Configuración del flujo de trabajo gRPC avanzado con Telegraf, InfluxDB y Grafana en Catalyst 9800

Contenido

Introducción

Este documento describe cómo implementar la pila Telegraf, InfluxDB y Grafana (TIG) e interconectarla con Catalyst 9800.

Prerequisites

Este documento demuestra las capacidades de interfaces programáticas de Catalyst 9800 a través de una integración compleja. Este documento tiene como objetivo mostrar cómo estos pueden ser totalmente personalizables en función de cualquier necesidad y ser ahorradores de tiempo diarios. La implementación mostrada aquí se basa en gRPC y presenta la configuración de telemetría para hacer que los datos inalámbricos del Catalyst 9800 estén disponibles en cualquier pila de observabilidad de Telegraf, InfluxDB, Grafana (TIG).

Requirements

Cisco recomienda que tenga conocimiento sobre estos temas:

- Modelo de configuración de Catalyst Wireless 9800.
- Programabilidad de la red y modelos de datos.
- Conceptos básicos de la pila TIG.

Componentes Utilizados

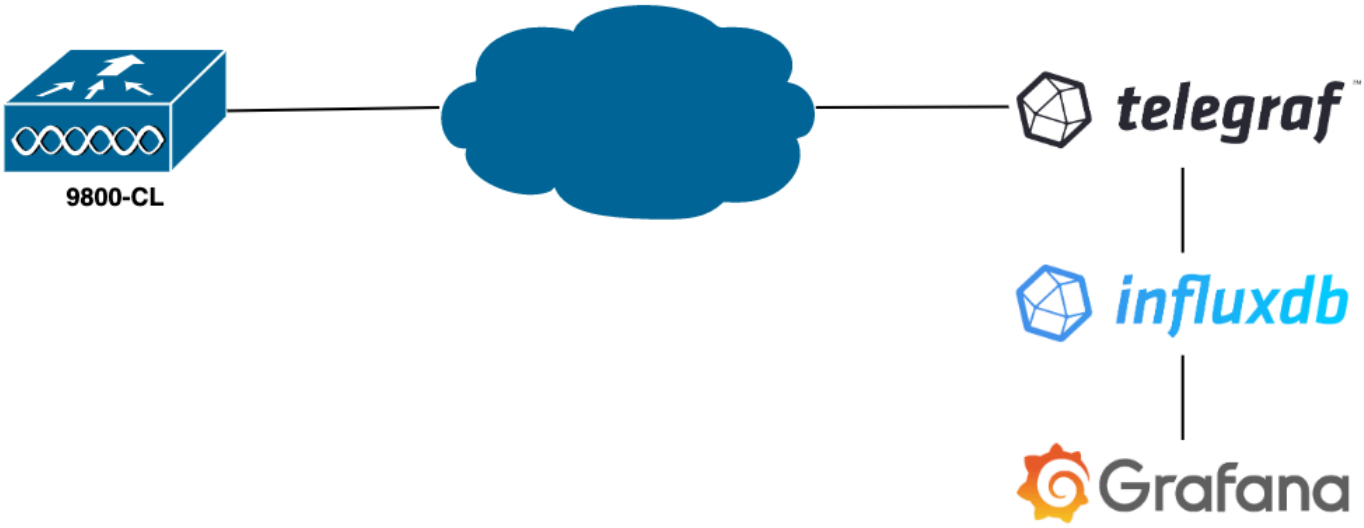
La información que contiene este documento se basa en las siguientes versiones de software y hardware.

- Catalyst 9800-CL (v. 17.12.03).
- Ubuntu (v. 22.04.03).
- InfluxDB (v. 1.06.07).
- Telegraf (v. 1.21.04).
- Grafana (v. 10.02.01).

La información que contiene este documento se creó a partir de los dispositivos en un ambiente de laboratorio específico. Todos los dispositivos que se utilizan en este documento se pusieron en funcionamiento con una configuración verificada (predeterminada). Si tiene una red en vivo, asegúrese de entender el posible impacto de cualquier comando.

Configurar

Diagrama de la red



Configuraciones

En este ejemplo, la telemetría se configura en un 9800-CL usando el marcado de salida gRPC para enviar información en una aplicación Telegraph almacenándola en una base de datos InfluxDB. Aquí, se utilizaron dos dispositivos,

- Un servidor Ubuntu que aloja toda la pila TIG.
- Un Catalyst 9800-CL.

Esta guía de configuración no se centra en la implementación completa de estos dispositivos, sino en las configuraciones requeridas en cada aplicación para que la información del 9800 se envíe, reciba y presente correctamente.

Paso 1. Preparación de la base de datos

Antes de entrar en la parte de configuración, asegúrese de que la instancia Influx se está ejecutando correctamente. Esto se puede hacer fácilmente usando el `systemctl status` comando, si está usando una distribución Linux.

```
admin@tig:~$ systemctl status influxd
● influxdb.service - InfluxDB is an open-source, distributed, time series database
     Loaded: loaded (/lib/systemd/system/influxdb.service; enabled; vendor preset: enabled)
     Active: active (running) since Wed 2023-06-14 13:06:18 UTC; 2 weeks 5 days ago
       Docs: https://docs.influxdata.com/influxdb/
   Main PID: 733 (influxd)
      Tasks: 15 (limit: 19180)
     Memory: 4.2G
        CPU: 1h 28min 47.366s
     CGroup: /system.slice/influxdb.service
             └─733 /usr/bin/influxd -config /etc/influxdb/influxdb.conf
```

Para que el ejemplo funcione, Telegraf necesita una base de datos para almacenar las métricas, así como un usuario para conectarse a esta. Estos se pueden crear fácilmente desde la CLI de

InfluxDB, usando estos comandos:

```
admin@tig:~$ influx
Connected to http://localhost:8086 version 1.8.10
InfluxDB shell version: 1.8.10
> create database TELEGRAF
> create user telegraf with password 'YOUR_PASSWORD'
```

La base de datos ahora creada, Telegraf se puede configurar para almacenar métricas en ella correctamente.

Paso 2. Prepare Telegraf

Para que este ejemplo funcione, sólo son interesantes dos configuraciones de Telegrabado. Éstas se pueden realizar (como es habitual para las aplicaciones que se ejecutan en Unix) desde el archivo de `/etc/telegraf/telegraf.conf` configuración.

El primero declara la salida utilizada por Telegraf. Como se indicó anteriormente, InfluxDB se utiliza aquí y se configura en la sección de salida del `telegraf.conf` archivo de la siguiente manera:

```
###############################################################################
#                            OUTPUT PLUGINS                                   #
###############################################################################
# Output Plugin InfluxDB
[[outputs.influxdb]]
## The full HTTP or UDP URL for your InfluxDB instance.
#   ##
#   ## Multiple URLs can be specified for a single cluster, only ONE of the
#   ## urls will be written to each interval.
  urls = [ "http://127.0.0.1:8086" ]
#   ## The target database for metrics; will be created as needed.
#   ## For UDP url endpoint database needs to be configured on server side.
  database = "TELEGRAF"
#   ## HTTP Basic Auth
  username = "telegraf"
  password = "YOUR_PASSWORD"
```

Esto indica al proceso de Telegraf que almacene los datos que recibe en InfluxDB que se ejecuta en el mismo host en el puerto 8086 y que utilice la base de datos llamada "TELEGRAF" (así como las credenciales telegraf/YOUR_PASSWORD para acceder a ella).

Si lo primero declarado fue el formato de salida, el segundo es, por supuesto, el formato de entrada. Para informar a Telemraf que los datos que recibe provienen de un dispositivo Cisco que utiliza telemetría, puede utilizar el módulo de entrada cisco_telemetry_mdt". Para configurar esto, solo necesita agregar estas líneas en el `/etc/telegraf/telegraf.conf` archivo:

```
###############################################################################
#                            INPUT PLUGINS                                    #
###############################################################################
# # Cisco model-driven telemetry (MDT) input plugin for IOS XR, IOS XE and NX-OS platforms
[[inputs.cisco_telemetry_mdt]]
#  ## Telemetry transport can be "tcp" or "grpc".  TLS is only supported when
#  ## using the grpc transport.
    transport = "grpc"
#
#  ## Address and port to host telemetry listener
    service_address = ":57000"
#  ## Define aliases to map telemetry encoding paths to simple measurement names
[inputs.cisco_telemetry_mdt.aliases]
    ifstats = "ietf-interfaces:interfaces-state/interface/statistics"
```

Esto hace que la aplicación Telegraf que se ejecuta en el host (en el puerto predeterminado 57000) pueda decodificar los datos recibidos que provienen del WLC.

