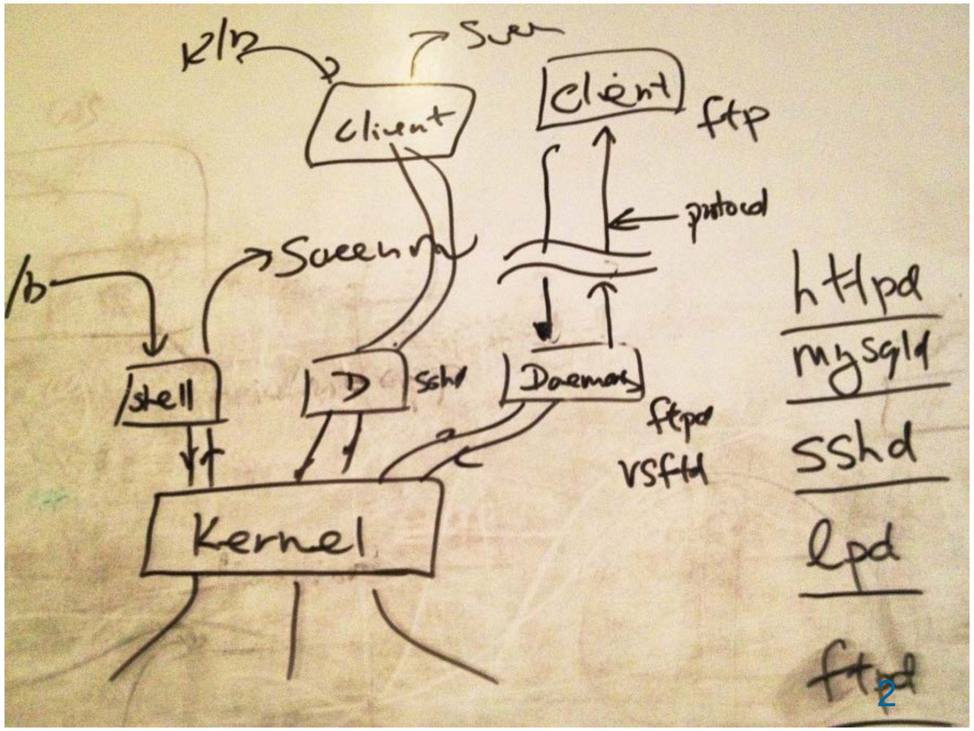
COMP 175

System Administration and Security

DAEMONS

Objectives

- Understand Linux daemon process
- Understand Linux run levels
- Understand the purpose of crontab
  - How to configure it
- Understand the inetd daemon



Daemons

- A computer program that runs in the background
- Usually initiated as background processes
- Become daemons by forking a child process
  - Having the parent process then exit
  - init 'adopts' the child process
- Parent process often (not always) the init process
- Typically daemons have names that end with "d"
  - syslogd - the system log daemon
  - sshd - handles incoming SSH connections
  - httpd – web server daemon

Daemon Terminology

- Name came out of a 1963 MIT CS & AI project
- Described background processes which worked tirelessly to perform system chores
- A reference to Maxwell's demon

MIT Computer Science and Artificial Intelligence Laboratory

View From Upper Floor Window

Night View

Daemon Attributes

- Not associated with any terminal
  - Output doesn't end up in another session
- Terminal generated signals (^C) aren't received
- Most servers run as a daemon process
- No terminal - must use something else:
  - file system
  - central logging facility
- Syslog is often used - provides central repository for system logging
- syslogd daemon provides system logging services to "clients"

Essential Daemons

- init: initialization - first process that runs (PID 1)
  - started by kernel
- crond: the general system scheduler
  - time-based job scheduler
- inetd: the super-server daemon (master)
- devfsd: device file system (devfs)

- Want a minimalist startup for maintenance work?
- *Runlevel* (see next slide)

runlevel

- Describes the state of the machine
- Characterized by the processes run
- `runlevel` command shows current level
- slackware
  - 0 = halt
  - 1 = single user mode
  - 2 = unused (configured same as runlevel 3)
  - 3 = multiuser mode (default runlevel)
  - 4 = X11 with KDM/GDM/XDM (session managers)
  - 5 = unused (configured same as runlevel 3)
  - 6 = reboot

