Born2beroot Part 3: 42 Evaluation

(preparation using born2beroot evaluation sheet)
Hostname: evong42
Host/Root password: Lev##20202
New User: Evangelene
New User ID*: evong
Password: Cayo##5782
Encryption passphrase: Asher#5782

* All print screens, User ID evong22 have been replaced with evong since subject paper requested use of intra login as user.
** All print screens in black are taken from the VirtualMachine and print screens in white are taken from local terminal.

Hostname
From root, check with <**hostnamectl**>
Change <**hostnamectl set-hostname new_hostname**>
Change /etc/hosts file <**sudo nano /etc/hosts**>
Change old hostname with new
<**127.0.0.1 localhost**>
<**127.0.0.1 new_hostname**>
Reboot to check change <**sudo reboot**>

Check system requirements
(Show requirements as stated in subject paper)
Type <**lsblk**> to show VM partitions
Type <**head -n 2 /etc/os-release**> or
<**head /etc/os-release**> to see operating system
(-n 2 shows first 2 lines of information block)
Type <**sudo /usr/sbin/aa-status**> to show apparmor module.
Type <**ss -tunlp**> to see ports
Type <**sudo /usr/sbin/ufw status**> to see ufw status

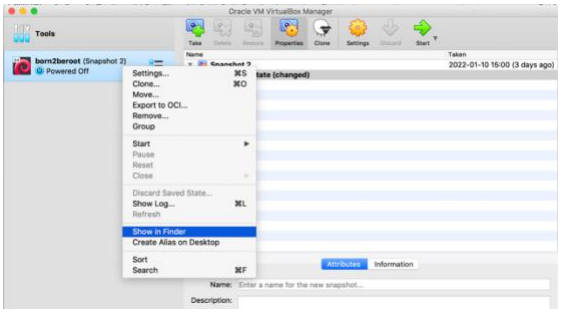
General instructions

General instructions

- During the defense, as soon as you need help to verify a point, the student evaluated must help you.
- Ensure that the "signature.txt" file is present at the root of the cloned repository.
- Check that the signature contained in "signature.txt" is identical to that of the ".vdi" file of the virtual machine to be evaluated. A simple "diff" should allow you to compare the two signatures. If necessary, ask the student being evaluated where their ".vdi" file is located.
- As a precaution, you can duplicate the initial virtual machine in order to keep a copy.
- Start the virtual machine to be evaluated.
- If something doesn't work as expected or the two signatures differ, the evaluation stops here.

1. Send signature.txt to repository before closing project
2. Duplicate virtual machine with snapshots
3. signature.txt identical to ".vdi" file of VM

Where is ".vdi" file located?
(Find directory for VM in local machine)
Right click on VM -> <**Show in Finder**>
For evong: Macintosh HD -> Users -> Vangie -> VirtualBox VMs -> born2beroot -> born2beroot.vdi



Take a snapshot of final version of VM

- Make sure you power off the VM with all necessary configurations -> then close the VM terminal
- From VirtualBox Manager, take a snapshot of the final version of your VM.
- If you are not one of those lucky ones to do all 3 evaluations at the same time, use current state for evaluation.
- After that, restate the current state with the snapshot to restore the VM since signature of VM vdi changes with every command that is executed.

