



SERVIDOR DE CENTRALIZACIÓN DE LOGS

MARIO CARO GARCÍA

CICLO SUPERIOR DE ADMINISTRACIÓN DE SISTEMAS INFORMÁTICOS EN RED

ÍNDICE

CONTEXTO Y JUSTIFICACIÓN.....	5
OBJETIVOS.....	6
ENFOQUE Y MÉTODO SEGUIDO.....	6
PLANIFICACIÓN DEL PROYECTO.....	7
TAREAS A REALIZAR.....	8
DIAGRAMA DE GANTT.....	9
PRIMER CASO.....	9
SEGUNDO CASO.....	10
COSTES DEL PROYECTO.....	10
LIMITACIONES.....	11
GLOSARIO.....	11
BIBLIOGRAFÍA.....	12
ENTORNO DE VIRTUALIZACIÓN.....	13
ESTUDIO DEL ENTORNO DE TRABAJO.....	16
INSTALACIÓN DE SERVIDOR.....	17
IMPLEMENTACIÓN DE PROGRAMAS.....	22
MANUAL DE INSTALACIÓN.....	23
GRAFANA.....	23
LOKI.....	26
PROMTAIL.....	29
PRIMER CASO (SWITCH SYSLOG).....	30
CONFIGURACIÓN DE PROMTAIL.....	31
RSYSLOG.....	32
LOGS.....	36
MEJORAS.....	40
SERVICIOS (Loki & Promtail).....	40
ALMACENAMIENTO PERSISTENTE.....	46
SEGUNDO CASO (WINDOWS DNS SERVER).....	50
MÉTODO DE IMPLEMENTACIÓN.....	51
WINDOWS SERVER.....	51
UBUNTU SERVER.....	55
SCRIPT.....	58
MONTAJE DE LA CARPETA COMPARTIDA.....	59
ANALIZADOR SINTÁCTICO.....	62
ELIMINAR LÍNEAS VACÍAS.....	65
TRADUCCIONES.....	65
DEFINIR ORIGEN Y DESTINO.....	66
ESPERAR NUEVAS LÍNEAS.....	67
SCRIPT PARSER (Servicio).....	70
EJECUCIÓN.....	72
CONCLUSIONES.....	73



Esta obra está sujeta a una licencia de
Reconocimiento - No Comercial - Sin Obra Derivada
[3.0 España de Creative Commons](https://creativecommons.org/licenses/by-nc-nd/3.0/es/)

FICHA DEL PROYECTO

Título del trabajo:	Servidor de Centralización de Logs
Nombre del autor:	Mario Caro García
Fecha de entrega (dd/mm/aa):	13/06/2023
Área del trabajo final:	Seguridad informática, monitoreo y análisis de registros
Ciclo Grado Superior:	Administración de Sistemas Informáticos en Red
Resumen del trabajo (máximo 250 palabras):	
<p>El proyecto tiene como objetivo principal la creación de un sistema de recepción de logs, centrándose en la instalación del servidor en una máquina virtual y su aplicación en dos casos prácticos. Además, se propone un método de estudio para evaluar las necesidades de los entornos de trabajo y se identifican las empresas que se beneficiarían de esta implementación.</p> <p>Para demostrar su utilidad, se presentan dos casos prácticos donde se muestra cómo el sistema de recepción de logs en la máquina virtual mejora la detección de amenazas y vulnerabilidades. Se describen los beneficios obtenidos en cada caso, destacando la eficacia de la solución propuesta en diferentes entornos de trabajo que utilicen máquinas virtuales.</p> <p>Además, se introduce un método de estudio para evaluar las necesidades específicas de los entornos de trabajo que utilicen máquinas virtuales. Se establecen pautas y criterios para identificar las empresas que requieren una mayor seguridad y control en sus sistemas informáticos, enfocando la implementación del sistema de recepción de logs en el entorno virtual de manera efectiva.</p>	

CONTEXTO Y JUSTIFICACIÓN

La ciberseguridad es un tema de suma importancia en la actualidad, especialmente para las empresas y organizaciones que manejan grandes volúmenes de información sensible. La tecnología ha experimentado avances exponenciales en los últimos tiempos, y los programas destinados a la ciberseguridad han evolucionado de manera significativa para hacer frente a las amenazas que pueden poner en peligro la integridad de los datos.

Sin embargo, los hackers y atacantes también han incrementado la eficacia de sus métodos de infiltración para penetrar firewalls y softwares de ciberseguridad, lo cual supone un riesgo a la hora de salvaguardar la integridad de los sistemas informáticos pertenecientes a una empresa u organización.

El correcto seguimiento del estado y actividad de los distintos sistemas informáticos que pertenecen a un entorno de trabajo es fundamental para garantizar la protección de los datos y la información.

Es mediante el desarrollo de este seguimiento que los departamentos de informática y ciberseguridad de una empresa pueden percatarse de posibles brechas de seguridad, intrusiones e incluso elaborar un control de errores en switches, servidores, routers, impresoras, redes internas...

En este aspecto, es donde se presenta el método propuesto por este proyecto para desarrollar un servidor de recepción y centralización de archivos de registro, que cuenta con la capacidad de normalizar formatos, permitir una visualización cómoda de los logs de dispositivos remotos, realizar filtrado de búsquedas y consultas, entre otras funcionalidades.

Este método está sobre todo enfocado a empresas que dependan de la integridad de su red telemática y el correcto funcionamiento de la infraestructura informática. Por ejemplo, una empresa pequeña que se dedica a vender ropa normalmente tiene ordenadores y datos, pero no dependen del funcionamiento de su infraestructura para que el negocio se mantenga a flote. Sin embargo, una empresa de automatización o de ciberseguridad depende completamente de que todos los elementos de la infraestructura informática funcionen correctamente. Este servidor permite tener auditorías en tiempo real de los registros de dichos elementos, asegurando la alta

disponibilidad de los eventos y errores que puedan darse durante los procesos de comunicación entre dispositivos o a la hora de manipular los mismos.

Todos los programas que utilizaremos son de software libre y accesibles para cualquier tipo de empresa que desee incrementar la cibervigilancia de su entorno informático, a excepción de la propia herramienta de virtualización (VMware Workstation Pro)

OBJETIVOS

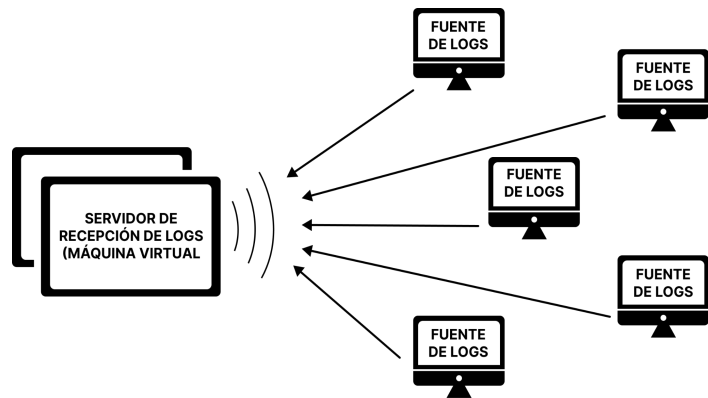
- **Detección de amenazas y vulnerabilidades.**
- **Incrementar el control y la seguridad de los sistemas informáticos de la organización.**
- **Monitorización a tiempo real del entorno de trabajo**
- **Asegurar la alta disponibilidad en cuanto a detección de caídas de conexión, errores en resoluciones...**
- **Reservar inversiones económicas para cubrir otras necesidades**

ENFOQUE Y MÉTODO SEGUIDO

Antes de explicar cómo conseguiremos alcanzar los objetivos propuestos, veo necesario recalcar que se requieren de algunos conocimientos técnicos en Ubuntu y bash script para el desarrollo del servidor, entraremos en profundidad en el apartado de los costes del proyectos. Teniendo en cuenta lo anteriormente expuesto, definiremos el método de implementación:

En primer lugar, vamos a crear una máquina virtual en VMware Workstation y crearemos la máquina que actuará como nuestro servidor. Posteriormente, vamos a documentar dos casos en los cuales se puede implementar este método. En el primero, veremos cómo recoger archivos de registro de switches conectados a nuestra red, lo cual nos puede ayudar a la hora de detectar problemas de red, inicios de sesión no autorizados, brechas en el cumplimiento de normativas y auditorías....

En el segundo, veremos cómo recoger logs de un Windows Server 2016 el cual actúa como servidor DNS de una organización para, de esta forma, garantizar la seguridad y detección de amenazas mediante la identificación de consultas sospechosas, patrones de acceso no autorizados o resolución de dominios maliciosos, entre otras funcionalidades que se explicaré más adelante. Aquí podremos observar un esquema de lo que se pretende conseguir. Un conjunto de fuentes de logs que emiten sus registros a un servidor remoto sustentado en una máquina virtual que se mantiene a la espera de procesar estas emisiones y traducirlas.



PLANIFICACIÓN DEL PROYECTO

Respecto a los recursos necesarios para el desarrollo de este método, debemos tener en cuenta las posibilidades que queramos abarcar, pues los requisitos de hardware pueden variar dependiendo del sistema operativo que queramos monitorear.

Vamos a instalar un Ubuntu Server 22.04 con algunos programas encargados de recoger e interpretar los logs del dispositivo que queremos monitorear. En los ejemplos que veremos más adelante, vamos a utilizar un Cisco Catalyst Series 1000 y un Windows Server DNS

Estas son las especificaciones para el servidor:

- **4GB de RAM**
- **Procesador de 4 núcleos (o 2 núcleos y 2 hilos)**

El equipo donde se llevará a cabo el desarrollo de este servidor posee las siguientes características:

- **Intel(R) Core(TM) i7-10750H CPU @ 2.60GHz 2.59 GHz**
- **16,0 GB de RAM**
- **Nvidia GeForce GTX 1060**

TAREAS A REALIZAR

A continuación, vamos a enumerar y explicar brevemente todas las tareas a realizar para desarrollar el servidor de recepción de logs.

- **Descargar y configurar entorno de virtualización con VMware (Fase 1):**

En esta sección, nos enfocaremos en descargar y configurar la herramienta VMware, la cual nos permitirá crear los entornos necesarios para llevar a cabo el método propuesto. Esta fase finalizará una vez instalada la utilidad y las redes internas necesarias para cada método.

- **Estudio de entorno de trabajo y elección de dispositivo (Fase 2):**

Ahora debemos escoger aquello que queramos monitorear. En este documento aprenderemos a monitorear tanto un sistema operativo como un dispositivo remoto el cual tiene la capacidad de enviar sus logs. Esta decisión requiere de un estudio del entorno de trabajo y posterior valoración de necesidades más críticas. En cualquier caso, esta fase terminará una vez concluido dicho estudio.

- **Instalar y configurar el servidor de recepción de logs (Fase 3):**

Instalaremos el propio servidor que sustentará las aplicaciones que vamos a utilizar así como el sistema operativo Ubuntu Server 22.04. Vamos a configurarlo de tal forma que cuente con IP estática y pueda ser visto por los dispositivos emisores, momento en el cual habrá concluido la tercera fase.

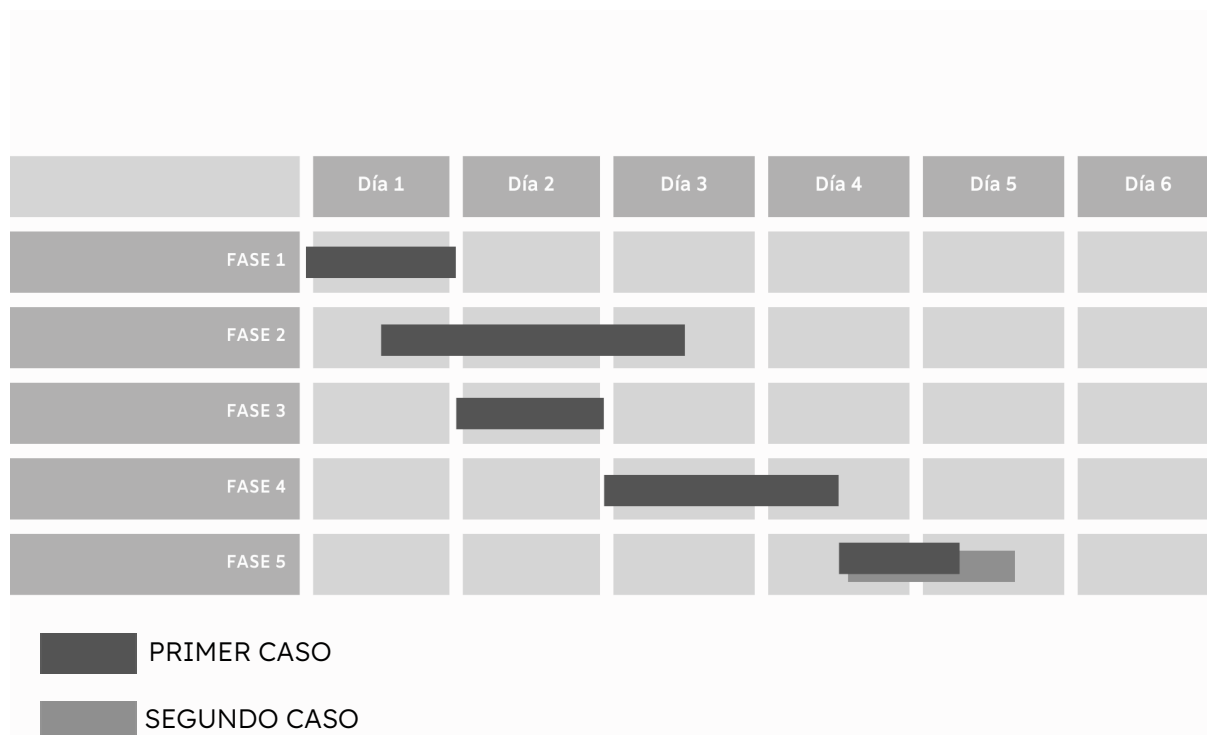
- **Implementar las utilidades pertinentes (Fase 4):**

En este apartado aprenderemos a instalar los programas que nos ayudarán en la recepción y centralización de logs. También las configuraremos y explicaremos algunas posibles mejoras. Al finalizar esta fase, todos los programas deben estar listos para recopilar, visualizar y formatear los logs.

- **Configurar el dispositivo emisor de logs (Fase 5):**

Aquí aprenderemos a configurar el dispositivo emisor de logs. Procederemos de distinta forma para cada uno de los casos. Esta fase finalizará una vez que el servidor reciba correctamente los logs de la fuente en cuestión.

DIAGRAMA DE GANTT



PRIMER CASO

La durabilidad de ejecución de este método no varía en cuanto a la instalación del y configuración del servidor y las utilidades pertinentes, así como en la descarga de la máquina virtual y en el estudio del entorno de trabajo (fases 1, 2, 3 y 4). Sin embargo, la configuración del dispositivo de emisión puede variar bastante dependiendo de su naturaleza.

En el caso del primer escenario donde se podría aplicar este servidor de recepción, la duración puede variar dependiendo del tipo de switch que vayamos a configurar. En la documentación podremos apreciar la configuración de un Switch Cisco Catalyst 1000, al cual vamos a acceder a través de un cable de consola, ya que será necesario ejecutar algunos comandos para definir el servidor al que serán enviados los archivos de registro haciendo uso del protocolo syslog.

SEGUNDO CASO

Sin embargo, el segundo caso puede extenderse más en el tiempo, ya que vamos a configurar un sistema operativo Windows Server 2016 para que guarde sus logs en una carpeta compartida con el servidor (entraremos en detalle más adelante). Esto implica que, si ya tenemos un servidor instalado, tendremos que adaptar sus políticas para permitir la implementación de este método.

COSTES DEL PROYECTO

Como mencioné anteriormente, los programas que utilizaremos son gratuitos y de código abierto a excepción de VMware Workstation Pro, que necesitaremos adquirir una licencia de uso comercial. Aunque existen programas que intentan abordar los problemas expuestos en este documento, todos ellos ofrecen únicamente períodos de prueba limitados y requieren la compra de una versión premium para aprovechar todo su potencial. Sin embargo, la combinación de las utilidades que veremos más adelante permite crear un sistema de vigilancia y monitorización igual de eficiente que estos programas de pago.

