# Práctica de laboratorio: Leer archivos de registro de un servidor

## Objetivos

**Parte 1: Leer archivos de registro (log files) con Cat, More, Less, y Tail**

**Parte 2: Archivos de registro y Syslog**

**Parte 3: Archivos de registro y Journalctl**

**NOTA: RESPONDER CADA PREGUNTA CON CAPTURAS DE PANTALLA COMPLETAS DONDE SE VEAN LAS SIGLAS DE SU NOMBRE AL LADO DEL RELOJ. EN LUGAR DE COLOCAR UN TEXTO DEBE DE SELECCIONAR EL TEXTO INVESTIGADO EN UNA PÁGINA WEB Y HACERLE CAPTURA DE PANTALLA COMPLETA, PARA LUEGO COLOCARLA AQUÍ. SI SON VARIAS PREGUNTAS, DEBE COLOCAR VARIAS CAPTURAS DE PANTALLAS COMPLETAS. NO COLOCAR INFORMACIÓN ENCONTRADA EN NETACAD, COLOCAR INFORMACIÓN INVESTIGADA EN OTRAS PÁGINAS WEB O LO REALIZADO POR USTED.**

## Antecedentes / Escenario

Los archivos de registro son una herramienta importante para la solución de problemas y el monitoreo. Cada aplicación genera archivos de registro diferentes, y cada uno contiene su propio conjunto de campos e información. Si bien la estructura de los campos puede variar de un archivo de registro a otro, las herramientas que se utilizan para leerlos son mayormente las mismas. En esta práctica de laboratorio practicarán para aprender a utilizar herramientas comunes que se emplean para leer archivos de registro.

## Recursos necesarios

- Máquina virtual Security Workstation

## Instrucciones

### Parte 1: Leer archivos de registro con Cat, More, Less y Tail

Los archivos de registro son archivos que se utilizan para registrar eventos específicos iniciados por aplicaciones, servicios o el mismo sistema operativo. Suelen almacenarse como texto plano, y son un recurso indispensable para la solución de problemas.

## Paso 1: Abrir archivos de registro

Comúnmente, los archivos de registro contienen información en texto plano que puede ser vista con prácticamente cualquier programa capaz de manejar texto (un editor de texto, por ejemplo). Sin embargo, por motivos de conveniencia, practicidad de uso y velocidad, algunas herramientas se utilizan más que otras. Esta sección se enfoca en cuatro programas basados en la línea de comandos: **cat**, **more**, **less** y **tail**.

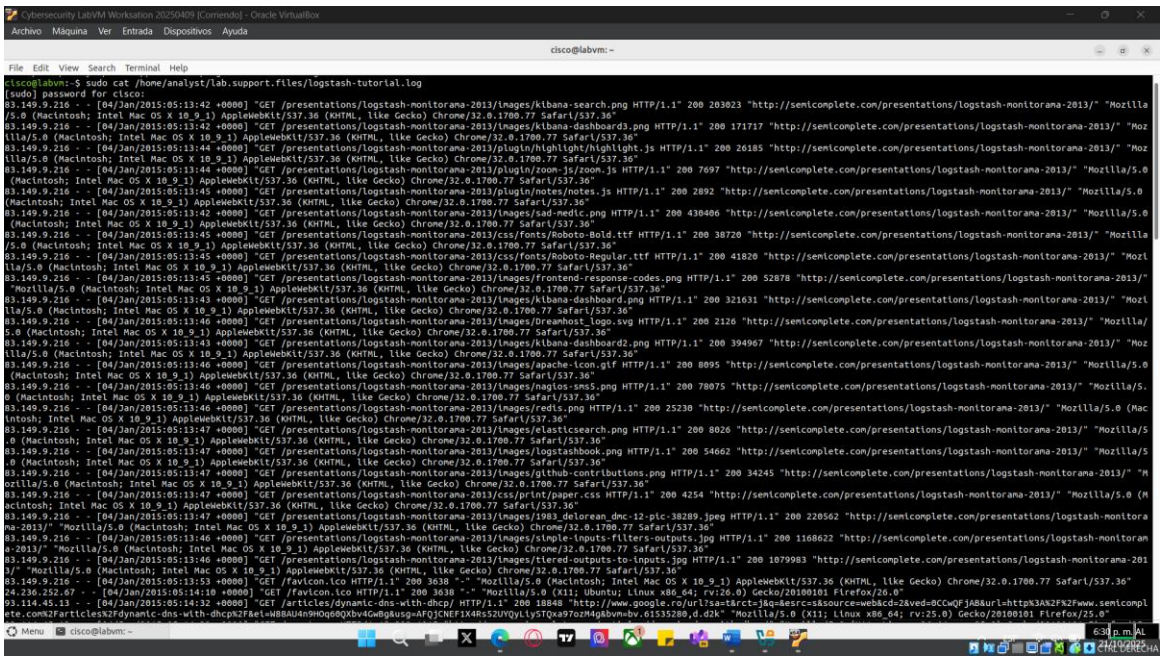
**Cat**, derivado de la palabra 'concatenar', es una herramienta de UNIX basada en la línea de comandos que se utiliza para leer y mostrar el contenido de un archivo en la pantalla. Por su simplicidad y capacidad para abrir un archivo de texto y mostrarlo en un terminal de solo texto, **cat** sigue siendo muy utilizado en la actualidad.

a. Abra el Security Workstation VM y, luego, una ventana del terminal.

b. En la ventana del terminal, emitan el siguiente comando para mostrar el contenido del archivo **logstash-tutorial.log**, que está ubicado en la carpeta **/home/analyst/lab.support.files/**:

```
analyst@secOps ~$ cat /home/analyst/lab.support.files/logstash-tutorial.log
```

El contenido del archivo debería desplazarse por la ventana del terminal hasta haber mostrado todo el contenido.



Indicar una desventaja de utilizar **cat** con archivos de texto grandes.

cat vuela todo el contenido al terminal sin paginación; con archivos muy grandes inunda la pantalla, es difícil navegar/leer, puede causar que se pierda la salida anterior y consume tiempo/memoria si se redirige. No permite retroceder o buscar fácilmente.