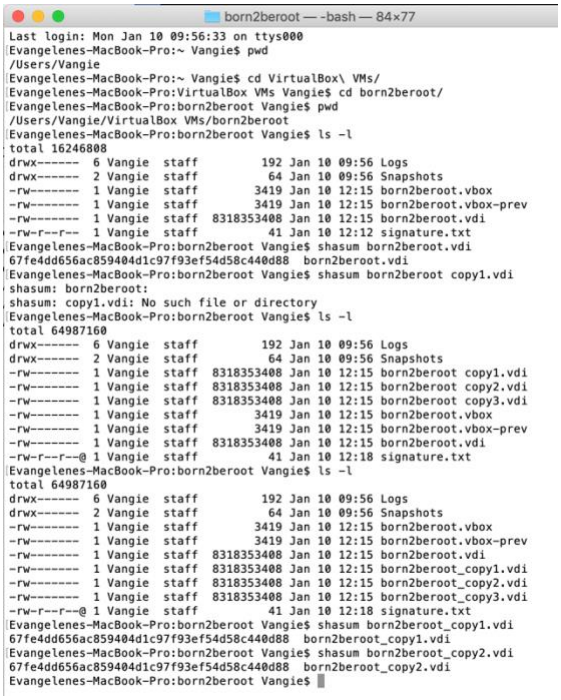
Start Mac Terminal
Command + Space Bar -> Terminal ->

How to get signature of VM?

- From the local terminal, go into the directory where the VM .vdi file is located. (Here, make sure you are not in the virtual machine)
- For Mac, run <**shasum born2beroot.vdi**>
- Put the signature of the VM .vdi (together with vdi filename) into a text file and git add/commit/push to the repository.

```
born2beroot — -bash — 93×41
Last login: Fri Jan  7 11:53:05 on ttys000
Evangelenes-MacBook-Pro:VirtualBox VMs Vangie$ shasum born2beroot.vdi
shasum: born2beroot.vdi:
Evangelenes-MacBook-Pro:VirtualBox VMs Vangie$ ls -l
total 0
drwxr-xr-x  7 Vangie  staff  224 Jan  7 09:10 born2beroot
Evangelenes-MacBook-Pro:VirtualBox VMs Vangie$ cd born2beroot/
Evangelenes-MacBook-Pro:born2beroot Vangie$ ls -l
total 16246800
drwx------  6 Vangie  staff         192 Jan  7 09:10 Logs
drwx------  2 Vangie  staff          64 Jan  7 09:10 Snapshots
-rw-------  1 Vangie  staff        3995 Jan  7 09:10 born2beroot.vbox
-rw-------  1 Vangie  staff        4052 Jan  6 20:26 born2beroot.vbox-prev
-rw-------  1 Vangie  staff  8318353408 Jan  7 11:55 born2beroot.vdi
Evangelenes-MacBook-Pro:born2beroot Vangie$ shasum born2beroot.vdi
7ba493fb9cf21b0cb407430d06e6bc81a091f14a  born2beroot.vdi
Evangelenes-MacBook-Pro:born2beroot Vangie$
```

```
born2beroot — evong22@evong42: / — -bash — 77×42
evong22@evong42:~$ pwd
/home/evong22
evong22@evong42:~$ cd ..
evong22@evong42:/home$ cd ..
evong22@evong42:/$ cd ..
evong22@evong42:/$ pwd
/
evong22@evong42:/$ logout
Connection to localhost closed.
Evangelenes-MacBook-Pro:~ Vangie$ pwd
/Users/Vangie
Evangelenes-MacBook-Pro:~ Vangie$ cd VirtualBox\ VMs/
Evangelenes-MacBook-Pro:VirtualBox VMs Vangie$ pwd
/Users/Vangie/VirtualBox VMs
Evangelenes-MacBook-Pro:VirtualBox VMs Vangie$ cd born2beroot/
Evangelenes-MacBook-Pro:born2beroot Vangie$ pwd
/Users/Vangie/VirtualBox VMs/born2beroot
Evangelenes-MacBook-Pro:born2beroot Vangie$ shasum born2beroot.v
born2beroot.vbox       born2beroot.vbox-prev  born2beroot.vdi
Evangelenes-MacBook-Pro:born2beroot Vangie$ shasum born2beroot.vdi
088940fa7e5d5bbe85e2dd5a2e0d1c7c4e0d7a7e  born2beroot.vdi
Evangelenes-MacBook-Pro:born2beroot Vangie$
```



Mandatory part

The project consists of creating and configuring a virtual machine following strict rules. The student being evaluated will have to help you during the defense. Make sure that all of the following points are observed.