Por otro lado, la implementación de este método está pensada para personas instruidas en script y gestión de sistemas Linux. En cuestión de recursos humanos, será necesario dedicar a un técnico al desarrollo de este servidor de monitorización para cumplir las fases dentro del plazo previsto (5 o 6 días). También es recomendable que este técnico participe en el estudio del entorno de trabajo para poder adaptar las soluciones que aporta este documento a cada caso concreto.

Necesitaremos por tanto:

- **Un técnico en administración de sistemas informáticos** con conocimientos en script y gestión de distribuciones Linux
- **Una licencia comercial de VMware Workstation Pro.** En este manual estamos usando la versión 17, que cuesta unos 193 € (pago único)

LIMITACIONES

Debido a la complejidad técnica presente en los manuales, es posible que sea necesario contar con personal capacitado o buscar apoyo externo para su instalación y configuración adecuada.

También hay que tener en cuenta que, dependiendo de la infraestructura y los sistemas informáticos ya implementados en la organización, puede ser necesario realizar ajustes o adaptaciones para garantizar la compatibilidad y la integración adecuada del servidor de recepción de logs. Esto puede requerir tiempo y esfuerzo adicional.

GLOSARIO

- **LOGS:** información generada y almacenada de forma secuencial sobre eventos, acciones o sucesos que ocurren en un sistema informático, una aplicación, un dispositivo o una red. Estos registros son una herramienta fundamental para el diagnóstico de problemas, la monitorización de sistemas y la auditoría de eventos.
- **UDP 514:** "UDP" hace referencia al protocolo de datagramas de usuario, que es un protocolo de transporte utilizado para la transmisión de datos en redes IP. "514" hace referencia al puerto 514 que, en el contexto de UDP, se utiliza comúnmente para el servicio de registro remoto
- **SYSLOG:** Syslog es un protocolo estándar utilizado para el envío y la recepción de logs o registros desde dispositivos de red, sistemas operativos, aplicaciones, o cualquier otro componente que genere información de registro.
- **RFC:** Los RFC son documentos técnicos y estándares desarrollados por la comunidad de Internet Engineering Task Force (IETF) para definir protocolos, especificaciones, prácticas y otros aspectos relacionados con el funcionamiento de Internet y las tecnologías de red. Son relevantes porque varios estándares y protocolos utilizados en la generación, transmisión y gestión de registros están definidos en documentos RFC. Por ejemplo, el protocolo syslog, que se utiliza para enviar y recibir logs, se basa en el RFC 3164 y el RFC 5424.

- **DNS:** el DNS es el sistema que permite la resolución de nombres de dominio en direcciones IP, proporcionando una capa de abstracción que facilita la accesibilidad y usabilidad de Internet.
- **BASH:** explica para un glosario qué es el DNS (resumido)intérprete de comandos y un lenguaje de scripting utilizado en sistemas operativos basados en Unix y Linux.
- **DEPURACIÓN DE REGISTROS (LOG DEBUGGING):** proceso de examinar y analizar los registros o logs generados por un sistema, aplicación o dispositivo con el objetivo de identificar y solucionar problemas, errores o comportamientos inesperados.

SEGUIMIENTO Y CONTROL

Es esencial contar con métodos efectivos para asegurarnos de que las aplicaciones y configuraciones implementadas funcionen correctamente.

Una de las preocupaciones principales es la posible caída temporal del servidor, ya que esto podría resultar en la pérdida de visualización de registros que hayan sido recibidos durante el tiempo de inactividad. Sin embargo, existen soluciones que nos permiten guardar los datos recibidos y visualizarlos una vez que el servidor vuelva a estar en funcionamiento.

Una forma de asegurarnos de que los registros no se pierdan durante una caída del servidor es implementar un sistema de almacenamiento en caché o buffering. Esto implica utilizar un componente como Rsyslog para recopilar y almacenar temporalmente los registros entrantes en un archivo local o en memoria antes de enviarlos al servidor de monitoreo. De esta manera, aunque el servidor no esté disponible temporalmente, los registros se guardarán en la caché y no se perderán.

BIBLIOGRAFÍA

- **Instalación de Grafana:**
 - [Install Grafana on Debian or Ubuntu | Grafana documentation](#)
- **Protocolo Syslog:**
 - [RFC 5424: The Syslog Protocol \(rfc-editor.org\)](#)
- **Instalación de Loki y Promtail:**
 - [Releases · grafana/loki \(github.com\)](#)
 - [Local | Grafana Loki documentation](#)
 - [Configuration | Grafana Loki documentation](#)
 - [Scraping | Grafana Loki documentation](#)
 - [Promtail | Grafana Loki documentation](#)
- **Instalación de VMware**
 - [Máquina virtual Windows | Workstation Pro | VMware | ES](#)
- **Depuración de consultas DNS en Windows Server 2016**
 - [Using server debug logging options: Domain Name System\(DNS\) | Microsoft Learn](#)
 - [Secrets from the Deep – The DNS Analytical Log – Part 5 - Microsoft Community Hub](#) (Eric Jansen)

ENTORNO DE VIRTUALIZACIÓN

En esta fase vamos a desarrollar toda la infraestructura de virtualización que necesitaremos para sustentar las máquinas virtuales, imprescindible para la creación del servidor y necesaria para emular el servidor DNS de Windows. Por ello, es muy importante realizar esta fase correctamente, pues todos los errores que cometamos se van a extender hasta el final del proceso de creación.

En este punto, ya debemos tener nuestro VMware Workstation Pro instalado y activado para empezar a crear las subredes necesarias.

Dicho esto, vamos a justificar la elección de esta utilidad y, posteriormente, explicaremos cómo instalarla y configurarla.

He elegido esta herramienta porque es la más utilizada por empresas a nivel mundial y ofrece una amplia gama de posibilidades en entornos de trabajo.

La licencia de VMware Workstation Pro permite su uso comercial, a diferencia de su versión gratuita. Además, la compra de este producto nos permite escalar el servidor a otras utilidades de VMware (como vSphere, una plataforma de virtualización en la nube muy utilizada).

Una vez descargada y activada la utilidad, vamos a comprobar si la virtualización de nuestra BIOS está operativa. Este procedimiento puede diferir dependiendo del fabricante de tu ordenador o del tipo de placa que tengas instalada. Aún así, los siguientes pasos podrán servirte de guía para activar esta característica de tu sistema.

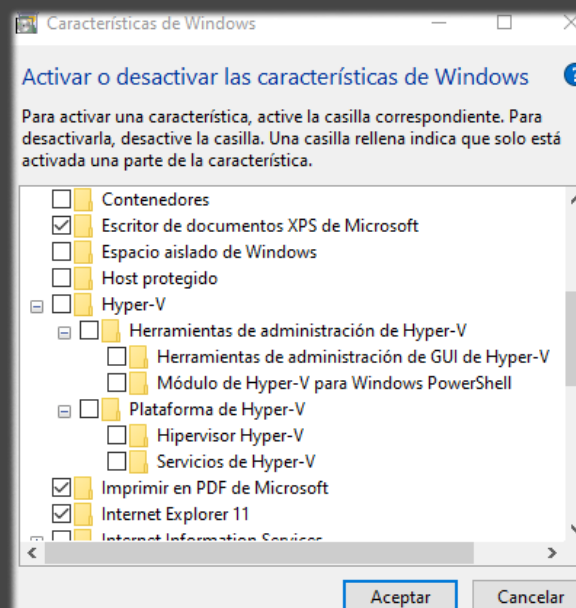
Para entrar a la BIOS, puedes hacerlo desde **Inicio de Windows -> Configuración -> Actualización y Seguridad -> Recuperación -> Inicio avanzado** y cuando se reinicie el ordenador debes acceder a la sección **“Configuración de firmware UEFI”**. Si utilizas otro sistema operativo, otra opción es presionar las teclas “F1”, “F2”, “Supr”, o “F12” al aparecer la pantalla del fabricante de tu placa después de encender el ordenador. De todas formas, en esta pantalla aparecerá la tecla que debes pulsar para acceder a la BIOS.

Una vez dentro, debemos irnos a la sección “Opciones Avanzadas” o similar.

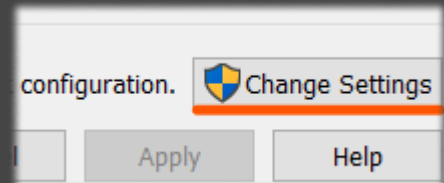
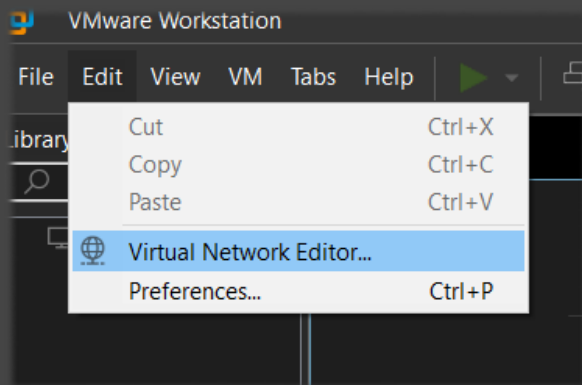
Solamente tienes que encontrar la sección “Tecnología de virtualización” o Virtualization Technology y cambiar su valor para que aparezca como “Activado”. Para ello tendrás que usar el teclado.

Otra forma de activar la virtualización en la BIOS, en diferentes modelos, es buscar la sección llamada “VT-x” o “Virtualization” en diferentes menús de la BIOS. Explóralo hasta que lo encuentres y asegúrate de activarlo con la ayuda del teclado o el ratón.

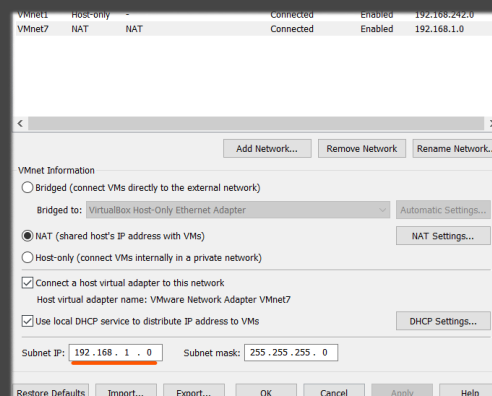
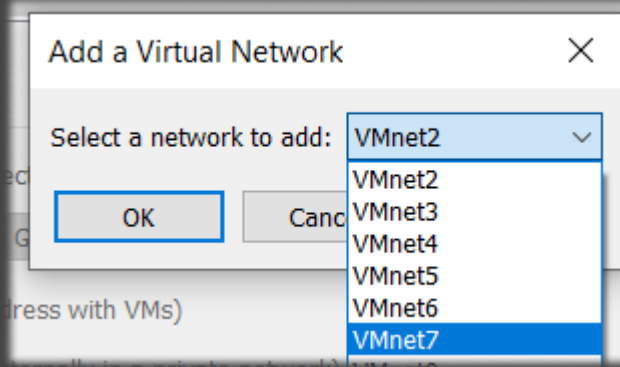
También debemos comprobar si tenemos otras utilidades que puedan interferir con la ejecución de VMware. En el caso de Windows, debemos desactivar la característica Hiper-V desmarcando la casilla “Hyper-V” en el apartado de características de Windows



Ya tenemos el equipo preparado, ahora abriremos VMware Workstation Pro y crearemos la siguiente subred. Nos dirigiremos a la siguiente sección y presionaremos el botón “Change Settings”



Crearemos una nueva red virtual que configuraremos como red NAT. le asignaremos una IP. En mi caso, la vmnet7 tendrá asignada el segmento de red 192.168.1.0



Cuando hayamos terminado, guardamos los cambios y ya tendremos la subred creada, por lo que podremos concluir esta fase del proyecto

ESTUDIO DEL ENTORNO DE TRABAJO

Debemos hacer un estudio adecuado de las necesidades de nuestro entorno informático para elegir el sistema del que necesitemos datos en tiempo real. Para llevar a cabo este estudio, debemos seguir los siguientes pasos

- **Identifica tus objetivos:**

Empieza por definir tus objetivos específicos para la monitorización. ¿Qué aspectos de tu entorno de trabajo deseas supervisar? Algunas cuestiones a tener en cuenta podrían ser el rendimiento del servidor, la disponibilidad de servicios, el consumo de recursos, la seguridad de la red...

- **Enumera los componentes del entorno:**

Debes realizar una lista de todos los componentes informáticos de tu entorno de trabajo, como servidores, switches, routers, dispositivos de almacenamiento, estaciones de trabajo, impresoras, etc. Incluye tanto los componentes físicos como los virtuales.

- **Define los parámetros a monitorear:**

Determina los parámetros específicos que deseas monitorear en cada componente. Por ejemplo, podrías querer supervisar el uso de CPU, el uso de memoria, la utilización de ancho de banda, la latencia de red, los errores de disco, entre otros. Esto te ayudará a recopilar los datos necesarios para evaluar el estado y el rendimiento de cada componente.

- **Considera la infraestructura de red:**

Evalúa la topología de tu red y considera cómo se comunican tus componentes. Identifica los puntos críticos de la red que podrían afectar el rendimiento general y la disponibilidad de los sistemas. Asegúrate de incluir la monitorización de estos puntos críticos en tu estudio.

- **Adaptación de necesidades al servidor de monitorización:**

Una vez hayas considerado los aspectos de los sistemas que deseas visualizar, deberás adaptar tu caso específico a las funcionalidades del servidor. Este manual te ayudará a guiar esa adaptación, ya que explicaremos los casos más comunes y cómo personalizar los parámetros de los programas que utilizaremos para satisfacer tus necesidades en relación a los servicios que ofrece el servidor.

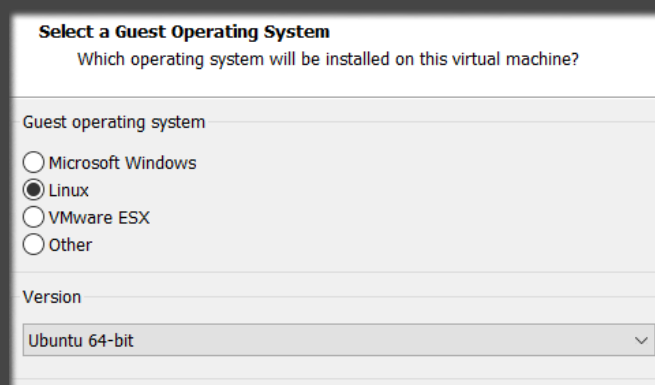
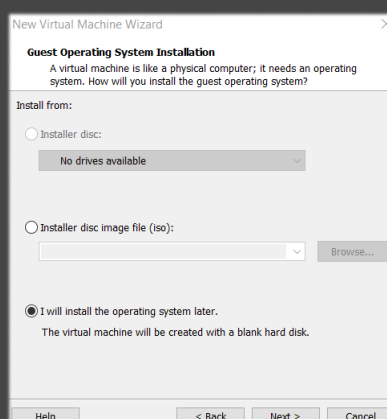
Vamos a especificar en qué consisten los casos que vamos a documentar. En estos casos, estaremos representando las conclusiones del estudio del entorno de trabajo de dos escenarios distintos. En uno de ellos, se ha valorado la necesidad de disponer de auditoría en tiempo real de los registros de los switches de la empresa, debido a que esta organización posee una gran cantidad de enrutadores y, por ende, un gran tráfico de red que se desea monitorear. En el segundo caso, se ha valorado la necesidad de visualizar los logs de las resoluciones de nombres que realiza el servidor DNS por el cual se rigen todos los dispositivos informáticos del entorno estudiado. Esta conclusión se hizo en base a que esta empresa sufrió un ataque de infección botnet por el cual todos sus ordenadores empezaron a lanzar ataques de denegación de servicio a un dominio. Esta infección sucedió debido a la resolución de un dominio malicioso que lentamente empezó a infectar a todos los ordenadores. Si se hubiesen registrado estas consultas, se podrían haber tomado medidas para prevenir o amainar el ataque.