runlevel

- Ubuntu, Debian

  look in /etc/init.d   then /etc/rc#.d

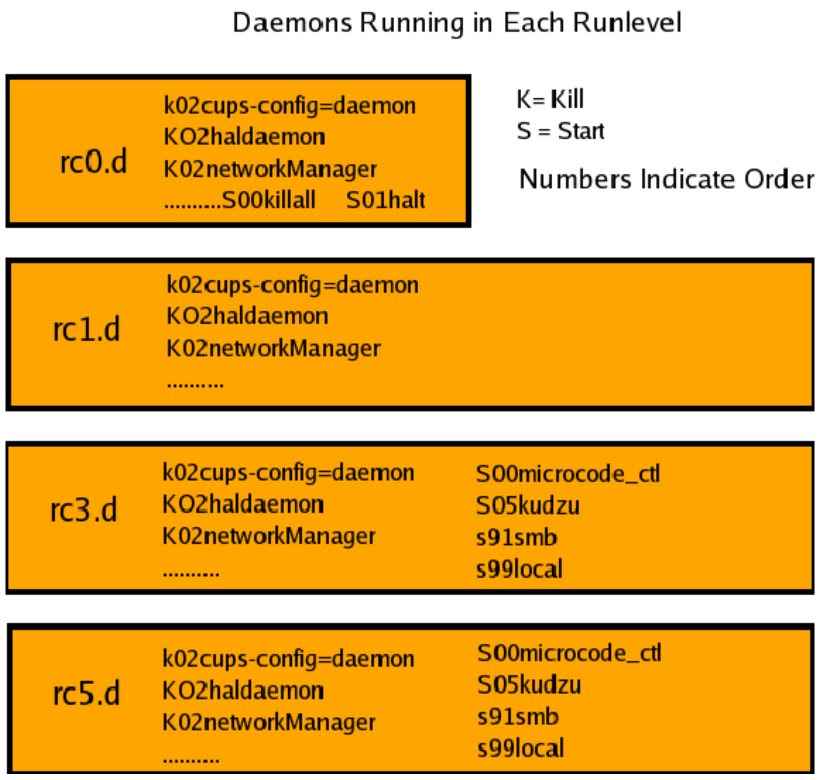
  or install BootUpManager GUI tool   *bleah*

  - 0 System Halt
  - 1 Single user - no daemons started
  - 2 Full multi-user mode (Default)
  - 3-5 Same as 2
  - 6 System Reboot

  - Jammy Desktop
  - $ runlevel   answer is?
  - $ man runlevel   what does it mean?

telinit command

- primary command used to change run levels
- telinit n
- * shutdown better than telinit 0


Daemons Running in Each Runlevel

K= Kill
S = Start

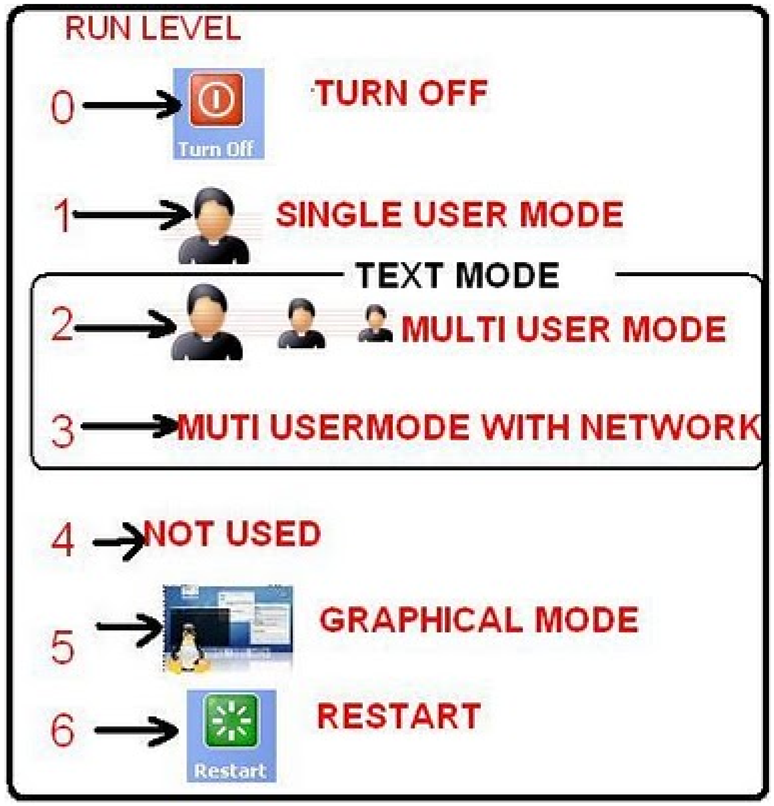
Numbers Indicate Order

rc0.d
k02cups-config=daemon
KO2haldaemon
K02networkManager
..........S00killall S01halt

rc1.d
k02cups-config=daemon
KO2haldaemon
K02networkManager
....

