Catalyst 9800でのTelegraf、InfluxDB、およびGrafanaを使用した高度なgRPCワークフローの設定

内容

はじめに

このドキュメントでは、Telegraf、InfluxDB(ECD)、およびGrafana(TIG)スタックを導入し、Catalyst 9800と相互接続する方法について説明します。

前提条件

このドキュメントでは、複雑な統合によるCatalyst 9800のプログラマチックインターフェイス機能について説明します。このドキュメントの目的は、これらをニーズに基づいて完全にカスタマイズし、毎日の時間を節約する方法を示すことです。ここに示す導入はgRPCに依存し、Catalyst 9800からのワイヤレスデータをTelegraf、InfluxDB、Grafana(TIG)監視可能スタックで使用できるようにするテレメトリ設定を提示します。

要件

次の項目に関する知識があることが推奨されます。

- Catalyst Wireless 9800設定モデル。
- ネットワークプログラマビリティとデータモデル
- TIGスタックの基本。

使用するコンポーネント

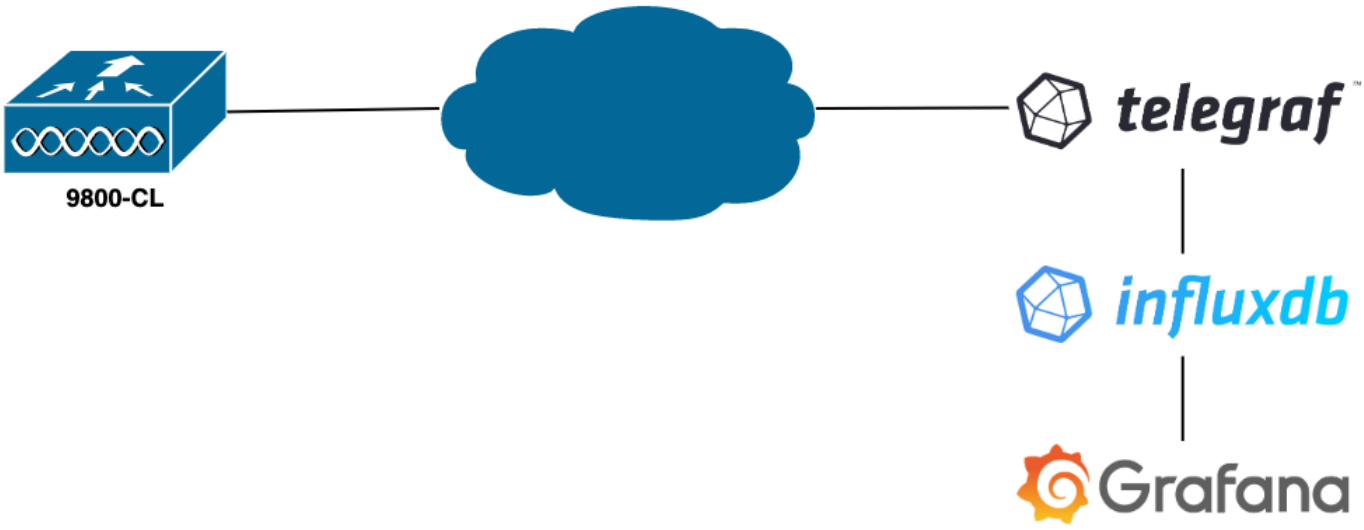
このドキュメントの情報は、次のソフトウェアとハードウェアのバージョンに基づいています。

- Catalyst 9800-CL(v. 17.12.03)
- Ubuntu (v. 22.04.03)。
- InfluxDB (v. 1.06.07)。
- テレグラフ(v. 1.21.04)
- グラファナ(v. 10.02.01)

このドキュメントの情報は、特定のラボ環境にあるデバイスに基づいて作成されました。このドキュメントで使用するすべてのデバイスは、クリアな（デフォルト）設定で作業を開始しています。本稼働中のネットワークでは、各コマンドによって起こる可能性がある影響を十分確認してください。

設定

ネットワーク図



コンフィギュレーション

この例では、9800-CLでgRPCダイヤルアウトを使用してテレメトリを設定し、Telegrafアプリケーションの情報をInfluxDBデータベースに格納します。ここでは、2つのデバイスが使用されています。

- TIGスタック全体をホストするUbuntuサーバ。
- Catalyst 9800-CLを使用します。

この設定ガイドでは、これらのデバイスの導入全体を扱うのではなく、9800情報の送信、受信、および表示を適切に行うために各アプリケーションで必要な設定を扱います。

ステップ 1：データベースの準備

設定部分に入る前に、Influxインスタンスが正しく動作していることを確認します。これは、Linuxディストリビューションを使用している場合は、systemctl status コマンドを使用して簡単に実行できます。

```
admin@tig:~$ systemctl status influxd ● influxdb.service - InfluxDB is an open-source, distributed, time serie
```

この例が機能するには、Telegrafにメトリックを保存するデータベースと、このデータベースに接続するユーザが必要です。これらは、次のコマンドを使用して、InfluxDB CLIから簡単に作成できます。

```
admin@tig:~$ influx Connected to http://localhost:8086 version 1.8.10 InfluxDB shell version: 1.8.10 > 
```

これでデータベースが作成され、Telegrafにメトリックを適切に保存するように設定できます。

ステップ 2：Telegrafの準備

この例では、2つのTelegraf設定だけが動作します。これらの設定は（Unix上で動作するアプリケーションの場合は通常どおり）コンフィ/etc/telegraf/telegraf.conf ギュレーションファイルから行うことができます。

最初のコマンドは、Telegrafが使用する出力を宣言します。前述したように、ここではInfluxDBが使用されており、telegraf.conf ファイルの出力セクションで次のように設定されています。

```
############################################################################### # OUTPUT PLUGINS # ####
```

これは、受信したデータをポート8086上の同じホスト上で実行されているInfluxDBに保存し、「TELEGRAF」と呼ばれるデータベース（およびアクセスするための資格情報telegraf/YOUR_PASSWORD）を使用するようにTelegrafプロセスに指示します。

最初に宣言されたものが出力フォーマットであった場合、2番目のフォーマットは当然、入力フォーマットです。テレメトリを使

用するシスコデバイスから受信したデータであることをTelegrafに通知するには、cisco_telemetry_mdt入力モジュールを使用します。これを設定するには、/etc/telegraf/telegraf.conf ファイルに次の行を追加するだけです。

