**2. LINUX (30 ptos)**

1. ¿ Qué comando permite monitorear los recursos del servidor(memoria/procesador) ?

Comando : \_FREE O FREE – H / \_\_TOP O HTOP\_\_\_\_\_\_\_\_

Ejemplo : \_MEMORIA RAM\_\_\_/ \_PROCESADOR\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. ¿ Qué comando permite listar el contenido de un directorio incluyendo archivos ocultos ?

Comando : \_\_LS\_\_\_\_\_\_\_\_\_

Ejemplo : directorio de una carpeta\_\_ [root@LINUX1]# ls -lF -d 1 /root/.\*\_\_\_

1. ¿ Qué comando permite empaquetar archivos o directorios ?

Comando : \_mkdir y tar y gzip\_\_\_\_\_\_\_\_\_\_

Ejemplo :prueba@admin:~$ mkdir prueba

prueba@admin:~$ for N in `seq 1 1000`; do cp TEXTO "prueba/"$N"\_TEXTO"; done

prueba@admin:~$ tar -cvf dbigcloud.tar /directorio/

prueba@admin:~$ gzip dbigcloud

1. ¿ Qué comando permite borrar directorios con todo su contenido ?

Comando : \_\_ rm –r\_\_\_\_\_\_\_\_\_

Ejemplo : \_\_ sudo rm -r nombre del directorio\_\_

1. ¿ Con qué comando verifico si un proceso específico se encuentra en ejecución ?

Comando : $ sudo top

Ejemplo : $ sudo pidof nombre-del-proceso \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

$ ps axu

USER PID %CPU %MEM VSZ RSS TTY STAT START TIME COMMAND

root 1 0.0 0.0 1516 536 ? S 09:43 0:00 init [2]

root 2 0.0 0.0 0 0 ? S 09:43 0:00 [migration/0]

root 3 0.0 0.0 0 0 ? SN 09:43 0:00 [ksoftirqd/0]

root 4 0.0 0.0 0 0 ? S 09:43 0:00 [migration/1]

.............

tomas 6475 0.1 4.9 140180 50920 ? Sl 09:51 0:18 /usr/lib/mozilla-thunderbird/mozilla-thunderbird-bin

tomas 6528 0.2 3.6 116396 37948 ? Sl 10:01 0:25 /usr/lib/mozilla-firefox/firefox-bin -a firefox

1. ¿ Cuál es el comando para visualizar los puertos abiertos por procesos en ejecución?

Comando : \_\_sudo isof\_\_\_\_\_\_

cat /etc/services \_\_\_

Ejemplo : $ sudo lsof -i -P -n

$ sudo lsof -i -P -n | grep LISTEN

1. ¿ Qué comando permite visualizar las últimas líneas de un archivo de manera continua?

Comando : \_\_\_\_TAIL\_\_\_\_\_\_\_

Ejemplo : \_\_\_\_\_ Tail –n\* ruta\_de\_archivo

tail -n2 devisers.txt

¿ Cómo se cambia el propietario y grupo de una carpeta con todo contenido ?

Comando : \_\_\_\_ chmod/chown. \_\_\_\_\_\_\_

Ejemplo : chown -R 755 /etc/prueba

1. ¿ Cómo se cambia los permisos de un fichero ?

Comando : \_\_\_\_\_ chmod permisos nombre \_\_\_\_\_\_

Ejemplo : \_\_\_\_ $ ls -l prueba

drwxr-xr-x 3 user2 1024 Ago 18 19:15 prueba

$ chmod g+w prueba

$ ls -l prueba

drwxrwxr-x 3 user2 1024 Ago 18 19:15 prueba

$ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. ¿ Cómo puedo ver las unidades, particiones y espacio en disco del sistema operativo ?

Comando : \_ df –h

df –Th

sudo blkid /dev/sda1

pydf

Ejemplo : \_\_\_\_\_\_\_ sudo lsblk -fm \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. ¿ Cómo puedo saber el tiempo de ejecución de un proceso ?

Comando : \_  ps y pdof \_y\_time\_\_\_\_\_\_\_\_

Ejemplo : \_\_\_\_ # time du -sh /root/

464K /root/

real 0m0.007s

user 0m0.002s \_\_

1. ¿ Cómo realizo una transferencia de un archivo o un directorio entre servidores ?

Comando : \_\_\_SCP\_\_\_\_\_\_\_\_

Ejemplo : scp nombre\_archivo usuario@servidor:ruta\_servidor\_donde\_colocar\_archivo

scp user@1.2.3.4:/ruta/servidor/remoto/archivo.tgz archivo-en-local.tgz

1. ¿ Cómo puedo listar los directorios con sus respectivos tamaños en MB ?

