



Universidad
de Huelva



Escuela Técnica Superior de Ingeniería
Universidad de Huelva

Memoria de Prácticas

ADMINISTRACIÓN DE SERVIDORES

Grado en Ingeniería Informática

Autor: Alejandro Gordillo Pedraza

16 de octubre de 2023

Resumen

Esta práctica ha consistido en instalar dos servidores en distintos contenedores y aprender a monitorizar ambos en dos situaciones.

Estas vendrían siendo una en estado normal y otra en un estado de estrés. Durante esta práctica compararemos el estado de los servidores y comprobaremos nuestra capacidad de monitorización y compresión. Cabe destacar que, para la realización de estas práctica y posiblemente próximas prácticas, haremos uso de ProxMox.

Índice general

1. Práctica de monitorización

1.1 Creación el entorno de pruebas.....	4
1.2 Estudio de las capacidades de un servidor.....	4
1.3 Monitorización del servidor	7
1.4 Monitorización del servidor ante situaciones de estrés	12
1.5 Conclusión.....	13

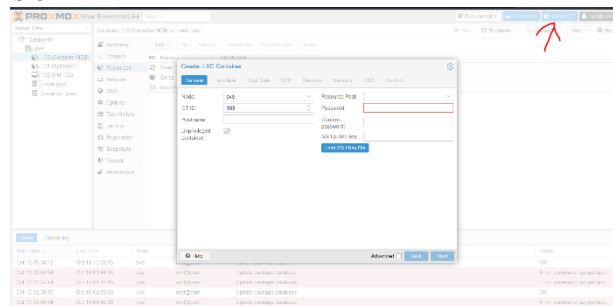
Capítulo 1

Práctica de monitorización

1.1 Creación el entorno de pruebas

Para crear un nuevo contenedor en Proxmox accederemos a nuestra maquina e iniciaremos sesión con el usuario con permisos que vayamos a usar para nuestro contenedor.

Después pulsaremos en el botón de Crear CT y nos abrirá un menú de configuración para la creación del contenedor. Aquí podremos ponerle un nombre y una contraseña a nuestro CT. Captura se puede ver una previsualización antes de darle a finalizar para comprobar si todo lo que hemos puesto está bien.



Seguimos haciendo los siguientes pasos, donde nos pedirá que elijamos una cantidad de cores, el tamaño del disco y el tamaño de la memoria de la que va a disponer nuestro contenedor.

Para finalizar llegaremos al paso final donde comprobaremos que este todo en orden y ya crearemos nuestro Contenedor.

1.2 Estudio de las capacidades de un servidor

Obtenga la información referente a la capacidad de el o los procesadores (puede consultar el fichero `/proc/cpuinfo`).

```
[root@Sysbench:~]# cat /proc/cpuinfo
processor       : 0
vendor_id      : GenuineIntel
cpu family     : 6
model          : 85
model name     : Intel(R) Xeon(R) Silver 4110 CPU @ 2.10GHz
stepping       : 4
microcode     : 0x1
cpu MHz        : 2095.076
cache size     : 16384 KB
physical id    : 0
siblings       : 4
core id        : 0
cpu cores      : 4
apicid         : 0
initial apicid : 0
fpu            : yes
fpu_exception  : yes
cpuid level    : 22
wp             : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx fx
sr sse sse2 ss ht syscall nx pdpe1gb rdtscp lm constant_tsc arch_perfmon rep_good nopl xtopology cpuid t
sc_known_freq pni pclmulqdq vmx ssse3 fma cx16 pcid sse4_1 sse4_2 xzavic moveb popcnt tsc_deadline_timer
aes xsave avx f16c rdrand hypervisor lahf_lm abm 3dnowprefetch cpuid_fault invpcid_single pti sdbg bna
l1bpb st1bpb tpr_shadow vmml flexpriority ept vpid ept_ad fsgbase tsc_adjust bml hle avx2 smep bml2 erm
s invpcid rtm mpx avx512f avx512dq rdseed adx smap clflushopt clwb avx512cd avx512bw avx512vl xsaveopt x
savec xgetbv1 xsaues arat unip pku ospke arch_capabilities
bugs           : cpu_meltdown spectre_v1 spectre_v2 spec_store_bypass l1tf mds swapsys taa
bogomips       : 4190.15
clflush size   : 64
cache alignme  : 64
address sizes  : 40 bits physical, 48 bits virtual
power managem  :

processor       : 1
vendor_id      : GenuineIntel
cpu family     : 6
model          : 85
model name     : Intel(R) Xeon(R) Silver 4110 CPU @ 2.10GHz
stepping       : 4
microcode     : 0x1
cpu MHz        : 2095.076
cache size     : 16384 KB
physical id    : 0
siblings       : 4
core id        : 2
cpu cores      : 4
apicid         : 2
initial apicid : 2

[root@ContainerMDB:~]# cat /proc/cpuinfo
processor       : 0
vendor_id      : GenuineIntel
cpu family     : 6
model          : 85
model name     : Intel(R) Xeon(R) Silver 4110 CPU @ 2.10GHz
stepping       : 4
microcode     : 0x1
cpu MHz        : 2095.076
cache size     : 16384 KB
physical id    : 0
siblings       : 4
core id        : 1
cpu cores      : 4
apicid         : 1
initial apicid : 1
fpu            : yes
fpu_exception  : yes
cpuid level    : 22
wp             : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx fx
sr sse sse2 ss ht syscall nx pdpe1gb rdtscp lm constant_tsc arch_perfmon rep_good nopl xtopology cpuid t
sc_known_freq pni pclmulqdq vmx ssse3 fma cx16 pcid sse4_1 sse4_2 xzavic moveb popcnt tsc_deadline_timer
aes xsave avx f16c rdrand hypervisor lahf_lm abm 3dnowprefetch cpuid_fault invpcid_single pti sdbg bna
l1bpb st1bpb tpr_shadow vmml flexpriority ept vpid ept_ad fsgbase tsc_adjust bml hle avx2 smep bml2 erm
s invpcid rtm mpx avx512f avx512dq rdseed adx smap clflushopt clwb avx512cd avx512bw avx512vl xsaveopt x
savec xgetbv1 xsaues arat unip pku ospke arch_capabilities
bugs           : cpu_meltdown spectre_v1 spectre_v2 spec_store_bypass l1tf mds swapsys taa
bogomips       : 4190.15
clflush size   : 64
cache alignme  : 64
address sizes  : 40 bits physical, 48 bits virtual
power managem  :

processor       : 1
vendor_id      : GenuineIntel
cpu family     : 6
model          : 85
model name     : Intel(R) Xeon(R) Silver 4110 CPU @ 2.10GHz
stepping       : 4
microcode     : 0x1
cpu MHz        : 2095.076
cache size     : 16384 KB
physical id    : 0
siblings       : 4
core id        : 3
cpu cores      : 4
apicid         : 3
initial apicid : 3
```

```

root@Sysbench:~# free
              total        used        free      shared  buff/cache   available
Mem:           524288        40380        315360        10560        168548        483908
Swap:          1048576           0         1048576

root@Sysbench:~# vmstat
procs-----memory-----swap-----io-----system-----cpu-----
 r b swpd free buff cache si so bi bo in cs us sy id wa st
0 0 1 0 315128 0 168548 0 0 39 8 7 6 0 0 100 0 0

root@Sysbench:~#

root@ContainerMDB:~# free
              total        used        free      shared  buff/cache   available
Mem:           524288        38636        38680        10164        446972        485652
Swap:          1048576           0         1048576

root@ContainerMDB:~# vmstat
procs-----memory-----swap-----io-----system-----cpu-----
 r b swpd free buff cache si so bi bo in cs us sy id wa st
0 0 0 0 38540 0 447104 0 0 39 8 7 6 0 0 100 0 0

root@ContainerMDB:~#

```

Los comandos **free** y **vmstat** pueden servir para monitorizar el uso de la memoria. Use ambos comandos para obtener la información referente a la capacidad de la memoria (total de la memoria, memoria virtual usada, espacio destinado a las cachés y buffers, etc).

```

root@Sysbench:~# df
Filesystem            1K-blocks    Used Available Use% Mounted on
/dev/mapper/pve-vm--101--disk--0 8191416 725596 7030008 10% /
none                  492        4      488    1% /dev
udev                 4049500      0 4049500  0% /dev/tty
tmpfs                 4077380      0 4077380  0% /dev/shm
tmpfs                 4077380 10608 4066772  1% /run
tmpfs                 4077380      0 4077380  0% /sys/fs/cgroup
tmpfs                 52432       0  52432  0% /run/user/0

root@Sysbench:~#

root@ContainerMDB:~# df
Filesystem            1K-blocks    Used Available Use% Mounted on
/dev/mapper/pve-vm--100--disk--0 8191416 1044004 6711600 14% /
none                  492        4      488    1% /dev
udev                 4049500      0 4049500  0% /dev/tty
tmpfs                 4077380      0 4077380  0% /dev/shm
tmpfs                 4077380 10416 4066964  1% /run
tmpfs                 4077380      0 4077380  0% /sys/fs/cgroup
tmpfs                 52432       0  52432  0% /run/user/0

root@ContainerMDB:~#

```

El comando **df** muestra el espacio libre y ocupado en los distintos sistemas de ficheros que tiene montados el sistema, obtenga la información referente a la capacidad de los discos.

```

root@Sysbench:~# df
Filesystem            1K-blocks    Used Available Use% Mounted on
/dev/mapper/pve-vm--101--disk--0 8191416 725596 7030008 10% /
none                  492        4      488    1% /dev
udev                 4049500      0 4049500  0% /dev/tty
tmpfs                 4077380      0 4077380  0% /dev/shm
tmpfs                 4077380 10608 4066772  1% /run
tmpfs                 4077380      0 4077380  0% /sys/fs/cgroup
tmpfs                 52432       0  52432  0% /run/user/0

root@Sysbench:~#

root@ContainerMDB:~# df
Filesystem            1K-blocks    Used Available Use% Mounted on
/dev/mapper/pve-vm--100--disk--0 8191416 1044004 6711600 14% /
none                  492        4      488    1% /dev
udev                 4049500      0 4049500  0% /dev/tty
tmpfs                 4077380      0 4077380  0% /dev/shm
tmpfs                 4077380 10416 4066964  1% /run
tmpfs                 4077380      0 4077380  0% /sys/fs/cgroup
tmpfs                 52432       0  52432  0% /run/user/0

root@ContainerMDB:~#

```

Los comandos **lsblk** e **iostat** muestran información sobre los distintos sistemas de ficheros montados. En lo que se refiere al nombre del dispositivo, ¿aprecia alguna diferencia?

```

root@Sysbench:~# lsblk
lsblk: dm-1: failed to get device path
lsblk: dm-8: failed to get device path
lsblk: dm-6: failed to get device path
lsblk: dm-4: failed to get device path
lsblk: dm-2: failed to get device path
lsblk: dm-0: failed to get device path
lsblk: dm-7: failed to get device path
lsblk: dm-1: failed to get device path
lsblk: dm-2: failed to get device path
lsblk: dm-0: failed to get device path
lsblk: dm-3: failed to get device path
lsblk: dm-5: failed to get device path
lsblk: dm-3: failed to get device path
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sr0   11:0  1  886M  0 rom
sda   8:0  0 128G  0 disk
|-sda2 8:2  0 512M  0 part
|-sda3 8:3  0 127.5G  0 part
|-sda1 8:1 1007K  0 part

root@Sysbench:~#

root@ContainerMDB:~# lsblk
lsblk: dm-1: failed to get device path
lsblk: dm-8: failed to get device path
lsblk: dm-6: failed to get device path
lsblk: dm-4: failed to get device path
lsblk: dm-2: failed to get device path
lsblk: dm-0: failed to get device path
lsblk: dm-7: failed to get device path
lsblk: dm-1: failed to get device path
lsblk: dm-2: failed to get device path
lsblk: dm-0: failed to get device path
lsblk: dm-3: failed to get device path
lsblk: dm-5: failed to get device path
lsblk: dm-3: failed to get device path
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sr0   11:0  1  886M  0 rom
sda   8:0  0 128G  0 disk
|-sda2 8:2  0 512M  0 part
|-sda3 8:3  0 127.5G  0 part
|-sda1 8:1 1007K  0 part

root@ContainerMDB:~#

```

Aparecen los mismos errores en los mismos contenedores, creo que pueda ser debido a que la BD de udevdb no este disponible. Puesto que el comando lsblk lee el sistema de archivos sysfs y el sistema udevdb. Dado que me salta un error no encuentro diferencias aparentes.

[root@Sysbench ~]# iostat									
Linux 5.4.106-1-pve (Sysbench) 10/18/23 _x86_64_ (4 CPU)									
avg-cpu: %user %nice %system %iowait %steal %idle									
0.01 0.00 0.00 0.00 0.00 99.99									
Device:	tps	kB_read/s	kB_wrtn/s	kB_read	kB_wrtn				
scd0	0.00	0.00	0.00	2096	0				
sda	2.84	105.07	21.64	178046091	36672200				
dm-0	0.00	0.00	0.00	3264	4				
dm-1	2.21	0.17	18.99	283197	32178452				
dm-2	0.00	0.00	0.01	1220	15912				
dm-3	0.60	0.61	3.73	1030559	6317292				
dm-4	0.60	0.61	3.73	1030559	6317292				
dm-6	0.35	0.55	1.94	927073	3283712				
dm-7	0.25	0.06	1.49	108866	2522396				
dm-8	0.00	0.00	0.00	5325	0				
[root@Sysbench ~]#									

[root@ContainerMDB ~]# iostat									
Linux 5.4.106-1-pve (ContainerMDB) 10/18/23 _x86_64_ (4 CPU)									
avg-cpu: %user %nice %system %iowait %steal %idle									
0.01 0.00 0.00 0.00 0.00 99.99									
Device:	tps	kB_read/s	kB_wrtn/s	kB_read	kB_wrtn				
scd0	0.00	0.00	0.00	2096	0				
sda	2.84	105.05	21.64	178046091	36672146				
dm-0	0.00	0.00	0.00	3264	4				
dm-1	2.21	0.17	18.99	283197	32178412				
dm-2	0.00	0.00	0.01	1220	15912				
dm-3	0.60	0.61	3.73	1030559	6317264				
dm-4	0.60	0.61	3.73	1030559	6317264				
dm-6	0.35	0.55	1.94	927073	3283688				
dm-7	0.25	0.06	1.49	108866	2522388				
dm-8	0.00	0.00	0.00	5325	0				
[root@ContainerMDB ~]#									

Estos serían los datos en el caso de usar iostat, el cual no funcionaba hasta que no realice la instalación del “paquete” de comandos. “sudo yum install sysstat -y”

Obtenga la información referente a la capacidad de la red.

Para esto haremos uso del comando netstat -s

[root@Sysbench ~]# netstat -s		[root@ContainerMDB ~]# netstat -s	
Ip:		Ip:	
1660289 total packets received		1662143 total packets received	
2 with invalid addresses		1 with invalid addresses	
0 forwarded		0 forwarded	
0 incoming packets discarded		0 incoming packets discarded	
13070 incoming packets delivered		14597 incoming packets delivered	
9494 requests sent out		9016 requests sent out	
Icmp:		Icmp:	
0 ICMP messages received		0 ICMP messages received	
0 input ICMP message failed.		0 input ICMP message failed.	
ICMP input histogram:		ICMP input histogram:	
0 ICMP messages sent		0 ICMP messages sent	
0 ICMP messages failed		0 ICMP messages failed	
ICMP output histogram:		ICMP output histogram:	
Tcp:		Tcp:	
353 active connections openings		74 active connections openings	
4 passive connection openings		3 passive connection openings	
0 failed connection attempts		0 failed connection attempts	
7 connection resets received		2 connection resets received	
1 connections established		1 connections established	
11937 segments received		13774 segments received	
8522 segments send out		8418 segments send out	
0 segments retransmitted		1 segments retransmitted	
1 bad segments received.		0 bad segments received.	
55 resets sent		0 resets sent	
InSumErrors: 1		Udp:	
Udp:		605 packets received	
927 packets received		0 packets to unknown port received.	
0 packets to unknown port received.		0 packet receive errors	
0 packet receive errors		599 packets sent	
922 packets sent		0 receive buffer errors	
0 receive buffer errors		0 send buffer errors	
0 send buffer errors		IgnoredMulti: 218	
IgnoredMulti: 206		UdpLite:	
UdpLite:		TcpExt:	
TcpExt:		28 TCP sockets finished time wait in fast timer	
307 TCP sockets finished time wait in fast timer		3 delayed acks sent	
15 delayed acks sent		Quick ack mode was activated 47 times	
8611 packet headers predicted		12162 packet headers predicted	
525 acknowledgments not containing data payload received		308 acknowledgments not containing data payload received	
524 predicted acknowledgments		235 predicted acknowledgments	
1 congestion windows recovered without slow start by DSACK		Detected reordering 2 times using SACK	
1 other TCP timeouts		TCPLOSSProbes: 1	
TCPLOSSProbes: 5		TCPLOSSProbeRecovery: 1	
TCPBacklogCoalesce: 1		47 DSACKs sent for old packets	
5 DSACKs received		TCP sack shift fallback: 2	
21 connections reset due to unexpected data		TCPOFQueue: 406	
5 connections reset due to early user close		TCPAutoCorking: 17	
IPReversePathFilter: 1		TCPOrigDataSent: 713	
TCPRecvCoalesce: 6		TCPDelivered: 779	
TCPOFQueue: 895		TCPAckCompressed: 236	
TCPAutoCorking: 57		IpExt:	
TCPOrigDataSent: 1708		InBcastPkts: 224	
TCPDelivered: 2058		InOctets: 157743913	
TCPAckCompressed: 811		OutOctets: 601434	
IpExt:		InBcastOctets: 73605	
InBcastPkts: 211		InNoECTPkts: 1693616	
InOctets: 136363263		[root@ContainerMDB ~]#	
OutOctets: 765845			
InBcastOctets: 69341			
InNoECTPkts: 1681203			
[root@Sysbench ~]#			

1.3 Monitorización del servidor

Determine los procesos activos y la relación que existe entre ellos.
Compruebe que el servidor Mariadb esté activo.

```
[root@ContainerMDB ~]# systemctl is-active mariadb
unknown
[root@ContainerMDB ~]# systemctl start mariadb.service
[root@ContainerMDB ~]# systemctl is-active mariadb
active
[root@ContainerMDB ~]#
```

