KOÇ
ÜNİVERSİTESİ

COMP304 PS 3

Project 1

Ismayil Ismayilov
iismayilov21@ku.edu.tr

Basic Commands

- Command line inputs should be interpreted as program invocation
- Fork and Exec child processes
- Use execv()

Basic Commands

- Fork a child process
- Execute command in child process with **execv()**
  - Do your own path resolving
- In parent
  - If command is in background => wait for child
  - Otherwise => terminate
- Project 1 - Unix Shell Part-I from Chapter 3 of book might help

Custom Commands

- filesearch
- cdh
- take
- joker
- Your awesome command

Custom Commands - filesearch

- 1 arguments
  - Keyword to be used for matching filenames
- Options
  - -r => Recursive search => Current directories and children directories
  - -o => Open all matched files

Custom Commands - cdh

- Allows you to quickly navigate between recently visited directories
- Use letter or numeric index to switch to directory
- Keep track of 10 directories
- Persist across shell sessions
- No need to handle duplicate

Custom Commands - take

- Allows you to create a directory and immediately switch to it
- Should create intermediate directories along the way (if they do not exist)

Custom Commands - joker

- Command outputs random joke to screen every 15 minutes
- Random joke => https://icanhazdadjoke.com/
- Explore *crontab*, *notify-send*, *curl*

Crontab

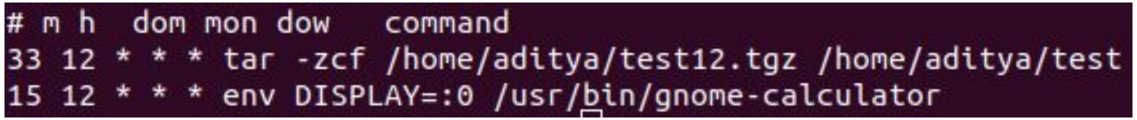
- A program that schedules the execution of Linux commands at specified time periodically.
- Command to edit crontab schedule: crontab -e
- Format of each scheduled command:

```
*    *    *   *    *  Command_to_execute
|    |    |    |   |
|    |    |    |    Day of the Week ( 0 - 6 ) ( Sunday = 0 )
|    |    |    |
|    |    |    Month ( 1 - 12 )
|    |    |
|    |    Day of Month ( 1 - 31 )
|    |
|    Hour ( 0 - 23 )
|
Min ( 0 - 59 )
```

Crontab

- To see the list of scheduled commands: crontab -l

  Example of scheduled commands:



```
# m h  dom mon dow   command
33 12 * * * tar -zcf /home/aditya/test12.tgz /home/aditya/test
15 12 * * * env DISPLAY=:0 /usr/bin/gnome-calculator
```

- To remove all scheduled commands: crontab -r

- To add a new schedule to crontab from command line:

```
crontab -l | { cat; echo "51 13 * * * env DISPLAY=:0 /usr/bin/gnome-calculator"; } | crontab -
```

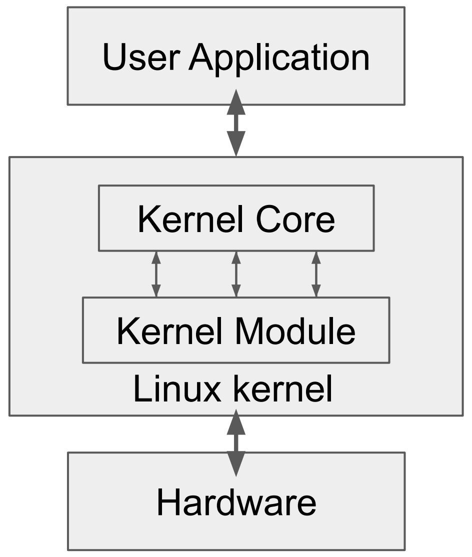
  - “crontab -l” lists all scheduled commands.
  - “cat” prints the list.
  - “echo” prints the quoted command
  - “crontab -” adds the printed command list to the crontab file

Crontab

- Your task in the project: schedule a joke to pop up every 15 minutes
  - To create a pop up message, you can use notify-send program.
  - Example:
    - /usr/bin/notify-send "Don’t forget to sleep"