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

これにより、ホスト(デフォルトポート57000)で実行されているTelegrafアプリケーションは、WLCから受信したデータをデコードできます。

設定を保存したら、Telegrafを再起動してサービスに適用します。また、サービスが正しく再起動されていることを確認します。

```
admin@tig:~$ sudo systemctl restart telegraf admin@tig:~$ systemctl status telegraf.service ● telegraf.s
```

ステップ 3：必要なメトリックを含むテレメトリサブスクリプションの判別

前述のとおり、他の多くのシスコデバイスと同様に、メトリックはYANGモデルに従って編成されます。IOS XEの各バージョン（9800で使用）に対応する特定のCisco YANGモデルについては、ここを参照してください。この例で使用されているIOS XE Dublin 17.12.03に対応するモデルを参照してください。

この例では、使用されている9800-CLインスタンスからCPU使用率メトリックを収集することに焦点を当てます。Cisco IOS XEダブリン17.12.03のYANGモデルを調べることで、どのモジュールにコントローラのCPU使用率、特に最後の5秒間が含まれているかを確認できます。これらは、CPU使用率グループ（リーフ5秒）下のCisco-IOS-XE-process-cpu-operモジュールの一部です。

ステップ 4：コントローラでNETCONFを有効にする

gRPCダイヤルアウトフレームワークは、NETCONFを使用して同じように動作します。したがって、この機能を9800で有効にする必要があります。有効にするには、次のコマンドを実行します。

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

ステップ 5：コントローラでのテレメトリサブスクリプションの設定

YANGモデルから決定されたメトリックのXPaths(**別名*XML*パス言語**)が終了したら、9800 CLIから簡単にテレメトリサブスクリプションを設定し、これらのサブスクリプションをステップ2で設定したTelegrafインスタンスにストリーミングできます。これを行うには、次のコマンドを実行します。

```
WLC(config)#telemetry ietf subscription 101 WLC(config-mdt-subs)#encoding encode-kvgpb WLC(config-mdt-s
```

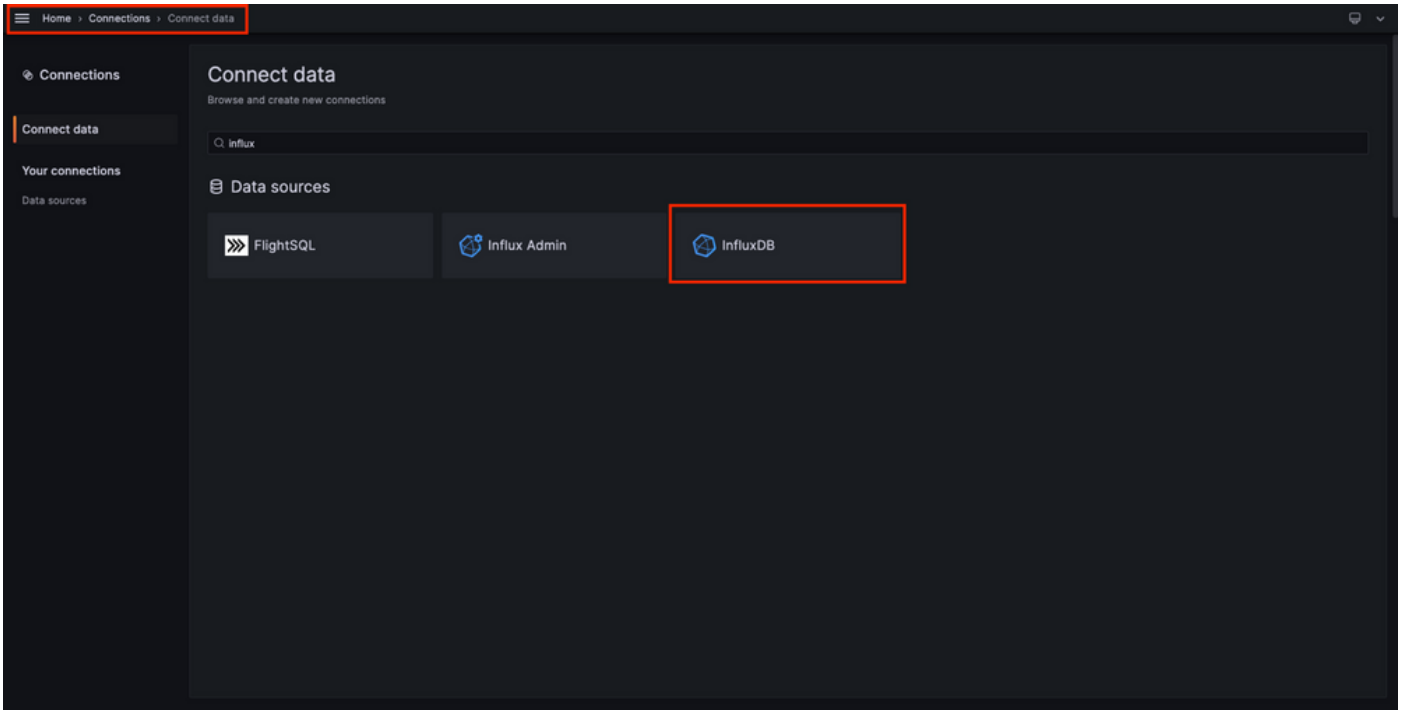
このコードブロックでは、最初に識別子101のテレメトリサブスクリプションが定義されます。サブスクリプションIDは、別のサブスクリプションと重複しない限り、<0 ~ 2147483647>の任意の数字にすることができます。このサブスクリプションは、次の順序で設定します。

- 使用するエンコード方式。gRPC転送プロトコルを使用する場合はkvGPBである必要があります。
- サブスクリプションによって送信されるメトリックのフィルタ。対象のメトリックを定義するXPath(つまり、 /process-cpu-ios-xe-oper:cpu-usage/cpu-utilization/five-seconds)です。
- コントローラがメトリックを送信するために使用する送信元IPアドレス。
- メトリックの通信に使用されるストリーム・タイプ（この場合はYANGプッシュIETF標準）。
- 100 秒でサブスクライバにデータを送信するためにコントローラが使用する周波数。この例では、アップデートを毎秒定期的に送信するように設定されています。
- レシーバのIPアドレスとポート番号、およびコントローラとサブスクライバ間の通信に使用されるプロトコル。この例では、gRPC-TCPを使用して、ポート57000のホスト10.48.39.98にメトリックを送信します。

