



UNIVERSIDAD DE GRANADA

Ingeniería de Servidores

Cuestiones opcionales

Javier León Palomares

19 de enero de 2017

Índice

1. Práctica 1.	5
1.1. Cuestión opcional 1: Muestre (con capturas de pantalla) cómo ha comprobado que el RAID1 funciona.	5
1.2. Cuestión opcional 2: ¿Qué relación hay entre los atajos de teclado de emacs y los de la consola bash? ¿Y entre los de vi y las páginas del manual? . .	7
2. Práctica 2.	7
2.1. Cuestión opcional 1: Instale y pruebe terminator y/o tmux. Con screen, pruebe su funcionamiento dejando sesiones ssh abiertas en el servidor y recuperándolas posteriormente.	7
2.2. Cuestión opcional 2: Instale el servicio (<i>fail2ban</i>) y pruebe su funcionamiento.	9
2.3. Cuestión opcional 3: Instale el servicio (<i>rkhunter</i>) y pruebe su funcionamiento.	11
3. Práctica 3.	15
3.1. Cuestión opcional 1: Indique qué comandos ha utilizado para realizarlo, así como capturas de pantalla del proceso de reconstrucción del RAID. . .	15
3.2. Cuestión opcional 2: Instale Nagios en su sistema (el que prefiera) documentando el proceso y muestre el resultado de la monitorización de su sistema comentando qué aparece.	19
3.3. Cuestión opcional 5: Pruebe a instalar este monitor (Cacti) en alguno de sus tres sistemas. Realice capturas de pantalla del proceso de instalación y comente capturas de pantalla del programa en ejecución.	22
4. Práctica 4.	27
4.1. Cuestión opcional 1: ¿Qué es Scala? Instale Gatling y pruebe los escenarios por defecto.	27
5. Práctica 5.	29
5.1. Cuestión opcional 1: Realice lo mismo que en la cuestión 6 pero para otro servicio.	29

Índice de figuras

1.	Simulamos un fallo mediante <code>mdadm</code>	5
2.	Podemos ver cómo el disco principal ha pasado a un estado incorrecto y no está activo.	5
3.	Mediante la opción <code>-r</code> eliminamos el disco y mediante la opción <code>-a</code> lo volvemos a añadir.	5
4.	El dispositivo se encuentra al 73 % del proceso de reconstrucción al momento de ver su estado.	6
5.	Examinamos de nuevo los detalles tras la recuperación del dispositivo. . .	6
6.	Varias terminales abiertas simultáneamente con <i>terminator</i>	7
7.	<i>terminator</i> junto con dos terminales que usan <i>screen</i> . La fecha que se ve en la sesión de ssh corresponde a antes de hacer <code>detach</code> con <i>screen</i>	8
8.	En la terminal que antes contenía ssh se ve que hemos dejado de trabajar con dicho servicio momentáneamente.	8
9.	Seleccionando a qué tarea queremos volver.	9
10.	Como se aprecia, pocos minutos después hemos vuelto a la sesión de ssh sin problemas.	9
11.	Comprobamos que <i>fail2ban</i> está bien instalado.	10
12.	Añadimos el parámetro <code>maxretry</code> con valor 2 para que nos bloquee la IP al segundo intento fallido de login con ssh.	10
13.	Intentos sucesivos de login por ssh desde mi máquina local, identificada por 192.168.56.1	11
14.	El primer intento de desbloqueo de la IP, llevado a cabo antes de fallar al hacer login, resulta en error ya que no estaba bloqueada. El segundo, realizado tras el bloqueo, no da error; esto nos permite entrar normalmente, como se evidencia en el último intento de la captura anterior.	11
15.	Salida parcial por consola de una instalación correcta de <i>rkhunter</i>	11
16.	Comprobación de programas típicos del sistema.	12
17.	Búsqueda de <i>rootkits</i> (conjuntos de herramientas maliciosas que buscan dar acceso no autorizado en condiciones normales).	13
18.	Detección de más <i>rootkits</i> , malware y operaciones específicas de Linux. . .	13
19.	Escaneos de red y del host. Aquí vemos que detecta que el administrador tiene algún tipo de acceso, como se comentaba anteriormente.	14
20.	Comprobaciones finales de algunas versiones y resumen de todo lo realizado. Además, nos indica dónde se ha guardado el log correspondiente. . . .	14
21.	Sección del log generado por la ejecución de <i>rkhunter</i> . Aquí se evidencian dos de los falsos positivos.	15
22.	El RAID identificado por <code>md0</code> funciona correctamente, con los dos discos operativos.	15
23.	Ahora hay otro disco (sin usar) en la máquina virtual.	15
24.	Ejecución de <code>lsblk</code> antes de tocar el RAID.	16
25.	Creando una partición con <code>fdisk</code>	16
26.	Extracto de <code>lsblk</code> mostrando que, efectivamente, se ha creado una partición. .	16
27.	Arriba vemos que sólo hay uno de los dos discos funcionando (<code>[2/1]</code>) tras ejecutar el comando de la parte de abajo.	17

28.	Eliminamos sda1 y comprobamos que desaparece de la lista de discos activos en /proc/mdstat	17
29.	Añadimos el nuevo disco al RAID.	18
30.	Proceso completado.	18
31.	La salida de lsblk nos dice que el dispositivo sdc1 ha reemplazado a sda1	19
32.	Le decimos que no queremos cambiar la configuración del correo.	19
33.	Elegimos una contraseña para acceder a <i>Nagios</i>	20
34.	Cuadro de diálogo para introducir usuario y contraseña. El usuario por defecto es nagiosadmin	20
35.	Página principal de <i>Nagios</i>	20
36.	Aquí vemos los datos más relevantes de la monitorización. Todo está correcto, excepto por un error que muestra en la sección de servicios. Por ello, iremos a ver qué ocurre.	21
37.	El error detectado en la captura anterior pertenece al sistema de discos. Sabemos que funciona, pero por falta de permisos <i>Nagios</i> no es capaz de acceder a los datos que necesita y por ello lanza un error.	21
38.	Página específica para el disco. Obtenemos la misma respuesta: no puede acceder a algunos datos y, debido a eso, lanza un error. Sin embargo, el parámetro de latencia, con un valor realista, parece indicarnos que está activo de todas formas.	22
39.	Pantalla que nos pregunta si queremos que la base de datos que utilizará <i>Cacti</i> sea configurada automáticamente.	22
40.	También nos solicita una contraseña para trabajar con <i>Cacti</i>	23
41.	Pantalla que nos da a elegir qué servidor queremos que use el monitor.	23
42.	Cuando llegamos a la página principal, se inicia un pequeño proceso de instalación previo a poder usar el monitor.	23
43.	Le decimos que es una nueva instalación.	24
44.	Paso final de la instalación web.	24
45.	Ventana de login.	25
46.	Ventana de cambio de contraseña la primera vez que entramos a <i>Cacti</i>	25
47.	Pantalla principal de <i>Cacti</i>	26
48.	Aquí podemos crear nuevos gráficos a partir de plantillas. Los disponibles inicialmente son los que aparecen en el menú desplegable.	26
49.	Gráfico de uso de memoria. Debido a que apenas acaba de empezar a monitorizar, no está relleno horizontalmente. Además, ya que el sistema está totalmente en reposo al momento de hacer las capturas, no se consume mucha memoria.	27
50.	Gráfico de carga media. El sistema no presenta carga significativa porque no se ha estado usando mientras exploraba los menús de <i>Cacti</i>	27
51.	Ejecución de gatling.sh , en la que se nos da a elegir entre varias simulaciones tras (a juzgar por el tiempo que tarda) compilarlas todas.	28
52.	Resultados finales de la ejecución de <i>Gatling</i> mostrados por consola.	28
53.	Aquí se ve una de las gráficas generadas para el documento HTML que nos da <i>Gatling</i> . Comprobamos cómo, de las 13 solicitudes realizadas, 12 de ellas han tardado menos de 800 milisegundos, mientras que una ha tardado entre 800 y 1200. Esto concuerda con la información de la figura anterior.	28

54.	Página inicial de <i>nginx</i> , que está ejecutándose en mi máquina virtual de Ubuntu Server (192.168.56.101).	29
55.	Ejecución de <code>ab</code> antes de modificar parámetros de <i>nginx</i>	29
56.	Ejecución de <code>ab</code> tras modificar el parámetro <code>access_log</code>	30
57.	Ejecución de <code>ab</code> con el parámetro <code>sendfile off</code>	31

1. Práctica 1.

1.1. Cuestión opcional 1: Muestre (con capturas de pantalla) cómo ha comprobado que el RAID1 funciona.

He seguido las instrucciones del enlace proporcionado en el guión de prácticas [1].

1. En primer lugar hemos de simular un fallo en el disco principal del RAID:

```
jlp@UbuntuServerISEdom oct 23:~$ sudo mdadm --manage --set-faulty /dev/md0 /dev/sda1
[ 3255.514671] md/raid1:md0: Disk failure on sda1, disabling device.
[ 3255.514671] md/raid1:md0: Operation continuing on 1 devices.
mdadm: set /dev/sda1 faulty in /dev/md0
```

Figura 1: Simulamos un fallo mediante mdadm.

2. Después comprobamos el estado del RAID:

```
jlp@UbuntuServerISEdom oct 23:~$ sudo mdadm --detail /dev/md0
/dev/md0:
  Version : 1.2
  Creation Time : Fri Oct 21 16:12:09 2016
  Raid Level : raid1
  Array Size : 8382464 (7.99 GiB 8.58 GB)
  Used Dev Size : 8382464 (7.99 GiB 8.58 GB)
  Raid Devices : 2
  Total Devices : 2
  Persistence : Superblock is persistent

  Update Time : Sun Oct 23 12:34:48 2016
  State : clean, degraded
Active Devices : 1
Working Devices : 1
Failed Devices : 1
Spare Devices : 0

    Name : UbuntuServerISE:0 (local to host UbuntuServerISE)
    UUID : 30489afe:7ca1b1ee:ce739b0f:bf6bf3d0
    Events : 23

   Number Major Minor RaidDevice State
    0         0      0        0  removed
    1         8     17        1  active sync  /dev/sdb1
    0         8      1        -  faulty    /dev/sda1
```

Figura 2: Podemos ver cómo el disco principal ha pasado a un estado incorrecto y no está activo.

3. Posteriormente deberemos quitar el disco y volverlo a conectar para poder restaurar el funcionamiento normal del RAID:

```
jlp@UbuntuServerISEdom oct 23:~$ sudo mdadm /dev/md0 -r /dev/sda1
mdadm: hot removed /dev/sda1 from /dev/md0
jlp@UbuntuServerISEdom oct 23:~$ sudo mdadm /dev/md0 -a /dev/sda1
mdadm: added /dev/sda1
```

Figura 3: Mediante la opción `-r` eliminamos el disco y mediante la opción `-a` lo volvemos a añadir.

4. A continuación, podemos examinar de nuevo el estado para ver que se está reconstruyendo:

```
jlp@UbuntuServerISEdom oct 23:~$ sudo mdadm --detail /dev/md0
/dev/md0:
    Version : 1.2
  Creation Time : Fri Oct 21 16:12:09 2016
    Raid Level : raid1
    Array Size : 8382464 (7.99 GiB 8.58 GB)
  Used Dev Size : 8382464 (7.99 GiB 8.58 GB)
    Raid Devices : 2
  Total Devices : 2
 Persistence : Superblock is persistent

 Update Time : Sun Oct 23 12:40:49 2016
   State : clean, degraded, recovering
 Active Devices : 1
Working Devices : 2
 Failed Devices : 0
  Spare Devices : 1

Rebuild Status : 73% complete

    Name : UbuntuServerISE:0 (local to host UbuntuServerISE)
   UUID : 30489afe:7ca1b1ee:ce739b0f:bf6bf3d0
  Events : 73

   Number  Major   Minor  RaidDevice State
     2       8       1        0   spare rebuilding /dev/sda1
     1       8      17        1   active sync /dev/sdb1
jlp@UbuntuServerISEdom oct 23:~$
```

Figura 4: El dispositivo se encuentra al 73% del proceso de reconstrucción al momento de ver su estado.