Una vez guardada la configuración, asegúrese de reiniciar Telegraf para aplicarlo al servicio. Asegúrese también de que el servicio se haya reiniciado correctamente:

```
admin@tig:~$ sudo systemctl restart telegraf
admin@tig:~$ systemctl status telegraf.service
● telegraf.service - Telegraf
     Loaded: loaded (/lib/systemd/system/telegraf.service; enabled; vendor preset: enabled)
     Active: active (running) since Mon 2023-07-03 17:12:49 UTC; 2min 18s ago
       Docs: https://github.com/influxdata/telegraf
   Main PID: 110182 (telegraf)
      Tasks: 10 (limit: 19180)
     Memory: 47.6M
        CPU: 614ms
     CGroup: /system.slice/telegraf.service
             └─110182 /usr/bin/telegraf -config /etc/telegraf/telegraf.conf -config-directory /etc/telegraf/teleg
```

Paso 3. Determinación de la suscripción de telemetría que contiene la métrica deseada

Como se ha indicado, tanto en los dispositivos de Cisco como en muchos otros, las métricas se organizan según el modelo YANG. Los modelos específicos de Cisco YANG para cada versión de IOS XE (utilizados en el 9800) se pueden encontrar aquí, en particular el modelo para IOS XE Dublin 17.12.03 utilizado en este ejemplo.

En este ejemplo, nos centramos en recopilar las métricas de utilización de la CPU de la instancia 9800-CL utilizada. Al inspeccionar el modelo YANG para Cisco IOS XE Dublin 17.12.03, se puede determinar qué módulo contiene la utilización de la CPU del controlador y, en particular, durante los últimos 5 segundos. Estos son parte del módulo Cisco-IOS-XE-process-cpu-oper, bajo la agrupación cpu-utilization (cinco segundos de hoja).

Paso 4. Habilite NETCONF en el controlador

El marco de marcado de salida gRPC depende de NETCONF para funcionar de la misma manera. Por lo tanto, esta función debe estar habilitada en el 9800 y esto se logra ejecutando estos comandos:

```
WLC(config)#netconf ssh
WLC(config)#netconf-yang
```

Paso 5. Configuración de la suscripción de telemetría en el controlador

Una vez que XPath (alias, XML Paths Language) de las métricas determinadas a partir del modelo YANG, una suscripción de telemetría se puede configurar fácilmente desde la CLI 9800 para comenzar a transmitir estos a la instancia de Telegraf configurada en el Paso 2. Esto se realiza mediante la ejecución de estos comandos:

```
WLC(config)#telemetry ietf subscription 101
WLC(config-mdt-subs)#encoding encode-kvgpb
WLC(config-mdt-subs)#filter xpath /process-cpu-ios-xe-oper:cpu-usage/cpu-utilization/five-seconds
WLC(config-mdt-subs)#source-address 10.48.39.130
WLC(config-mdt-subs)#stream yang-push
WLC(config-mdt-subs)#update-policy periodic 100
WLC(config-mdt-subs)#receiver ip address 10.48.39.98 57000 protocol grpc-tcp
```

En este bloque de código, primero se define la suscripción de telemetría con el identificador 101. El identificador de suscripción puede ser cualquier número entre <0-2147483647>, siempre y cuando no se superponga con otra suscripción. Para esta suscripción se han configurado, en este orden:

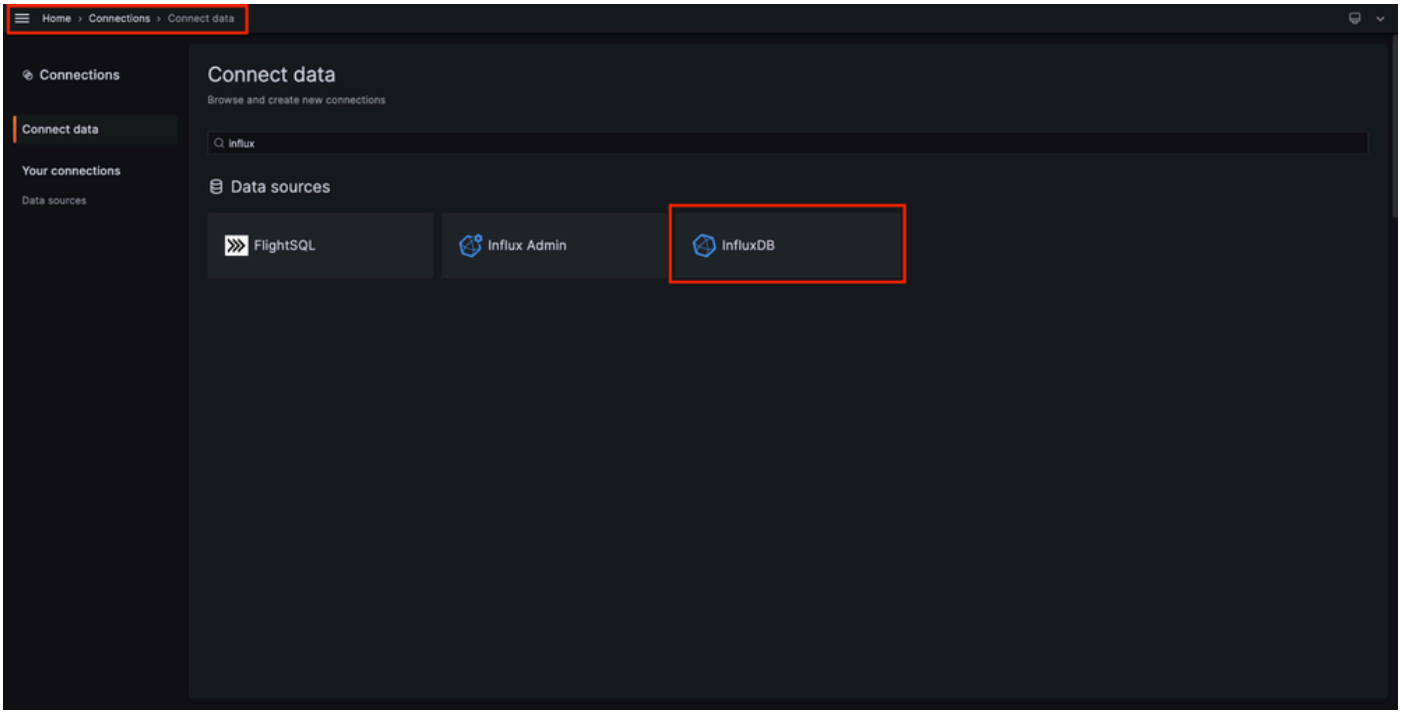
- El método de codificación utilizado, que debe ser kvGPB al trabajar con el protocolo de transporte gRPC.
- El filtro para las métricas enviadas por la suscripción, siendo la XPath que define la métrica interesante para nosotros (saber, /process-cpu-ios-xe-oper:cpu-usage/cpu-utilization/five-seconds).
- La dirección IP de origen utilizada por el controlador para enviar las métricas.
- El tipo de flujo utilizado para comunicar las métricas, en este caso el estándar YANG Push IETF.
- La frecuencia utilizada por el controlador para enviar datos al suscriptor en 100^{th} of seconds. En este caso, se configuró para enviar actualizaciones periódicamente cada segundo.
- La dirección IP del receptor y el número de puerto, así como el protocolo utilizado para la comunicación entre el controlador y el suscriptor. En este ejemplo, gRPC-TCPse utiliza para enviar la métrica al host 10.48.39.98 en el puerto 57000.

