Registros de eventos

Peso	3
Tópico Cubierto	108.2 Registros de del sistema
Descripción	Los alumnos deberían ser capaces de configurar el daemon syslog. Este objetivo incluye también la configuración del daemon de logging para enviar la salida de logs a un servidor de logs central. Se cubre el subsistema systemd journal. Además, incluye conocimientos básicos de syslog como sistemas de logging alternativos.
Temas	• Configuración del daemon de syslog. • Comprensión de los servicios estándares, prioridades y acciones. • Configuración de logrotate. • Conocimiento básico de rsyslog y syslog-ng.
Ejemplos	* syslog.conf
	* syslogd
	* klogd
	* /var/log
	* logger
	* logrotate
	* journalctl
	* /etc/systemd/journald.conf
	* /var/log/journal

Peso: Indica el valor de importancia que tiene este tópico en la certificación.

Tópico Cubierto: Indica, según el programa de certificación LPI, qué tópico le corresponde a este tema.

Descripción: Un resumen de lo que se verá.

Temas: Un resumen de los conceptos primordiales que están cubiertos.

Ejemplos: Palabras claves que se deben tener en cuenta.

Introducción

Durante el transcurso de este tópico, veremos temas relacionados con la administración de los logs (registros) del sistema, una función muy importante del equipo que nos permite diagnosticar posibles fallas, prevenirlas o simplemente, ayudarnos a entender que está sucediendo.

Para poder aplicar y trabajar con los logs se analizarán los servicios que estén relacionados con este y su forma de configurarlo; también utilizaremos otras herramientas importantes.

¿Qué son los registros de eventos/logs ?

Antes de explicar, veamos qué nos dice la wikipedia:

```
Un log es un registro oficial de eventos durante un rango de tiempo en
particular.
Para los profesionales en seguridad informática es usado para registrar
datos o información sobre quién, qué, cuándo, dónde y por qué (who, what,
when, where y why) un evento ocurre para un dispositivo en particular o
aplicación.
La mayoría de los logs son almacenados o desplegados en el formato
estándar, el cual es un conjunto de caracteres para dispositivos comunes y
aplicaciones.
De esta forma cada log generado por un dispositivo en particular puede ser
leído y desplegado en otro diferente.
También se le considera como aquel mensaje que genera el programador de un
sistema operativo, alguna aplicación o algún proceso, en virtud del cual
se muestra un evento del sistema.
```

Tener un sistema que nos provea de información acerca de lo que está pasando en nuestro equipo es de fundamental importancia, dado que podemos entender cómo está funcionando nuestro equipo desde diferentes puntos de vista.
Todas nuestras aplicaciones o funciones críticas reportarán cada evento en nuestro equipo; a partir de esto, podemos generar estadísticas para solucionar problemas repetitivos o detectar fallas que nos ayudarán a solucionar diferentes problemas o criterios.

Servicios de registros de eventos/Logs

Cuando hablamos de un servicio, nos referimos a que tendremos un servicio que estará activo y administrando los diferentes eventos que necesitemos reportar; para eso tenemos algunos que son muy conocidos.

Cuando un servicio falla, o algo funciona mal en nuestro equipo (por ejemplo, alguna falla de hardware), esto es reportado. Estos mensajes pueden ser encontrados en el directorio **/var/log**. Por ejemplo, muchos mensajes son reportados en el archivo **/var/log/syslog** o en el **/var/log/messages**. Por otro lado, si un servicio genera muchos mensajes, probablemente estos serán escritos en un archivo separado como lo hace el servidor Web Apache o un servicio de correo.

Todo esto es hecho por un demonio llamado **syslogd**, una utilidad del sistema que provee soporte para el registro de mensajes en sistemas *nix. El demonio **klogd** es el encargado de extraer los mensajes del kernel, para luego enviárselos al **syslogd**. Pero los registros de eventos están reservados únicamente para las aplicaciones del sistema, sin embargo, nosotros también podemos registrar mensajes usando **syslogd** o **rsyslog**, configurando una regla apropiada en el archivo **/etc/syslog.conf** o **/etc/rsyslog.conf**.

