Information Security Practices Recommended For Use PERPOS2

Mr. Dariush Molavi

9 January 2004

PERPOS TR ITTL-CSITD 04-2

Table of Contents

1. Summary

The primary purpose of this paper is to briefly outline the information security policies and procedures that are recommended for use on the PERPOS2 server. The basic methods outlined here are all based upon established guidelines for enhancing existing security policies.

The bulk of the procedures presented here center around software solutions to the information security issue. The first software product discussed is **iptables** and the second is **PortSentry**. Both of these products are freely available under either the Common Public License (CPL) or the GNU General Public License (GPL). These software packages have been proven to provide effective entry-level protection against the most common security exploits.

What this paper does *not* discuss are topics such as basic physical security, partitioning rules, password security, etc. This is not intended to be a "one stop shop" for information security needs; rather, it is, as stated above, simply an outline of what policies should be used on PERPOS2.

2. Background Information

The machine on which these policies are recommended for use upon is the PERPOS2 server. This server has the following specifications:

Dual Pentium 4 Xeon @ 3.06Ghz
2GB RAM
Two 36.4GB HDD in a RAID configuration
Five 146.8 GB HDD in storage controller

Listing 1. PERPOS2 Server Configuration

The operating system on the server is Gentoo Linux 1.4, with the 2.4.24 Linux kernel sources.

The following services are running on PERPOS2:

Apache Web Server (1.3.29)
MySQL Database Server (3.23.57)
SSH Daemon (OpenSSH 3.7.1p2)
Postfix Mail Daemon (2.0.11)

Listing 2. Services on PERPOS2

Each service is running on its default port: httpd:80, mysql:3306, sshd:22, postfix:25.

Services with known security issues or which are known to decrease the overall security of the machine have not been installed. Examples of such services include FTP and telnet.

The primary purpose of this server is to provide the PERPOS development team with a platform on which to test various information assurance policies. PERPOS2 will eventually house an Oracle database and web portal, and will be the subject to various methods of attacks from the development team.

3. Iptables

In a nutshell, iptables[1] is a program which provides packet filtering, network address translation, and other packet mangling capabilities. In the context of this paper, the focus will be on the packet filtering capability of iptables.

Simply put, packet filtering is the backbone of a firewall. The firewall software examines each packet that traverses the network interface and, depending on the content of the header, performs certain predefined actions. Iptables stores sets of rules in various tables (hence the name of the software) which it uses for the comparison.

The following two diagrams illustrate an IP and a TCP packet header.

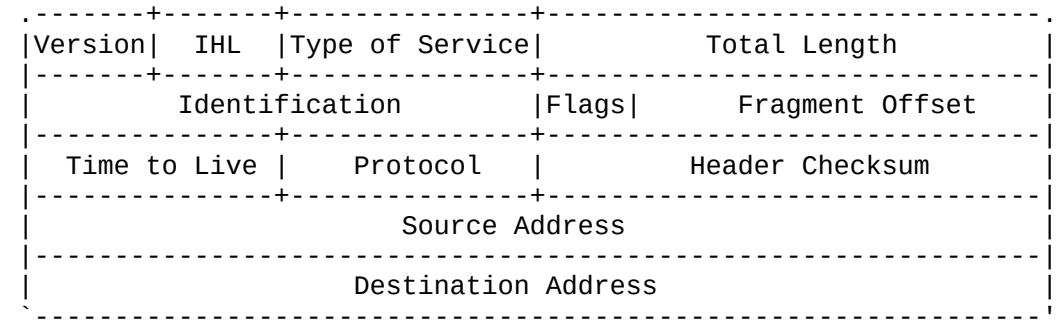

Figure 1. IPv4 Packet Header

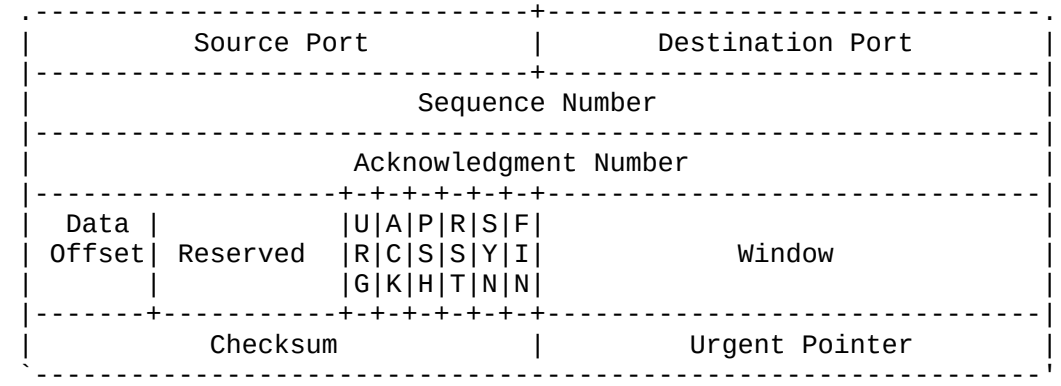

Figure 2. TCP Packet Header

[1] More detailed information on iptables can be found at http://www.netfilter.org.