5. Para finalizar, se muestra cómo el RAID ha sido reactivado con éxito y vuelve a tener un estado correcto:

```
jlp@UbuntuServerISEdom oct 23:~$ sudo mdadm --detail /dev/md0
/dev/md0:
    Version : 1.2
  Creation Time : Fri Oct 21 16:12:09 2016
    Raid Level : raid1
    Array Size : 8382464 (7.99 GiB 8.58 GB)
  Used Dev Size : 8382464 (7.99 GiB 8.58 GB)
    Raid Devices : 2
  Total Devices : 2
 Persistence : Superblock is persistent

 Update Time : Sun Oct 23 12:41:42 2016
   State : clean
 Active Devices : 2
Working Devices : 2
 Failed Devices : 0
  Spare Devices : 0

    Name : UbuntuServerISE:0 (local to host UbuntuServerISE)
   UUID : 30489afe:7ca1b1ee:ce739b0f:bf6bf3d0
  Events : 83

   Number  Major   Minor  RaidDevice State
     2       8       1        0   active sync /dev/sda1
     1       8      17        1   active sync /dev/sdb1
jlp@UbuntuServerISEdom oct 23:~$
```

Figura 5: Examinamos de nuevo los detalles tras la recuperación del dispositivo.

1.2. Cuestión opcional 2: ¿Qué relación hay entre los atajos de teclado de emacs y los de la consola bash? ¿Y entre los de vi y las páginas del manual?

Bash utiliza la librería **readline**, que por defecto usa atajos de teclado de emacs para la interacción con el usuario. Por ello, bash también los emplea. [2]

Por su parte, las páginas del manual se pueden explorar mediante los atajos de teclado de vi (por ejemplo, / para buscar o las flechas para desplazarse).

2. Práctica 2.

2.1. Cuestión opcional 1: Instale y pruebe terminator y/o tmux. Con screen, pruebe su funcionamiento dejando sesiones ssh abiertas en el servidor y recuperándolas posteriormente.

Voy a realizar esta cuestión con *terminator*. Con `apt-cache search terminator` sé que está en los repositorios que usa mi máquina local, así que puedo instalarlo con el gestor de paquetes. Una muestra de su correcto funcionamiento se ve en la figura 7.

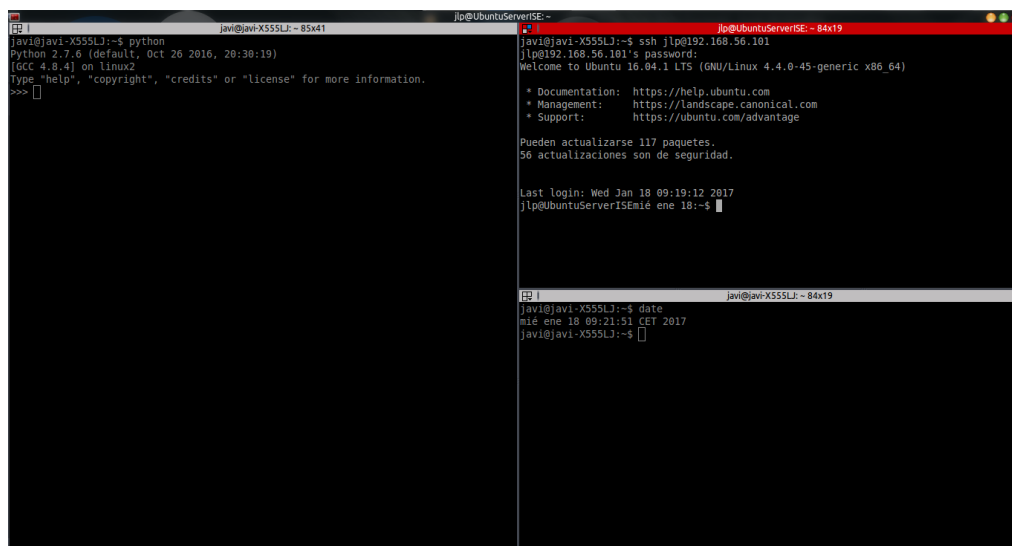


Figura 6: Varias terminales abiertas simultáneamente con *terminator*.

Una vez hecha la primera prueba, es el momento de introducir *screen*. De forma similar se puede averiguar cómo instalarlo con el gestor de paquetes. En la siguiente captura (figura 7) podemos observar dos terminales abiertas con *screen*: una de ellas muestra atajos de teclado y otra tiene una sesión de ssh abierta.

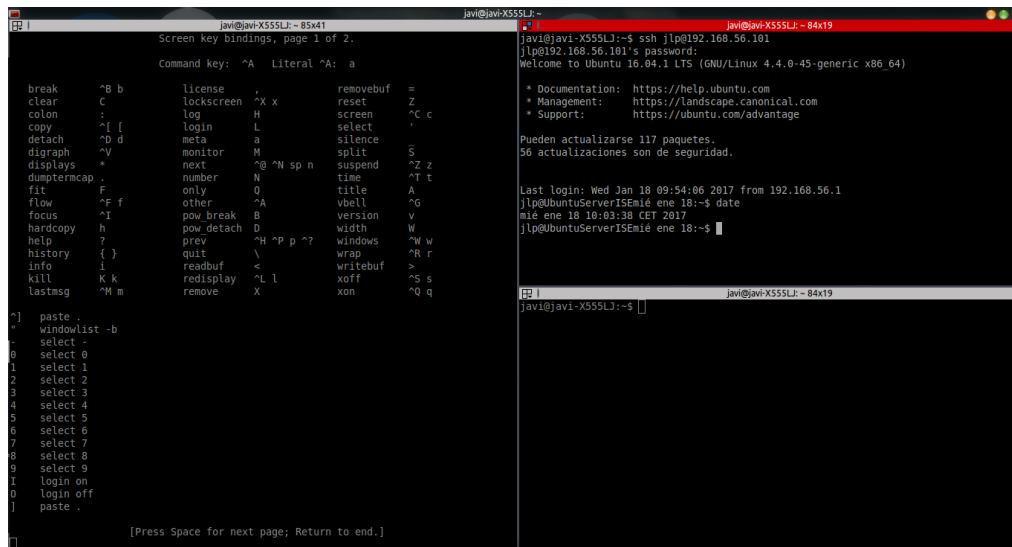


Figura 7: *terminator* junto con dos terminales que usan *screen*. La fecha que se ve en la sesión de ssh corresponde a antes de hacer **detach** con *screen*.

Tras esto, dejaremos la sesión abierta haciendo **detach** con el atajo de teclado correspondiente, y ocurrirá algo como lo mostrado en la figura 8.

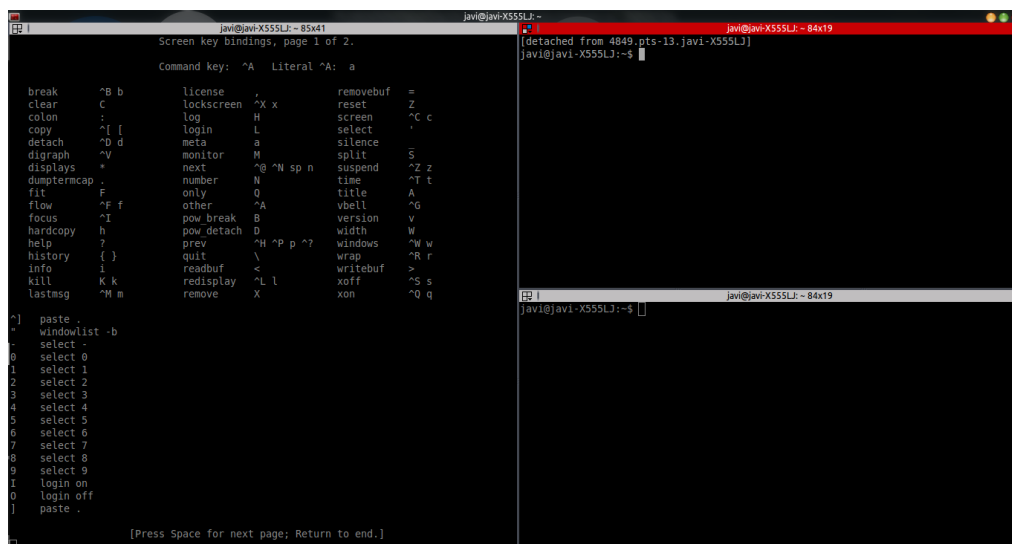


Figura 8: En la terminal que antes contenía ssh se ve que hemos dejado de trabajar con dicho servicio momentáneamente.

Ahora vamos a volver a la sesión de ssh. Intentando **screen -r** vemos que hay más de una tarea a la que volver, así que seleccionaremos la que acabamos de dejar especificándola como se nos indica (figura 9).

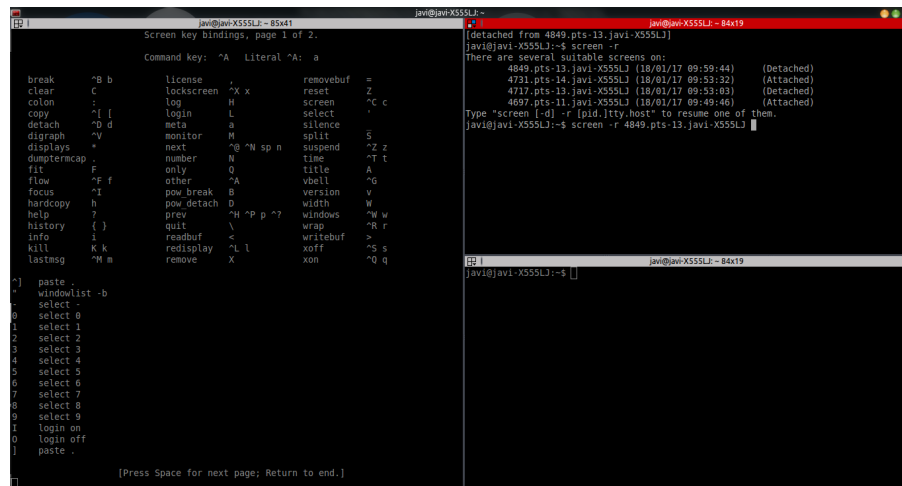


Figura 9: Seleccionando a qué tarea queremos volver.

Finalmente, en la figura 10 tenemos la prueba de que hemos vuelto a la sesión de ssh que habíamos dejado abierta.

