Prometheus and Grafana

Introduction

CPS system, application statistics and Key Performance Indicators (KPI) are collected by the system and are displayed using a browser-based graphical metrics tool. This chapter provides a high-level overview of the tools CPS uses to collect and display these statistics.

Prometheus

Prometheus is an application that is used to actively gather statistics and trigger alerts from the running virtual machines and application services. The CPS vDRA cluster deploys the following Prometheus services on each control node and on the master node:

- Prometheus Hi-Res – this instance of the Prometheus service is monitoring the system at 5 second intervals with 48-hour history
- Prometheus Trending – this instance of the Prometheus service is monitoring the system at 20 second intervals with 30-day history
- Prometheus Planning – this instance of the Prometheus service is monitoring the system at 120 second intervals with 365-day history

Internally, the Prometheus servers scrape statistics from target statistics sources on a regular basis. The following target data sources are included:

- Host Node Exporter for Host VM statistics.
- Mongo DB Exporter for Database statistics.
- Application Statistics.

In addition to scrapping, statistics in the Prometheus servers can be configured using the Management CLI alert rule command to trigger alerts on error conditions. In this scenario, a user defines the alert rule and the configuration for that rule is pushed into the Prometheus servers. It can generate SNMPv2 and SNMPv3 alarm based on the NMS destination configured in the system. You can configure multiple SNMP destination (SNMPv2, SNMPv3) to receive the alarms at multiple NMS.

Note Currently, SNMP get and walk facility is not supported.

For more information on Prometheus, refer https://prometheus.io/.

Prometheus Queries

The CPS vDRA supports exposing of Prometheus API queries on OAM network using HAProxy. vDRA allows operators to fetch necessary statistics from the system through the Prometheus API and further analyze in a single consolidated view. The following functions are supported:

- vDRA data gets pulled from Prometheus API and loaded directly into system for visualization. This includes data from the following three data stores:
  - Prometheus Hi-Res
  - Prometheus Trending
  - Prometheus Planning
- The Maximum TPS that is required for enabling these queries at required intervals are:
  - TPS: The TPS value is 160K.
  - Intervals: one minute and five minutes
- GET queries: GET /api/v1/query
- TLS or HTTPS-based authentication

Following statistics are collected from target sources and are available through Prometheus APIs:

- Host Node Exporter for Host VM statistics
- Mongo DB Exporter for Database statistics
- Application Statistics
- Scrape interval takes fewer samples and keep data for a longer interval. This is performed through CLI configurations:
  - To change the scrape interval timings for Hi-Res container, Planning container, Trending container. For example, you can change interval timing for Hi-res to 30s, Planning to 125s, Trending to 60s.
  - Through default Scrape interval timings. For example, default scrape interval timing for Hi-res is 5s, Planning is 120s Trending is 20s , and Maximum scrape interval timing for Hi-res is 30s, Planning is 125s and Trending is 60s.
  - Observes disc utilization improvements after changing the scrape interval.

- Verifies the hi-res, trending and planning data disc pattern with default for Hi-res 5s, Planning 120s Trending 20s, Maximum scrape interval timing for Hi-res 30s, Planning 125s Trending 60s.

  An example for reduction in log size.

```
#### Disk Size with Default Scrape Interval with 20KTPS Traffic ####
Hi-res [5s], Planning[120s], Trending [20s]

du -sh /stats
2 GB -> prometheus-hi-res
9 GB -> prometheus-planning
4.5 GB -> prometheus-trending

**** Disk Size with Maximum Scrape Interval with 20KTPS Traffic ****
Hi-res [30s], Planning[125s], Trending [60s]

du -sh /stats
419 MB  ->  prometheus-hi-res---70% disk got reduced
4.7 GB ->  prometheus-planning---50% disk got reduced
2.3 GB  ->  prometheus-trending----50% disk got reduced
```

Important: Scrape interval configuration is important because it determines the number of data points per minute (DPM) scraped in your active series. However, using scrape intervals of more than 2 minutes in Prometheus is not recommended. Due to the default staleness period of 5 minutes, a scrape interval of 2 minutes allows for one failed scrape without the metrics being treated as stale.

For more information see the *CLI Commands* chapter in the *vDRA Operations Guide.*

Configuring HAProxy

To expose the Prometheus data to external users, you should modify HAProxy configurations on haproxy-common containers.

Set up HAProxy configurations to accept incoming requests on port 443. HAProxy then checks their URL paths or Prometheus and then forwards them to the correct backend.