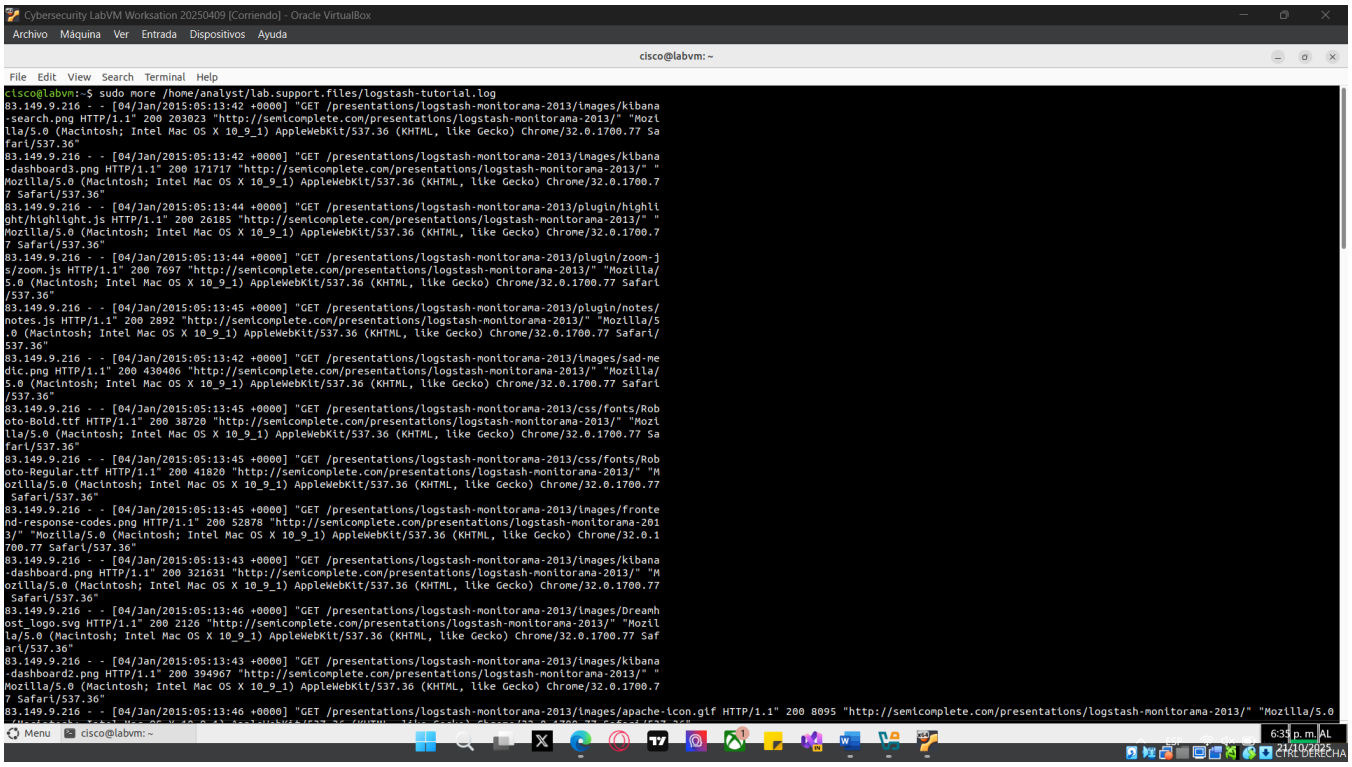
Otra herramienta popular para visualizar archivos de registro es **more**. Al igual que **cat**, **more** también es una herramienta de UNIX basada en la línea de comandos que puede abrir un archivo de texto y mostrar su contenido en la pantalla. La principal diferencia entre **cat** y **more** es que **more** admite saltos de páginas y eso permite que el usuario vea el contenido de un archivo una página por vez. Esto se puede hacer utilizando la barra espaciadora para mostrar la página siguiente.

c. En la misma ventana del terminal, utilicen el siguiente comando para volver a mostrar el contenido del archivo **logstash-tutorial.log**. Esta vez con **more**:

```
analyst@secOps ~$ more /home/analyst/lab.support.files/logstash-tutorial.log
```

El contenido del archivo debería desplazarse por la ventana del terminal y detenerse al llegar a una página en pantalla. Presionen la barra espaciadora para avanzar a la página siguiente. Presionen Intro para mostrar la siguiente línea de texto?




¿Cuál es la desventaja de utilizar **more**?

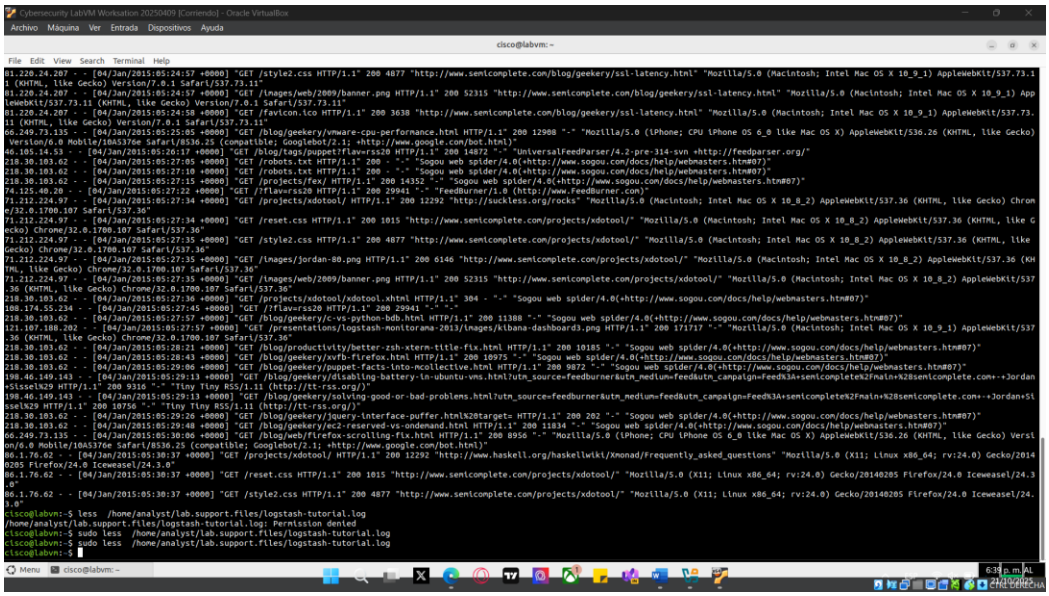
Que permite avanzar por pagina pero no permite retroceder

Sobre la base de la funcionalidad de **cat** y **more**, la herramienta **less** permite mostrar el contenido de un archivo página por página y, a la vez, permitir que el usuario opte por visualizar páginas ya mostradas en pantalla.

d. En la misma ventana del terminal, utilicen **less** para volver a mostrar el contenido del archivo **logstash-tutorial.log**:

```
analyst@secOps ~$ less /home/analyst/lab.support.files/logstash-tutorial.log
```

El contenido del archivo debería desplazarse por la ventana del terminal y detenerse al llegar a una página en pantalla. Presionen la barra espaciadora para avanzar a la página siguiente. Presionen Intro para mostrar la siguiente línea de texto? Utilicen las teclas de las flechas hacia arriba y hacia abajo para avanzar y retroceder por el archivo de texto.




Presionen la tecla "**q**" del teclado para salir de la herramienta **less**.

