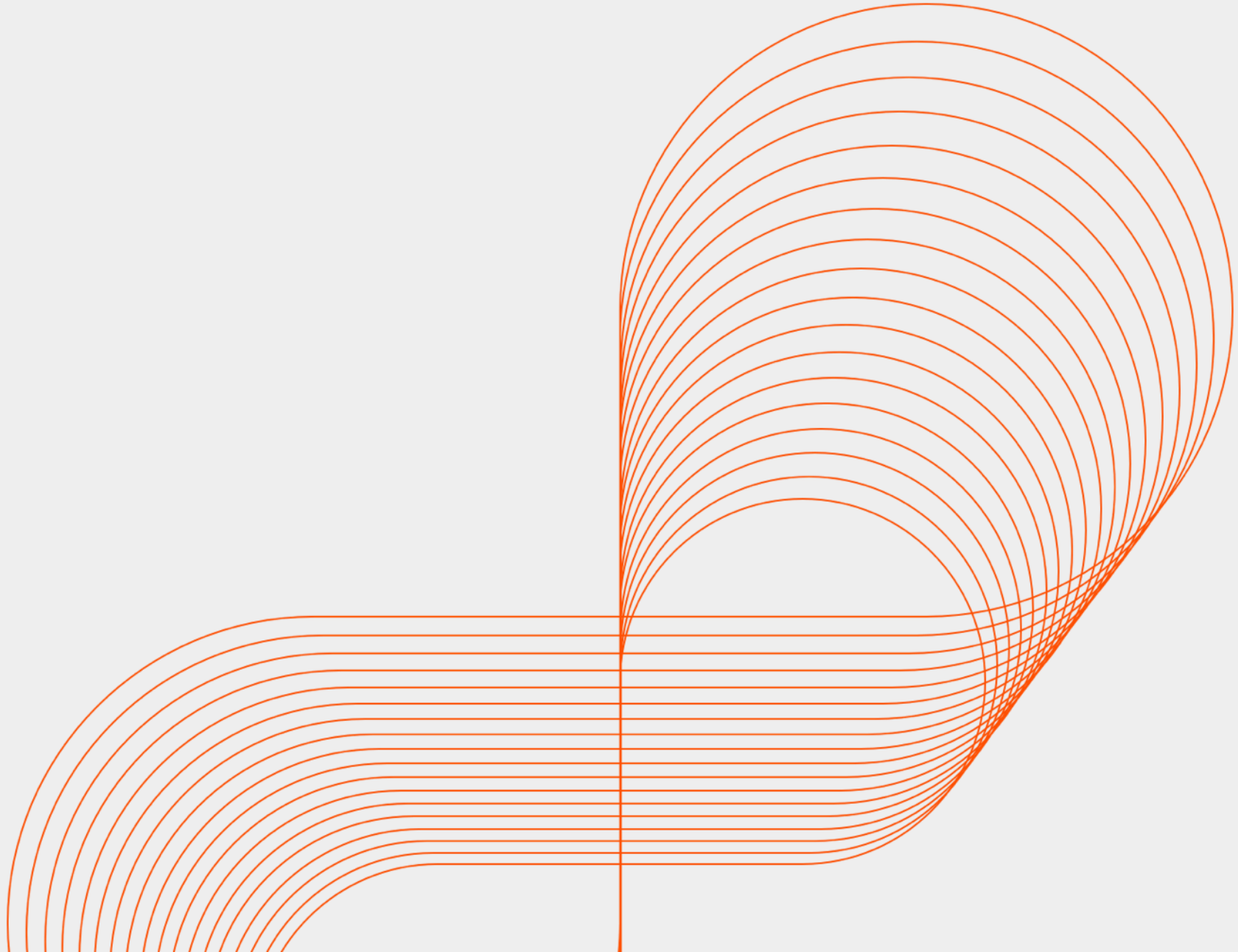
Persistent

UNIX



Processes & Jobs

Process

- A process is a running instance of an application
- In Unix, there are two types of processes
  - one type is associated with user’s login session
  - other type is daemon process
- The first type processes are usually started by the user by running a command or application. Such processes can be controlled by the user by providing input to the process
- The second type of processes run in the background and are not directly controllable by the user. These are usually started early during system initialization

Process Management

Command	Meaning	Command	
at	schedule execution once	**pkill**	send a signal to a process
cron	time-based job scheduler	**pstree**	show the running processes as a tree
kill	terminate a process	**sleep**	suspend program execution
nice	change the priority of a process	**time**	determine duration of execution of a command
pidof	display PID of a process	**top**	list running processes and update frequently
ps	show running processes	**wait**	pause execution and wait for background process to finish

Process Status (ps)

- This command gives the snapshot of all the processes running in the system
- Some common options are
  - -a : show all processes except those not associated with a terminal
  - -e : show all processes
  - -H : show process hierarchy

top Command

- The top command is a useful tool for quickly showing processes sorted by various criteria
- It is an interactive tool that updates frequently and shows information about physical and virtual memory, CPU usage, etc
- This command is usually found in BSD, Linux & Mac OS X systems

Stopping Processes

- You can end a process in several ways:
  - Send a CTRL+C keystroke
  - Use the *kill* command
- Technically, *kill* command does not kill a command but sends a special signal to the process
- The default signal for the *kill* command is *SIGTERM*
- To list the possible signal names, run *kill* with the *-l* switch

Zombie Processes

- After a process finishes execution, the parent process is told via a SIGCHLD signal. Then the parent can do some other task or restart a new child as needed
- Sometimes, the child process entry still remains in the process table after it has finished execution (this is required to allow the parent to read its exit status). Such a process is called as a zombie or defunct process
- This means the child process has died but has not been reaped yet
- To remove zombies from a system, the SIGCHLD signal can be sent to the parent manually, using the kill command
- If the parent process still refuses to reap the zombie, the next step would be to remove the parent process.
- When a process loses its parent, init becomes its new parent. Init periodically executes the *wait()* system call to reap any zombies with init as parent.