El directorio **/var/log** contiene los logs principales:

 # ls -ld /var/log/*

drwxr-x--- 2 root adm 4096 Sep 30 18:02 /var/log/apache2

drwxr-xr-x 2 root root 4096 Dec 4 07:43 /var/log/apt

-rw-r--r-- 1 root root 0 May 27 2014 /var/log/aptitude

-rw-r--r-- 1 root root 1097 May 26 2014 /var/log/aptitude.1.gz

-rw-r--r-- 1 root root 813 Jun 5 2013 /var/log/aptitude.2.gz

-rw-r--r-- 1 root root 492 Apr 19 2013 /var/log/aptitude.3.gz

-rw-r----- 1 root adm 84378 Dec 9 21:53 /var/log/auth.log

-rw-r----- 1 root adm 144929 Dec 7 08:01 /var/log/auth.log.1

-rw-r----- 1 root adm 9712 Nov 30 08:06 /var/log/auth.log.2.gz

-rw-r----- 1 root adm 7331 Nov 23 07:56 /var/log/auth.log.3.gz

-rw-r----- 1 root adm 7015 Nov 16 08:08 /var/log/auth.log.4.gz

-rw-r----- 1 root adm 31 Apr 18 2013 /var/log/boot

drwxr-xr-x 2 root root 4096 Dec 9 08:06 /var/log/cups

-rw-r----- 1 root adm 9503 Dec 9 21:46 /var/log/daemon.log

(...salida cortada...)

Sysklogd

Este servicio actualmente ya casi no se usa más. Lo podemos encontrar en versiones anteriores a Red Hat 6, CentOS 6. En el caso de Debian, se encontrará en versiones anteriores a Lenny.

Cuando instalamos el paquete correspondiente, contendrá dos utilitarios (**syslogd** y **klogd**) que proveerán soporte para el registro de eventos del sistema.

Syslogd y klogd corren como demonios (procesos en segundo plano) y envían los mensajes del sistema a diferentes lugares (logs de distintos servicios, correo, seguridad, errores, autentificación,etc)

El demonio **syslogd** (Syslog Daemon) se lanza automáticamente al arrancar un sistema Unix, siendo el encargado de guardar informes sobre el funcionamiento de la máquina. Recibe mensajes de las diferentes partes del sistema (núcleo, programas…) y los envía y/o almacena en diferentes localizaciones, tanto locales como remotas, siguiendo un criterio definido en el fichero de configuración **/etc/syslog.conf**, donde especificamos las reglas a seguir para gestionar el almacenamiento de mensajes del sistema.

Archivos de Configuración

En el archivo de configuración se pueden definir las reglas para especificar dónde se redireccionan los eventos.

Archivo de Configuración (**/etc/syslog.conf**)

Las líneas de este archivo que comienzan por # son comentarios, por lo que son ignoradas de la misma forma que las líneas en blanco; si ocurriera un error al interpretar una de las líneas del archivo, se ignora la línea completa. Un ejemplo de fichero **/etc/syslog.conf** es el siguiente:

```
[root@CentOS-5 ~]# cat /etc/syslog.conf
# Log all kernel messages to the console.
# Logging much else clutters up the screen.
#kern.*                                                 /dev/console

# Log anything (except mail) of level info or higher.
# Don't log private authentication messages!
*.info;mail.none;news.none;authpriv.none;cron.none
        /var/log/messages

# The authpriv file has restricted access.
authpriv.*                                              /var/log/secure

# Log all the mail messages in one place.
mail.*                                                  -/var/log/maillog


# Log cron stuff
cron.*                                                  /var/log/cron

# Everybody gets emergency messages
*.emerg                                                 *

# Save news errors of level crit and higher in a special file.
uucp,news.crit                                          /var/log/spooler

# Save boot messages also to boot.log
local7.*                                                /var/log/boot.log

#
# INN
#
news.=crit                                              /var/log/news/news.crit
news.=err                                               /var/log/news/news.err
news.notice
/var/log/news/news.notice
```

Cada regla del archivo tiene dos campos: un campo de **selección** y un campo de **acción**, separados ambos por espacios o tabuladores.

El campo de **selección** está compuesto, a su vez, de dos partes separadas por un signo punto "**.**" una indica el **servicio** (facility) que envía el mensaje y otra que marca su **prioridad** o nivel (level), ambas son indiferentes a mayúsculas y minúsculas.

La parte del **servicio** contiene una de las siguientes palabras clave: **auth**, **auth-priv**, **cron**, **daemon**, **kern**, **lpr**, **mail**, **mark**, **news**, **security** (equivalente a auth), **syslog**, **user**, **uucp** y **local0** hasta **local7**; esta parte define el subsistema que ha generado ese mensaje (por ejemplo, todos los programas relacionados con el correo generarán mensajes ligados al servicio mail).