Paso 6. Configurar origen de datos Grafana

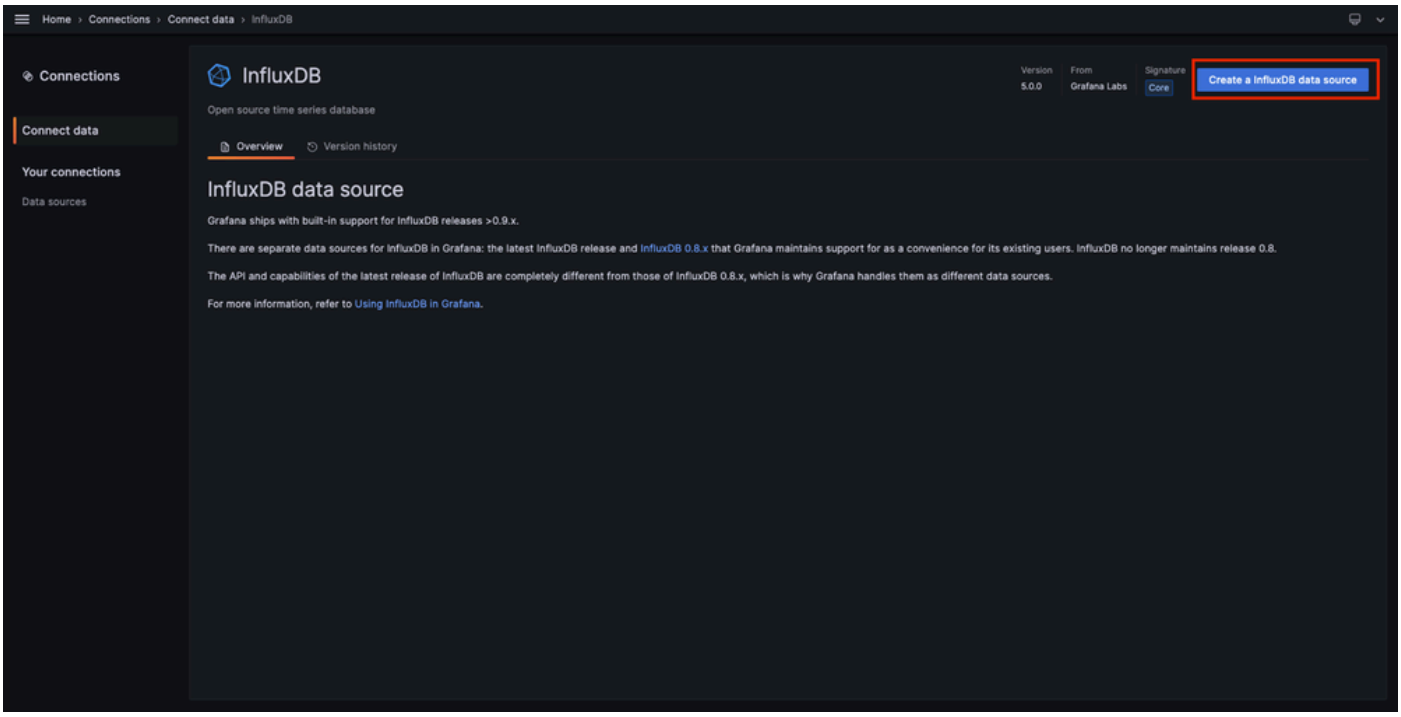
Ahora que el controlador comienza a enviar datos a Telegraf y que estos se almacenan en la

base de datos TELEGRAF InfluxDB, es hora de configurar Grafana para que pueda examinar estas métricas.

Desde la GUI de Grafana, navegue hasta Inicio > Conexiones > Conectar datos y utilice la barra de búsqueda para encontrar el origen de datos de InfluxDB.



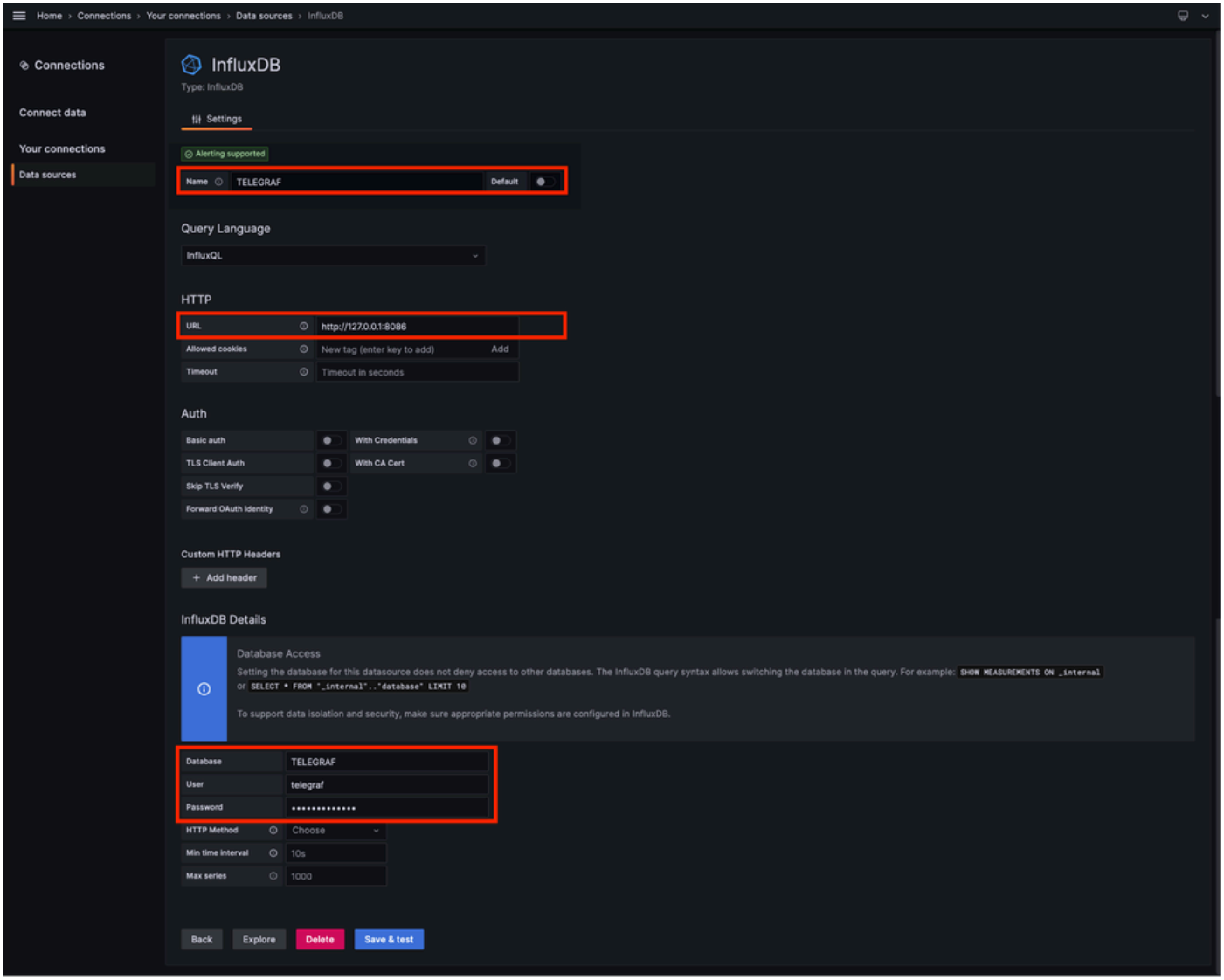
Seleccione este tipo de origen de datos y utilice el botón "Crear un origen de datos InfluxDB" para conectar Grafana y la base de datos TELEGRAPH creada en el Paso 1.



Rellene el formulario que aparece en la pantalla, especialmente proporcione:

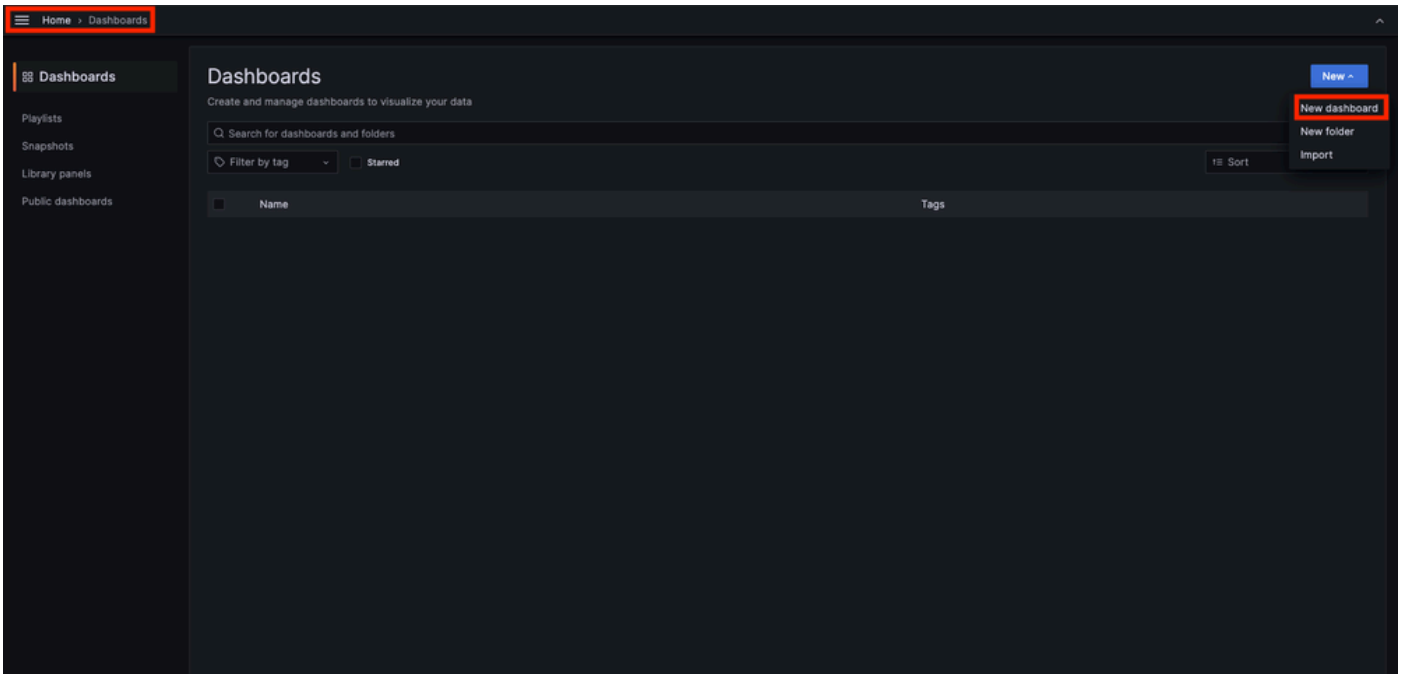
- Nombre del origen de datos.