Determine el consumo de los diferentes recursos identificando los procesos que más consuman.

```
[root@ContainerMDB ~]# systemctl start mariadb.service
[root@ContainerMDB ~]# systemctl is-active mariadb
active
[root@ContainerMDB ~]# ps aux
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  1.0 43568 5396 ?        Ss   Sep28   0:16 /usr/lib/systemd/systemd --system --deserialize 14
root        40  0.0  1.2 39092 6728 ?        Ss   Sep28   0:08 /usr/lib/systemd/systemd-journald
dbus        59  0.0  0.8 58120 4288 ?        Ss   Sep28   0:02 /usr/bin/dbus-daemon --system --address=systemd: --noforroot
logind      85  0.0  0.3 6524 1692 console Ss+  Sep28   0:00 /sbin/agetty --noclear --keep-baud console 115200,38400root
root      87  0.0  0.8 90968 4692 ?        Ss   Sep28   0:00 login -- root
root        90  0.0  0.5 22736 2660 ?        Ss   Sep28   0:05 /usr/sbin/crond -n
root       244  0.0  0.9 102904 4784 ?        Ss   Sep28   0:01 /sbin/dhclient -1 -q -lf /var/lib/dhclient/dhclient--etroot
root       306  0.0  1.4 218556 7424 ?        Ssl  Sep28   2:52 /usr/sbin/rsyslogd -n
root       311  0.0  0.5 11832 3832 tty1     Ss   Sep28   0:00 bash
root       332  0.0  1.3 133448 7284 tty1     Ss   Sep28   0:00 sudo su
root       333  0.0  0.7 81992 4872 tty1     Ss   Sep28   0:00 su
root       334  0.0  0.5 11832 3844 tty1     Ss   Sep28   0:00 bash
root     13588  0.0  1.8 155352 9676 ?        Ss   06:34   0:00 sshd: root@pts/2
root     13590  0.0  0.5 11836 3836 pts/2    Ss+  06:34   0:00 -bash
root     13843  0.0  1.8 155352 9788 ?        Rss  08:51   0:00 sshd: root@pts/3
root     13845  0.0  0.5 11836 3840 pts/3    Ss   08:51   0:00 bash
mysql     13846  0.0  0.4 9708 2568 ?        Ss   08:53   0:00 /bin/sh /usr/bin/mysqld_safe --basedir=/usr
mysql     14110  0.0 16.3 969004 85844 ?       Sl   08:53   0:00 /usr/libexec/mysqld --basedir=/usr --datadir=/var/lib/mysql
[root@ContainerMDB ~]#
```

Verifique si existen clientes de red conectados.

```
[root@ContainerMDB ~]# netstat
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 ContainerMDB:ssh        10.8.2.7:53371          ESTABLISHED
tcp        0      0 ContainerMDB:ssh        10.8.2.7:57532          ESTABLISHED
Active UNIX domain sockets (w/o servers)
Proto RefCnt Flags               Type               I-Node  Path
unix   10      [ ] DGRAM              11174775 /dev/log
unix    3      [ ] DGRAM              11174767 /run/systemd/notify
unix    2      [ ] DGRAM              11174768 /run/systemd/cgroups-agent
unix    2      [ ] DGRAM              11174772 /run/systemd/shutdown
unix    4      [ ] DGRAM              11174774 /run/systemd/journal/socket
unix    3      [ ] STREAM             11177891
unix    3      [ ] STREAM             11176462 /run/systemd/journal/stdout
unix    2      [ ] DGRAM              49877135
unix    2      [ ] DGRAM              11177179
unix    3      [ ] STREAM             11176480 /run/systemd/journal/stdout
unix    2      [ ] DGRAM              11174861
unix    3      [ ] STREAM             11176488 /run/dbus/system_bus_socket
unix    3      [ ] STREAM             51609965
unix    3      [ ] STREAM             11177185
unix    3      [ ] STREAM             51609966 /run/dbus/system_bus_socket
unix    3      [ ] STREAM             11177172
unix    3      [ ] STREAM             11176747 /run/systemd/journal/stdout
unix    2      [ ] DGRAM              11182065
unix    2      [ ] DGRAM              51781265
unix    3      [ ] STREAM             11177384
unix    2      [ ] DGRAM              51611079
unix    2      [ ] DGRAM              11176788
unix    3      [ ] STREAM             11176455
unix    2      [ ] DGRAM              51581733
unix    3      [ ] STREAM             11176415
unix    2      [ ] DGRAM              11183126
unix    3      [ ] STREAM             11176456
unix    2      [ ] DGRAM              11176501
unix    2      [ ] DGRAM              11176622
unix    3      [ ] STREAM             11176416 /run/systemd/journal/stdout
[root@ContainerMDB ~]#
```

En las conexiones a internet, todas salen en estado ESTABLISHED , el cual corresponde como conexión establecidas. Por lo que todas las redes están conectadas y disponibles.

Ejecute el comando **iostat** con la opción **-c** para comprobar el estado de uso de la CPU. Haga la monitorización a espacios regulares de 2 segundos durante 20 segundos. Eluso de la CPU, ¿sufré cambios significativos?

```
[root@ContainerMDB ~]# iostat -c 2
Linux 5.4.106-1-pve (ContainerMDB)      10/18/23      _x86_64_      (4 CPU)

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           0.01    0.00    0.00    0.00    0.00   99.99

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           0.00    0.00    0.00    0.00    0.00  100.00

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           0.00    0.00    0.00    0.00    0.00  100.00

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           0.25    0.00    0.00    0.00    0.00   99.75

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           0.00    0.00    0.00    0.00    0.00  100.00

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           0.00    0.00    0.00    0.00    0.00  100.00

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           0.25    0.00    0.00    0.00    0.00   99.75

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           0.25    0.00    0.00    0.00    0.00   99.75

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           0.00    0.00    0.00    0.00    0.00  100.00

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           0.00    0.00    0.00    0.00    0.00  100.00

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           0.00    0.00    0.00    0.00    0.00  100.00

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           0.00    0.00    0.00    0.00    0.00  100.00

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           0.25    0.00    0.00    0.00    0.00   99.75

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           0.00    0.00    0.00    0.00    0.00  100.00

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           0.00    0.00    0.00    0.00    0.00  100.00
```

Como podremos comprobar en la captura siguiente, el servidor está casi en reposo porque no sufre grandes cambios muy significativos. Algunas veces está durante muy poco tiempo en modo usuario, pero casi siempre la CPU no está gestionando ninguna solicitud.

Mediante la opción **-d** del comando **iostat**, verifique a periodos regulares de 2 segundos el estado de las operaciones de entrada y salida.

```
[root@ContainerMDB ~]# iostat -d 2
Linux 5.4.106-1-pve (ContainerMDB)      10/18/23      _x86_64_      (4 CPU)
```

Device:	tps	kB_read/s	kB_wrtn/s	kB_read	kB_wrtn
sdd0	0.00	0.00	0.00	2096	0
sda	2.84	104.92	21.62	178671535	36820350
dm-0	0.00	0.00	0.00	3264	4
dm-1	2.21	0.17	18.95	283197	32273844
dm-2	0.00	0.00	0.01	1304	16424
dm-3	0.60	0.61	3.74	1036143	6375968
dm-4	0.60	0.61	3.74	1036143	6375968
dm-6	0.35	0.55	1.96	932593	3333628
dm-7	0.25	0.06	1.49	108930	2530564
dm-8	0.00	0.00	0.00	5325	0

Device:	tps	kB_read/s	kB_wrtn/s	kB_read	kB_wrtn
sdd0	0.00	0.00	0.00	0	0
sda	2.00	0.00	16.00	0	32
dm-0	0.00	0.00	0.00	0	0
dm-1	4.00	0.00	16.00	0	32
dm-2	0.00	0.00	0.00	0	0
dm-3	0.50	0.00	2.00	0	4
dm-4	0.50	0.00	2.00	0	4
dm-6	0.00	0.00	0.00	0	0
dm-7	0.50	0.00	2.00	0	4
dm-8	0.00	0.00	0.00	0	0

Device:	tps	kB_read/s	kB_wrtn/s	kB_read	kB_wrtn
sdd0	0.00	0.00	0.00	0	0
sda	5.50	384.00	12.00	768	24
dm-0	0.00	0.00	0.00	0	0
dm-1	2.00	0.00	10.00	0	20
dm-2	0.00	0.00	0.00	0	0
dm-3	0.50	0.00	2.00	0	4
dm-4	0.50	0.00	2.00	0	4
dm-6	0.50	0.00	2.00	0	4
dm-7	0.00	0.00	0.00	0	0
dm-8	0.00	0.00	0.00	0	0

Device:	tps	kB_read/s	kB_wrtn/s	kB_read	kB_wrtn
sdd0	0.00	0.00	0.00	0	0
sda	1.50	0.00	6.00	0	12
dm-0	0.00	0.00	0.00	0	0
dm-1	2.00	0.00	8.00	0	16
dm-2	0.00	0.00	0.00	0	0
dm-3	0.00	0.00	0.00	0	0
dm-4	0.00	0.00	0.00	0	0
dm-6	0.00	0.00	0.00	0	0
dm-7	0.00	0.00	0.00	0	0
dm-8	0.00	0.00	0.00	0	0

Device:	tps	kB_read/s	kB_wrtn/s	kB_read	kB_wrtn
sdd0	0.00	0.00	0.00	0	0
sda	1.50	0.00	12.00	0	24
dm-0	0.00	0.00	0.00	0	0
dm-1	1.50	0.00	10.00	0	20
dm-2	0.00	0.00	0.00	0	0
dm-3	0.50	0.00	2.00	0	4
dm-4	0.50	0.00	2.00	0	4
dm-6	0.00	0.00	0.00	0	0
dm-7	0.50	0.00	2.00	0	4
dm-8	0.00	0.00	0.00	0	0

Device:	tps	kB_read/s	kB_wrtn/s	kB_read	kB_wrtn
sdd0	0.00	0.00	0.00	0	0
sda	0.50	0.00	2.00	0	4
dm-0	0.00	0.00	0.00	0	0
dm-1	0.00	0.00	0.00	0	0
dm-2	0.00	0.00	0.00	0	0
dm-3	0.50	0.00	2.00	0	4
dm-4	0.50	0.00	2.00	0	4
dm-6	0.50	0.00	2.00	0	4
dm-7	0.00	0.00	0.00	0	0
dm-8	0.00	0.00	0.00	0	0

Use el comando **w**, para obtener la carga de la CPU en los últimos 1, 5 y 15 minutos.
¿Se encuentra el sistema sobrecargado?

```
[root@ContainerMDB ~]# w
 09:43:17 up 19 days, 17:35,  3 users,  load average: 0.00, 0.00, 0.00
USER      TTY      FROM            LOGIN@   IDLE   JCPU   PCPU WHAT
root      tty1                28Sep23 15days  0.03s  0.01s  bash
root      pts/2    10.8.2.7        09:11   31:50  0.00s  0.00s  -bash
root      pts/3    10.8.2.7        09:40    0.00s  0.00s  0.00s  w
[root@ContainerMDB ~]#
```

Como podemos apreciar en la imagen la CPU no se encuentra sobrecargada en ningún momento.

El comando **iostat** no devuelve información sobre la carga específica de una CPU, para estos casos usamos el comando **mpstat**. Compare la salida de ambos comandos.

Para la comparación he buscado alguna información acerca de los comandos, la cual adjunto para ayudarnos.

%irq: muestra el porcentaje de tiempo empleado por la CPU o las CPU para dar servicio a las interrupciones de hardware.

%soft: muestra el porcentaje de tiempo empleado por la CPU o las CPU para reparar las interrupciones del software

%guest: muestra el porcentaje de tiempo empleado por la CPU o las CPU para ejecutar un procesador virtual

```
[root@ContainerMDB ~]# mpstat 2
Linux 5.4.106-1-pve (ContainerMDB)      10/18/23      _x86_64_      (4 CPU)

10:35:01   CPU    %usr   %nice    %sys %iowait    %irq   %soft  %steal  %guest  %gnice   %idle
10:35:03   all     0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00   100.00
10:35:05   all     0.25    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00   99.75
10:35:07   all     0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00   100.00
10:35:09   all     0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00   100.00
10:35:11   all     0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00   100.00
10:35:13   all     0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00   100.00
10:35:15   all     0.25    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00   99.75
10:35:17   all     0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00   100.00
10:35:19   all     0.25    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00   99.75
10:35:21   all     0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00   100.00
10:35:23   all     0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00   100.00
10:35:25   all     0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00   100.00
10:35:27   all     0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00   100.00
10:35:29   all     0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00   100.00
10:35:31   all     0.25    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00   99.75
10:35:33   all     0.25    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00   99.75
10:35:35   all     0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00   100.00
```

La utilidad **sar** (system activity report"), recopila información de distintos parámetros del sistema. Esta aplicación incluye dos shell scripts. El primer script, sa1,recopila datos de forma regular, mientras que el script sa2 se utiliza para crear los informes resumidos (uno por día en **/var/log/sa/sarDD**). Ambos scripts se ejecutan usando cron. Si queremos obtener los datos en tiempo real podemos invocar directamente al comando sar.

```
[root@ContainerMDB ~]# sar -u 2
Linux 5.4.106-1-pve (ContainerMDB)      10/18/23      _x86_64_      (4 CPU)

10:48:30      CPU      %user      %nice      %system      %iowait      %steal      %idle
10:48:32      all       0.00       0.00       0.00       0.00       0.00      100.00
10:48:34      all       0.25       0.00       0.00       0.00       0.00      99.75
10:48:36      all       0.00       0.00       0.00       0.00       0.00     100.00
10:48:38      all       0.25       0.00       0.00       0.00       0.00      99.75
10:48:40      all       0.00       0.00       0.00       0.00       0.00     100.00
10:48:42      all       0.00       0.00       0.00       0.00       0.00     100.00
10:48:44      all       0.25       0.00       0.00       0.00       0.00      99.75
10:48:46      all       0.00       0.00       0.00       0.00       0.00     100.00
10:48:48      all       0.00       0.00       0.00       0.00       0.00     100.00
10:48:50      all       0.50       0.00       0.00       0.00       0.00      99.50
10:48:52      all       0.00       0.00       0.00       0.00       0.00     100.00
10:48:54      all       0.25       0.00       0.00       0.00       0.00      99.75
10:48:56      all       0.00       0.00       0.00       0.00       0.00     100.00
10:48:58      all       0.00       0.00       0.00       0.00       0.00     100.00
10:49:00      all       0.25       0.00       0.00       0.00       0.00      99.75
```

1.4 Monitorización del servidor ante situaciones de estrés

Cree el esquema de base de datos sbtest y dótele de 5 tablas con 50000 filas cada una

Todos los comandos que pondré en los siguientes apartados y en este se ejecutaron en sysbench aunque los cambios se hicieron en Mariadb.

```
/usr/share/sysbench/oltp_read_write.lua --threads=4 --mysql-host=192.168.50.176  
--mysql-user=root --mysql-password=**** --mysql-port=3306 --tables=5 --table-  
size=50000 prepare
```

Aquí se puede ver que la ip de mariadb es 192.168.50.27 y que el usuario donde se crearon las tablas era el root.

En la siguiente captura podemos ver el nivel de estrés de la base de datos de mariadb mediante el comando top.

```
root@Sysbench:~  
top - 16:12:31 up 20 days, 23:59, 2 users, load average: 1.04, 1.14, 1.10  
Tasks: 17 total, 1 running, 16 sleeping, 0 stopped, 0 zombie  
%Cpu(s): 0.0 us, 0.0 sy, 0.0 ni,100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st  
KiB Mem : 524288 total, 235424 free, 85880 used, 202984 buff/cache  
KiB Swap: 1048576 total, 1048576 free, 0 used. 438408 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1	root	20	0	43628	5484	4100	S	0.0	1.0	0:15.65	systemd
39	root	20	0	39092	7328	7060	S	0.0	1.4	0:09.01	systemd-journal
57	root	20	0	26868	3484	2652	S	0.0	0.7	0:08.74	systemd-logind
58	dbus	20	0	58116	4256	3672	S	0.0	0.8	0:03.62	dbus-daemon
64	root	20	0	22736	2744	2088	S	0.0	0.5	0:06.12	crond
65	root	20	0	6524	1712	1588	S	0.0	0.3	0:00.00	agetty
66	root	20	0	90968	4784	4136	S	0.0	0.9	0:00.00	login
67	root	20	0	6524	1764	1640	S	0.0	0.3	0:00.00	agetty
276	root	20	0	102904	4692	2628	S	0.0	0.9	0:01.82	dhclient
337	root	20	0	112928	7740	6716	S	0.0	1.5	0:00.02	sshd
338	root	20	0	218556	7536	6816	S	0.0	1.4	2:59.92	rsyslogd
343	root	20	0	11832	3040	2636	S	0.0	0.6	0:00.03	bash
15874	root	20	0	155352	9768	8420	S	0.0	1.9	0:00.19	sshd
15876	root	20	0	11836	2996	2592	S	0.0	0.6	0:00.06	bash
15900	root	20	0	155488	10496	9140	S	0.0	2.0	0:01.25	sshd
15902	root	20	0	72388	4860	3984	S	0.0	0.9	0:01.52	sftp-server
15942	root	20	0	56212	3836	3300	R	0.0	0.7	0:00.00	top

En la imagen anterior se puede apreciar que la memoria apenas se usa, y ya ni hablemos de la Cpu pero me equivoque de contenedor al usar el comando top la primera vez y no pude comprobar correctamente el consumo. Debería dar unos valores alrededor de 35% de memoria y cerca del 1% de cpu si no me equivoco.

Haga un test de tipo oltp_read_only durante 150 segundos empleando 3 hebras.

En este ejercicio el comando que utilicé fue:

```
/usr/share/sysbench/oltp_read_only.lua --threads=3 --events=0 --time=150 --mysql-host=192.168.50.176 --mysql-user=root --mysql-password=**** --mysql-port=3306 --tables=5 --table-size=50000 --range-selects=off --db-ps-mode=disable --report-interval=1 run
```

The image contains two terminal screenshots. The left screenshot shows the output of the 'top' command in a container named 'ContainerMDB'. It displays system statistics and a list of processes. The 'mysql' process (PID 14110) is shown with 169.0% CPU usage and 47.0% memory usage. The right screenshot shows the output of the 'sysbench' command in a container named 'Sysbench'. It displays a series of performance metrics over 69 seconds, showing a steady increase in transactions per second (tps) from approximately 2619 to 2697.

