily for testing.
#echo           stream  tcp     nowait
        root    internal
#echo           dgram   udp     wait
        root    internal
#discard        stream  tcp     nowait
        root    internal
#discard        dgram   udp     wait
        root    internal
#daytime                stream  tcp
        nowait  root    internal
#daytime                dgram   udp
        wait    root    internal
#chargen                stream  tcp
        nowait  root    internal
#chargen                dgram   udp
        wait    root    internal
#
# Wrappers
#
#ftp            stream  tcp     nowait
        root    /usr/sbin/in.ftpd
        /usr/sbin/in.ftpd
ftp-gw          stream  tcp     nowait
        root    /usr/local/etc/ftp-gw
        ftpd-gw
telnet          stream  tcp     nowait
        root    /usr/local/etc/tn-gw
#exec           stream  tcp     nowait
        root    /usr/local/etc/netacl
        in.telnetd
#shell          stream  tcp     nowait
        root    /usr/local/etc/netacl
        in.rshd
login           stream  tcp     nowait
        root    /usr/local/etc/rlogin-gw
#finger         stream  tcp     nowait
        nobody  /usr/local/etc/netacl
        in.fingerd
#smtp           stream  tcp     nowait
        root    /usr/local/etc/smap
        smap
```

```
#xserver          stream  tcp     nowait
        root    /usr/local/etc/netacl
        x-gw
#whois            stream  tcp     nowait
        root    /usr/local/etc/netacl
        whois-gw
#webster                  stream  tcp
        nowait  root    /usr/local/etc/plug-
gw              plug-gw webster
#nntp             stream  tcp     nowait
        root    /usr/local/etc/plug-gw
        plug-gw nntp
httpd             stream  tcp     nowait
        root    /usr/local/etc/http-gw
        httpd-gw
ssl-gw            stream  tcp     nowait
root    /usr/local/etc/http-gw  http-gw
#
# Authentication Server
#
auth              stream  tcp     nowait
        root    /usr/local/etc/authsrv
        authsrv
```

It can be noted that all of the services offered under inetd.conf are now redirected to the FWTK. These steps activate the firewall tool kit after rebooting the system or locating the inetd process and running the command kill -HUP # where # is the number of the inetd process. This is usually less than 100, and in the case of the example given above it is 88. At this point your machine is now a firewall with one major flaw. At this point neither side is allowed access to the firewall services, not even the host machine. The final step is to configure the firewall access privileges.

Firewall Access Control

The final step in configuring the firewall is to edit the netperm-table in the directory /usr/local/etc. This table gives the firewall its character and defines down to the user level, if need be, who can access network services. The listing given below is a typical netperm-table:

```
# Example netacl rules:
# ---------------------
# if the next 2 lines are uncommented, people
can get a login prompt
# on the firewall machine through the telnet
proxy
#netacl-in.telnetd: permit-hosts 127.0.0.1 -
exec /usr/sbin/in.telnetd
#netacl-in.telnetd: permit-hosts
xxx.xxx.xxx.xxx -exec /usr/sbin/in.telnetd
#netacl-in.telnetd: permit-hosts
xxx.xxx.xxx.xxx -exec /usr/sbin/in.telnetd
#
# if the next line is uncommented, the telnet
proxy is available
#netacl-in.telnetd: permit-hosts * -exec
/usr/local/etc/tn-gw
#
# if the next 2 lines are uncommented, people
can get a login prompt
# on the firewall machine through the rlogin
proxy
#netacl-in.rlogind: permit-hosts 127.0.0.1 -
exec /usr/sbin/in.rlogind -a
#netacl-in.rlogind: permit-hosts xxx.xxx.xxx.*
-exec /usr/sbin/in.rlogind -a
#
# if the next line is uncommented, the rlogin
proxy is available
#netacl-in.rlogind: permit-hosts * -exec
/usr/local/etc/rlogin-gw
#netcal-in.ftpd: permit-hosts 127.0.0.1 -exec
/usr/sbin/in.ftpd
#netcal-in.ftpd: permit-hosts xxx.xxx.xxx.*-
exec /usr/sbin/in.ftpd
#netcal-in.ftpd: permit-hosts * -exec
/usr/local/etc/ftp-gw
#
# to enable finger service uncomment these 2
lines
#netacl-fingerd: permit-hosts xxx.xxx.xxx.*-
exec /usr/libexec/fingerd
#netacl-fingerd: permit-hosts * -exec /bin/cat
/usr/local/etc/finger.txt