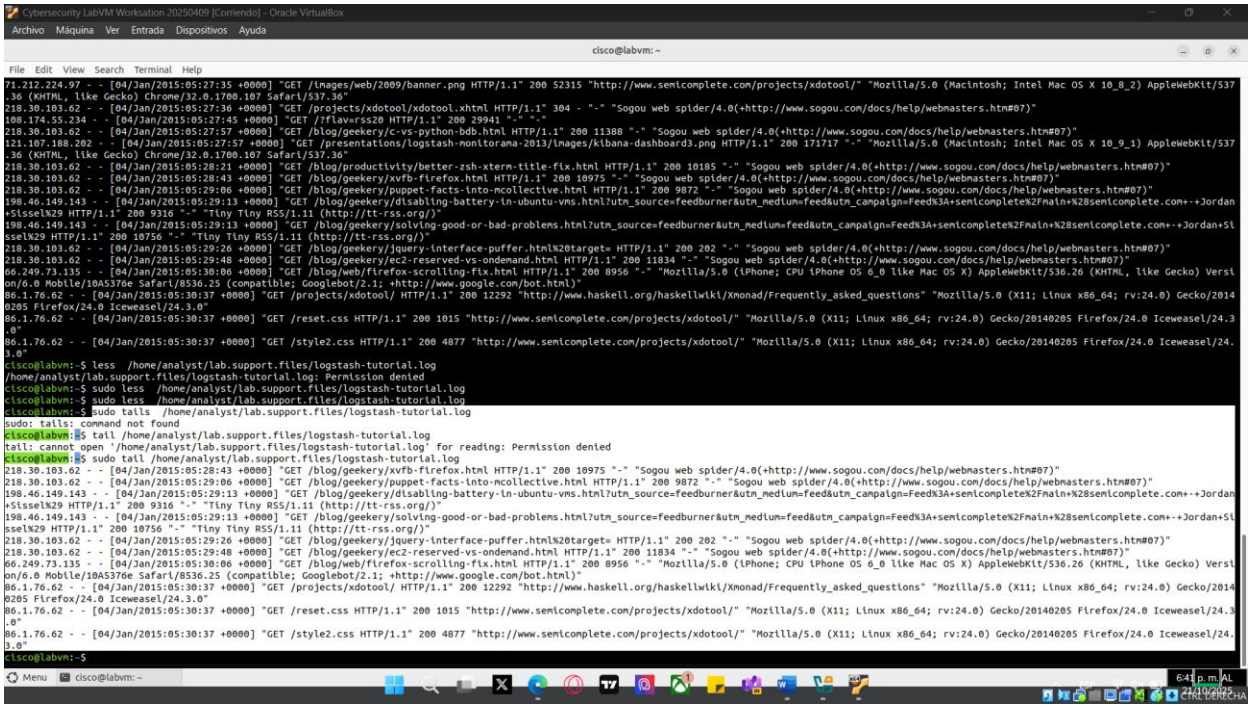
e. El comando **tail** muestra el final de un archivo de texto. De manera predeterminada, **tail** muestra las últimas diez líneas del archivo.

Utilicen tail para mostrar las últimas diez líneas del archivo **/home/analyst/lab.support.files/logstash-tutorial.log**.

```
analyst@secOps ~$ tail /home/analyst/lab.support.files/logstash-tutorial.log
```

```
218.30.103.62 - - [04/Jan/2015:05:28:43 +0000] "GET /blog/geekery/xvfb-firefox.html
HTTP/1.1" 200 10975 "-" "Sogou web
spider/4.0(+http://www.sogou.com/docs/help/webmasters.htm#07)"
218.30.103.62 - - [04/Jan/2015:05:29:06 +0000] "GET /blog/geekery/puppet-facts-into-
mcollective.html HTTP/1.1" 200 9872 "-" "Sogou web
spider/4.0(+http://www.sogou.com/docs/help/webmasters.htm#07)"
198.46.149.143 - - [04/Jan/2015:05:29:13 +0000] "GET /blog/geekery/disabling-battery-
in-ubuntu-
vms.html?utm_source=feedburner&utm_medium=feed&utm_campaign=Feed%3A+semicomplete%2Fmai
n+%28semicomplete.com+-+Jordan+Sissel%29 HTTP/1.1" 200 9316 "-" "Tiny Tiny RSS/1.11
(http://tt-rss.org/)"
198.46.149.143 - - [04/Jan/2015:05:29:13 +0000] "GET /blog/geekery/solving-good-or-
bad-
problems.html?utm_source=feedburner&utm_medium=feed&utm_campaign=Feed%3A+semicomplete%
2Fmain+%28semicomplete.com+-+Jordan+Sissel%29 HTTP/1.1" 200 10756 "-" "Tiny Tiny
RSS/1.11 (http://tt-rss.org/)"
218.30.103.62 - - [04/Jan/2015:05:29:26 +0000] "GET /blog/geekery/jquery-interface-
puffer.html%20target= HTTP/1.1" 200 202 "-" "Sogou web
spider/4.0(+http://www.sogou.com/docs/help/webmasters.htm#07)"
218.30.103.62 - - [04/Jan/2015:05:29:48 +0000] "GET /blog/geekery/ec2-reserved-vs-
ondemand.html HTTP/1.1" 200 11834 "-" "Sogou web
spider/4.0(+http://www.sogou.com/docs/help/webmasters.htm#07)"
66.249.73.135 - - [04/Jan/2015:05:30:06 +0000] "GET /blog/web/firefox-scrolling-
fix.html HTTP/1.1" 200 8956 "-" "Mozilla/5.0 (iPhone; CPU iPhone OS 6_0 like Mac OS X)
AppleWebKit/536.26 (KHTML, like Gecko) Version/6.0 Mobile/10A5376e Safari/8536.25
(compatible; Googlebot/2.1; +http://www.google.com/bot.html)"
86.1.76.62 - - [04/Jan/2015:05:30:37 +0000] "GET /projects/xdotool/ HTTP/1.1" 200
12292 "http://www.haskell.org/haskellwiki/Xmonad/Frequently_asked_questions"
```

```
"Mozilla/5.0 (X11; Linux x86_64; rv:24.0) Gecko/20140205 Firefox/24.0
Iceweasel/24.3.0"
86.1.76.62 - - [04/Jan/2015:05:30:37 +0000] "GET /reset.css HTTP/1.1" 200 1015
"http://www.semicomplete.com/projects/xdotool/" "Mozilla/5.0 (X11; Linux x86_64;
rv:24.0) Gecko/20140205 Firefox/24.0 Iceweasel/24.3.0"
86.1.76.62 - - [04/Jan/2015:05:30:37 +0000] "GET /style2.css HTTP/1.1" 200 4877
"http://www.semicomplete.com/projects/xdotool/" "Mozilla/5.0 (X11; Linux x86_64;
rv:24.0) Gecko/20140205 Firefox/24.0 Iceweasel/24.3.0"
```