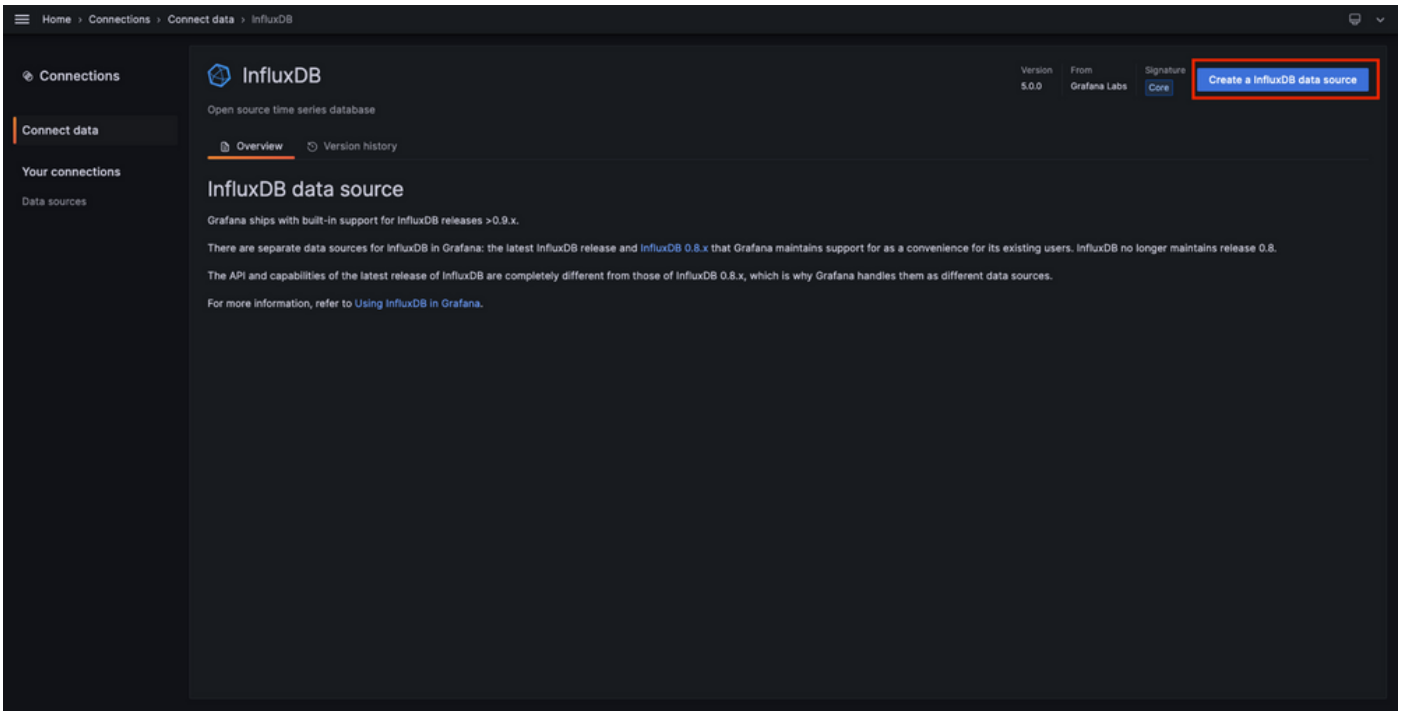
手順 6：Grafanaデータソースの構成

コントローラがTelegrafへのデータ送信を開始し、データがTELEGRAF InfluxDBデータベースに保存されました。次に、これらのメトリックを参照するようにGrafanaを設定します。

Grafana GUIから*Home* > *Connections* > *Connect data*の順に移動し、検索バーを使用してInfluxDBデータソースを見つけます。

