Crossroads
The ACM Student Magazine

Introduction to Linux Networking and Security

by ***Wei-Mei Shyr*** and ***Brian Borowski***

Linux is a member of the UNIX family but is different than most UNIX implementations because it provides a great UNIX server/workstation environment at a low cost, can be run on a wide variety of platforms, and contains no proprietary code. In this article, we will give a brief introduction to the IP networking services, how to configure them, and how to set up a relatively secure Linux workstation. Please note that the examples given here are from the Slackware distribution. The paths of the files might be different on other distributions of Linux.

Linux TCP/IP Network Services

Linux supports a full and high quality implementation of the TCP/IP networking protocols. With a network interface card or a modem and PPP, one can connect a machine to a local area network or the Internet and have access to many additional services and network utilities. Linux provides two methods of establishing host-network services. Servers can either run stand-alone or under the control of a program called `inetd`. Heavily used services will usually run stand-alone. This means the service does all the management and listening on a socket or port. The most common stand-alone services are `inetd`, `syslogd`, `portmapper`, `named`, and `routed`. The file `/etc/rc.d/rc.inet2` configures the stand-alone services. Here is an example of `/etc/rc.d/rc.inet2`

```
#!/bin/sh
#
# rPREc.inet2      This shell script boots up the entire INET system.
# Constants.
NET="/usr/sbin"
IN_SERV="lpd"
```

```
LPSPOOL="/var/spool/lpd"
echo -n "Starting daemons:"

# Start the SYSLOGD/Klogd daemons.  These must come first.
if [ -f ${NET}/syslogd ]; then
  echo -n " syslogd"
  ${NET}/syslogd & # Backgrounded to avoid an ugly notice from bash-2.0
  echo -n " klogd"
  ${NET}/klogd
fi
...
# Start the INET SuperServer
if [ -f ${NET}/inetd ]; then
  echo -n " inetd"
  ${NET}/inetd
else
  echo "no INETD found.  INET cancelled!"
  exit 1
fi
....
```

However, most services run through inetd. inetd is a daemon or background process that starts up near the beginning of the boot sequence in Linux. inetd listens on many ports, and when a connection to a port is requested, it starts up the process associated with that port.

Examples of services run from inetd are ftp, telnet, finger, pop, imap, and mail/smtp. inetd is like a switch-board operator who receives calls at the main number of an organization (the IP address of the machine), and then connects the caller to the extension they have requested (the port or socket).

There are two files that configure inetd: `/etc/services` and `/etc/inetd.conf` (which may be in `/etc/inet/inetd.conf`). Below is an example of `/etc/inetd.conf`

```
# See "man 8 inetd" for more information.
#
# <service_name <sock_type <server_path
#
# The last 3 services  ( pop3, imap, uucp) are really only used for
# debugging purposes, so we comment them out since they can
# otherwise be used for some nasty denial-of-service attacks.
# If you need them, uncomment them.
#
# ftp and telnet are the standard services.
#
ftp     stream  tcp     nowait  root    /usr/sbin/tcpd  wu.ftpd -l -i -a
telnet  stream  tcp     nowait  root    /usr/sbin/tcpd  in.telnetd
# installed the Pine package, you may wish to switch to ipop3d by
# commenting out the pop3 line above, and uncommenting the pop3 line below.
#pop3    stream  tcp     nowait  root    /usr/sbin/tcpd  ipop3d
# imap2   stream  tcp     nowait  root    /usr/sbin/tcpd  imapd
#
# The Internet UUCP service.
#
# uucp  stream  tcp     nowait  uucp    /usr/sbin/tcpd  /usr/lib/uucp/uucico
-l
....
```

Configuring Network Services

To configure the stand-alone services, edit /etc/rc.inet2. Disable a service by commenting out the lines related to that service. A line is commented out by placing a # before it. Here is an example of a commented out service:

```
#   Start the ROUTEd server.
# if [ -f ${NET}/routed ]; then
#   echo -n " routed"
#   ${NET}/routed -g -s
# fi
```

To configure the inetd services, edit /etc/services and /etc/inetd.conf. The /etc/services file associates services with their ports. It lists the name of the service, the port number for that service, and the protocol (udp or tcp). Here is the line for the ftp service:

```
ftp      21/tcp
```

/etc/inetd.conf contains parameters that determine how the services runs. Here is an example of the line for the ftp server:

```
ftp     stream  tcp     nowait  root    /usr/sbin/tcpd  wu.ftpd -l -i -a
```

To disable the ftp program, comment it out by putting a # at the beginning of the line. To activate the change, reload inetd. This is done by finding the process-id (PID) of inetd, and then sending it the hangup signal known as SIGHUP or just HUP.

```
{find out the PID}
$ ps -aux | grep inetd
root       479  0.0  0.2 1944 1520 ?        S   Mar 02  1:18 /usr/sbin/inetd
        {   ^ this is the PID}
$ kill -HUP 479
```

The file /etc/services will most likely only need to be edited when adding new services. This might be necessary when installing network utilities.