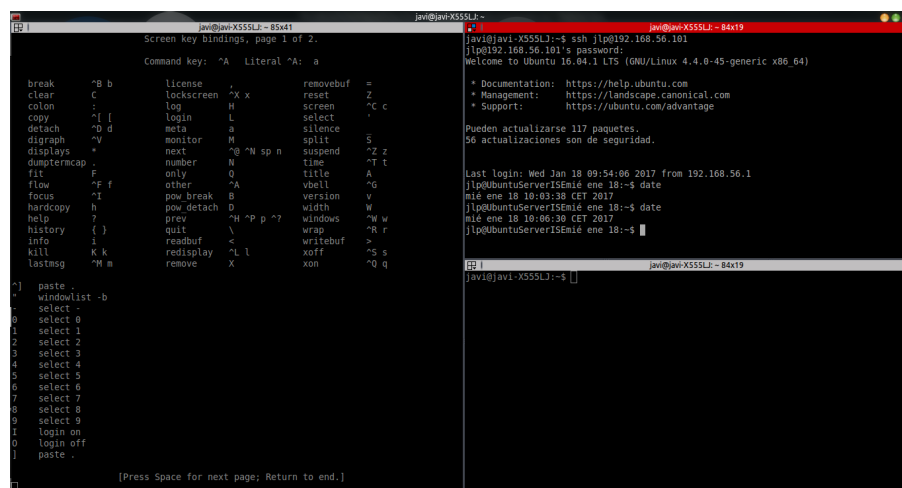


Figura 10: Como se aprecia, pocos minutos después hemos vuelto a la sesión de ssh sin problemas.

2.2. Cuestión opcional 2: Instale el servicio (*fail2ban*) y pruebe su funcionamiento.

Según la información disponible en el repositorio oficial de *fail2ban*, quizá podamos instalarlo directamente desde el gestor de paquetes. Una rápida búsqueda (`apt-cache search fail2ban`) indica que, efectivamente, está disponible; con `sudo apt-get install fail2ban` lo instalaremos.

Para comprobar que el proceso se ha realizado correctamente, nos dicen que ejecutemos `fail2ban-client -h` (que muestra la ayuda). Como se puede ver en la figura 11, no hay errores.

```
get <JAIL> action <ACT> actionstart      COMMAND ACTION INFORMATION
                                           gets the start command for the
                                           action <ACT> for <JAIL>
get <JAIL> action <ACT> actionstop         gets the stop command for the
                                           action <ACT> for <JAIL>
get <JAIL> action <ACT> actioncheck        gets the check command for the
                                           action <ACT> for <JAIL>
get <JAIL> action <ACT> actionban          gets the ban command for the
                                           action <ACT> for <JAIL>
get <JAIL> action <ACT> actionunban        gets the unban command for the
                                           action <ACT> for <JAIL>
get <JAIL> action <ACT> timeout            gets the command timeout in
                                           seconds for the action <ACT> for
                                           <JAIL>

get <JAIL> actionproperties <ACT>         GENERAL ACTION INFORMATION
                                           gets a list of properties for the
                                           action <ACT> for <JAIL>
get <JAIL> actionmethods <ACT>            gets a list of methods for the
                                           action <ACT> for <JAIL>
get <JAIL> action <ACT> <PROPERTY>        gets the value of <PROPERTY> for
                                           the action <ACT> for <JAIL>

Report bugs to https://github.com/fail2ban/fail2ban/issues
jlp@UbuntuServerISEmar ene 17:~$
```

Figura 11: Comprobamos que *fail2ban* está bien instalado.

A continuación, he seguido un tutorial [3] para probar su funcionamiento. Lo primero que hemos de hacer es crear una copia del archivo de configuración por defecto, `/etc/fail2ban/jail.conf`, que se llamará `/etc/fail2ban/jail.local`. Dentro de ese nuevo archivo, añadimos la línea `maxretry 2` bajo el apartado del servicio `ssh` (figura 12).

```
#
# JAILS
#

#
# SSH servers
#

[sshd]

port      = ssh
logpath   = %(sshd_log)s
maxretry  = 2
```

Figura 12: Añadimos el parámetro `maxretry` con valor 2 para que nos bloquee la IP al segundo intento fallido de login con `ssh`.

Después de reiniciar el servicio de *fail2ban*, vamos a ver si funciona el cambio. Como se puede apreciar en la figura 13, realizamos dos intentos fallidos; al tercero, se rechaza la conexión sin pedir siquiera la contraseña. El último intento, exitoso, ocurre tras desbloquear la IP manualmente (también podríamos haber esperado el tiempo que dura el bloqueo); esto lo podemos observar en la figura 14.

```
javi@javi-X555LJ:~$ ssh jlp@192.168.56.101
jlp@192.168.56.101's password:
Permission denied, please try again.
jlp@192.168.56.101's password:
Connection closed by 192.168.56.101
javi@javi-X555LJ:~$ ssh jlp@192.168.56.101
ssh: connect to host 192.168.56.101 port 22: Connection refused
javi@javi-X555LJ:~$ ssh jlp@192.168.56.101
jlp@192.168.56.101's password:
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-45-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

Pueden actualizarse 117 paquetes.
56 actualizaciones son de seguridad.

Last login: Tue Jan 17 17:07:51 2017 from 192.168.56.1
jlp@UbuntuServerISEmar ene 17:~$
```

Figura 13: Intentos sucesivos de login por ssh desde mi máquina local, identificada por 192.168.56.1 .

```
jlp@UbuntuServerISEmar ene 17:~$ sudo fail2ban-client set sshd unbanip 192.168.56.1
ERROR NOK: ('IP 192.168.56.1 is not banned',)
IP 192.168.56.1 is not banned
jlp@UbuntuServerISEmar ene 17:~$ sudo fail2ban-client set sshd unbanip 192.168.56.1
192.168.56.1
jlp@UbuntuServerISEmar ene 17:~$ _
```

Figura 14: El primer intento de desbloqueo de la IP, llevado a cabo antes de fallar al hacer login, resulta en error ya que no estaba bloqueada. El segundo, realizado tras el bloqueo, no da error; esto nos permite entrar normalmente, como se evidencia en el último intento de la captura anterior.

2.3. Cuestión opcional 3: Instale el servicio (*rkhunter*) y pruebe su funcionamiento.

Voy a utilizar el *readme* disponible para el programa [4] como guía.

Para comenzar, debemos descargar el archivo con la última versión oficial y descomprimirlo. A continuación, hay que ejecutar el script `installer.sh` con la opción `--install`. Si el proceso ha concluido con éxito, se debe mostrar algo parecido al extracto contenido en la figura 15.

```
Installing check_modules.pl: OK
Installing filehashsha.pl: OK
Installing stat.pl: OK
Installing readlink.sh: OK
Installing backdoorports.dat: OK
Installing mirrors.dat: OK
Installing programs_bad.dat: OK
Installing suspscan.dat: OK
Installing rkhunter.8: OK
Installing ACKNOWLEDGMENTS: OK
Installing CHANGELOG: OK
Installing FAQ: OK
Installing LICENSE: OK
Installing README: OK
Installing language support files: OK
Installing ClamAV signatures: OK
Installing rkhunter: OK
Installing rkhunter.conf: OK
Installation complete
jlp@UbuntuServerISEmar ene 17:~/rkhunter/rkhunter-1.4.2$
```

Figura 15: Salida parcial por consola de una instalación correcta de *rkhunter*.

Ya está todo listo para ejecutar *rkhunter* en la máquina virtual de Ubuntu Server. Las capturas que siguen a estas líneas corresponden al resultado de `sudo rkhunter --check`. En ellas podemos ver lo que posiblemente sean falsos positivos en muchos casos, ya que consultando el log final se puede ver (figura 21) que algunos comandos generen avisos debido a tener `POSIX shell` añadido al final de sus nombres. Sin embargo, y por poner un ejemplo, detecta correctamente que el administrador tiene acceso por `ssh`: sólo puede entrar mediante el sistema de clave pública-privada (`PermitRootLogin prohibit-password`), pero no está totalmente restringido.

```
jlp@UbuntuServerISEmar ene 17:~/rkhunter/rkhunter-1.4.2$ sudo rkhunter --check
[ Rootkit Hunter version 1.4.2 ]

Checking system commands...

Performing 'strings' command checks
  Checking 'strings' command                      [ Skipped ]

Performing 'shared libraries' checks
  Checking for preloading variables                [ None found ]
  Checking for preloaded libraries                 [ None found ]
  Checking LD_LIBRARY_PATH variable                [ Not found ]

Performing file properties checks
  Checking for prerequisites                       [ OK ]
  /usr/local/bin/rkhunter                         [ OK ]
  /usr/sbin/adduser                               [ Warning ]
  /usr/sbin/chroot                                [ OK ]
  /usr/sbin/cron                                  [ OK ]
  /usr/sbin/groupadd                              [ OK ]
  /usr/sbin/groupdel                              [ OK ]
  /usr/sbin/groupmod                              [ OK ]
  /usr/sbin/grpck                                  [ OK ]
  /usr/sbin/nologin                               [ OK ]
  /usr/sbin/pwck                                   [ OK ]
  /usr/sbin/rsyslogd                              [ OK ]
  /usr/sbin/sshd                                   [ OK ]
  /usr/sbin/tcpd                                   [ OK ]
  /usr/sbin/useradd                               [ OK ]
  /usr/sbin/userdel                               [ OK ]
  /usr/sbin/usermod                               [ OK ]
  /usr/sbin/vipw                                   [ OK ]
  /usr/bin/awk                                     [ OK ]
  /usr/bin/basename                               [ OK ]
  /usr/bin/chattr                                  [ OK ]
  /usr/bin/curl                                    [ OK ]
  /usr/bin/cut                                     [ OK ]
```

Figura 16: Comprobación de programas típicos del sistema.

```
Checking for rootkits...

Performing check of known rootkit files and directories
55808 Trojan - Variant A [ Not found ]
ADM Worm [ Not found ]
AjaKit Rootkit [ Not found ]
Adore Rootkit [ Not found ]
aPa Kit [ Not found ]
Apache Worm [ Not found ]
Ambient (ark) Rootkit [ Not found ]
Balaor Rootkit [ Not found ]
BeastKit Rootkit [ Not found ]
beX2 Rootkit [ Not found ]
BOBKit Rootkit [ Not found ]
cb Rootkit [ Not found ]
CiNIK Worm (Slapper.B variant) [ Not found ]
Danny-Boy's Abuse Kit [ Not found ]
Devil RootKit [ Not found ]
Dica-Kit Rootkit [ Not found ]
Dreams Rootkit [ Not found ]
Duarawkz Rootkit [ Not found ]
Enye LKM [ Not found ]
Flea Linux Rootkit [ Not found ]
Fu Rootkit [ Not found ]
```

Figura 17: Búsqueda de *rootkits* (conjuntos de herramientas maliciosas que buscan dar acceso no autorizado en condiciones normales).

```
Performing additional rootkit checks
Suckit Rootkit additional checks [ OK ]
Checking for possible rootkit files and directories [ None found ]
Checking for possible rootkit strings [ Skipped ]

Performing malware checks
Checking running processes for suspicious files [ None found ]
Checking for login backdoors [ None found ]
Checking for suspicious directories [ None found ]
Checking for sniffer log files [ None found ]
Suspicious Shared Memory segments [ None found ]

Performing Linux specific checks
Checking loaded kernel modules [ OK ]
Checking kernel module names [ OK ]
```

Figura 18: Detección de más *rootkits*, malware y operaciones específicas de Linux.

```
Checking the network...

Performing checks on the network ports
Checking for backdoor ports [ None found ]

Performing checks on the network interfaces
Checking for promiscuous interfaces [ None found ]

Checking the local host...

Performing system boot checks
Checking for local host name [ Found ]
Checking for system startup files [ Found ]
Checking system startup files for malware [ None found ]

Performing group and account checks
Checking for passwd file [ Found ]
Checking for root equivalent (UID 0) accounts [ None found ]
Checking for passwordless accounts [ None found ]
Checking for passwd file changes [ None found ]
Checking for group file changes [ None found ]
Checking root account shell history files [ None found ]

Performing system configuration file checks
Checking for an SSH configuration file [ Found ]
Checking if SSH root access is allowed [ Warning ]
Checking if SSH protocol v1 is allowed [ Not allowed ]
Checking for a running system logging daemon [ Found ]
Checking for a system logging configuration file [ Found ]
Checking if syslog remote logging is allowed [ Not allowed ]

Performing filesystem checks
Checking /dev for suspicious file types [ None found ]
Checking for hidden files and directories [ Warning ]
```