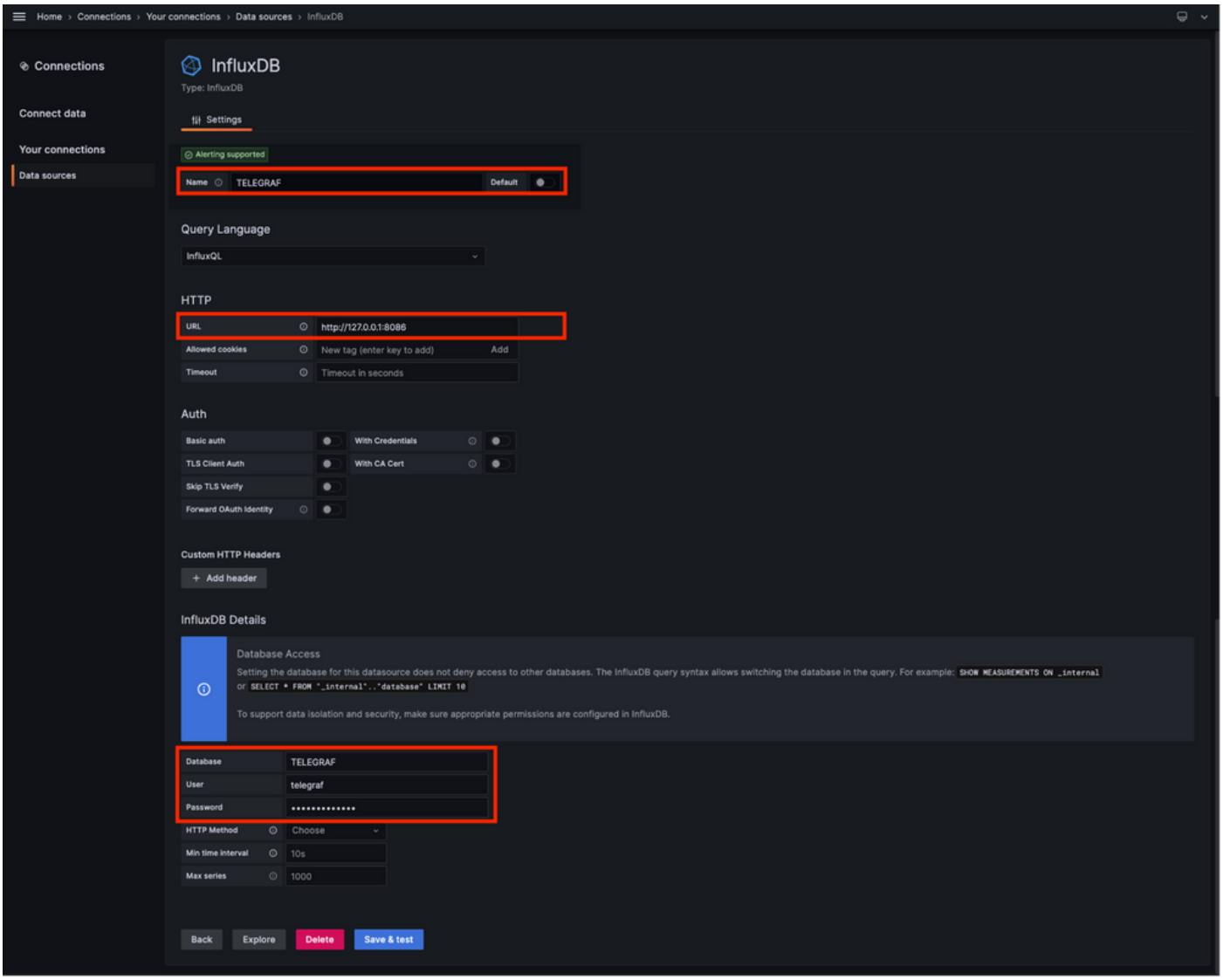


このデータソースタイプを選択し、「InfluxDBデータソースの作成」ボタンを使用してGrafanaとステップ1で作成したTELEGRAPHデータベースを接続します。



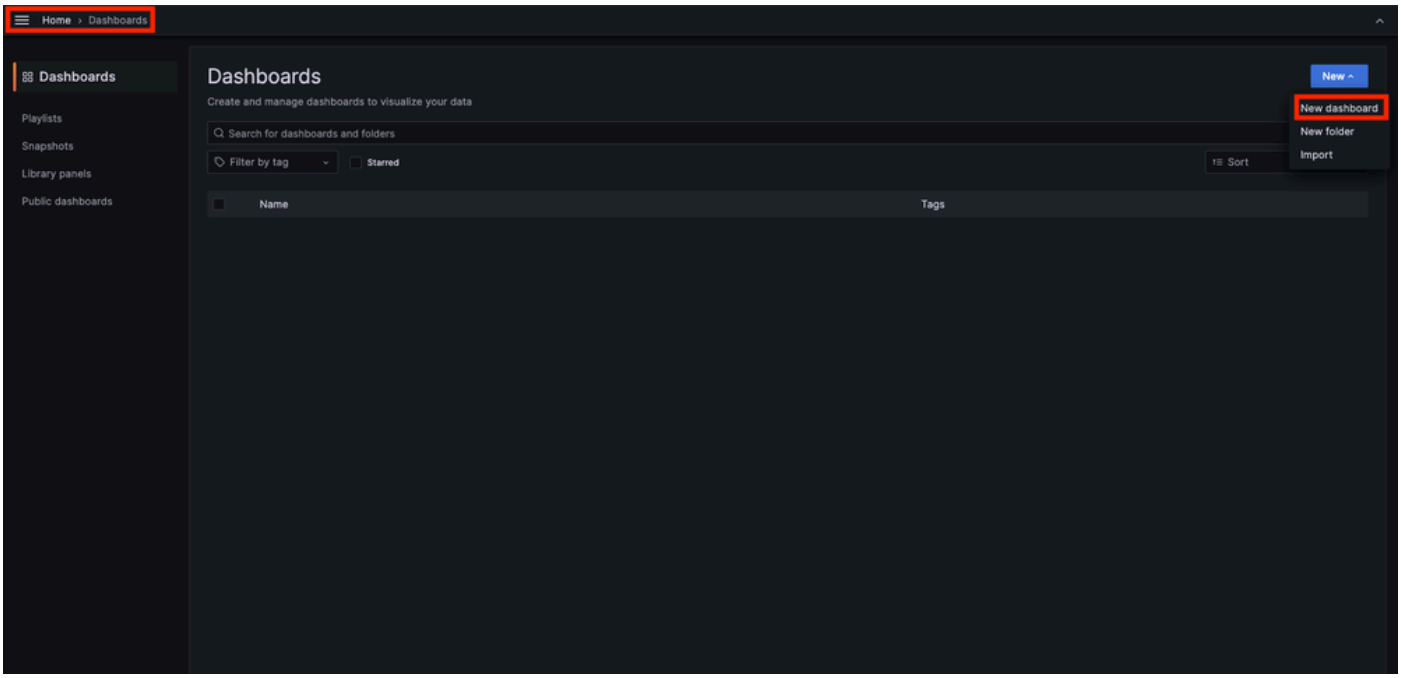
画面に表示されるフォームに入力します。特に、次の情報を入力します。

- データソースの名前。
- 使用されるInfluxDBインスタンスのURL。
- 使用するデータベース名（この例では「TELEGRAF」）。
- アクセスするために定義されたユーザのクレデンシャル（この例ではtelegraf/YOUR_PASSWORD）。

