

iDRAC Telemetry Reference Tools Setup Instructions (DRAFT)

This technical document describes instructions to setup Telemetry Reference Tools.

Dell Server Engineering
January 2021

Authors

Server Telemetry Team

Server Software Solutions Group

Revisions

Date	Description
January 2021	Add ELK stack setup instructions and details.
March 2021	Add Influx DB stack setup instructions.
April 2021	Add Timescale stack setup instructions.
April 2021	Add Prometheus DB stack setup instructions.
June 2021	Updated to reflect recent improvements in the aggregator applications

Table of contents

Revisions	2
1 Prerequisite - Setting up Servers with Telemetry Enabled and Reports Configured.....	4
2 Assumptions and other technical considerations.....	5
3 Toolset Architecture and Setting up the data ingestion pipeline.....	6
3.1 Software Components	6
3.2 Instructions to run the pipeline as standalone applications.....	7
3.3 Instructions to run the pipeline as container applications	8
3.4 Instructions to configure source iDRACs	9
4 Setting up the Elasticsearch data ingestion pipeline.....	11
4.1 Prerequisites.....	11
4.2 Instructions to run the pipeline in docker compose	11
4.3 Data Model and Applications.....	14
5 Setting up the Influxdb data ingestion pipeline	20
5.1 Prerequisites.....	20
5.2 Instructions to run the pipeline in docker compose	20
5.3 Data Model and Applications.....	23
6 Setting up the Timescale data ingestion pipeline	24
6.1 Prerequisites.....	24
6.2 Instructions to run the pipeline in docker compose	24
6.3 Data Model and Applications.....	28
7 Setting up the Prometheus data ingestion pipeline.....	29
7.1 Prerequisites.....	29
7.2 Instructions to run the pipeline in docker compose	29
7.3 Data Model and Applications.....	33

1 Prerequisite - Setting up Servers with Telemetry Enabled and Reports Configured

- 1) Install firmware version 4.00.00 or higher in PowerEdge iDRACs. Telemetry is a Datacenter licensed feature.
- 2) Download the telemetry utilities from- <https://github.com/dell/iDRAC-Telemetry-Scripting>

```
$ wget https://github.com/dell/iDRAC-Telemetry-Scripting/archive/master.zip -O iDRAC-TelemetryScripting-master.zip
$ unzip iDRAC-TelemetryScripting-master.zip
$ cd iDRAC-Telemetry-Scripting-master
```

- 3) The following steps should be performed on each iDRAC9 to enable telemetry reports. Please note that Telemetry streaming is only available with Datacenter license.

Note in the command below, replace \$target with the IP address or DNS name of the iDRAC9, replace \$user with an iDRAC9 username with administrator privileges, and replace \$password with the specified user's password

```
$ python3 ./ConfigurationScripts/EnableOrDisableAllTelemetryReports.py -ip $target -u $user -p $password -s Enabled
```

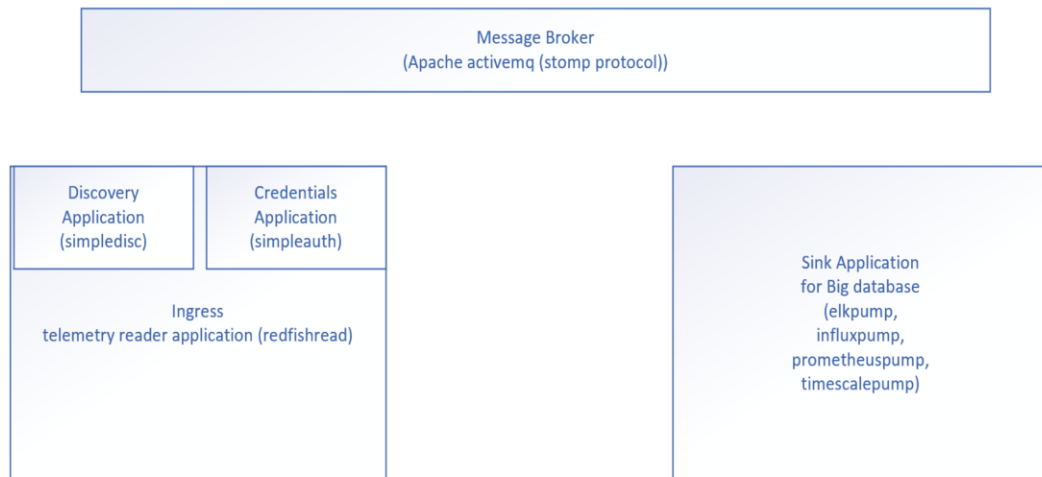
2 Assumptions and other technical considerations