Comando : \_\_\_\_du -h \_\_\_\_\_\_\_

Ejemplo : \_ [root@prueba SYSADMIT]# du -h

121M ./SUBDIR3

73M ./SUBDIR2

49M ./SUBDIR1/SUB1

169M ./SUBDIR1

361M . \_\_\_\_\_\_\_\_\_\_\_\_

**3. Sistema de versionamiento GIT**

**3.1 Indique los comandos GIT para realizar las siguientes acciones: (14 ptos )**

1. Ver la historia de una rama/commit

git log

1. Clonar un repositorio

git clone https://github.com/schacon/simplegit-progit

1. Crear una rama a partir de la versión de la rama master

$ git branch hola

$ git checkout hola

Switched to branch 'hola'

1. Crear una rama y cambiarse a la misma

git branch hola

git Checkout hola

1. Ver las diferencias entre 2 ramas o commits sin considerar cambios en espacios

Git status

git log

git diff

1. Restablecer una rama a un commit específico

**git** reset

1. Deshacer un commit conservando los archivos modificados

$ # Modificamos hola.php

$ cat hola.php

<?php

// Este comentario está mal y hay que borrarlo

$nombre = isset($argv[1]) ? $argv[1] : "Mundo";

@print "Hola, {$nombre}\n";

$ git add hola.php

$ git status

# On branch master

# Changes to be committed:

# (use "git reset HEAD <file>..." to unstage)

#

# modified: hola.php

1. Agregar cambios a un commit incluyendo archivos eliminados

$ git checkout master

Previous HEAD position was 3283e0d... Se añade un parámetro por defecto

Switched to branch 'master'

$ # Modificamos hola.php

$ cat hola.php

<?php

// Este comentario está mal y hay que borrarlo

$nombre = isset($argv[1]) ? $argv[1] : "Mundo";

@print "Hola, {$nombre}\n";

$ git status

# On branch master

# Changes not staged for commit:

# (use "git add <file>..." to update what will be committed)

# (use "git checkout -- <file>..." to discard changes in working direct

ory)

#

# modified: hola.php

#

no changes added to commit (use "git add" and/or "git commit -a")

1. Hacer un commit que deshaga los cambios de un commit específico

$ # Modificamos hola.php

$ cat hola.php

<?php

// Este comentario está mal y hay que borrarlo

$nombre = isset($argv[1]) ? $argv[1] : "Mundo";

@print "Hola, {$nombre}\n";

$ git add hola.php

$ git commit -m "Ups... este commit está mal."

master 5a5d067] Ups... este commit está mal

1 file changed, 1 insertion(+), 1 deletion(-)

1. Actualizar repositorio local con el remoto

git push

git push -f origin master

1. Publicar cambios desde el repositorio local hacia el remoto

git remote add upstream https://github.com/whoever/whatever.git

git rebase upstream/master

1. Restablecer una rama 2 commits atrás

cherry-pick

1. Ver todas las ramas locales y remotas

Git show

$ git fetch origin

remote: Counting objects: 7, done.

remote: Compressing objects: 100% (2/2), done.

remote: Total 3 (delta 0), reused 3 (delta 0)

Unpacking objects: 100% (3/3), done.

From https://github.com/schacon/simplegit

\* [new branch] serverfix -> origin/serverfix

1. Aplicar los cambios de otra rama sin generar commit de merge

$ git checkout master

Switched to branch 'master'

$ git commit -a -m 'made other changes'

**4. Arquitectura cliente - servidor ( 16 puntos )**

**1. La arquitectura cliente-servidor consiste básicamente en:**

a) Un servidor que realiza peticiones a otro programa (el cliente) que le da respuesta.

b) Un cliente que realiza peticiones a otro programa (el servidor) que le da respuesta

c) Un cliente que realiza peticiones a otro programa (el cliente) que le da respuesta.

d) Ninguna de las anteriores.

**2. La red cliente-servidor es aquella red de comunicaciones en la que todos los clientes están conectados a:**

a) Servidor

b) Cliente

c) Computador personal.

d) Ninguna de las anteriores.

**3. En la arquitectura C/S, todas las gestiones que se realizan se concentran en el:**

a) Cliente.

b) Router.

c) Servidor

d) Ninguna de las anteriores.

**4. El término front-end se refiere a :**

a) La red.

b) Los cables.

c) El cliente.

d) Ninguna de las anteriores.