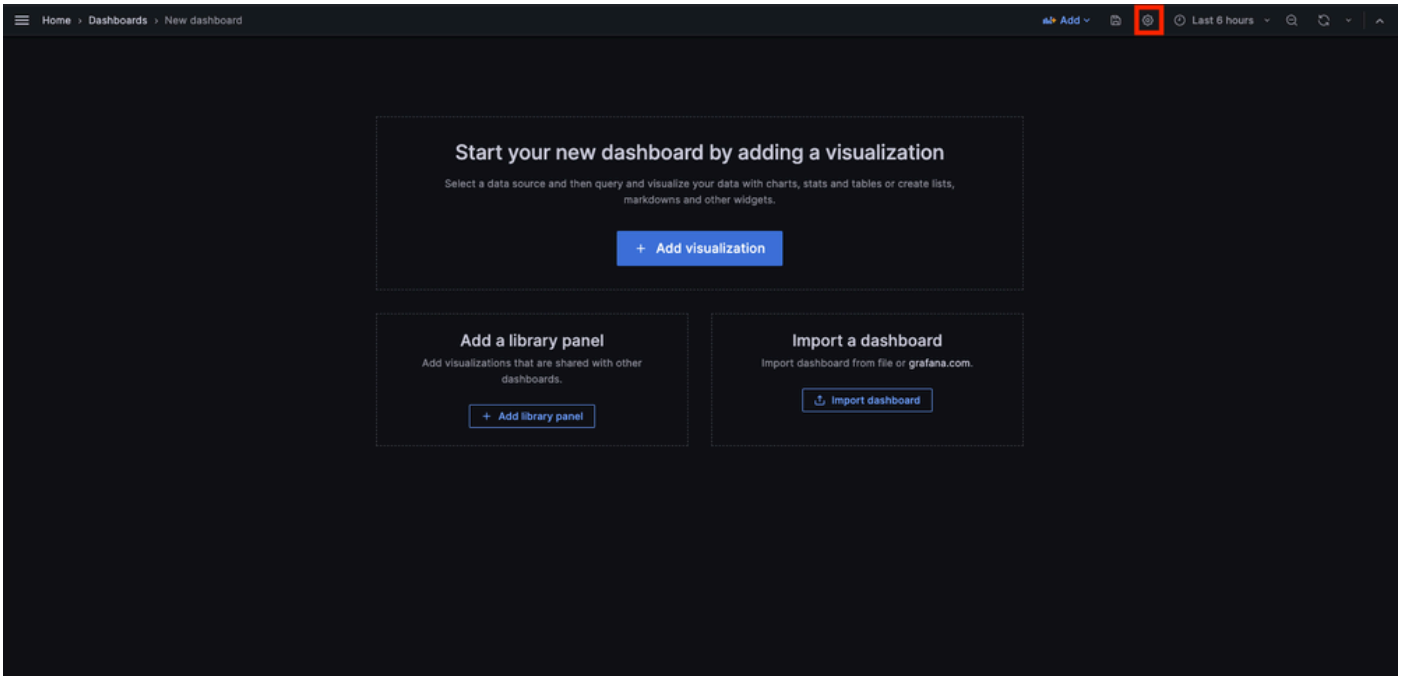


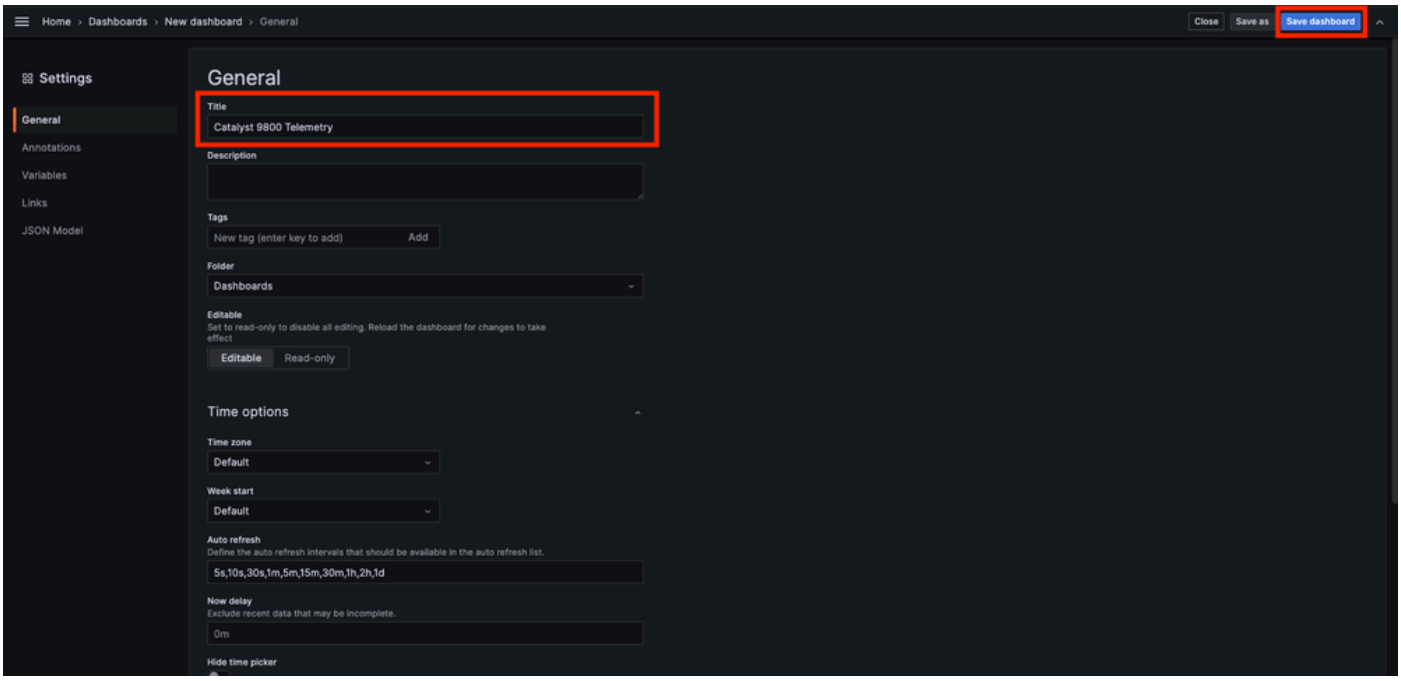
手順 7 : ダッシュボードの作成

Grafanaのビジュアライゼーションは、ダッシュボードに整理されています。Catalyst 9800のメトリックの視覚化を含むダッシュボードを作成するには、ホーム>ダッシュボードに移動し、「新規ダッシュボード」ボタンを使用します



作成した新しいダッシュボードが開きます。歯車アイコンをクリックしてダッシュボードパラメータにアクセスし、その名前を変更します。この例では、「Catalyst 9800テレメトリ」が使用されています。これを実行したら、「ダッシュボードの保存」ボタンを使用してダッシュボードを保存します。