Figura 19: Escaneos de red y del host. Aquí vemos que detecta que el administrador tiene algún tipo de acceso, como se comentaba anteriormente.

```
Checking application versions...

Checking version of GnuPG [ OK ]
Checking version of OpenSSL [ OK ]
Checking version of OpenSSH [ OK ]

System checks summary
=====

File properties checks...
Files checked: 139
Suspect files: 5

Rootkit checks...
Rootkits checked : 267
Possible rootkits: 0

Applications checks...
Applications checked: 3
Suspect applications: 0

The system checks took: 6 minutes and 48 seconds

All results have been written to the log file: /var/log/rkhunter.log

One or more warnings have been found while checking the system.
Please check the log file (/var/log/rkhunter.log)

jlp@UbuntuServerISEmar ene 17:~/rkhunter/rkhunter-1.4.2$
```

Figura 20: Comprobaciones finales de algunas versiones y resumen de todo lo realizado. Además, nos indica dónde se ha guardado el log correspondiente.

```

[21:08:32] /bin/dmesg [ OK ]
[21:08:32] /bin/echo [ OK ]
[21:08:32] /bin/ed [ OK ]
[21:08:32] /bin/egrep [ Warning ]
[21:08:32] Warning: The command '/bin/egrep' has been replaced by a script: /bin/egrep: POSIX shell$
[21:08:32] /bin/fgrep [ Warning ]
[21:08:32] Warning: The command '/bin/fgrep' has been replaced by a script: /bin/fgrep: POSIX shell$
[21:08:32] /bin/fuser [ OK ]
[21:08:32] /bin/grep [ OK ]
[21:08:32] /bin/zip [ OK ]

```

Figura 21: Sección del log generado por la ejecución de *rkhunter*. Aquí se evidencian dos de los falsos positivos.

3. Práctica 3.

3.1. Cuestión opcional 1: Indique qué comandos ha utilizado para realizarlo, así como capturas de pantalla del proceso de reconstrucción del RAID.

El archivo que muestra el estado de nuestro RAID es `/proc/mdstat`. La información correspondiente al dispositivo funcionando sin errores es la siguiente:

```

jlp@UbuntuServerISEmié ene 18:~$ cat /proc/mdstat
Personalities : [raid1] [linear] [multipath] [raid0] [raid6] [raid5] [raid4] [raid10]
md0 : active raid1 sda1[2] sdb1[1]
      8382464 blocks super 1.2 [2/2] [UU]

unused devices: <none>
jlp@UbuntuServerISEmié ene 18:~$

```

Figura 22: El RAID identificado por md0 funciona correctamente, con los dos discos operativos.

El siguiente paso es añadir un disco vacío a la máquina virtual de Ubuntu Server. El resultado se ve a continuación:

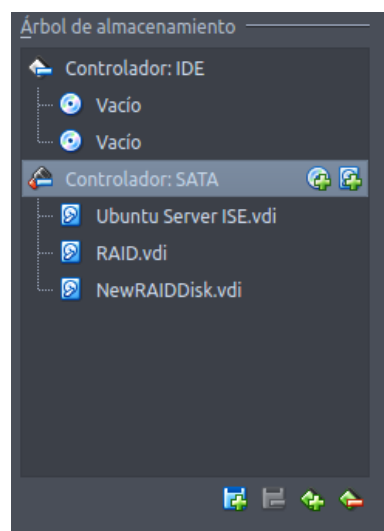


Figura 23: Ahora hay otro disco (sin usar) en la máquina virtual.

Mediante `lsblk` sabemos principalmente dos cosas. La primera, ya conocida, es que los dos discos que ya existían forman una unidad RAID funcional. La segunda es que el disco que hemos insertado se ha detectado correctamente (`sdc`), aunque no tiene aún una partición. Todo esto se muestra en la figura 24.

```
j1lp@UbuntuServerISEmié ene 18:~$ lsblk
NAME                                MAJ:MIN RM  SIZE RO TYPE  MOUNTPOINT
sda                                  8:0      0   8G  0 disk
├─sda1                              8:1      0   8G  0 part
│   └─md0                           9:0      0   8G  0 raid1
│       ├─Grupo1-arranq             252:0      0 408M  0 lvm  /boot
│       ├─Grupo1-hogar              252:1      0 472M  0 lvm
│       │   └─Grupo1-hogar_crypt    252:6      0 470M  0 crypt /home
│       ├─Grupo1-raiz               252:2      0  3,7G  0 lvm
│       │   └─Grupo1-raiz_crypt    252:4      0  3,7G  0 crypt /
│       ├─Grupo1-swap               252:3      0  1,4G  0 lvm
│       └─Grupo1-swap_crypt        252:5      0  1,4G  0 crypt [SWAP]
sdb                                  8:16     0   8G  0 disk
├─sdb1                              8:17     0   8G  0 part
│   └─md0                           9:0      0   8G  0 raid1
│       ├─Grupo1-arranq             252:0      0 408M  0 lvm  /boot
│       ├─Grupo1-hogar              252:1      0 472M  0 lvm
│       │   └─Grupo1-hogar_crypt    252:6      0 470M  0 crypt /home
│       ├─Grupo1-raiz               252:2      0  3,7G  0 lvm
│       │   └─Grupo1-raiz_crypt    252:4      0  3,7G  0 crypt /
│       ├─Grupo1-swap               252:3      0  1,4G  0 lvm
│       └─Grupo1-swap_crypt        252:5      0  1,4G  0 crypt [SWAP]
sdc                                  8:32     0   8G  0 disk
sr0                                 11:0     1 1024M  0 rom
sr1                                 11:1     1 1024M  0 rom
j1lp@UbuntuServerISEmié ene 18:~$
```

Figura 24: Ejecución de `lsblk` antes de tocar el RAID.

Para poder usar el nuevo disco en el RAID, debemos particionarlo [5]. Un resumen de esto se ve en las figuras 25 (inicio del proceso con `fdisk`) y 26 (comprobación de que tiene una partición).

```
j1lp@UbuntuServerISEmié ene 18:~$ sudo fdisk /dev/sdc
Welcome to fdisk (util-linux 2.27.1).
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.
Created a new DOS disklabel with disk identifier 0x74bde663.

Command (m for help): n
Partition type
  p   primary (0 primary, 0 extended, 4 free)
  e   extended (container for logical partitions)
Select (default p): _
```

Figura 25: Creando una partición con `fdisk`.

```
sdc                                  8:32     0   8G  0 disk
├─sdc1                              8:33     0   8G  0 part
```

Figura 26: Extracto de `lsblk` mostrando que, efectivamente, se ha creado una partición.

Después de los preparativos y las comprobaciones anteriores, vamos a comenzar el proceso de sustituir uno de los dos discos del RAID por el nuevo. Voy a usar `tmux` para dividir la pantalla en dos secciones: en la superior monitorizaré el contenido de `/etc/mdstat` usando `watch`, y en la inferior realizaré todas las operaciones necesarias.

Siguiendo la misma referencia que en la primera cuestión opcional de la práctica 1, lo primero que hay que hacer es poner uno de los dos discos del RAID en un estado con fallos:

```
Every 2.0s: cat /proc/mdstat                               Wed Jan 18 13:17:00 2017
Personalities : [raid1] [linear] [multipath] [raid0] [raid6] [raid5] [raid4] [raid10]
md0 : active raid1 sda1[2](F) sdb1[1]
      8382464 blocks super 1.2 [2/1] [_U]

unused devices: <none>

jlp@UbuntuServerISEmié ene 18:~$ sudo mdadm --manage --set-faulty /dev/md0 /dev/sda1
[sudo] password for jlp:
mdadm: set /dev/sda1 faulty in /dev/md0re on sda1, disabling device.
jlp@UbuntuServerISEmié ene 18:~$ ation continuing on 1 devices.
```

Figura 27: Arriba vemos que sólo hay uno de los dos discos funcionando ([2/1]) tras ejecutar el comando de la parte de abajo.

Es el momento de quitarlo del RAID:

```
Every 2.0s: cat /proc/mdstat                               Wed Jan 18 13:18:08 2017
Personalities : [raid1] [linear] [multipath] [raid0] [raid6] [raid5] [raid4] [raid10]
md0 : active raid1 sdb1[1]
      8382464 blocks super 1.2 [2/1] [_U]

unused devices: <none>

jlp@UbuntuServerISEmié ene 18:~$ sudo mdadm --manage --set-faulty /dev/md0 /dev/sda1
[sudo] password for jlp:
mdadm: set /dev/sda1 faulty in /dev/md0re on sda1, disabling device.
jlp@UbuntuServerISEmié ene 18:~$ ation continuing on 1 devices.
jlp@UbuntuServerISEmié ene 18:~$ sudo mdadm /dev/md0 -r /dev/sda1
mdadm: hot removed /dev/sda1 from /dev/md0
jlp@UbuntuServerISEmié ene 18:~$
```

Figura 28: Eliminamos `sda1` y comprobamos que desaparece de la lista de discos activos en `/proc/mdstat`.

Procedemos a añadir `sdc1` al RAID (figura 29) y vemos que el proceso se completa con éxito (figura 30).

```
Every 2.0s: cat /proc/mdstat                                Wed Jan 18 13:19:22 2017

Personalities : [raid1] [linear] [multipath] [raid0] [raid6] [raid5] [raid4] [raid10]
md0 : active raid1 sdc1[2] sdb1[1]
      8382464 blocks super 1.2 [2/1] [_U]
      [=>.....] recovery = 11.9% (1005568/8382464) finish=0.8min speed=143652K/sec

unused devices: <none>

jlp@UbuntuServerISEmié ene 18:~$ sudo mdadm --manage --set-faulty /dev/md0 /dev/sda1
[sudo] password for jlp:
mdadm: set /dev/sda1 faulty in /dev/md0re on sda1, disabling device.
jlp@UbuntuServerISEmié ene 18:~$ sudo mdadm /dev/md0 --rebuild /dev/sda1
mdadm: hot removed /dev/sda1 from /dev/md0
jlp@UbuntuServerISEmié ene 18:~$ sudo mdadm /dev/md0 --add /dev/sdc1
mdadm: added /dev/sdc1
jlp@UbuntuServerISEmié ene 18:~$
```

Figura 29: Añadimos el nuevo disco al RAID.

```
Every 2.0s: cat /proc/mdstat                                Wed Jan 18 13:21:12 2017

Personalities : [raid1] [linear] [multipath] [raid0] [raid6] [raid5] [raid4] [raid10]
md0 : active raid1 sdc1[2] sdb1[1]
      8382464 blocks super 1.2 [2/2] [UU]

unused devices: <none>

jlp@UbuntuServerISEmié ene 18:~$ sudo mdadm --manage --set-faulty /dev/md0 /dev/sda1
[sudo] password for jlp:
mdadm: set /dev/sda1 faulty in /dev/md0re on sda1, disabling device.
jlp@UbuntuServerISEmié ene 18:~$ sudo mdadm /dev/md0 --rebuild /dev/sda1
mdadm: hot removed /dev/sda1 from /dev/md0
jlp@UbuntuServerISEmié ene 18:~$ sudo mdadm /dev/md0 --add /dev/sdc1
mdadm: added /dev/sdc1
jlp@UbuntuServerISEmié ene 18:~$ _
```

Figura 30: Proceso completado.