En segundo lugar, la **prioridad** está compuesta de uno de los siguientes términos, en orden ascendente: **debug**, **info**, **notice**, **warn** (equivalente a **warning**), **err** (equivalente a **error**), **crit**, **alert**, **emerg**, y **panic** (equivalente a **emerg**).

La prioridad define la gravedad o importancia del mensaje almacenado. Todos los mensajes de la prioridad especificada y superiores son almacenados de acuerdo con la acción requerida. Si a un servicio le defino la prioridad **crit**, solamente almacenará mensajes de esa prioridad o una superior.
Cuando un programa quiere enviar un mensaje al demonio **syslogd** utiliza la llamada **syslog(3)**; al hacerlo, se ha de indicar tanto el **servicio** como la **prioridad** del mismo. Evidentemente, esto es válido para el código en C: si queremos enviar registros al demonio para que sean procesados desde un shellscript podemos utilizar la orden **logger**.

Chequeando servicio

Al reiniciar el servicio **syslog** automáticamente reiniciará el servicio **klog**

```
[root@CentOS-5 init.d]# service syslog restart
Desactivando el generador de logs del kernel:              [  OK  ]
Desactivando el generador de logs del sistema:             [  OK  ]
Iniciando logger del sistema:                              [  OK  ]
Iniciando el generador de logs del kernel:                 [  OK  ]
[root@CentOS-5 init.d]# service syslog status
Se está ejecutando syslogd (pid  2933)...
Se está ejecutando klogd (pid  2936)...
[root@CentOS-5 init.d]#
```

Syslog-ng

El servicio syslog-ng fue creado en el año 1998.

Algunas funcionalidades que tiene son:

- Trabaja con cualquier clase de datos que no está organizado de una manera predeterminada.
- Recibe y envía mensajes formateados en el lenguaje **JSON**.
- Clasifica y organiza los mensajes con analizadores de sintaxis incorporados.

Los mensajes de log ingresan por diferentes fuentes y van a diferentes destinos y el archivo de configuración principal es **syslog-ng.conf**.

Las últimas versiones de las principales distribuciones de Linux usa otra implementación predeterminada de syslog.

Rsyslog

Este proyecto empezó en el 2004 cuando su autor principal Rainer Gerhards decidió escribir un potente sistema de registro de eventos para que pueda competir con syslog-ng. Este fue usado por Fedora a partir del 2007, por SUSE a partir del 2009 y Debian a partir de su versión 5, entre otras distribuciones que decidieron usarlo.

Administrándolo:

```
[root@CentOS-6 ~]# service rsyslog restart
Shutting down system logger:                               [  OK  ]
Starting system logger:                                    [  OK  ]
[root@CentOS-6 ~]# service rsyslog status
rsyslogd (pid  1954) is running...
[root@CentOS-6 ~]#
```

Archivos de Configuración

La configuración de este servicio que se encuentra en **/etc/rsyslog.conf**, que difiere un poco de **syslog.conf** porque tiene categorías.

Categoría:

- **Módulos:** Es modular e indica los módulos que se pueden cargar o descargar
- **Directivas Globales:** Especifica todas las directivas que podemos expresar de forma global (empiezan con $)
- **Reglas:** Especifica las reglas que aplican con su acción determinada.
- **Selección:** Indica a qué categoría y tipo de prioridad (Ej: mail.debug)
- **Acción:** Indica qué hacer con el mensaje luego de pasar por la selección.

Contenido del archivo de configuración resumido

 # cat /etc/rsyslog.conf

```
#################

#### MODULES ####

#################

$ModLoad imuxsock # provides support for local system logging

$ModLoad imklog   # provides kernel logging support

$ModLoad imudp

$UDPServerRun 514

############################

#### GLOBAL DIRECTIVES ####

############################
```

```
$ActionFileDefaultTemplate RSYSLOG_TraditionalFileFormat

$FileOwner root

$FileGroup adm

$FileCreateMode 0640

$DirCreateMode 0755

$Umask 0022

$WorkDirectory /var/spool/rsyslog

$IncludeConfig /etc/rsyslog.d/*.conf


################

#### RULES ####

################

auth,authpriv.*                 /var/log/auth.log

*.*;auth,authpriv.none          -/var/log/syslog

cron.*                          /var/log/cron.log
```

```
daemon.*                        -/var/log/daemon.log

kern.*                          -/var/log/kern.log

lpr.*                           -/var/log/lpr.log

user.*                          -/var/log/user.log

mail.info                       -/var/log/mail.info


*.=debug;\

        auth,authpriv.none;\

        news.none;mail.none     -/var/log/debug

*.=info;*.=notice;*.=warn;\

        auth,authpriv.none;\

        cron,daemon.none;\

        mail,news.none          -/var/log/messages

*.emerg                         *

daemon.*;mail.*;\
```

news.err;\

.=debug;.=info;\

.=notice;.=warn |/dev/xconsole

Nivel de Prioridades y Tipo de Mensajes

En esta sección veremos los tipos de mensajes y prioridades.