Project overview

- The student being evaluated should explain to you simply:
- How a virtual machine works.
- Their choice of operating system.
- The basic differences between CentOS and Debian.
- The purpose of virtual machines.
- If the evaluated student chose CentOS: what SELinux and DNF are.
- If the evaluated student chose Debian: the difference between aptitude and apt, and what APPArmor is.

During the defense, a script must display information all every 10 minutes. Its operation will be checked in detail later.
If the explanations are not clear, the evaluation stops here.

How a virtual machine works?

ibm.com defines virtualization as "uses software to create an abstraction layer over computer hardware that allows the hardware elements of a single computer - processors, memory, storage and more - to be divided into multiple virtual computers, commonly called virtual machines (VMs). Each VM runs its own operating system (OS) and behaves like an independent computer, even though it is running on just a portion of the actual underlying computer hardware."

It has a CPU, memory, disks to store your files, and can connect to the internet if needed. While the parts that make up your computer (called hardware) are physical and tangible, VMs are often thought of as virtual computers or software-defined computers within physical servers, existing only as code.

vmware.com defines VM as "is a compute resource that uses software instead of a physical computer to run programs and deploy apps (same as any other physical computer). Each virtual machine runs its own operating system and functions separately from the other VMs, even when they are all running on the same host. This means that, for example, a virtual macOS virtual machine can run on a physical PC.".

Purpose of VMs

VMs may be deployed to accommodate different levels of processing power needs, to run software that requires a different operating system, or to test applications in a safe, sandboxed environment. VMs can perform specific tasks considered too risky to carry out in a host environment, such as accessing virus-infected data or testing operating systems. Since the virtual machine is partitioned from the rest of the system, the software inside the VM cannot tamper with the host computer's primary operating system.

Choice of Operating System

debian-11.2.0-amd64-netinst.iso

I am new to system administration and I follow the recommendation on the subject paper to use Debian (which is easier to install and configure than CentOS). It is also recommended for small or personal servers.

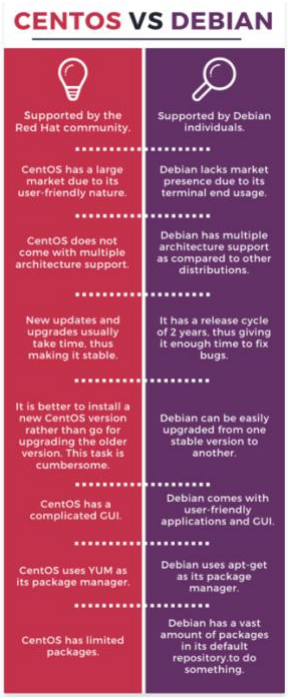
Basic differences between CentOS and Debian

Here are some key differences between both OS:

1. Supported architecture: Both have some commonly supported architectures, such as AArch64/ARM64, armhf/armhfp, i386 or ppc64el/ppc64le, but there are other architectures that are only supported in one of them.
2. Package management:
   - CentOS uses the RPM package format and YUM/DNF as the package manager.
   - Debian uses the DEB package format and dpkg/APT as the package manager.
3. Filesystems: CentOS supports XFS, while Debian EXT4. Both also support ext2/3, NFSv3/4, btrfs, SMB, GFS2...
4. Kernel: The latest Linux kernel is supported in Debian, while CentOS supports older versions.
5. Upgrading: Debian supports major version updates, while CentOS only support minor upgrades (CentOS major releases can have a time span of 10 years).
6. Support: Debian is primarily community-supported, with a larger community than CentOS. Some downstream distributions based on Debian have commercial support options, such as Ubuntu.