Cabe destacar que ambos casos son perfectamente adaptables y escalables a cualquier entorno que comparta las mismas necesidades (auditoría de DNS o tráfico de red). También te servirán de guía para adaptar tus necesidades a la implementación del servidor.

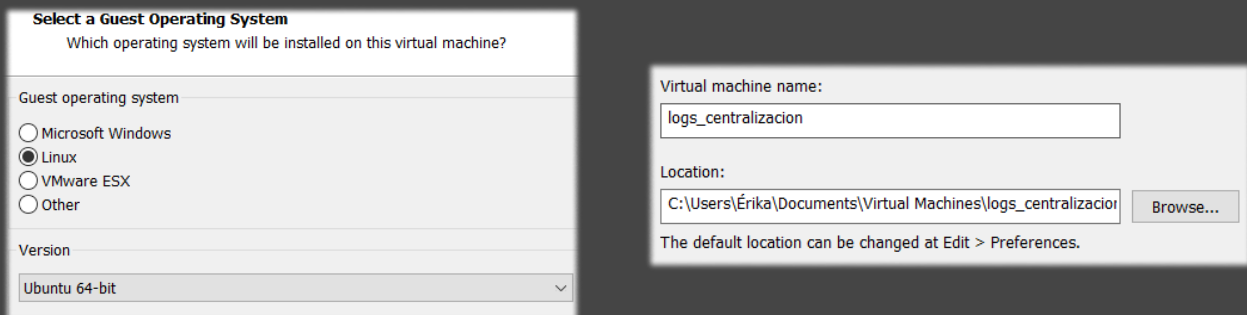
INSTALACIÓN DE SERVIDOR

La instalación del servidor podemos llevarla a cabo después de instalar y configurar VMware, mientras realizamos el estudio del entorno de trabajo.

Lo primero será crear una máquina virtual en VMware Workstation dirigiéndonos a **File -> New Virtual Machine**. Elegiremos la instalación recomendada ("Typical"). Marcaremos la casilla que vemos en la captura, pues vamos a añadir después la ISO de Ubuntu Server 22.04, la cual puedes descargar desde la página oficial de [Ubuntu](https://ubuntu.com/server)



Elegimos como sistema operativo Ubuntu 64 bits y asignamos un nombre y ubicación de la máquina virtual



Select a Guest Operating System
Which operating system will be installed on this virtual machine?

Guest operating system

☐ Microsoft Windows

☒ Linux

☐ VMware ESX

☐ Other

Version

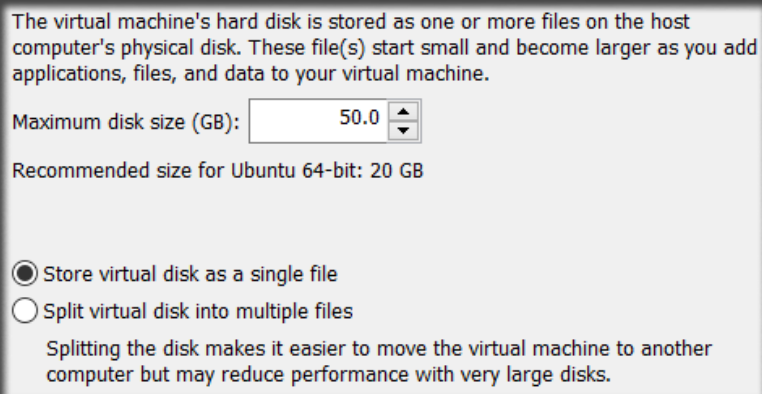
Ubuntu 64-bit

Virtual machine name:
logs_centralizacion

Location:
C:\Users\Érika\Documents\Virtual Machines\logs_centralizacion Browse...

The default location can be changed at Edit > Preferences.

Especificamos la capacidad del disco de la máquina virtual. Este tamaño depende del volumen de datos que queramos visualizar. En el caso de este documento, hemos escogido 50gb. También marcaremos la opción de que la máquina virtual se almacene en un sólo archivo para aumentar el rendimiento si vamos a manejar grandes cantidades de información.



The virtual machine's hard disk is stored as one or more files on the host computer's physical disk. These file(s) start small and become larger as you add applications, files, and data to your virtual machine.

Maximum disk size (GB): 50.0

Recommended size for Ubuntu 64-bit: 20 GB

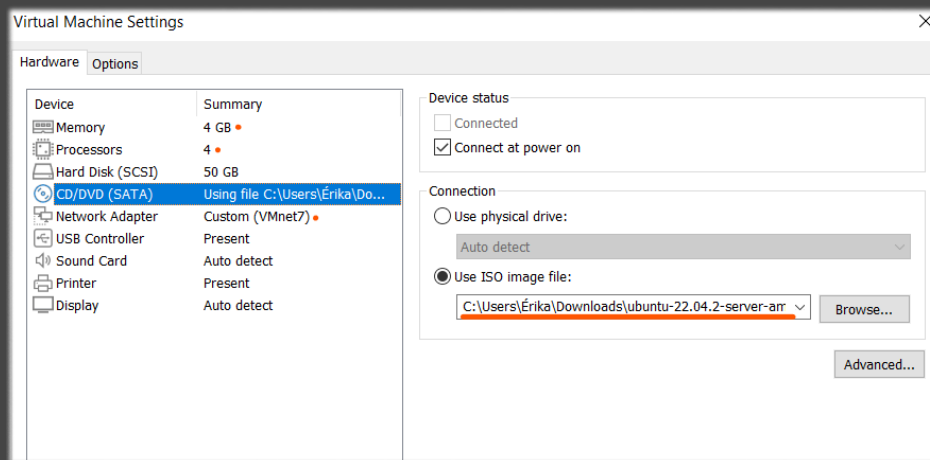
☒ Store virtual disk as a single file

☐ Split virtual disk into multiple files

Splitting the disk makes it easier to move the virtual machine to another computer but may reduce performance with very large disks.

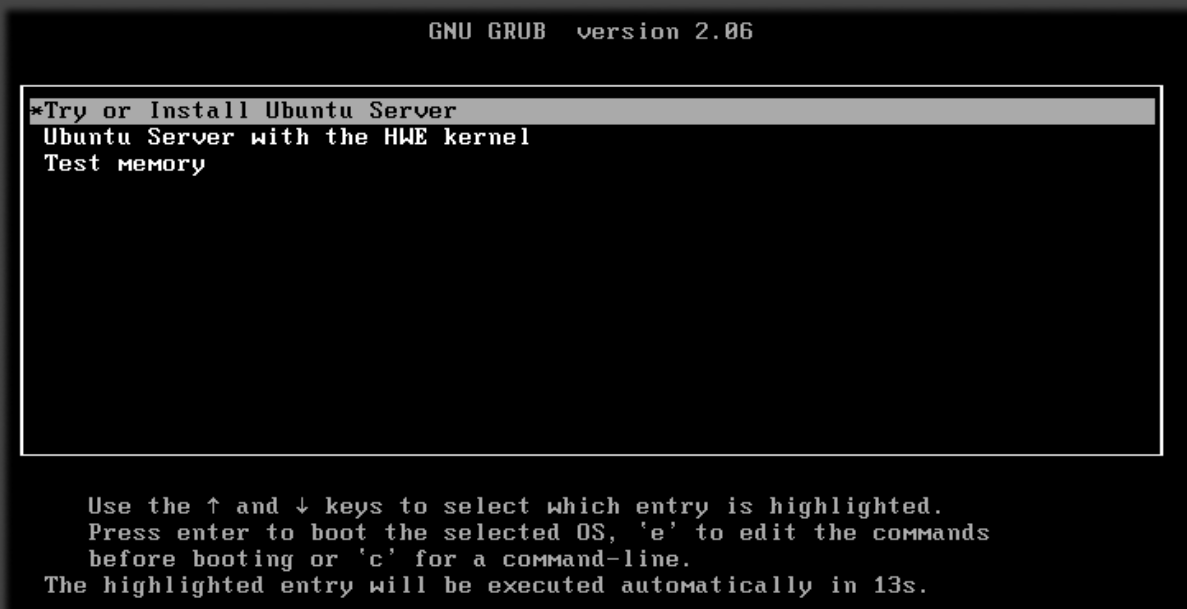
Una vez creada la máquina, nos iremos a “Virtual Machine Settings” , donde aplicaremos los requisitos del servidor (4GB de RAM, CPU de 4 núcleos, o 2 núcleos y 2

hilos, seleccionar la ISO de Ubuntu Server y la VMnet que hemos creado en la primera fase).



Cuando tengamos configurada la máquina virtual, vamos a abrirla para instalar el SO

Elegimos la primera opción



Instalaremos Ubuntu Server y, posteriormente, escogemos el protocolo DHCP para asignar una IP a nuestro sistema. Más adelante cambiaremos esta IP por una estática.

Choose type of install

Choose the base for the installation.

(X) Ubuntu Server

The default install contains a curated

Conexiones de red

Configure al menos una interfaz para suficiente para las actualizaciones.

NAME	TYPE	NOTES
[ens33	eth	-
DHCPv4	192.168.1.128/24	
00:0c:29:4e:b9:bb / Intel Corporati		
[Create bond ▶]		

El resto de apartados no los tocamos, dejaremos las opciones predeterminadas y asignaremos un nombre de usuario para acceder al sistema y el propio nombre del servidor. Una vez hecho esto, reiniciamos la máquina y aplicaremos la IP estática ejecutando los siguientes comandos

Antes de aplicar la dirección estática, vamos a ejecutar el siguiente comando para ver la puerta de enlace que nos está dando la red NAT. De esta forma, podremos tener acceso a internet

ip r

BASH

Una vez que hayamos identificado nuestra puerta de enlace, vamos a acceder al archivo de configuración de netplan para establecer una IP estática

sudo nano /etc/netplan/00-installer-config.yaml

BASH

Dentro de este archivo, copiaremos el siguiente código respetando el formato y definiendo los parámetros necesarios

```
# This is the network config written by 'subiquity'
network:
  ethernets:
    ens33:
#   dhcp4: true
    addresses:
      - 192.168.1.128/24 # ip estática / máscara de red
    nameservers:
      addresses: [ 208.67.222.222, 208.67.220.220 ]
    routes:
      - to: default
        via: 192.168.1.2 # puerta de enlace
  version: 2
```

BASH

Una vez que hayamos definido los parámetros dentro del archivo, ejecutaremos el siguiente comando para aplicar los cambios

```
sudo netplan apply
```

BASH

Por último, vamos a comprobar que los cambios se hayan guardado correctamente haciendo uso del siguiente comando

```
usuario@usuario:~$ ip r
default via 192.168.1.2 dev ens33 proto static
192.168.1.0/24 dev ens33 proto kernel scope link src 192.168.1.128
```

En este punto ya tendremos instalado y configurado el servidor.

Esta fase a concluido, ahora explicaremos como instalar las utilidades pertinentes

IMPLEMENTACIÓN DE PROGRAMAS

Ahora vamos a explicar en qué consisten los programas que vamos a utilizar:

- **Grafana:** Grafana es una plataforma de visualización y monitoreo de datos. Permite crear paneles interactivos y gráficos atractivos que muestran información en tiempo real sobre sistemas y aplicaciones. Esta aplicación nos permitirá visualizar el contenido de los logs, pues la podremos conectar con una o varias fuentes de datos.
- **Loki:** Loki es un sistema de almacenamiento de registros eficiente y escalable. En lugar de almacenar los registros completos, Loki almacena solo los índices y metadatos, lo que ayuda a reducir el espacio de almacenamiento requerido. Loki está diseñado para funcionar bien en entornos distribuidos y permite buscar y filtrar registros de manera eficiente.
- **Promtail:** Promtail es un agente de registro diseñado para recopilar y enviar registros al sistema de almacenamiento Loki. Promtail se instala en los servidores o máquinas donde se generan los registros y se encarga de recolectarlos de diferentes fuentes, como archivos de registro o mensajes del sistema, y enviarlos a Loki para su almacenamiento y análisis posterior. También se puede instalar en servidores remotos y tiene la capacidad de escuchar mensajes enviados a través del protocolo syslog (UDP 514)
- **Rsyslog:** Rsyslog es un programa nativo de Ubuntu que se utiliza para recopilar, procesar y enviar registros en sistemas operativos basados en Unix. Su función principal es gestionar y centralizar los registros generados por diferentes componentes de un sistema, como aplicaciones, servicios y eventos del sistema operativo y enviarlos a un destino centralizado para su almacenamiento y análisis

MANUAL DE INSTALACIÓN

A continuación, explicaremos el proceso de instalación y configuración de cada uno de los programas. De esta forma, lograremos crear el entorno de centralización y monitorización de Syslog remotos deseado:

GRAFANA

Para instalar Grafana en nuestro Ubuntu Server, debemos ejecutar los siguientes comandos en la terminal

```
sudo apt update
```

BASH

```
sudo apt upgrade
```

BASH

```
sudo apt install -y software-properties-common wget
```

BASH

```
sudo wget -q -O /usr/share/keyrings/grafana.key  
https://apt.grafana.com/gpg.key
```

BASH

```
sudo apt install -y apt-transport-https
```

BASH

Con esto habremos instalado las utilidades y autoridades certificadoras necesarias para el correcto funcionamiento de Grafana. El próximo paso será añadir los repositorios e instalar Grafana Server

```
echo "deb [signed-by=/usr/share/keyrings/grafana.key]  
https://apt.grafana.com stable main" | sudo tee -a  
/etc/apt/sources.list.d/grafana.list
```

BASH

```
echo "deb [signed-by=/usr/share/keyrings/grafana.key]  
https://apt.grafana.com beta main" | sudo tee -a  
/etc/apt/sources.list.d/grafana.list
```

BASH

Volvemos a actualizar para tener acceso a los nuevos repositorios e instalamos Grafana.

```
sudo apt update
```

BASH

```
sudo apt install grafana
```

BASH

```
sudo apt install grafana-enterprise
```

BASH

Ahora tenemos Grafana instalado como servicio. Podemos hacer que siempre se ejecute al inicio de la máquina virtual ejecutando los siguientes comandos

```
sudo /bin/systemctl daemon-reload
```

BASH

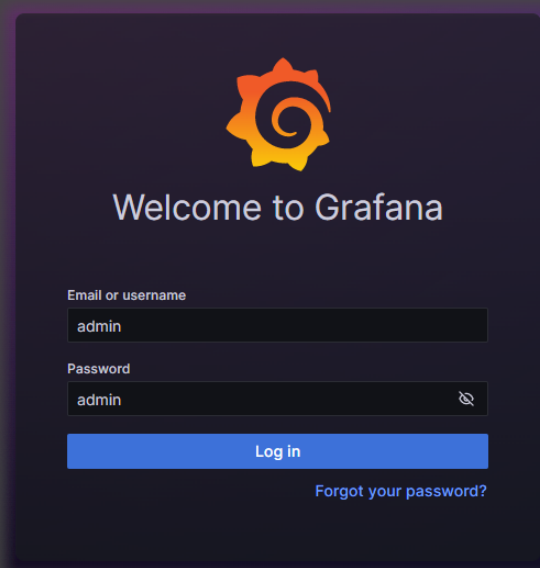
```
sudo /bin/systemctl enable grafana-server
```

BASH

```
sudo /bin/systemctl start grafana-server
```

BASH

De esta forma, ya tendremos Grafana funcionando en el puerto 3000. Sólo tendremos que introducir la ip de nuestro servidor junto al puerto 3000 en el buscador para acceder a la interfaz gráfica “http://<<ip>>:3000”



Usuario: admin
Contraseña: admin

Nota: Grafana solicitará un cambio de contraseña inmediatamente después de iniciar sesión con las credenciales predeterminadas

LOKI

Para instalar Loki, es necesario revisar los Assets de este [repositorio](#) de Github. En la fecha de creación de este documento (09/05/2023), la versión más reciente de loki es la 2.8.2.