¿Para qué nos sirven?

El sistema divide en categorías a los diferentes eventos, y a su vez, en prioridades para poder saber la criticidad del evento; para esto, existen diferentes categorías de eventos y tipos de mensajes. Este tipo de combinación se llama “reglas”, donde uno define cómo va a ser el comportamiento de cada evento para tomar un acción en particular para cada determinado caso.

La siguiente tabla define cada uno de ellos; luego veremos cómo armar las reglas:

Servicio Descripción

auth Mensajes de seguridad/autenticación
authpriv Mensajes de seguridad/autenticación (privado)
cron Demonio de tiempo (cron y at)
daemon Demonios del sistema sin valor de servicio separado
kern Mensajes del kernel
lpr Mensajes del servicio de impresión
mail Mensajes del servicio de correo
mark Para uso interno. No usar al hacer las reglas
news Mensajes del servicio de noticias USENET
security Obsoleto, usar auth
syslog Mensajes generados internamente por syslogd user Mensajes genéricos a nivel de usuario
uucp mensajes de UUCP
local0 a **local7** Reservado para definir por el usuario

La prioridad puede ser una de las palabras listadas en la siguiente tabla. Los mensajes serán reportados por prioridad, de forma ascendente. Por ejemplo, si se especifica la prioridad **alert**, se reportarán los mensajes con prioridad **alert**, **emer** y **panic**, mientras que los **crit**, **error**, hasta **debug** no serán reportados.

Prioridad Descripción

debug Usado para depurar servicios, por ejemplo si no están funcionando apropiadamente
info Usado para reportar mensajes informativos.
notice Como la prioridad info, pero haciendo notar algo que puede ser relevante
warning Usado para reportar advertencias. Puede darte pistas sobre errores (si los hubiera) o solo mostrarte que hay algo que no está trabajando como debería, pero que igual sigue funcionando.
warn Igual que warning
err Usado para reportar errores. Por ejemplo, si tienes un servicio mal configurado este reportará esos errores.
error Igual que err
crit Usado para reportar errores más críticos. Por ejemplo errores de hardware.
alert Usado para reportar errores aún más críticos. Se debe tomar algún correctivo inmediatamente. Por ejemplo, corrupción de una base de datos.
emerg Usado para reportar errores realmente críticos. Muy probablemente el servicio está inoperante
panic Igual que emerg
none Usado para deshabilitar el reporte de un servicio.

Configuración de Reglas

Cada regla sigue esta sintaxis:

selector acción

A su vez, el selector está compuesto de servicio.prioridad, entonces la sintaxis completa sería:

servicio.prioridad acción

La acción describe qué se debe hacer con el mensaje reportado. Comúnmente, todos los mensajes son escritos a un archivo de registros de eventos, pero también hay otras acciones, como reenviar los mensajes a otra máquina. De forma que el campo acción puede ser uno de los siguientes:

Acción	Descripción
/ruta/de/registro de evento	Escribir los mensajes a un archivo de registros de eventos (ej: /var/log/archivo.log)
| fifo	Usar un fifo o una tubería como el destino de los mensajes. Esto es útil para depuración o enviar correos. Note que el fifo debe ser creado con el comando mkfifo(1) antes de que syslogd(8) sea iniciado
/dev/tty[1-6]	Escribir mensajes en las consolas /dev/tty[1-6]. Note que /dev/console también funcionará
@192.168.0.1	Reenviar mensajes a la máquina 192.168.0.1 vía UDP. Debido a la naturaleza de UDP, probablemente se perderán mensajes en tránsito. Si esperas alto volumen de tráfico, debes esperar una pérdida considerable de mensajes. Nota: para aceptar mensajes, el servidor remoto debe correr syslogd con la opción -r (en Debian esta opción puede ser dada en el archivo /etc/default/syslogd o en el /etc/default/rsyslog)
@@**192.168.0.1**	Reenviar mensajes a la máquina 192.168.0.1 vía TCP, evita perdida de paquetes. Hay que cargar el módulo **imtcp** para que funcione
:omrelp:192.168.0.1:2514	Si quieres prevenir la pérdida de mensajes UDP, usa RELP
lgallard, atorres	Lista de usuarios. Por defecto, los mensajes críticos son enviados a root