Finalmente, vamos a ejecutar otra vez `lsblk` para asegurarnos de que todo está correcto y para comprobar las diferencias con el estado anterior del RAID:

```
j1peUbuntuServerISEmié ene 18:~$ lsblk
NAME                                MAJ:MIN RM  SIZE RO TYPE  MOUNTPOINT
sda                                  8:0      0   8G  0 disk
├─sda1                              8:1      0   8G  0 part
sdb                                  8:16     0   8G  0 disk
├─sdb1                              8:17     0   8G  0 part
└─md0                               9:0      0   8G  0 raid1
    ├─Grupo1-arranq                 252:0     0  408M  0 lvm    /boot
    ├─Grupo1-hogar                  252:1     0  472M  0 lvm
    │   └─Grupo1-hogar_crypt         252:6     0  470M  0 crypt  /home
    │   ├─Grupo1-raiz                252:2     0   3,7G  0 lvm
    │   │   └─Grupo1-raiz_crypt      252:4     0   3,7G  0 crypt  /
    │   └─Grupo1-swap               252:3     0   1,4G  0 lvm
    │       └─Grupo1-swap_crypt      252:5     0   1,4G  0 crypt  [SWAP]
sdc                                  8:32     0   8G  0 disk
├─sdc1                              8:33     0   8G  0 part
└─md0                               9:0      0   8G  0 raid1
    ├─Grupo1-arranq                 252:0     0  408M  0 lvm    /boot
    ├─Grupo1-hogar                  252:1     0  472M  0 lvm
    │   └─Grupo1-hogar_crypt         252:6     0  470M  0 crypt  /home
    │   ├─Grupo1-raiz                252:2     0   3,7G  0 lvm
    │   │   └─Grupo1-raiz_crypt      252:4     0   3,7G  0 crypt  /
    │   └─Grupo1-swap               252:3     0   1,4G  0 lvm
    │       └─Grupo1-swap_crypt      252:5     0   1,4G  0 crypt  [SWAP]
sr0                                  11:0     1 1024M  0 rom
sr1                                  11:1     1 1024M  0 rom
j1peUbuntuServerISEmié ene 18:~$
```

Figura 31: La salida de `lsblk` nos dice que el dispositivo `sdc1` ha reemplazado a `sda1`.

3.2. Cuestión opcional 2: Instale Nagios en su sistema (el que prefiera) documentando el proceso y muestre el resultado de la monitorización de su sistema comentando qué aparece.

Usando el tutorial disponible para Ubuntu [6], instalamos *Nagios* desde el gestor de paquetes con `sudo apt install nagios3 nagios-nrpe-plugin`; también hemos de tener en cuenta que necesita *Apache* para ofrecernos los datos. Durante el proceso, se nos pide que elijamos si queremos cambiar la configuración del correo (figura 32) y que definamos una contraseña (figura 33).

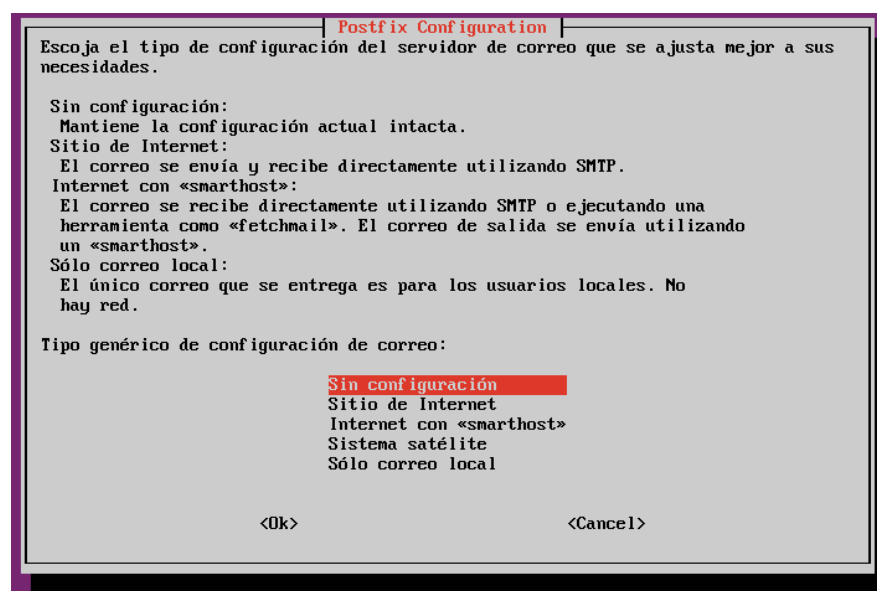
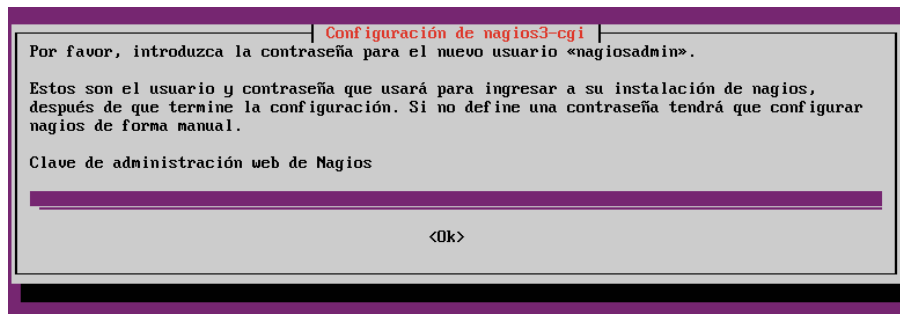
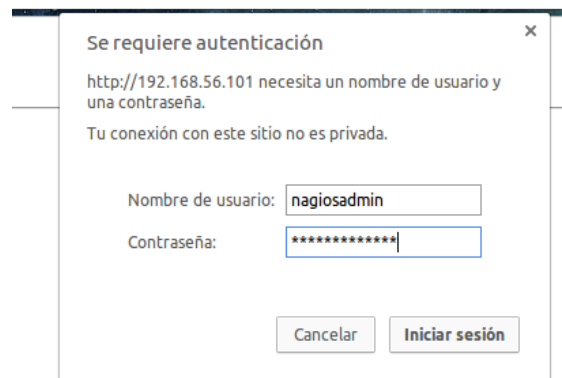
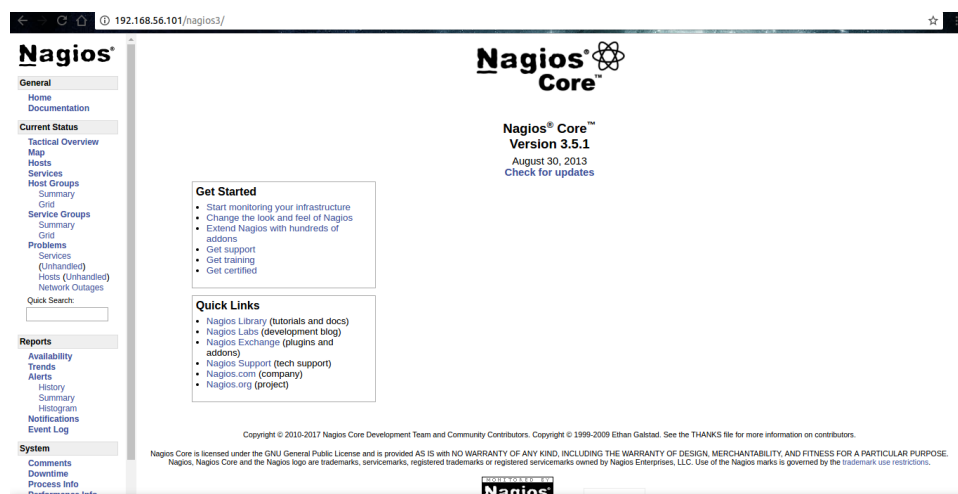


Figura 32: Le decimos que no queremos cambiar la configuración del correo.

Figura 33: Elegimos una contraseña para acceder a *Nagios*.

Si la instalación ha concluido satisfactoriamente, podemos acceder a *Nagios* desde el navegador usando la IP del servidor, en este caso `192.168.56.101/nagios3`. El cuadro de diálogo que se nos presenta para iniciar sesión es el de la figura 34, y la pantalla principal la vemos en la figura 35.

Figura 34: Cuadro de diálogo para introducir usuario y contraseña. El usuario por defecto es *nagiosadmin*.Figura 35: Página principal de *Nagios*.

Nagios nos permite controlar remotamente aspectos del sistema como los servicios o los hosts configurados, y además es capaz de generar informes. Para acabar la cuestión, vamos a explorar un par de características. La primera es un resumen de lo que está viendo el monitor (figura 36). La segunda es un análisis de los servicios que se están vigilando (figura 37).

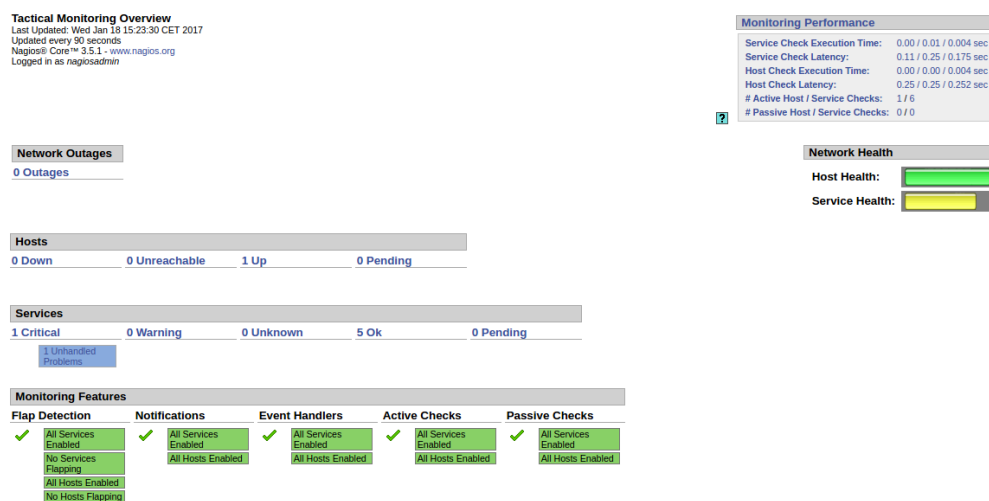


Figura 36: Aquí vemos los datos más relevantes de la monitorización. Todo está correcto, excepto por un error que muestra en la sección de servicios. Por ello, iremos a ver qué ocurre.

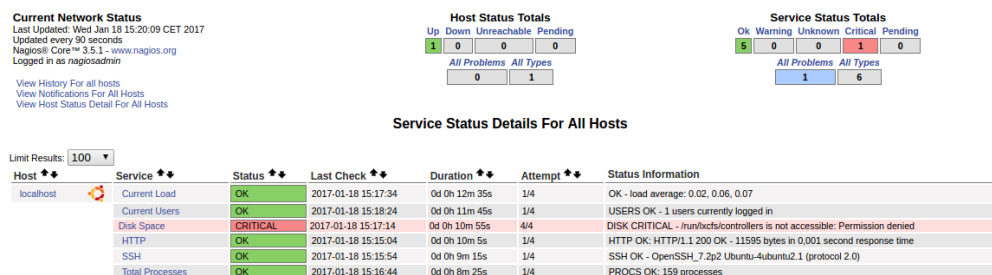


Figura 37: El error detectado en la captura anterior pertenece al sistema de discos. Sabemos que funciona, pero por falta de permisos *Nagios* no es capaz de acceder a los datos que necesita y por ello lanza un error.