- 1) The docker compose files included in this repository can serve as reference for deployments. Docker compose configuration and flow of data between the containers are not enabled to be secure or persistent and is required to be modified to suit the needs of your environment.
- 2) README at <https://github.com/dell/iDRAC-Telemetry-Reference-Tools> provides the high level architectural dataflow diagram for this reference toolset.
- 3) The reference toolset functions as an aggregator for the telemetry data from PowerEdge Servers and can store the data as timeseries metric data points in the supported big databases. While the reference docker compose files are tested with analytics and visualization tools like Kibana and Grafana, detailed dashboards on specific application use cases are out of scope. Please refer the documentation for Kibana or Grafana to come up with your preferred dashboards.
- 4) The toolset is designed with flexibility and scalability as a goal. Inter-process communication leverages the ActiveMQ message bus.
Flexibility - Major functionalities like remote source (iDRAC) discovery, credentials management for authorization, and telemetry report processing are abstracted as separate standalone applications. All IPC are through the message bus.
 - Provided the IPC message interface structure is maintained, applications can be easily replaced or extended to suite the environments these applications are targeted to use.
 - One or more ingest applications can be run to perform metrics ingestion into one or more database of choice.Scalability - Additional endpoints of iDRACs can be supported by adding more containers as it is needed to support the additional processing and data load in the environment.
- 5) The reference toolset can be run as docker containers or standalone.

3 Toolset Architecture and Setting up the data ingestion pipeline

3.1 Software Components

Toolset has following logical blocks



idrac-telemetry-receiver (iDRAC-Telemetry-Reference-Tools/cmd/ idrac-telemetry-receiver.sh)

redfishread application - Make SSE (Server Sent Event) connection with each discovered data sources(iDRACs) and forwards the telemetry report streams to sink applications through a shared message bus connection. iDRAC Telemetry reports are DMTF redfish compliant.

configgui applications - Graphical User Interface application to configure telemetry source services (iDRACs)

dbdiscauth applications - Database (mysql) based discovery and authentication functions

[Optional]simpleauth and simplifiedisc applications (Abstracts a file based (following the sample - config.ini) discovery and authentication functions

Sink Applications

Read the telemetry reports from message bus and ingest the report streams into specific analytical solution.

timescalepump - Ingest timeseries metrics into Elasticsearch database.

influxpump - Ingest timeseries metrics into InfluxDB database.

prometheuspump - Ingest timeseries metrics into Prometheus database.

timescalepump - Ingest timeseries metrics into TimeScale database.

Application reads and uses following environment variables. Please refer the docker compose files for further information.

MESSAGEBUS_HOST

MESSAGEBUS_PORT

CONFIGUI_HTTP_PORT

MYSQL_DATABASE

MYSQL_USER

MYSQL_PASSWORD

MYSQL_HOST

MYSQL_HOST_PORT

3.2 Instructions to run the pipeline as standalone applications

Once you have Go installed you should be able to build each of the command binaries. Please replace <go> with the absolute path of the go binary if the installed binary is not in the default execution path.

idrac-telemetry-receiver components

* <go> build .\cmd\dbdiscauth\dbdiscauth.go

* <go> build .\cmd\configui\configui.go

* <go> build .\cmd\redfishread\redfishread.go

* <go> build .\cmd\simpledisc\simpledisc.go (Optional for config.ini based discovery)

* <go> build .\cmd\simpleauth\simpleauth.go (Optional for config.ini based discovery)