Modificadores

Básicamente, existen tres modificadores: **=**, **!** y *****. El modificador "**=**" le indica a syslogd que debe reportar solo los mensajes con la prioridad exacta. Por ejemplo:

mail.=error	**/var/log/mail.error**

Aquí syslogd reportará sólo los mensajes de **error**. Sin el modificador =, syslogd reportaría los mensajes tipo **error**, **crit**, **alert** y **panic**. Este modificador sólo puede usarse con las prioridades.

El segundo modificador es "**!**", que invierte el significado de la regla. Por ejemplo:

mail.!error /var/log/mail.error

Syslogd reportará los mensajes con menor prioridad que **error**, o sea, **warning**, **notice**, **info** y **debug**. Si queremos excluir sólo una prioridad, debemos usar la combinación **!=**
Ejemplo:

mail!=error /var/log/mail.error

Finalmente, el modificador "*****" permite seleccionar entre los distintos servicios y prioridades. Por ejemplo:

mail.* /var/log/mail.log

Aquí, todos los mensajes provenientes del servicio de correo serán guardados en el archivo **/var/log/mail.log**, no importando su prioridad. Otro ejemplo:

***.info /var/log/info.log**

No importa el servicio, todos los mensajes cuya prioridad sean **info** serán guardados en el archivo **/var/log/info.log**. Operadores coma "**,**" y punto y coma "**;**"

El operador punto y coma permite escribir varias reglas en una forma más compacta. Por ejemplo:

mail.=info /var/log/info.log

mail.=notice /var/log/info.log

auth.=info /var/log/info.log

Las reglas anteriores pueden ser escrita en una sola línea:

mail.=info;mail.=notice;auth.=info /var/log/info.log

Por otro lado, si queremos seleccionar varios servicios, podemos usar el operador coma. Por ejemplo:

mail.=info /var/log/info.log

auth.=info /var/log/info.log

Podemos escribir las reglas anteriores en una línea, de la siguiente forma:

mail,auth.=info /var/log/info.log

La gran diferencia entre el operador coma y el operador punto y coma, es que el primero solo separa servicios, mientras que el último puede separar prioridades y servicios, incluso si estos son incompatibles entre sí.

Registro síncrono

Algunos registros de eventos deben ser monitoreadas en tiempo real, por ejemplo cuando se está depurando un servicio. El asunto es que syslogd escribe mensajes solo cuando su buffer está lleno, es decir, de forma asincrónica. Si queremos escribir mensajes sincrónicamente debemos colocar un "**-**" antes de la ruta del archivo donde se guardarán los registros de eventos.

Ejemplos

mail.info -/var/log/mail.log

Todos los mensajes serán guardados en el archivo /var/log/mail.log de forma sincrónica.

Resumen:

```
auth.warn    muestra mensajes de auth, con prioridad warn o superior.\\
auth.=warn  muestra mensajes de auth, con prioridad warn unicamente.\\
auth.!warn   muestra mensajes de auth, con prioridad menor a warn.\\
auth.!=warn muestra mensajes de auth, con cualquier nivel de prioridad
excepto warn.\\

Separación:
Con el ; (punto y coma), podemos especificar distintos mensajes
que apunten a un mismo lugar:

auth.*;cron.warn          /var/log/varios.log

En este caso estaríamos enviando a el fichero varios.log, la
salida de los mensajes de autenticación para todos los niveles de
prioridad, en conjunto con los mensajes de cron el nivel de prioridad
warn o superior.\\

Y con la , (coma) podemos separar distintos tipos de mensajes, con los
mismos niveles de prioridad; por ejemplo:

auth,authpriv.*         /var/log/auth.log
```

Uso de logger

Con este comando podremos hacer llamadas a syslog(3) para que el sistema pueda escribir un log donde se le indique:

Opciones

-p prioridad (puede utilizarse servicio.prioridad)

-t marca, agrega un texto para identificar el mensaje

 # logger -p mail.info "Mensaje de prueba"

 # tail /var/log/mail.log

Dec 9 22:43:35 debian evillarreal: Mensaje de prueba

 # logger -t "marca de prueba" -p mail.info "Mensaje de prueba"

 # tail /var/log/mail.log

Dec 9 22:46:47 buegsevi marca de prueba: Mensaje de prueba

Uso de lastlog, last y faillog

Con estos comandos podremos obtener información acerca de los intentos fallidos de logueo de los usuarios del sistema y de las veces que se pudieron conectar.