Por otro lado, debemos tener en cuenta el sistema operativo en el que estamos trabajando. En el caso de este manual, se trata de un Ubuntu Server 22.04, por lo que debemos instalar el archivo “loki-linux-amd64.zip”

```
sudo wget  
https://github.com/grafana/loki/releases/download/v  
2.8.2/loki-linux-amd64.zip
```

BASH

```
sudo apt install unzip
```

BASH

```
sudo unzip loki-linux-amd64.zip
```

BASH

Con los comandos anteriores acabamos de descargar un archivo comprimido que contiene el ejecutable de Loki. Luego, hemos instalado la utilidad “unzip” para poder descomprimir el archivo. Posteriormente, lo hemos descomprimido en la ubicación donde hemos ejecutado el comando. Recomendando que esta ubicación se sitúe en una carpeta donde tengamos permisos de escritura y de ejecución.

Ahora, descargaremos el archivo de configuración de loki en la misma ubicación

```
sudo wget
https://raw.githubusercontent.com/grafana/loki/master/c
md/loki/loki-local-config.yaml
```

BASH

```
sudo nano loki-local-config.yaml
```

BASH

El comando “nano” nos permite editar un archivo de texto. En este caso, estamos editando el archivo de configuración de loki. Vamos a explicar los parámetros más relevantes y copiaremos el código de la siguiente página:

auth_enabled: false: Este parámetro especifica si se debe requerir autenticación para acceder al servidor.

server: Este bloque de configuración especifica opciones relacionadas con el servidor HTTP que el servidor Loki usa para aceptar conexiones entrantes. **http_listen_port** establece el puerto en el que el servidor escuchará conexiones HTTP

common: Este bloque de configuración establece opciones generales para el servidor. **instance_addr** establece la dirección IP a la que se unirán los componentes de Loki (en este caso, 127.0.0.1, solo se escucharán conexiones entrantes desde el host local). **replication_factor** especifica el número de réplicas que se deben almacenar para garantizar la redundancia y la disponibilidad de los datos, y en este caso, se establece en 1.

results_cache: establece opciones para la caché de resultados de consulta, y en este caso, se utiliza una caché incrustada que está habilitada y tiene un tamaño máximo de 100 MB.

schema_config: Este bloque de configuración especifica cómo se almacenan los datos de registro y los metadatos asociados. **configs** es una lista de objetos que especifican diferentes esquemas de almacenamiento para diferentes períodos de tiempo. En este caso, se usa una base de datos BoltDB para almacenar los datos de registro y un sistema de archivos local para almacenar los metadatos, y usa el esquema de índice “v11”.

```
auth_enabled: false

server:
  http_listen_port: 3100
  grpc_listen_port: 9096

common:
  instance_addr: 127.0.0.1
  path_prefix: /tmp/loki
  storage:
    filesystem:
      chunks_directory: /tmp/loki/chunks
      rules_directory: /tmp/loki/rules
  replication_factor: 1
  ring:
    kvstore:
      store: inmemory

query_range:
  results_cache:
    cache:
      embedded_cache:
        enabled: true
        max_size_mb: 100

schema_config:
  configs:
    - from: 2020-10-24
      store: boltdb-shipper
      object_store: filesystem
      schema: v11
      index:
        prefix: index_
        period: 24h

ruler:
  alertmanager_url: http://localhost:9093
```

BASH

ruler: Este bloque de configuración especifica opciones relacionadas con la regla de alerta en Loki. `alertmanager_url` especifica la URL del servidor Alertmanager que se usa para enviar alertas. En este caso, se usa una instancia de Alertmanager que se ejecuta en el mismo host que Loki, en el puerto 9093.

PROMTAIL

Siempre es esencial comprender lo que estamos haciendo para estar preparados ante posibles contratiempos al actualizar o modificar las utilidades explicadas en este documento. No obstante, es cierto que la mayor parte del peso y lo que realmente impulsa el funcionamiento de este proceso de centralización de archivos de registro son los programas que estamos a punto de presentar: Promtail y Rsyslog (este último en el primer caso).

Con esto quiero dejar claro que la configuración de Loki no es realmente relevante, Loki simplemente se encarga de recoger los archivos que Promtail procesa. Dicho esto, procederemos con la instalación de Promtail:

```
sudo wget
https://github.com/grafana/loki/releases/download/v
2.8.2/promtail-linux-amd64.zip
```

BASH

```
sudo unzip promtail-linux-amd64.zip
```

BASH

Hemos descargado el ejecutable de Promtail y lo hemos descomprimido en la misma carpeta donde acabamos de instalar y configurar Loki. Ahora descargaremos el archivo de configuración. Más adelante enseñaré como se debe definir este archivo para cada caso concreto

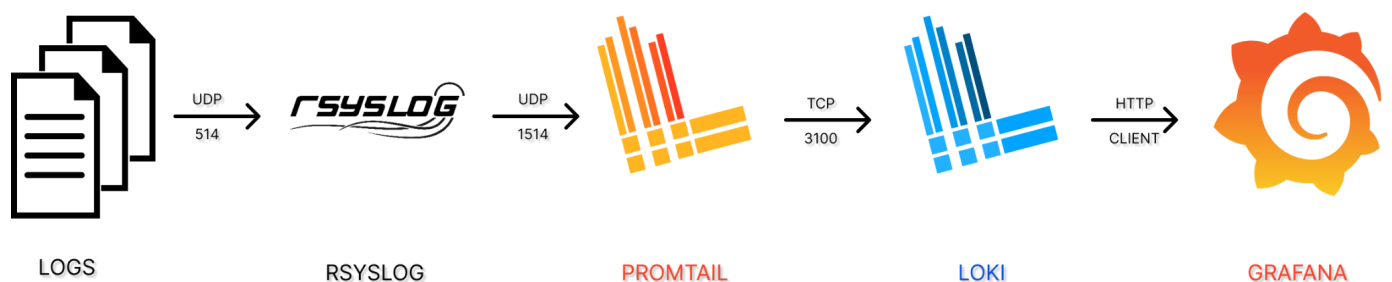
```
sudo wget  
https://raw.githubusercontent.com/grafana/loki/main/c  
lients/cmd/promtail/promtail-local-config.yaml
```

BASH

PRIMER CASO (SWITCH SYSLOG)

A continuación, explicaremos el recorrido que realizan los logs desde el switch de emisión hasta el programa de visualización, pasando por los programas de procesado que filtran y etiquetan cada archivo de registro.

Los logs serán enviados al servidor de Ubuntu de forma remota, el cual tendrá habilitada y configurada una utilidad “Rsyslog” para escuchar por el puerto 514 en formato UDP (puerto predeterminado para el protocolo syslog). Asimismo, Rsyslog reenviará la información recibida por el puerto 514 al puerto 1514 en formato UDP. Cabe destacar que Rsyslog tiene la capacidad de cambiar el formato de los logs al que nosotros queramos (por ejemplo, RFC 3164 > RFC 5424), lo cual es extremadamente útil para normalizar los distintos tipos de logs y evitar problemas de compatibilidad. La utilidad “Promtail” nos servirá para agregar las etiquetas que consideremos necesarias a los paquetes recibidos. Una vez se hayan modificado los logs, estos se enviarán a Loki a través de un cliente HTTP. Este programa es el encargado de crear un centro de recepción donde se recolectarán logs en distintos formatos y de diferentes fuentes. Finalmente, toda esta información procesada pasará a Kibana, donde podremos visualizar cómodamente la información contenida en los logs:



En la imagen anterior se puede apreciar un esquema del recorrido completo de los logs desde su dispositivo de origen hasta el programa de monitorización.

CONFIGURACIÓN DE PROMTAIL

Editamos el archivo de configuración y escribimos el siguiente código:

```
server:
  http_listen_port: 9080
  grpc_listen_port: 0

positions:
  filename: /tmp/positions.yaml

clients:
  - url: http://localhost:3100/loki/api/v1/push

scrape_configs:
  - job_name: syslog
    syslog:
      listen_address: 0.0.0.0:1514
      listen_protocol: udp
      labels:
        job: syslog
    relabel_configs:
      - source_labels: [__syslog_message_hostname]
        target_label: host
      - source_labels: [__syslog_message_hostname]
        target_label: hostname
      - source_labels: [__syslog_message_severity]
        target_label: level
      - source_labels: [__syslog_message_app_name]
        target_label: application
      - source_labels: [__syslog_message_facility]
        target_label: facility
      - source_labels: [__syslog_connection_hostname]
        target_label: connection_hostname
```

BASH

`server`: especifica la configuración del servidor que aloja Promtail, incluyendo el puerto que se utilizará para escuchar solicitudes HTTP (`http_listen_port`) y el puerto para escuchar solicitudes gRPC (`grpc_listen_port`).

`positions`: especifica el archivo donde se almacenan las posiciones de registro actuales de Promtail. En caso de que Grafana o Loki sufran una caída temporal, los archivos de registro quedarán grabados en esta ubicación y no se perderán.

`clients`: especifica la configuración de los clientes de Loki a los que Promtail enviará los registros recopilados. En este caso, se define un cliente con una URL de `http://localhost:3100/loki/api/v1/push`.

`scrape_configs`: especifica la configuración de los “raspadores” de registros que Promtail utilizará para recopilar. En este caso, se define un raspador con el nombre de trabajo `syslog`, que escucha en el puerto 1514 para registros syslog enviados por protocolo udp por parte de Rsyslog.

`relabel_configs`: especifica la configuración de las etiquetas de registro que se agregarán a los registros recopilados. En este caso, se agregan varias etiquetas de registro a partir de la información contenida en los registros syslog, como el `host`, `hostname`, `level`, `application`, `facility` y `connection_hostname`.

En términos generales, este archivo de configuración permitirá a Promtail escuchar por el puerto 1514 UDP y añadirá etiquetas adicionales a los archivos de registro para poder identificarlos y filtrarlos más fácilmente en Grafana.

RSYSLOG

Esta utilidad nativa de Ubuntu nos ayudará a formatear y redirigir los logs a Promtail. Tendremos que agregar un archivo de configuración dentro de la carpeta `/etc/rsyslog/conf.d/` que permita al servicio escuchar por el puerto 514 UDP (puerto predeterminado para la recepción de syslogs). Copiaremos el siguiente archivo de configuración y explicaremos sus parámetros:

```
cd /etc/rsyslog/conf.d
```

BASH

```
sudo nano 00-promtail-relay.conf
```

BASH

```
ruleset(name="remote"){
    *.* action(type="omfwd" protocol="tcp"
target="localhost" port="1514"
Template="RSYSLOG_SyslogProtocol23Format"
TCP_Framing="octet-counted" KeepAlive="on")
    *.* action(type="omfwd" protocol="udp"
target="localhost" port="1514"
Template="RSYSLOG_SyslogProtocol23Format")
}

#
https://www.rsyslog.com/doc/v8-stable/configuration/modules/imudp.html
module(load="imudp")
input(type="imudp" port="514" ruleset="remote")

#
https://www.rsyslog.com/doc/v8-stable/configuration/modules/imtcp.html
module(load="imtcp")
input(type="imtcp" port="514" ruleset="remote")
```

BASH

Tenemos tres secciones en este archivo de configuración: una regla definida como “remote”, un módulo de entrada UDP y un módulo de entrada TCP. ¿Qué hace exactamente cada sección?

La sección "remote" define dos reglas de envío de logs, una para el protocolo TCP y otra para UDP, ambas con la misma plantilla de formato "SyslogProtocol23Format". Ambas envían los mensajes a un servidor local en el puerto 1514 y utilizan el formato SyslogProtocol23. Este formateo de los archivos de registro es necesario para asegurar la compatibilidad de los logs con el procesado de Loki.

En general, los registros o logs generados por diferentes aplicaciones y sistemas pueden tener formatos muy diferentes, lo que puede dificultar el procesamiento y análisis de estos registros. Por lo tanto, es importante que se aplique un formato estándar a los logs antes de ser enviados a Loki para que puedan ser interpretados correctamente.

En este caso, la regla de formato utiliza el formato SyslogProtocol23 que es un formato ampliamente utilizado para los registros de Syslog, y que es compatible con Loki. Además, se ha especificado el tipo de framing para la transmisión TCP, que es "octet-counted", y se ha habilitado el KeepAlive para mejorar la conexión con el servidor Loki. Todo esto asegura que los registros lleguen a Loki en un formato estandarizado y en un estado confiable, lo que facilita el análisis y el procesamiento.

Para iniciar la nueva configuración, debemos guardar el archivo de configuración "Ctrl + O" y reiniciar el servicio Rsyslog:

```
sudo systemctl restart rsyslog.service
```

BASH

En este punto, Rsyslog ya debería estar funcionando y redirigiendo sin problemas. A continuación explicaremos el proceso de comprobación, como enviar syslog de prueba y como arreglar posibles errores.

En primer lugar, podemos comprobar el estado del servicio con el siguiente comando:

```
sudo systemctl status rsyslog.service
```

BASH

```
● rsyslog.service - System Logging Service
   Loaded: loaded (/lib/systemd/system/rsyslog.service; enabled; vendor preset: enabled)
   Active: active (running) since Thu 2023-05-11 09:39:45 UTC; 6s ago
     TriggeredBy: ● syslog.socket
       Docs: man:rsyslogd(8)
             man:rsyslog.conf(5)
             https://www.rsyslog.com/doc/
    Main PID: 2226 (rsyslogd)
      Tasks: 10 (limit: 2193)
     Memory: 1.6M
        CPU: 22ms
    CGroup: /system.slice/rsyslog.service
            └─2226 /usr/sbin/rsyslogd -n -iNONE

may 11 09:39:45 usuario systemd[1]: Starting System Logging Service...
may 11 09:39:45 usuario systemd[1]: Started System Logging Service.
may 11 09:39:45 usuario rsyslogd[2226]: imuxsock: Acquired UNIX socket '/run/systemd/journal/syslog' (fd 3) from systemd. [v8.2112.0]
may 11 09:39:45 usuario rsyslogd[2226]: rsyslogd's groupid changed to 113
may 11 09:39:45 usuario rsyslogd[2226]: rsyslogd's userid changed to 107
may 11 09:39:45 usuario rsyslogd[2226]: [origin software="rsyslogd" swVersion="8.2112.0" x-pid="2226" x-info="https://www.rsyslog.com"] start
```

Ahora comprobaremos que la redirección funciona. Para ello, utilizaremos el comando “tcpdump” para habilitar un sniffer en el puerto 1514 de la interfaz de red predeterminada:

```
sudo tcpdump -i any port 1514
```

BASH

```
documentacion@usuario:~/lok1$ sudo tcpdump -i any port 1514
tcpdump: data link type LINUX_SLL2
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on any, link-type LINUX_SLL2 (Linux cooked v2), snapshot length 262144 bytes
```

Después, enviaremos un log de prueba desde otro ordenador a la IP del servidor donde estamos instalando el centro de recepción de archivos de registro. En el caso de este manual, imitaremos el formato de log de un Cisco Catalyst Series 1000

```
echo "<133>1 2023-03-03T10:00:00.000Z MyComputerN
MyAppName 1234 - - Prueba" | nc -N -u 192.168.1.133 514
```

BASH

Pulsaremos “Enter” y comprobaremos si el sniffer a detectado el mensaje que acabamos de enviar

```
documentacion@usuario:~/loki$ sudo tcpdump -i any port 1514
tcpdump: data link type LINUX_SLL2
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on any, link-type LINUX_SLL2 (Linux cooked v2), snapshot length 262144 bytes
10:23:46.201600 lo In IP localhost.42150 > localhost.1514: Flags [S], seq 1755381165, win 65495, options [mss 65495,sackOK,TS val 1735586610
ecr 0,nop,wscale 7], length 0
10:23:46.201607 lo In IP localhost.1514 > localhost.42150: Flags [R.], seq 0, ack 1755381166, win 0, length 0
10:23:46.201755 lo In IP localhost.53975 > localhost.1514: UDP, length 73
```

Nota: Si el sniffer no ha detectado el log de prueba y pretende comprobar o cambiar la sintaxis del archivo de configuración, es **imprescindible** cambiar el mensaje antes de realizar la siguiente prueba. Rsyslog descarta por defecto todos los paquetes duplicados en la redirección para no saturar el tráfico, por lo que debe añadir algún carácter adicional al mensaje de prueba. Por ejemplo, si el campo de texto era "Prueba", se podría modificar a "Prueba1".