```
root@ContainerMDB:~
top - 16:18:28 up 21 days, 11 min,  2 users,  load average: 3.39, 1.79, 1.33
Tasks: 20 total,  1 running, 19 sleeping,  0 stopped,  0 zombie
%Cpu(s): 69.3 us,  0.0 sy,  0.0 ni, 30.7 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem : 524288 total,  364 free, 267016 used, 256908 buff/cache
KiB Swap: 1048576 total, 1048572 free,  4 used. 257272 avail Mem

  PID USER      PR  NI   VIRT   RES   SHR S  %CPU  %MEM    TIME+  COMMAND
14110 mysql    20   0 1253848 246368 14836 S 169.0  47.0   28:39.07 mysql
   1 root      20   0  43568   5388   4016 S   0.0   1.0    0:18.19 systemd
  40 root      20   0  39092   7228   6944 S   0.0   1.4    0:09.13 systemd-journal
  59 dbus      20   0  58120   4232   3656 S   0.0   0.8    0:03.73 dbus-daemon
  82 root      20   0  26868   3256   2428 S   0.0   0.6    0:08.77 systemd-logind
  85 root      20   0   6524   1652   1528 S   0.0   0.3    0:00.00 agetty
  86 root      20   0   6524   1676   1548 S   0.0   0.3    0:00.00 agetty
  87 root      20   0  90968   4636   3988 S   0.0   0.9    0:00.03 login
  90 root      20   0  22736   2660   2012 S   0.0   0.5    0:06.45 crond
 244 root      20   0 102904   4196   2136 S   0.0   0.8    0:02.00 dhclient
 305 root      20   0 112928   7836   6808 S   0.0   1.5    0:00.04 sshd
 306 root      20   0 218556   7508   6576 S   0.0   1.4   3:02.75 rsyslogd
 311 root      20   0  11832   3032   2632 S   0.0   0.6    0:00.01 bash
 332 root      20   0 133448   6848   5716 S   0.0   1.3    0:00.01 sudo
 333 root      20   0  81992   4024   3484 S   0.0   0.8    0:00.00 su
 334 root      20   0  11832   3044   2624 S   0.0   0.6    0:00.02 bash
13946 mysql    20   0   9708   2568   2280 S   0.0   0.5    0:00.01 mysqld_safe
16123 root      20   0 155352   9792   8440 S   0.0   1.9    0:00.05 sshd
16125 root      20   0  11836   2936   2672 S   0.0   0.6    0:00.00 bash
16139 root      20   0  56208   3848   3304 R   0.0   0.7    0:00.01 top

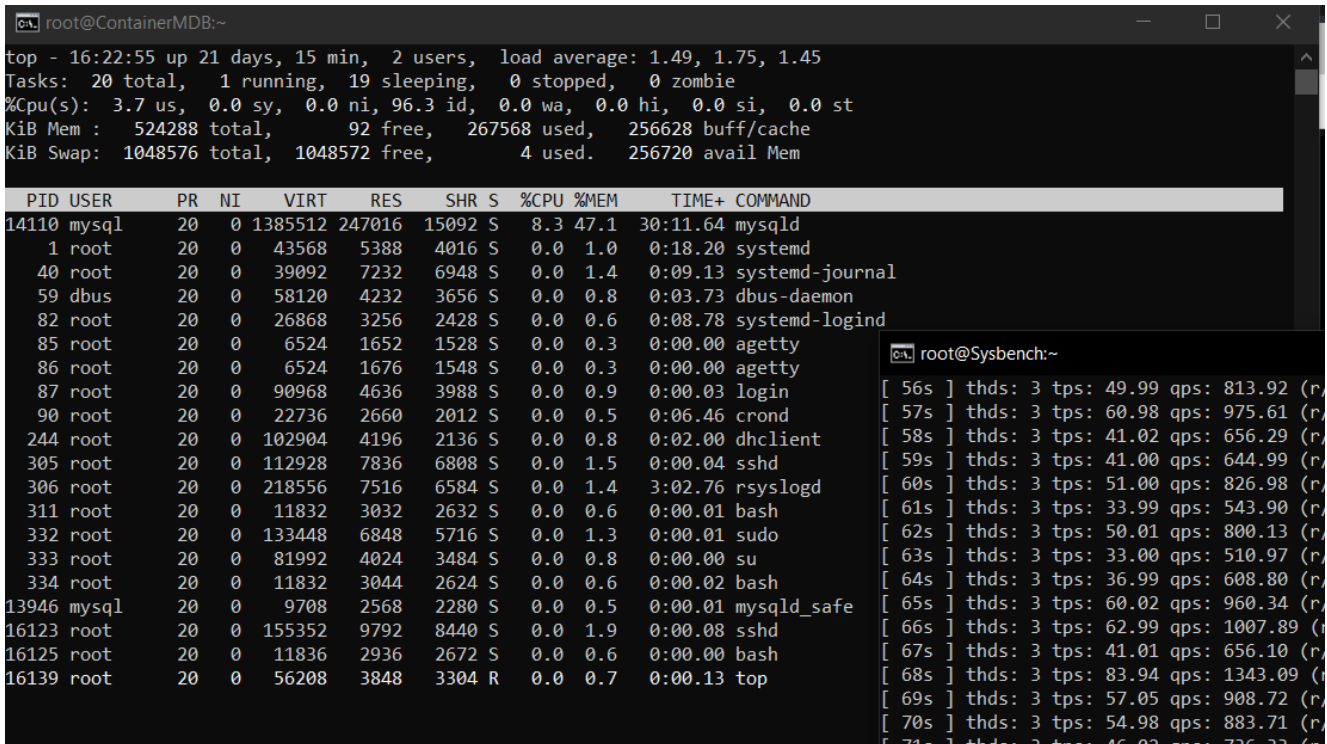
root@Sysbench:~
[ 42s ] thds: 3 tps: 2619.84 qps: 31
[ 43s ] thds: 3 tps: 2432.01 qps: 29
[ 44s ] thds: 3 tps: 2594.05 qps: 31
[ 45s ] thds: 3 tps: 2581.52 qps: 30
[ 46s ] thds: 3 tps: 2406.80 qps: 28
[ 47s ] thds: 3 tps: 2277.60 qps: 27
[ 48s ] thds: 3 tps: 2255.11 qps: 27
[ 49s ] thds: 3 tps: 2318.96 qps: 27
[ 50s ] thds: 3 tps: 2365.26 qps: 28
[ 51s ] thds: 3 tps: 2427.78 qps: 29
[ 52s ] thds: 3 tps: 2417.99 qps: 29
[ 53s ] thds: 3 tps: 2355.95 qps: 28
[ 54s ] thds: 3 tps: 2269.39 qps: 27
[ 55s ] thds: 3 tps: 2304.84 qps: 27
[ 56s ] thds: 3 tps: 2392.16 qps: 28
[ 57s ] thds: 3 tps: 2596.01 qps: 31
[ 58s ] thds: 3 tps: 2281.99 qps: 27
[ 59s ] thds: 3 tps: 2317.88 qps: 27
[ 60s ] thds: 3 tps: 2379.96 qps: 28
[ 61s ] thds: 3 tps: 2301.50 qps: 27
[ 62s ] thds: 3 tps: 2351.24 qps: 28
[ 63s ] thds: 3 tps: 2269.43 qps: 27
[ 64s ] thds: 3 tps: 2343.93 qps: 28
[ 65s ] thds: 3 tps: 2398.39 qps: 28
[ 66s ] thds: 3 tps: 2528.05 qps: 30
[ 67s ] thds: 3 tps: 2427.99 qps: 29
[ 68s ] thds: 3 tps: 2603.97 qps: 31
[ 69s ] thds: 3 tps: 2697.99 qps: 32
```

En este podemos observar cómo la CPU ha pasado de estar en casi reposo a un 172,4% aunque la memoria se ha mantenido casi igual que antes.

Haga un test de tipo oltp_read_write durante 150 segundos empleando 3 hebras.

Por último, el comando aquí era el mismo que en el otro pero esta vez era de lectura y escritura:

```
/usr/share/sysbench/oltp_read_write.lua --threads=3 --events=0 --time=150 --mysql-host=192.168.50.176 --mysql-user=root --mysql-password=**** --mysql-port=3306 --tables=5 --table-size=50000 --range_selects=off --db-ps-mode=disable --report-interval=1 run
```



The screenshot shows two terminal windows. The left window, titled 'root@ContainerMDB:~', displays the output of the 'top' command, showing system load and a list of running processes. The right window, titled 'root@Sysbench:~', shows the output of the sysbench test, displaying performance metrics over time.

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
14110	mysql	20	0	1385512	247016	15092	S	8.3	47.1	30:11.64	mysqld
1	root	20	0	43568	5388	4016	S	0.0	1.0	0:18.20	systemd
40	root	20	0	39092	7232	6948	S	0.0	1.4	0:09.13	systemd-journal
59	dbus	20	0	58120	4232	3656	S	0.0	0.8	0:03.73	dbus-daemon
82	root	20	0	26868	3256	2428	S	0.0	0.6	0:08.78	systemd-logind
85	root	20	0	6524	1652	1528	S	0.0	0.3	0:00.00	agetty
86	root	20	0	6524	1676	1548	S	0.0	0.3	0:00.00	agetty
87	root	20	0	90968	4636	3988	S	0.0	0.9	0:00.03	login
90	root	20	0	22736	2660	2012	S	0.0	0.5	0:06.46	crond
244	root	20	0	102904	4196	2136	S	0.0	0.8	0:02.00	dhclient
305	root	20	0	112928	7836	6808	S	0.0	1.5	0:00.04	sshd
306	root	20	0	218556	7516	6584	S	0.0	1.4	3:02.76	rsyslogd
311	root	20	0	11832	3032	2632	S	0.0	0.6	0:00.01	bash
332	root	20	0	133448	6848	5716	S	0.0	1.3	0:00.01	sudo
333	root	20	0	81992	4024	3484	S	0.0	0.8	0:00.00	su
334	root	20	0	11832	3044	2624	S	0.0	0.6	0:00.02	bash
13946	mysql	20	0	9708	2568	2280	S	0.0	0.5	0:00.01	mysqld_safe
16123	root	20	0	155352	9792	8440	S	0.0	1.9	0:00.08	sshd
16125	root	20	0	11836	2936	2672	S	0.0	0.6	0:00.00	bash
16139	root	20	0	56208	3848	3304	R	0.0	0.7	0:00.13	top

Time	thds	tps	qps
56s	3	49.99	813.92
57s	3	60.98	975.61
58s	3	41.02	656.29
59s	3	41.00	644.99
60s	3	51.00	826.98
61s	3	33.99	543.90
62s	3	50.01	800.13
63s	3	33.00	510.97
64s	3	36.99	608.80
65s	3	60.02	960.34
66s	3	62.99	1007.89
67s	3	41.01	656.10
68s	3	83.94	1343.09
69s	3	57.05	908.72
70s	3	54.98	883.71
71s	3	46.02	736.33

En este caso baja considerablemente el consumo de cpu pero aumenta levemente el consumo de memoria.

1.5 Conclusión.

Para Finalizar en esta primera practica hemos podido aprender la creación de contenedores en ProxMox e instalar nuestro entorno de trabajo en el mismo. También hemos aprendido a como se encuentra/monitoriza una base de datos tanto en estado de reposo como en estado de estrés. Esto último permitiéndonos ver como respondo a estas circunstancias.

En conclusión, lo que puedo apreciar es que no es bueno ninguno de los extremos, porque si tenemos el servidor siempre en casi reposo quiere decir que estamos malgastando recursos y dinero en nuestro entorno de trabajo, lo cual no es bueno ni a nivel personal si es un entorno propio, ni a nivel de empresa. Y tampoco es nada bueno que nuestro servidor se encuentre siempre en estado de estrés exprimiéndole hasta el último recurso, si este fuera el caso durante bastante tiempo y no en momentos puntuales deberíamos hacer un estudio y ver que necesitamos ampliar del mismo para

un funcionamiento correcto de este.



Universidad
de Huelva



Escuela Técnica Superior de Ingeniería
Universidad de Huelva

Memoria de Prácticas

ADMINISTRACIÓN DE SERVIDORES

Grado en Ingeniería Informática

Autor: Alejandro Gordillo Pedraza

6 de noviembre de 2023

Resumen

Esta práctica consiste en crear una Máquina Virtual (VM) en ProxMox e instalarle CentOS7. Una vez tengamos instalada esta variante de Linux deberemos cambiar el kernel que nos viene de serie por uno de nuestra elección. En nuestro caso elegiremos uno un poco más actual (4.10).

Antes de elegir este kernel empezó probando uno de los mas actuales 6.* pero no conseguí configurarlo tras varios intentos y varias máquinas, así que probe con uno que fuera levemente mas actual que el que teníamos de default y conseguimos avanzar en nuestro aprendizaje.

Índice general

Contenido

1.1 Creación el entorno de pruebas.....	4
1.2 Una vez creada la maquina virtual, instale CentOS 7	4
1.3 Comprobar SSH.....	5
1.4 Instalando un núcleo personalizado.....	5
1.5 Verifican la identidad e integridad del núcleo descargado antes de proceder al desempaquetado.	6
1.6 Desempaque el nuevo núcleo en el directorio.....	6
1.7 Prepare el fichero de configuración tomando como referencia alguno de los ya instalados. 7	
1.8 Compile el núcleo y sus módulos.	7
1.9 Instale el núcleo y sus módulos.	8
1.10 Haga una copia del núcleo y RAM Disk actuales. Los nuevos ficheros se llamarán como los antiguos añadiendo un prefijo que identifique al alumno.	9
1.11 Creación del RAM Disk.....	9
1.12 Gestión de módulos.....	11
1.13 Compilación de módulos de terceros (optativa).....	14
1.14 Conclusión.....	15

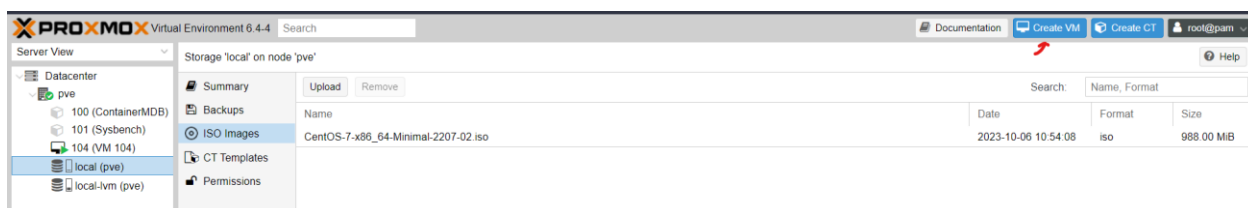
Capítulo 1

Instalación de CentOS 7

1.1 Creación el entorno de pruebas

Cree una máquina virtual completa, con las características antes indicadas, en la instancia de Proxmox creada para cada estudiante.

Antes de crear una máquina virtual tenemos que descargarnos la imagen ISO en internet (Yo me descargué la versión minimalista de 2009). Una vez se descarga por completo, nos vamos a Proxmox y vamos a las direcciones que están en azul y una vez ahí le damos a Upload y subimos la imagen



Una vez hecho esto, le damos a Create VM y la configuramos con 1 Gb de Ram, 2 CPU y 32 GB de disco duro.

1.2 Una vez creada la maquina virtual, instale CentOS 7

Una vez creada nuestra VM con las especificaciones que nos pedían, nos aparecerá un menú con 3 opciones, elegiremos la opción de **Test this media to install CentOS 7**.

Seguidamente nos realizará un test y nos abrirá una interfaz grafica de instalación, la cual se ira colocando automáticamente todo menos unos datos que deberemos poner manualmente.

1º Destino de instalación, aquí podremos seleccionar en que disco duro queremos realizar la instalación o como hacer las particiones de nuestro disco, en nuestro caso como solo tenemos un disco y queremos que las particiones las haga automáticas simplemente abrimos la opción y damos a listo.

2º Conexión Ethernet, debemos abrir esta opción y activar la conexión LAN, debido que si omitimos este paso cada vez que arranquemos nuestra maquina deberemos meter el comando **eth 0** para poder tener acceso a internet.

3º Una vez que corregimos esas dos opciones le daremos a siguiente o seguir con la instalación, y nos pedirá una contraseña para el root y un usuario (opcional). **IMPORTANTE** acordarse de la contraseña del root pues la necesitaremos cada vez que arranquemos nuestra máquina.

Finalmente, solo quedaría esperar a que termine la instalación de nuestro sistema. Una vez que termine se nos iniciara una línea de comandos para poder trabajar sobre ella.

1.3 Comprobar SSH

Compruebe que el servicio ssh se encuentra activo y configurado adecuadamente. En caso de no estarlo, siga lo indicado en “How to Enable, Install, & Configure SSH on CentOS 7 | PhoenixNAP KB.”

Si hemos activado la conexión a ethernet en los pasos anteriores deberíamos tener el ssh activado por default.

Systemctl status sshd

```
[root@localhost ~]# systemctl status sshd
■ sshd.service - OpenSSH server daemon
   Loaded: loaded (/usr/lib/systemd/system/sshd.service; enabled; vendor preset: enabled)
   Active: active (running) since lun 2023-10-30 09:50:22 CET; 24s ago
     Docs: man:sshd(8)
           man:sshd_config(5)
   Main PID: 1012 (sshd)
    CGroup: /system.slice/sshd.service
            └─1012 /usr/sbin/sshd -D

oct 30 09:50:22 localhost.localdomain systemd[1]: Starting OpenSSH server daemon...
oct 30 09:50:22 localhost.localdomain sshd[1012]: Server listening on 0.0.0.0 port 22.
oct 30 09:50:22 localhost.localdomain sshd[1012]: Server listening on :: port 22.
oct 30 09:50:22 localhost.localdomain systemd[1]: Started OpenSSH server daemon.
[root@localhost ~]#
```

En caso de no estar activado deberemos activarlo haciendo uso del siguiente comando:

systemctl start sshd

1.4 Instalando un núcleo personalizado

Descargar el código fuente de una versión del núcleo igual o superior a la que ahora mismo está ejecutando.

Para saber que núcleo tiene nuestro CentOS basta con poner:

uname -r

```
[root@localhost ~]# cd /usr/src
[root@localhost src]# uname -r
3.10.0-1160.71.1.el7.x86_64
[root@localhost src]#
```

Con esto, sabemos que nuestra versión del núcleo es la 3.10.0 y descargamos otra con los siguientes pasos:

Primero nos vamos al directorio usr/src con el comando `cd usr/src`.

Luego descargamos el fichero con el código fuente y el fichero de firma con los comandos:

wget https://mirrors.edge.kernel.org/pub/linux/kernel/v4.x/linux-4.10.tar.xz

wget https://mirrors.edge.kernel.org/pub/linux/kernel/v4.x/linux-4.10.tar.sign

En el caso de no tener instalada la funcionalidad wget, debemos usar el comando **yum install wget**

1.5 Verifican la identidad e integridad del núcleo descargado antes de proceder al desempaquetado.

Para verificar que el núcleo es fiable confirmamos la firma del autor, para ello utilizaremos el siguiente comando:

```
gpg --verify linux-4.10.tar.sign
```

Donde en el comando Linux-4.10.tar.sign es nuestro kernel seleccionado y descargado.

Si nos falla el comando es porque la clave no esta disponible y necesitaremos comprobarlo con la clave pública:

```
gpg --recv-keys <ID_de_la_clave>
```

1.6 Desempaque el nuevo núcleo en el directorio.

Para realizar este paso hay que ver que tipo de empaquetado tiene nuestro kernel, en nuestro caso tenemos .tar.xz así que los descomprimiremos con el siguiente comando:

```
tar xvf Linux-4.10.tar.xz -C /usr/src/
```

Este comando nos permite descomprimir los tipos de empaquetado con un solo comando y nos lo extrae en el directorio /usr/src, que es donde queremos tener nuestro kernel para el posterior compilado e instalación.

Si quisiéramos ir haciéndolo de a poco, porque no tengamos los dos empaquetados serian estos comandos:

xz -d para él .xz

tar xf para él .tar

```
[root@localhost src]# ls
debug  kernels  linux  linux-4.10  linux.4.10
```

1.7 Prepare el fichero de configuración tomando como referencia alguno de los ya instalados.

Aquí se nos presentan dos opciones de hacerlo.

Una sería usando el archivo config de nuestro kernel actual que sabemos que ya funciona, el cual deberíamos poder encontrar haciendo un filtrado ls.

```
ls /boot/config*
```

Y luego lo copiaríamos a la carpeta de nuestro nuevo kernel.

```
cp /boot/config-3.10.0-1160.el7.x86_64 /usr/src/linux-4.10/.config
```

La segunda opción, que es la que he utilizado yo, consiste en usar una de las opciones de los comandos del paquete de instrucciones de **make**.