Note that we use tcpd to control access to the ftp daemon. The tcpd program is a wrapper program that can be set up to monitor incoming requests for telnet, finger, ftp and other Internet services. It works as follows: whenever a request for service arrives, the inetd daemon runs tcpd, which logs the request and does some checking. When all is well, tcpd runs the appropriate server program and goes away. For details, see the tcpd manual page. Access control for tcpd is configured using the /etc/hosts.allow and /etc/hosts.deny files. tcpd looks at hosts.allow then hosts.deny. It stops at the first match. Consequently, one can permit a few machines to have ftp or telnet access and then deny access to everybody else in hosts.deny. Here is a sample /etc/hosts.allow:

```
ALL: 10.100.10.0/255.255.255.0
```

The ALL refers to all wrapped inetd services. This does not include stand-alone services. The second field 10.100.10.0/255.255.255.0 means all machines on the 10.100.10.0 subnet have access to all the services. Now we want to disallow access for everybody else. Put the following line into /etc/hosts.deny:

```
ALL: ALL
```

The non-existence of the /etc/hosts.* files or empty /etc/hosts.* means no restriction. This is an insecure configuration unless legitimate connections might come from many diverse networks.

Security: An Overview

People often ask, "How secure is my machine?" The answer is that any publicly accessible machine is necessarily insecure and vulnerable to security problems. Hence, we should take proper steps to minimize the vulnerability. There are three different aspects of security: physical, system, and network.

Physical security is the first layer of security. Home users probably need not worry about this too much. However, in a public environment, this aspect of security is a much larger concern. Keep in mind that re-booting the system is only a ctl-alt-del away if users have access to the console. If users can reboot the system, it is trivial to manipulate the data on the system. Whenever possible limit user access to the console.

System Security is a topic all by itself and addresses issues such as restricting user accounts to the minimal necessary privileges. For example do users really need a full shell environment or will a restricted menu system do? System security also involves choosing secure, hard-to-guess passwords; reading CERT bulletins and applying patches when necessary; and not allowing root to log in from any terminal except the console. This means the file `/etc/securetty` should have only one line in it:

```
console
```

System administrators have to log in as themselves first, then run **su.** For increased accountability, this program logs the user name of those who became root.

Network security is the most vulnerable part of your system. The following recommendations will significantly improve network security:

- Strip down the OS
  In standard Linux installations such as Slackware, Debian or Red Hat, many network services are enabled by default. This may be a good thing if when setting up a server, but when configuring a user's workstation, many of these services have no benefit, and may pose serious security concerns. Disabling these services is a good idea. In fact, the rule that most users should follow is that any services you do not intend to use should be disabled.

  Under Linux, system processes are started at boot time by adding and removing files in /etc/rc.d. For example, `sendmail` is started from the file `rc.M`. To disable such a service, you comment out the corresponding lines. In some Linux distributions, these services are in `/etc/rc.d/rcN.d`, where N is a number (the system run level). Disable services by deleting or renaming the files in the `/etc/rc.d/rcN.d` directory.

Other candidates are named, routed, and httpd.

- Disable unnecessary inetd network services
  Disable unneeded inetd services, in the manner described above **(inetd.conf)**.

  Many inetd services are not necessary. Comment out any that are not needed. Good candidates are: nntp (news), finger, uucp, the ``r-commands'' like rsh, rlogin, and rexec. Use SSH instead, see below.

  Some services to possibly leave enabled are: ftp (in.ftpd), but configure ftp not to permit anonymous access unless absolutely necessary; telnet (in.telnetd), the user interface for remote access; and auth (in.identd), the user identification program.
- Disable unnecessary stand-alone services (`/etc/rc.d/rc.inet2`)

  Only `inetd` and `syslogd` are essential. The rest can be commented out if not needed.
- Use SSH as a secure replacement for rlogin, telnet and rcp
  SSH uses cryptography to mutually authenticate users and hosts. It also encrypts the stream of data for confidentiality. When SSH is used, all data sent across the network is encrypted; this assumes that it is operating in a secure mode with the normal RSA authentication and public-key encryption enabled. This makes it very difficult for eavesdroppers to obtain useful data by intercepting the stream of traffic.
  For more information, see http://www.cs.hut.fi/ssh/.
- Use TCP-wrappers to control the access to inetd services
  Define the access lists in `/etc/hosts.allow` and `/etc/hosts.deny`.
- Use the latest `sendmail`

  Keep up with the latest stable version of sendmail. Disable it completely if email services are accessible elsewhere.
- Use Tripwire as an early intrusion detection system
  Tripwire maintains a checksum database of important system files. It is available via anonymous ftp from `ftp://ftp.auscert.org.au/pub/coast/COAST/Tripwire`

Recent Security Incidents

The following are a few Linux security advisories that have been announced recently. You can find more in-depth descriptions of the incidents at **cert.org** .