My VirtualBox settings

Name: born2beroot.
Type of OS: Linux.
Version: Debian (64-bit).
Memory Size: 1024 MB.
Hard disk: Create a virtual hard disk file.
File size: 8.00 GB.
Hard disk file type: VDI (VirtualBox Disk Image).
Storage on physical hard disk: Dynamically allocated.
Scale Factor: 300%
It is possible to load the Virtual Optical Disk File of the OS (.iso) once the created VM is started, or do it from the VM Settings in Virtual Box, by going to Storage → Controller: IDE → ⊙ (under Attributes).



Debian OS: Difference between aptitude and apt

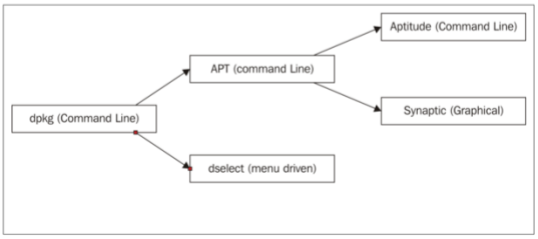
1. The main difference being that Aptitude is a high level package manager while APT is a lower level package manager that can be used by other higher level package managers.
2. Aptitude is more feature-rich than apt-get and incorporates functionality from apt-get and its other variants, including apt-mark and apt-cache. While apt-get handles all package installation, upgrading, system upgrading, package purging, dependency resolution, etc., Aptitude handles a lot more things than apt, including apt-mark and apt-cache functionality, i.e. finding a package in the list of installed packages, marking a package to be installed automatically or manually, containing a package making it unavailable for upgrading, etc.

2. What is the difference between aptitude and apt?

"Apart from main difference being that **Aptitude** is a high-level package manager while **APT** is lower-level package manager which can be used by other higher-level package managers, other main highlights that separate these two package managers are:

1. **Aptitude** is vaster in functionality than **apt-get** and integrates functionalities of apt-get and its other variants including **apt-mark** and **apt-cache**.

While **apt-get** handles all the package installation, up-gradation, system-upgradation, purging package, resolving dependencies etc., Aptitude handles lot more stuff than apt, including functionalities of **apt-mark** and **apt-cache** i.e. searching for a package in list of installed packages, marking a package to be automatically or manually installed, holding a package making it unavailable for up-gradation and so on." (source)



Debian OS: What is APPArmor?

AppArmor (Application Armor) is a security software for Linux systems made under GNU General Public License.

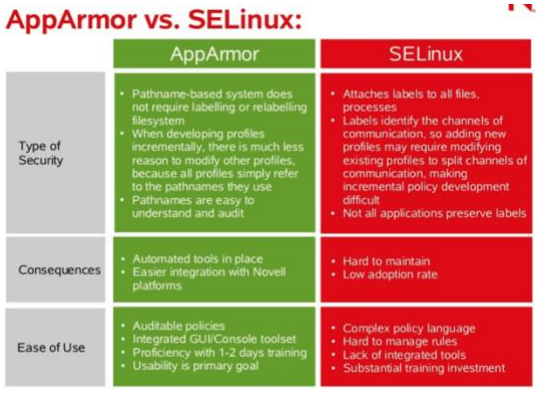
AppArmor allows the system admin to associate to each program a security profile which limits its accesses to the operating system. It completes Unix's traditional way of Discretionary access control (DAC) by allowing the use of Mandatory access control (MAC).

3. Difference between SELinux and APPArmor?

"These security systems provide tools to isolate applications from each other and in turn isolate an attacker from the rest of the system when an application is compromised.

SELinux rule sets are incredibly complex but with this complexity you have more control over how processes are isolated. Generating these policies can be automated. A strike against this security system is that its very difficult to independently verify.