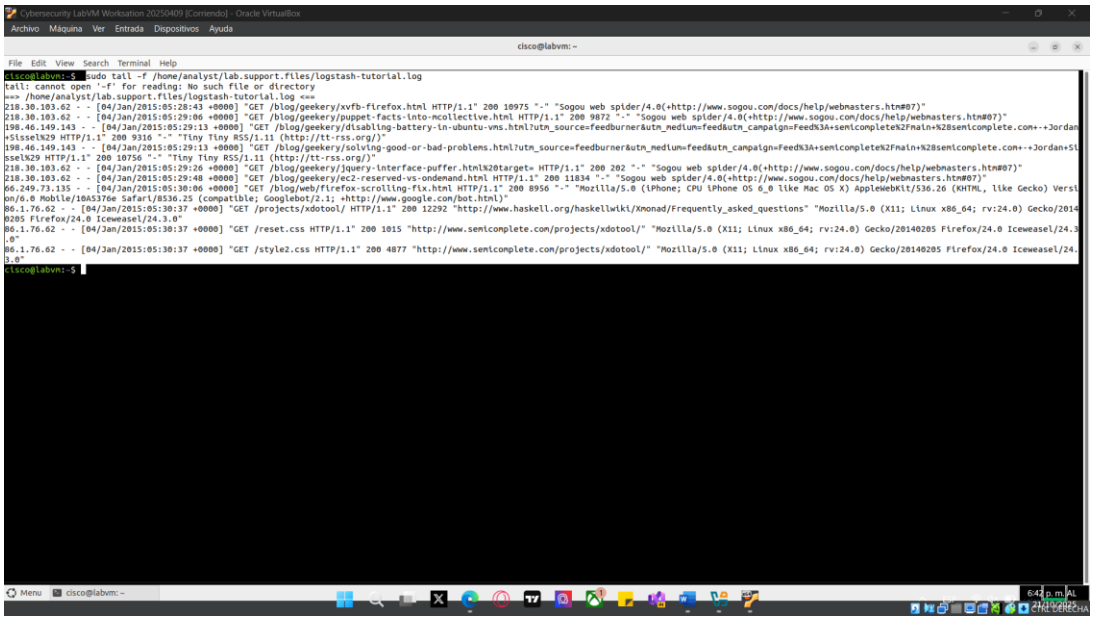



## Paso 2: Seguir archivos de registro en forma activa

En algunas situaciones, lo aconsejable es monitorear archivos de registro a medida que se les escriben las entradas de registro. El comando **tail -f** es muy útil para esos casos.

a. Utilicen tail -f para monitorear en forma activa el contenido del archivo **/var/log/syslog**:

```
analyst@secOps ~$ sudo tail -f /home/analyst/lab.support.files/logstash-
tutorial.log
```



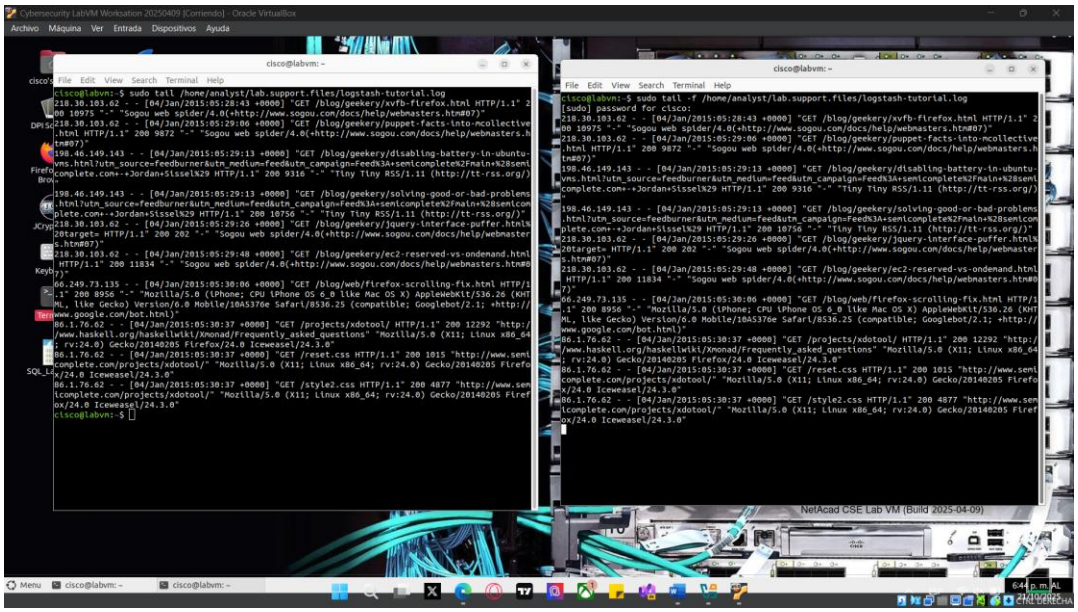

¿En qué difieren las salidas de **tail** y de **tail -f**? Explique.

tail muestra las últimas N líneas (por defecto 10) y finaliza. tail -f muestra las últimas líneas y además **permanece en ejecución**, mostrando en tiempo real cualquier nueva línea que se agregue al archivo hasta que lo detengas (Ctrl+C).

b. Abran una segunda ventana del terminal para ver tail –f en acción. Organicen la pantalla de modo que puedan ver ambas ventanas del terminal. Cambien el tamaño de las ventanas para poder verlas a la vez, tal como se muestra en la siguiente imagen:

En la ventana del terminal de arriba se está ejecutando **tail -f** para monitorear el archivo **/home/analyst/lab.support.files/logstash-tutorial.log**. Utilicen la ventana del terminal de abajo para agregar información al archivo monitoreado.

Para simplificar la visualización, seleccionen la ventana del terminal de arriba (donde se está ejecutando **tail -f**) y presionen Intro un par de veces. Con esto se agregarán algunas líneas entre el contenido actual del archivo y la información nueva que se debe sumar.



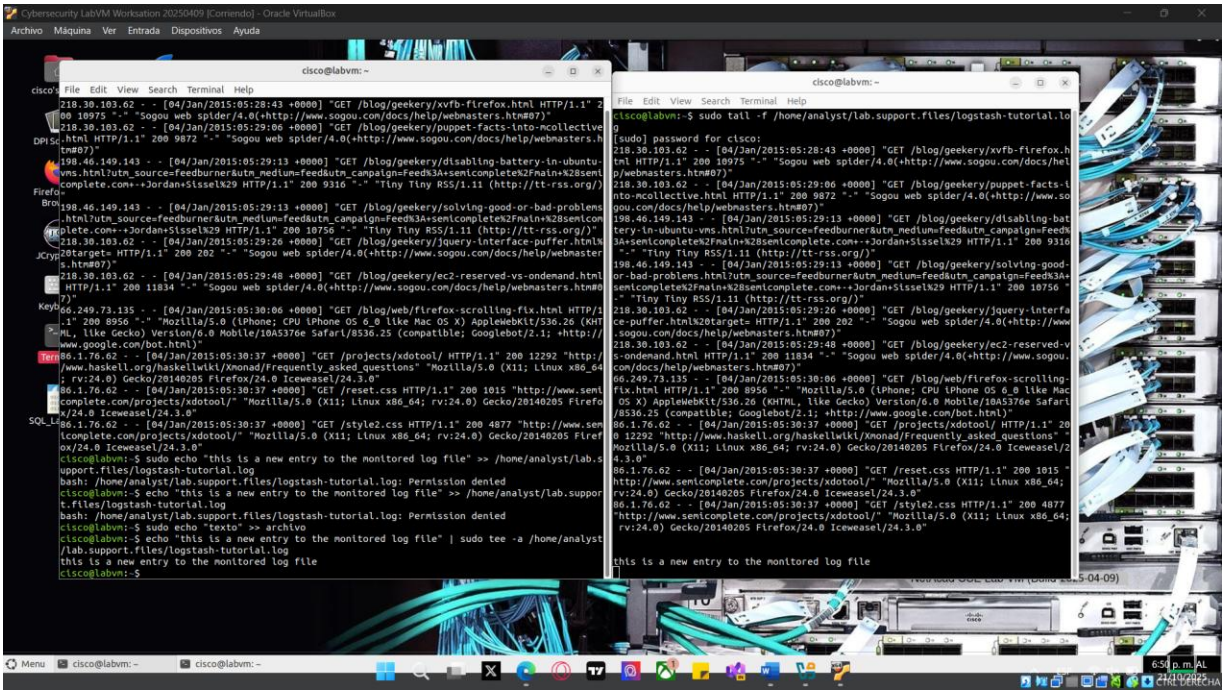
c. Seleccionen la ventana del terminal de abajo e introduzcan el siguiente comando:

```
[analyst@secOps ~]$ echo "this is a new entry to the monitored log file" >>
lab.support.files/logstash-tutorial.log
```

