LPIC-1 102-400 – Lesson 19

110.2 Setup host security

The *inetd* and *xinetd* superservers

- The **inetd** and **xinetd** daemons are services that listen to **TCP** and **UDP** ports and they start different applications based on their configuration e.g. ssh, ftp, http etc
- The advantage of running ssh, telnet, ftp, tftp, through a superserver, instead of their own autonomous daemon, is having only one service listening to many ports, so we are saving system resources
- Using a superserver you can also convert applications that do not have their own daemon (e.g tftp, cvs) to services
- The disadvantage of using a superserver is the latency caused when different ports all called and so different applications are started at the same time. Thus superservers are not recommended on high network traffic systems

The *inetd* and *xinetd* superservers

- A superserver listens to a port and assigns that to a service or application when an external connection is initiated

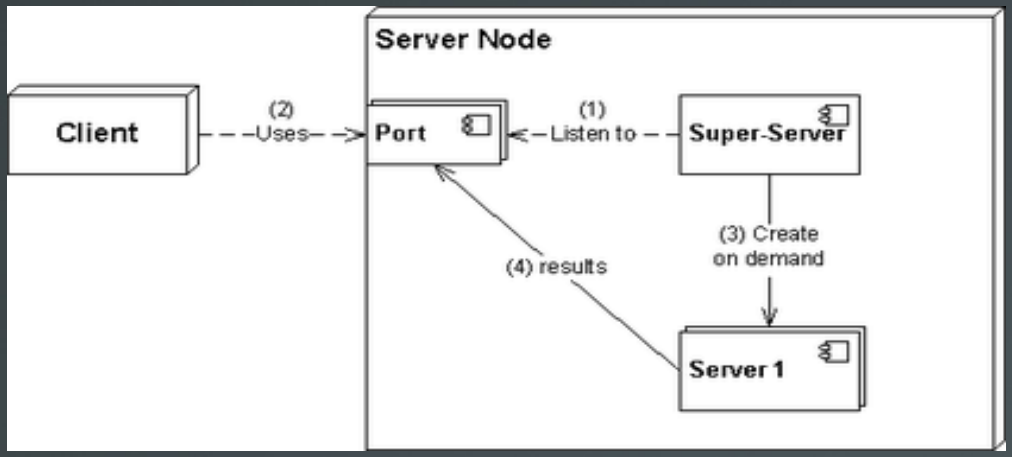

Image by
User:UlrichAAB
Wikipedia

- A superserver can serve several services simultaneously:

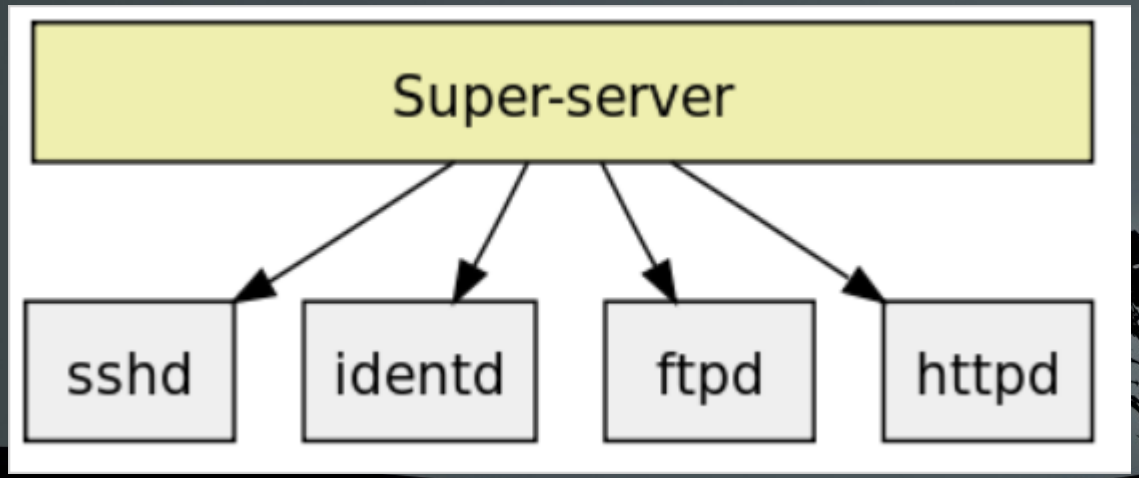

Image by
User:Frap
Wikipedia

The *inetd* superserver

- The **inetd** superserver has been traditionally one of the first superservers in existence
- **`# apt-get install inetutils-inetd | openbsd-inetd`**
  `# installation in Debian`
- It’s main configuration file is **/etc/inetd.conf** and all files under **/etc/inetd.d/**. The format of the configuration file looks like:

```
# <service_name> <sock_type> <proto> <flags> <user> <server_path> <args>
```

The */etc/inetd.conf* configuration file

- **service_name**: has to be a service name from the **/etc/services** file
- **socket_type**: it can have values like **stream**, **dgram**, **raw** etc. For TCP we use **stream** and for UDP we use **dgram**
- **proto**: some protocol from the **/etc/protocols** file. Usually TCP or UDP
- **flags**: its values can be **wait** or **nowait**. **wait** is used in case where **inetd** must wait for the calling service to be initialized before occupying the listening port
- **user**: the user under which the service will run. **root** should be avoid for security reasons
- **server_path**: the path of the called service/application
- **args**: arguments that need to pass to the calling service. **internal** is used for processes internal to **inetd**

The */etc/inetd.conf* configuration file

- An example of configured services in **/etc/inetd.conf**:

```
#discard        stream  tcp     nowait  root    internal
#discard        dgram   udp     wait    root    internal
#daytime        stream  tcp     nowait  root    internal
#time           stream  tcp     nowait  root    internal
talk            dgram   udp     wait    root    /usr/sbin/talkd
telnet          stream  tcp     wait    root    /usr/sbin/telnetd
```

- After starting **inetd**:
- `# netstat -lnptu | grep inet`

```
tcp   0    0.0.0.0 :23     0.0.0.0:*  LISTEN  13463/inetutils-ine
udp   0    0 0.0.0.0:517   0.0.0.0:*          13463/inetutils-ine
```

- `# /etc/init.d/inetutils-inetd restart` # restart **inetutils-inetd**
- `# /etc/init.d/openbsd-inetd restart` # restart **openbsd-inetd**

The *xinetd* superserver

- The **xinetd** superserver is a more recent system and provides more features.
- Default in RedHat systems
- `# apt-get install xinetd # install xinetd in Debian`
- Its main configuration file is **/etc/xinetd.conf** and the custom configuration filed are under **/etc/xinetd.d/**. The format of the configuration file looks like:

```
service rsync{
    disable no # yes to disable
    socket_type = stream # dgram, raw are other options
    wait = no # yes to enable
    user = root # the user that owns the calling service
    server = /usr/bin/rsync # path of the calling service
    server_args = --daemon# arguments of the calling service
}
```

Enabling/disabling services in *xinetd*

- If we change the **disable** parameter in **/etc/xinetd.d/rsync** to **yes** the **rsync** service will be disabled at the next **xinetd** restart
- **# /etc/rc.d/init.d/xinetd restart** # restart in RedHat
- **# /etc/init.d/xinetd restart** # restart in Debian
- Verify if everything is OK:

```
# netstat -lnptu | grep inet
tcp   0    0.0.0.0:873     0.0.0.0:*  LISTEN  24950/xinetd
```

The */etc/passwd* and */etc/shadow* files

- Traditionally the user passwords were stored in the **/etc/passwd** file. This turned out to be a serious security issue because even though they were encrypted (hashed), they could be read by all users because of the mandatory **644** permissions. This happens because this file is supposed to be read by all users/services
- To solve this problem the **shadow passwords** system was created. In the password field of **/etc/passwd** an "**x**" is placed and the actual, salted and hashed, password goes to the **/etc/shadow** file
- The **/etc/shadow** file is not read by others, just the **root** user.
- The one-way hashing algorithms used typically on modern systems are SHA256 and SHA512