rc3.d
k02cups-config=daemon
KO2haldaemon
K02networkManager
....
S00microcode_ctl
S05kudzu
s91smb
s99local

rc5.d
k02cups-config=daemon
KO2haldaemon
K02networkManager
....
S00microcode_ctl
S05kudzu
s91smb
s99local



Run Levels

- Startup Directories
  - /etc/rc0.d Run level 0
  - /etc/rc1.d Run level 1
  - /etc/rc2.d Run level 2
  - /etc/rc3.d Run level 3
  - /etc/rc4.d Run level 4
  - /etc/rc5.d Run level 5
  - /etc/rc6.d Run level 6

Boot Time Services

- Start/Stop boot time services
- All services startup scripts which start with S will start at boot time
- All startup scripts which start with K will not start at boot time.
  - The number after S or K is the priority
- To start, stop or restart a service from command line, use:

```
service <service name> start/stop/restart
```

Slackware

init.d/
rc.0@
rc.4*
rc.6*
rc.K*
rc.M*
rc.S*
rc.bind
rc.dhcpd
rc.firewall

rc.httpd
rc.inetd
rc.ntpd
rc.samba
rc.sendmail
rc.sshd
rc.syslog

SystemV vs. BSD

CRONTAB

cron

- Cron enables users to schedule jobs (commands or shell scripts) to run at preset times or dates
  - Daily cleanup
  - Filesystem management
- Driven by a /etc/crontab (cron table)
- /etc/cron.hourly
- /etc/cron.daily
- /etc/cron.weekly
- /etc/cron.monthly
- Users can have their own individual crontab files

crontab commands

- Lists the current cron jobs

```
 crontab –l
```

- Edit current crontab file
  - Add/remove/edit crontab tasks

```
crontab –e          # default editor: vi
```