El comando anterior anexa el mensaje "this is a new entry to the monitored log file" ("esta es una entrada nueva que se agrega al archivo de registro monitoreado") al archivo **/home/analyst/lab.support.files/logstash-tutorial.log**. Como tail –f está monitoreando el archivo en ese momento, se agregar una línea al archivo. En la ventana de arriba debería aparecer la línea nueva en tiempo real.

d. Presionen CTRL + C para detener la ejecución de **tail -f** y regresar al cursor del shell.

e. Cierren una de las dos ventanas del terminal.



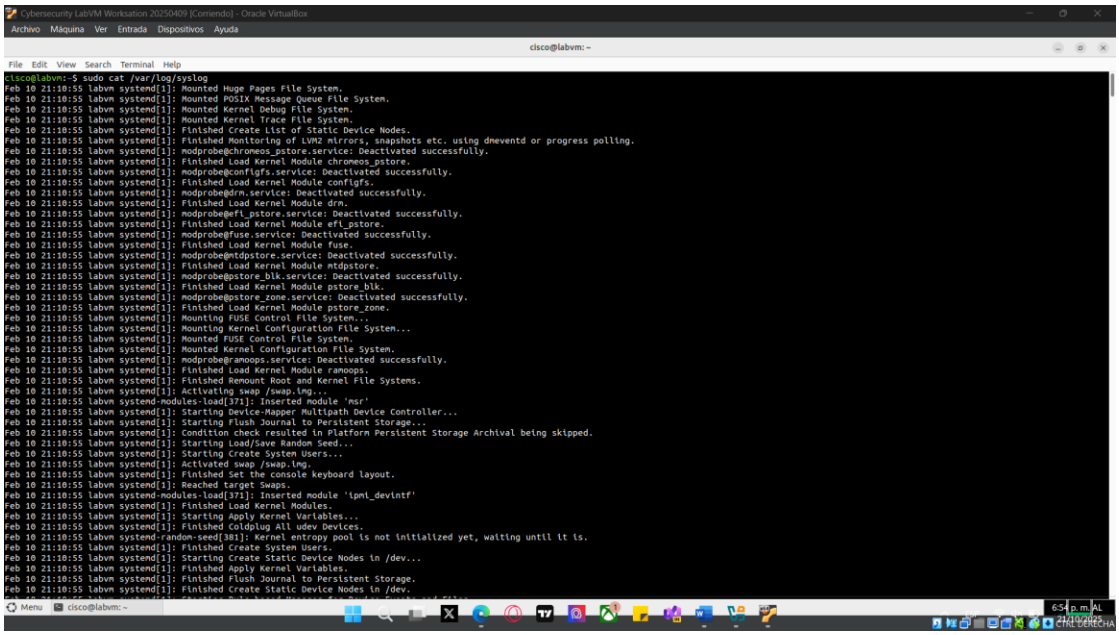
## Parte 2: Archivos de registro y Syslog

Debido a su importancia, es común concentrar archivos de registro en una computadora de monitoreo. **Syslog** es un sistema diseñado para permitir que los dispositivos envíen sus archivos de registro a un servidor centralizado, que se conoce como servidor **syslog**. Los clientes se comunican con un servidor syslog por medio del protocolo **syslog**. La implementación de **Syslog** es generalizada y admite prácticamente todas las plataformas informáticas.

La VM Security Worstation genera archivos de registro al nivel del sistema operativo y se los entrega a **syslog**.

a. Utilice el comando **cat** como usuario **root** para generar una lista del contenido del archivo **/var/log/syslog.1**. Este archivo contiene las entradas de registro (log entries) generadas por el sistema operativo de la VM Security Workstation y las enviadas al servicio **syslog**.

```
analyst@secOps ~$ sudo cat /var/log/syslog.1
[sudo] contraseña para analyst:
Feb 7 13:23:15 secOps kernel: [ 5.458959] psmouse serio1: hgpk: ID: 10 00 64
Feb 7 13:23:15 secOps kernel: [ 5.467285] input: ImExPS/2 BYD TouchPad as
/devices/platform/i8042/serio1/input/input6
Feb 7 13:23:15 secOps kernel: [ 5.502469] RAPL PMU: API unit is 2^-32 Joules, 4 fixed
counters, 10737418240 ms ovfl timer
Feb 7 13:23:15 secOps kernel: [ 5.502476] RAPL PMU: hw unit of domain pp0-core 2^-0
Joules
Feb 7 13:23:15 secOps kernel: [ 5.502478] RAPL PMU: hw unit of domain package 2^-0
Joules
Feb 7 13:23:15 secOps kernel: [ 5.502479] RAPL PMU: hw unit of domain dram 2^-0 Joules
Feb 7 13:23:15 secOps kernel: [ 5.502480] RAPL PMU: hw unit of domain pp1-gpu 2^-0
Joules
Feb 7 13:23:15 secOps kernel: [ 5.672547] ppdev: user-space parallel port driver
Feb 7 13:23:15 secOps kernel: [ 5.709000] pcnet32 0000:00:03.0 enp0s3: renamed from
eth0
Feb 7 13:23:16 secOps kernel: [ 6.166738] pcnet32 0000:00:03.0 enp0s3: link up,
100Mbps, full-duplex
Feb 7 13:23:16 secOps kernel: [ 6.706058] random: crng init done
Feb 7 13:23:18 secOps kernel: [ 8.318984] floppy0: no floppy controllers found
Feb 7 13:23:18 secOps kernel: [ 8.319028] work still pending
Feb 7 14:26:35 secOps kernel: [ 3806.118242] hrtimer: interrupt took 4085149 ns
Feb 7 15:02:13 secOps kernel: [ 5943.582952] pcnet32 0000:00:03.0 enp0s3: link down
Feb 7 15:02:19 secOps kernel: [ 5949.556153] pcnet32 0000:00:03.0 enp0s3: link up,
100Mbps, full-duplex
```