LOGS

Para terminar, explicaremos cómo visualizar los logs en Grafana. Lo primero será ejecutar Loki y Promtail con sus respectivos archivos de configuración. Para ello, nos posicionaremos en la carpeta donde tenemos instaladas las utilidades y los archivos de configuración y ejecutaremos los siguientes comandos:

```
./loki-linux-amd64 -config.file=loki-local-config.yaml
```

BASH

```
./promtail-linux-amd64 -config.file=promtail-local-config.yaml
```

BASH

Necesitaremos iniciar sesión en dos terminales para poder levantar ambas utilidades a la vez. Más adelante explicaremos algunas formas de optimizar el servidor, entre las cuales está convertir Loki y Promtail en servicios que se ejecutan automáticamente.

Lo siguiente que debemos hacer será comprobar el estado de Loki. Lo haremos introduciendo la siguiente URL en nuestro navegador:

```
http://<<localhost_ip>>:3100/ready
```

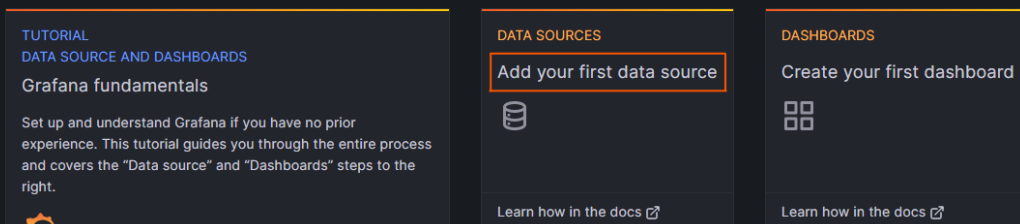
HTTP

Nos pedirá que esperemos 15 segundos a que el inyector esté listo. Pasado el tiempo, accederemos a Grafana en el navegador introduciendo la siguiente URL:

```
http://<<localhost_ip>>:3000/
```

HTTP

En la página principal veremos una opción llamada “add your first data source”



Buscamos “Loki” y lo seleccionamos:

Add data source

Choose a data source type

Q Loki



Loki

Like Prometheus but for logs.

Core

Introducimos la IP de loki en el apartado HTTP -> URL (`http://<localhost_ip>:3100/`):



Loki

Type: Loki

Settings



Configure your Loki data source below

Or skip the effort and get Loki (and Prometheus) as fully-managed, scalable, and hosted data sources from

Alerting supported

Name

Loki

Default



HTTP

URL

`http://192.168.1.133:3100/`

Allowed cookies

New tag (enter key to add)

Add

Timeout

Timeout in seconds

Auth

Basic auth



With Credentials



TLS Client Auth



With CA Cert



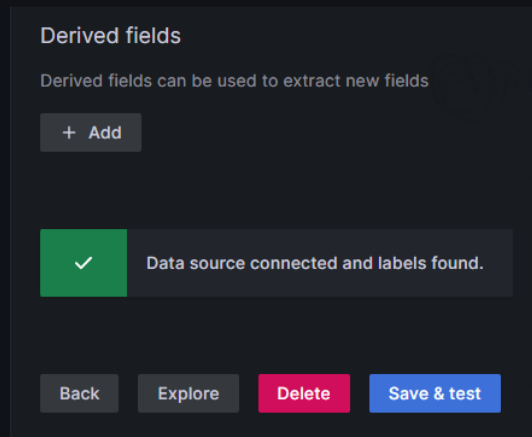
Skip TLS Verify



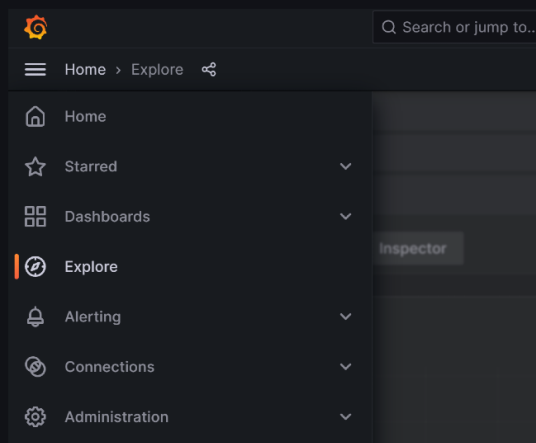
Forward OAuth Identity



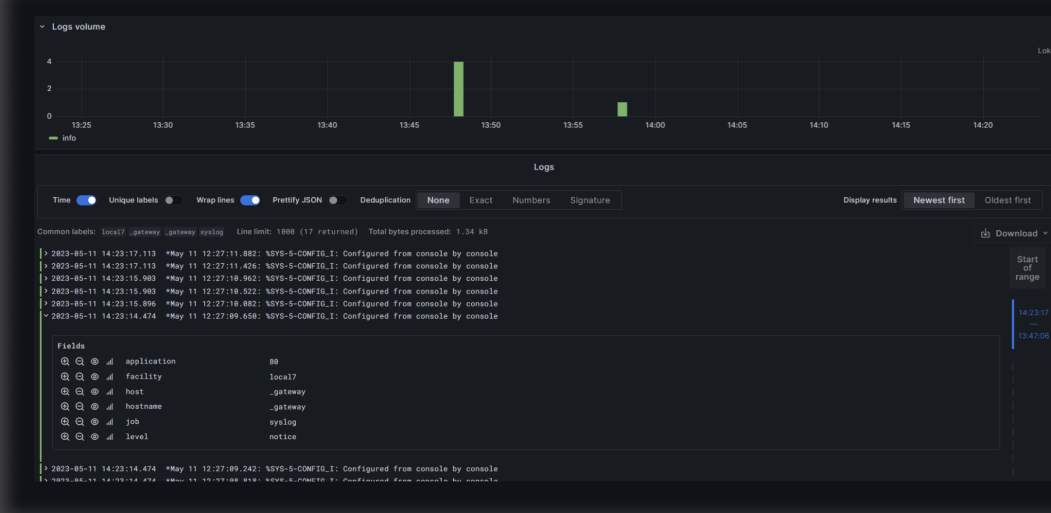
Enviamos unos cuantos logs para que Loki reciba las labels y, posteriormente, presionamos el botón “Save and test”.



Como podemos comprobar, Loki ha encontrado las label y está listo para ser usado. Volveremos a la página principal y nos iremos a “Explore”



De esta forma, ya podremos visualizar nuestros logs



MEJORAS

Este es el último punto de la documentación sobre cómo instalar y configurar Grafana Loki en un servidor Ubuntu 22.04 para centralizar syslogs (principalmente de switches Cisco). Aquí explicaremos algunas mejoras adicionales que recomendamos implementar para optimizar al máximo el funcionamiento de este sistema.

SERVICIOS (Loki & Promtail)

Durante la instalación de Loki y Promtail, mencionamos brevemente la posibilidad de ejecutar estas utilidades como servicios del sistema operativo para configurar un iniciado automático. A continuación, explicaremos como realizar dicho proceso:

Lo primero que debemos hacer es crear un usuario que tenga permisos para ejecutar el archivo “loki-linux-amd64”. Para ello, ejecutaremos los siguientes comandos

```
sudo adduser loki
```

BASH

Ahora, le daremos al usuario “loki” los permisos necesarios para ejecutar, parar y reiniciar el servicio que estamos a punto de crear. Hay varias formas de realizar este proceso, nosotros lo haremos modificando el archivo “visudo” (/etc/sudoers.tmp) y agregando la siguiente línea de código .

```
sudo visudo
```

BASH

```
loki ALL=(ALL) NOPASSWD: /bin/systemctl start loki.service,  
/bin/systemctl stop loki.service, /bin/systemctl restart  
loki.service
```

BASH

Guardamos el archivo (Ctrl + O) y volvemos a la terminal de comandos donde vamos a añadir el servicio “loki.service” a la ruta “/etc/systemd/system/” con el siguiente código:

```
sudo nano /etc/systemd/system/loki.service
```

BASH

```
[Unit]
Description=Loki service

[Service]
ExecStart=/home/loki/loki-linux-amd64
-config.file=/home/loki/loki-local-config.yaml
Restart=always
User=loki
Group=loki

[Install]
WantedBy=multi-user.target
```

BASH

El parámetro ExecStart indica la ruta que ejecutará el servicio. En este caso, se trata del ejecutable de loki “loki-linux-amd64”. De la misma forma, el parámetro config.file especifica la ruta del archivo de configuración con el cual se ejecutará el archivo definido en ExecStart.

Guardamos el archivo (Ctrl + O) y reiniciamos los demonios del sistema operativo para posteriormente iniciar el servicio Loki. Lo haremos con los siguientes comandos:

```
sudo systemctl daemon-reload
```

BASH

```
sudo systemctl start loki.service
```

BASH

En este punto, el servicio debería estar ejecutado correctamente. Podemos comprobar el estado del servicio como el siguiente comando

```
sudo systemctl status loki.service
```

BASH

```

• loki.service - Loki service
  Loaded: loaded (/etc/systemd/system/loki.service; disabled;
  Active: active (running) since Mon 2023-05-15 06:42:30 UTC;
  Main PID: 1782 (loki-linux-amd64)
  Tasks: 10 (limit: 2193)
  Memory: 17.5M
  CPU: 193ms
  CGroup: /system.slice/loki.service
          └─1782 /home/loki/loki-linux-amd64 -config.file=/home/

may 15 06:42:31 usuario loki-linux-amd64[1782]: level=info ts=2023-0
loki-linux-amd64[1782]: level=info ts=2023-05-15T06:42:31.076863761
may 15 06:42:31 usuario loki-linux-amd64[1782]: level=info ts=2023-0
may 15 06:42:31 usuario loki-linux-amd64[1782]: level=info ts=2023-0
may 15 06:42:31 usuario loki-linux-amd64[1782]: level=info ts=2023-0
may 15 06:42:31 usuario loki-linux-amd64[1782]: level=info ts=2023-0
may 15 06:42:31 usuario loki-linux-amd64[1782]: level=info ts=2023-0
may 15 06:42:34 usuario loki-linux-amd64[1782]: level=info ts=2023-0
loki-linux-amd64[1782]: level=info ts=2023-05-15T06:42:34.080854975
may 15 06:42:36 usuario loki-linux-amd64[1782]: level=info ts=2023-0
loki-linux-amd64[1782]: level=info ts=2023-05-15T06:42:36.083298164

```

Todo funciona perfectamente, pero lo realmente interesante es activar el iniciado automático cuando se inicie el sistema operativo. Para ello, ejecutaremos el siguiente comando:

```
sudo systemctl enable loki.service
```

BASH

Podemos hacer un reinicio rápido para comprobar que Loki se ejecuta al inicio del sistema operativo. Ahora, procederemos de la misma forma con Promtail. Lo primero será crear un usuario que tenga los permisos para ejecutar Promtail como servicio:

```
sudo adduser promtail
```

BASH

```
sudo visudo
```

BASH

```
promtail ALL=(ALL) NOPASSWD: /bin/systemctl start  
promtail.service, /bin/systemctl stop promtail.service,  
/bin/systemctl restart promtail.service
```

BASH

Una vez creado el usuario, añadiremos el servicio de la misma forma que hemos hecho con Loki

```
sudo nano /etc/systemd/system/promtail.service
```

BASH

```
[Unit]  
Description=Promtail service  
  
[Service]  
ExecStart=/home/loki/promtail-linux-amd64  
-config.file=/home/loki/promtail-local-config.yaml  
Restart=always  
User=loki  
#Group=loki  
  
[Install]  
WantedBy=multi-user.target
```

BASH

Reiniciamos los demonios del sistema operativo para posteriormente iniciar el servicio Promtail. Lo haremos con los siguientes comandos:

```
sudo systemctl daemon-reload
```

BASH

```
sudo systemctl start promtail.service
```

BASH

Comprobamos el estado del servicio, el cual funciona sin problemas:

```
● promtail.service - Loki service
   Loaded: loaded (/etc/systemd/system/promtail.service; disabled; ve
   Active: active (running) since Mon 2023-05-15 07:58:18 UTC; 5s ago
   Main PID: 1489 (promtail-linux-)
     Tasks: 9 (limit: 2193)
    Memory: 72.1M
       CPU: 302ms
    CGroup: /system.slice/promtail.service
            └─1489 /home/loki/promtail-linux-amd64 -config.file=/home/

may 15 07:58:18 usuario systemd[1]: Started Loki service.
may 15 07:58:19 usuario promtail-linux-amd64[1489]: level=info ts=2023-
may 15 07:58:19 usuario promtail-linux-amd64[1489]: level=info ts=2023-
may 15 07:58:19 usuario promtail-linux-amd64[1489]: level=info ts=2023-
may 15 07:58:19 usuario promtail-linux-amd64[1489]: level=info ts=2023-
may 15 07:58:19 usuario promtail-linux-amd64[1489]: level=warn ts=2023-
lines 1-16/16 (END)
```

Por último, activaremos el servicio para que se ejecute automáticamente al inicio del sistema.

```
sudo systemctl enable promtail.service
```

BASH

ALMACENAMIENTO PERSISTENTE

De manera predeterminada, Grafana reserva una porción de la memoria del servidor para almacenar archivos de registro. Esta funcionalidad resulta útil si deseamos eliminar los registros de días anteriores, ya que el historial se borrará cada vez que reiniciemos el servidor. No obstante, es recomendable tener una carpeta para respaldar los registros en caso de que Grafana experimente una caída temporal o sea necesario reiniciar el servidor. En esta sección, aprenderemos cómo guardar la información de los archivos de registro en la máquina local.

Procederemos modificando el archivo de configuración de Rsyslog destinado a la redirección del puerto 514 al 1514 (archivo dedicado a la recepción de logs). Para editarlo, vamos a introducir el siguiente comando en la terminal:

```
sudo nano 00-promtail-relay.conf
```

BASH

Dentro de la sección de reglas de redirección "remote", es necesario agregar una línea que redirija los paquetes a un archivo de texto antes de enviarlos al puerto 1514. Esto asegurará que la información de los archivos de registro se guarde en dicho archivo, independientemente de la capacidad de Grafana para visualizar los registros almacenados en su memoria.

La línea en concreto es [". " /ruta/de/ejemplo] (Debemos introducirla al principio del ruleset)

```
ruleset(name="remote"){
  *.* /var/log/syslogsloki
  *.* action(type="omfwd" protocol="tcp"
target="localhost" port="1514"
Template="RSYSLOG_SyslogProtocol23Format"
TCP_Framing="octet-counted" KeepAlive="on")
  *.* action(type="omfwd" protocol="udp"
target="localhost" port="1514"
Template="RSYSLOG_SyslogProtocol23Format")
}

#
https://www.rsyslog.com/doc/v8-stable/configuration/modules/imudp.html
module(load="imudp")
input(type="imudp" port="514" ruleset="remote")

#
https://www.rsyslog.com/doc/v8-stable/configuration/modules/imtcp.html
module(load="imtcp")
input(type="imtcp" port="514" ruleset="remote")
```

BASH

Es crucial asegurarse de que el archivo designado exista y sea accesible por el usuario que ejecuta el servicio de Rsyslog, el cual se denomina "syslog" y pertenece al grupo "adm". A continuación, se proporcionarán los comandos necesarios para crear el archivo especificado en el ejemplo anterior y ajustar su propiedad de manera adecuada.