```
make menuconfig
```

Con este comando lo que vamos a obtener es una interfaz gráfica para crear nuestro propio config para el kernel que queremos instalar, en mi caso deje todo de serie. Para ello simplemente ejecutamos el comando y en la interfaz gráfica **Save** y **Exit**.

1.8 Compile el núcleo y sus módulos.

Antes de comenzar con este paso, e incluso antes de usar el comando de make menuconfig deberemos instalar las herramientas entre las que se encuentran los make.

Para esto usaremos el siguiente comando:

```
yum group install "Development Tools"
```

Para compilar nuestro nuevo kernel deberemos estar en el directorio /usr/src y a la vez en el ejecutaremos el comando **make**

El siguiente paso compilar los módulos del kernel y para ello haremos uso del comando **make modules**

```
root@localhost Linux1# make modules
SYNC include/config/auto.conf.cmd
.config:485:warning: symbol value 'm' invalid for IBK
.config:697:warning: symbol value 'm' invalid for CPU_FREQ_STAT
.config:941:warning: symbol value 'm' invalid for NF_CT_PROTO_GRE
.config:969:warning: symbol value 'm' invalid for NF_NAT_REDIRECT
.config:972:warning: symbol value 'm' invalid for NF_TABLES_INET
.config:1139:warning: symbol value 'm' invalid for NF_TABLES_IPV4
.config:1143:warning: symbol value 'm' invalid for NF_TABLES_ARP
.config:1184:warning: symbol value 'm' invalid for NF_TABLES_IPV6
.config:1559:warning: symbol value 'm' invalid for NET_DEVLINK
.config:2719:warning: symbol value 'm' invalid for ISDN_CAPI
.config:3281:warning: symbol value 'm' invalid for PINCTRL_AMD
.config:3664:warning: symbol value 'm' invalid for LIRC
*
```

Veremos que nos da diferentes warning, pero mientras ninguno de ellos sea algo en concreto que necesitemos de alguna forma en específico, todo debería estar okay. Estos datos se solucionan en el config antes de la compilación del núcleo y de los módulos.

1.9 Instale el núcleo y sus módulos.

Después de hacer el make en el apartado anterior, hay que ejecutar el comando:

make bzImage

Esto nos permitirá generar el Kernel main file, en algunos kernel mas actuales este puede ser generado por el make, sin necesidad de usar este comando. En nuestro caso necesitamos hacer uso de el para poder generarlo.

Ahora cuando acudamos con un ls a nuestro directorio boot deberíamos tener en los diferentes archivos necesarios para seguir con nuestra instalación.

```
[root@localhost src]# ls /boot
config-3.10.0-1160.71.1.el7.x86_64  initramfs-0-rescue-38b19c6a829848e886a01c859b8e0100.img  rebu-vmlinuz-4.10.0  System.map-4.10.0  vmlinuz-4.10.0
efi                                  initramfs-3.10.0-1160.71.1.el7.x86_64.img                symvers-3.10.0-1160.71.1.el7.x86_64.gz  vmlinuz
grub                                initramfs-4.10.0.img                                       System.map                               vmlinuz-0-rescue-38b19c6a829848e886a01c859b8e0100
grub2                               rebu-initramfs-4.10.0.img                                   System.map-3.10.0-1160.71.1.el7.x86_64  vmlinuz-3.10.0-1160.71.1.el7.x86_64
[root@localhost src]#
```

Para instalar los módulos simplemente deberemos usar el comando:

make modules_install

Esto instalará los módulos de nuestro nuevo kernel, y deberá aparecer como última línea depmod y sin errores.

Es una práctica común y considera como buena, una vez instalados los módulos de nuestro nuevo kernel hacer uso del comando

depmod <versión del kernel> (depmod 4.10.0 en nuestro caso)

Esto asegurara que los módulos se actualicen y ejecuten adecuadamente en nuestro nuevo sistema.

Importante: Todos estos pasos deben realizarse desde el directorio del kernel que queremos instalar, es decir /usr/src/Linux-4.10

1.10 Haga una copia del núcleo y RAM Disk actuales. Los nuevos ficheros se llamarán como los antiguos añadiendo un prefijo que identifique al alumno.

Nos debemos dirigir al directorio /boot y hacer una copia de estos dos archivos:

```
cp vmlinuz-4.10.0 rebu-vmlinuz-4.10.0
```

```
cp initramfs-4.10.0.img rebu-initramfs-4.10.0.img
```

El ultimo archivo que hemos copiado sería el Ramdisk y el primero el generado por bzImage.

```
root@localhost src# ls /boot
config-3.10.0-1160.71.1.el7.x86_64  initramfs-0-rescue-38b19c6a829848e886a01c859b8e0100.img  rebu-vmlinuz-4.10.0  System.map-4.10.0  vmlinuz-4.10.0
initramfs-3.10.0-1160.71.1.el7.x86_64.img  initramfs-3.10.0-1160.71.1.el7.x86_64.img  sysext-3.10.0-1160.71.1.el7.x86_64.gz  vmlinuz
initramfs-4.10.0.img  rebu-initramfs-4.10.0.img  System.map  vmlinuz-0-rescue-38b19c6a829848e886a01c859b8e0100
rebu-initramfs-4.10.0.img  System.map-3.10.0-1160.71.1.el7.x86_64  vmlinuz-3.10.0-1160.71.1.el7.x86_64
root@localhost src#
```

1.11 Creación del RAM Disk

Usando la utilidad mkinitrd cree una RAM Disk para el núcleo que actualmente está en ejecución.

Para esto debemos acceder al directorio /boot y usar las funcionalidades de **mkinitrd**

```
mkinitrd -f -v rebu-initramfs-4.10.0.img rebu-vmlinuz-4.10.0
```

Compruebe los contenidos del RAM Disk creado, indique los módulos que contiene.

Para ello utilizo el comando:

```
lsinitrd /boot/rebu-initramfs-4.10.0.img
```

Los módulos son los siguientes (Los que empiezan por etc)


```

crw-r--r-- 1 root root 1, 11 Nov 2 04:34 dev/kmsg
crw-r--r-- 1 root root 1, 3 Nov 2 04:34 dev/null
lrwxrwxrwx 1 root root 7 Nov 2 04:34 bin -> usr/bin
drwxr-xr-x 2 root root 0 Nov 2 04:34 dev
drwxr-xr-x 12 root root 0 Nov 2 04:34 etc
drwxr-xr-x 2 root root 0 Nov 2 04:34 etc/cmdline.d
drwxr-xr-x 2 root root 0 Nov 2 04:34 etc/conf.d
-rw-r--r-- 1 root root 124 Nov 2 04:34 etc/conf.d/systemd.conf
-rw-r--r-- 1 root root 262 Sep 30 2020 etc/dhclient.conf
-rw-r--r-- 1 root root 0 Nov 2 04:34 etc/fstab.empty
-rw-r--r-- 1 root root 172 Nov 2 04:34 etc/group
-rw-r--r-- 1 root root 22 Oct 31 08:07 etc/hostname
-rw-r--r-- 1 root root 170 Nov 2 04:34 etc/initrd-release
-rw-r--r-- 1 root root 8043 Nov 2 04:34 etc/ld.so.cache
-rw-r--r-- 1 root root 28 Feb 27 2013 etc/ld.so.conf
drwxr-xr-x 2 root root 0 Nov 2 04:34 etc/ld.so.conf.d
-rw-r--r-- 1 root root 26 Feb 23 2022 etc/ld.so.conf.d/bind-export-x86_64.conf
-rw-r--r-- 1 root root 19 Aug 9 2019 etc/ld.so.conf.d/dyninst-x86_64.conf
-rw-r--r-- 1 root root 63 Jun 28 2022 etc/ld.so.conf.d/kernel-3.10.0-1160.71.1.el7.x86_64.conf
-rw-r--r-- 1 root root 17 Oct 1 2020 etc/ld.so.conf.d/mariadb-x86_64.conf
drwxr-xr-x 2 root root 0 Nov 2 04:34 etc/libnl
-rw-r--r-- 1 root root 1130 Aug 3 2017 etc/libnl/classid
-rw-r--r-- 1 root root 19 Oct 31 08:07 etc/locale.conf
drwxr-xr-x 2 root root 0 Nov 2 04:34 etc/lvm
-rw-r--r-- 1 root root 95859 Nov 2 04:34 etc/lvm/lvm.conf
-rw-r--r-- 1 root root 33 Oct 31 08:00 etc/machine-id
drwxr-xr-x 2 root root 0 Nov 2 04:34 etc/modprobe.d
-rw-r--r-- 1 root root 215 Jun 28 2022 etc/modprobe.d/dccp-blacklist.conf
-rw-r--r-- 1 root root 166 Apr 28 2021 etc/modprobe.d/firewalld-sysctls.conf
-rw-r--r-- 1 root root 674 Mar 21 2019 etc/modprobe.d/tuned.conf
lrwxrwxrwx 1 root root 17 Nov 2 04:34 etc/mtab -> /proc/self/mounts
lrwxrwxrwx 1 root root 14 Nov 2 04:34 etc/os-release -> initrd-release
-rw-r--r-- 1 root root 101 Nov 2 04:34 etc/passwd
drwxr-xr-x 2 root root 0 Nov 2 04:34 etc/plymouth
-rw-r--r-- 1 root root 72 Oct 1 2020 etc/plymouth/plymouthd.conf
-rw-r--r-- 1 root root 449 Nov 16 2020 etc/sysctl.conf
drwxr-xr-x 2 root root 0 Nov 2 04:34 etc/sysctl.d
lrwxrwxrwx 1 root root 14 Nov 2 04:34 etc/sysctl.d/99-sysctl.conf -> ../sysctl.conf
drwxr-xr-x 2 root root 0 Nov 2 04:34 etc/systemd
-rw-r--r-- 1 root root 1047 Nov 2 04:34 etc/systemd/journald.conf
-rw-r--r-- 1 root root 1552 Jan 13 2022 etc/systemd/system.conf
drwxr-xr-x 3 root root 0 Nov 2 04:34 etc/udev
drwxr-xr-x 2 root root 0 Nov 2 04:34 etc/udev/rules.d
-rw-r--r-- 1 root root 142 Sep 12 2013 etc/udev/rules.d/11-dm.rules
-rw-r--r-- 1 root root 681 Nov 2 04:34 etc/udev/rules.d/59-persistent-storage-dm.rules
-rw-r--r-- 1 root root 298 Nov 2 04:34 etc/udev/rules.d/59-persistent-storage.rules
-rw-r--r-- 1 root root 1031 Nov 2 04:34 etc/udev/rules.d/61-persistent-storage.rules
-rw-r--r-- 1 root root 776 Sep 12 2013 etc/udev/rules.d/64-lvm.rules
-rw-r--r-- 1 root root 49 Jan 13 2022 etc/udev/udev.conf
-rw-r--r-- 1 root root 37 Oct 31 08:07 etc/vconsole.conf
-rw-r--r-- 1 root root 1982 Dec 15 2020 etc/virc
lrwxrwxrwx 1 root root 23 Nov 2 04:34 init -> usr/lib/systemd/systemd

```

Compruebe que puede arrancar con el nuevo núcleo creado

En mi caso no tuve que modificar el archivo del boot loader porque una vez que ya reinicié el sistema me arrancaba sin problemas en el nuevo kernel.

Por lo tanto, seleccione el nuevo kernel al reiniciar:

```

CentOS Linux (4.10.0) ? (Core)
CentOS Linux (3.10.0-1160.71.1.el7.x86_64) ? (Core)
CentOS Linux (0-rescue-38b19c6a829848e886a01c859b8e0100) ? (Core)

```

El kernel justo arriba del seleccionado en la imagen y una vez que inicia nos confirma que ha cargado correctamente el 4.10.0

```

CentOS Linux 7 (Core)
Kernel 4.10.0 on an x86_64

localhost login:

```

1.12 Gestión de módulos.

Localice el módulo que nos permite tener acceso a la unidad de cdrom.

Para encontrar el módulo que gestiona el cdrom nos vamos a `/lib/modules` y usamos el comando

lsmod

```
libcrc32c      16384  2 xfs,nf_nat
sd_mod        49152  3
sr_mod        24576  0
virtio_scsi    20480  2
cdrom         61440  1 sr_mod
virtio_net     36864  0
ata_generic    16384  0
```

En la foto encontramos diferentes módulos, pero a nosotros nos interesa el `cdrom` y el `sr_mod` que es el que usa al `cdrom`.

Indique los parámetros que admite dicho módulo

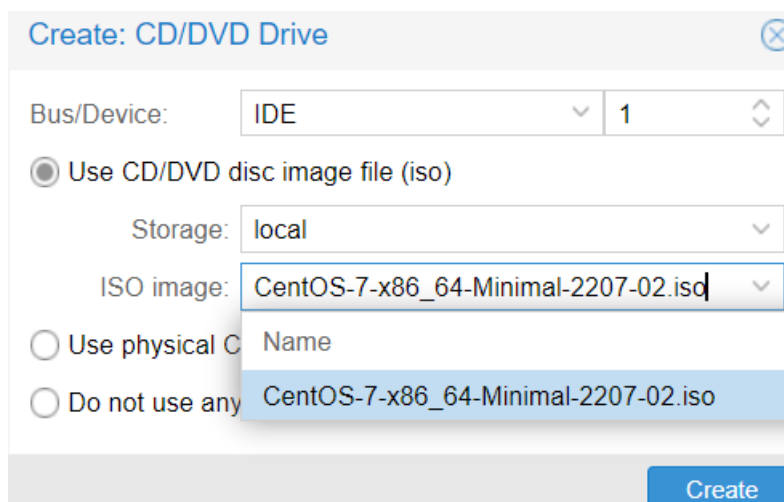
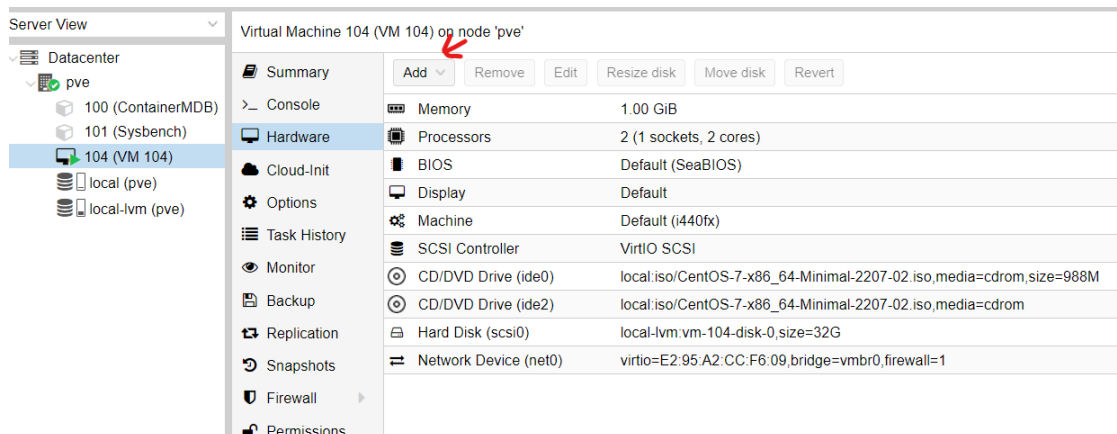
Para esto basta con poner el siguiente comando que nos dará unos `parm`:

modinfo cdrom

```
[root@localhost ~]# modinfo cdrom
filename:       /lib/modules/4.10.0/kernel/drivers/cdrom/cdrom.ko
license:       GPL
srcversion:     D29A5304E21F7ACCD44CC4E
depends:
intree:        Y
vermagic:      4.10.0 SMP mod_unload modversions
parm:          debug:bool
parm:          autoclose:bool
parm:          autoeject:bool
parm:          lockdoor:bool
parm:          check_media_type:bool
parm:          mrw_format_restart:bool
```

Vincule un fichero ISO como unidad de cdrom de nuestra máquina.

Para este apartado tenemos que irnos al servidor en proxmox. Seleccionamos nuestra máquina virtual, nos vamos a la parte de hardware y le damos a Add como se ve en la imagen.



Pruebe que funciona dicho módulo ejecutando el comando `mount /dev/cdrom /mnt` y comprobando que el contenido cdrom lo podemos consultar accediendo al directorio.

Como dice el enunciado primero ponemos el comando y luego accedemos con ls. Como podemos ver en la foto, nos ha dejado acceder al directorio por lo que lo hemos montado bien

```
[root@localhost ~]# mount /dev/cdrom /mnt
mount: /dev/sr0 is write-protected, mounting read-only
[root@localhost ~]# ls /mnt
CentOS_BuildTag  EULA  images  LiveOS  repodata  RPM-GPG-KEY-CentOS-Testing-7
EFI              GPL  isolinux Packages RPM-GPG-KEY-CentOS-7  TRANS.TBL
[root@localhost ~]#
```

Descargue el módulo que da acceso a la unidad de cdrom.

Para este paso primero deberemos desmontar el cdrom para que deje de estar en uso los módulos.

umount /mnt

Para posteriormente desinstalar los módulos instalados y volver a instalarlos después.

1º **rmmod sr_mod** (Porque es el modulo que usa cdrom)

2º **rmmod cdrom**

Tras hacer usar estos dos comandos tendremos desinstalados los módulos que nos daban acceso a los montajes del cdrom.

Aquí podemos ver cómo están instalados **antes** de usar los comandos:

```
libcrc32c      16384  2 xfs,nf_nat
sd_mod         49152  3
sr_mod         24576  0
virtio_scsi    20480  2
cdrom          61440  1 sr_mod
virtio_net     36864  0
ata_generic    16384  0
```

Y aquí podremos ver como no están una vez usamos los comandos:

```
libcrc32c      16384  2 xfs,nf_nat
sd_mod         49152  3
virtio_scsi    20480  2
virtio_net     36864  0
ata_generic    16384  0
```

Para obtener ambas imágenes usamos **lsmod** como hicimos al principio de este apartado.

Ahora cuando intentamos montar otra vez el cdrom nos dira que no se ha podido, por lo que deberemos volver a instalar dichos módulos

Para esto seguiremos estos dos sencillos pasos:

1º **modprobe cdrom**

2º **modprobe sr_om**

Ahora si nos dejara montar nuestro cdrom y acceder a él. Una vez hechos estos pasos nos aparecerán los módulos listados al usar un **lsmod** en las primeras líneas de los módulos en vez de en su posición anterior.

1.13 Compilación de módulos de terceros (optativa)

Compile este módulo haciendo uso de la utilidad DKMS

Para empezar, deberemos descargar o clonar el código fuente desde el repositorio de github

git clone https://github.com/umlaeute/v4l2loopback.git

Después accedemos al repositorio local de nuestra VM

cd v4l2loopback

Para proseguir con nuestra compilación necesitaremos instalar la utilidad DKMS

1º yum install --enablerepo=extras epel-release

2º yum remove ipa-common ipa-common-client ipa-client

3º yum install kernel-debug-devel dkms

Una vez tengamos la utilidad instalada deberemos hacer uso del **make** nuevamente para la instalación

make dkms

Esto nos permitirá que se instale el módulo en el directorio de DKMS para que se compile e instale automáticamente cada vez que se actualice el kernel.

Después procederemos a cargar el módulo:

modprobe v4l2loopback

Después ejecutaremos un **ls** para saber si se ha instalado y deberemos ver una dependencia de video