Iptables examines only the header content, not the content of the data contained within the packet's payload, and thus can't be used for virus checking, spam checking, etc. What iptables can check is the source/destination address, source/destination port, and the protocol of the data packet.

Each packet is compared against each of the rules in the firewall rule set, and if there is a match, the corresponding action is taken. The most common action for the firewall would be to drop the packet, which makes the host seem non-existent to the remote user. An example of such a rule is shown in code listing 1, below:

```
iptables –A INPUT –s 130.207.202.91 –j DROP
```

Code Listing 1. Example iptables rule.

This rule, when entered at the command prompt, tells the iptables program that any packet coming into PERPOS2 from the network whose source IP address is 130.207.202.91 should be dropped, preventing any connection from being made. No response will be sent back to the source computer. This type of packet filtering is called "stealth" firewalling.[2] This has the effect of making it seem as if no computer exists when a user from the source IP address tries to initiate a connection. If the user is a hacker, this is a good thing, as they may conclude that the IP address associated with PERPOS2 is not active, and therefore should not be explored further.

However, if the user is a regular, everyday internet surfer, such a broad blocking action may not be the wisest choice. If the user's computer has become infected with a virus, they may not be aware that their computer is attacking others. Combined with PortSentry, discussed below, iptables allows administrators to block not only based on the source IP address, but also on the destination port.

For example, if a user's computer has become infected with the Blaster worm, it will start sending out large amounts of traffic to other machines, specifically, to port 135 of the target machine. If an administrator were to block just the IP address of the attacking machine, the user will not be able to access a web site being hosted on the server, since no distinction is being made between the individual ports being accessed. In this case, it is desirable to block access to only port 135 from the user's machine, 130.207.202.91:

```
iptables –A INPUT –s 130.207.202.91 –p tcp –dports 135 –j DROP
```

Code Listing 2. More detailed iptables rule.

This rule tells iptables to block access to 130.207.202.91 only when the protocol being used is TCP, and when the destination port on PERPOS2 is port 135. This will allow the user to continue viewing web pages (which are served on TCP port 80).

---

[2] *LinuxPro*, March 2003, Page 22

4. PortSentry

PortSentry[3] is an excellent tool to assist network administrators with detecting port scans on their servers. Scanning ports is a common first step that hackers use to determine if a machine is running vulnerable services which they may then be able to exploit to gain root (administrator) access on the machine, run arbitrary code, etc. Detecting a port scan is just one of many ways to detect an attempt at unauthorized external access to your network.

PortSentry detects scans by listening to a user-defined list of ports. If it detects that someone is trying to initiate a connection on that port it can respond in any of four ways:

- Make a log entry.
- Add the remote hosts to hosts.deny to drop the connection via TCP wrappers.
- Reconfigure the local host to set up a "black hole" route which routes packets to non-existent address.
- Reconfigure the local host to drop the packets via a packet filter.

In the context of this paper, the fourth method is the method of choice, with the packet filter being iptables.

PortSentry runs in the background, with no user intervention required once it is started. Configuration is simple: list the ports to which access is not desired. On a server such as PERPOS2, the only ports which should be open should include port 80 (for http traffic), 25 (for email), and 22 (for remote SSH access). All other ports can be included on the list of denied ports.

The next configuration step is to decide what action to take when unauthorized access has been detected. As stated above, the "kill" command that should be used for PERPOS2 is to initiate an iptables script that will automatically drop all packets from the originating host.

Combined, PortSentry and iptables provide a good first line of defense against unauthorized access to a server. What they do not protect against are exploiting applications and/or daemons that run on allowed ports, for example, if there is a bug in Apache that can be exploited to get root access or run arbitrary code.

5. Summary

It is important to note that this document covered basic firewall and port scanning detection techniques only. The steps outlined in this document provide a good first step to assuring that the PERPOS2 server remains secure. However, security measures beyond this include physical security, password security, and insuring that all

[3] More detailed information on PortSentry can be found at http://sourceforget.net/projects/sentrytools.

applications that are available to remote users are kept up to date and do not contain any vulnerabilities which may undermine other protection mechanisms.