- URL de la instancia InfluxDB utilizada.
- El nombre de base de datos utilizado (en este ejemplo, "TELEGRAF").
- La credencial del usuario definida para acceder a ella (en este ejemplo, telegraf/YOUR_PASSWORD).

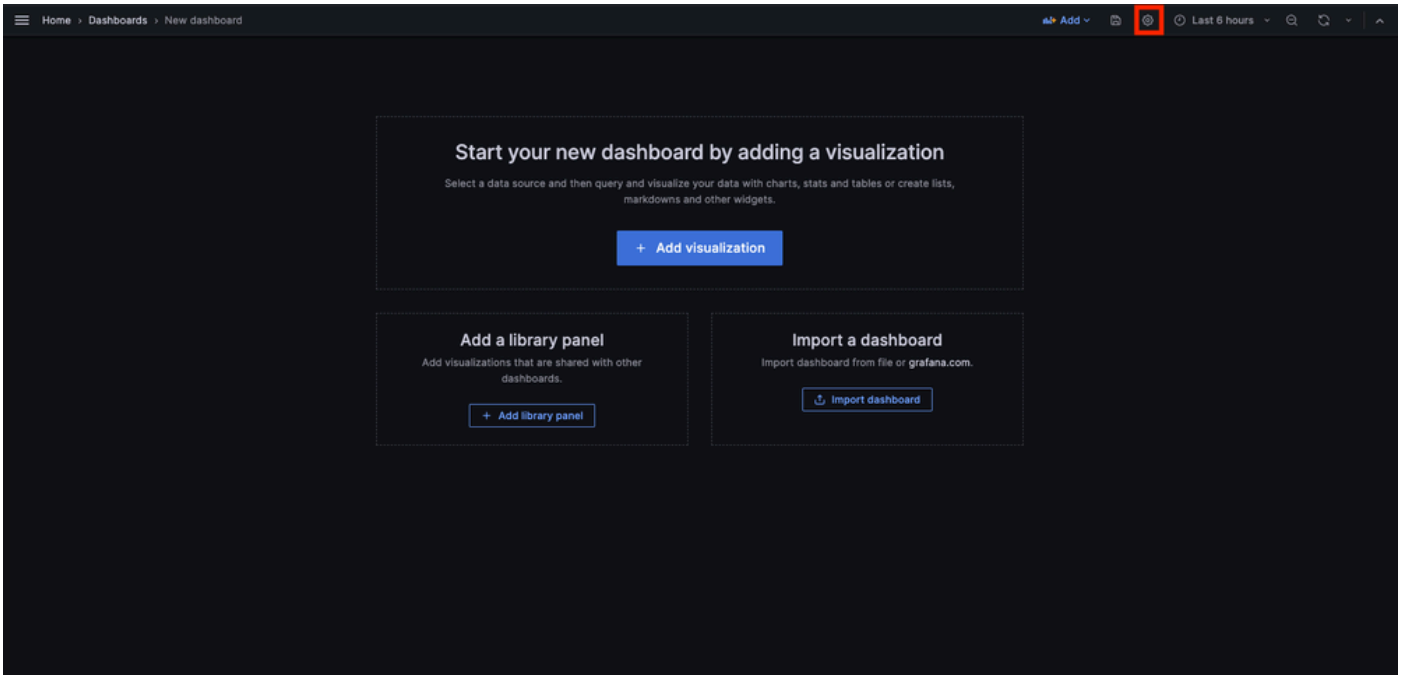


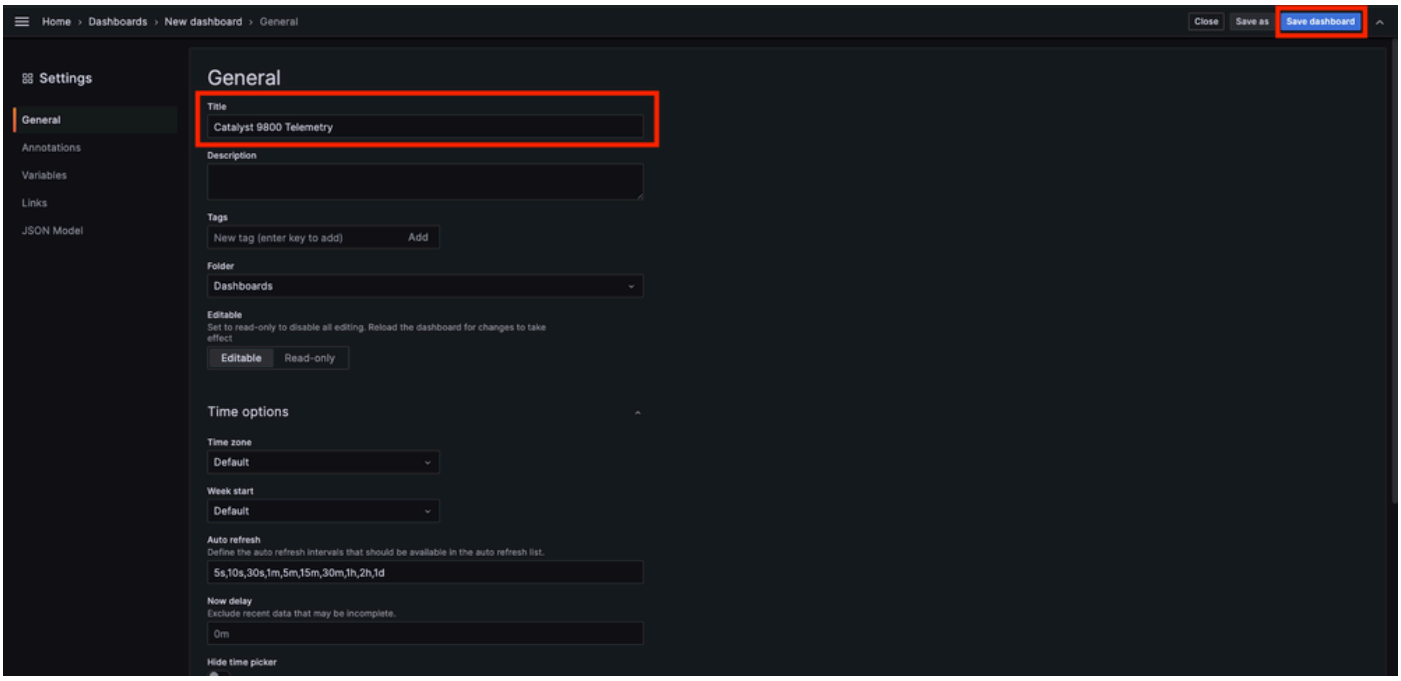
Paso 7. Crear un panel

Las visualizaciones Grafana se organizan en Paneles. Para crear un panel que contenga las visualizaciones de métricas de Catalyst 9800, navegue hasta Inicio > Paneles y utilice el botón "Nuevo panel"



Se abrirá el nuevo panel creado. Pulse en los iconos de engranaje para acceder al parámetro del tablero de mandos y cambiar su nombre. En el ejemplo, se utiliza "Telemetría de Catalyst 9800". Una vez realizado este paso, utilice el botón "Guardar panel" para guardar el panel.