Para crear el archivo:

```
sudo touch /var/log/syslogsloki
```

BASH

Para cambiar el usuario propietario del archivo

```
sudo chown syslog /var/log/syslogsloki
```

BASH

Para cambiar el grupo propietario del archivo

```
sudo chgrp adm /var/log/syslogsloki
```

BASH

Con el siguiente comando comprobaremos que los permisos se hayan cambiado correctamente

```
ls -l /var/log/syslogsloki
```

BASH

La salida debería mostrar el usuario “syslog” y el grupo “adm” como propietarios del archivo

```
-rw-r--r-- 1 syslog adm 2368 may 16 07:05  
/var/log/syslogsloki
```

BASH

Finalmente, reiniciamos el servicio de Rsyslog y enviaremos algunos registros para verificar que se están escribiendo en la carpeta que previamente hemos designado.

```
sudo systemctl restart rsyslog.service
```

BASH

Comprobamos el archivo “syslogsloki” después de enviar algunos archivos de registro

```
GNU nano 6.2 /var/log/syslogsloki
May 16 06:15:49_gateway 274: *May 16 06:19:24.836: %SYS-5-CONFIG-I: Configured from console by console
May 16 06:19:09_gateway 275: *May 16 06:21:44.256: %SYS-5-CONFIG-I: Configured from console by console
May 16 06:18:12_gateway 276: *May 16 06:21:45.312: %SYS-5-CONFIG-I: Configured from console by console
May 16 06:18:15_gateway 277: *May 16 06:21:45.736: %SYS-5-CONFIG-I: Configured from console by console
May 16 06:18:18_gateway 278: *May 16 06:21:46.192: %SYS-5-CONFIG-I: Configured from console by console
May 16 06:18:21_gateway 279: *May 16 06:21:46.648: %SYS-5-CONFIG-I: Configured from console by console
May 16 06:18:24_gateway 280: *May 16 06:21:47.088: %SYS-5-CONFIG-I: Configured from console by console
May 16 06:18:27_gateway 281: *May 16 06:21:47.536: %SYS-5-CONFIG-I: Configured from console by console
May 16 06:19:30_gateway 282: *May 16 06:21:48.032: %SYS-5-CONFIG-I: Configured from console by console
May 16 06:18:33_gateway 283: *May 16 06:21:48.535: %SYS-5-CONFIG-I: Configured from console by console
May 16 06:18:36_gateway 284: *May 16 06:21:49.048: %SYS-5-CONFIG-I: Configured from console by console
May 16 06:18:39_gateway 285: *May 16 06:21:49.552: %SYS-5-CONFIG-I: Configured from console by console
May 16 06:28:14_gateway 286: *May 16 06:31:49.925: %SYS-5-CONFIG-I: Configured from console by console
May 16 06:54:51_gateway 287: *May 16 06:58:26.839: %SYS-5-CONFIG-I: Configured from console by console
May 16 06:54:54_gateway 288: *May 16 06:58:28.207: %SYS-5-CONFIG-I: Configured from console by console
May 16 06:54:57_gateway 289: *May 16 06:58:29.279: %SYS-5-CONFIG-I: Configured from console by console
May 16 06:56:13_gateway 290: *May 16 06:59:49.700: %SPANTREE-2-BLOCK_BPDUGUARD: Received BPDU on port Gi1/0/15 with BPDU Guard enabled. Disabling port.
May 16 06:56:16_gateway 291: *May 16 06:59:49.700: %PM-4-ERR_DISABLE: bpduguard error detected on Gi1/0/15, putting Gi1/0/15 in err-disable state
May 16 06:56:19_gateway 292: *May 16 06:59:49.760: %SPANTREE-2-BLOCK_BPDUGUARD: Received BPDU on port Gi1/0/14 with BPDU Guard enabled. Disabling port.
May 16 06:56:22_gateway 293: *May 16 06:59:49.760: %PM-4-ERR_DISABLE: bpduguard error detected on Gi1/0/14, putting Gi1/0/14 in err-disable state
May 16 07:05:59_gateway 294: *May 16 07:09:35.220: %SYS-5-CONFIG-I: Configured from console by console
```

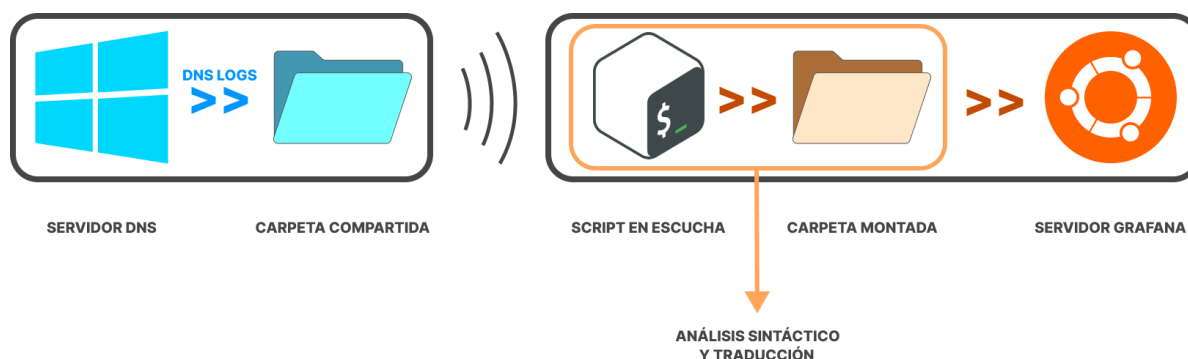
SEGUNDO CASO (WINDOWS DNS SERVER)

Windows Server 2016 cuenta con una función que permite volcar todos los registros de consultas DNS realizadas en una ubicación predefinida. Aprovecharemos esta herramienta para analizar y visualizar las actividades de resolución del sistema DNS instalado en Active Directory.

El sistema propuesto en este documento es simple y eficiente. Crearemos una carpeta compartida donde guardaremos un archivo de registro que contendrá todas las consultas y resoluciones DNS de nuestro servidor. Compartir esta carpeta permitirá que un servidor remoto pueda leer, traducir y visualizar este archivo de registro en un entorno donde podremos filtrar las solicitudes de manera personalizada.

Esto también se aplica a los archivos de registro DHCP, sin embargo, nos centraremos principalmente en los registros DNS, ya que necesitaremos desarrollar un script para traducir su formato. Esta medida resulta fundamental para comprender e interpretar de manera precisa las resoluciones de nombres realizadas por nuestro servidor. El script que crearemos se conoce en programación como un parser (analizador sintáctico). Una vez implementado el script, mostraré una comparación entre una línea de registro sin procesar y otra procesada.

Pero antes de considerar estos aspectos, debemos configurar Windows Server para volcar sus logs en una carpeta compartida que debemos crear. Windows Server será el dueño de esta carpeta y el servidor remoto accederá a ella como invitado. Más adelante explicaremos cómo securizar este proceso. A continuación, podremos observar un esquema del recorrido que realizan los logs para llegar al servidor remoto



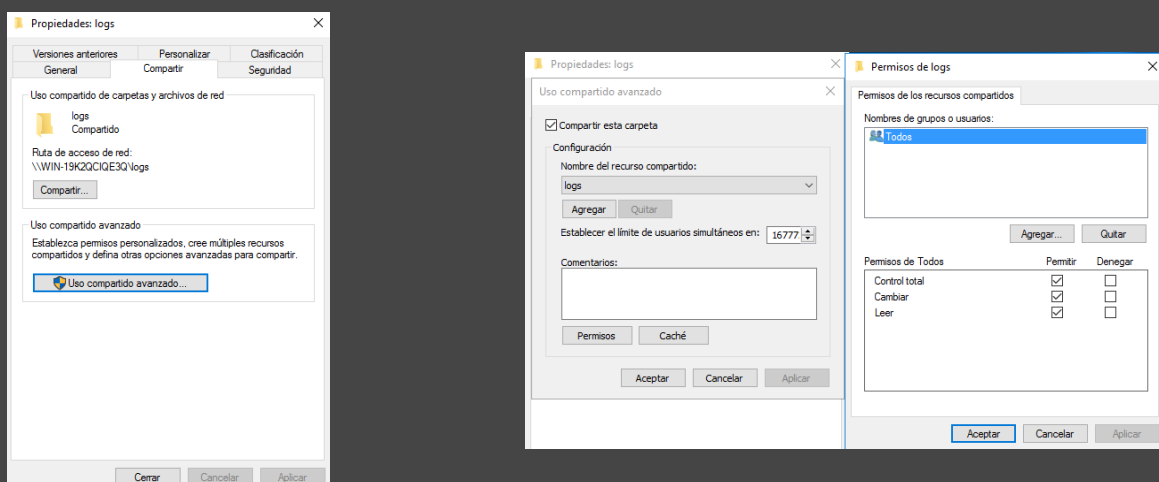
MÉTODO DE IMPLEMENTACIÓN

En este apartado, explicaremos la configuración necesaria de nuestros servidores para lograr los objetivos planteados previamente. Además, proporcionaremos instrucciones sobre los programas que utilizaremos y su correspondiente configuración. No nos adentraremos en los detalles de la instalación de estos, ya que dicho proceso está detalladamente documentado en el archivo de centralización de syslogs con Grafana Loki.

WINDOWS SERVER

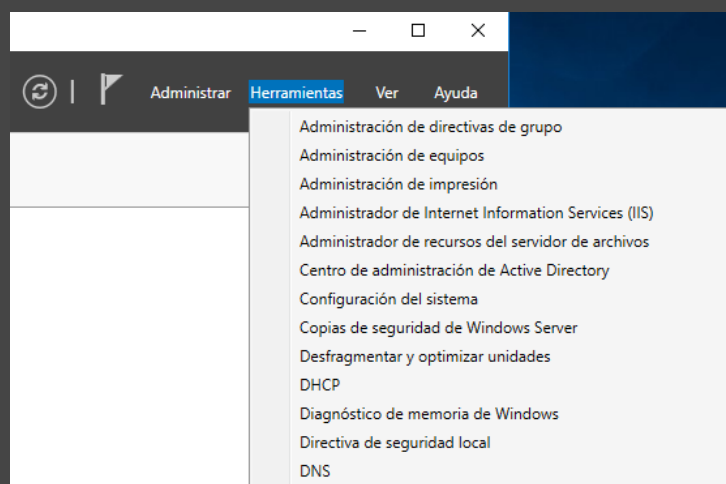
Con el propósito de no extender la documentación de forma innecesaria, voy a suponer que el Windows Server que actúa como servidor principal tiene instalada y configurada la utilidad DNS de Active Directory

Lo primero que debemos crear es la carpeta compartida. Le asignamos un nombre y nos vamos a propiedades. Una vez dentro, debemos posicionarnos en el apartado de “Compartir” y pulsaremos el botón “Uso compartido avanzado”

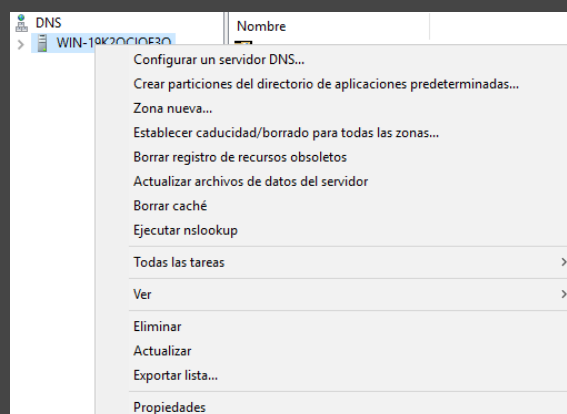


Aplicamos permisos totales a todos los usuarios. Como mencioné anteriormente, más adelante veremos cómo podemos securizar esta carpeta para que el acceso sea concedido exclusivamente a administradores y al equipo que sustenta el servidor de recepción de logs. Pulsamos en “Aplicar” y “Aceptar” y ya tendremos nuestra carpeta preparada. Ahora debemos configurar el servidor DNS para que mande sus logs a esta ruta

Para volcar los registros de consultas y resolución DNS del servidor, vamos a irnos a las herramientas del administrador del servidor y daremos click donde pone “DNS”

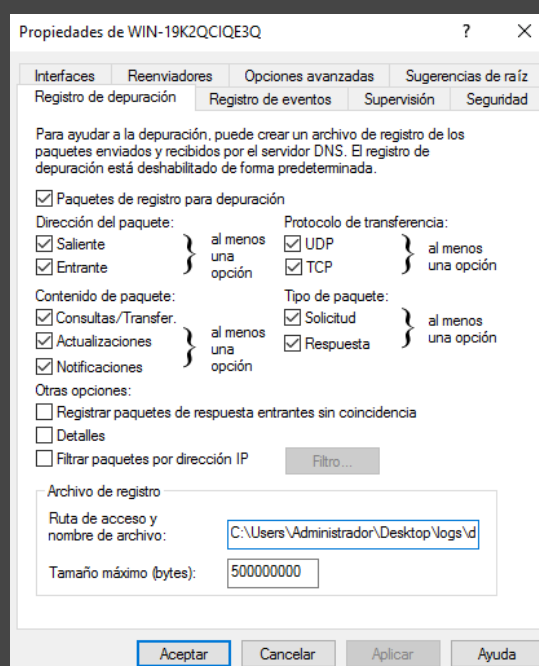


Una vez dentro, daremos click derecho en el servidor y seleccionaremos la opción de “propiedades”



Dentro de esta sección, debemos posicionarnos en el apartado de “Registro de depuración” y seleccionaremos las siguientes casillas. Después, vamos a especificar la ruta absoluta de la carpeta que acabamos de crear junto al nombre del archivo de registro que vayamos a crear. En mi caso, la he creado en el escritorio del usuario “Administrador” para que así solo pueda ser accesible por él y aquellos que tengan sus credenciales.

El nombre del archivo log puede ser cualquiera, en mi caso lo he llamado “debug”. Este archivo se creará automáticamente una vez configurado el apartado de registro de depuración



Pulsamos “Aplicar” y luego “Aceptar”. Ahora, vamos a resolver algunos nombres e IPs para ver si el archivo de registro que hemos creado está recogiendo las consultas

nslookup www.covapmario.local

CMD

Ejecutamos el comando anterior en la consola. Podemos sustituir el nombre “www.covapmario.local” por cualquier cosa, el log va a registrar cualquier consulta.