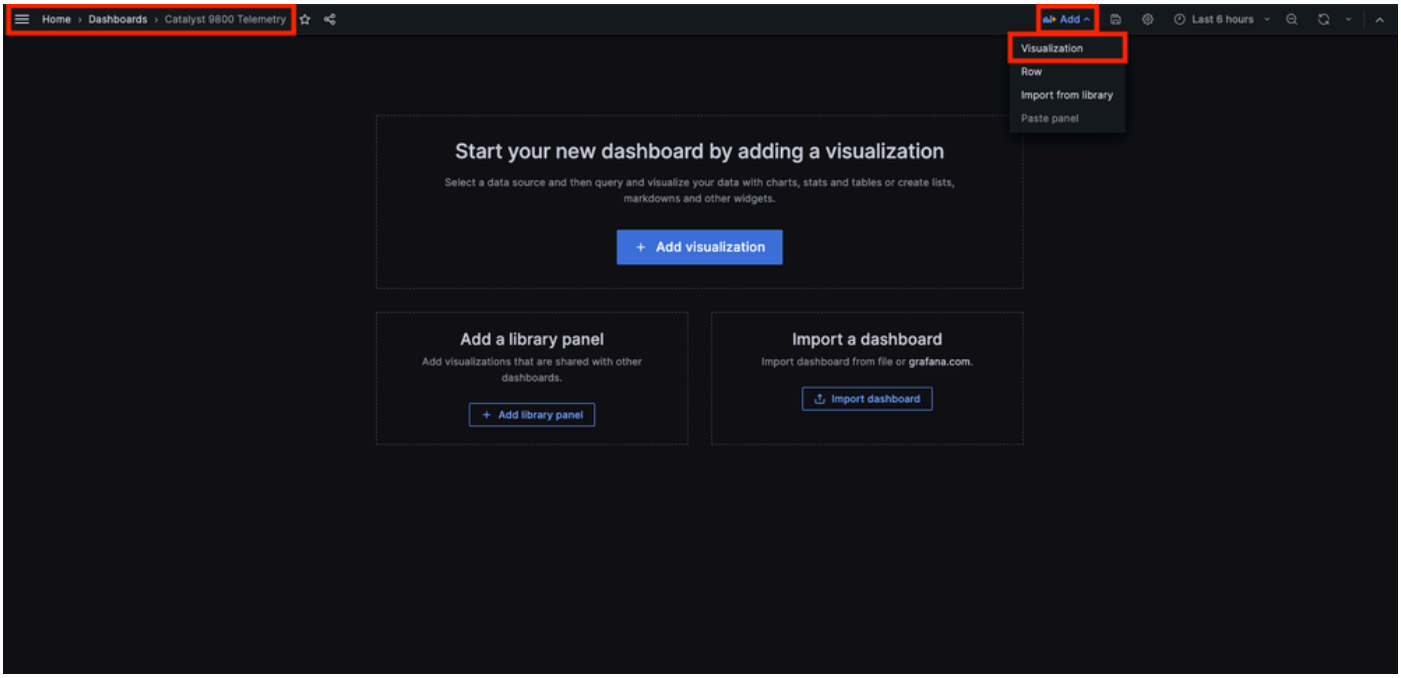




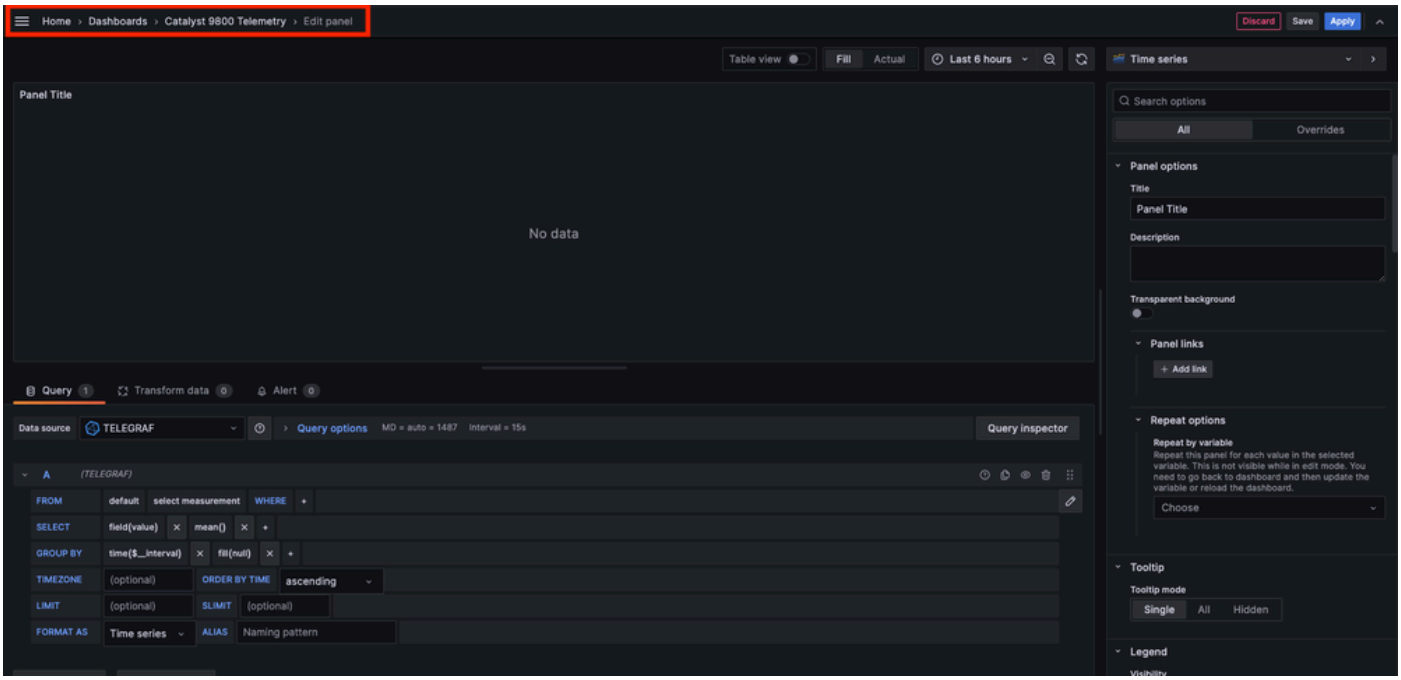
ステップ 8：ダッシュボードへのビジュアル化の追加

データが適切に送信、受信、保存され、Grafanaがこのストレージの場所にアクセスできるようになったので、次にビジュアライゼーションを作成します。

任意のGrafanaダッシュボードから「追加」ボタンを使用して、表示されるメニューから「可視化」を選択し、メトリクスの可視化を作成します。

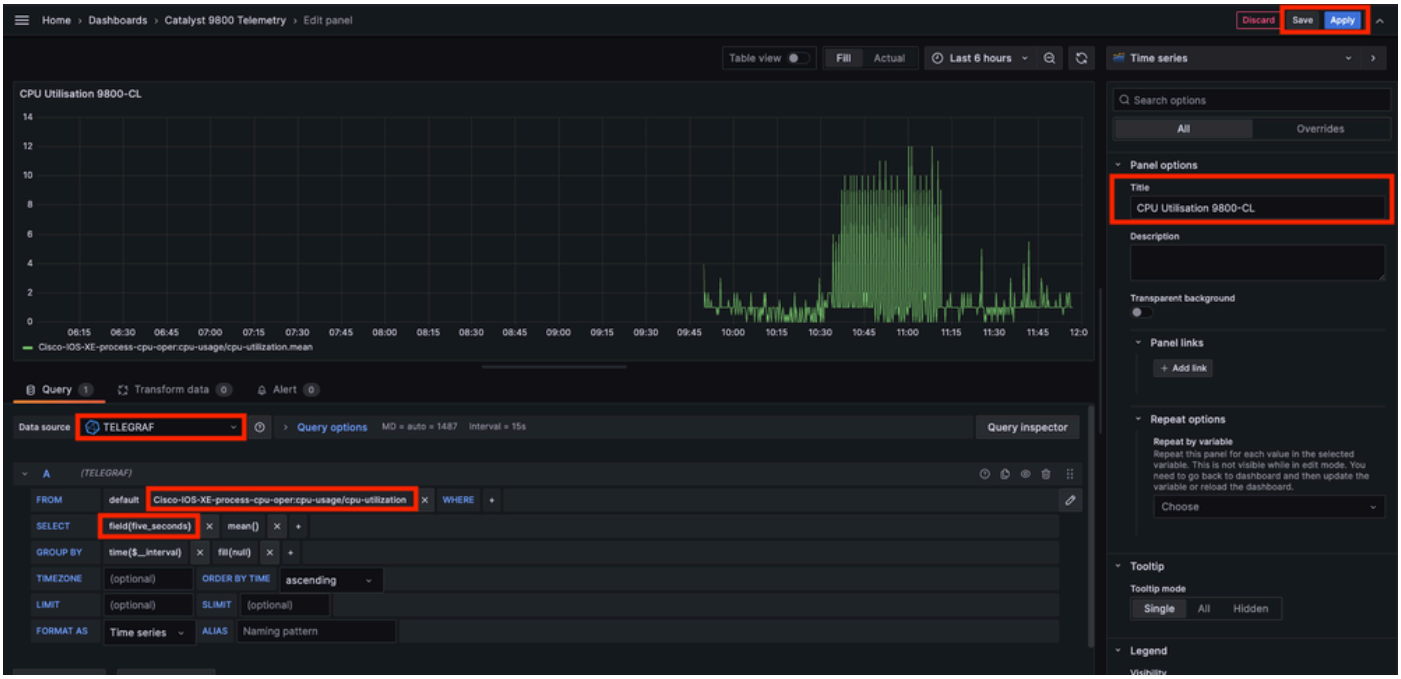


作成したビジュアル化の編集パネルが開きます。



このパネルから、

- ステップ6で作成したデータソースの名前。この例ではTELEGRAFです。
- 視覚化するデータを含む測定（スキーマ）は、この例では「Cisco-IOS-XE-process-cpu-oper:cpu-usage/cpu-utilization」です。
- 視覚化するメトリックを表すデータベースのフィールド。この例では「five_seconds」です。
- この例では、視覚化のタイトルは「CPU Utilization 9800-CL」です。



前の図の「Save/Apply」ボタンを押すと、Catalyst 9800コントローラのCPU使用率を時系列で示す図がダッシュボードに追加され

ます。ダッシュボードに加えた変更は、フロッピーディスクのアイコンボタンを使用して保存できます。

確認

WLCの実行コンフィギュレーション

```
Building configuration... Current configuration : 112215 bytes ! ! Last configuration change at 14:28:3
```

Telegrafの設定

```
# Configuration for telegraf agent [agent] metric_buffer_limit = 10000 collection_jitter = "0s" debug =
```

InfluxDBの設定

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

Grafanaの設定

```
################################### Server ###################################### [server] http_addr = 1
```

トラブルシュート

WLCワンストップショップリフレックス

WLC側で最初に確認することは、プログラム可能なインターフェイスに関連するプロセスが稼働していることです。

```