Si queremos indagar un poco más en el error, podemos entrar a la sección concreta del sistema de discos (figura 38).

Service Information
 Last Updated: Wed Jan 18 15:36:41 CET 2017
 Updated every 90 seconds
 Nagios® Core™ 3.5.1 - www.nagios.org
 Logged in as [nagiosadmin](#)
[View Information For This Host](#)
[View Status Detail For This Host](#)
[View Alert History For This Service](#)
[View Trends For This Service](#)
[View Alert Histogram For This Service](#)
[View Availability Report For This Service](#)
[View Notifications For This Service](#)

Service
Disk Space
 On Host
localhost
(localhost)
 Member of
No servicegroups.
 127.0.0.1

Service State Information

Current Status:	CRITICAL (for 0d 0h 27m 27s)
Status Information:	DISK CRITICAL - /run/xcfs/controllers is not accessible: Permission denied
Performance Data:	
Current Attempt:	4/4 (HARD state)
Last Check Time:	2017-01-18 15:32:14
Check Type:	ACTIVE
Check Latency / Duration:	0.199 / 0.002 seconds
Next Scheduled Check:	2017-01-18 15:37:14
Last State Change:	2017-01-18 15:09:14
Last Notification:	2017-01-18 15:12:24 (notification 1)
Is This Service Flapping?	NO (5.72% state change)
In Scheduled Downtime?	NO
Last Update:	2017-01-18 15:36:34 (0d 0h 0m 7s ago)

Active Checks:	ENABLED
Passive Checks:	ENABLED
Obsessing:	ENABLED
Notifications:	ENABLED
Event Handler:	ENABLED
Flap Detection:	ENABLED

Figura 38: Página específica para el disco. Obtenemos la misma respuesta: no puede acceder a algunos datos y, debido a eso, lanza un error. Sin embargo, el parámetro de latencia, con un valor realista, parece indicarnos que está activo de todas formas.

3.3. Cuestión opcional 5: Pruebe a instalar este monitor (Cacti) en alguno de sus tres sistemas. Realice capturas de pantalla del proceso de instalación y comente capturas de pantalla del programa en ejecución.

Como se comenta en la web de *Cacti* [7], instalarlo es tan fácil como usar nuestro gestor de paquetes. Durante este proceso, nos preguntará si queremos que la base de datos necesaria sea configurada automáticamente (figura 39), nos pedirá una contraseña para usar el monitor (figura 40) y nos dará a elegir el servidor que configurará para poder funcionar (figura 41). Un poco más de información acerca de qué hacer tras esto se encuentra en [8], aunque la mayoría del contenido lo podemos ignorar.

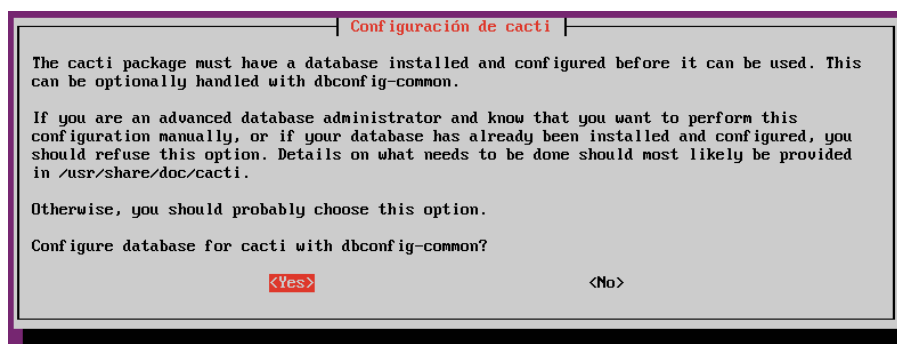


Figura 39: Pantalla que nos pregunta si queremos que la base de datos que utilizará *Cacti* sea configurada automáticamente.

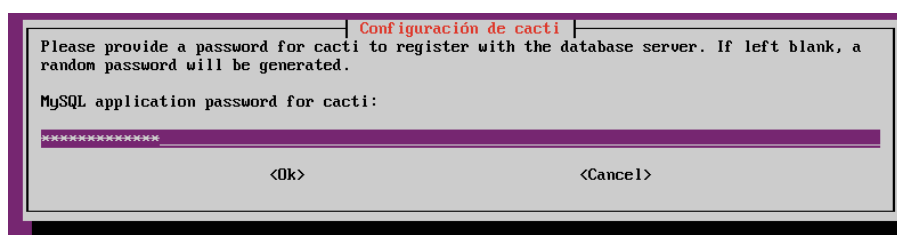


Figura 40: También nos solicita una contraseña para trabajar con *Cacti*.

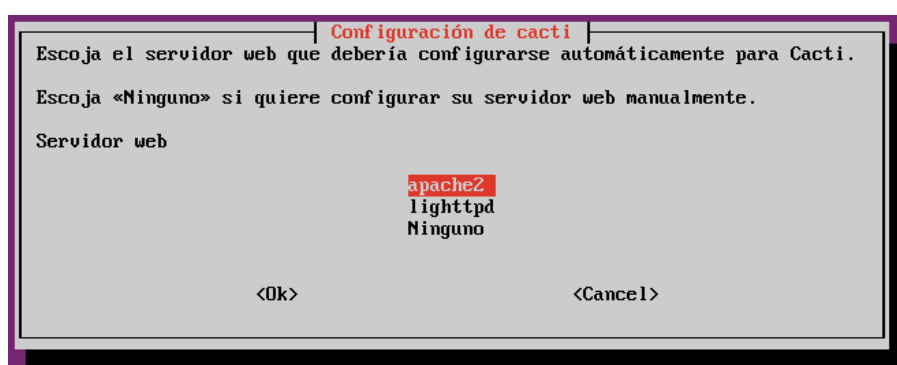


Figura 41: Pantalla que nos da a elegir qué servidor queremos que use el monitor.

A continuación, nos dirigimos a la página principal de *Cacti* (192.168.56.101/cacti en nuestro caso) usando el navegador de la máquina local; esto se muestra en la figura 42:

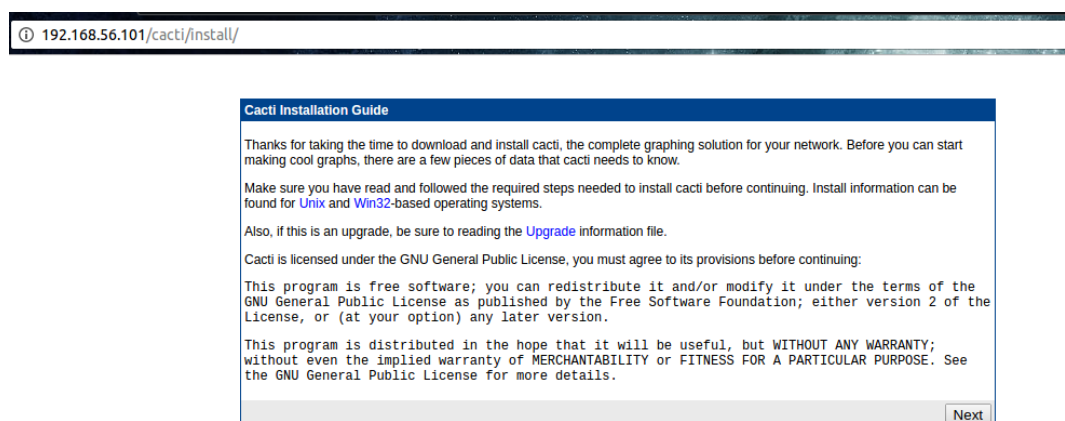


Figura 42: Cuando llegamos a la página principal, se inicia un pequeño proceso de instalación previo a poder usar el monitor.

Nos preguntará si estamos realizando una instalación limpia o una actualización. Elegiremos la primera opción:

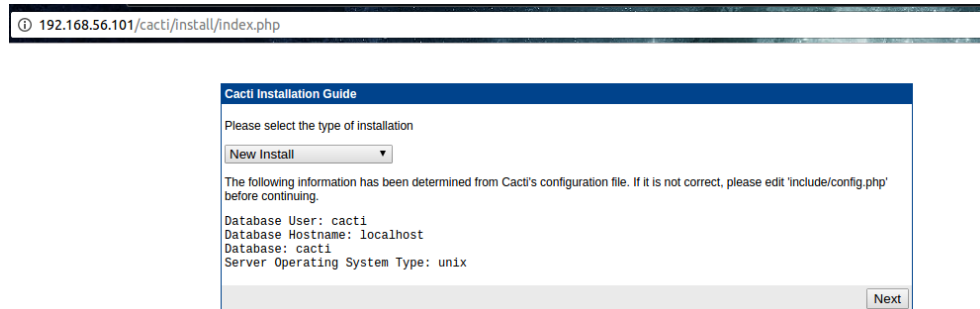


Figura 43: Le decimos que es una nueva instalación.

Como último paso, el instalador buscará ciertos archivos necesarios y nos pedirá que confirmemos que son correctos. Le decimos que sí:

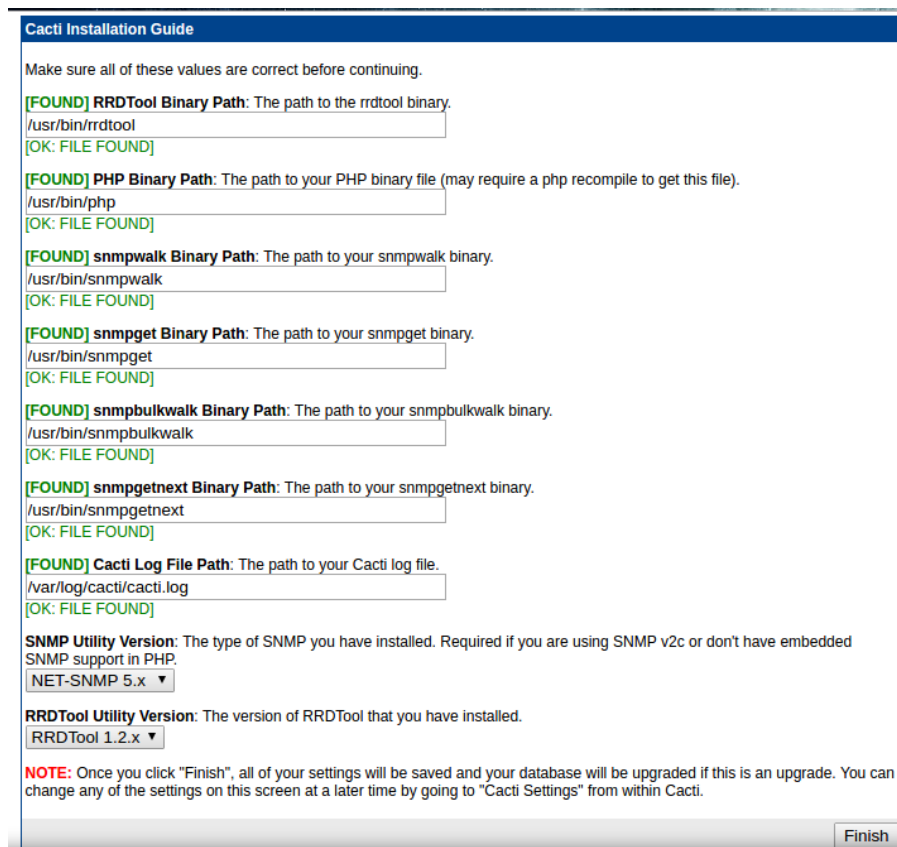



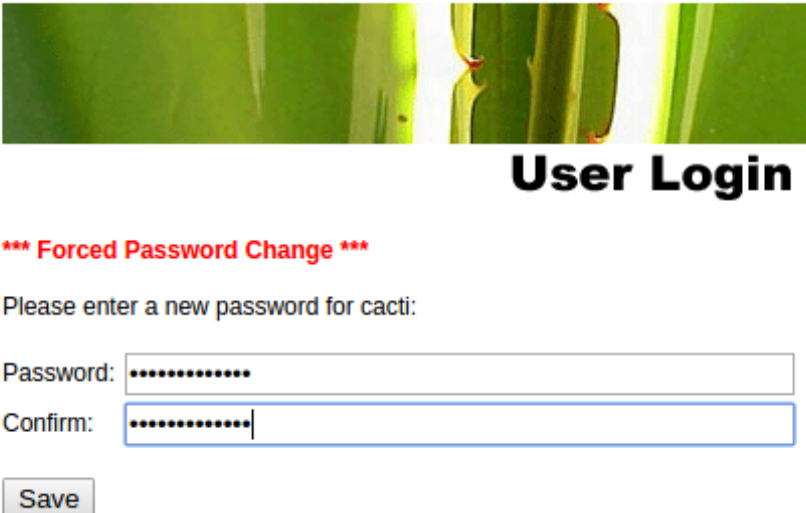
Figura 44: Paso final de la instalación web.