The frontend and backend settings are segregated based on the URLs that are used for querying and the backend data stores respectively.

For example, different configurations are considered for frontend of Prometheus hi-res data where the backend is the Prometheus hi-res data store. Similarly, different configurations are used for Prometheus planning and trending data stores.

The following endpoint evaluates an instant query at a single point in time: `GET /api/v1/query`

Example 1:

```
curl -v -k -u
admin:admin https://172.18.63.223/trending_prometheus
/api/v1/query_range?query="sum((docker_service_up%7Bcontainer_name%
20%3D~%20%22diameter-endpoint-s.*%22%7D%3D%3D2)%2F2)
&start=1639029600&end=1639029915&step=15"
```

Example 2:

```
curl -v -k -u
admin:admin https://172.18.63.223/trending_prometheus
/api/v1/query_range?query="sum((docker_service_up
%7Bcontainer_name%20%3D~%20%22binding-s.*%22%7D%3D%3D2)%2F2)&
start=1639029675&end=1639029990&step=15"
```

Exposing Prometheus Hi-res, Trending, and Planning Data

Use the following table details to expose Prometheus Hi-res, trending, and planning data.

Note Installer IP refers to the virtual IP of OAM servers (master/control-0/control-1) that are exposed to external users. The **Query** field is provided with the required Prometheus queries.

Prometheus Service	Description	URL	Authentication
Prometheus - Hi-res data	This instance of the Prometheus service monitors the system at 5 second intervals with 48-hour history.	`"https://installer/hi_res_prometheus /api/v1/query_range?query="" "`	HTTPS or TLS-based authentication is supported.
Prometheus - Trending	This Prometheus service monitors the system at 20 second intervals with 30-day history.	`https://installer/trending_prometheus /api/v1/query_range?query=” ”`	HTTPS or TLS-based authentication is supported.
Prometheus - Planning	This Prometheus service monitors the system at 120 second intervals with 365-day history.	`https://installer/planning_prometheus /api/v1/query_range?query=” ”`	HTTPS or TLS-based authentication is supported,

Configuring Prometheus Federate Server

To expose the Prometheus API metrics from Prometheus clusters, you should configure the Prometheus Federate Server. You can configure the Prometheus Clusters of multiple vDRA sites in one Prometheus external Server.

Here, the Prometheus server which contains service-level metrics pull in the cluster resource usage metrics about its specific service from the cluster Prometheus. Hence, both the sets of metrics can be used within that server.

Install the Prometheus packages on a remote server and verify if the remote server is reachable to the DRA Environment.

As a part of this feature, the below configuration is followed to expose the Prometheus KPI metrics to the external server via an external interface:

1. Edit the Prometheus.yml file to configure the federation job.

```
#cat Prometheus.yml
   scrape_configs:
  - job_name: 'federate'   -> Mention the job name.
    scrape_interval: 15s

    honor_labels: true
    scheme: 'https'
    metrics_path: '/hi_res_prometheus/federate'  -> Metrics path exposed from DRA
Environment.It supports hi_res data from Prometheus clusters.


    params:
      'match[]':
        - '{job="cisco-app"}'  -> Metrics which need to be pulled from Prometheus
Clusters.
        - '{__name__=~"job:.*"}' -> Type of metrics need to be pulled.
        - '{job="orchestrator"}'
    static_configs:
      - targets:
        - 'dra-master-site-1:443'   -> Mention the IP of Master VM as a target host.

    tls_config:
       insecure_skip_verify: true
    basic_auth:
       username: 'admin'    -> Mention the CLI user name.
       password: 'xxxxx'  -> Mention the CLI Password configured on Master VM.
```

Note The CLI password should be in plain text format.

2. Use the below command to start the prometheus service.

```
./prometheus --log.level=debug
```

3. Pull and review the Prometheus metrics from an external server in Prometheus data format.

```
#curl -i -G --data-urlencode 'match[]={job=~"cisco-app"}' http://localhost:9090/federate
 -> To view the particular metrics.
   #curl -i -G --data-urlencode 'match[]={job=~".+"}' http://localhost:9090/federate ->
 To view all the configured metrics.
```