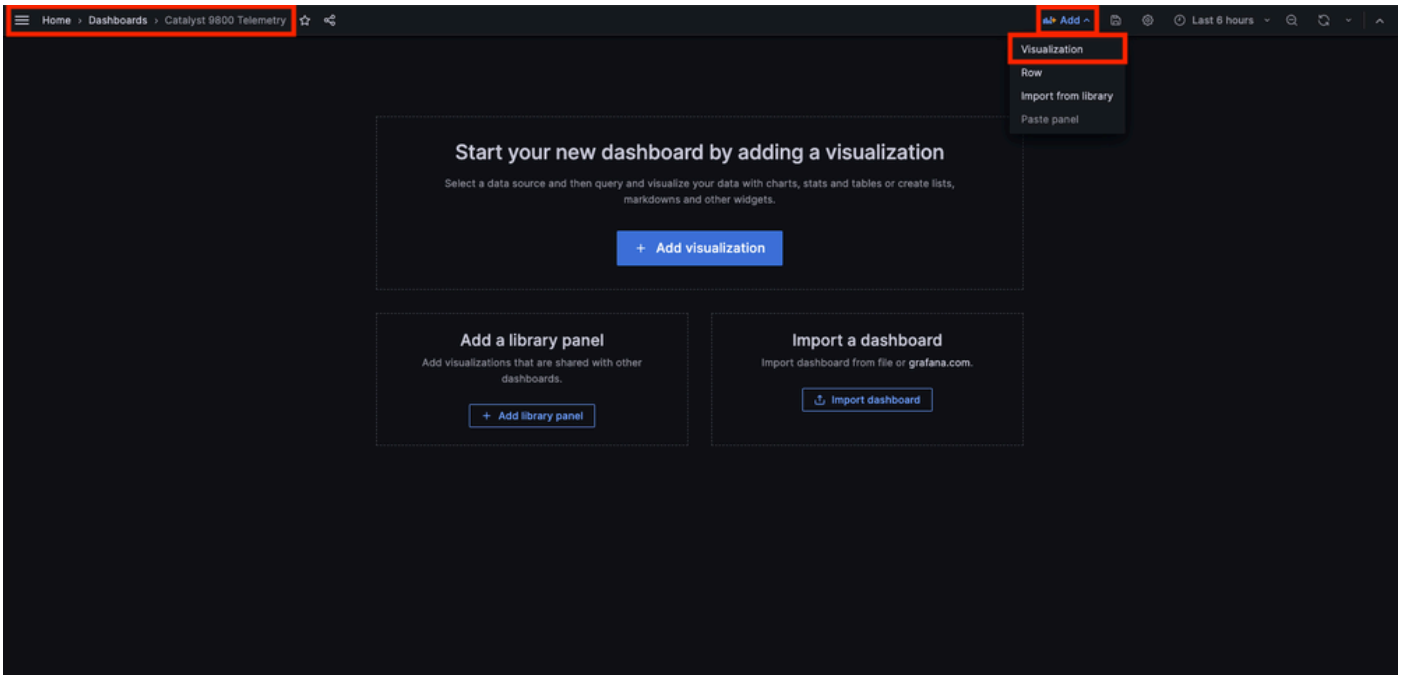




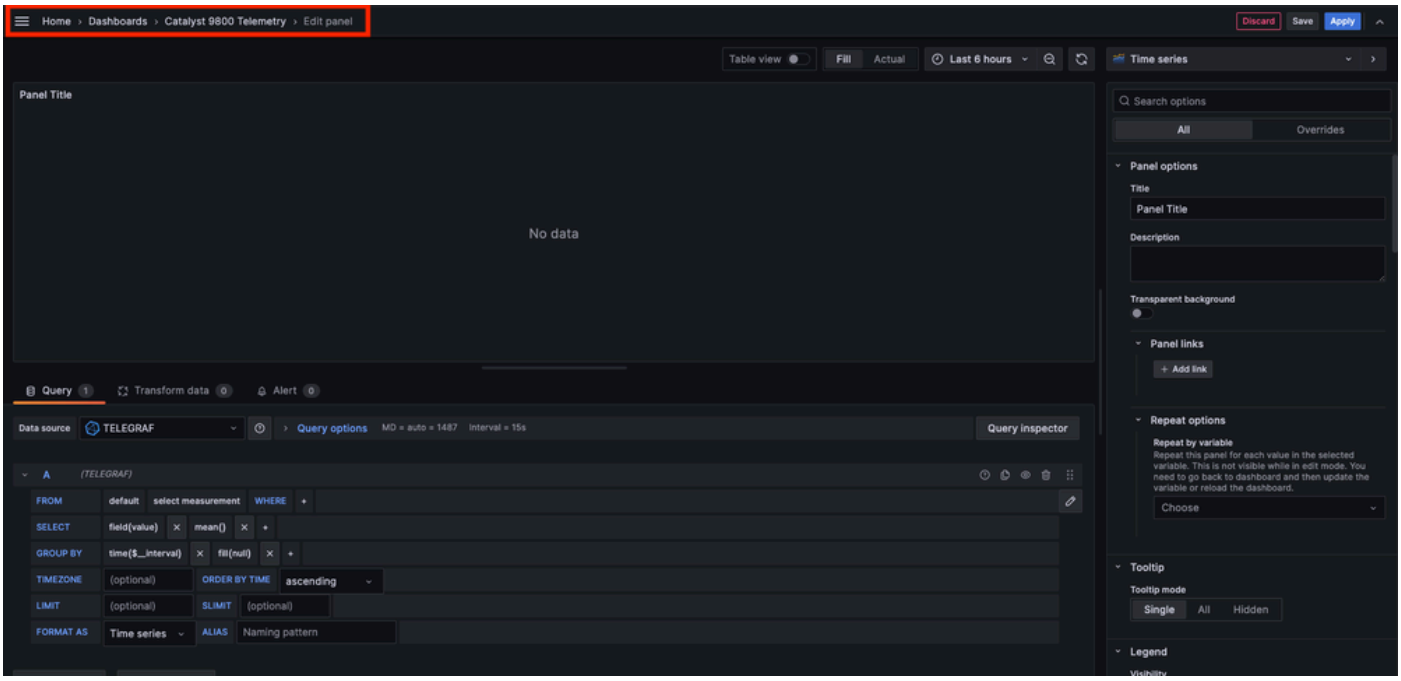
Paso 8. Agregar una visualización al panel

Ahora que los datos son enviados, recibidos y almacenados correctamente y que Grafana tiene acceso a esta ubicación de almacenamiento, es el momento de crear una visualización para ellos.

Desde cualquier panel de Grafana, utilice el botón "Agregar" y seleccione "Visualización" en el menú que aparece para crear una visualización de sus métricas.

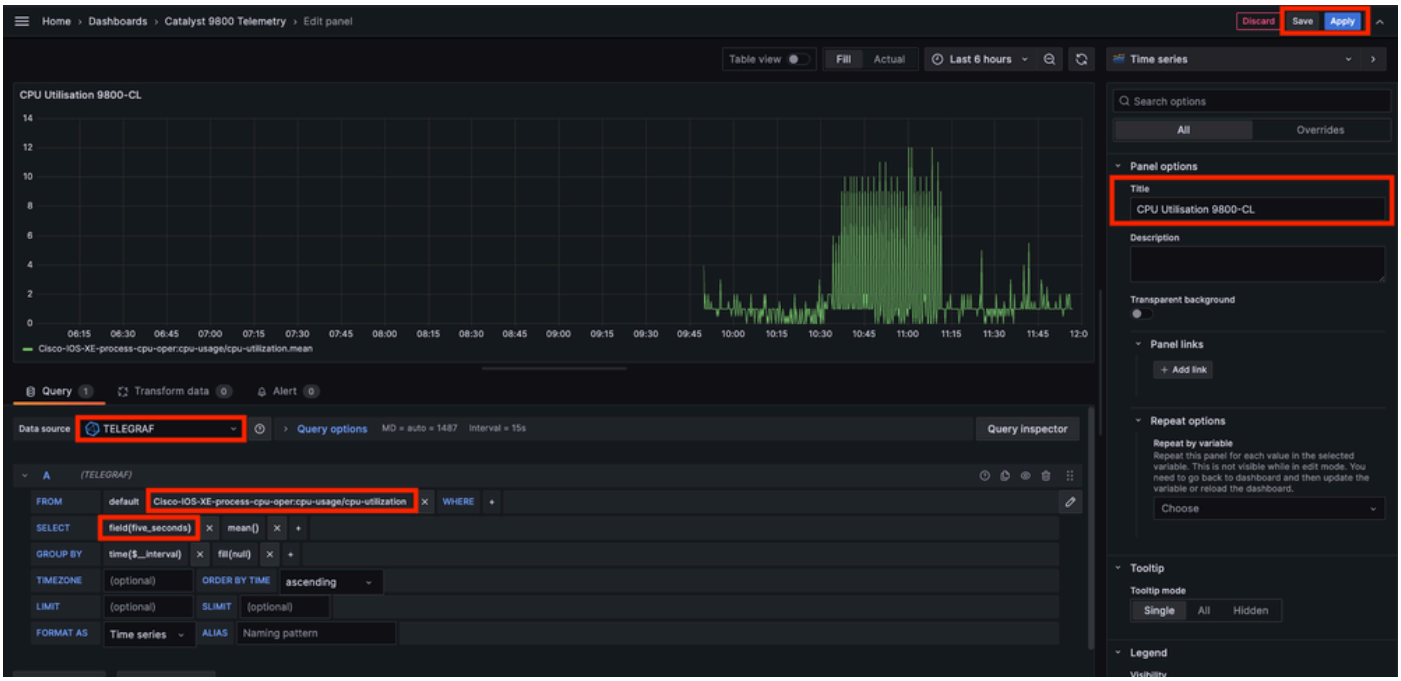


Se abrirá el panel Editar de la visualización creada:



En este panel, seleccione

- El nombre del origen de datos que creó en el Paso 6, TELEGRAF en este ejemplo.
- La medida (esquema) que contiene los datos que desea visualizar, "Cisco-IOS-XE-process-cpu-oper:cpu-usage/cpu-utilization" en este ejemplo.
- Campo de la base de datos que representa las métricas que desea visualizar, "five_seconds" en este ejemplo.
- El título de la visualización, "CPU Utilization 9800-CL" en este ejemplo.



