**Monitoring VPN tunnels in Prometheus and Grafana using Cloudwatch-exporter**

We often need to collect custom metrics such as memory usage, disk usage etc for right-sizing or daily monitoring: We install agent on running servers and let agent collect data there and send it over to cloudwatch. VPN tunnel is an important aspect of VSOC (Virtual Service Operation Centre) which gives us insights about the state of tunnel to a specific cluster/customer VPN. A Prometheus CloudWatch exporter is a key element for anyone wanting to monitor AWS CloudWatch. Exporting CloudWatch metrics to a Prometheus server allows leveraging of the power of PromQL queries, integrating AWS metrics with those from other applications or cloud providers, and creating advanced dashboards for digging down into problems.

Monitoring is an important part of maintaining the reliability, availability, and performance of your AWS Site-to-Site VPN connection. You should collect monitoring data from all of the parts of your solution so that you can more easily debug a multi-point failure if one occurs. Before you start monitoring your Site-to-Site VPN connection; however, you should create a monitoring plan that includes answers to the following questions:

* What are your monitoring goals?
* What resources will you monitor?
* How often will you monitor these resources?
* What monitoring tools will you use?
* Who will perform the monitoring tasks?
* Who should be notified when something goes wrong?

The next step is to establish a baseline for normal VPN performance in your environment, by measuring performance at various times and under different load conditions. As you monitor your VPN, store historical monitoring data so that you can compare it with current performance data, identify normal performance patterns and performance anomalies, and devise methods to address issues.

To establish a baseline, you should monitor the following items:

* The state of your VPN tunnels
* Data into the tunnel
* Data out of the tunnel

**Steps to set-up Cloudwatch-Exporter (For VPN/any metrics for that matter) in already existing AWS infrastructure:**

**Step 1**: Clone the Cloudwatch-Exporter repository to your local system by using https or ssh. The following is a https link <https://github.com/prometheus/cloudwatch_exporter.git> , the following is the ssh link to clone the repository <git@github.com:prometheus/cloudwatch_exporter.git>

**Step 2:** Open the repository in any of the IDE, in my case I had used IntelliJ. Create a new file by name config.yaml, this is the file in which you would be putting in the metrics that you are intending to scrape out of AWS CloudWatch. The following is the example config file given in the repository:

---

region: eu-west-1

metrics:

- aws\_namespace: AWS/ELB

aws\_metric\_name: RequestCount

aws\_dimensions: [AvailabilityZone, LoadBalancerName]

aws\_dimension\_select:

LoadBalancerName: [myLB]

aws\_statistics: [Sum]

In my case, where in I wanted to export/scrape VPN metrics the following was the config file:

---  
region: eu-central-1  
metrics:  
 - aws\_namespace: AWS/VPN  
 aws\_metric\_name: TunnelState  
 aws\_dimensions: [TunnelIpAddress]  
 aws\_dimension\_select:  
 LoadBalancerName: [xx.xx.xx.xx]  
  
 - aws\_namespace: AWS/VPN  
 aws\_metric\_name: TunnelDataIn  
 aws\_dimensions: [TunnelIpAddress]  
 aws\_dimension\_select:  
 LoadBalancerName: [xx.xx.xx.xx]  
  
 - aws\_namespace: AWS/VPN  
 aws\_metric\_name: TunnelDataOut  
 aws\_dimensions: [TunnelIpAddress]  
 aws\_dimension\_select:  
 LoadBalancerName: [xx.xx.xx.xx]  
  
 - aws\_namespace: AWS/ClientVPN  
 aws\_metric\_name: ActiveConnectionsCount  
 aws\_dimensions: [Endpoint]  
 aws\_dimension\_select:  
 LoadBalancerName: [cvpn-endpoint-xxxxxxxxxxxxxxx]

The available Cloudwatch metrics of AWS can be fetched by the following command:

aws cloudwatch list-metrics --namespace "AWS/SNS"

In our case the namespaces were AWS/VPN and AWS/ClientVPN.

**Step 3:** Once the config file is created, in the Docker file add the below, so that the config.yml file is mounted to the image while it is getting built

FROM prom/cloudwatch-exporter  
ADD config.yml /config/

**Step 4:** Once the above changes are made, you can go ahead and build the docker image of the Cloudwatch-Exporter. The same image can be pushed to docker hub so that the image can be pulled while the pod is created.

**Step 5:** Add the below in the deployment file(yaml file) of the exporter so that Prometheus is created as a target.

spec:

  progressDeadlineSeconds: 600

  replicas: 1

  revisionHistoryLimit: 10

  selector:

   matchLabels:

    app: cloudwatch-exporter

  strategy:

   rollingUpdate:

    maxSurge: 25%

    maxUnavailable: 25%

   type: RollingUpdate

  template:

   metadata:

    creationTimestamp: null

    labels:

     app: cloudwatch-exporter

annotations:

prometheus.io/scrape: true

prometheus.io/port: 9106

Now modify the config map or values.yaml for the metrics to be picked by Prometheus. By completing this step, one must be able to see the metrics on the Prometheus dashboard. Below are the few images from queried VPN metrics Prometheus Dashboard.

**Note:**  One use the following docker image customized for VPN metrics from [docker pull faraaz2709/cloudwatch-exporter:v3](docker%20pull%20faraaz2709/cloudwatch-exporter:v3) and use kubectl run command to create the pod

Chart

Description automatically generated

**Image 1.** The above image shows the VPN tunnel state, with this one can infer if the tunnel to a particular site is UP or DOWN

Table

Description automatically generated

**Image 2.** In the above image one can see the tunnel Data In

A picture containing graphical user interface

Description automatically generated

**Image 3.** In the above Image one can see the Tunnel Data Out

Graphical user interface, application, table, Excel

Description automatically generated

**Image 4.** In the above image one can see the Client VPN Active Connections Count

With the above-mentioned steps one can visualize the VPN metrics on Prometheus. Now to get the data/metrics in **Grafana** follow the below steps.

**Step 1:** Create a Prometheus Data source in Grafana by inputting the needed values.

**Step 2:** Once the Data source is configured use it to explore just to visualize and try out if the data source is working.

**Step 3:** Once the metrics are visualized, head to create Dashboard and add a panel by clicking new panel. In that select data source as Prometheus and input the metric name to be queried. Follow this step to add all the metrics as separate panels and then save the Dashboard finally.

Graphical user interface

Description automatically generated

**Image 5**. In the above image one can see the Grafana Dashboard configured VPN metrics

I hope by following the above steps one must be able to visualize metrics in Prometheus and Grafana. Happy Visualization and Monitoring!