Sample Output:-

```
used_heap_memory{instance="binding-s111",job="cisco-app",monitor="monitor",vmname="site2-dra-worker2"}
 14668 1666179533350
used_heap_memory{instance="binding-s109",job="cisco-app",monitor="monitor",vmname="dra1-sys04-worker-2"}
 9020 1666179612593
used_heap_memory{instance="binding-s110",job="cisco-app",monitor="monitor",vmname="site2-dra-worker1"}
 29750 1666179532477
used_heap_memory{instance="diameter-endpoint-s107",job="cisco-app",monitor="monitor",vmname="site2-dra-director5"}
```

```
 6014 1666179532858
used_heap_memory{instance="diameter-endpoint-s104",job="cisco-app",monitor="monitor",vmname="dra1-sys04-director-1"}
 20574 1666179613440
used_heap_memory{instance="diameter-endpoint-s104",job="cisco-app",monitor="monitor",vmname="site2-dra-director2"}
 12824 1666179534287
used_heap_memory{instance="diameter-endpoint-s105",job="cisco-app",monitor="monitor",vmname="dra1-sys04-director-2"}
 23384 1666179614630
used_heap_memory{instance="diameter-endpoint-s105",job="cisco-app",monitor="monitor",vmname="site2-dra-director3"}
 12284 1666179532952
used_heap_memory{instance="diameter-endpoint-s106",job="cisco-app",monitor="monitor",vmname="site2-dra-director4"}
 10004 1666179534972
used_heap_memory{instance="binding-s108",job="cisco-app",monitor="monitor",vmname="dra1-sys04-worker-1"}
 19204 1666179613616
```

Grafana

Grafana is a third-party metrics dashboard and graph editor provided with CPS 7.0 and higher. Grafana provides a graphical or text-based representation of statistics and counters collected in the Prometheus database.

Note After the DRA Director (DD) failover/reboot, the TPS values in Grafana dashboards takes approx. 5 minutes to fetch and display the latest updated values. Until the values are updated, Grafana displays the old data.

Additional Grafana Documentation

This chapter provides information about the CPS implementation of Grafana. For more information about Grafana, or access the general Grafana documentation, refer to: http://docs.grafana.org.

Data Source Supported

The CPS implementation uses the Prometheus data source and does not use Graphite for queries. This requires the definition of queries to use the Prometheus query format as defined in https://prometheus.io/docs/querying/basics/.

Note After changing respective KPI panel's width to 24 (which is maximum), you can get all the spikes captured for 6 hours duration. So, if tou need to analyse longevity report for 12 hours or more, you can grep data by grouping in 6 hours interval.

Note If the control VM that hosts Grafana goes down, then the Prometheus data also not available during that downtime after the same control VM (hosting Grafana) is back. This results in some missing data. As a workaround, you can add the Prometheus datasource of other control VM in Grafana UI that was up during that downtime and view the missing statistics.

Note The `top` command output must not be compared with the Grafana CPU statistics panel display.

Manage Grafana Users

Note In Grafana, admin users can invite new users by email or a link. However, this is not supported in CPS vDRA.

Perform the following to add a new Grafana:

1. Enter config mode

```
scheduler# config
Entering configuration mode terminal
scheduler(config)#
```

2. Enter the **aaa authentication** command to create the user:

```
scheduler(config)# aaa authentication users user test2 gid 100 uid 9000 homedir / password
 testpassword ssh_keydir /
scheduler(config-user-test2)# commit
scheduler(config-user-test2)# exit
```

Note The **gid**, **uid**, **homedir** and **ssh_keydir** are required but not used by the application.

Add User To A Viewer Operational Group

In config mode, add the user to the “oper” group and commit as follows:

```
scheduler(config)# nacm groups group oper user-name test2
scheduler(config-group-oper)# commit
```

Add User To A Grafana Editor Group

In config mode, add the user to the “grafana-editor” group and commit as follows:

```
scheduler(config)# nacm groups group grafana-editor user-name test2
scheduler(config-group-grafana-editor)# commit
```

Add User To A Grafana Admin Group

In config mode, add the user to the “grafana-admin” group and commit as follows:

```
scheduler(config)# nacm groups group grafana-admin user-name test2
scheduler(config-group-grafana-admin)# commit
```

Change A Grafana Users Password

In the Management CLI, issue the **aaa authentication users** user **change-password** command as follows:

```
scheduler# aaa authentication users user test2 change-password
Value for 'old-password' (<string>): ************
Value for 'new-password' (<string>): ********
```

```
Value for 'confirm-password' (<string>): ********
scheduler#
System message at 2017-03-08 21:17:18...
Commit performed by system via system using system.
```