Una vez que se presiona el botón "Guardar/Aplicar" de la figura anterior, se agrega al panel la visualización que muestra el uso de la CPU del controlador Catalyst 9800 a lo largo del tiempo. Los cambios realizados en el panel se pueden guardar mediante el botón de icono de disquete.

Verificación

Configuración en Ejecución de WLC

```
Building configuration...
Current configuration : 112215 bytes
!
! Last configuration change at 14:28:36 UTC Thu May 23 2024 by admin
! NVRAM config last updated at 14:28:23 UTC Thu May 23 2024 by admin
!
version 17.12
[...]
aaa new-model
!
!
aaa authentication login default local
aaa authentication login local-auth local
aaa authentication dot1x default group radius
aaa authorization exec default local
aaa authorization network default group radius
[...]
vlan internal allocation policy ascending
!
vlan 39
!
vlan 1413
 name VLAN_1413
!
!
interface GigabitEthernet1
 switchport access vlan 1413
 negotiation auto
 no mop enabled
 no mop sysid
!
interface GigabitEthernet2
```

```
 switchport trunk allowed vlan 39,1413
 switchport mode trunk
 negotiation auto
 no mop enabled
 no mop sysid
!
interface Vlan1
 no ip address
 no ip proxy-arp
 no mop enabled
 no mop sysid
!
interface Vlan39
 ip address 10.48.39.130 255.255.255.0
 no ip proxy-arp
 no mop enabled
 no mop sysid
[...]
telemetry ietf subscription 101
 encoding encode-kvgpb
 filter xpath /process-cpu-ios-xe-oper:cpu-usage/cpu-utilization
 source-address 10.48.39.130
 stream yang-push
 update-policy periodic 1000
 receiver ip address 10.48.39.98 57000 protocol grpc-tcp
[...]
netconf-yang
```

Configuración de Telegrabado

```
# Configuration for telegraf agent
[agent]
  metric_buffer_limit = 10000
  collection_jitter = "0s"
  debug = true
  quiet = false
  flush_jitter = "0s"
  hostname = ""
  omit_hostname = false

###############################################################################
#                            OUTPUT PLUGINS                                   #
###############################################################################
# Configuration for sending metrics to InfluxDB
[[outputs.influxdb]]
  urls = ["http://127.0.0.1:8086"]
  database = "TELEGRAF"
  username = "telegraf"
  password = "Wireless123#"

###############################################################################
#                            INPUT PLUGINS                                    #
###############################################################################

###############################################################################
#                            SERVICE INPUT PLUGINS                            #
###############################################################################
```

```
# # Cisco model-driven telemetry (MDT) input plugin for IOS XR, IOS XE and NX-OS platforms
[[inputs.cisco_telemetry_mdt]]
  transport = "grpc"
  service_address = "10.48.39.98:57000"
 [inputs.cisco_telemetry_mdt.aliases]
    ifstats = "ietf-interfaces:interfaces-state/interface/statistics"
```

Configuración de InfluxDB

```
### Welcome to the InfluxDB configuration file.
reporting-enabled = false
[meta]
  dir = "/var/lib/influxdb/meta"

[data]
  dir = "/var/lib/influxdb/data"
  wal-dir = "/var/lib/influxdb/wal"

[retention]
  enabled = true
  check-interval = "30m"
```

Configuración de Grafana

```
################################### Server ####################################
[server]
http_addr = 10.48.39.98
domain = 10.48.39.98
```

Troubleshoot

WLC One Stop-Shop Reflex

Desde el lado del WLC, lo primero que hay que verificar es que los procesos relacionados con las interfaces programáticas están funcionando.

```
#show platform software yang-management process
confd : Running
nesd : Running
syncfd : Running
ncsshd : Running               <-- NETCONF / gRPC Dial-Out
dmiauthd : Running             <-- For all of them, Device Managment Interface needs to be up.
nginx : Running                <-- RESTCONF
ndbmand : Running
pubd : Running
```

```
gnmib : Running                <-- gNMI
```

Para NETCONF (utilizado por marcado de salida gRPC), estos comandos también pueden ayudar a verificar el estado del proceso.

```
WLC#show netconf-yang status
netconf-yang: enabled
netconf-yang candidate-datastore: disabled
netconf-yang side-effect-sync: enabled
netconf-yang ssh port: 830
netconf-yang turbocli: disabled
netconf-yang ssh hostkey algorithms: rsa-sha2-256,rsa-sha2-512,ssh-rsa
netconf-yang ssh encryption algorithms: aes128-ctr,aes192-ctr,aes256-ctr,aes128-cbc,aes256-cbc
netconf-yang ssh MAC algorithms: hmac-sha2-256,hmac-sha2-512,hmac-sha1
netconf-yang ssh KEX algorithms: diffie-hellman-group14-sha1,diffie-hellman-group14-sha256,ecdh-sha2-ni
```

Una vez que se verifica el estado del proceso, otra verificación importante es el estado de la conexión de telemetría entre el Catalyst 9800 y el receptor Telegraf. Se puede ver mediante el comando "show telemetry connection all".

```
WLC#show telemetry connection all
Telemetry connections

Index Peer Address               Port  VRF Source Address             State      State Description
----- -------------------------- ----- --- -------------------------- ---------- --------------------
28851 10.48.39.98                57000 0   10.48.39.130               Active     Connection up
```

Si la conexión de telemetría está activa entre el WLC y el receptor, también se puede asegurar que las suscripciones configuradas sean válidas usando el show telemetry ietf subscription all brief comando.

```
WLC#show telemetry ietf subscription all brief
ID         Type       State      State Description
101        Configured Valid      Subscription validated
```

La versión detallada de este comando, show telemetry ietf subscription all detail, proporciona más información sobre las suscripciones y puede ayudar a señalar un problema de su configuración.

```
WLC#show telemetry ietf subscription all detail
Telemetry subscription detail:

  Subscription ID: 101
  Type: Configured
```

```
  State: Valid
  Stream: yang-push
  Filter:
    Filter type: xpath
    XPath: /process-cpu-ios-xe-oper:cpu-usage/cpu-utilization
  Update policy:
    Update Trigger: periodic
    Period: 1000
  Encoding: encode-kvgpb
  Source VRF:
  Source Address: 10.48.39.130
  Notes: Subscription validated

  Named Receivers:
    Name                                                Last State Change  State                Explanat
    ----------------------------------------------------------------------------------------------------
    grpc-tcp://10.48.39.98:57000                        05/23/24 08:00:25  Connected
```

Confirmar disponibilidad de red

El controlador Catalyst 9800 envía datos gRPC al puerto receptor configurado para cada suscripción de telemetría.

```
WLC#show run | include receiver ip address
receiver ip address 10.48.39.98 57000 protocol grpc-tcp
```

Para verificar la conectividad de red entre el WLC y el receptor en este puerto configurado, varias herramientas están disponibles.

Desde el WLC, se puede utilizar telnet en el puerto/IP del receptor configurado (aquí 10.48.39.98:57000) para verificar que éste esté abierto y accesible desde el controlador mismo. Si el tráfico no está siendo bloqueado, el puerto debe aparecer como abierto en la salida:

```
WLC#telnet 10.48.39.98 57000
Trying 10.48.39.98, 57000 ... Open <-------
```

Alternativamente, se puede utilizar Nmap desde cualquier host para asegurarse de que el receptor esté expuesto correctamente en el puerto configurado.

```
$ sudo nmap -sU -p 57000 10.48.39.98
Starting Nmap 7.95 ( https://nmap.org ) at 2024-05-17 13:12 CEST
Nmap scan report for air-1852e-i-1.cisco.com (10.48.39.98)
Host is up (0.020s latency).

PORT      STATE         SERVICE
57000/udp open|filtered unknown
```

```
Nmap done: 1 IP address (1 host up) scanned in 0.35 seconds
```