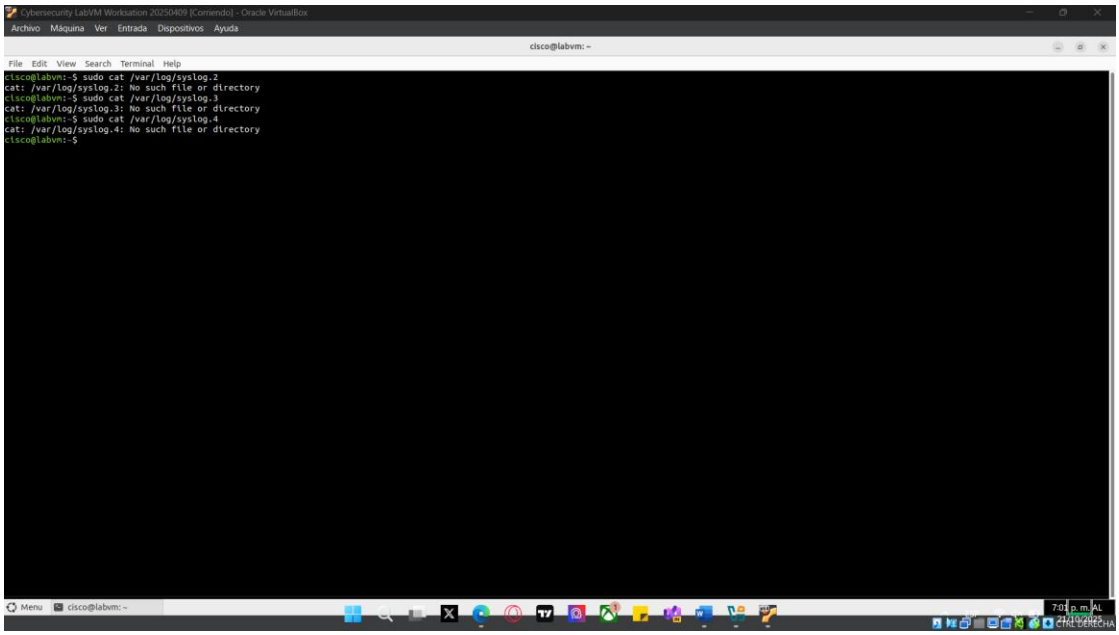

¿Por qué se tuvo que ejecutar el comando **cat** como usuario **root**?

Se tuvo que ejecutar el comando cat como usuario root porque muchos archivos de registro del sistema, como los que están en /var/log/, tienen permisos restringidos por razones de seguridad.

b. Observen que el archivo **/var/log/syslog** solo almacena las entradas de registro más recientes. Para que el archivo de syslog no se extienda demasiado, el sistema operativo rota periódicamente los archivos de registro y les cambia el nombre a los más antiguos por **syslog.1**, **syslog.2**, y así sucesivamente.

Utilicen el comando **cat** para generar una lista de archivos de **syslog** más antiguos:

```
analyst@secOps ~$ sudo cat /var/log/syslog.2
analyst@secOps ~$ sudo cat /var/log/syslog.3
analyst@secOps ~$ sudo cat /var/log/syslog.4
```



¿Puede pensar en algún motivo por el cual es importante mantener sincronizadas la fecha y la hora de las computadoras?

una hora precisa y sincronizada garantiza que los registros sean confiables y coherentes en toda la red.

## Parte 3: Archivos de registro y Journalctl

Otro sistema de administración de registros muy utilizado se conoce como **journal**. Administrador por el daemon **journald**, el sistema está diseñado para centralizar la administración de archivos de registro independientemente de dónde se estén originando los mensajes. En el contexto de esta práctica de laboratorio, la características más evidente del daemon del sistema **journal** es el uso de archivos binarios "append-only" (solo anexar) que actúan como sus **archivos de registro**.

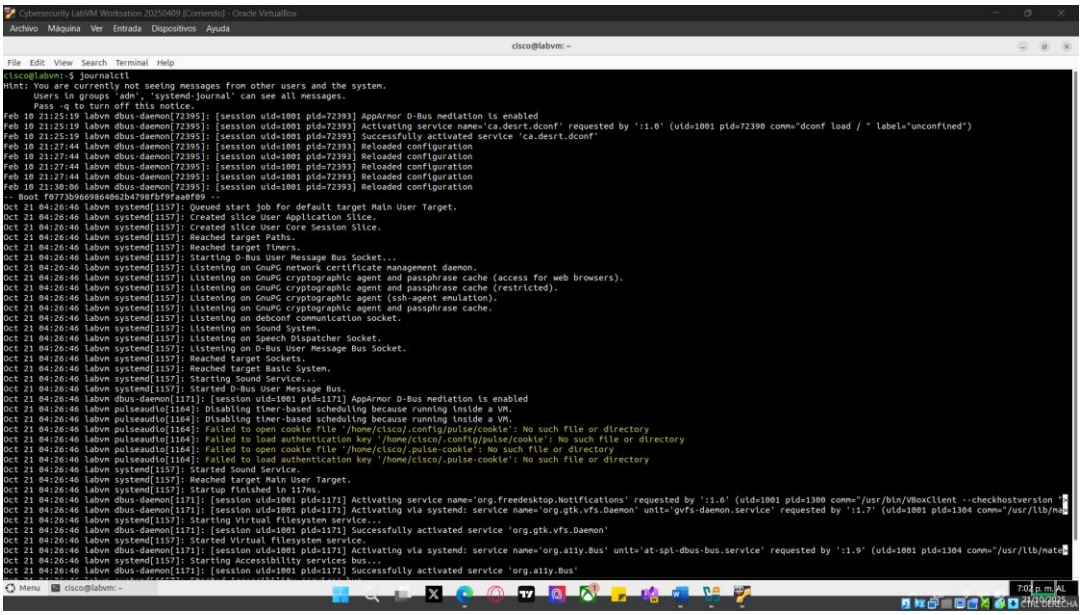
### Paso 1: Ejecutar journalctl sin opciones.