AppArmor (and SMACK) is very straight forward. The profiles can be hand written by humans, or generated using `aa-logprof`. AppArmor uses path based control, making the system more transparent so it can be independently verified." (source)



Script display every 10 minutes

```
Broadcast message from root@evong42 (somewhere) (Wed Jan  5 16:10:01 2022):

#Architecture: Linux evong42 5.10.0-10-amd64 #1 SMP Debian 5.10.84-1 (2021-12-0
8) x86_64 GNU/Linux
#CPU physical : 1
#vCPU : 1
#Memory Usage: 140/976MB (14.34%)
#Disk Usage: 968/1GB (57%)
#CPU load: 0.05%
#Last boot: 2022-01-05 15:49
#LVM use: yes
#Connexions TCP : 4 ESTABLISHED
#User log : 1
#Network : IP10.0.2.15  (08:00:27:92:e5:be)
#Sudo : 112 cmd
```

- The architecture of your operating system and its kernel version.
  - The number of physical processors.
  - The number of virtual processors.
  - The current available RAM on your server and its utilization rate as a percentage.
  - The current available memory on your server and its utilization rate as a percentage.
  - The current utilization rate of your processors as a percentage.
- The date and time of the last reboot.
  - Whether LVM is active or not.
  - The number of active connections.
  - The number of users using the server.
  - The IPv4 address of your server and its MAC (Media Access Control) address.
  - The number of commands executed with the sudo program.

Simple setup

Remember: Whenever you need help checking something, the student being evaluated should be able to help you.
- Ensure that the machine does not have a graphical environment at launch.
A password will be requested before attempting to connect to this machine.
Finally, connect with a user with the help of the student being evaluated.
This user must not be root.
Pay attention to the password chosen, it must follow the rules imposed in the subject.
- Check that the UFW service is started with the help of the evaluator.
- Check that the SSH service is started with the help of the evaluator.
- Check that the chosen operating system is Debian or CentOS with the help of the evaluator.
If something does not work as expected or is not clearly explained,
the evaluation stops here.

1. Machine does not have a graphical environment at launch.
2. Password request for connection to machine.
3. Connect with vangie22 (not root).
4. Operating system is Debian 11 <**head -n 2 /etc/os-release**>
5. Password must follow rules: Password Policy
6. UFW started? <**sudo ufw status**>
7. SSH started? <**sudo service ssh status**>

User

Remember: Whenever you need help checking something, the student being evaluated should be able to help you.

The subject requests that a user with the login of the student being evaluated is present on the virtual machine. Check that it has been added and that it belongs to the "sudo" and "user42" groups.

Make sure the rules imposed in the subject concerning the password policy have been put in place by following the following steps.

First, create a new user. Assign it a password of your choice, respecting the subject rules. The student being evaluated must now explain to you how they were able to set up the rules requested in the subject on their virtual machine.
Normally there should be one or two modified files. If there is any problem, the evaluation stops here.

- Now that you have a new user, ask the student being evaluated to create a group named "evaluating" in front of you and assign it to this user. Finally, check that this user belongs to the "evaluating" group.

- Finally, ask the student being evaluated to explain the advantages of this password policy, as well as the advantages and disadvantages of its implementation. Of course, answering that it is because the subject asks for it does not count.

If something does not work as expected or is not clearly explained, the evaluation stops here.

1. evong exists? <**id**>
2. evong in sudo? <**getent group sudo**>
3. evong in user42? <**getent group user42**>
4. Create a new_user & assign a password to new_user with <**sudo adduser newuser**>
5. How to set up rules? For policy, <**sudo nano /etc/login.defs**>. For pw strength, <**sudo nano /etc/pam.d/common-password**>
6. One or two modified files? wtf?
7. Create a group “evaluating”. <**sudo addgroup evaluating**>
8. Assign user to evaluating. <**sudo usermod -aG user42 your_username**> <**sudo adduser newuser evaluating**>
9. Check user in group/evaluating. <**getent group evaluating**>
10. Advantages of this pw policy? Password is kept secure and updated
11. +/- of pw policy implementation? + implemented for a class of users. + enforce for root/groups. – 30 days expiry.

Hostname and partitions

Remember: Whenever you need help checking something, the student being evaluated should be able to help you.