Buffer-Overflows
In some programs, boundary checking is not done for the pre-allocated buffers. When such buffers are overflowed, the executing program (daemon or set-uid program) can be tricked

into performing various abnormal operations or functions. Generally this works by overwriting a function's return address on the stack to point to another location, then executing either a root shell or code that might change the protection on a program such that it can then acquire root privileges.

99-03: FTP-Buffer-Overflows
By supplying carefully designed commands to the ftp server, intruders can force the server to execute arbitrary commands with root privilege. Any server running the latest version of ProFTPD (1.2.0pre1) or the latest version of Wuarchive ftpd (2.4.2-academ[BETA-18]) is vulnerable.

98-12: Buffer Overflow in Some Implementations of IMAP Servers
The overflow is in library code from the University of Washington IMAP server that handles SASL server-level authentication. Remote intruders can execute arbitrary commands under the privileges of the process running the vulnerable IMAP server. If the vulnerable IMAP server is running as root, remote intruders can gain root access.

Remotely Exploitable Buffer Overflow Vulnerability in mountd
On some systems, the vulnerable NFS server is enabled by default. This vulnerability can be exploited even if the NFS server does not share any file systems. All un-updated versions of Red Hat Linux are vulnerable.

"sscan" Scanning Tool
The sscan tool performs probes against victim hosts to identify services which may potentially be vulnerable to exploitation. Though sscan itself does not attempt to exploit vulnerabilities, it can be configured to automatically execute malicious scripts to exploit vulnerabilities. Watch your logs for port scanning.

Denial of Service Attacks
There is a great increase in the number and variety of denial of service attacks in recent years. A well-known one is the *smurf attack*. Basically, a large amount of ICMP echo (ping) traffic is sent to a host or hosts, all of it having a spoofed source address of a victim. On a multi-access broadcast network, there could be hundreds of machines replying to each packet. In the common scenario, users with Internet access through a slow link will work hard to gain access to a high-powered machine located on a high-speed link, install the various utilities used to attack other hosts, and then launch the attack from this host.
For more information about the smurf attack, see **http://users.quadrunner.com/chuegen/smurf.txt**.

Conclusion

Because Linux supports so many avenues of networking, care should be taken to secure your Linux server. The general rule of thumb is "Only turn on the services you need". Edit down `/etc/inetd.conf`, `rc.inet2` and `/etc/rc.d/rcN.d`. Keep up with the security patches. Use good password policies. Most of the recent Linux distributions include 'passwd' programs that do not allow you to set an easily guessed password. Make sure your passwd program is up to date and has these features. Check your system's logs daily for abnormal activities like port scanning. Become familiar with the processes that normally run on your system and check regularly for unusual processes (beware of processes with names that might be very close to regularly running tasks). Scan your systems for unusual or suspicious files or directories. For example, filenames that start with '.', directories named '...', and unusual device names like '/dev/ttypx'. Use SSH instead of telnet and FTP for more secure communication.

There are many web sites and mailing lists on UNIX Security in general and Linux security in particular. It is important to keep current with the security issues happening around the Internet; this might include becoming familiar with the latest tools. Here are a few useful sites:

UNIX Configuration Guidelines
ftp://info.cert.org/pub/tech_tips/UNIX_configuration_guidelines

Security Tools
ftp://info.cert.org/pub/tech_tips/security_tools

Bugtraq
http://www.mit.edu:8008/menelaus/bt/

References

1
Kevin Fenzi (kevin@scrye.com) & Dave Wreski (dave@nic.com). ***Linux Security HOWTO*** v0.9.11, 1 May 1998.

2
Matt Welsh, Phil Hughes, David Bandel, Boris Beletsky, Sean Dreilinger, Robert Kiesling, Evan Liebovitch, Henry Pierce. *Linux Installation and Getting Started* Red Hat Version 3.2, 20 Feb 1998.

3
Terry Dawson, VK2KTJ, Alessandro Rubini (maintainer),alessandro.rubini@linux.it.

Linux NET-3-HOWTO, Linux Networking. v1.3, 1 April 1998.

4

Wietse Venema *TCP Wrapper: Network Monitoring, Access Control and Booby Traps.* USENIX Proceedings, UNIX Security Symposium III, September 1992.

5

Maintained by Peter Baer Galvin *The Solaris Security FAQ* SunWorld, URL: **http://www.sunworld.com/common/ security-faq.html**, Last modified: Thursday, April 01, 1999.

Biography

Wei-Mei Shyr worked as a system administrator for the Unix Support Group at the Department of Information Technology Services, University of Western Ontario.

Brian Borowski is a network administrator who supports a wide range of network equipment at the University of Western Ontario.