Security in */etc/inittab*

- Several Linux security guides recommend disabling **Ctrl-Alt-Del** and making password mandatory even for **single user mode**. These can be adjusted in **/etc/inittab**
- **~~:S:wait:/sbin/sulogin** # prompt for password even on single user mode. This should be matched with a **boot loader** password
- **# ca::ctrlaltdel:/sbin/shutdown -r now** # this **inittab** line, allows the system restart bu pressing **Ctrl-Alt-Del**. It can be disabled by adding a "#" to comment it out. Or delete the line completely

Detect and disable unnecessary services

- Using the **netstat -lnptu**, **ss -lnptu**, or **lsof -i** commands we can detect listening ports and the services that occupy them. If there are any unnecessary services running, these should be disabled
- For disabling **System V init** services the symbolic links in the **rc[1-6].d** directories should be renamed with a **K** as the first character e.g.:
  `/etc/rc3.d/S19postgresql -> ../init.d/postgresql` to
  `/etc/rc3.d/K19postgresql -> ../init.d/postgresql`
  This is also possible with the **chkconfig** command in RedHat and the **update.rc-d** command in Debian
- Services already running should be stopped:
  `# /etc/init.d/postgresql stop` or
  `# service postgresql stop`
- Services running under the **inetd** or **xinetd** superservers should be disabled from their configuration files and the superserver restarted
- For **systemd** systems we can use these commands:
  `# systemctl disable postgresql # disable service`
  `# systemctl stop postgresql # stop service`

Disable login by normal users with */etc/nologin*

- Sometimes when a system need to be in maintenance mode, the system administrator wants to prevent users from logging into the system
- In this case the system administrator can create the **/etc/nologin** file. When this file is present, users are not allowed to login either locally or remotely and the contents of the **nologin** file will be showed to those who try
- `# echo "Offline for maintenance" > /etc/nologin # prevent all logins except root and display an explanation`
- `# rm /etc/nologin # don't forget to delete it after maintenance works are completed`

Restrict network access with *TCP Wrapper*

- **TCP Wrapper** is an Access Control Lists (ACL) system which can restrict network connection to serviced that support it
- Service that support it have been compiled against the **libwrap** library. This can be verified with **ldd**:

```
# ldd /usr/sbin/sshd | grep libwrap
libwrap.so.0 => /lib/x86_64-linux-gnu/libwrap.so.0
(0x00007f2262807000)
```

- **TCP Wrapper** uses the **/etc/hosts.allow** and **/etc/hosts.deny** files to set networks, hosts and services where access should be allowed or denied
- These filed have an effect only to applications that use the **libwrap** library

The */etc/hosts.allow* and */etc/hosts.deny* files

- The priority by which the **/etc/hosts.allow** and **/etc/hosts.deny** files operate are as follows:
  - If there is a network, domain, IP or hostname in **/etc/hosts.allow**, access is permitted to it
  - If there is a network, domain, IP or hostname in **/etc/hosts.deny**, access to it is denied
  - For host that do not exist in either file, access is allowed
- If we want to prevent access to all and allow access only to some hosts, we should set **ALL: ALL** in **hosts.deny** and add allowed systems and networks in **hosts.allow**

The */etc/hosts.allow* and */etc/hosts.deny* files

- **# cat /ets/hosts.deny**
  ALL: **ALL** # deny access to all services from everywhere
- **# cat /etc/hosts.allow**

  sshd: 10.0.1.0/24 EXCEPT 10.0.1.64/26 # allow access to **sshd** for the **10.0.1.0/24** network with exception to the **10.0.1.64/26** subnetwork

  ALL EXCEPT tftpd: .example.com EXCEPT vpn.example.com # allow access to all services, except tftpd,from the **example.com** (take note the initial dot!) except the **vpn.example.com** node

  mysqld: LOCAL, @netgroup # allow local access and access from the **netgroup** group, to **mysqld**

  telnetd: 10.0.1.0/24, .example.com EXCEPT 10.0.1.23 # allow access to the **telnetd** service from the **10.0.1.0/24** network and the **example.com** domain but deny access to the **10.0.1.23** IP Address

License