a. Para ver los archivos de registro de **journald** utilicen el comando **journalctl**. La herramienta **journalctl** interpreta y muestra las entradas de registro almacenadas anteriormente en los archivos de registro binario de **journal**.

```
analyst@secOps ~$ journalctl
-- Logs begin at Fri 2014-09-26 14:13:12 EDT, end at Tue 2017-02-07 13:23:29 ES
Sep 26 14:13:12 dataAnalyzer systemd[1087]: Starting Paths.
Sep 26 14:13:12 dataAnalyzer systemd[1087]: Reached target Paths.
Sep 26 14:13:12 dataAnalyzer systemd[1087]: Starting Timers.
Sep 26 14:13:12 dataAnalyzer systemd[1087]: Reached target Timers.
Sep 26 14:13:12 dataAnalyzer systemd[1087]: Starting Sockets.
Sep 26 14:13:12 dataAnalyzer systemd[1087]: Reached target Sockets.
Sep 26 14:13:12 dataAnalyzer systemd[1087]: Starting Basic System.
Sep 26 14:13:12 dataAnalyzer systemd[1087]: Reached target Basic System.
Sep 26 14:13:12 dataAnalyzer systemd[1087]: Starting Default.
Sep 26 14:13:12 dataAnalyzer systemd[1087]: Reached target Default.
Sep 26 14:13:12 dataAnalyzer systemd[1087]: Startup finished in 18ms.
Sep 26 14:14:24 dataAnalyzer systemd[1087]: Stopping Default.
Sep 26 14:14:24 dataAnalyzer systemd[1087]: Stopped target Default.
Sep 26 14:14:24 dataAnalyzer systemd[1087]: Stopping Basic System.
Sep 26 14:14:24 dataAnalyzer systemd[1087]: Stopped target Basic System.
Sep 26 14:14:24 dataAnalyzer systemd[1087]: Stopping Paths.
Sep 26 14:14:24 dataAnalyzer systemd[1087]: Stopped target Paths.
Sep 26 14:14:24 dataAnalyzer systemd[1087]: Stopping Timers.
Sep 26 14:14:24 dataAnalyzer systemd[1087]: Stopped target Timers.
Sep 26 14:14:24 dataAnalyzer systemd[1087]: Stopping Sockets.
<output omitted>
```

**Nota**: Si se ejecuta journalctl como usuario root se mostrará información más detallada.

b. Presionen CTRL+C para salir de la pantalla.

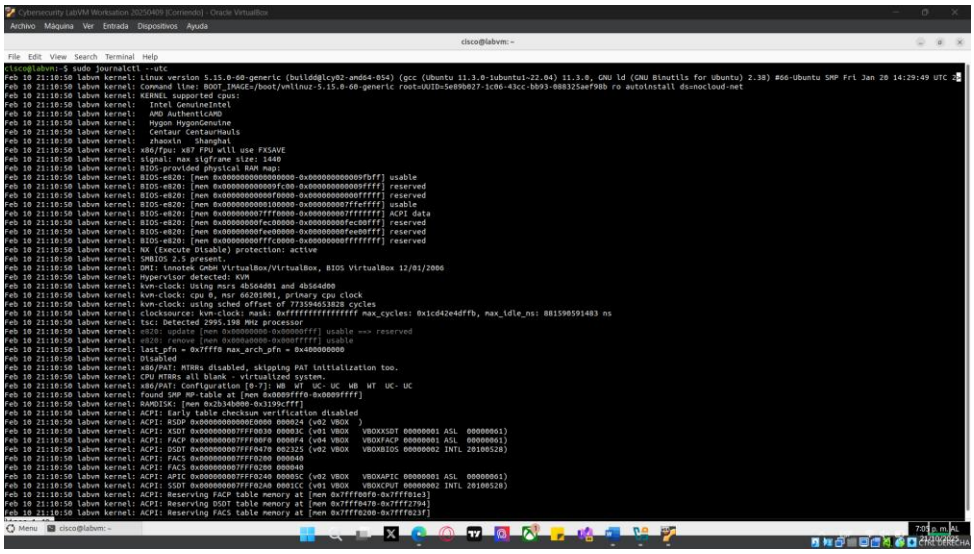


## Paso 2: Journalctl y algunas opciones

Parte de las ventajas de utilizar **journalctl** radica en sus opciones. Para los siguientes comandos, utilice CRTL+C para salir de la pantalla.

a. Utilice **journalctl --utc** para mostrar todas las marcas de hora UTC:

```
analyst@secOps ~$ sudo journalctl –utc
```

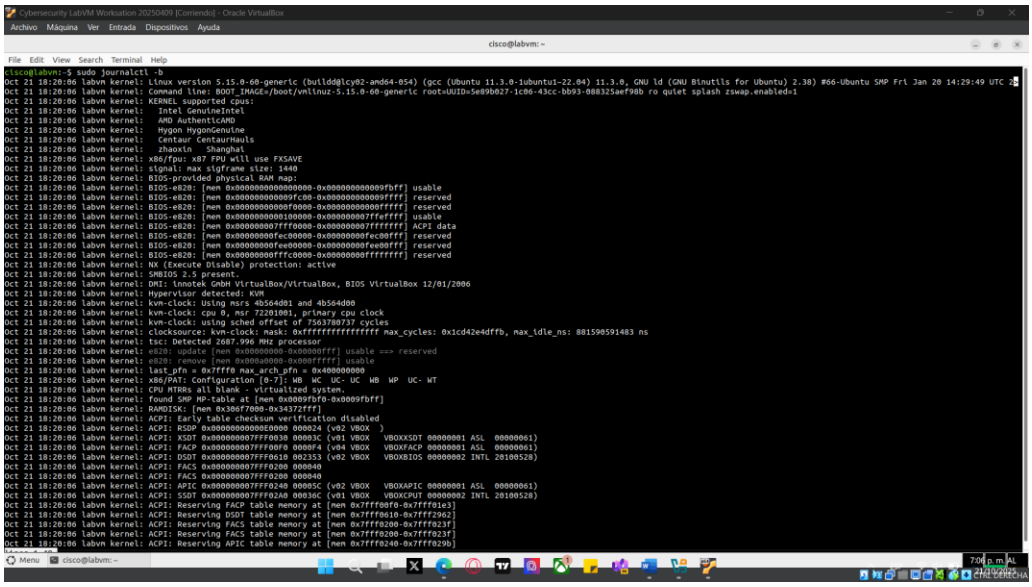


b. Utilicen **journalctl -b** para mostrar las entradas de registro registradas durante el último arranque:

```
analyst@secOps ~$ sudo journalctl –b
Feb 07 08:23:13 secOps systemd-journald[172]: Time spent on flushing to /var is
Feb 07 08:23:13 secOps kernel: Linux version 4.8.12-2-ARCH (builduser@andyrtr)
Feb 07 08:23:13 secOps kernel: x86/fpu: Supporting XSAVE feature 0x001: 'x87 fl
Feb 07 08:23:13 secOps kernel: x86/fpu: Supporting XSAVE feature 0x002: 'SSE re
Feb 07 08:23:13 secOps kernel: x86/fpu: Supporting XSAVE feature 0x004: 'AVX re
Feb 07 08:23:13 secOps kernel: x86/fpu: xstate_offset[2]: 576, xstate_sizes[2]
Feb 07 08:23:13 secOps kernel: x86/fpu: Enabled xstate features 0x7, context si
```