Lastlog

Muestra la última vez que se logueó cada usuario

Opciones

-u define el usuario a consultar

-t días muestra solo los registros que no sean más antiguos que la cantidad de días definida

-b días muestra solo los registros que sean más antiguos que la cantidad de días definida

 # lastlog -u root

Username Port From Latest

root tty2 Fri Sep 19 15:20:39 -0300 2014

Faillog

Muestra los logueos fallidos de los usuarios.

Opciones

-a muestra todos los eventos

-l bloquea por un tiempo determinado el login luego de fallar

-u muestra información del usuario definido

```
**[root@CentOS-5 init.d]# faillog -u root**
Usuario      Fallos Máximo Último                    Activo

root             0        0   12/31/69 21:00:00 -0300
```

Este comando no existe en algunas distribuciones, tales como CentOS 6 y CentOS 7. En su lugar se usa un módulo de pam que se puede usar para bloquear logins fallidos luego de una determinada cantidad de intentos llamado **pam_faillock**.

last

El comando **last** muestra las últimas veces que un usuario ingresó en el sistema pero también las veces que se apago o reinicio el equipo. Indica también la versión de kernel con la que se inició lo cual es muy importante.

Opciones

-f archivo Lee otro archivo en lugar de usar /var/log/wtmp

-numero cantidad de líneas a mostrar

-F más datos acercas de login y logout

-w más datos acerca del usuario y dominios

 # last

user1 pts/12 10.1.5.23 Tue Dec 9 22:51 still logged in

user1 pts/11 alm-s01-v01 Tue Dec 9 22:50 - 23:00 (00:09)

user1 pts/6 10.1.5.23 Tue Dec 9 21:52 still logged in

user1 pts/6 10.1.5.23 Mon Dec 8 13:03 - 07:54 (18:51)

user1 pts/6 10.1.5.23 Sat Dec 6 09:49 - 09:11 (23:22)

(...salida cortada...)

Registro de eventos/Logs Remoto

Si quisiéramos centralizar los logs en único equipo, para evitarnos que puedan borrar los logs locales, dado que siempre van a replicar en otro equipo, tendríamos que configurar ciertos archivos, dependiendo de la versión de syslog o rsyslog.

Hay que modificar el archivo de configuración:

```
[root@CentOS-5 ~]# cat /etc/sysconfig/syslog
SYSLOGD_OPTIONS="-m 0"
KLOGD_OPTIONS="-x"
SYSLOG_UMASK=077
```

Cambiar de esta manera **SYSLOGD_OPTIONS=”-m 0 -r”**

El siguiente paso es agregar la configuración en la máquina que va a ser la encargada de enviar los logs. Editar el archivo **/etc/rsyslog.conf**
En este ejemplo se enviarán todos los servicios sin importar la prioridad a la IP 192.168.0.1 vía UDP. Si fuera una red remota lo conveniente es realizarlo con TCP.

. @192.168.1.10

En este ejemplo se visualiza como se recibe la información

```
Dec 17 03:57:46 192.168.1.100 CentOS-6 sshd[2070]: Connection closed by
192.168.1.101
Dec 17 03:57:49 192.168.1.100 CentOS-6 sshd[2072]: Accepted password for
root from 192.168.1.101 port 33702 ssh2
Dec 17 03:57:50 192.168.1.100 CentOS-6 sshd[2072]: pam_unix(sshd:session):
session opened for user root by (uid=0)
[root@CentOS-5 log]
```

En cambio, si quisiéramos hacer lo mismo para rsyslog:

Primero, deberíamos descomentar dos líneas en la sección de módulos para habilitar el módulo de UDP y reiniciar el servicio.

$ModLoad imudp

$UDPServerRun 514

Rotación de Logs

Erik Troan, y Preston Brown son los autores de **logrotate**, una utilidad para administrar las políticas de los logs de tu equipo.

Logrotate es un estándar en sistemas RedHat y Debian. Con esta herramienta, podremos especificar todo tipo de parámetros a la hora de administrar nuestros logs.

