Table Of Contents

nixCraft: Linux Tips, Hacks, Tutorials, And Ideas In Blog Format
http://www.cyberciti.biz/

Explain Linux / UNIX TCP Wrappers / Find Out If Program Compiled With TCP Wrappers

Posted by Vivek Gite <vivek@nixcraft.com>

[1]

What are TCP Wrappers? How do I find out if program / server / service is compile with TCP Wrappers? What are the advantages and disadvantages of TCP Wrappers over Firewalls like netfilter or pf? How do I protect my Mac OS X or Sun Solaris or Linux workstation using TCP Wrappers?


[2]

Almost all BSD / UNIX / Linux like operating systems are compiled with TCP Wrappers support. For e.g. Solaris 9, various Linux / *BSD distributions, and Mac OS X have TCP Wrappers configured to run out-of-the-box. It is a library which provides simple access control and standardized logging for supported applications which accept connections over a network.

TCP Wrapper is a host-based Networking ACL system, used to filter network access to Internet. TCP wrappers was original written to monitor and stop cracking activities on the UNIX workstation in 90s. It was best solution in 90s to protect the UNIX workstations over the Internet. However it has few disadvantages:

1. All UNIX apps must be compiled with the libwrap library.
2. The wrappers do not work with RPC services over TCP.
3. The user name lookup feature of TCP Wrappers uses identd to identify the username of the remote host. By default, this feature is disabled, as identd may appear hung when there are large number of TCP connections.

However, it has one strong advantage over firewall. It works on the application layer. It can filter requests when encryption is used. Basically, you need to use both host based and network based security. Common services such as pop3, ftp, sshd, telnet, r-services are supported by TCP Wrappers.

TCPD Benefits

1. **Logging** - Connections that are monitored by tcpd are reported through the syslog facility.
2. **Access Control** - tcpd supports a simple form of access control that is based on pattern matching. You can evern hook the execution of shell commands / script when a pattern matches.
3. **Host Name Verification** - tcpd verifies the client host name that is returned by the address->name DNS server by looking at the host name and address that are returned by the name->address DNS server.
4. **Spoofing Protection**

How do I Find Out If Program Is Compiled With TCP Wrappers Or Not?

To determine whether a given executable daemon /path/to/daemon supports TCP Wrapper, check the man page, or ennter:

```
$ ldd /path/to/daemon | grep libwrap.so
```

If this command returns any output, then the daemon probably supports TCP Wrapper. In this example, find out of if sshd supports tcp wrappers on not, enter:

```
$ whereis sshd
```

Sample Output:

```
sshd: /usr/sbin/sshd /usr/share/man/man8/sshd.8.gz
```

```
$ ldd /usr/sbin/sshd | grep libwrap.so
```

Sample Output:

```
libwrap.so.0 => /lib64/libwrap.so.0 (0x00002b759b381000)
```

ldd is used to see if libwrap.so is a dependency or not. An alternative to TCP Wrapper support is packet filtering using iptables.

Important Files

- **tcpd** - access control facility for internet services.
- **/etc/hosts.allow** - This file describes the names of the hosts which are allowed to use the local INET services, as decided by the /usr/sbin/tcpd server.
- **/etc/hosts.deny** - This file describes the names of the hosts which are NOT allowed to use the local INET services, as decided by the /usr/sbin/tcpd server.
- If the same client / user / ip is listed in both hosts.allow and hosts.deny, then hosts.allow takes precedence and access is permitted. If the client is listed in hosts.allow, then is access permitted. If the client is listed in hosts.deny, then access is denied.
- **tcpdchk** and **tcpdmatch** - test programs for tcpd

Syntax (format) Of Host Access Control Files

Both /etc/hosts.allow and /etc/hosts.deny uses the following format:

```
daemon_list : client_list [ : shell_command ]
```

Where,

- **daemon_list** - a list of one or more daemon process names.
- **client_list** - a list of one or more host names, host addresses, patterns or wildcards that will be matched against the client host name or address.