Comprobamos si el archivo de registro está recogiendo las consultas. Hay que aclarar que tarda un poco en registrar los logs, podéis hacer más consultas para acelerar un poco el proceso. En cualquier caso, vemos que funciona perfectamente

debug: Bloc de notas																
Archivo	Edition	Formato	Ver	Ayuda												
06/06/2023	9:42:51	0C84	PACKET	000001E618AA1830	UDP Snd ::1	0001 R Q [8281	DR SERVFAIL	PTR	(1)1(1)0(1)0(1)0(1)0(1)0(1)0(1)0(1)0(1)0(1)0(1)0(1)							
06/06/2023	9:42:51	0C84	PACKET	000001E618AA1830	UDP Snd ::1	0001 R Q [8281	DR SERVFAIL	PTR	(1)1(1)0(1)0(1)0(1)0(1)0(1)0(1)0(1)0(1)0(1)0(1)0(1)							
06/06/2023	9:43:07	1328	PACKET	000001E6274F51B0	UDP Rcv ::1	0001 Q [0001	D NOERROR	PTR	(1)1(1)0(1)0(1)0(1)0(1)0(1)0(1)0(1)0(1)0(1)0(1)0(1)							
06/06/2023	9:43:07	1328	PACKET	000001E627505AD0	UDP Snd 2001:500::f9::42	057b Q [0000	NOERROR	PTR	(1)1(1)0(1)0(1)0(1)0(1)0(1)0(1)0(1)0(1)0(1)0(1)0(1)							
06/06/2023	9:43:09	1328	PACKET	000001E6274ECD80	UDP Rcv ::1	0002 Q [0001	D NOERROR	A	(3)www(10)covapmario(5)local(10)covapmario(5)local(0)							
06/06/2023	9:43:09	1328	PACKET	000001E6274ECD80	UDP Snd ::1	0002 R Q [8385 A DR	NXDOMAIN	A	(3)www(10)covapmario(5)local(10)covapmario(5)local(0)							
06/06/2023	9:43:09	1328	PACKET	000001E618AA96E0	UDP Rcv ::1	0003 Q [0001	D NOERROR	AAAA	(3)www(10)covapmario(5)local(10)covapmario(5)local(0)							
06/06/2023	9:43:09	1328	PACKET	000001E618AA96E0	UDP Snd ::1	0003 R Q [8385 A DR	NXDOMAIN	AAAA	(3)www(10)covapmario(5)local(10)covapmario(5)local(0)							
06/06/2023	9:43:09	1328	PACKET	000001E627513720	UDP Rcv ::1	0004 Q [0001	D NOERROR	A	(3)www(10)covapmario(5)local(0)							
06/06/2023	9:43:09	1328	PACKET	000001E627513720	UDP Snd ::1	0004 R Q [8085 A DR	NOERROR	A	(3)www(10)covapmario(5)local(0)							
06/06/2023	9:43:09	1328	PACKET	000001E618AA1830	UDP Rcv ::1	0005 Q [0001	D NOERROR	AAAA	(3)www(10)covapmario(5)local(0)							
06/06/2023	9:43:09	1328	PACKET	000001E618AA1830	UDP Snd ::1	0005 R Q [8085 A DR	NOERROR	AAAA	(3)www(10)covapmario(5)local(0)							
06/06/2023	9:43:10	1328	PACKET	000001E618806990	UDP Rcv ::1	0001 Q [0001	D NOERROR	PTR	(1)1(1)0(1)0(1)0(1)0(1)0(1)0(1)0(1)0(1)0(1)0(1)0(1)							
06/06/2023	9:43:12	0C84	PACKET	000001E627505AD0	UDP Snd 2001:dc3::35	057b Q [0000	NOERROR	PTR	(1)1(1)0(1)0(1)0(1)0(1)0(1)0(1)0(1)0(1)0(1)0(1)0(1)							

Sabemos que VM-DC1 (Servidor DNS de COVAP) tiene reenviadores configurados (a diferencia del ejemplo anterior). Por tanto, vamos a insertar las IPs de OpenDNS en los reenviadores para comprobar si las consultas que estos resuelven se registran también en el archivo que hemos creado

Una vez introducidos, vamos a resolver el nombre de `www.google.es`

nslookup www.google.es

CMD

06/06/2023	9:49:16	1328	PACKET	000001E618AB9F40	UDP	Snd	208.67.222.222	0437	Q	[0001	D	NOERROR	A	(3)www(6)google(2)es(0)
06/06/2023	9:49:16	1328	PACKET	000001E61C75D0C0	UDP	Rcv	208.67.222.222	0437	R	Q [8081	DR	NOERROR	A	(3)www(6)google(2)es(0)
06/06/2023	9:49:16	1328	PACKET	000001E618AA96E0	UDP	Snd	::1	0004	R	Q [8081	DR	NOERROR	A	(3)www(6)google(2)es(0)
06/06/2023	9:49:16	1328	PACKET	000001E618AB4720	UDP	Rcv	::1	0005	Q	[0001	D	NOERROR	AAAA	(3)www(6)google(2)es(0)
06/06/2023	9:49:16	1328	PACKET	000001E618AA96E0	UDP	Snd	208.67.222.222	84fb	Q	[0001	D	NOERROR	AAAA	(3)www(6)google(2)es(0)
06/06/2023	9:49:16	1328	PACKET	000001E61CFC6520	UDP	Rcv	208.67.222.222	84fb	R	Q [8081	DR	NOERROR	AAAA	(3)www(6)google(2)es(0)

UBUNTU SERVER

Vamos a instalar un Ubuntu Server 22.04 en una máquina virtual de VMWare. Haremos uso de la utilidad CIFS (Common Internet File System), la cual es una implementación del protocolo de red SMB (Server Message Block) utilizado para compartir archivos y recursos en redes locales.

Ejecutaremos los siguientes comandos en la terminas para actualizar el sistema

```
sudo apt update
```

BASH

```
sudo apt upgrade
```

BASH

Una vez hecho esto, vamos a instalar CIFS ejecutando el siguiente comando en la terminal

```
sudo apt install cifs-utils
```

BASH

Las siguientes utilidades que vamos a instalar son Loki, Promtail y Grafana. Todo el proceso de instalación está documentado en este [documento](#). A continuación, vamos a explicar como debemos configurar cada utilidad para leer el archivo de registro de la carpeta compartida

En el caso de Loki, debemos escribir su archivo de configuración de la siguiente manera:


```
auth_enabled: false

server:
  http_listen_port: 3100
  grpc_listen_port: 9096

common:
  instance_addr: 127.0.0.1
  path_prefix: /tmp/loki
  storage:
    filesystem:
      chunks_directory: /tmp/loki/chunks
      rules_directory: /tmp/loki/rules
  replication_factor: 1
  ring:
    kvstore:
      store: inmemory

query_range:
  results_cache:
    cache:
      embedded_cache:
        enabled: true
        max_size_mb: 100

schema_config:
  configs:
    - from: 2020-10-24
      store: boltdb-shipper
      object_store: filesystem
      schema: v11
      index:
        prefix: index_
        period: 24h

ruler:
  alertmanager_url: http://localhost:9093
```

BASH

Una vez terminado el archivo de configuración de Loki, vamos a configurar el de Promtail.

Simplemente debemos especificar el archivo que va a leer Promtail y posteriormente enviar a Loki. Vamos a seguir la siguiente sintaxis para definir dicho archivo

```
server:
  http_listen_port: 9080
  grpc_listen_port: 0

positions:
  filename: /tmp/positions.yaml

clients:
  - url: http://localhost:3100/loki/api/v1/push

scrape_configs:
- job_name: system
  static_configs:
    - targets:
      - localhost
    labels:
      job: varlogs
      __path__: /home/logswindows/dnslogs/logs
```

BASH

En la sección “__path__” escribiremos la ruta del archivo de registro que queramos que interprete Promtail. Más adelante explicaremos de dónde proviene el archivo mostrado en el ejemplo anterior.

Por último, necesitamos configurar Loki y Promtail como servicios de Ubuntu (proceso que también está documentado en la guía de centralización de logs con Grafana Loki)

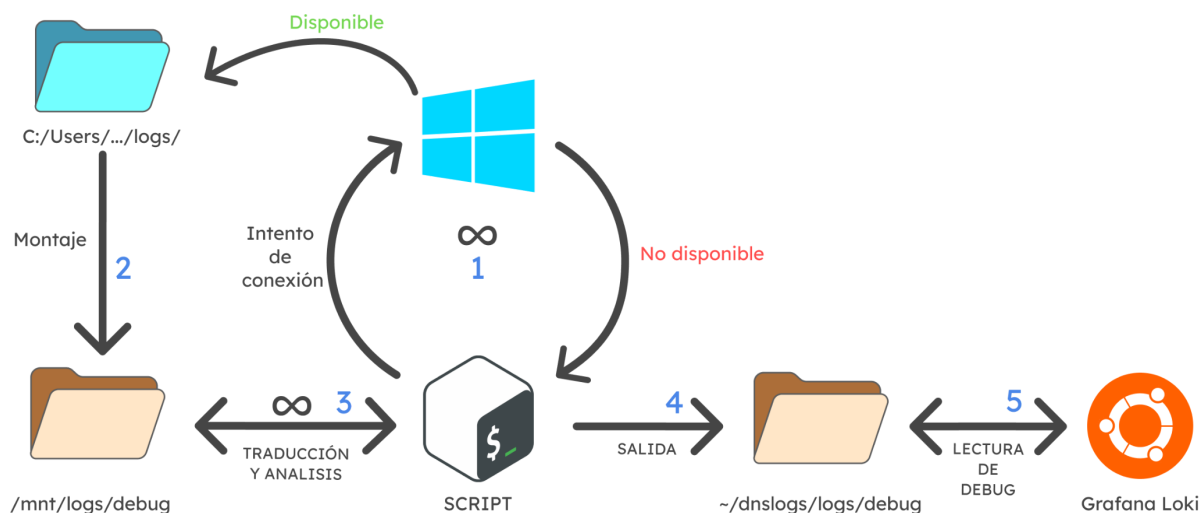
Una vez instaladas y configuradas todas las utilidades pertinentes, vamos a explicar la ruta que recorren los logs y cómo funciona el analizador sintáctico que debemos utilizar para formatear los archivos de registro

SCRIPT

Como comenté al principio del documento, la creación de este script es fundamental para adaptar el archivo de registro a un formato más legible y visual. De la misma forma, necesitaremos el script para garantizar la conectividad entre Ubuntu y Windows, puesto que este será el encargado de establecer la conexión a la carpeta compartida por parte del servidor remoto.

Por tanto, podemos concluir que el script que vamos a crear se dividirá en dos partes fundamentales; el montaje de la carpeta compartida y el analizador sintáctico.

Primero, el script estará constantemente intentando establecer conexión con Windows (1) para montar la carpeta “logs” en la ruta /mnt/logs (2). Una vez montada, el script entrará en un bucle infinito mediante el cual estará leyendo las líneas del archivo de registro (3). Cuando haya leído todas las líneas, creará un archivo con la traducción pertinente (4) y permanecerá a la espera de nuevas líneas. Durante todo este proceso, Promtail estará leyendo el archivo de traducción y enviándolo a Loki para ser visualizado en Grafana (5). Esto sería un esquema de todo el proceso:



Aquí podéis ver el código del [script](#). De todas formas, entraremos en profundidad en cómo funciona cada una de sus partes y explicaremos línea por línea los parámetros y comandos utilizados

MONTAJE DE LA CARPETA COMPARTIDA

Antes de explicar cómo funciona el código, vamos a hablar de la utilidad CIFS y cómo la estamos implementando en este manual.

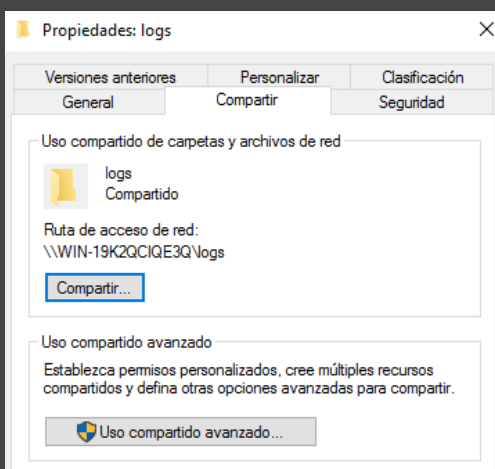
El comando que vamos a usar para montar la carpeta es el siguiente

```
sudo mount -t cifs -o credentials=/etc/win-credentials  
//192.168.1.137/logs /mnt/logs
```

BASH

Con este comando estamos especificando que vamos a montar el archivo `//192.168.1.137/logs` en la carpeta `/mnt/logs` utilizando el sistema de archivos CIFS y las credenciales contenidas en la carpeta `win-credentials`

Para empezar, la cadena que contiene la IP `192.168.1.137` (IP del servidor DNS) está apuntando a la carpeta que previamente hemos compartido en Windows. La ruta de la carpeta compartida podemos obtenerla de la pestaña “Compartir” dentro de la sección “Propiedades” de la carpeta `logs`



Sustituimos el nombre “WIN-19K2QCIQE3Q” por la IP de nuestro servidor para que Ubuntu pueda acceder a Windows a través de la red.

La ruta /mnt/logs define la carpeta donde se montará el contenido de la carpeta compartida de Windows.

Por último, vamos a explicar el parámetro “credentials=/etc-wincredentials”. Este parámetro apunta a un archivo que contiene las credenciales de un usuario con permisos para acceder a la carpeta compartida. En el caso de este manual, hemos puesto las credenciales del administrador. Repito, más adelante veremos cómo securizar esta conexión para no comprometer la integridad de los usuarios al escribir sus credenciales en un archivo de texto plano:

Debemos crear un archivo que tenga las credenciales del administrador del servidor DNS. En mi caso:

```
username=Administrador  
password=Usuario.  
domain=192.168.1.137
```

BASH

Recomiendo ejecutar el comando por separado antes de empezar a probar el script para comprobar si la carpeta se monta correctamente

Ahora, vamos a escribir toda la parte del código relacionada con el montaje de la carpeta y vamos a explicar qué hace cada comando.

```
# Intentos de montar la carpeta compartida
share_mounted=false
retries=0
max_retries=10000
delay=5 # segundos entre cada intento de montaje
if findmnt | grep -q "/mnt/logs"; then
    share_mounted=true
fi
while [ "$share_mounted" = false ] && [ "$retries" -lt "$max_retries" ];
do
    sudo mount -t cifs -o credentials=/etc/win-credentials
//192.168.1.137/logs /mnt/logs
    if [ $? -eq 0 ]; then
        share_mounted=true
    else
        retries=$((retries+1))
        sleep "$delay"
    fi
done
if [ "$share_mounted" = true ]; then
fi
```

BASH

Primero, establecemos la variable “share_mounted” en “false”. Se utilizará para verificar si la carpeta compartida está montada correctamente.

Se establecen las variables “retries”, “max_retries” y “delay” para controlar el proceso de montaje de la carpeta. De esta forma, evitamos saturar el servidor.

Si la carpeta compartida está montada en “/mnt/logs” aparecerá en la salida del comando “findmnt”. Esto hará que el script omita el bucle de intentos de conexión en caso de que se haya establecido conexión con la carpeta compartida

A continuación, encontraremos el bucle “while”, el cual se repetirá hasta que la variable “share_mounted” sea igual a “true” y el número de intentos sea menor al máximo permitido. Si el comando “sudo mount -t cifs -o credentials=/etc/win-credentials //192.168.1.137/logs /mnt/logs” no emite ninguna salida (`$? -eq 0`) significa que se ha ejecutado correctamente y la variable “share_mounted” cambiará su valor a “true”. De lo contrario, se sumará “1” al número de intentos y se esperarán los segundos que defina la variable “delay” antes de volver a entrar en el bucle

ANALIZADOR SINTÁCTICO

Este es el código perteneciente al analizador sintáctico contenido en el script

PARSER (1/3)

```

if [ "$share_mounted" = true ]; then
    # Ruta del archivo de depuración DNS
    archivo_entrada="/mnt/logs/debug"

    # Obtener la última posición procesada desde el archivo de marca
    ultima_cantidad_lineas=$(cat "$archivo_marca" 2>/dev/null)

    # Comprobar si el archivo de marca existe, si no, crearlo con el valor inicial de 0
    if [ ! -e "$archivo_marca" ]; then
        echo 0 > "$archivo_marca"
    fi

    # Verificar si ultima_cantidad_lineas está vacío y asignar un valor predeterminado
    if [ -z "$ultima_cantidad_lineas" ]; then
        ultima_cantidad_lineas=0
    fi

    while true; do
        if [ -e "$archivo_entrada" ]; then
            # Obtener la cantidad actual de líneas en el archivo
            cantidad_lineas=$(wc -l < "$archivo_entrada")

            # Verificar si hay nuevas líneas agregadas al archivo
            if [ "$cantidad_lineas" -gt "$ultima_cantidad_lineas" ]; then
                # Leer las líneas nuevas desde la última posición procesada
                sed -n "$((ultima_cantidad_lineas+1)),\$p" "$archivo_entrada" | while IFS=
                read -r linea; do
                    # Procesar la línea
                    campos=($linea)
                    num_campos=${#campos[@]}
                    if [ "$num_campos" -lt 2 ]; then
                        continue
                    else
                        caso=$(echo "$linea" | awk '{print $(NF-2)}')
                        registro=$(echo "$linea" | awk '{print $(NF-1)}')
                        ip2=$(echo "$linea" | awk '{print $(NF)}')
                    fi
                done
            fi
        fi
    done

```

PARSER (2/3)

[illegible]

PARSER (3/3)

```

        case $accionnuevo in
            "Envio") registro_modificado="$(printf "Fecha %-8s Hora %-8s
Resolucion %-19s Respuesta %-20s Accion %-10s Formato %-6s Origen %-20s Destino
%s\n" "$fecha" "$hora" "$registronuevo" "$respuesta" "$accionnuevo" "$formato"
"$nuevaip" "$nuevaip2")";;
            "Respuesta") registro_modificado="$(printf "Fecha %-8s Hora %-8s
Resolucion %-19s Respuesta %-20s Accion %-10s Formato %-6s Origen %-20s Destino
%s\n" "$fecha" "$hora" "$registronuevo" "$respuesta" "$accionnuevo" "$formato"
"$nuevaip2" "$nuevaip")";;
        esac

        echo "$registro_modificado" | sed -e 's/^M//g' >> "$archivo_salida"

    fi
done

# Actualizar la última posición procesada y la cantidad de líneas
ultima_cantidad_lineas="$cantidad_lineas"

# Guardar la última posición procesada en el archivo de marca
echo "$ultima_cantidad_lineas" > "$archivo_marca"
fi
fi
sleep 1
done
else
    echo "Error: No se pudo montar la carpeta compartida después de $max_retries
intentos."
fi

```

A continuación, vamos a enumerar y explicar las funcionalidades de este analizador sintáctico

ELIMINAR LÍNEAS VACÍAS

Este script tiene la capacidad de discriminar las líneas leídas por número de campos. Si una línea no tiene los suficientes campos, no será procesada. Esto se hace con el objetivo de eliminar las instrucciones iniciales del script (las cuales, en teoría, ayudan a interpretar los archivos de registro) y las líneas vacías.