```
root@localhost v4l2loopback]# ls /dev/video*  
/dev/video0
```

Seguido deberíamos hacer uso de los ejemplos de prueba que vienen en el repositorio que clonamos, pero al probarlos se queda colgada la VM y no he conseguido avanzar más.

1.14 Conclusión

Como conclusión lo que he podido sacar en claro desde primera hora, es que debes saber manejarte con Linux para que este tipo de practica se te haga medianamente llevadera por que debes descargar bastantes utilidades y repositorios.

Una vez que te adaptas a esas pequeñas cosas ya la cosa empieza a fluir un poco mejor, pero es entonces cuando empiezas a tener los odiosos problemas de compilado. En mi caso cuando quise instalar un kernel actual tuve que crear 4 maquinas distintas haciendo pruebas, hasta que di con el make menuconfig que fue el que me pudo solucionar los problemas. Una vez que di con esa utilidad todo fue mucho más fluido y empecé a comprender más de lo que hacía.

En general me parece una practica que hay que dedicarle bastantes horas si no sabes lo que estas haciendo, pero que una vez que consigues cogerle el hilo resulta interesante y se acortan bastante los tiempos en el proceso.



Universidad
de Huelva



Escuela Técnica
Superior de Ingeniería
Universidad de Huelva

Memoria de Prácticas

ADMINISTRACIÓN DE
SERVIDORES

Grado en Ingeniería
Informática

Alejandro Gordillo Pedraza
04/12/2023

Contenido

ADMINISTRACIÓN DE SERVIDORES	1
Grado en Ingeniería Informática.....	1
Resumen.....	3
Cápítulo 1	4
1. Nivel de arranque del sistema	4
2. Servicios Web	6
3. Recuperación del sistema	7
4. Protección Grub	9
5. CONCLUSION:	10

Resumen

En esta practica aprenderemos más acerca del arranque de los sistemas. Para ello haremos uso de dos SO diferentes, centos 6 y rocky Linux 8.

Entre las funciones que probaremos tenemos mirar los servicios activos de diferentes niveles, instalar un servidor web para que arranque con nuestro sistema y recuperar la contraseña root en el caso de perderla.

C  pulo 1

1. Nivel de arranque del sistema

Arranque la maquina basada en centos 6 he indique el nivel de arranque que se ejecuta por defecto.

Una vez instaladas e iniciadas las maquinas virtuales con nuestros sistemas usaremos un comando diferente en cada maquina para ver su nivel de arranque:

Centos: **grep "^id" /etc/inittab**

```
root@localhost ~]# grep "^id" /etc/inittab
id:3:initdefault:
```

Rocky: **systemctl get-default**

```
[root@localhost ~]# systemctl get-default
multi-user.target
```

Una vez usados los comandos en su correspondiente sistema, nos devuelven la informaci  n que vemos en las im  genes anteriores.

En el caso de centos nos dice que usamos un nivel de arranque 3 y en el caso de Rocky nos dice que usar un arranque multi-user.target, lo cual equivale a un nivel 3 de centos.

¿Podría indicar los servicios que se ejecutan en el nivel por defecto de cada una de las dos máquinas?

Centos:

```
[root@localhost ~]# chkconfig --list
```

auditd	0:desactivado	1:desactivado	2:activo	3:activo	4:activo	5:activo
activado						
blk-availability	0:desactivado	1:activo	2:activo	3:activo	4:activo	5:activo
:desactivado						
crond	0:desactivado	1:desactivado	2:activo	3:activo	4:activo	5:activo
activado						
ip6tables	0:desactivado	1:desactivado	2:activo	3:activo	4:activo	5:activo
activado						
iptables	0:desactivado	1:desactivado	2:activo	3:activo	4:activo	5:activo
activado						
iscsi	0:desactivado	1:desactivado	2:desactivado	3:activo	4:activo	5:activo
activado						
iscsid	0:desactivado	1:desactivado	2:desactivado	3:activo	4:activo	5:activo
activado						
lvm2-monitor	0:desactivado	1:activo	2:activo	3:activo	4:activo	5:activo
activado						
mdmonitor	0:desactivado	1:desactivado	2:activo	3:activo	4:activo	5:activo
activado						
multipathd	0:desactivado	1:desactivado	2:desactivado	3:desactivado	4:desactivado	5:desactivado
activado						
netconsole	0:desactivado	1:desactivado	2:desactivado	3:desactivado	4:desactivado	5:desactivado
activado						
netfs	0:desactivado	1:desactivado	2:desactivado	3:activo	4:activo	5:activo
activado						
network	0:desactivado	1:desactivado	2:activo	3:activo	4:activo	5:activo
activado						
postfix	0:desactivado	1:desactivado	2:activo	3:activo	4:activo	5:activo
activado						
rdisc	0:desactivado	1:desactivado	2:desactivado	3:desactivado	4:desactivado	5:desactivado
activado						
restorecond	0:desactivado	1:desactivado	2:desactivado	3:desactivado	4:desactivado	5:desactivado
activado						
rsyslog	0:desactivado	1:desactivado	2:activo	3:activo	4:activo	5:activo
activado						
ssslauthd	0:desactivado	1:desactivado	2:desactivado	3:desactivado	4:desactivado	5:desactivado
activado						
sshd	0:desactivado	1:desactivado	2:activo	3:activo	4:activo	5:activo
activado						
udev-post	0:desactivado	1:activo	2:activo	3:activo	4:activo	5:activo
activado						

Rocky:

```
[root@localhost ~]# systemctl list-dependencies multi-user.target
```

multi-user.target	●	dm-event.socket
● auditd.service	●	sssd-kcm.socket
● chronyd.service	●	systemd-coredump.socket
● crond.service	●	systemd-initctl.socket
● firewallld.service	●	systemd-journald-dev-log.socket
● irqbalance.service	●	systemd-journald.socket
● kdump.service	●	systemd-udev-control.socket
● NetworkManager.service	●	systemd-udev-kernel.socket
● rsyslog.service	●	sysinit.target
● sshd.service	●	dev-hugepages.mount
0 ● sssd.service	●	dev-mqueue.mount
● systemd-ask-password-wall.path	●	dracut-shutdown.service
● systemd-logind.service	●	kmod-static-nodes.service
0 ● systemd-update-utmp-runlevel.service	●	ldconfig.service
● systemd-user-sessions.service	●	lvm2-lvmpolld.socket
● basic.target	0	lvm2-monitor.service
● .mount	●	nis-domainname.service
0 ● microcode.service	●	proc-sys-fs-binfmt_misc.automount
● paths.target	●	selinux-autorelabel-mark.service
● slices.target	●	sys-fs-fuse-connections.mount
● .slice	●	sys-kernel-config.mount
● system.slice	●	sys-kernel-debug.mount
● sockets.target	0	sys-kernel-tracing.mount
● dbus.socket	0	systemd-ask-password-console.path
		systemd-binfmt.service
		systemd-boot-system-token.service

Cambie el nivel por defecto en ambos sistemas y compruebe las diferencias que hay en los servicios lanzados

Para cambiar el nivel por defecto en CentOS escribiremos:

vi /etc/inittab

Y cambiamos el fichero de configuración al nivel 1.

Para RockyLinux:

systemctl set-default graphical.

Con esto cambiaría a un nivel de arranque 5

2. Servicios Web

En el sistema basado en Rocky instale el servidor web wginx

Para instalar este servidor web usaremos los siguientes comandos:

dnf install nginx
systemctl start nginx
systemctl enable nginx

El últimos nos permitirá hacer que el servidor se inicie con el arranque de nuestro sistema.

```
[root@localhost ~]# sudo systemctl start nginx
[root@localhost ~]# sudo systemctl enable nginx
Created symlink /etc/systemd/system/multi-user.target.wants/nginx.service → /usr/lib/systemd/system/nginx.service.
[root@localhost ~]# sudo systemctl status nginx
● nginx.service - The nginx HTTP and reverse proxy server
   Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: disabled)
   Active: active (running) since Wed 2023-11-15 09:37:43 CET; 22s ago
     Main PID: 15907 (nginx)
       Tasks: 3 (limit: 11112)
      Memory: 2.8M
         CPU: 47ms
    CGroup: /system.slice/nginx.service
            └─15907 "nginx: master process /usr/sbin/nginx"
              └─15908 "nginx: worker process"
                └─15909 "nginx: worker process"

Nov 15 09:37:42 localhost.localdomain systemd[1]: Starting The nginx HTTP and reverse proxy server...
Nov 15 09:37:43 localhost.localdomain nginx[15905]: nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
Nov 15 09:37:43 localhost.localdomain nginx[15905]: nginx: configuration file /etc/nginx/nginx.conf test is successful
Nov 15 09:37:43 localhost.localdomain systemd[1]: Started The nginx HTTP and reverse proxy server.
lines 1-16/16 (END)
```

En el sistema basado en centos, instale el servidor web apache

Para instalar el servidor web apache en nuestro sistema basado en centos usaremos los siguientes comandos:

```
yum install httpd
service httpd start
chkconfig httpd on
```

El ultimo comando es como en el apartado anterior, el que nos facilita la opción para arrancar apache con nuestro sistema

```
root@localhost ~]# service httpd status
Se está ejecutando httpd (pid 1380)...
```

3. Recuperación del sistema

Asumiendo que desconocemos la clave de root, indique, detalladamente, los pasos que realizaría para restaurar la clave del root en ambos sistemas.

Rocky:

Como recuperar nuestra contraseña o cambiar la contraseña del usuario Root en rocky Linux:

1º Deberemos reiniciar nuestro sistema y en el menú Boot del sistema pulsar la tecla **E** sobre el kernel que queremos modificar. Esto nos permitirá acceder al menú de edición grub del kernel.

Una vez accedamos a este menú deberemos escribir **systemd.debug-shell** al final de la línea que comienza por Linux

```
load_video
set gfx_payload=keep
insmod gzio
linux ($root)/vmlinuz-4.18.0-513.5.1.el8_9.x86_64 root=/dev/mapper/rl-root ro \
crashkernel=auto resume=/dev/mapper/rl-swap rd.lvm.lv=rl/root rd.lvm.lv=rl/swap
p systemd.debug-shell
initrd ($root)/initramfs-4.18.0-513.5.1.el8_9.x86_64.img $tuned_initrd
```

Esto nos permitirá iniciar en modo de rescate, esta función del sistema nos arranca el sistema sin necesidad de iniciar con ningún usuario.

Una vez escrito usaremos **Ctrl + X** para guardar los cambios aplicados y nos arrancara el sistema de forma normal. Esto quiere decir que aun no nos encontramos en modo

rescate, para acceder a este modo cuando nos pida el usuario y contraseña haremos uso de la combinación **ALT + F9**. La cual si nos introdujera al modo rescate.

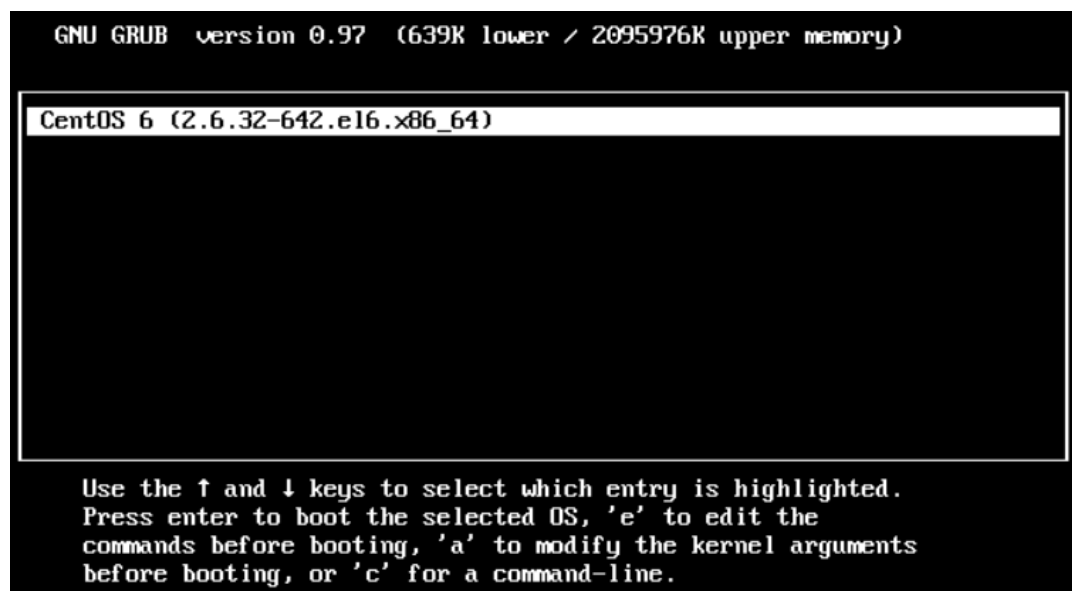
Una vez no encontremos en el modo arranque deberemos escribir **passwd** y nos permitirá cambiar la contraseña del usuario que queramos, incluido el usuario root.

Después de hacer el cambio de contraseña y que nos devuelva un Ok de respuesta podremos reiniciar el sistema e iniciar con nuestra nueva contraseña.

Centos:

En el caso de Centos haremos algunos pasos parecidos pero lo desarrollaremos para no inducir a fallos.

Cuando arranquemos nuestro sistema en Centos deberemos pulsar cualquier tecla menos ESC y eso nos abrirá un menú GRUB donde aparecerán los kernel disponibles.



Una vez seleccionado el kernel que queremos modificar pulsamos la letra **E** y nos abrirá un menú como el de rocky donde volveremos a pulsar la tecla **E** sobre el kernel, acto seguido pondremos la siguiente línea al final de la línea que aparece:

Init=/bin/bash

Después pulsaremos enter y se grabará, seguido volveremos al menú anterior y veremos nuevamente el kernel y podremos iniciarlo haciendo uso de la letra **B**.

Una vez ya dentro del Bash podremos cambiar la contraseña como hicimos en rocky Linux.

Importante: Si por algún caso nos dice que no se puede actualizar el token de la contraseña deberemos remontar el sistema en escritura y lectura, eso quiere decir que solo esta disponible para lectura. Para ello usaremos este comando y todo correcto:

Mount -o remount,rw /

4. Protección Grub

En el caso del sistema basado en Rocky, modifique la configuración del grub para protegerla con contraseña

1º Deberemos saber que tipo de grub usa nuestro sistema, para eso usaremos el comando:

rpm -qa | grep grub

```
[root@localhost ~]# rpm -qa | grep grub
grub2-tools-extra-2.02-150.el8.rocky.0.1.x86_64
grub2-pc-modules-2.02-150.el8.rocky.0.1.noarch
grub2-tools-2.02-150.el8.rocky.0.1.x86_64
grub2-pc-2.02-150.el8.rocky.0.1.x86_64
grub2-common-2.02-150.el8.rocky.0.1.noarch
grub2-tools-minimal-2.02-150.el8.rocky.0.1.x86_64
grubby-8.40-48.el8.x86_64
```

Como podemos ver nuestro sistema usa grub2, una vez tenemos esa información. Escribiremos el siguiente comando para protegerlo con una contraseña:

grub2-setpassword

Elegimos la nueva contraseña que queremos usar y actualizamos los datos del grub con:

grub2-mkconfig

5. CONCLUSION:

En esta practica hemos podido aprender un poco más acerca del arranque de los sistemas basados en Linux, entre las cosas que mas siento que he aprendido son como proteger mi sistema de arranque y como recuperar la contraseña de serlo necesario. Y como actualizar los repositorios de centos, debido a que durante la practica he usado un centos que no es muy reciente y esto hacia bastante difícil poder descargar algunas de las herramientas que necesitaba para su correcto funcionamiento. Si controlas los repositorios de centos la practica es bastante practica y rápida, pero hasta que entiendes como arreglar eso se pone bastante cuesta arriba, pero por lo general me ha gustado y he aprendido que es lo importante.

6. Soluciones Problemas

Si necesitas actualizar los repositorios de centos con este comando te actualiza todos los repositorios y los mirrors que es lo mas tedioso dentro de estos:

```
sudo sed -i  
'/^mirrorlist/s/^/#;/^#baseurl/{s/#//;s/mirror.centos.org/centos/$releasever/vault.centos.org/6.8/}' /etc/yum.repos.d/*B*
```




**Universidad
de Huelva**



Escuela Técnica Superior de
Ingeniería Universidad de Huelva

Memoria de Prácticas

ADMINISTRACIÓN DE SERVIDORES

Grado en Ingeniería Informática

Alejandro Gordillo Pedraza
12/12/2023

Contenido

ADMINISTRACIÓN DE SERVIDORES.....1

Grado en Ingeniería Informática1

Resumen.....3

Cápítulo 14

1. Ejercicios4

2. Conclusión..... 11

Resumen

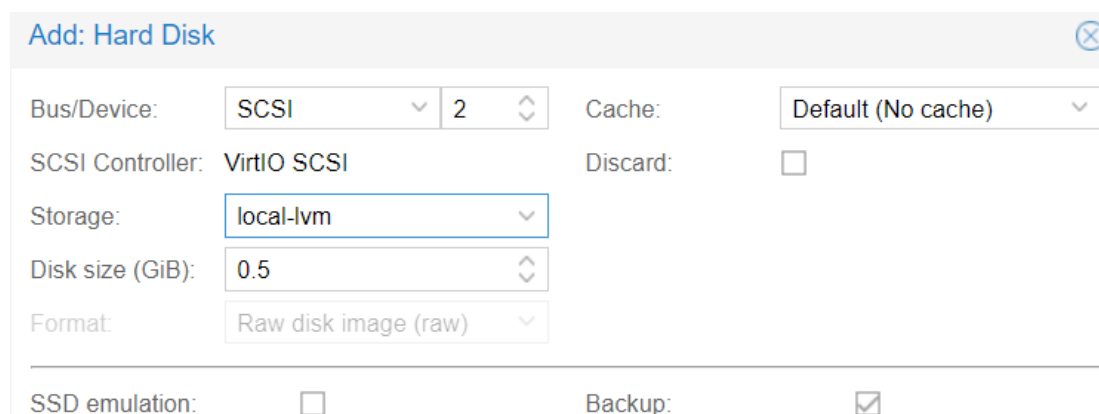
En esta práctica aprenderemos más acerca de Sistemas de Ficheros y dispositivos, para ellos necesitaremos añadir un disco duro nuevo a nuestra maquina virtual, crearle unas particiones y unas serie de modificaciones que le realizaremos a lo largo de la practica.

C  pulo 1

1. Ejercicios

Asocie a cualquiera de las m  quinas virtuales creadas un nuevo disco duro de 500 Mb.

Primero tenemos que elegir la m  quina virtual, yo en este caso voy a elegir la que tiene CentOS 7 ya que est   m  s actualizado. Una vez seleccionada, nos vamos a hardware y le damos a   adir un nuevo disco duro. Como tiene que ser de 500Mb y solo deja poner en gigas, ponemos 0.5.



Add: Hard Disk

Bus/Device: SCSI 2 Cache: Default (No cache)

SCSI Controller: VirtIO SCSI Discard: ☐

Storage: local-lvm

Disk size (GiB): 0.5

Format: Raw disk image (raw)

SSD emulation: ☐ Backup: ☒

Cree dos particiones de 256 Mb cada una.

Para saber cu  l es el nuevo disco duro, nos vamos a `/sys/block` y ah   aparecen `sda` y `sdb`, si hacemos:

```
[root@localhost ~]# cat /sys/block/sdb/device/model
DEMU HARDDISK
```

`cat /sys/block/sdb/device/model`

Podemos ver qu   tipo es, en este caso es nuestro disco duro ya que es un hard disk y porque al ser el segundo que hemos   adido su letra es una b. Una vez hecho esto, creamos las dos particiones con `fdisk` poniendo:

`fdisk /dev/sdb`

Y ponemos n para crear las particiones

```

troot@localhost /l# fdisk /dev/sdb
Welcome to fdisk (util-linux 2.23.2).

Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Device does not contain a recognized partition table
Building a new DOS disklabel with disk identifier 0x7b4fb2fb.

Orden (m para obtener ayuda): n
Partition type:
   p   primary (0 primary, 0 extended, 4 free)
   e   extended
Select (default p): primary
Número de partición (1-4, default 1): 1
Primer sector (2048-1048575, valor predeterminado 2048): 2048
Last sector, +sectors or +size(K,M,G) (2048-1048575, valor predeterminado 1048575): +256M
Partition 1 of type Linux and of size 256 MiB is set

Orden (m para obtener ayuda): n
Partition type:
   p   primary (1 primary, 0 extended, 3 free)
   e   extended
Select (default p): primary
Número de partición (2-4, default 2): 2
Primer sector (526336-1048575, valor predeterminado 526336): 526336
Last sector, +sectors or +size(K,M,G) (526336-1048575, valor predeterminado 1048575):
1048575
Partition 2 of type Linux and of size 255 MiB is set

```

Como podemos ver, ambas particiones se han creado con la capacidad que nos pedían. Para guardar los cambios, ponemos w.

```

Orden (m para obtener ayuda): w
¡Se ha modificado la tabla de particiones!

Llamando a ioctl() para volver a leer la tabla de particiones.
Se están sincronizando los discos.

```

En la primera partición, cree un sistema de fichero tipo ext4. En la segunda partición, cree un sistema de ficheros tipo btrfs.

Con el comando mkfs podemos crear un sistema de ficheros del tipo que le digamos, de este modo:

```
mkfs -t ext4 /dev/sdb1
```

Siendo sdb1 la primera partición y sdb2 la segunda.

Monte los sistemas de ficheros creados en /mnt/ext4 y /mnt/btrfs respectivamente.

Nos vamos a /mnt y creamos los directorios donde los vamos a montar, una vez que los tengamos creados usamos el comando:

```
mount /dev/sdb1 /mnt/ext4
```

```
[root@localhost ~]# mount /dev/sdb1 /mnt/ext4
[root@localhost ~]# mount /dev/sdb2 /mnt/btrfs
```

Usando el comando du y df, calcule el espacio libre y ocupado en los sistemas de ficheros creados. ¿Observa alguna diferencia entre los valores devueltos por df y du?

Dentro del directorio mnt ponemos df y du.

```
[root@localhost mnt]# df
S.ficheros          bloques de 1K  Usados Disponibles Uso% Montado en
devtmpfs             237280         0      237280    0% /dev
tmpfs                249216         0      249216    0% /dev/shm
tmpfs                249216      4576      244640    2% /run
tmpfs                249216         0      249216    0% /sys/fs/cgroup
/dev/mapper/centos-root 31433732 10764212    20669520   35% /
/dev/sda1            1038336      211064      827272   21% /boot
tmpfs                49844         0       49844    0% /run/user/0
/dev/sdb1            245671       2062       226406    1% /mnt/ext4
/dev/sdb2            261120      16704      178176    9% /mnt/btrfs

[root@localhost mnt]# du
12      ./ext4/lost+found
13      ./ext4
0       ./btrfs/snapshot
0       ./btrfs/home
16      ./btrfs
29      .
```

Como podemos ver, la primera partición usa 2062 bloques de un Kb, por lo que está ocupando un poco más de dos Mb y tiene disponibles más de 226 megas.

En cambio, la segunda partición está usando 17,7 Mb y tiene libre 178,2 Mb.

Sí hay diferencias entre df y du: En la primera partición pone en df que hay 2062 bloques usados, pero en du pone que hay en disco doce megas y esto se debe a que df solo cuenta los archivos abiertos en ese momento y du suma los archivos relacionados en cada carpeta.

Modifique las etiquetas de los sistemas de ficheros montados, las nuevas etiquetas serán, respectivamente, EXT4 Y BTRFS.

Usando la función e2label podemos modificar etiquetas, ponemos:

```
e2label /dev/sdb1/ EXT4
```

```
[root@localhost mnt]# e2label /dev/sdb1 EXT4
```

Aunque para btrfs esta función no sirve, por lo que usaremos:

```
btrfs filesystem label /mnt BTRFS
```

```
[root@localhost mnt]# btrfs filesystem label /mnt BTRFS
```

En el sistema de ficheros de tipo ext4 modifique el porcentaje reservado para el root y compruebe si se ve reflejado en los valores devueltos por df y du.

Para modificar el porcentaje utilizamos el comando

```
tune2fs -m 2 /dev/sdb1
```

```
[root@localhost /]# tune2fs -m 2 /dev/sdb1
tune2fs 1.42.9 (28-Dec-2013)
Setting reserved blocks percentage to 2% (5242 blocks)
```

```
[root@localhost /]# df
```

S.ficheros	bloques de 1K	Usados	Disponibles	Uso%	Montado en
devtmpfs	237280	0	237280	0%	/dev
tmpfs	249216	0	249216	0%	/dev/shm
tmpfs	249216	4592	244624	2%	/run
tmpfs	249216	0	249216	0%	/sys/fs/cgroup
/dev/mapper/centos-root	31433732	10766228	20667504	35%	/
/dev/sdb2	261120	16704	178176	9%	/mnt/btrfs
/dev/sdb1	245671	2062	234271	1%	/mnt/ext4
/dev/sda1	1038336	211064	827272	21%	/boot
tmpfs	49844	0	49844	0%	/run/user/0

En la partición btrfs cree dos subvolúmenes. El primero se denominará home y el segundo snapshot.

Primero, dentro de /mnt/btrfs tenemos que crear dos directorios, esto lo hacemos con mkdir, poniendo:

```
mkdir /btrfs/home
```

```
mkdir /btrfs/snapshot
```

Una vez hecho esto utilizamos la función subvolume create y ponemos dónde queremos crearlo.

```
btrfs subvolume create /mnt/btrfs/home
```

```
[root@localhost /]# cd /mnt
[root@localhost mnt]# btrfs subvolume create /mnt/btrfs/snapshot
Create subvolume '/mnt/btrfs/snapshot'
```

```
[root@localhost mnt]# btrfs subvolume create /mnt/btrfs/home
Create subvolume '/mnt/btrfs/home'
```

Modifique el fichero `/etc/fstab` para que los sistemas de ficheros creados se monten al arrancar. El sistema de ficheros ext4 se montará en `/mnt/ext4` mientras que el subvolumen home del sistema de ficheros btrfs se debe montar en home.

Para modificar el fichero ponemos:

```
vi /etc/fstab
```

Como uno es un subvolumen se hace de forma distinta. Para ambas, primero ponemos la partición que queremos montar, a continuación, ponemos dónde lo queremos montar y ponemos qué tipo de sistema de ficheros es. Luego, para ext4 ponemos que se monte automáticamente. Por último, para el subvolumen ponemos defaults.

```
#
# /etc/fstab
# Created by anaconda on Fri Nov 11 10:27:18 2022
#
# Accessible filesystems, by reference, are maintained under '/dev/disk'
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info
#
/dev/mapper/centos-root / xfs defaults 0 0
UUID=517f3c3c-127a-40b8-8e7e-c2160c4815f6 /boot xfs defaults 0 0
/dev/mapper/centos-swap swap swap defaults 0 0
/dev/sdb1 /mnt/ext4 ext4 auto,x-systemd.automount 0 0
/dev/sdb2 /mnt/btrfs/home btrfs defaults,subvol=home 0 0
```

Luego hacemos reboot.

Pruebe a hacer una instantánea del subvolumen home. Modifique los datos en home y restaure la instantánea antes creada.

Para crear un subvolumen se hace con la utilidad que hemos usado antes, la de subvolume, tenemos que poner:

```
btrfs subvolume snapshot /mnt/btrfs/home /mnt/btrfs/home_snapshot
```

```
[root@localhost mnt]# btrfs subvolume snapshot /mnt/btrfs/home /mnt/btrfs/home_snapshot
Create a snapshot of '/mnt/btrfs/home' in '/mnt/btrfs/home_snapshot'
```


Ahora creamos ficheros en home y restauramos la instantánea.

```
[root@localhost home]# cat > texto1.txt
hola
[root@localhost home]# cat > texto2.txt
Hoy es viernes 9 de diciembre
[root@localhost home]# cat > texto3.txt
Práctica 4 de adm de servidores
[root@localhost home]# ls
texto1.txt  texto2.txt  texto3.txt
```

```
[root@localhost /]# rsync -avz --delete /mnt/btrfs/home_snapshot /mnt/btrfs/home
sending incremental file list
home_snapshot/

sent 67 bytes  received 20 bytes  174.00 bytes/sec
total size is 0  speedup is 0.00
[root@localhost /]# cd mnt
[root@localhost mnt]# cd btrfs
[root@localhost btrfs]# cd home
[root@localhost home]# ls
home_snapshot  texto1.txt  texto2.txt  texto3.txt
```

dm-crypt es una utilidad que permite cifrar de forma transparente dispositivo de bloques. Haciendo uso de esta herramienta, cree una nueva partición que se montará dentro del directorio /privado y cuyos contenidos deben ir cifrados.

Primero creamos la partición con fdisk y el disco donde la vayamos a crear.

Luego instalamos cryptsetup y cryptmount y cargamos los módulos aed, md-crypt y sha256.

```
yum install cryptsetup
```

```
yum install cryptmount
```

```
modprobe aed
```

```
modprobe sha256
```

```
modprobe md-crypt
```

Una vez hecho esto, inicializamos la partición usando cryptsetup y le ponemos una contraseña, esta contraseña debe ser obligatoriamente robusta.

```
cryptsetup -s 512 luksFormat /dev/sdb2
```

```
[root@localhost ~]# cryptsetup -s 512 luksFormat /dev/sdb2

WARNING!
=====
Esto sobrescribirá los datos en /dev/sdb2 de forma irrevocable.

Are you sure? (Type uppercase yes): YES
Introduzca la frase contraseña de /dev/sdb2:
Verifique la frase contraseña:
```

A continuación, abrimos la partición y le damos un nombre como identificador.

```
cryptsetup luksOpen /dev/sdb2 micifrado
```

Hecho esto, se habrá creado el dispositivo /dev/mapper/micifrado y ahora tendríamos que crear el sistema de ficheros de este dispositivo.

```
mkfs.btrfs /dev/mapper/micifrado
```

```
[root@localhost ~]# mkfs.btrfs /dev/mapper/micifrado
btrfs-progs v4.9.1
See http://btrfs.wiki.kernel.org for more information.

Label:                (null)
UUID:                 e482f555-ae4b-41f3-9c6c-0f49bd8824de
Node size:            16384
Sector size:          4096
Filesystem size:      253.00MiB
Block group profiles:
  Data:               single          8.00MiB
  Metadata:           DUP             32.00MiB
  System:             DUP             8.00MiB
SSD detected:         no
Incompat features:    extref, skinny-metadata
Number of devices:    1
Devices:
  ID     SIZE  PATH
  1     253.00MiB /dev/mapper/micifrado
```

Por último, en el directorio raíz crearíamos el directorio /privado y montaríamos ahí el dispositivo.

```
mkdir /privado
```

```
mount /dev/mapper/micifrado /privado
```

2. Conclusión

En esencia lo que he podido sacar de esta práctica ha sido que me ha parecido un tema muy interesante, porque siempre viene bien saber un poco mas acerca de las modificaciones o particiones que le podemos hacer a nuestras unidades de almacenamiento. Esto es algo de lo que ya sabia un poco pero no a nivel de comandos, lo cual me ha gustado bastante aprender.



Universidad
de Huelva



Escuela Técnica Superior de
Ingeniería Universidad de Huelva

Memoria de Prácticas

ADMINISTRACIÓN DE SERVIDORES

Grado en Ingeniería Informática

Alejandro Gordillo Pedraza
16/12/2023

Contenido

Tabla de contenido

Contenido	2
Resumen	3
C�pitulo 1	4
5. Ejercicios	4
1. Cree una m�quina virtual (o use alguna de las ya creadas) y asocie un nuevo disco virtual de 512Mb ...	4
2. En dicho disco duro virtual, cree tres particiones del mismo tama�o.	4
3. Configure las tres particiones en RAID5.....	4
4. Monte las raid en /mnt/RAID.....	5
5. Cree distintos ficheros.	5
6. Simule la rotura de uno de los elementos del RAID.	6
7. Compruebe que puede seguir accediendo a los datos aunque uno de los discos est� marcado como estropeado.	7
8. Cree m�s ficheros en el RAID.	7
9. A�ada un nuevo disco, cree una partici�n de igual tama�o a las ya creadas y config�relo dentro del RAID.....	7
10. Compruebe que el RAID se recupera y sincroniza los datos.....	8
11. Cuando acabe la sincronizaci�n pruebe a eliminar otro elemento del RAID y compruebe que sigue funcionando	8
Ejercicios sobre los LVM.....	9
12. Cree un volumen f�sico.	9
13. Cree un grupo de vol�menes formado por el volumen f�sico creado en el punto 1.	10
14. Cree dos vol�menes l�gicos de 128Mb cada uno. El primero se usar� para almacenar datos temporales y el segundo para datos de usuario.	10
15. Monte el primer volumen en /temp y el segundo en /home (en este caso aseg�rese de haber copiado, previamente, los datos contenidos en /home).....	10
16. Cree ficheros en /home	11
17. Aumente el tama�o de /home en 100Mb y compruebe que el redimensionado se ha hecho efectivo.	11
Ejercicios sobre iSCSI.....	12
18. En el iSCSI Target (m�quina virtual que ser� el “servidor” iSCSI) creamos un disco virtual de 512Mb que ser� compartido (v�a iSCSI) con uno de los iSCSI Initiator. Este iSCSI Initiator montar� el volumen importado en /mnt/remote al arrancar el sistema.	12
Conclusi�n.....	13

Resumen

En esta práctica aprenderemos más acerca de Sistemas de Ficheros y dispositivos, para ellos necesitaremos añadir un disco duro nuevo a nuestra maquina virtual, crearle unas particiones y unas serie de modificaciones que le realizaremos a lo largo de la practica.

Cápítulo 1

5. Ejercicios

1. Cree una máquina virtual (o use alguna de las ya creadas) y asocie un nuevo disco virtual de 512Mb

Para esta practica he usado la maquina virtual de rocky Linux 8 para no volver a usar centos, y a si probar diferentes sistema o distribuciones y aprender más con la práctica.

2. En dicho disco duro virtual, cree tres particiones del mismo tamaño.

Para crear las particiones usamos el comando fdisk. Las he creado cada una de 150M

`fdisk /dev/sdc`

```
Orden (m para obtener ayuda): g
Se ha creado una nueva etiqueta de disco GPT (GUID: 5B88ADB3-3DD3-414E-865D-A89928477C81).

Orden (m para obtener ayuda): n
Número de partición (1-128, valor predeterminado 1): 1
Primer sector (2048-1048542, valor predeterminado 2048):
Último sector, +sectores o +tamaño{K,M,G,T,P} (2048-1048542, valor predeterminado 1048542): +156M

Crea una nueva partición 1 de tipo 'Linux filesystem' y de tamaño 156 MiB.

Orden (m para obtener ayuda): n
Número de partición (2-128, valor predeterminado 2): 2
Primer sector (321536-1048542, valor predeterminado 321536):
Último sector, +sectores o +tamaño{K,M,G,T,P} (321536-1048542, valor predeterminado 1048542): +156M

Crea una nueva partición 2 de tipo 'Linux filesystem' y de tamaño 156 MiB.

Orden (m para obtener ayuda): n
Número de partición (3-128, valor predeterminado 3): 3
Primer sector (641024-1048542, valor predeterminado 641024):
Último sector, +sectores o +tamaño{K,M,G,T,P} (641024-1048542, valor predeterminado 1048542): +156M

Crea una nueva partición 3 de tipo 'Linux filesystem' y de tamaño 156 MiB.
```

Cuando nos pida una nueva orden pondremos W si hemos terminado de crear las particiones, estos nos grabara estas y actualizara los discos.

3. Configure las tres particiones en RAID5.

Para crear las particiones usamos la utilidad mdadm donde tenemos que indicar dónde se va a alojar el raid, en este caso en /dev/md0. Qué nivel de raid, el número de dispositivos y la ruta de estos.

Si no tenemos instalado el paquete mdadm, lo podremos instalar con el siguiente comando:

`Sudo dnf install mdadm`

Una vez hecho eso usaremos el comando de crear el raid

```
sudo mdadm --create --verbose /dev/md0 --level=5 --raid-devices=3 /dev/sdc1 /dev/sdc2 /dev/sdc3
```

```
[root@localhost ~]# sudo mdadm --create --verbose /dev/md0 --level=5 --raid-devices=3 /dev/sdc1 /dev/sdc2 /dev/sdc3
mdadm: layout defaults to left-symmetric
mdadm: chunk size defaults to 512K
mdadm: size set to 157696K
mdadm: Defaulting to version 1.2 metadata
mdadm: array /dev/md0 started.
```

Si queremos formatear el nuevo arraid para que este limpio podremos usar este comando: mkfs.ext4 /dev/md0

4. Monte las raid en /mnt/RAID.

Primero tenemos que crear el directorio donde lo vamos a montar, nos vamos a /mnt y lo creamos.

```
mkdir /mnt/RAID
```

Una vez hecho esto, tenemos que construir el sistema de ficheros de tipo ext4, y luego, hacemos mount en /mnt/RAID.