Ya está todo listo para entrar y hacer algunas pruebas. Se nos solicitará un usuario y contraseña (figura 45) e, inmediatamente después, que cambiemos esta última por una de nuestra elección (figura 46).



The image shows the Cacti User Login interface. At the top is a banner image of a green cactus. Below it, the text "User Login" is displayed in a large, bold, black font. Underneath, a message says "Please enter your Cacti user name and password below:". There are two input fields: "User Name:" with the text "admin" entered, and "Password:" with four dots representing a masked password. A "Login" button is located below the password field.

Figura 45: Ventana de login.



The image shows the Cacti Forced Password Change interface. At the top is the same green cactus banner image. Below it, the text "User Login" is displayed in a large, bold, black font. Underneath, the text "*** Forced Password Change ***" is shown in red. A message says "Please enter a new password for cacti:". There are two input fields: "Password:" with ten dots representing a masked password, and "Confirm:" with ten dots representing a masked password. A "Save" button is located below the confirm field.

Figura 46: Ventana de cambio de contraseña la primera vez que entramos a *Cacti*.

La primera pantalla que nos aparece al entrar es la siguiente:

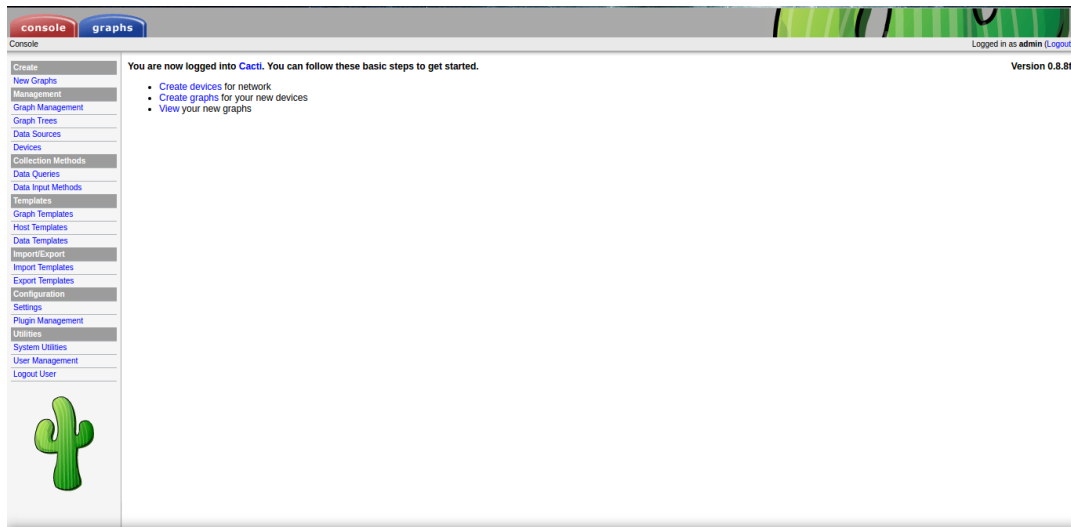


Figura 47: Pantalla principal de *Cacti*.

Podemos crear gráficos para monitorizar nuestro sistema. Para ello, pulsamos en la opción que se nos ofrece con este fin (**Create graphs**) y llegamos a la sección que se ve en la figura 48. En esta demostración, vamos a elegir gráficos que nos muestren el uso de memoria y la carga media del sistema, representados en las figuras 49 y 50 respectivamente.

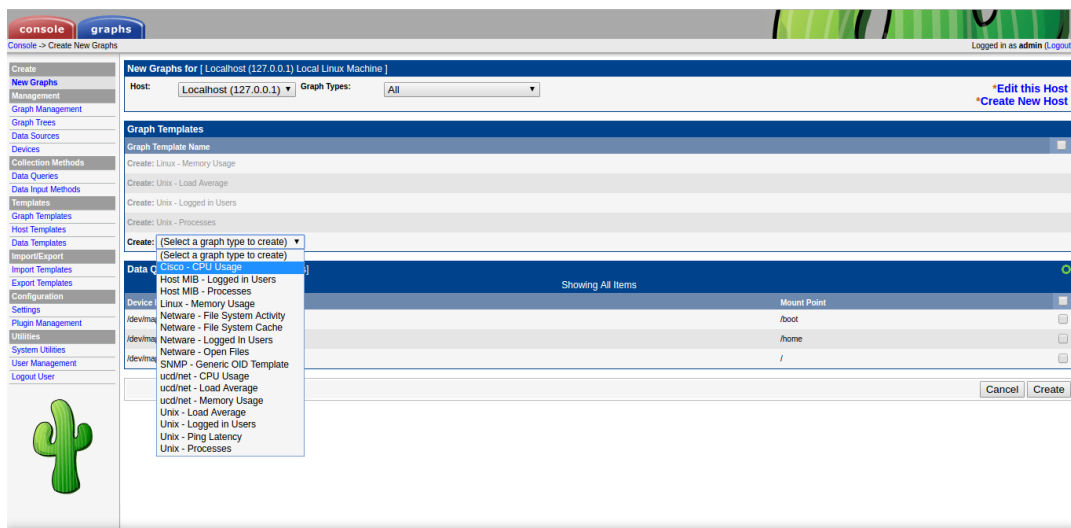


Figura 48: Aquí podemos crear nuevos gráficos a partir de plantillas. Los disponibles inicialmente son los que aparecen en el menú desplegable.

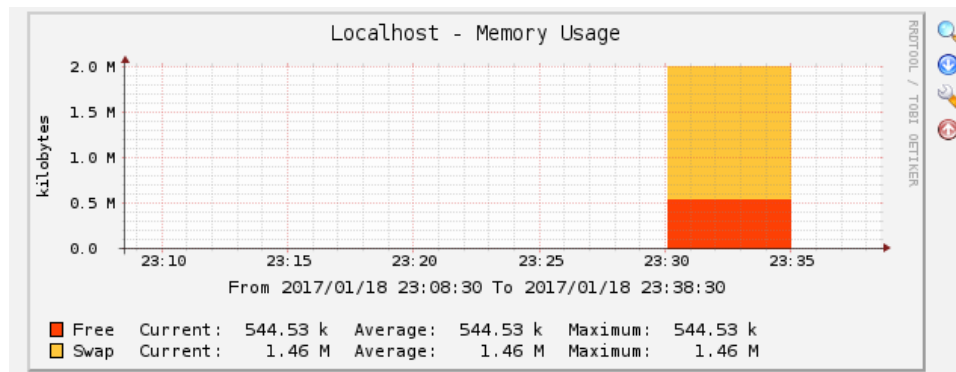


Figura 49: Gráfico de uso de memoria. Debido a que apenas acaba de empezar a monitorizar, no está relleno horizontalmente. Además, ya que el sistema está totalmente en reposo al momento de hacer las capturas, no se consume mucha memoria.

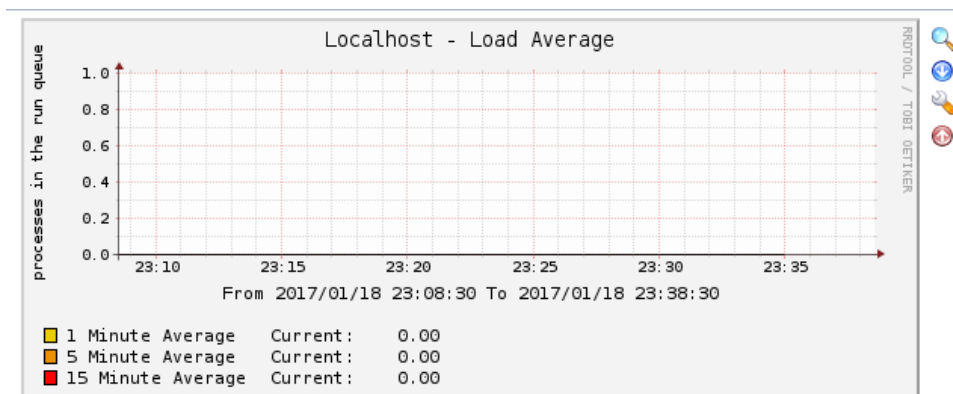


Figura 50: Gráfico de carga media. El sistema no presenta carga significativa porque no se ha estado usando mientras exploraba los menús de *Cacti*.

4. Práctica 4.

4.1. Cuestión opcional 1: ¿Qué es Scala? Instale Gatling y pruebe los escenarios por defecto.

Scala [9] (Scalable Language) es un lenguaje puramente orientado a objetos que incorpora características de los lenguajes funcionales; posee inferencia de tipos y una sintaxis simple, y por ello parece un lenguaje de scripting. Se ejecuta en la máquina virtual de Java (JVM).

Para probar Gatling [10], descargamos el archivo comprimido [11] y lo descomprimos; en esto consiste la instalación. Después, situándonos en su directorio principal, ejecutamos el script `gatling.sh` de la carpeta `bin` (figura 51).

```

jlp@UbuntuServer:~$ cd /home/jlp/gatling/gatling-charts-highcharts-bundle-2.2.3$ bin/gatling.sh
GATLING HOME is set to /home/jlp/gatling/gatling-charts-highcharts-bundle-2.2.3
22:04:15.643 [WARN ] i.g.c.ZincCompiler$ - Pruning sources from previous analysis, due to incompatib
le CompileSetup.
Choose a simulation number:
[0] computerdatabase.BasicSimulation
[1] computerdatabase.advanced.AdvancedSimulationStep01
[2] computerdatabase.advanced.AdvancedSimulationStep02
[3] computerdatabase.advanced.AdvancedSimulationStep03
[4] computerdatabase.advanced.AdvancedSimulationStep04
[5] computerdatabase.advanced.AdvancedSimulationStep05

```

Figura 51: Ejecución de `gatling.sh`, en la que se nos da a elegir entre varias simulaciones tras (a juzgar por el tiempo que tarda) compilarlas todas.