- Remove the crontab file

```
 crontab –r
```

- Display the last time you edited your crontab file

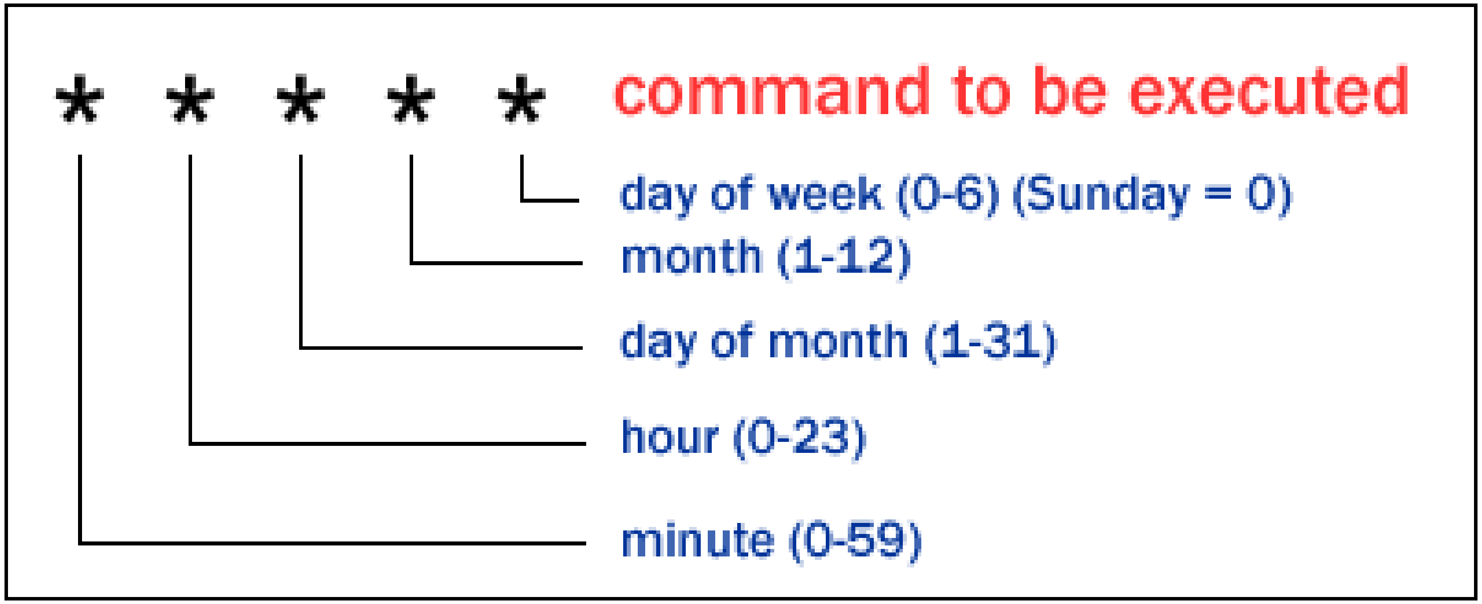
```
 crontab -v
```

crontab

To schedule a script

```
30 13 * * * /home/user/run-me.sh >/dev/null 2>&1
```

- Run when clock minute is 30 and hour is 13
- Suppress default email at job completion


* * * * * command to be executed
day of week (0-6) (Sunday = 0)
month (1-12)
day of month (1-31)
hour (0-23)
minute (0-59)

Crontab entries

- Run the script every day at 12:00, 14:00 and 16:00

```
0 12,14,16 * * * /home/user/run-me.sh >/dev/null 2>&1
```

- Run the script every Sunday at 13:30

```
30 13 * * 0 /home/user/run-me.sh >/dev/null 2>&1
```

- Run the script every Saturday at 12:00, 14:00 and 16:00

```
0 12,14,16 * * 6 /home/user/run-me.sh >/dev/null 2>&1
```

- Run the script on the 1st, 15th and 20th of every month

```
0 0 1,15,20 * * /home/user/run-me.sh >/dev/null 2>&1
```

Crontab Entries

- Run script every last Saturday of month at 8pm

```
00 20 25-31 1,3,5,7,8,10,12 6 run-me.sh
00 20 24-30 4,6,9,11 6 run-me.sh
00 20 22-29 2 6 run-me.sh
```

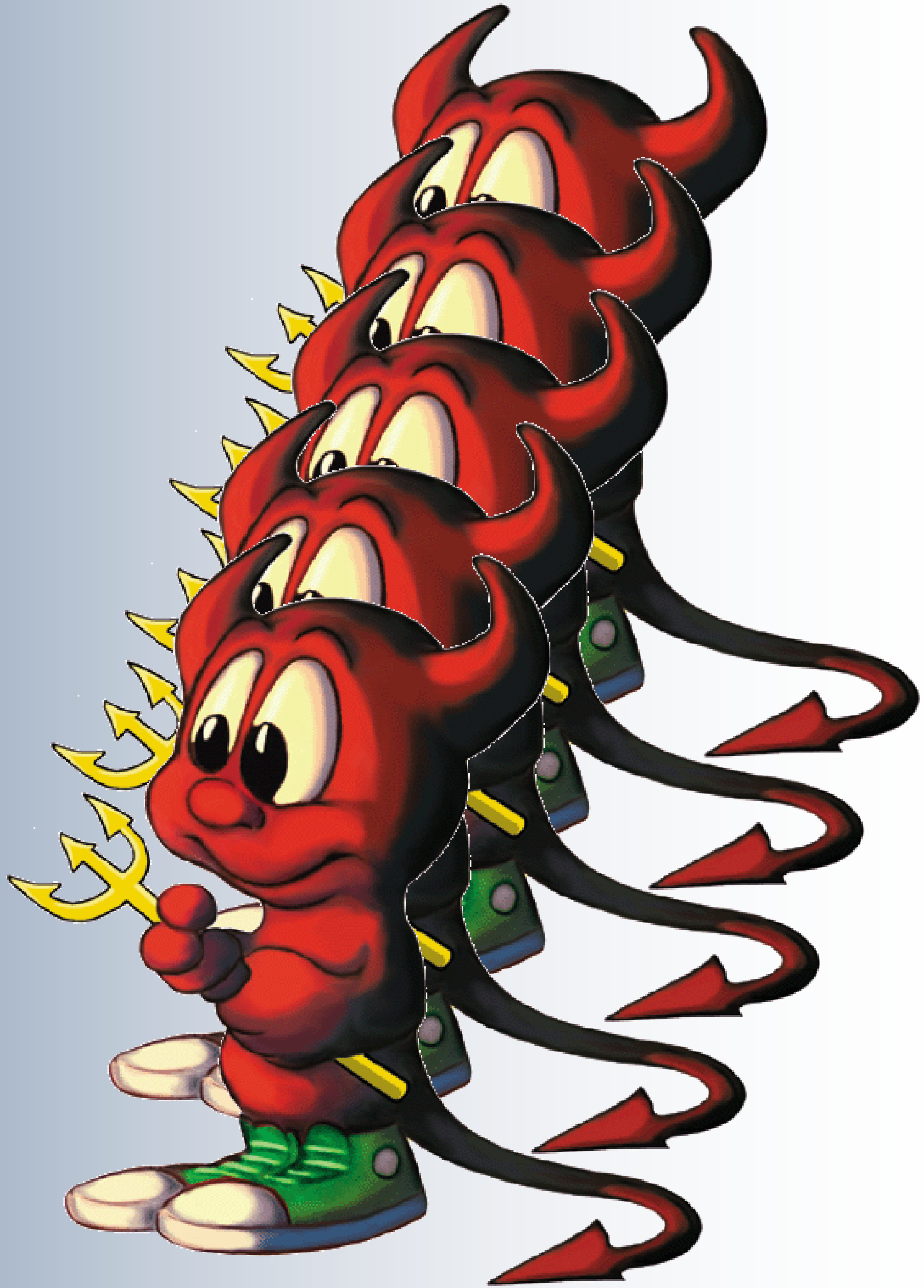
```
First line is for 31-day months
Second lines is for 30-day months
Third line runs twice in 2020  ȫ
```