**5. El término back-end se refiere a :**

a) La red.

b) El servidor

c) El cliente.

d) Ninguna de las anteriores.

**6. Las funciones que lleva a cabo el proceso cliente son:**

a) Aceptar los requerimientos de bases de datos que hacen los clientes, procesar requerimientos de bases de datos, formatear datos para transmitirlos a los clientes, etc

b) Administrar la interfaz de usuario, interactuar con el usuario, recibir resultados del servidor, etc.

c) Eliminación de datos.

d) Ninguna de las anteriores.

**7. Las funciones que lleva a cabo el proceso servidor son:**

a) Aceptar los requerimientos de bases de datos que hacen los clientes, procesar requerimientos de bases de datos, formatear datos para transmitirlos a los clientes, etc

b) Administrar la interfaz de usuario, interactuar con el usuario, recibir resultados del servidor, etc.

c) Eliminación de datos.

d) Ninguna de las anteriores.

**8. Una de las ventajas de la arquitectura C/S es la Escalabilidad y se refiere:**

a) Los accesos, recursos y la integridad de los datos son controlados por el servidor.

b) Se puede aumentar la capacidad de clientes y servidores por separado.

c) Es imposible aumentar la capacidad de clientes y servidores por separado.

d) Ninguna de las anteriores.

**9. Otra de las ventajas de la arquitectura C/S es la Centralización del control, esta se refiere a:**

a) Los accesos, recursos y la integridad de los datos son controlados por el cliente.

b) Los accesos, recursos y la integridad de los datos son controlados por el back end.

c) Los accesos, recursos y la integridad de los datos son controlados por el servidor

d) Ninguna de las anteriores.

**10. El paradigma de C/S. Cuando una gran cantidad de clientes envían peticiones simultáneas al mismo servidor. Hablamos de una desventaja, esta es:**

a) No tiene la robustez.

b) La congestión del tráfico.

c) El cliente no dispone de los recursos que puedan existir en el servidor.

d) Ninguna de las anteriores.

**11. En lo referente a los servicios proporcionados por múltiples servidores, consiste:**

a) Disponer de múltiples servidores que se distribuyan las peticiones (cluster).

b) Disponer de múltiples clientes que se distribuyan las peticiones(cluster).

c) Disponer de múltiples front-end que se distribuyan las peticiones (cluster).

d) Ninguna de las anteriores.

**12. El componente de seguridad en la que se realiza la protección contra la alteración o corrupción. Hablamos de:**

a) Confidencialidad.

b) Disponibilidad.

c) Integridad.

d) Ninguna de las anteriores.

**13. El middleware**:

a) Es el intermediario entre el usuario y el cliente y se ejecuta en ambas partes.

b) Es el intermediario entre el cliente y el servidor y se ejecuta solo en el cliente.

c) Es el intermediario entre el cliente y el servidor y se ejecuta en ambas partes.

d) Ninguna de las anteriores.

**14. Una de las principales características de un middleware es:**

a) Facilita el desarrollo de sistemas complejos con diferentes tecnologías y arquitecturas.

b) Permite la eliminación de datos.

c) Aumenta la capacidad del usuario.

d) Ninguna de las anteriores.

**15. Escalar verticalmente significa**:

a) El añadir más recursos a un solo nodo en particular dentro de un sistema.

b) El añadir más recursos a todos los nodos dentro de un sistema.

c) No añadir recursos

d) Ninguna de las anteriores.

**16. Escalar horizontalmente significa:**

a) Agregar más nodos a un sistema

b) El añadir más recursos a un solo nodo en particular dentro de un sistema.

c) Quitar nodos a un sistema.

d) Ninguna de las anteriores.

**5. PRÁCTICA ( En cualquier PC que tenga Docker )**

Para toda esta sección, debe adjuntar link de **un video** que evidencia el correcto despliegue y funcionamiento de lo solicitado.

**SonarQube ( 10 ptos )**

Desplegar un SonarQube mediante Docker:

<https://docs.sonarqube.org/latest/setup/get-started-2-minutes/>

**Grafana ( 10 ptos )**

Desplegar un Grafana mediante Docker:

<https://grafana.com/docs/grafana/latest/installation/docker/>

**Proyecto Java (20 ptos)**

Desplegar por medio de Docker compose el siguiente proyecto Java:

<https://gitlab.com/pruebadevops/springbootgrettings.git>

Configurar para que el servicio sea accedido por medio del puerto 28080 y comprobar consumiendo el siguiente curl:

curl -H Host:springboot.docker.localhost http://127.0.0.1:28080/greeting