Después de elegir la opción 0, se ejecuta durante un tiempo y muestra el resultado que se observa en la figura 52; como se comprueba, incluye información estadística, como por ejemplo los tiempos mínimos y máximos de respuesta, la media o la desviación estándar. Asimismo, nos indica la ruta donde se ha guardado un informe en formato HTML. Para verlo, añadiré al directorio del servidor web la carpeta entera y abriré el documento principal desde el navegador de mi máquina local (figura 53).

```

Simulation computerdatabase.BasicSimulation completed in 25 seconds
Parsing log file(s)...
Parsing log file(s) done
Generating reports...
=====
----- Global Information -----
> request count                13 (OK=13 KO=0 )
> min response time            122 (OK=122 KO=- )
> max response time            838 (OK=838 KO=- )
> mean response time           210 (OK=210 KO=- )
> std deviation                 186 (OK=186 KO=- )
> response time 50th percentile 146 (OK=146 KO=- )
> response time 75th percentile 148 (OK=148 KO=- )
> response time 95th percentile 493 (OK=493 KO=- )
> response time 99th percentile 769 (OK=769 KO=- )
> mean requests/sec            0.52 (OK=0.52 KO=- )
----- Response Time Distribution -----
> t < 800 ms                   12 ( 92%)
> 800 ms < t < 1200 ms        1 (  8%)
> t > 1200 ms                   0 (  0%)
> failed                        0 (  0%)
=====
Reports generated in 0s.
Please open the following file: /home/jlp/gatling/gatling-charts-highcharts-bundle-2.2.3/results/basicsimulation-1484600775530/index.html
jlp@UbuntuServer:~$ cd /home/jlp/gatling/gatling-charts-highcharts-bundle-2.2.3$ _

```

Figura 52: Resultados finales de la ejecución de *Gatling* mostrados por consola.

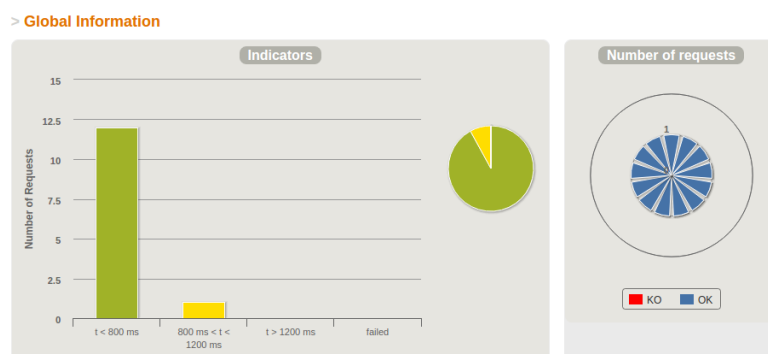


Figura 53: Aquí se ve una de las gráficas generadas para el documento HTML que nos da *Gatling*. Comprobamos cómo, de las 13 solicitudes realizadas, 12 de ellas han tardado menos de 800 milisegundos, mientras que una ha tardado entre 800 y 1200. Esto concuerda con la información de la figura anterior.

5. Práctica 5.

5.1. Cuestión opcional 1: Realice lo mismo que en la cuestión 6 pero para otro servicio.

Para desarrollar esta cuestión he elegido *nginx*. Tras seguir el tutorial de instalación [12], sabemos que está correctamente instalado gracias al contenido de la figura 54.

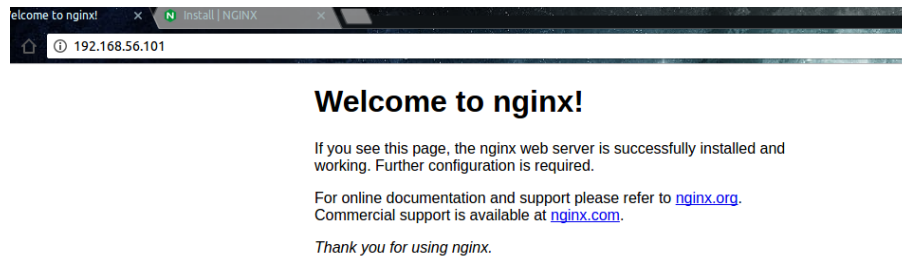


Figura 54: Página inicial de *nginx*, que está ejecutándose en mi máquina virtual de Ubuntu Server (192.168.56.101).

En primer lugar, vamos a ejecutar ab contra el servidor solicitando la misma página que en la cuestión 6 de la práctica 5, con un nivel de concurrencia de 100 (en la práctica hay un error tipográfico y se menciona 200, a diferencia de lo que aparece en las capturas) y un total de 10000 solicitudes:

```
Server Software:      nginx/1.10.2
Server Hostname:      192.168.56.101
Server Port:          80

Document Path:        /index2.html
Document Length:      11321 bytes

Concurrency Level:     100
Time taken for tests:   7.806 seconds
Complete requests:     10000
Failed requests:        0
Total transferred:     115570000 bytes
HTML transferred:      113210000 bytes
Requests per second:    1281.11 [#/sec] (mean)
Time per request:       78.057 [ms] (mean)
Time per request:       0.781 [ms] (mean, across all concurrent requests)
Transfer rate:          14458.76 [Kbytes/sec] received

Connection Times (ms)
  min  mean[+/-sd] median  max
Connect:    0    0  0.2    0    3
Processing:  5   78  8.6   74   91
Waiting:    5   77  8.5   73   91
Total:       6   78  8.5   74   91

Percentage of the requests served within a certain time (ms)
 50%    74
 66%    81
 75%    88
 80%    89
 90%    90
 95%    90
 98%    90
 99%    90
100%    91 (longest request)
javi@javi-X555LJ:~$ date
lun ene 16 17:59:36 CET 2017
javi@javi-X555LJ:~$ Javier León Palomares
```

Figura 55: Ejecución de ab antes de modificar parámetros de *nginx*.

A continuación, es el momento de modificar algunos parámetros en busca de un incremento del rendimiento [13]. Una rápida consulta a `man nginx` nos indica que el archivo principal de configuración es `/etc/nginx/nginx.conf`. En dicho archivo, cambiamos el parámetro `access_log /var/log/nginx/access.log main;` a `access_log off;`. Quizá no sea lo más adecuado deshabilitar los logs, pero tras varias pruebas, una de las cuales se muestra en la figura 56, se observa una ligera mejora de velocidad de $\frac{7,806}{7,509} = 1,04$ veces.

Sin embargo, experimentando con otros parámetros, he encontrado una mejora importante de velocidad al configurar `sendfile off`, en lugar de `on`, que es como se recomienda. Una muestra con ganancia $\frac{7,806}{6,333} = 1,23$ veces se puede encontrar en la figura 57. Es algo extraño, ya que según la referencia utilizada escribe mucho más rápido en el socket y consume menos ciclos de CPU. Además, consultando la página del manual de dicha llamada al sistema (`man sendfile`), vemos al principio cómo nos indica que es más eficiente que combinar `read` y `write`. No he encontrado una explicación para este fenómeno, pero he decidido incluirlo por los destacables resultados.

```

Server Software:      nginx/1.10.2
Server Hostname:      192.168.56.101
Server Port:          80

Document Path:        /index2.html
Document Length:      11321 bytes

Concurrency Level:     100
Time taken for tests:   7.509 seconds
Complete requests:     10000
Failed requests:        0
Total transferred:     115570000 bytes
HTML transferred:     113210000 bytes
Requests per second:   1331.73 [#/sec] (mean)
Time per request:      75.090 [ms] (mean)
Time per request:      0.751 [ms] (mean, across all concurrent requests)
Transfer rate:         15030.05 [Kbytes/sec] received

Connection Times (ms)
      min    mean[+/-sd] median    max
Connect:    0      0  0.1      0      2
Processing:  4     75  7.2     72     96
Waiting:    4     74  7.2     72     96
Total:      6     75  7.2     72     96

Percentage of the requests served within a certain time (ms)
 50%    72
 66%    75
 75%    77
 80%    78
 90%    84
 95%    90
 98%    93
 99%    95
100%    96 (longest request)
javi@javi-X555LJ:~$ date
lun ene 16 19:29:15 CET 2017
javi@javi-X555LJ:~$ Javier León Palomares

```

Figura 56: Ejecución de ab tras modificar el parámetro `access_log`.

```
Server Software:      nginx/1.10.2
Server Hostname:      192.168.56.101
Server Port:          80

Document Path:        /index2.html
Document Length:      11321 bytes

Concurrency Level:    100
Time taken for tests:  6.333 seconds
Complete requests:    10000
Failed requests:       0
Total transferred:    115570000 bytes
HTML transferred:     113210000 bytes
Requests per second:  1579.15 [#/sec] (mean)
Time per request:     63.325 [ms] (mean)
Time per request:     0.633 [ms] (mean, across all concurrent requests)
Transfer rate:        17822.53 [Kbytes/sec] received

Connection Times (ms)
      min    mean[+/-sd] median   max
Connect:    0      0   0.2      0      3
Processing:  5     63   4.9     62     76
Waiting:    5     63   4.9     61     76
Total:      6     63   4.9     62     76

Percentage of the requests served within a certain time (ms)
 50%    62
 66%    63
 75%    64
 80%    66
 90%    69
 95%    72
 98%    73
 99%    76
100%    76 (longest request)
javi@javi-X555LJ:~$ date
lun ene 16 19:50:46 CET 2017
javi@javi-X555LJ:~$ Javier León Palomares
```

Figura 57: Ejecución de ab con el parámetro `sendfile off`.

Referencias

- [1] “Wiki de RAID software de Linux. https://raid.wiki.kernel.org/index.php/Detecting,_querying_and_testing,” consultado el 23 de Octubre de 2016.
- [2] “Wiki de Arch Linux: librería readline. <https://wiki.archlinux.org/index.php/readline>,” consultado el 27 de Octubre de 2016.
- [3] “Cómo proteger SSH con fail2ban. <https://www.digitalocean.com/community/tutorials/how-to-protect-ssh-with-fail2ban-on-ubuntu-14-04>,” consultado el 17 de Enero de 2017.
- [4] “Readme de rkhunter. <http://rkhunter.cvs.sourceforge.net/viewvc/rkhunter/rkhunter/files/README>,” consultado el 17 de Enero de 2017.
- [5] “Cómo instalar un nuevo disco. https://help.ubuntu.com/community/InstallingANewHardDrive#Command_Line_Partitioning,” consultado el 18 de Enero de 2017.
- [6] “Cómo instalar Nagios. <https://help.ubuntu.com/lts/serverguide/nagios.html>,” consultado el 18 de Enero de 2017.
- [7] “Cómo instalar Cacti. http://www.cacti.net/download_cacti.php,” consultado el 18 de Enero de 2017.
- [8] “Cómo instalar Cacti en Ubuntu desde cero. http://docs.cacti.net/manual:088:1_installation.1_install_unix.8_distribution_specific_installation_instructions.1_ubuntu,” consultado el 18 de Enero de 2017.
- [9] “¿Qué es Scala? <https://www.scala-lang.org/what-is-scala.html>,” consultado el 16 de Enero de 2017.
- [10] “Empezando a usar Gatling. <http://gatling.io/docs/2.2.3/quickstart.html>,” consultado el 16 de Enero de 2017.
- [11] “Descargar Gatling. <http://gatling.io/#/resources/download>,” consultado el 16 de Enero de 2017.
- [12] “Tutorial de instalación de nginx. <https://www.nginx.com/resources/wiki/start/topics/tutorials/install/>,” consultado el 16 de Enero de 2017.
- [13] “Consejos para optimizar nginx. <https://www.nginx.com/blog/tuning-nginx/>,” consultado el 16 de Enero de 2017.