Ubuntu Desktop

- crontab assumes system is running continuously
  - If system is off, jobs will not be run
  - Not optimal for desktops, laptops

- So..... anacron
- anacron performs periodic command scheduling
- anacron moves tasks to when the system is on
- Only sysadmin can configure anacron
- anacron can only run tasks once a day

INETD



Too Many Daemons?

- There can be many servers running as daemons - and idle most of the time
- Much of the startup code is the same for these servers
- Most of the servers are asleep most of the time, but use up space in the process table

- *The inetd super-server solution*

inetd

- `inetd`
  - Executes the startup code required by a bunch of servers (note: servers)
  - Waits for incoming requests destined for the same bunch of servers
  - When a request arrives - starts up the right server and gives it the request

inetd

- This single daemon creates multiple sockets and waits for (multiple) incoming requests.
- `inetd` typically uses `select` to watch multiple sockets for input.
- When a request arrives, `inetd` will fork and the child process handles the client
- The child process closes all unnecessary sockets
- The child exec’s the real server program, which handles the request and exits

/etc/inetd.conf

- `inetd` reads a configuration file that lists all the services it should handle
  - `/etc/inetd.conf`
- `inetd` creates a socket for each listed service, and adds the socket to a list given to select().
- `inetd` reads files to get the name, aliases, login name process should run as, pathname, arguments, port and protocol to use for each service
  - `/etc/services`
  - `/etc/protocol`

Example inetd.conf

```
service-name socket-type protocol wait-flag login-name server-program server-program-argument
# comments start with #
echo      stream  tcp  nowait  root  internal
echo      dgram   udp  wait    root  internal
chargen   stream  tcp  nowait  root  internal
chargen   dgram   udp  wait    root  internal
ftp       stream  tcp  nowait  root  /usr/sbin/ftpd ftpd -l
telnet    stream  tcp  nowait  root  /usr/sbin/telnetd telnetd
finger    stream  tcp  nowait  root  /usr/sbin/fingerd fingerd
# Authentication
auth      stream  tcp  nowait  nobody  /usr/sbin/in.identd
   in.identd -l -e -o
# TFTP
tftp     dgram   udp   wait    root    /usr/sbin/tftpd tftpd -s
   /tftpboot
```

inetd.conf

- Specifying WAIT means that `inetd` should not look for new clients for the service until the child (the real server) has terminated.
- TCP servers usually specify nowait - this means inetd can start multiple copies of the TCP server program - providing concurrency!
- Most UDP services run with `inetd` told to wait until the child server has died
- Some UDP servers hang out for a while, handling multiple clients before exiting
- `inetd` told to wait will ignore the socket until the UDP server exits

Remember

- daemons are processes running in background
- init – first process to run (PID 1)
- Runlevel describes state of OS
  - Single user, multiuser, GUI, reboot
- Command to change runlevels – telinit [n]
- crontab – is a job scheduler
- Most Servers run as daemons
- Typically end in 'd'
  - named, dhcpd, httpd, ntpd, syslogd
- Many run out of inetd
  - When request arrives, inetd will fork
  - Child process handles the request