Specify Access Restrictions for a Group

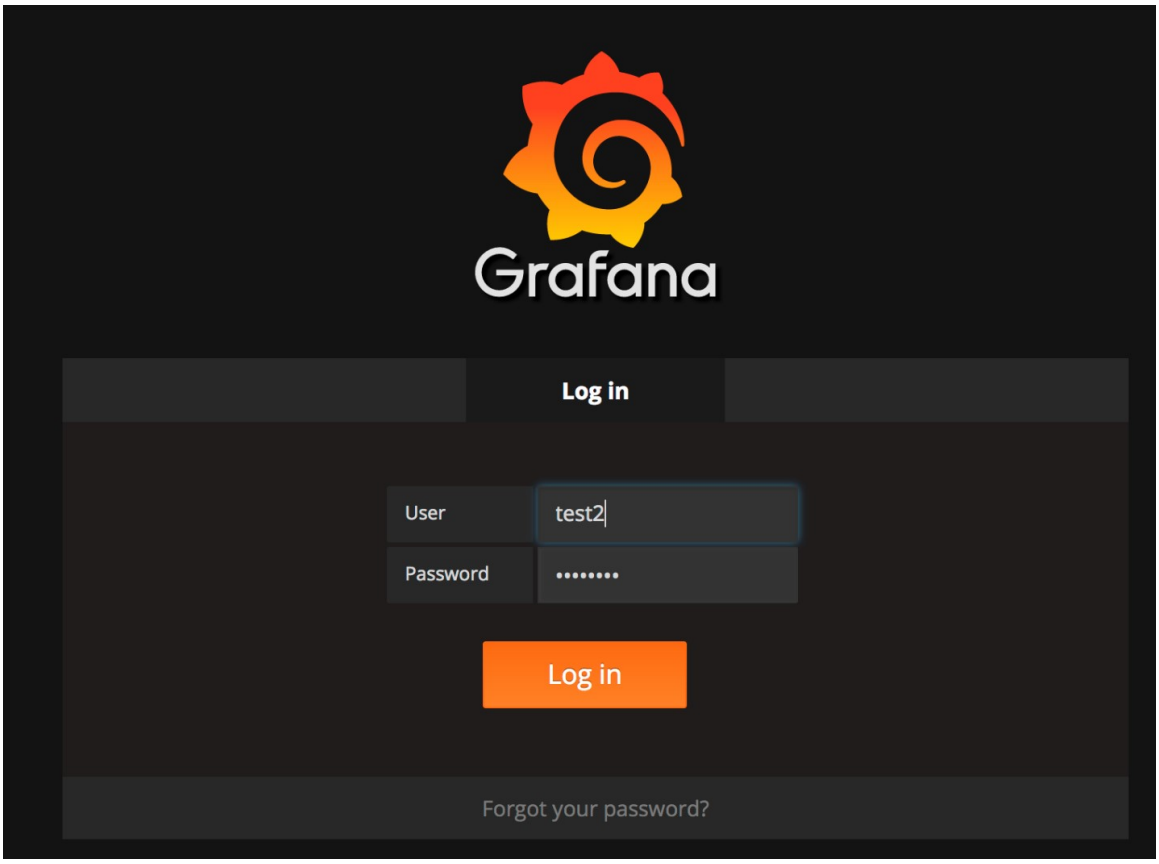
For more information, see the nacm rule-list command.

Connect to Grafana

Use the following URL to access Grafana and enter the user name and password:

https://<*masterip*>/grafana/

Figure 1: Grafana Login



Attention DRA is using the Grafana login page maintained as a part of Grafana code base. By default, when you open a web page in a new tab by clicking on a link with `target="_blank"`, you allow an attacker to redirect users clicking such a link to another web page. The issue is that the redirect concerns the initial tab (your web page), not the newly opened window. Also, the redirect is done without any warning. This can be used as a very effective phishing method. This kind of phishing method is called (reverse) tab nabbing. This issue of `target="_blank"` attribute is present in Grafana 5.2.3 used by DRA.

If you have to use `target="_blank"` attribute, you must also add `: rel="noopener"`. This attribute sets the **window.opener** value to null (forbids any URL change on the referring page). The `rel="noopenener"` attribute has been added in the latest version of Grafana for fixing this issue.

This is not a security vulnerability in CPS product. CPS uses Grafana in a controlled environment and no tab nabbing is possible.

Grafana Roles

The following types of user roles are supported:

- Admin: An admin user can view, update and create dashboards. Also, the admin can edit and add data sources and organization users.
- Viewer: A viewer can only view dashboards and cannot not save or create them.
- Editor: An editor can view, update and create dashboards.