Un archivo de configuración de logrotate, consiste en una serie de especificaciones para los grupos de archivos de log que vamos a administrar. Las opciones especificadas fuera de cada contexto de un log concreto, (errors, rotate, weekly…) se aplican a todos ellos, pero pueden ser reemplazadas con una especificación concreta para un log en particular.

En nuestro sistema la utilización de logs es algo imprescindible y es por eso que estos crecen constantemente y hay que tener alguna utilidad para especificarle el comportamiento. El directorio de configuración global se encuentra en **/etc/logrotate.conf**; sino, también, tenemos otro directorio en **/etc/logrotate.d/** ,dónde podremos poner individualmente cada configuración. Para que cada una de las configuraciones tenga efecto, se programa una entrada en el crontab del sistema, para que corran cada determinado tiempo (**/etc/crond.daily/logrotate**).

Opciones:

-d se utiliza para debug no hace nada, simula la rotación

-f fuerza la rotación

-v nos da más información.

Ejemplo:

 # logrotate -d /etc/logrotate.conf

reading config file /etc/logrotate.conf

including /etc/logrotate.d

reading config file apache2

reading config file apt

reading config file aptitude

reading config file cups

reading config file dpkg

reading config file mysql-server

reading config file pm-utils

reading config file rsyslog

reading config file samba

reading config file vsftpd

Handling 29 logs

rotating pattern: /var/log/apache2/*.log weekly (52 rotations)

empty log files are not rotated, old logs are removed

considering log /var/log/apache2/access.log

log does not need rotating

considering log /var/log/apache2/error.log

log does not need rotating

not running prerotate script, since no logs will be rotated

not running postrotate script, since no logs were rotated

rotating pattern: /var/log/apt/term.log monthly (12 rotations)

empty log files are not rotated, old logs are removed

considering log /var/log/apt/term.log

log does not need rotating

El directorio **/etc/logrotate.d** es un lugar estándar para los archivos de configuración de **logrotate**.

Todos los paquetes software conscientes de logrotate (la gran mayoría) se integran con este sistema de administración de logs en la parte de su proceso de instalación, lo que simplifica ampliamente la administración.

Ejemplo:

 # ls /etc/logrotate.d

apache2 cups lighttpd ppp

apt dirmngr monit vsftpd

aptitude dpkg mysql-server rsyslog

Configuración de Ejemplo :

 # cat /etc/logrotate.d/vsftpd

/var/log/vsftpd.log

{

create 640 root adm

missingok

notifempty

rotate 4

weekly

}

Opciones:

missingok No se producirá ningún error si el archivo de log no existe.

notifempty No rotar el log si este está vacío.

sharedscripts Los script de postrotate solo se ejecutarán una vez que los logs viejos fueron comprimidos.

delaycompress Sirve por si algún programa está escribiendo y necesita al archivo este no se comprime.

postrotate/endscript Lo que esté dentro de estas directivas, se ejecutará luego de la rotación de archivos.

compress Comprime los archivos rotados

daily Rotar diariamente

weekly Rotar semanalmente

monthly Rotar mensualmente

yearly Rotar anualmente

Lo que no esté definido se tomará del archivo de configuración global **/etc/logrotate.conf**

Systemd-Journald

Es un servicio que recoge y almacena la información de registro de eventos. Crea y mantiene una estructura indexada de la información recibida del kernel, los procesos de usuario y servicios del sistema. Los datos serán guardados de manera segura, de manera que no puedan ser falsificados.

Los datos se almacenan de manera binaria en el directorio **/var/log/journal**

Configuración

La configuración principal se encuentra en **/etc/systemd/journald.conf**

Opción	Descripción
Storage=opción	Controla donde se guardará la información. Las opciones posibles son: volatile, persistent, auto y none. volatile: guarda la información la memoria RAM en el directorio /run/log/journal. Se pierde al reiniciar persistent: guardará la información en **/var/log/journal** auto: Es la opción predeterminada. Si no existe /var/log/journal, guardará los registros en RAM. none: desactiva el almacenamiento de registros.
Compress=	Comprimir o no la información almacenada. De manera predeterminada el valor es **yes.**
Seal=	Utilizar o no la protección de los registros por modificaciones no autorizadas. Valor predeterminado **yes**
SplitMode=	Como se dividirán los registros. uid=por userID. login=usuarios con acceso al sistema. Los usuarios sin acceso (usuarios de sistema) utilizarán los registros de sistema. none=utilizará un único archivo para almacenar los registros.
RateLimitInterval= RateLimitBurst=	Aplica un límite a los mensajes generados en el sistema. Si en el intervalo definido en RateLimitInterval supera la cantidad de mensajes en RateLimitBurst los descartará. Valor predeterminado 1000 mensajes cada 30 segundos.
MaxFileSec=	Tiempo máximo para rotar los registros. Predeterminado un mes.
MaxRetentionSec=	Período máximo de retención de registros.
ForwardToSyslog=	Reenviar o no los mensajes recibidos al servicio de syslog