#show platform software yang-management process confd : Running nesd : Running syncfd : Running ncsshd
```

NETCONF（gRPCダイヤルアウトで使用）の場合、これらのコマンドはプロセスのステータスのチェックにも役立ちます。

```
WLC#show netconf-yang status netconf-yang: enabled netconf-yang candidate-datastore: disabled netconf-y
```

プロセスのステータスを確認した後、もう1つの重要な確認は、Catalyst 9800とTelegrafレシーバ間のテレメトリ接続のステータスです。これは「show telemetry connection all」コマンドを使用して表示できます。

```
WLC#show telemetry connection all Telemetry connections Index Peer Address Port VRF Source Address Stat
```

WLCと受信側の間でテレメトリ接続がアップしている場合は、show telemetry ietf subscription all brief コマンドを使用して、設定されているサブスクリプションが有効であることも確認できます。

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

このコマンドの詳細バージョンである show telemetry ietf subscription all detailを使用すると、サブスクリプションに関する詳細情報が表示され、設定の問題を簡単に指摘できます。

```
WLC#show telemetry ietf subscription all detail Telemetry subscription detail: Subscription ID: 101 Typ
```

ネットワーク到達可能性の確認

Catalyst 9800コントローラは、テレメトリサブスクリプションごとに設定されたレシーバポートにgRPCデータを送信します。

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

この設定済みポート上のWLCとレシーバ間のネットワーク接続を確認するには、いくつかのツールを使用できます。

WLCから、設定済みのレシーバIP/ポート(ここでは10.48.39.98:57000)でTelnetを使用して、このIPアドレスが開いていて、コントローラ自体から到達可能であることを確認できます。トラフィックがブロックされていない場合、ポートは出力でオープンとして表示される必要があります。