- Check that the hostname of the machine is correctly formatted as follows:
login42 (login of the student being evaluated).
- Modify this hostname by replacing the login with yours, then restart the machine.
If on restart, the hostname has not been updated, the evaluation stops here.
- You can now restore the machine to the original hostname.
- Ask the student being evaluated how to view the partitions for this virtual machine.
- Compare the output with the example given in the subject. Please note: if the student evaluated makes the bonuses, it will be necessary to refer to the bonus example.

This part is an opportunity to discuss the scores! The student being evaluated should give you a brief explanation of how LVM works and what it is all about.
If something does not work as expected or is not clearly explained,
the evaluation stops here.

1. Hostname: evong42
2. <**lsblk**> to view the partitions for this VM
3. Output same as bonus example
4. Logical Volume Manager in wiki.ubuntu.com, "It is a system of managing logical volumes, or filesystems, that is much more advanced and flexible than the traditional method of partitioning a disk into one or more segments and formatting that partition with a filesystem.".

<u>Hostname</u>
Check with <**hostnamectl**>
Modify <**hostnamectl set-hostname new_hostname**>
Change /etc/hosts file <**sudo nano /etc/hosts**>
Change old hostname with new
<**127.0.0.1 localhost**>
<**127.0.0.1 new_hostname**>
Reboot to check change <**sudo reboot**>

SUDO

Remember: Whenever you need help checking something, the student being evaluated should be able to help you.

- Check that the "sudo" program is properly installed on the virtual machine.
- The student being evaluated should now show assigning your new user to the "sudo" group.
- The subject imposes strict rules for sudo. The student being evaluated must first explain the value and operation of sudo using examples of their choice.
In a second step, it must show you the implementation of the rules imposed by the subject.
- Verify that the "/var/log/sudo/" folder exists and has at least one file. Check the contents of the files in this folder, You should see a history of the commands used with sudo.
Finally, try to run a command via sudo. See if the file (s) in the "/var/log/sudo/" folder have been updated.
If something does not work as expected or is not clearly explained, the evaluation stops here.

1. Verify of sudo program is properly installed <**apt-cache policy sudo**>
2. Create new user <**sudo adduser newuser**>
3. Assign new user to sudo <**sudo adduser newuser sudo**>
4. Explain value and operation of sudo: root pw not disclosed; temporary privileges to individuals; only certain commands via sudo to minimize mistakes
5. Implementation of the rules
6. Verify /var/log/sudo/ exists and has at least one file, contents has history of commands used with sudo: <**sudo ls -la /var/log/sudo/sudolog**> or <**sudo cat /var/log/sudo/sudolog**>

7. Run a command via sudo, files updated? <**sudo aa-status**> <**sudo systemctl status sshd**>

UFW

Remember: Whenever you need help checking something, the student being evaluated should be able to help you.

- Check that the "UFW" program is properly installed on the virtual machine.
- Check that it is working properly.
- The student being evaluated should explain to you basically what UFW is and the value of using it.
- List the active rules in UFW. A rule must exist for port 4242.
- Add a new rule to open port 8080. Check that this one has been added by listing the active rules.
- Finally, delete this new rule with the help of the student being evaluated.

If something does not work as expected or is not clearly explained, the evaluation stops here.

1. UFW properly installed? <**dpkg -l | grep ufw**>
2. UFW is working properly? <**sudo ufw status verbose**>
3. What UFW is and value of using it? Uncomplicated firewall. Controls external access via ssh
4. List the active rules in UFW. A rule must exist for port 4242. <**sudo ufw status verbose**>
5. Add new rule to open port 8080. <**sudo ufw allow 8080**>
6. List active rules. <**sudo ufw status verbose**>
7. Delete this new rule. <**sudo ufw delete RULE | number**>
8. <**sudo ufw delete allow 8080**> or <**sudo ufw delete deny 8080**>
9. Deny this new rule. <**sudo ufw deny 8080**>
10. Help: <**sudo ufw –help**>

SSH

Remember: Whenever you need help checking something, the student being evaluated should be able to help you.