Este es el fragmento de código que define la función:

```
# Procesar la línea
campos=($línea)
num_campos=${#campos[@]}
if [ "$num_campos" -lt 2 ]; then
    continue
else
```

BASH

Si observamos la variable “campos” en el contexto del código nos daremos cuenta de que está dentro de un bucle destinado a leer un archivo línea por línea (este archivo se define cuando “sahre_mounted” pasa a ser “true” y es el archivo “debug” contenido en “/mnt/logs/”).

Por tanto, sabemos que “campos” hace referencia a la línea que el bucle está leyendo en ese momento. Teniendo esto en cuenta, veremos que la siguiente variable está contando los campos que hay en una línea “num_campos”. La condición siguiente dice que si el número de campos de la línea que se está leyendo en ese momento es menor que “2”, entonces el bucle de lectura debe continuar a la siguiente línea, descartando la actual.

TRADUCCIONES

Los campos más relevantes de las líneas de los archivos de registro DNS son las “acciones”, los “registros” y los “casos”.

Las acciones hacen referencias al tipo de intercambio de paquetes que se está realizando (Snd = Envío ; Rcv = Respuesta).

Los registros se refieren al tipo de resolución que está teniendo lugar. El formato predeterminado de los registros es muy críptico e ilegible, así que me he encargado de traducir todos las posibilidades:

```
case $registro in
  "AAAA") registronuevo="RESOLUCION IPV6";;
  "A") registronuevo="RESOLUCION IPV4";;
  "PTR") registronuevo="RESOLUCION INVERSA";;
  "SOA") registronuevo="INICIO DE AUTORIDAD";;
  "SRV") registronuevo="REGISTRO DE SERVICIO";;
  "SPF") registronuevo="REGISTRO DE POLITICA DE ENVIO";;
esac
```

BASH

Los casos son el estado en el que ha concluido la resolución. De nuevo, me he encargado de traducir todos los casos posibles:

```
case $caso in
  "NOERROR") respuesta="EXITO";;
  "NXDOMAIN") respuesta="NO ENCONTRADO";;
  "SERVFAIL") respuesta="FALLO EN EL SERVIDOR";;
  "NOTIMP") respuesta="NO COMPATIBLE";;
  "REFUSED") respuesta="RECHAZADO";;
  "FORMERR") respuesta="FORMATO ERRONEO";;
  *) respuesta="DESCONOCIDO";;
esac
```

BASH

DEFINIR ORIGEN Y DESTINO

Los logs DNS de Windows siempre van a mostrar primero la dirección local, lo cual supone que la IP de destino y la IP de origen difieran (a pesar de estar siempre en la misma posición) dependiendo del tipo de paquete que se ha intercambiado.

Este detalle también lo he solucionado. El script mostrará la IP de origen y la IP de destino dependiendo del tipo de acción que se esté realizando:

```
case $accionnuevo in
    "Envio") registro_modificado="$(printf "Fecha %-8s Hora %-8s
Resolucion %-19s Respuesta %-20s Accion %-10s Formato %-6s
Origen %-20s Destino %s\n" "$fecha" "$hora" "$registronuevo"
"$respuesta" "$accionnuevo" "$formato" "$nuevaip"
"$nuevaip2")";;
    "Respuesta") registro_modificado="$(printf "Fecha %-8s Hora
%-8s Resolucion %-19s Respuesta %-20s Accion %-10s Formato
%-6s Origen %-20s Destino %s\n" "$fecha" "$hora"
"$registronuevo" "$respuesta" "$accionnuevo" "$formato"
"$nuevaip2" "$nuevaip")";;
esac
```

BASH

ESPERAR NUEVAS LÍNEAS

Como he mencionado varias veces, este script es un bucle infinito que está constantemente leyendo el archivo “debug” en busca de nuevos registros. Esto lo consigue mediante el uso de una condición que nunca puede ser falsa (“while true; do”). De esta manera, no habrá forma de romper el bucle.

Sin embargo, los bucles infinitos pueden ser configurados con un control de flujo para que no saturen el servidor. Esto se puede conseguir mediante el uso del parámetro “sleep”, el cual define los segundos que debe esperar el script antes de saltar a la siguiente línea de código. En nuestro caso, hemos definido que el script espere un segundo antes de volver a leer el archivo de texto (“sleep 1”).

Un bucle infinito es realmente sencillo de crear y controlar. Ahora voy a explicar como he implementado la capacidad de esperar líneas nuevas.

Primero, debemos definir la variable “archivo_marca” fuera del bucle. Esta variable apunta a un archivo el cual almacenará el número de líneas que ha leído el script. Al definirlo fuera del bucle, el conteo no se reiniciará. Cuando el script apruebe la comprobación de la variable “share_mounted”, iniciará una verificación de existencia de archivo de marca.

En caso de que no exista, el script lo creará con el valor inicial “0”. Por otro lado, si el archivo existe, otra variable leerá su contenido para tener constancia de la última cantidad de líneas leída “ultima_cantidad_lineas”. Si esta variable está vacía, se le aplicará el valor 0.

```
# Ruta del archivo de salida modificado
archivo_salida="/home/logswindows/dnslogs/logs"

# Archivo de marca para recordar la última posición procesada
archivo_marca="/home/logswindows/dnslogs/last_processed_position"

# Resto de código ...

if [ "$share_mounted" = true ]; then
    # Ruta del archivo de depuración DNS
    archivo_entrada="/mnt/logs/debug"

    # Obtener la última posición procesada desde el archivo de marca
    ultima_cantidad_lineas=$(cat "$archivo_marca" 2>/dev/null)

    # Comprobar si el archivo de marca existe, si no, crearlo con el valor inicial de 0
    if [ ! -e "$archivo_marca" ]; then
        echo 0 > "$archivo_marca"
    fi

    # Verificar si ultima_cantidad_lineas está vacío y asignar un valor predeterminado
    if [ -z "$ultima_cantidad_lineas" ]; then
        ultima_cantidad_lineas=0
    fi
```

BASH

Esto sería todo lo que ocurre fuera del bucle. Hay algunas funciones que no he explicado porque entran dentro del apartado en el que configuramos este script como un servicio de Ubuntu

Ahora vamos a explicar como trabaja esta función de guardar líneas dentro del bucle

Una vez dentro de “while true”, veremos que se verifica la existencia del “archivo_entrada”. Después, una variable contará las líneas de este archivo “cantidad_lineas”. Si la cantidad de líneas es mayor que la última cantidad de líneas procesadas “if [“\$cantidad_lineas” -gt “\$ultima_cantidad_lineas”]; then” entonces se utiliza el comando “sed” para leer las líneas nuevas desde la última posición procesada.

Cuando termina el proceso de traducción (es decir, el resto del código hasta que llegamos a la próxima línea relacionada con la función de esperar líneas nuevas) se actualiza la última posición procesada igualando la variable “ultima_cantidad_lineas” a “cantidad_lineas”

esta misma posición se guardará también en el archivo de marca.

Una vez terminado este proceso, en la próxima vuelta del bucle el script sabrá por dónde empezar y así no repetirá las líneas que ha procesado anteriormente

```
    echo "$registro_modificado" | sed -e 's/^M//g' >>
"$archivo_salida"

fi
done

# Actualizar la última posición procesada y la cantidad
de líneas
ultima_cantidad_lineas="$cantidad_lineas"

# Guardar la última posición procesada en el archivo de
marca
echo "$ultima_cantidad_lineas" > "$archivo_marca"
```

BASH

SCRIPT PARSER (Servicio)

En este apartado vamos a crear un demonio asignado al script para que el servicio se inicie automáticamente al encender el sistema operativo.

El proceso de creación de servicio también está documentado en la guía de centralización y recepción de logs. Sin embargo, como este script lo hemos creado desde cero, vamos a explicar algunas funciones asignadas al reinicio del demonio.

Primero debemos crear el archivo de configuración del servicio. Podemos ponerle el nombre que queramos, pero en mi caso lo he llamado “logsparser.service”.

```
sudo nano /etc/systemd/systemd/logsparser.service
```

BASH

El contenido de este archivo debe respetar el siguiente formato, en el cual “ExecStart” hace referencia al script que queremos iniciar. Después, “WorkingDirectory” define la carpeta donde se encuentra dicho script y, por último, “User” y “Group” son parámetros que especifican el usuario y el grupo propietarios del servicio. Deben ser objetos con permisos para iniciar el script y los componentes subyacentes a este.

```
[Unit]
Description=DNS_DHCP_SCRIPT
After=network.target

[Service]
ExecStart=/home/logswindows/dnslogs/dns_script.sh
WorkingDirectory=/home/logswindows/dnslogs
Restart=always
User=root
Group=root

[Install]
WantedBy=multi-user.target
```

BASH

Ahora, reiniciamos los demonios e iniciamos el servicio

```
sudo systemctl daemon-reload
```

BASH

```
sudo systemctl start logsparser.service
```

BASH

Por último, ejecutamos el siguiente comando para que el servicio se inicie al encender el servidor

```
sudo systemctl enable logsparser.service
```

BASH

Una vez creado el servicio, vamos a explicar algunas características relacionadas con el reinicio del demonio.

La función “vaciar_contenido_carpeta” permite vaciar los archivos de traducción y de conteo de líneas cuando el script vuelve a iniciarse. Esto se hace con el objetivo de no llenar el espacio del servidor. Para no perder datos, siempre que el servidor se reinicie se hará una copia de seguridad con la función “crear_archivo_logs”. En caso de que no se reinicie, todos los días a las 15:00 se creará una copia.

Función “vaciar_contenido_carpeta” que se activa al reiniciar el servicio:

```
vaciar_contenido_carpeta() {  
    carpeta="$1"  
    find "$carpeta" -type f -exec sh -c '{}' \;  
}  
  
# Vaciar el contenido de los archivos  
vaciar_contenido_carpeta "$archivo_salida"  
vaciar_contenido_carpeta "$archivo_marca"
```

BASH

Copias de seguridad al salir del script “crear_archivos_log”

```
# Función para crear el archivo de logs con nombre
dnslogs(fecha_hora)
crear_archivo_logs() {
    fecha_hora=$(date +"%Y-%m-%d_%H-%M-%S")

    archivo_logs="/home/logswindows/dnslogs/dnslogs_$(fecha_h
ora"
    cp "$archivo_salida" "$archivo_logs"
    echo "crear archivos"
}

# Funcion para manejar la señal de cierre del script

on_script_exit() {
    crear_archivo_logs
}

trap on_script_exit EXIT
```

BASH

EJECUCIÓN

Primero encendemos el servidor Ubuntu. Este empezará a intentar establecer conexión con Windows para montar la carpeta compartida en el directorio que hemos especificado.

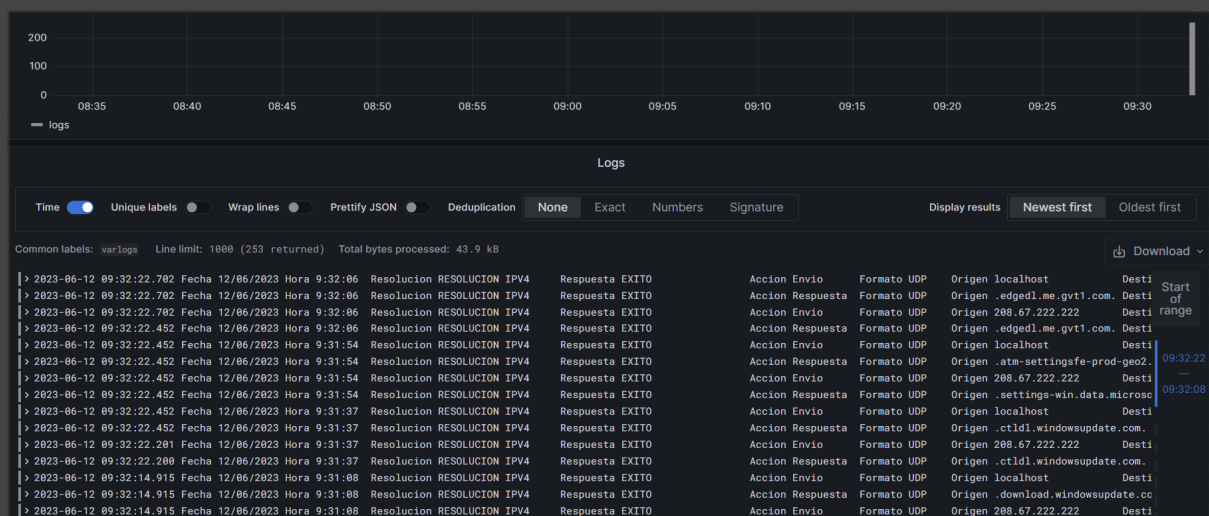
Cuando establezca conexión, se montará la carpeta \\192.168.1.137\logs en /mnt/logs y se asignará la variable “archivo_entrada” con la ruta del archivo de registro “/mnt/logs/debug”

En este punto, el script empezará a traducir el archivo “/mnt/logs/debug” y lo enviará a la carpeta que hayamos especificado en la variable “archivo_salida”. Cuando termine de traducir el archivo, se quedará a la espera de nuevas líneas comparando las cantidad de líneas leídas con la cantidad de líneas del archivo “/mnt/logs/debug”

Promtail estará leyendo la carpeta del archivo de salida (en mi caso ~/dnslogs/logs) y enviando la información a Loki, el cual a su vez estará enviando los logs a Grafana

La forma de visualizar los logs DNS en Grafana es la misma que en el primer caso. Debemos asignar la fuente de datos de Loki a Grafana

Por último, así se verían los logs procesados



CONCLUSIONES

Hemos creado un servidor de recepción que visualiza los registros en tiempo real. Cada caso es adaptable y escalable a las empresas cuyo estudio del entorno de trabajo haya concluido de forma similar a las necesidades expuestas en cada manual. En adición, estos casos te ayudarán a guiarte en la adaptación del servidor a tus necesidades concretas.

A través de la detección de amenazas y vulnerabilidades, se ha logrado fortalecer la seguridad de los sistemas informáticos de la organización, permitiendo una identificación temprana y una respuesta rápida ante posibles incidentes de seguridad.

La monitorización en tiempo real del entorno de trabajo ha proporcionado una visibilidad continua de la infraestructura informática, permitiendo detectar y abordar rápidamente problemas como caídas de conexión y errores en resoluciones.

Esta capacidad de respuesta inmediata ha mejorado la disponibilidad de los sistemas y ha minimizado los tiempos de inactividad

Gracias a que la implementación ha sido bastante económica, el sistema de recepción de logs ha ofrecido un valor agregado al permitir una toma de decisiones informada y basada en datos, lo que ha facilitado la asignación adecuada de los recursos disponibles y ha permitido que la organización pueda destinar presupuesto a otras áreas críticas.