```
mkfs.ext4 /dev/md0
```

```
[root@localhost ~]# sudo mkfs.ext4 /dev/md0
mke2fs 1.45.6 (20-Mar-2020)
/dev/md0 contiene un sistema de ficheros ext4
    fecha de creación Fri Dec 15 09:25:37 2023
¿Continuar de todas formas? (s,N) s
Se está creando un sistema de ficheros con 315392 bloques de 1k y 78936 nodos-i
UUID del sistema de ficheros: e558dedd-71f6-4d04-8e89-99704964d77e
Respalos del superbloque guardados en los bloques:
    8193, 24577, 40961, 57345, 73729, 204801, 221185

Reservando las tablas de grupo: hecho
Escribiendo las tablas de nodos-i: hecho
Creando el fichero de transacciones (8192 bloques): hecho
Escribiendo superbloques y la información contable del sistema de ficheros: hecho
```

```
mount /dev/md0 /mnt/RAID
```

5. Cree distintos ficheros.

Para crear distintos ficheros ponemos:

```
cat > fichero1.txt
```

Nos saldrá una línea sin nada y entonces podremos escribir lo que queramos, para guardar pulsamos ctrl+D

6. Simule la rotura de uno de los elementos del RAID.

Primero comprobamos el estado del raid, como vemos está todo bien y funcionan los tres.

cat /proc/mdstat

```
[root@localhost RAID]# cat /proc/mdstat
Personalities : [raid6] [raid5] [raid4]
md0 : active raid5 sdc3[3] sdc2[1] sdc1[0]
      315392 blocks super 1.2 level 5, 512k chunk, algorithm 2 [3/3] [UUU]
```

Con mdadm aparte de crear raids, podemos simular fallos poniendo -fail y el dispositivo que queremos que se rompa.

mdadm -fail /dev/md0 /dev/sdc1

```
[root@localhost RAID]# sudo mdadm --manage /dev/md0 --fail /dev/sdX1
mdadm: stat failed for /dev/sdX1: No such file or directory
[root@localhost RAID]# sudo mdadm --manage /dev/md0 --fail /dev/sdc1
mdadm: set /dev/sdc1 faulty in /dev/md0
[root@localhost RAID]# cat /proc/mdstat
Personalities : [raid6] [raid5] [raid4]
md0 : active raid5 sdc3[3] sdc2[1] sdc1[0](F)
      315392 blocks super 1.2 level 5, 512k chunk, algorithm 2 [3/2] [_UU]
```

Si queremos tener más datos de la rotura de la raid o información del raid, podemos hacer uso de esta utilidad de mdadm:

mdadm --detail /dev/md0

```
[root@localhost RAID]# mdadm --detail /dev/md0
/dev/md0:
  Version : 1.2
  Creation Time : Fri Dec 15 09:23:14 2023
  Raid Level : raid5
  Array Size : 315392 (308.00 MiB 322.96 MB)
  Used Dev Size : 157696 (154.00 MiB 161.48 MB)
  Raid Devices : 3
  Total Devices : 3
  Persistence : Superblock is persistent

  Update Time : Fri Dec 15 10:57:20 2023
  State : clean, degraded
  Active Devices : 2
  Working Devices : 2
  Failed Devices : 1
  Spare Devices : 0

  Layout : left-symmetric
  Chunk Size : 512K

Consistency Policy : resync

  Name : localhost.localdomain:0 (local to host localhost.localdomain)
  UUID : 1fe39f97:19e8da0c:5baf1e6e:83c87b0a
  Events : 20

   Number Major Minor RaidDevice State
    -   -   -   -   -   -
    0     8     0         0 removed
    1     8    34         1 active sync  /dev/sdc2
    3     8    35         2 active sync  /dev/sdc3

    0     8    33         - faulty   /dev/sdc1
```

7. Compruebe que puede seguir accediendo a los datos aunque uno de los discos esté marcado como estropeado.

Con cat podemos ver el contenido de los ficheros.

```
[root@localhost RAID]# cat /mnt/RAID/fichero1.txt
Fichero 1
```

8. Cree más ficheros en el RAID.

Creemos más ficheros para el raid con el cat, como en el apartador visto con anterioridad.

```
[root@localhost RAID]# cat > fichero5.txt
Fichero para roturas
[root@localhost RAID]# cat > fichero6.txt
Fichero de mas
```

9. Añada un nuevo disco, cree una partición de igual tamaño a las ya creadas y configúrelo dentro del RAID.

Con la utilidad fdisk creamos la partición de 150M.

```
[root@localhost ~]# fdisk /dev/sdb
Bienvenido a fdisk (util-linux 2.32.1).
Los cambios solo permanecerán en la memoria, hasta que decida escribirlos.
Tenga cuidado antes de utilizar la orden de escritura.

El dispositivo no contiene una tabla de particiones reconocida.
Se ha creado una nueva etiqueta de disco DOS con el identificador de disco 0x0fff668c.

Orden (m para obtener ayuda): g
Se ha creado una nueva etiqueta de disco GPT (GUID: F8D600EB-31B3-CE48-99E1-CE94523711EA).

Orden (m para obtener ayuda): n
Número de partición (1-128, valor predeterminado 1): 1
Primer sector (2048-1048542, valor predeterminado 2048):
Último sector, +sectores o +tamaño{K,M,G,T,P} (2048-1048542, valor predeterminado 1048542): +156M

Crea una nueva partición 1 de tipo 'Linux filesystem' y de tamaño 156 MiB.

Orden (m para obtener ayuda): w
Se ha modificado la tabla de particiones.
Llamando a ioctl() para volver a leer la tabla de particiones.
Se están sincronizando los discos.
```

Una vez hecha la partición toca añadirla al raid. Usamos mdadm con la función de añadir e indicamos dónde está el raid y dónde está el disco.

```
mdadm -add /dev/md0 /dev/sdb1
```

```
[root@localhost ~]# mdadm -add /dev/md0 /dev/sdb1
mdadm: An option must be given to set the mode before a second device
(/dev/md0) is listed
```

10. Compruebe que el RAID se recupera y sincroniza los datos.

Primero comprobamos el estado del RAID

```
[root@localhost ~]# cat /proc/mdstat
Personalities : [raid6] [raid5] [raid4]
md0 : active raid5 sdb1[4] sdc3[3] sdc2[1] sdc1[0](F)
      315392 blocks super 1.2 level 5, 512k chunk, algorithm 2 [3/2] [_UU]
      [====>.....] recovery = 30.0% (47996/157696) finish=0.2min speed=6856K/sec
```

Aquí podemos ver que el RAID está activo y que la partición nueva que hemos añadido aparece. Es importante recordar que hay que darle el formato EXT4 a la partición que queremos añadir al raid antes de añadirla, ya que esta se encuentra en ese formato.

```
mkfs.ext4 /dev/sdb1
```

11. Cuando acabe la sincronización pruebe a eliminar otro elemento del RAID y compruebe que sigue funcionando

Volvemos a eliminar otro elemento con mdadm y la utilidad fail.

```
mdadm --manage /dev/md0 --fail /dev/sdc2
```

```
[root@localhost ~]# sudo mdadm --manage /dev/md0 --fail /dev/sdc2
mdadm: set /dev/sdc2 faulty in /dev/md0
[root@localhost ~]# cat /proc/mdstat
Personalities : [raid6] [raid5] [raid4]
md0 : active raid5 sdb1[4] sdc3[3] sdc2[1](F) sdc1[0](F)
      315392 blocks super 1.2 level 5, 512k chunk, algorithm 2 [3/2] [U_U]

unused devices: <none>
[root@localhost ~]# sudo mdadm --remove /dev/md0 /dev/sdc2
mdadm: hot removed /dev/sdc2 from /dev/md0
[root@localhost ~]# cat /proc/mdstat
Personalities : [raid6] [raid5] [raid4]
md0 : active raid5 sdb1[4] sdc3[3] sdc1[0](F)
      315392 blocks super 1.2 level 5, 512k chunk, algorithm 2 [3/2] [U_U]

unused devices: <none>
```

Para remover la partición que hemos marcado como dañada usamos:

```
mdadm --remove /dev/md0 /dev/sdc2
```

Para comprobar que el raid sigue funcionando hacemos cat a un fichero de la misma para ver si nos deja leer el contenido.

```
[root@localhost ~]# cat /mnt/RAID/ficheros.txt
Fichero para roturas
```

Finalmente podemos ver que podemos acceder al fichero por lo que sigue funcionando.

Ejercicios sobre los LVM

12. Cree un volumen físico.

Utilizamos pvcreate para crear el volumen físico, primero ponemos pvcreate y luego el disco que queramos.

```
[root@localhost ~]# sudo pvcreate /dev/sdc
Physical volume "/dev/sdc" successfully created.
```

Para ver si se ha creado bien el volumen físico ponemos pvdisplay.

```
[root@localhost ~]# pvdisplay
--- Physical volume ---
PV Name           /dev/sda2
VG Name           centos
PV Size           <31,00 GiB / not usable 3,00 MiB
Allocatable       yes
PE Size           4,00 MiB
Total PE          7935
Free PE           1
Allocated PE      7934
PV UUID           jH17sd-zpFR-XBpB-q8xT-xOth-ZtWN-THOamu

"/dev/sdc" is a new physical volume of "512,00 MiB"
--- NEW Physical volume ---
PV Name           /dev/sdc
VG Name
PV Size           512,00 MiB
Allocatable       NO
PE Size           0
Total PE          0
Free PE           0
Allocated PE      0
PV UUID           rX97Ac-N07w-8yZ0-Wfz9-90bz-s4uY-vkcdf0
```

13. Cree un grupo de volúmenes formado por el volumen físico creado en el punto 1.

Utilizando el comando `vgcreate` podemos crear un grupo de volúmenes. Hay que indicar el nombre del grupo y los dispositivos que lo van a componer, en este caso solo va a haber un dispositivo y va a ser el volumen físico

```
vgcreate mi_grupo_volumenes/dev/sdc
```

```
[root@localhost ~]# sudo vgcreate mi_grupo_volumenes /dev/sdc
Volume group "mi_grupo_volumenes" successfully created
[root@localhost ~]#
```

14. Cree dos volúmenes lógicos de 128Mb cada uno. El primero se usará para almacenar datos temporales y el segundo para datos de usuario.

En este caso se utilizaría `lvcreate`, con la opción `-L` indicamos de cuánto espacio queremos que sea. Luego indicamos el grupo de volúmenes al que nos referimos y por último con `-n` añadimos el nombre al volumen lógico

```
lvcreate -L 128M -n vol1 mi_grupo_volumenes
```

```
[root@localhost ~]# lvcreate -L 128M -n vol1 mi_grupo_volumenes
Logical volume "vol1" created.
[root@localhost ~]# lvcreate -L 128M -n vol2 mi_grupo_volumenes
Logical volume "vol2" created.
[root@localhost ~]#
```

15. Monte el primer volumen en `/temp` y el segundo en `/home` (en este caso asegúrese de haber copiado, previamente, los datos contenidos en `/home`).

Para copiar las cosas de `home` usamos `cp`, con `-R` copiamos todos los directorios de forma recursiva y lo guardamos en `/mnt/home`.

```
sudo cp -r /home/* /mnt/home
```

```
[root@localhost ~]# sudo cp -r /home/* /mnt/home
```

Una vez hecho esto, antes de hacer nada más, tenemos que crear el sistema de ficheros para poder montarlo.

```
mkfs.ext4 /dev/grupovol1/vol1
```

```
[root@localhost ~]# mkfs.ext4 /dev/mi_grupo_volumenes/vol1
mke2fs 1.42.9 (28-Dec-2013)
Discarding device blocks: hecho
Etiqueta del sistema de ficheros=
OS type: Linux
Tamaño del bloque=1024 (bitácora=0)
Tamaño del fragmento=1024 (bitácora=0)
Stride=0 blocks, Stripe width=0 blocks
32768 inodes, 131072 blocks
6553 blocks (5.00%) reserved for the super user
Primer bloque de datos=1
Número máximo de bloques del sistema de ficheros=33685504
16 bloque de grupos
8192 bloques por grupo, 8192 fragmentos por grupo
2048 nodos-i por grupo
Respaldo del superbloque guardado en los bloques:
    8193, 24577, 40961, 57345, 73729

Allocating group tables: hecho
Escribiendo las tablas de nodos-i: hecho
Creating journal (4096 blocks): hecho
Escribiendo superbloques y la información contable del sistema de ficheros: hecho
```

Luego lo montamos con mount.

```
[root@localhost ~]# mount /dev/mi_grupo_volumenes/vol1 /mnt/tmp
[root@localhost ~]# mount /dev/mi_grupo_volumenes/vol2 /mnt/home
[root@localhost ~]#
```

16. Cree ficheros en /home

Creamos nuevos ficheros en el directorio home con:

`touch /mnt/home/archivo1`

```
[root@localhost ~]# touch /mnt/home/archivo1
[root@localhost ~]# touch /mnt/home/archivo2
[root@localhost ~]# ls /mnt/home
archivo1  archivo2  lost+found
```

17. Aumente el tamaño de /home en 100Mb y compruebe que el redimensionado se ha hecho efectivo.

Para aumentar el tamaño usamos `lvextend` con `-L` y ponemos la cantidad en la que lo queremos aumentar y luego indicamos el volumen.

`lvextend -L +100 /dev/grupovol1/vol2`

```
[root@localhost ~]# lvextend -L +100M /dev/mi_grupo_volumenes/vol2
Size of logical volume mi_grupo_volumenes/vol2 changed from 128,00 MiB (32 extents) to 228,00 MiB (57 extents).
Logical volume mi_grupo_volumenes/vol2 successfully resized.
[root@localhost ~]#
```

Ejercicios sobre iSCSI

18. En el iSCSI Target (máquina virtual que será el “servidor” iSCSI) creamos un disco virtual de 512Mb que será compartido (vía iSCSI) con uno de los iSCSI Initiator. Este iSCSI Initiator montará el volumen importado en /mnt/remote al arrancar el sistema.

Primero instalamos las utilidades del iscsi, el epel y por último el targetcli.

```
yum -y install epel-release && yum install iscsi-initiator-utils
```

```
yum install targetcli
```

```
yum install scsi-target-utils
```

```
systemctl enable tgtd
```

```
systemctl start tgtd
```

Una vez instalado todo, si usamos el paquete iscsi-target-utils modificamos el fichero /etc/tgt/targets.conf:

```
vi /etc/tgt/targets.conf
```

```
<target iqn.2023-12.localhost:CentOs7>
```

```
backing-store /dev/sdb
```

```
initiator-address 192.168.50.238
```

```
</target>
```

Donde 2023-12 es el año y el mes, localhost es el nombre del dominio en el cliente y CentOs7 es el nombre del objetivo. Backing-store es la ruta donde se encuentra lo que queremos compartir e initiator-address para indicar la IP que puede acceder a nuestro disco duro.

Hecho todo esto, queda iniciar el servicio y abrir el puerto

```
systemctl enable target
```

```
systemctl restart target
```

```
firewall-cmd --add-service=iscsi-target --permanent
```

```
firewall-cmd --reload
```

```
[root@localhost ~]# [root@localhost ~]# systemctl enable target
Created symlink from /etc/systemd/system/multi-user.target.wants/target.service to /usr/lib/systemd/system/target.service.
[root@localhost ~]# systemctl restart target
[root@localhost ~]# firewall-cmd --add-service=iscsi-target --permanent
success
[root@localhost ~]# firewall-cmd --reload
success
```


Ahora nos vamos a la parte del cliente donde instalamos el paquete iscsi-initiator-utils y editamos el archivo /etc/iscsi/initiatorname.iscsi y ponemos:

```
InitiatorName=localhost
```

A continuación, ponemos:

```
Iscsiadm -m Discovery -t sendtargets -p 192.168.50.56
```

```
systemctl restart iscsi
```

Ahora que ya tenemos disco duro compartido creamos un directorio en /mnt llamado remote para montarlo ahí. Para que lo monte cada vez que arranque editamos el fichero /etc/fstab

```
vi /etc/fstab
```

Finalmente escribimos en el fichero lo siguiente y cada vez que arranquemos se montará:

```
dev/sdb /mnt/remote ext4 auto,x-systemd.automount 0 0
```

Conclusión

En esta practica he aprendido bastante acerca de RAID y conjuntos de discos de almacenamiento, lo cual me ha gustado bastante y además la practica en si no me ha resultado pesada, a diferencia de las primeras que siempre estaban llenas de errores y bugs. Practicas asi te dan mas tranquilidad y son más cómodas y entretenidas de hacer, aunque no quiere decir que mejores.

La parte final de ISCSI es de lo que menos he entendido pero lo demás bastante conforme.



Universidad
de Huelva



Escuela Técnica Superior de
Ingeniería Universidad de Huelva

Memoria de Prácticas

ADMINISTRACIÓN DE SERVIDORES

Grado en Ingeniería Informática

Alejandro Gordillo Pedraza
28/12/2023

Contenido

ADMINISTRACIÓN DE SERVIDORES.....1

Grado en Ingeniería Informática1

Resumen3

Cápítulo 14

1. Ejercicios.....4

2. Conclusión.....11

Resumen

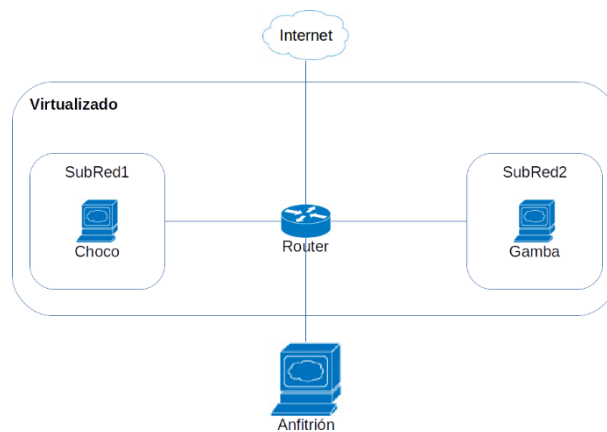
En esta practica vamos construir una red interna que consta de 2 VM choco y gamba que estarán conectados a una tercera vm la cual llamaremos router y hará de router para darle acceso a internet a estas dos maquinas.

C  pulo 1

La empresa ACME S.A, dispone de una red formado por tres ordenadores

- Choco: equipado con CentOS, un disco duro y una tarjeta de red.
- Gamba: equipado con CentOS, un disco duro y una tarjeta de red
- Router: equipado con CentOS y cuatro tarjetas de red

El ordenador router act  a como pasarela para las subredes en las que se encuentran gamba y choco. Adem  s, router permite tanto el acceso a internet de todos los equipos (choco, gamba, router) como el acceso del equipo anfitri  n al router.



Para realizar esta practica necesitaremos primero crear 3 m  quinas

virtuales:

Choco y gamba que les estableceremos el SO de Centos 7 y 1 tarjeta de red

Router que le estableceremos el SO Centos 7 y 4 tarjetas de red, esto lo haremos mediante proxmox en el apartado de hardware de la VM y ADD y a  ndiremos 3 tarjetas extras.

Summary	Add	Remove	Edit	Resize disk	Move disk	Revert
> Console	Memory	2.00 GiB				
Hardware	Processors	1 (1 sockets, 1 cores)				
Cloud-Init	BIOS	Default (SeaBIOS)				
Options	Display	Default				
Task History	Machine	Default (i440fx)				
Monitor	SCSI Controller	VirtIO SCSI				
Backup	CD/DVD Drive (ide2)	local:iso/CentOS-7-x86_64-Minimal-2207-02.iso,media=cdrom				
Replication	Hard Disk (scsi0)	local-lvm/vm-109-disk-0,size=32G				
Snapshots	Network Device (net0)	virtio=6E:CD:75:A5:22:80,bridge=vmbro,firewall=1				
Firewall	Network Device (net1)	virtio=22:C6:75:0E:2D:26,bridge=vmbro,firewall=1				
Permissions	Network Device (net2)	virtio=3A:1D:41:21:9A:20,bridge=vmbro,firewall=1				
	Network Device (net3)	virtio=D6:8E:CF:F2:D3:B9,bridge=vmbro,firewall=1				

Una vez tengamos instalado y arrancado nuestro sistema, como hemos visto en practicas anteriores, deberemos empezar a configurar nuestras tarjetas de ethernet.

Para el caso de Choco y Gamba estableceremos la ip como estática y deberemos tener la opción de onboot en Yes. Después como es una IP estática deberemos asignarle una Ip y una puerta de enlace, la cual será la Ip que le daremos a alguna de las tarjetas de red el router en el siguiente apartado. Nosotros también le añadiremos 2 DNS de Google, aunque no sé si sea necesario, y el Tipo de mascara.

Esto último lo haremos haciendo uso de la palabra PREFIX. La configuración debería quedar de la siguiente forma:

CHOCO