- Check that the SSH service is properly installed on the virtual machine.
- Check that it is working properly.
- The student being evaluated must be able to explain to you basically what SSH is and the value of using it.
- Verify that the SSH service only uses port 4242.
- The student being evaluated should help you use SSH in order to log in with the newly created user. To do this, you can use a key or a simple password. It will depend on the student being evaluated. Of course, you have to make sure that you cannot use SSH with the "root" user as stated in the subject.

If something does not work as expected or is not clearly explained, the evaluation stops here.

1. SSH properly installed and working: <**sudo service ssh status**>
2. What is SSH and the value of using it? Secure Shell. Network communication protocol that enables two computers to communicate and share data.
3. SSH only uses port 4242: <**cat /etc/ssh/sshd_config**> Port 4242
4. Use SSH to log in with new user (use key or simple password): <**ssh evong22@localhost -p 4242**>
5. Show cannot ssh with root user: <**cat /etc/ssh/sshd_config**> PermitRootLogin no

Script monitoring

Remember: Whenever you need help checking something, the student being evaluated should be able to help you.

The student being evaluated should explain to you simply:
- How their script works by showing you the code.
- What "cron" is.
- How the student being evaluated set up their script so that it runs every 10 minutes from when the server starts.

Once the correct functioning of the script has been verified, the student being evaluated should ensure that this script runs every minute. You can run whatever you want to make sure the script runs with dynamic values correctly. Finally, the student being evaluated should make the script stop running when the server has started up, but without modifying the script itself. To check this point, you will have to restart the server one last time. At startup, it will be necessary to check that the script still exists in the same place, that its rights have remained unchanged, and that it has not been modified.

If something does not work as expected or is not clearly explained, the evaluation stops here.

1. Explain how script works: <**cat monitoring.sh**>
   - The architecture of your operating system and its kernel version.
   - The number of physical processors.
   - The number of virtual processors.
   - The current available RAM on your server and its utilization rate as a percentage.
   - The current available memory on your server and its utilization rate as a percentage.
   - The current utilization rate of your processors as a percentage.
   - The date and time of the last reboot.
   - Whether LVM is active or not.
   - The number of active connections.
   - The number of users using the server.
   - The IPv4 address of your server and its MAC (Media Access Control) address.
   - The number of commands executed with the sudo program.
2. What is cron? The crontab is a list of commands that you want to run on a regular schedule, and also the name of the command used to manage that list. Crontab stands for “cron table, ” because it uses the job scheduler cron to execute tasks; cron itself is named after “chronos, ” the Greek word for time.
3. How set up script? <**nano monitoring.sh**> <**bash ./monitoring.sh**>, need access to sudolog file, update rwx
4. Stop script when server starts without modifying script?? Restart server – script still exists in the same place, unchanged, not modified? How to interrupt script without modifying it?
   Do this by stopping crontab ->
   <**systemctl status name.service**>
   <**systemctl status cron.service**>
   <**sudo systemctl stop cron.service**>
   <**sudo systemctl start cron.service**>

Bonus

Evaluate the bonus part if, and only if, the mandatory part has been entirely and perfectly done, and the error management handles unexpected or bad usage. In case all the mandatory points were not passed during the defense, bonus points must be totally ignored.

Bonus

Check, with the help of the subject and the student being evaluated, the bonus points authorized for this project:
- Setting up partitions is worth 2 points.
- Setting up WordPress, only with the services required by the subject, is worth 2 points.
- The free choice service is worth 1 point.

Verify and test the proper functioning and implementation of each extra service.
For the free choice service, the student being evaluated has to give you a simple explanation about how it works and why they think it is useful.
Please note that NGINX and Apache2 are prohibited.

1. 2 points for setting up partitions
2. 2 points for setting up WordPress