Registro y depuración

```
2024/05/23 14:40:36.566486156 {pubd_RO-0}{2}: [mdt-ctrl] [30214]: (note): **** Event Entry: Configured
2024/05/23 14:40:36.566598609 {pubd_RO-0}{2}: [mdt-ctrl] [30214]: (note): Use count for named receiver
2024/05/23 14:40:36.566600301 {pubd_RO-0}{2}: [mdt-ctrl] [30214]: (note): {subscription receiver event=
[...]
2024/05/23 14:40:36.572402901 {pubd_RO-0}{2}: [pubd] [30214]: (info): Collated data collector filters fo
2024/05/23 14:40:36.572405081 {pubd_RO-0}{2}: [pubd] [30214]: (debug): Creating periodic sensor for sub
2024/05/23 14:40:36.572670046 {pubd_RO-0}{2}: [pubd] [30214]: (info): Creating data collector type 'ei_
2024/05/23 14:40:36.572670761 {pubd_RO-0}{2}: [pubd] [30214]: (debug): Creating crimson data collector
2024/05/23 14:40:36.572671763 {pubd_RO-0}{2}: [pubd] [30214]: (debug): Need new data collector instance
2024/05/23 14:40:36.572675434 {pubd_RO-0}{2}: [pubd] [30214]: (debug): Creating CRIMSON periodic data co
2024/05/23 14:40:36.572688399 {pubd_RO-0}{2}: [pubd] [30214]: (debug): tree rooted at cpu-usage
2024/05/23 14:40:36.572715384 {pubd_RO-0}{2}: [pubd] [30214]: (debug): last container/list node 0
2024/05/23 14:40:36.572740734 {pubd_RO-0}{2}: [pubd] [30214]: (debug): 1 non leaf children to render fro
2024/05/23 14:40:36.573135594 {pubd_RO-0}{2}: [pubd] [30214]: (debug): URI:/cpu_usage;singleton_id=0 SIB
2024/05/23 14:40:36.573147953 {pubd_RO-0}{2}: [pubd] [30214]: (debug): 0 non leaf children to render fro
2024/05/23 14:40:36.573159482 {pubd_RO-0}{2}: [pubd] [30214]: (debug): Timer created for subscription 10
2024/05/23 14:40:36.573166451 {pubd_RO-0}{2}: [mdt-ctrl] [30214]: (note): {subscription receiver event=
2024/05/23 14:40:36.573197750 {pubd_RO-0}{2}: [pubd] [30214]: (debug): Starting batch from periodic col
2024/05/23 14:40:36.573198408 {pubd_RO-0}{2}: [pubd] [30214]: (debug): Building from the template
2024/05/23 14:40:36.575467870 {pubd_RO-0}{2}: [pubd] [30214]: (debug): Created dbal batch:133, for crim
2024/05/23 14:40:36.575470867 {pubd_RO-0}{2}: [pubd] [30214]: (debug): Done building from the template
2024/05/23 14:40:36.575481078 {pubd_RO-0}{2}: [pubd] [30214]: (debug): Executing batch:133 for periodic
2024/05/23 14:40:36.575539723 {pubd_RO-0}{2}: [mdt-ctrl] [30214]: (note): {subscription id=101 receiver
2024/05/23 14:40:36.575558274 {pubd_RO-0}{2}: [mdt-ctrl] [30214]: (note): {subscription receiver event=
2024/05/23 14:40:36.577274757 {ndbmand_RO-0}{2}: [ndbmand] [30690]: (info): get__next_table reached the
2024/05/23 14:40:36.577279206 {ndbmand_RO-0}{2}: [ndbmand] [30690]: (debug): Cleanup table for /service
2024/05/23 14:40:36.577314397 {ndbmand_RO-0}{2}: [ndbmand] [30690]: (info): get__next_object cp=ewlc-op
2024/05/23 14:40:36.577326609 {ndbmand_RO-0}{2}: [ndbmand] [30690]: (debug): yield ewlc-oper-db
2024/05/23 14:40:36.579099782 {iosrp_RO-0}{1}: [parser_cmd] [26295]: (note): id= A.B.C.D@vty0:user= cmd
2024/05/23 14:40:36.580979429 {pubd_RO-0}{2}: [pubd] [30214]: (debug): Batch response received for crim
2024/05/23 14:40:36.580988867 {pubd_RO-0}{2}: [pubd] [30214]: (debug): Green response: Result rc 0, Len
2024/05/23 14:40:36.581175013 {pubd_RO-0}{2}: [pubd] [30214]: (debug): Green Resp cursor len 63
2024/05/23 14:40:36.581176173 {pubd_RO-0}{2}: [pubd] [30214]: (debug): There is no more data left to be
2024/05/23 14:40:36.581504331 {iosrp_RO-0}{2}: [parser_cmd] [24367]: (note): id= 10.227.65.133@vty1:use
[...]
2024/05/23 14:40:37.173223406 {pubd_RO-0}{2}: [pubd] [30214]: (info): Added queue (wq: tc_inst 60293411
2024/05/23 14:40:37.173226005 {pubd_RO-0}{2}: [pubd] [30214]: (debug): New subscription (subscription 10
2024/05/23 14:40:37.173226315 {pubd_RO-0}{2}: [pubd] [30214]: (note): Added subscription for monitoring
2024/05/23 14:40:37.173230769 {pubd_RO-0}{2}: [pubd] [30214]: (debug): Stats updated for Q (wq: tc_inst
2024/05/23 14:40:37.173235969 {pubd_RO-0}{2}: [pubd] [30214]: (debug): (grpc::events) Processing event (
2024/05/23 14:40:37.173241290 {pubd_RO-0}{2}: [pubd] [30214]: (debug): GRPC telemetry connector update (
2024/05/23 14:40:37.173257944 {pubd_RO-0}{2}: [pubd] [30214]: (debug): Encoding path is Cisco-IOS-XE-pro
2024/05/23 14:40:37.173289128 {pubd_RO-0}{2}: [pubd] [30214]: (debug): Creating kvgpb encoder
2024/05/23 14:40:37.173307771 {pubd_RO-0}{2}: [pubd] [30214]: (debug): Creating combined parser
2024/05/23 14:40:37.173310050 {pubd_RO-0}{2}: [pubd] [30214]: (debug): Beginning MDT yang container wal
2024/05/23 14:40:37.173329761 {pubd_RO-0}{2}: [pubd] [30214]: (debug): Dispatching new container [data_
2024/05/23 14:40:37.173334681 {pubd_RO-0}{2}: [pubd] [30214]: (debug): Container 'Cisco-IOS-XE-process-
2024/05/23 14:40:37.173340313 {pubd_RO-0}{2}: [pubd] [30214]: (debug): add data in progress
2024/05/23 14:40:37.173343079 {pubd_RO-0}{2}: [pubd] [30214]: (debug): GRPC telemetry connector continu
2024/05/23 14:40:37.173345689 {pubd_RO-0}{2}: [pubd] [30214]: (debug): (grpc::events) Processing event (
2024/05/23 14:40:37.173350431 {pubd_RO-0}{2}: [pubd] [30214]: (debug): Dispatching new container [data_
2024/05/23 14:40:37.173353194 {pubd_RO-0}{2}: [pubd] [30214]: (debug): Deferred container cpu-utilizati
```