# Example smap rules:
# -------------------
#smap, smapd:    userid 6
#smap, smapd:    directory /var/spool/smap
#smapd:          executable
/usr/local/etc/smapd
#smapd:          sendmail /usr/sbin/sendmail
#smap:           timeout 3600
# Example ftp gateway rules:
# --------------------------
#ftp-gw:         authserver localhost 7777
ftp-gw:  denial-msg     /usr/local/etc/ftp-
deny.txt
#ftp-gw: welcome-msg    /usr/local/etc/ftp-
welcome.txt
#ftp-gw: help-msg       /usr/local/etc/ftp-
help.txt
```

```
ftp-gw:          timeout 3600
# uncomment the following line if you want
internal users to be
ftp-gw:          permit-hosts xxx.xxx.xxx.*
# uncomment the following line if you want
external users to be
# able to do FTP with the internal network
using authentication
# Example telnet gateway rules:
# ----------------------------
tn-gw:           denial-msg
        /usr/local/etc/telnet-deny.txt
#tn-gw:          welcome-msg
        /usr/local/etc/tn-welcome.txt
#tn-gw:          help-msg
        /usr/local/etc/tn-help.txt
#tn-gw:          timeout 3600
tn-gw:           permit-hosts xxx.xxx.xxx.* -
passok -xok
# if this line is uncommented incoming traffic
is permitted WITH
# authentication required
# Example rlogin gateway rules:
# ----------------------------
rlogin-gw:       denial-msg
        /usr/local/etc/rlogin-deny.txt
#rlogin-gw:      welcome-msg
        /usr/local/etc/rlogin-welcome.txt
#rlogin-gw:      help-msg
        /usr/local/etc/rlogin-help.txt
#rlogin-gw:      timeout 3600
rlogin-gw:       permit-hosts xxx.xxx.xxx.* -
passok -xok
# if this line is uncommented incoming traffic
is permitted WITH
# authentication required
# an attempt at http rules
http-gw: timeout 3600
http-gw:     denial-msg
/usr/local/etc/httpd-deny.txt
#http-gw:         permit-hosts 128.169.6.73
#http-gw:         permit-hosts 128.169.132.31
http-gw: permit-hosts xxx.xxx.xxx.*
# ssl-gw proxy
ssl-gw:          permit-hosts xxx.xxx.xxx.* -
dest { !127.0.0.1 ! xxx.xxx.xxx.* *:443:563 }
plug-gw:         permit-hosts xxx.xxx.xxx.*
# Example auth server and client rules
# -----------------------------------
authsrv: hosts xxx.xxx.xxx.1
authsrv: database /usr/local/etc/fw-authdb
authsrv: badsleep 1200
authsrv: nobogus true
# clients using the auth server
*:               authserver xxx.xxx.xxx.1
7777

# X-forwarder rules
#tn-gw, rlogin-gw:        xforwarder
/usr/local/etc/x-gw
```

This netperm table, disables all but console login to the firewall machine and only allows the secure subnet to pass outward. DNS, SNMP and other dangerous services are denied the use of the firewall. In particular x services are disabled. One point of interest is that the TIS kit looks for the first instance. This can create an accidental security hole in your firewall. This can be accomplished by reversing the order of the commands. A typical example is

```
ftp-gw permit-host 111.111.111.*
ftp-gw deny-host 111.111.111.123
```

where the network to be protected is 111.111.111.* and the host to be denied access is 123. This will not work as the deny is after the global permit. The order on these two commands needs to be reversed in order for this to work.

5. Results of Satan Tests

In order to test the firewall effectiveness SATAN was used against the system at full user load to check for security leaks. SATAN was used in unfriendly mode and there was a noticeable effect on firewall performance. SATAN was unable to find any security holes in the system to exploit and often gave the user false information. A typical example is on the machine type. In most instances SATAN returned the wrong machine type to the operator.

6. Conclusions

We have successfully installed and demonstrated the practicality of building a Linux based firewall based upon the TIS tool kit. In addition, it has been demonstrated that the firewall is an effective deterrent to both the casual and serious hacker through the use of SATAN.