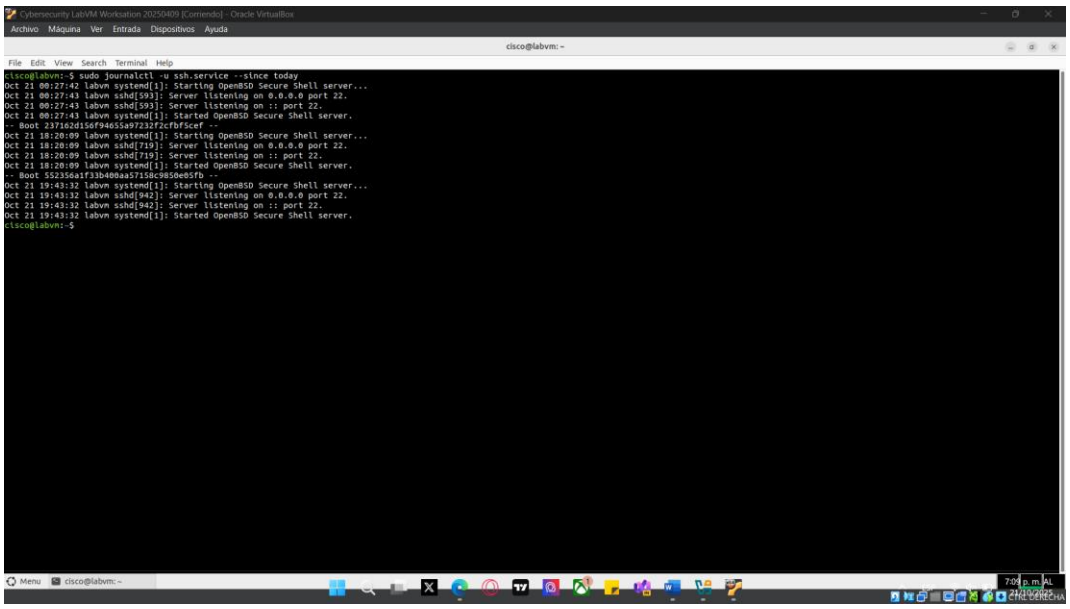
 

```
Feb 07 08:23:13 secOps kernel: x86/fpu: Using 'eager' FPU context switches.
Feb 07 08:23:13 secOps kernel: e820: BIOS-provided physical RAM map:
<output omitted>
```



c. Utilicen **journalctl** para especificar el servicio y el período para las entradas de registro. El siguiente comando muestra todos los archivos de registro de **nginx** que se registraron hoy:

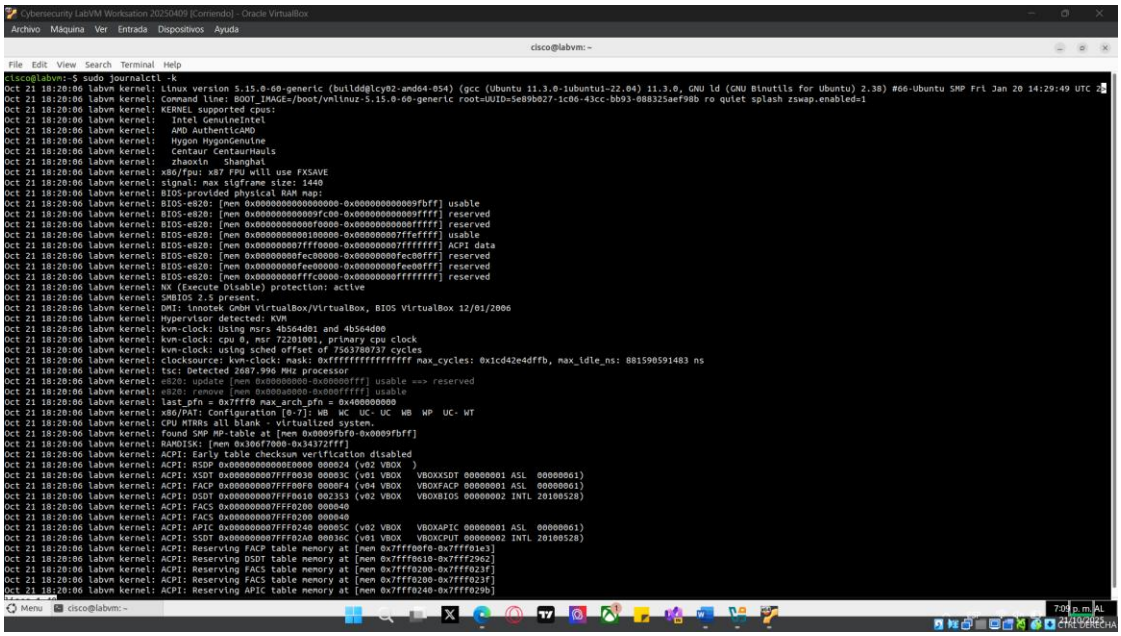
```
analyst@secOps ~$ sudo journalctl -u nginx.service --since today
```



d. Utilicen el switch **-k** para mostrar solo mensajes generados por el kernel:

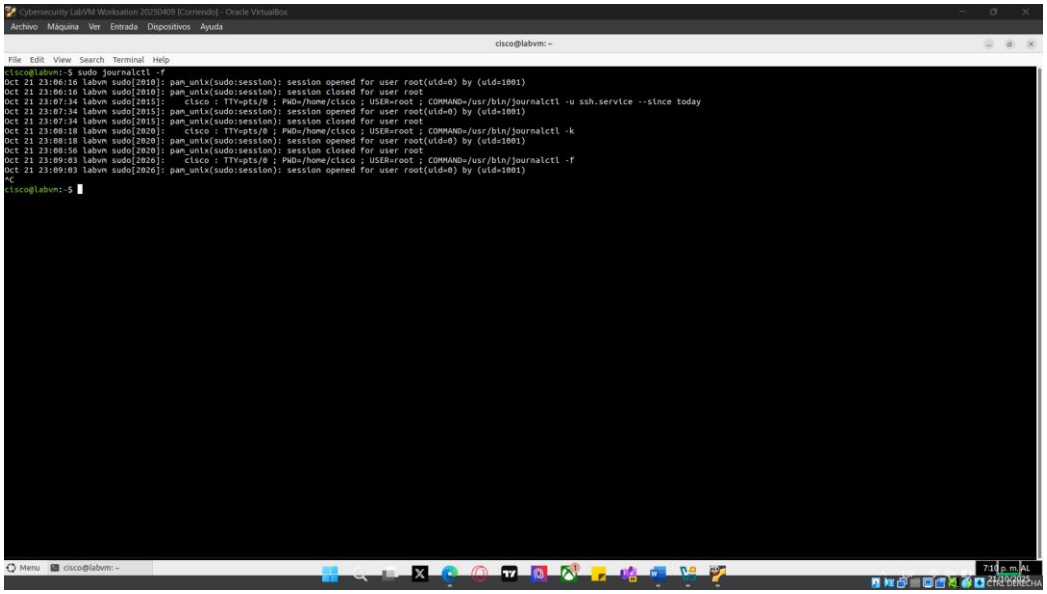
```
analyst@secOps ~$ sudo journalctl -k
```




e. En forma similar a lo que sucede con **tail -f** antes descrito, utilicen el switch **-f** parea seguir los archivos de registro en forma activa a medida que se los escribe:

```
analyst@secOps ~$ sudo journalctl –f
```




## Pregunta de reflexión

Comparen Syslog con Journald. ¿Cuáles son las ventajas y desventajas de cada uno?

**Syslog:** simple, texto plano, fácil de leer, pero poca información extra y difícil de filtrar.
**Journald:** más completo, guarda metadatos, fácil de buscar y seguro, pero binario y necesita journalctl para leerlo.