```
TYPE=Ethernet
PROXY_METHOD=none
BROWSER_ONLY=no
BOOTPROTO=static
DEFROUTE=yes
IPV4_FAILURE_FATAL=no
IPV6INIT=yes
IPV6_AUTOCONF=yes
IPV6_DEFROUTE=yes
IPV6_FAILURE_FATAL=no
IPV6_ADDR_GEN_MODE=stable-privacy
NAME=eth0
UUID=1364300a-ad36-4af3-9483-b1399871bcba
DEVICE=eth0
ONBOOT=yes
IPADDR=192.168.51.10
PREFIX=20
GATEWAY=192.168.51.142
DNS1=8.8.8.8
DNS2=8.8.4.4
```

GAMBA

```
TYPE="Ethernet"
PROXY_METHOD="none"
BROWSER_ONLY="no"
BOOTPROTO="static"
DEFROUTE="yes"
IPV4_FAILURE_FATAL="no"
IPV6INIT="yes"
IPV6_AUTOCONF="yes"
IPV6_DEFROUTE="yes"
IPV6_FAILURE_FATAL="no"
IPV6_ADDR_GEN_MODE="stable-privacy"
NAME="eth0"
UUID="e7a717c6-7658-4422-94af-614bd2a4006a"
DEVICE="eth0"
ONBOOT="yes"
IPADDR=192.168.52.10
PREFIX=20
GATEWAY=192.168.51.144
DNS1=8.8.8.8
DNS2=8.8.4.4
```

En el caso de nuestra VM Router deberemos configurar las 4 tarjetas de red. En el caso de la Eth0 estableceremos una Ip dinámica, debido a que esta será la tarjeta que nos de acceso a internet, para esto le estableceremos una ip dhcp o simplemente dejaremos la configuración de default.

Para las siguientes 3 tarjetas tendremos que crear documentos nuevos desde 0, para ello bastara con copiar y pegar el contenido de la configuración de la Eth0 y modificarlo como queramos nosotros y añadir algunos datos como hicimos con choco y gamba.

En el caso de estas 3 tarjetas de red restantes deberemos darle una IP estática y la Ip que le pongamos será la que hayamos elegido como puerta de enlace en las otras 2 maquinas virtuales.

En nuestro caso Eth1 hará de puerta para Choco y Eth2 hará de puerta para Gamba. Dejando así Eth3 para el anfitrión/Host.

Eth0

```
PROXY_METHOD="none"
BROWSER_ONLY="no"
BOOTPROTO="dhcp"
DEFROUTE="yes"
IPV4_FAILURE_FATAL="no"
IPV6INIT="yes"
IPV6_AUTOCONF="yes"
IPV6_DEFROUTE="yes"
IPV6_FAILURE_FATAL="no"
IPV6_ADDR_GEN_MODE="stable-privacy"
NAME="eth0"
UUID="dd27887a-e4d1-4c21-a66d-989fe7d4d800"
DEVICE="eth0"
ONBOOT="yes"
DNS1=8.8.8.8
DNS2=8.8.4.4
```

Eth1

```
TYPE=Ethernet
BOOTPROTO=static
DEFROUTE=no
PEERDNS=yes
PEERROUTES=yes
NAME=eth1
DEVICE=eth1
ONBOOT=yes
IPADDR=192.168.51.142
PREFIX=20
```

Eth2

```
TYPE=Ethernet
BOOTPROTO=static
DEFROUTE=no
PEERDNS=yes
PEERROUTES=yes
NAME=eth2
DEVICE=eth2
ONBOOT=yes
IPADDR=192.168.51.144
PREFIX=20
```

Eth3

```
TYPE=Ethernet
BOOTPROTO=static
DEFROUTE=no
PEERDNS=yes
PEERROUTES=yes
NAME=eth3
DEVICE=eth3
ONBOOT=yes
IPADDR=192.168.51.143
PREFIX=20
```

Una vez que tengamos nuestras tarjetas de red bien configuradas en cada maquina virtual deberemos reiniciar la red para asegurarnos de que todo esta correcto, para ello haremos uso del comando:

Service restart network

```
[root@localhost ~]# service network restart
Restarting network (via systemctl): [ OK ]
[root@localhost ~]#
```


Una vez nuestras redes están correctas y no presentan ningún fallo le daremos al router la configuración para que pueda hacer de router, osea la función para que enrute las redes que le llegan:

Vi /etc/sysctl.conf

net.ipv4.ip_forward = 1

```
# sysctl settings are defined through files in
# /usr/lib/sysctl.d/, /run/sysctl.d/, and /etc/sysctl.d/.
#
# Vendors settings live in /usr/lib/sysctl.d/.
# To override a whole file, create a new file with the same in
# /etc/sysctl.d/ and put new settings there. To override
# only specific settings, add a file with a lexically later
# name in /etc/sysctl.d/ and put new settings there.
#
# For more information, see sysctl.conf(5) and sysctl.d(5).
net.ipv4.ip_forward = 1
```

Después comprobamos que esto haya sido cambiado correctamente

sysctl -p

```
[root@localhost ~]# sudo sysctl -p
net.ipv4.ip_forward = 1
[root@localhost ~]#
```

Una vez lo comprobamos y confirmamos procedemos a configurar las iptables, deberemos establecer la eth que nos de acceso a internet en la iptable.

iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE

sudo service iptables save

En mi caso no podía guardar las iptables por que tenia que instalar paquetes que no tenía y use estos comandos de actualización/instalación:

sudo yum install iptables-services

sudo systemctl enable iptables

sudo systemctl start iptables

Una vez hemos acabado de configurar las iptables pasaremos a configurar el firewall. Para ello simplemente le estableceremos que de acceso a las dns y a las masquerade:

```
sudo firewall-cmd --zone=public --add-service=dns --permanent
```

```
sudo firewall-cmd --zone=public --add-masquerade --permanent
```

```
sudo firewall-cmd --reload
```

```
[root@localhost ~]# sudo firewall-cmd --zone=public --add-masquerade --permanent
success
[root@localhost ~]# sudo firewall-cmd --reload
success
```

Si queremos revisar que todo esta correcto podremos comprobar el firewall con este comando:

```
sudo firewall-cmd --list-all
```

```
[root@localhost ~]# sudo firewall-cmd --list-all
public (active)
  target: default
  icmp-block-inversion: no
  interfaces: eth0 eth1 eth2 eth3
  sources:
  services: dhcpv6-client dns ssh
  ports:
  protocols:
  masquerade: yes
  forward-ports:
  source-ports:
  icmp-blocks:
  rich rules:
```

Una vez tengamos ya todo configurado, deberemos probar la conectividad entre maquinas y el acceso a internet:

Gamba Router

```
[root@localhost ~]# ping 192.168.51.144
PING 192.168.51.144 (192.168.51.144) 56(84) bytes of data:
64 bytes from 192.168.51.144: icmp_seq=1 ttl=64 time=0.925 ms
64 bytes from 192.168.51.144: icmp_seq=2 ttl=64 time=0.549 ms
64 bytes from 192.168.51.144: icmp_seq=3 ttl=64 time=0.704 ms
64 bytes from 192.168.51.144: icmp_seq=4 ttl=64 time=0.510 ms
64 bytes from 192.168.51.144: icmp_seq=5 ttl=64 time=0.666 ms
^C
```

Gamba Internet

```
[root@localhost ~]# ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data:
64 bytes from 8.8.8.8: icmp_seq=1 ttl=115 time=14.4 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=115 time=13.6 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=115 time=13.8 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=115 time=13.7 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=115 time=13.7 ms
64 bytes from 8.8.8.8: icmp_seq=6 ttl=115 time=13.7 ms
^Z
```

Choco Router

```
[root@localhost ~]# ping 192.168.51.142
PING 192.168.51.142 (192.168.51.142) 56(84) bytes of data:
64 bytes from 192.168.51.142: icmp_seq=1 ttl=64 time=1.48 ms
64 bytes from 192.168.51.142: icmp_seq=2 ttl=64 time=1.05 ms
64 bytes from 192.168.51.142: icmp_seq=3 ttl=64 time=0.477 ms
64 bytes from 192.168.51.142: icmp_seq=4 ttl=64 time=0.563 ms
^Z
```

Choco Internet

```
[root@localhost ~]# ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data:
64 bytes from 8.8.8.8: icmp_seq=1 ttl=115 time=15.1 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=115 time=13.6 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=115 time=13.4 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=115 time=13.3 ms
```

Conclusion

En esta practica hemos podido aprender como convertir una VM con Centos 7 en un router funcional para nuestro sistema de red o d ordenadores. También creo poder formar una especie de intranet gracias a lo que aprendí en esta practica.



Universidad
de Huelva



Escuela Técnica
Superior de Ingeniería
Universidad de Huelva

Memoria de Prácticas

ADMINISTRACIÓN DE
SERVIDORES

Grado en Ingeniería
Informática

Alejandro Gordillo Pedraza
06/01/2023

Contenido

ADMINISTRACIÓN DE SERVIDORES	1
Grado en Ingeniería Informática.....	1
Resumen.....	3
Cápítulo 1	4
Capítulo 2	5
Conclusión	10

Resumen

En esta práctica aprenderemos más acerca de instalación de aplicaciones por repositorio y del traspaso de archivos entre ordenadores conectados entre si mediante una red. Para la instalación de repositorios usaremos joe como prueba y transferiremos la instalación del archivo comprimida a otra VM para ver como se instala con solo la configuración.

C  pulo 1

Instale desde el c  digo fuente la versi  n 2.9.4 del JOE. Tenga en cuenta que es necesario instalar las herramientas de desarrollo oportunas para proceder a la compilaci  n del editor JOE.

Para instalar JOE en nuestra VM (Router en nuestro caso) usaremos la instrucci  n wget y el repositorio lo obtendremos de sourceforge.

En nuestro caso usaremos la   ltima versi  n de JOE en vez de la recomendada para reducir las posibilidades de fallos. La descarga la deberemos hacer desde el directorio /usr/src

`Cd /usr/src`

`Wget https://sourceforge.net/projects/joe-editor/`

Si no nos inicia la descarga puede ser porque no tengamos wget instalado, yum install wget

En mi caso se me descarg   con el nombre de download, as   que le cambie el nombre haciendo uso del mv:

`mv download joe-4.6.tar.gz`

```
[root@localhost ~]# ls
anaconda-ks.cfg  index.html  joe-4.6.tar.gz
```

Una vez descargado el archivo en nuestra VM deberemos tener instalado el paquete de developer, el cual incluye el compilador de C.

`Yum groupinstall "Development Tools"`

Una vez descargado e instalado el paquete developer, tendremos que descomprimir el archivo:

`tar -xvzf joe-4.6.tar.gz`

El siguiente paso que debemos hacer una vez descomprimido el archivo es configurarlo. Para ello entraremos en la dirección de Joe y lo configuraremos desde allí:

```
cd joe-4.6
```

```
./configure
```

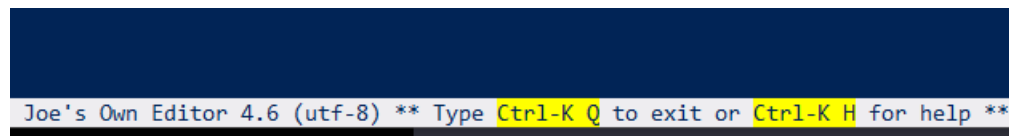
Seguido haremos Make para la instalación del JOE

```
Make
```

Y después el install

```
Make install
```

Una vez terminado el proceso comprobamos que esta instalado y la versión que queríamos, para ello usaremos `joe --version` o `joe`, de no funciona el versión en nuestro SO



Capítulo 2

Una vez hecha la instalación, instálelo en otra máquina virtual. Para ello:

1. Usando el comando oportuno (`tar` o `cpio`), haga una copia de seguridad de todos los ficheros que se crearon durante la instalación del editor JOE

Para hacer una copia de seguridad de todos los ficheros, primero voy a compilar todos los ficheros y ejecutables que tengan que ver con el funcionamiento de JOE.

```
tar -czvf joe-backup.tar.gz /usr/local/bin/joe /usr/local/etc/joe
```



```
[root@localhost joe-4.6]# tar -czvf joe-backup.tar.gz /usr/local/bin/joe /usr/local/etc/joe
tar: Eliminando la '/' inicial de los nombres
/usr/local/bin/joe
/usr/local/etc/joe/
/usr/local/etc/joe/joerc
/usr/local/etc/joe/jicerc.ru
/usr/local/etc/joe/jmacsrc
/usr/local/etc/joe/jstarrc
/usr/local/etc/joe/rjoerc
/usr/local/etc/joe/jpicorc
/usr/local/etc/joe/joerc.zh_TW
/usr/local/etc/joe/ftyporc
/usr/local/etc/joe/shell.sh
/usr/local/etc/joe/shell.csh
[root@localhost joe-4.6]# ls
acinclude.m4  colors          config.sub      depcomp        install-sh      Makefile.in    rc
aclocal.m4    compile        configure      desktop        joe             man            README.md
autojoe       config.guess   configure.ac    docs           joe-backup.tar.gz missing         setup.hint
ChangeLog     config.log     COPYING        INSTALL.AMIGA  Makefile        NEWS.md        syntax
charmmaps     config.status  cygbuild       INSTALL.md     Makefile.am     po
```

2. Transfiera el fichero resultante a gamba usando el comando scp o rsync

Para este paso necesitaremos saber la dirección IP, un usuario y la dirección destino. Todo esto de la VM a la que queremos transferir el archivo.

En nuestro caso mandaremos el archivo a la carpeta raíz por lo que veremos a continuación:

scp joe-backup.tar.gz [root@192.168.52.10:/](mailto:root@192.168.52.10/)

La primera vez que enviemos un archivo a una VM nos pedirá permisos de confianza y algunas preguntas más. Y cada vez que lo hagamos nos pedirá la contraseña de la VM

```
[root@localhost joe-4.6]# scp joe-backup.tar.gz root@192.168.52.10:/root
The authenticity of host '192.168.52.10 (192.168.52.10)' can't be established.
ECDSA key fingerprint is SHA256:gL2IRjHKz87zyPTqAQJAqDrCxxv50GPnBWYwD9+6HN4w.
ECDSA key fingerprint is MD5:56:25:88:5f:03:4c:88:d0:f4:68:ff:9f:37:28:46:6e.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.52.10' (ECDSA) to the list of known hosts.
root@192.168.52.10's password:
joe-backup.tar.gz                                     100% 947KB 8.3MB/s 00:00
```

***No hacer caso al /root de la foto, fue el primer intento y me equivoque, hay que madarlo a la carpeta raíz, no a la root.**

3. Descomprima el fichero y compruebe que funciona

Nos vamos a gamba y descomprimos el fichero como vimos antes. Si todo esta bien y el comprimido contenía los archivos necesarios deberán colocarse solos automáticamente en sus carpetas.

De no colocarse solo en sus direcciones, deberemos hacerlo nosotros manualmente.

Una vez descomprimido comprobaremos que se haya instalado bien con el comando Joe

Joe



Tras realizar la instalación de dicho software, mande un mensaje a todos los usuarios de ambos sistemas (estén o no conectados) indicando que está disponible este nuevo editor.

Para ello haremos uso del wall y funciona de la siguiente manera:

Wall [Mensaje] Ctrl + D para guardar el mensaje

```
[root@localhost joe-4.6]# wall
hola equipo ya tenemos joe activo
[root@localhost joe-4.6]#
Broadcast message from root@localhost.localdomain (pts/0) (Fri Jan  5 10:46:42 2024):
hola equipo ya tenemos joe activo
```

Por último, se propone realizar un sistema de copias de seguridad incrementales de los directorios /home de ambos servidores. Estas copias se realizarán todos los días. La idea es que las copias de seguridad del servidor1 se almacenen en el directorio /backups/ del servidor2 y las del servidor2 en el directorio /backups/ del servidor1. Para programar las copias de seguridad, use el comando cron.

Primero deberemos instalar la función rsync

yum install rsync

```
[root@localhost ~]# sudo yum install rsync
Complementos cargados:fastestmirror
Determining fastest mirrors
 * base: ftp.cica.es
 * extras: ftp.cica.es
 * updates: ftp.cica.es
base | 3.6 kB 00:00:00
extras | 2.9 kB 00:00:00
updates | 2.9 kB 00:00:00
El paquete rsync-3.1.2-12.el7_9.x86_64 ya se encuentra instalado con su versión más reciente
Nada para hacer
```

Después crearemos el directorio de los backups.

mkdir -p /backups/

Seguido crearemos el contador de copias de seguridad, que hará que nos hagan copia de seguridad a las 2 am cada día:

Contrab -e

Y escribiremos esto en el archivo:

0 2 * * * rsync -avz /home/ root@192.168.52.10:/backups/router_home_backup

```
0 2 * * * rsync -avz /home/ root@192.168.52.10:/backups/router_home_backup
```

En mi caso he hecho una copia de seguridad manual para verificar que funciona correctamente:

rsync -avz /home/ root@192.168.52.10:/backups/router_home_backup

```
root@localhost /]# cd /backups
root@localhost backups]# ls
router_home_backup
root@localhost backups]#
```

Para facilitar todo un poco, hare uso de la herramienta de ssh keys, para que guarde la conexión como segura y la clave de la maquina contraria y así no tener que poner la contraseña con cada copia de seguridad:

```
ssh-keygen -t rsa -b 4096
```

```
[root@localhost ~]# ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa): clave
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in clave.
Your public key has been saved in clave.pub.
The key fingerprint is:
SHA256:p5kTIE4E6QyKA1pdnxgvVPh573ewpR8xuupSxDYLM6o root@localhost.localdomain
The key's randomart image is:
+---[RSA 4096]-----+
|  .+..+O.          |
| O O....= .        |
|== O +.+..         |
| = oo . oo..=      |
| . .   S.B.O o     |
|   O o. o +       |
|   * .. . *       |
|   . o . = o      |
|   E. oo.o o.     |
+---[SHA256]-----+
```

Después tocara mandar la clave privada al servidor remoto. En nuestro caso a gamba:

```
ssh-copy-id -i /root/clave.pub root@192.168.52.10
```

Y una vez recibida la clave, ya no debería pedirnos contraseña al traspasar archivos entre máquina. Si todo está correcto.

Conclusión

En esta practica he de decir que lo que más me costo fue la descarga del JOE, ya que escuchaba a mis compañeros decir que la versión 2.9.4 no les estaba resultando fácil de usar. Entonces decidí probar con una versión más nueva y esta fue una solución acertada, porque no me ha presentado casi ningún problema.

Respecto al resto de la práctica, las cosas han sido medio intuitivas, pero si es verdad que he tenido que transferir más archivos de los que quisiera porque me equivoque con algunas rutas.