Comando journalctl

El comando **journalctl** se utiliza para consultar los registros escritos por el servicio systemd-journald.service

Opciones

-f Muestra de manera continua los nuevos registros (similar a “tail -f “)

-n [número] Muestra el número de líneas definido

-o [salida] Muestra el formato en la salida definida (short, verbose, json)

-k Muestra mensajes del kernel (igual comando dmesg)

-p [prioridad] Muestra los mensajes de la prioridad definida (debug, info, notice, warning, err, crit, alert, emerg)

_PID [número] Muestra los mensajes del PID definido

_UID [número] Muestra los mensajes del usuario definido

Ejemplos

Muestra los registros del **PID 1** con la prioridad **err**

 # journalctl -p err _PID=1

Muestra los últimos 10 registros del servicio gdm

 # journalctl -n 10 _SYSTEMD_UNIT=gdm.service

Muestra de manera continua los registros del usuario apache

 # journalctl -f -u apache

Snoopy

Esta herramienta sirve para registrar todos los comandos ejecutados en los logs. Lo hace mediante la carga previa anterior a la ejecución de cualquier programa de la librería **libsnoopy.so**.

Luego de instalado se debe hacer lo siguiente:

 # snoopy-enable

SNOOPY: Removing from /etc/ld.so.preload: /lib/snoopy.so

SNOOPY: Adding to /etc/ld.so.preload: /lib/libsnoopy.so

SNOOPY: Hint #1: Reboot your machine to load snoopy system-wide.

SNOOPY: Hint #2: Check your log files for output.

SNOOPY: Enabled.

Una vez reiniciado el sistema, podremos ver con **journalctl -f**:

may 10 15:58:40 debian.educacionit.local snoopy[1160]: [uid:0 sid:1144 tty:/dev/pts/0 cwd:/root filename:/bin/ls]: ls

may 10 15:58:43 debian.educacionit.local snoopy[1161]: [uid:0 sid:1144 tty:/dev/pts/0 cwd:/root filename:/bin/journalctl]: journalctl -f

may 10 15:58:49 debian.educacionit.local snoopy[1162]: [uid:0 sid:1144 tty:/dev/pts/0 cwd:/root filename:/usr/bin/clear]: clear

may 10 15:59:21 debian.educacionit.local snoopy[1190]: [uid:0 sid:1144 tty:/dev/pts/0 cwd:/root filename:/usr/bin/vi]: vi .bashrc

may 10 15:59:27 debian.educacionit.local snoopy[1191]: [uid:0 sid:1144 tty:/dev/pts/0 cwd:/root filename:/bin/journalctl]:

En lugar de habilitarlo para todo el sistema, se puede activar para una sesión de shell determinada, por ejemplo:

 # chmod 755 /lib/libsnoopy.so.0.0.0 LD_PRELOAD = /lib/libsnoopy.so.0.0.0 bash

Para deshabilitarlo, sencillamente salimos de bash

 # bash

Si en cambio la habilitamos globalmente y queremos revertirlo:

 # snoopy-disable

SNOOPY: Removing from /etc/ld.so.preload: /lib/libsnoopy.so

SNOOPY: Disabled.

SNOOPY: Hint: Your system needs to be restarted to finish snoopy cleanup.

Luego solamente resta reiniciar.

Bibliografía

Libros:

CompTIA Linux+ / LPIC-1 Cert Guide: (Exams LX0-103 & LX0-104/101-400 & 102-400)

CompTIA Linux+ /LPIC-1 Certification All-In-One Exam Guide, Premium Second Edition With Online Practice Labs (Exams LX0-103 & LX0-104/101-400 & 102-400)

Páginas:

[1] Wikipedia Log

[2] Linux Syslog Server and Log Management

[3] Montar un Servidor syslog en Linux con rsyslog y LogAnalyzer

[4] ¿Qué es journalctl con systemd en Linux y cómo funciona?

[5] Stunnel y cómo cifrar los mensajes de log de GNU/Linux