The /proc File System

- This is a dynamically generated file system that can be used to retrieve information about processes running on a system
- It contains a directory entry for active process named after the PID and these directories contain files that provide various attributes about the process
- For example, to see information about the cpu, view the file */proc/cpuinfo*

Job Control

- Unix provides the capability to run commands in the background, so you can run multiple programs at a time
- Job control refers to the orchestration of multiple batch jobs
- It can also be used to suspend commands and restart suspended commands
- Some commands might take too long to finish. Such commands can be run in the background as follows:

  makewhatis &
- The & causes the command to run in the background and the shell command prompt becomes usable for other commands

jobs Command

- Most shells have a built-in *jobs* command that can be used to show your running shell jobs
- It shows all the jobs that are running
- You can bring a job in foreground through the *fg* command
- Once the command comes to foreground, the shell prompt does not display until the process is ended, and you cannot enter any other command until then

fg Command

- *fg* command is used to bring a command to foreground
- Before bringing a command to foreground, you need to check its job number through *jobs* command
- *fg* requires the job number of the command that is to be brought in the foreground
- You can suspend a foreground command by pressing Ctrl+Z
- A suspended job can be resumed by *fg* command or *bg* command

bg Command

- *bg* runs the command in the background, so that you can run other commands through the shell prompt
- *bg* also requires the job number of the command that is to be run in the background

Example

```
[root@vmrhelu4 ~]# sleep 100 &
[1] 28124
[root@vmrhelu4 ~]# jobs
[1]+  Running                 sleep 100 &
[root@vmrhelu4 ~]# fg 1
sleep 100

[1]+  Stopped                 sleep 100
[root@vmrhelu4 ~]# fg 1
sleep 100

[1]+  Stopped                 sleep 100
[root@vmrhelu4 ~]# bg 1
[1]+ sleep 100 &
[root@vmrhelu4 ~]# jobs
[1]+  Running                 sleep 100 &
[root@vmrhelu4 ~]# kill -18 %1
[root@vmrhelu4 ~]# jobs
[1]+  Running                 sleep 100 &
```

Scheduling with at

- To schedule command/s or a script to run at a specific time only once, use the at command
- For example, if you need to backup a critical file called payments, follow these steps
    1. locate the document and check the time using the date command
    2. type the at command and give the time at which you want the job to occur
    3. Press Ctrl+D to exit to the shell
    4. Use the *atq* command to check that the job has been scheduled
    5. To delete a command before it has been executed, use the *atrm* command

Example

```
[root@vmrhelu4 reports]# ls
payments
[root@vmrhelu4 reports]# date
Thu Mar 12 16:10:52 IST 2009
[root@vmrhelu4 reports]# at 16:12
at> cp payments payments.backup
at> <EOT>
job 4 at 2009-03-12 16:12
[root@vmrhelu4 reports]# atq
4       2009-03-12 16:12 a root
[root@vmrhelu4 reports]# atrm 4
[root@vmrhelu4 reports]# atq
```

Access Control of at

- Access to at can be controlled by a Unix administrator
- One method of giving access is to list names of users that are allowed to use *at* in a file called */etc/at.allow*
- The opposite is to deny access to users by listing them in a file called */etc/at.deny*
- If the */etc/at.deny* file is empty, then every user can use the *at* command
- If neither of these files exist, only the superuser is allowed to use at

Scheduling with cron

- *at* command can be used for scheduling jobs that run only once
- Instead, the *cron* program enables Unix users to execute commands, scripts and applications at specified times and/or dates
- *cron* uses a daemon called *crond* that is started with the system and lies dormant until it is required
- When the time arrives to start a job, *cron* spawns a shell in which to run the job, thus allowing the job to execute independently of *cron* itself
- A *cron* job executes with the identity and privileges assigned to the system user who scheduled the job

Crontab (CRON Table)

- This program manipulates the *cron* daemon, making it easier to schedule tasks
- Crontab is also a file that contains commands that will be run by the system at scheduled time
- A system-wide *crontab* file resides at */etc/crontab* and generally contains system scheduled tasks
- To create your user specific *crontab*, use the *crontab* command
  - crontab -l : lists the current cron jobs
  - crontab -e : edit your current crontab
  - crontab -r : remove the crontab file
  - crontab -v : displays the last time you edited your crontab file

Crontab File

- The crontab file uses the following syntax

Crontab Examples

- Schedule temporary files cleanup every day at midnight
- Schedule builds every day at midnight
- Schedule bulk backups on 30 of every month
- Take backup of important documents every 10 minutes
- Run a script every hour
- Run a script every day at 10:30 am, 1:30 pm, 3:30 pm and 5:30 pm
- Run a back every Friday & Monday midnight

Links for objective multiple choice questions.

- http://www.sanfoundry.com/linux-command-mcq-1/
- http://www.sanfoundry.com/linux-command-mcq-2/
- http://www.sanfoundry.com/linux-command-mcq-3/
- http://www.indiabix.com/computer-science/unix/
- http://www.avatto.com/computer-science/test/mcqs/questions-answers/unix/153/1.html
- http://www.gkseries.com/computer-engineering/unix/multiple-choice-questions-and-answers-on-unix-and-shell-programming
- http://www.withoutbook.com/online_test.php?quiz=38&quesNo=10&subject=Top%2010%20UNIX%20Online%20Practice%20Test%20%7C%20Multiple%20Choice