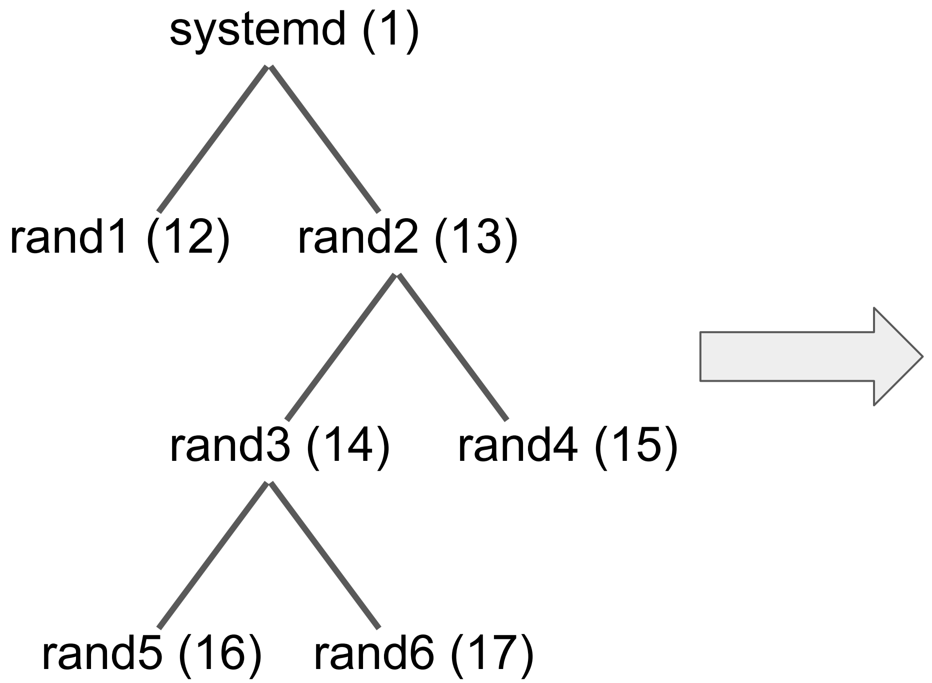
Kernel Module

- Piece of code that extends the functionality of OS kernel
  - It can be loaded and unloaded into the OS kernel without modifying the original kernel source code.
  - No need to reboot the machine.
  - Device drivers are examples of kernel modules.



Kernel Module

- In shellington: pstraverse <PID> <-d or -b>
  - Example 1: pstraverse 13 -d

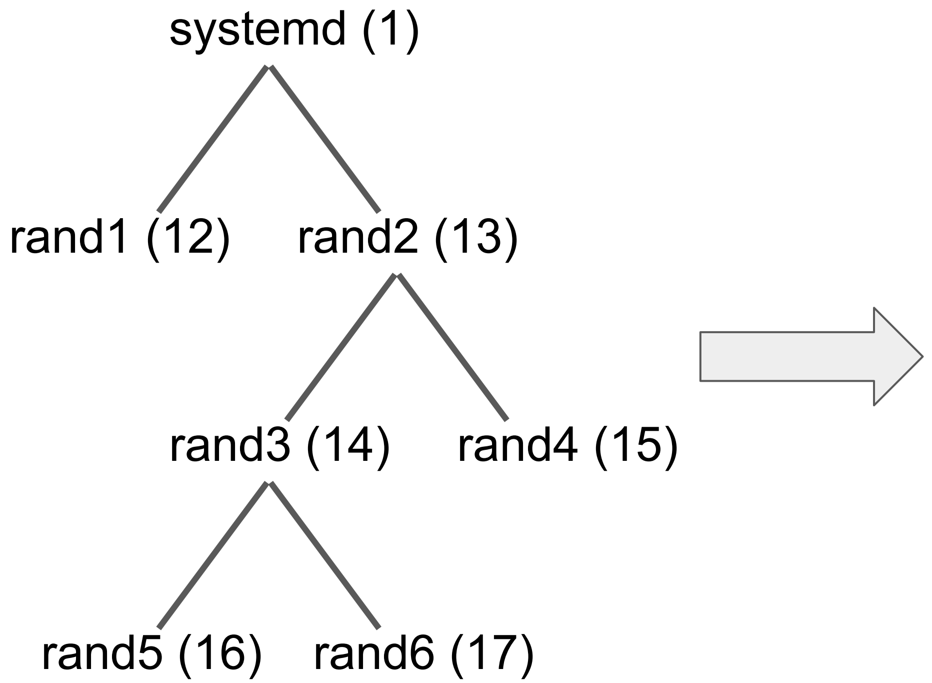


Output when you run dmesg:

```
PID: 13, Name: rand2
PID: 14, Name: rand3
PID: 16, Name: rand5
PID: 17, Name: rand6
PID: 15, Name: rand4
```

Kernel Module

- In shellington: pstraverse <PID> <-d or -b>
  - Example 2: pstraverse 13 -b



Output when you run dmesg:

```
PID: 13, Name: rand2
PID: 14, Name: rand3
PID: 15, Name: rand4
PID: 16, Name: rand5
PID: 17, Name: rand6
```

Necessary Knowledge

- These are the tool sets that you need to implement your kernel module
    - module_param and MODULE_PARM_DESC for Linux kernel module
    - struct file definition in Linux source code
    - struct task_struct definition in Linux source code
    - How to use list_for_each and list_entry functions in Linux kernel
    - find_vpid function
    - ioctl()
    - How to traverse a tree using DFS and BFS approaches

Important Remarks

- Create kernel module and compile
- First invocation loads kernel module. Kernel module should not be loaded again
  - When **pstraverse** is first called, can perform traversal in **init**
  - Otherwise, can use **ioctl**
- Can use ioctl to perform operation on kernel module
- Remove kernel module when shell exits