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

または、任意のホストからNmapを使用して、設定したポートで受信側が正しく認識されるようにすることもできます。

```
$ sudo nmap -sU -p 57000 10.48.39.98 Starting Nmap 7.95 ( https://nmap.org ) at 2024-05-17 13:12 CEST Nm
```

ロギングとデバッグ

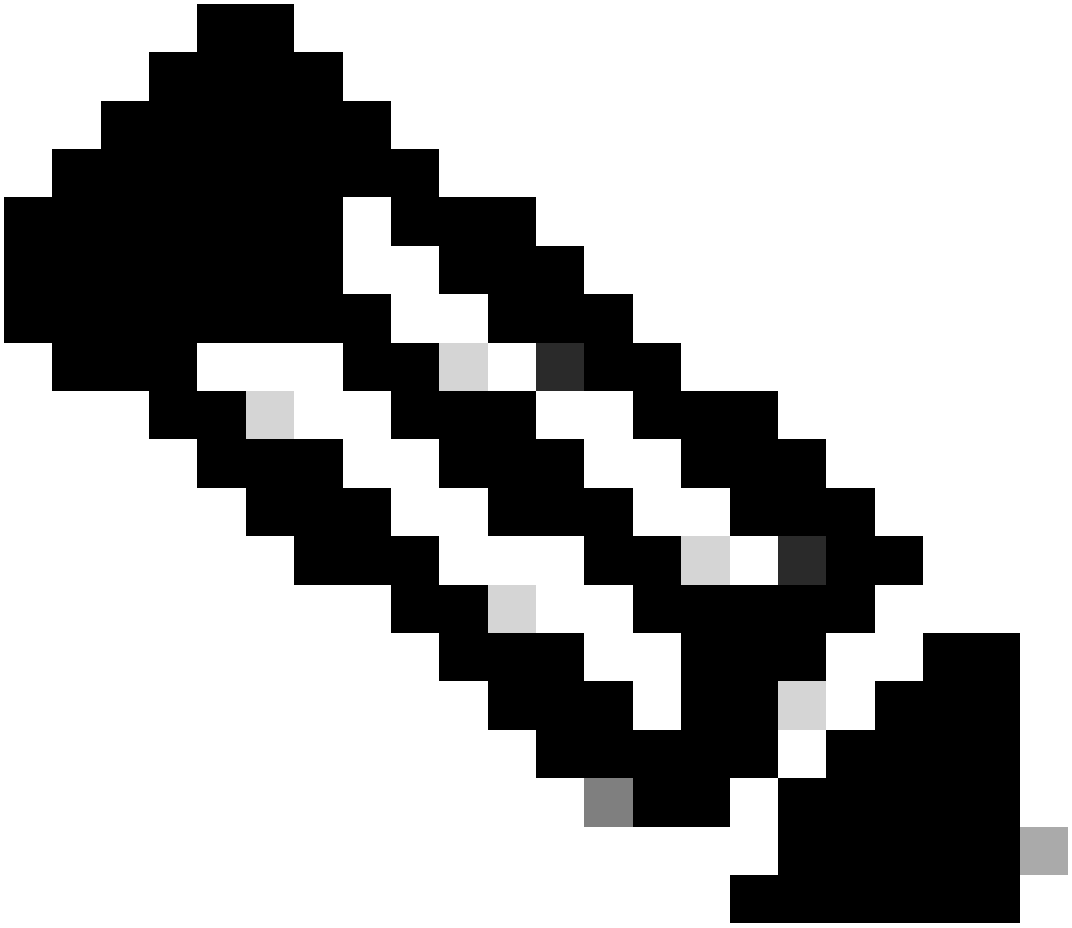
```
2024/05/23 14:40:36.566486156 {pubd_RO-0}{2}: [mdt-ctrl] [30214]: (note): **** Event Entry: Configured
```

メトリックがTIGスタックに到達することの確認

InfluxDB CLIから

他のデータベースシステムと同様に、InfluxDBにはCLIが付属しており、メトリクスがTelegrafによって正しく受信され、定義されたデータベースに保存されているかどうかを確認するために使用できます。InfluxDBは、ポイントと呼ばれるメトリックを、それ自体がシリーズとして編成された測定値に編成します。ここで説明する基本的なコマンドの一部は、InfluxDB側のデータスキームを検証し、データがこのアプリケーションに到達することを確認するために使用できます。

最初に、シリーズ、測定値、およびそれらの構造（キー）が正しく生成されていることを確認できます。これらは、使用されるRPCの構造に基づいて、TelegrafとInfluxDBによって自動的に生成されます。



注：この構造は、TelegrafおよびInfluxDBの設定から完全にカスタマイズできます。ただし、これはこの設定ガイドの対象範囲外です。

```
$ influx Connected to http://localhost:8086 version 1.6.7~rc0 InfluxDB shell version: 1.6.7~rc0 > USE T
```

データ構造（整数、文字列、ブール値、...）が明確になると、特定のフィールドに基づいてこれらの測定値に格納されているデータポイントの数を取得できます。

```
# Get the number of points from "Cisco-IOS-XE-process-cpu-oper:cpu-usage/cpu-utilization" for the field
```

特定のフィールドのポイント数と最後のオカレンスのタイムスタンプが増加する場合、TIGスタックがWLCによって送信されたデータを適切に受信して保存することは適切な兆候です。

Telegrafから

Telegraf受信側が実際にコントローラからメトリックを取得し、その形式を確認したことを確認するには、Telegrafメトリックをホスト上の出力ファイルにリダイレクトします。これは、デバイスの相互接続のトラブルシューティングに非常に便利です。これを行うには、 /etc/telegraf/telegraf.confから設定可能なTelegrafの「file」出力プラグインを使用します。

```
# Send telegraf metrics to file(s) [[outputs.file]] # ## Files to write to, "stdout" is a specially han
```

参考資料

ハードウェア規模評価ガイドライン

Grafanaの要件

翻訳について
シスコは世界中のユーザにそれぞれの言語でサポート コンテンツを提供するために、機械と人による翻訳を組み合わせて、本ドキュメントを翻訳しています。ただし、最高度の機械翻訳であっても、専門家による翻訳のような正確性は確保されません。シスコは、これら翻訳の正確性について法的責任を負いません。原典である英語版（リンクからアクセス可能）もあわせて参照することを推奨します。