WildCards

The access control language supports explicit wildcards (quoting from the man page):

```
       ALL    The universal wildcard, always matches.

       LOCAL  Matches any host whose name does not contain a dot character.

       UNKNOWN
       Matches  any  user  whose  name is unknown, and matches any host
       whose name or address are unknown.  This pattern should be  used
       with  care:  host names may be unavailable due to temporary name
       server problems. A network address will be unavailable when  the
       software cannot figure out what type of network it is talking
       to.

       KNOWN  Matches any user whose name is known, and matches any host whose
       name  and  address  are  known. This pattern should be used with
       care: host names may be unavailable due to temporary name server
       problems.   A network address will be unavailable when the soft-
       ware cannot figure out what type of network it is talking to.

       PARANOID
       Matches any host whose name does not match  its  address.   When
       tcpd  is built with -DPARANOID (default mode), it drops requests
       from such clients even before  looking  at  the  access  control
       tables. Build  without -DPARANOID  when you want more control
       over such requests.
```

TCPD Configuration Examples

Set default policy to to deny access. Only explicitly authorized hosts are permitted to access. Update /etc/hosts.deny as follows:

```
 # The default policy (no access) is implemented with a trivial deny file
ALL: ALL
```

Above will denies all service to all hosts, unless they are permitted access by entries in the allow file. For example, allow access as follows via /etc/hosts.allow:

```
  ALL: LOCAL @devels
  ALL: .nixcraft.net.in EXCEPT boobytrap.nixcraft.net.in
```

Log and deny access (booby traps) - we do not allow connections from crackers.com:

```
ALL : .crackers.com \
    : spawn (/bin/echo %a from %h attempted to access %d >> \
      /var/log/connections.log) \
    : deny
```

A Typical UNIX Example

Allow access to various service inside LAN only via /etc/hosts.allow:

```
popd : 192.168.1.200 192.168.1.104
imapd : 192.168.1.0/255.255.255.0
sendmail : 192.168.1.0/255.255.255.0
sshd : 192.168.1.2 172.16.23.12
```

Deny everything via /etc/hosts.deny:

```
 ALL : ALL
```

Reject All Connections

Restrict all connections to non-public services to localhost only. Suppose sshd and ftpd are the names of service which must be accessed remotely. Edit /etc/hosts.allow. Add the following lines:

```
sshd ,ftpd : ALL
ALL: localhost
```

Save and close the file. Edit /etc/hosts.deny. Add the following line:

```
ALL: ALL
```

Default Log Files

TCP Wrappers will do all its logging via syslog according to your /etc/syslog.conf file. The following table lists the standard locations where messages from TCP Wrappers will appear:

1. AIX - /var/adm/messages
2. HP-UX - /usr/spool/mqueue/syslog
3. Linux - /var/log/messages
4. FreeBSD / OpenBSD / NetBSD - /var/log/messages
5. Mac OS X - /var/log/system.log
6. Solaris - /var/log/syslog

Use the following command to view logs:

```
# tail -f /path/to/log/file
# grep 'ip' /path/to/log/file
# egrep -i 'ip|hostname' /path/to/log/file
```

How Do I Predicts How The Tcp Wrapper Would Handle a Specific Request For Service?

Use tcpdmatch command. predict how tcpd would handle a sshd request from the local system:

```
tcpdmatch sshd localhost
```

The same request, pretending that hostname lookup failed:

```
tcpdmatch sshd 192.168.1.5
```

To predict what tcpd would do when the client name does not match the client address:

```
tcpdmatch sshd paranoid
```

Replace sshd with in.telnetd, or ftpd and so on. You can use all daemon names specified in inetd.conf or xinetd.conf file.

How do I Examines My TCP Wrapper Config File?

Use tcpdchk command toexamines your tcp wrapper configuration and reports all potential and real problems it can find.

```
tcpdchk
tcpdchk -v
```

A Note About TCP Wrappers and Firewall

- You need to use both (firewall and tcpd) to fight against crackers.
- TCP Wrappers are most commonly employed to match against IP addresses and host level protection.
- Never configure TCP Wrappers on firewall host.
- Put TCP Wrappers on all UNIX / Linux / BSD workstations .
- Do not use NIS (YP) netgroups in TCP Wrappers rules.
- Put TCP Wrappers behind a firewall systems as TCP Wrappers is no substitute for netfilter or pf firewall.
- TCP Wrappers does provide increased security as firewall cannot examine encrypted connections (read as packets).

References:

1. man pages - tcpd(8), tcpdchk(8), tcpdmatch(8), and hosts_access(5).
2. TCP Wrappers [3] Release Notes from HP-UX
3. TCP Wrappers [4] configuration under FreeBSD operating systems.
4. What are TCP Wrappers [5] in Red Hat Enterprise Linux / CentOS Linux? What are some of the best practices and known issues?