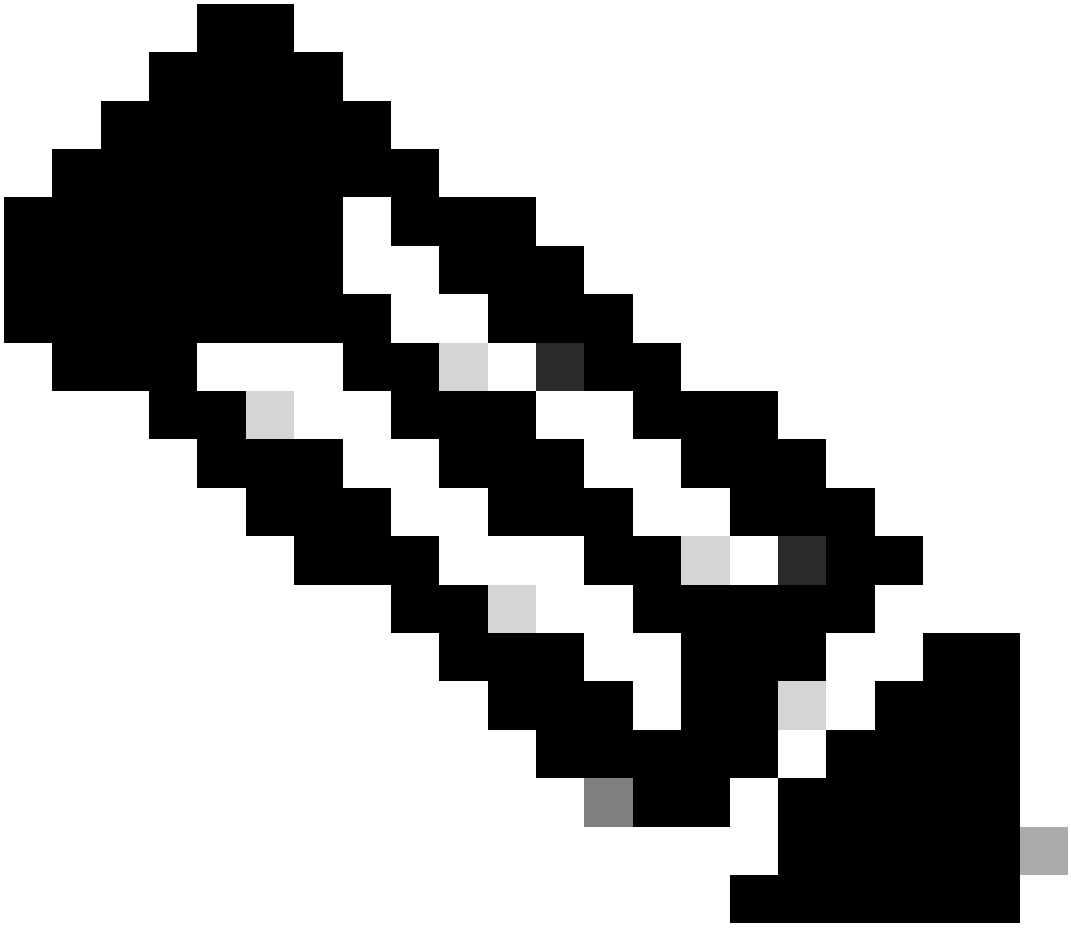
```
2024/05/23 14:40:37.173355275 {pubd_RO-0}{2}: [pubd] [30214]: (debug): Container 'cpu-utilization' star
2024/05/23 14:40:37.173380121 {pubd_RO-0}{2}: [pubd] [30214]: (debug): Dispatching new leaf [name=five-
2024/05/23 14:40:37.173390655 {pubd_RO-0}{2}: [pubd] [30214]: (debug): Leaf 'five-seconds' added succes
2024/05/23 14:40:37.173393529 {pubd_RO-0}{2}: [pubd] [30214]: (debug): add data in progress
2024/05/23 14:40:37.173395693 {pubd_RO-0}{2}: [pubd] [30214]: (debug): GRPC telemetry connector continu
2024/05/23 14:40:37.173397974 {pubd_RO-0}{2}: [pubd] [30214]: (debug): (grpc::events) Processing event
2024/05/23 14:40:37.173406311 {pubd_RO-0}{2}: [pubd] [30214]: (debug): Dispatching new leaf [name=five-
2024/05/23 14:40:37.173408937 {pubd_RO-0}{2}: [pubd] [30214]: (debug): Leaf 'five-seconds-intr' added s
2024/05/23 14:40:37.173411575 {pubd_RO-0}{2}: [pubd] [30214]: (debug): add data in progress
[...]
```

Asegurarse de que las métricas alcanzan la pila TIG

Desde InfluxDB CLI

Al igual que cualquier otro sistema de base de datos, InfluxDB viene con una CLI que se puede utilizar para verificar que las métricas son recibidas correctamente por Telegraf y almacenadas en la base de datos definida. InfluxDB organiza las métricas, llamadas puntos, en mediciones que se organizan como series. Algunos comandos básicos presentados aquí se pueden utilizar para verificar el esquema de datos en InfluxDB y asegurarse de que los datos lleguen a esta aplicación.

En primer lugar, puede comprobar que las series, las medidas y su estructura (claves) se generan correctamente. Telegraf e InfluxDB las generan automáticamente en función de la estructura del RPC utilizado.



Nota: Por supuesto, esta estructura es totalmente personalizable a partir de las configuraciones Telegraf e InfluxDB. Sin embargo, esto va más allá del alcance de esta guía de configuración.

```
$ influx
Connected to http://localhost:8086 version 1.6.7~rc0
InfluxDB shell version: 1.6.7~rc0
> USE TELEGRAF
Using database TELEGRAF
> SHOW SERIES
key
---
Cisco-IOS-XE-process-cpu-oper:cpu-usage/cpu-utilization,host=ubuntu-virtual-machine,path=Cisco-IOS-XE-p
> SHOW MEASUREMENTS
name: measurements
name
----
Cisco-IOS-XE-process-cpu-oper:cpu-usage/cpu-utilization
> SHOW FIELD KEYS FROM "Cisco-IOS-XE-process-cpu-oper:cpu-usage/cpu-utilization"
name: Cisco-IOS-XE-process-cpu-oper:cpu-usage/cpu-utilization
```

```
fieldKey                                                fieldType
--------                                                ---------
cpu_usage_processes/cpu_usage_process/avg_run_time      integer
cpu_usage_processes/cpu_usage_process/five_minutes      float
cpu_usage_processes/cpu_usage_process/five_seconds      float
cpu_usage_processes/cpu_usage_process/invocation_count integer
cpu_usage_processes/cpu_usage_process/name              string
cpu_usage_processes/cpu_usage_process/one_minute        float
cpu_usage_processes/cpu_usage_process/pid               integer
cpu_usage_processes/cpu_usage_process/total_run_time    integer
cpu_usage_processes/cpu_usage_process/tty               integer
five_minutes                                            integer
five_seconds                                            integer
five_seconds_intr                                       integer
one_minute                                              integer
```

Una vez aclarada la estructura de datos (entero, cadena, booleano, ...), se puede obtener el número de puntos de datos que se almacenan en estas mediciones basadas en un campo determinado.

```
# Get the number of points from "Cisco-IOS-XE-process-cpu-oper:cpu-usage/cpu-utilization" for the field
> SELECT COUNT(five_seconds) FROM "Cisco-IOS-XE-process-cpu-oper:cpu-usage/cpu-utilization"
name: Cisco-IOS-XE-process-cpu-oper:cpu-usage/cpu-utilization
time count
---- -----
0    1170
> SELECT COUNT(five_seconds) FROM "Cisco-IOS-XE-process-cpu-oper:cpu-usage/cpu-utilization"
name: Cisco-IOS-XE-process-cpu-oper:cpu-usage/cpu-utilization
time count
---- -----
0    1171

# Fix timestamp display
> precision rfc3339
# Get the last point stored in "Cisco-IOS-XE-process-cpu-oper:cpu-usage/cpu-utilization" for the field
> SELECT LAST(five_seconds) FROM "Cisco-IOS-XE-process-cpu-oper:cpu-usage/cpu-utilization"
name: Cisco-IOS-XE-process-cpu-oper:cpu-usage/cpu-utilization
time                    last
----                    ----
2024-05-23T13:18:53.51Z 0
> SELECT LAST(five_seconds) FROM "Cisco-IOS-XE-process-cpu-oper:cpu-usage/cpu-utilization"
name: Cisco-IOS-XE-process-cpu-oper:cpu-usage/cpu-utilization
time                     last
----                     ----
2024-05-23T13:19:03.589Z 2
```

Si el número de puntos para un campo determinado y la marca de tiempo para la última aparición aumentan, es buena señal que la pila TIG recibe y almacena correctamente los datos enviados por el WLC.

De Telegraf

Para verificar que el receptor de Telegraf realmente obtenga algunas métricas del controlador y verifique su formato, puede redirigir las métricas de Telegraf a un archivo de salida en el host. Esto puede ser muy útil cuando se trata de la resolución de problemas de interconexión de dispositivos. Para lograr esto, simplemente haga uso del plugin de salida "archivo" de Telegraf, configurable desde el /etc/telegraf/telegraf.conf.

```
# Send telegraf metrics to file(s)
[[outputs.file]]
#   ## Files to write to, "stdout" is a specially handled file.
   files = ["stdout", "/tmp/metrics.out", "other/path/to/the/file"]
#
#   ## Use batch serialization format instead of line based delimiting.  The
#   ## batch format allows for the production of non line based output formats and
#   ## may more efficiently encode metric groups.
#   # use_batch_format = false
#
#   ## The file will be rotated after the time interval specified.  When set
#   ## to 0 no time based rotation is performed.
#   # rotation_interval = "0d"
#
#   ## The logfile will be rotated when it becomes larger than the specified
#   ## size.  When set to 0 no size based rotation is performed.
#   # rotation_max_size = "0MB"
#
#   ## Maximum number of rotated archives to keep, any older logs are deleted.
#   ## If set to -1, no archives are removed.
#   # rotation_max_archives = 5
#
#   ## Data format to output.
#   ## Each data format has its own unique set of configuration options, read
#   ## more about them here:
#   ## https://github.com/influxdata/telegraf/blob/master/docs/DATA_FORMATS_OUTPUT.md
   data_format = "influx"
```

Referencias

Directrices de dimensionamiento de hardware

Requisitos de Grafana

Acerca de esta traducción
Cisco ha traducido este documento combinando la traducción automática y los recursos humanos a fin de ofrecer a nuestros usuarios en todo el mundo contenido en su propio idioma.

Tenga en cuenta que incluso la mejor traducción automática podría no ser tan precisa como la proporcionada por un traductor profesional.

Cisco Systems, Inc. no asume ninguna responsabilidad por la precisión de estas traducciones y recomienda remitirse siempre al documento original escrito en inglés (insertar vínculo URL).