Sink Applications

* <go> build .\cmd\elkpump\elkpump-basic.go

* <go> build .\cmd\influxpump\influxpump.go

* <go> build .\cmd\timescalepump\timescalepump.go

* <go> build .\cmd\prometheuspump\prometheuspump.go

Now get ActiveMQ running, the default config is fine

- * cd to the directory you unpacked or installed ActiveMQ to

- * bin\activemq start

Update the config.ini file with following details

Stomp host names (activemq) and stomp port as configured in the local configuration file of activemq

Sample file:

```
[General]
StompHost=activemq
StompPort=61613
```

Now start the other daemons in any order (I tend to start from the bottom up, but shouldn't matter)

- * simpledisc (config.ini based discovery)
- * simpleauth (config.ini based authentication)
- * redfishread
- * dbdiscauth -> (mysql db based discovery and authentication)
- * configui -> web UI with interface to update source services in mysql db
- * elkpump-basic
- * influxpump
- * timescalepump
- * prometheuspump

3.3 Instructions to run the pipeline as container applications

This is further explained in detail in section 4.

iDRAC-Telemetry-Reference-Tools/docker-compose-files/ has reference docker compose pipelines.

3.4 Instructions to configure source iDRACs

simpleauth and simplifiedisc applications allow file based discovery and auth.

Update the [Services] section of config.ini file with following details

IP and credentials of iDRACs in the services section

Sample file:

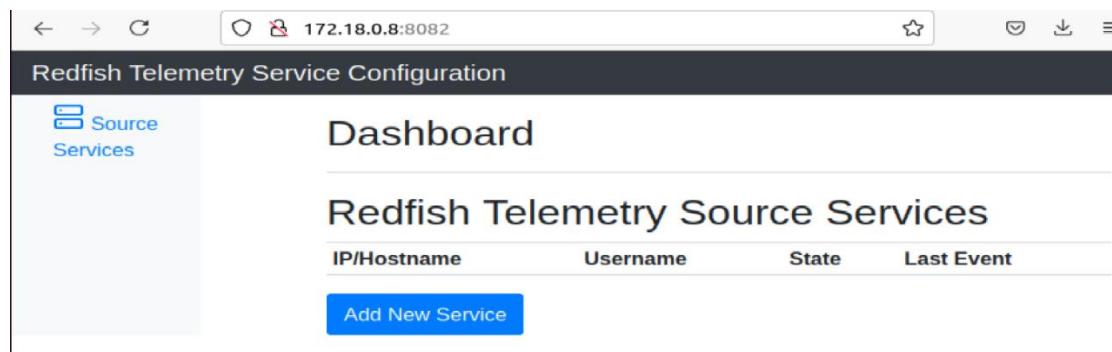
```
[Services]
Types=iDRAC,iDRAC,iDRAC
IPs=ip1,ip2,ip3

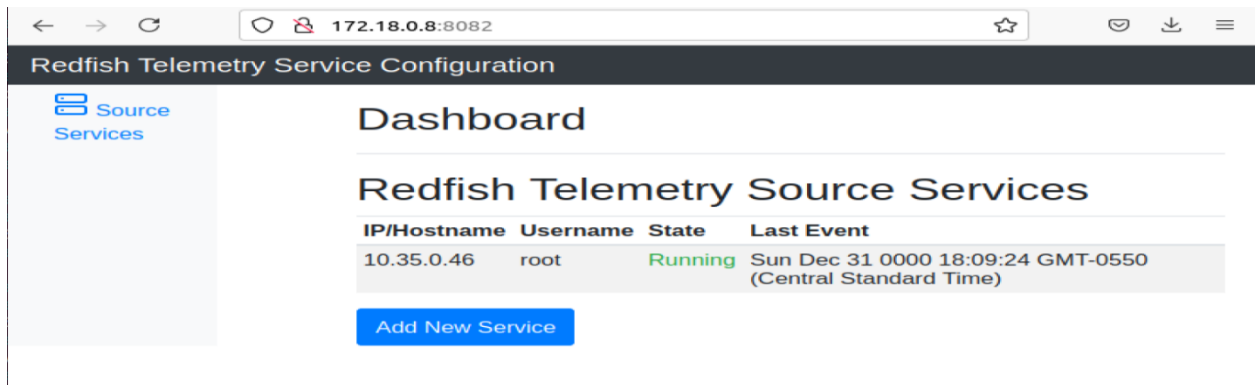
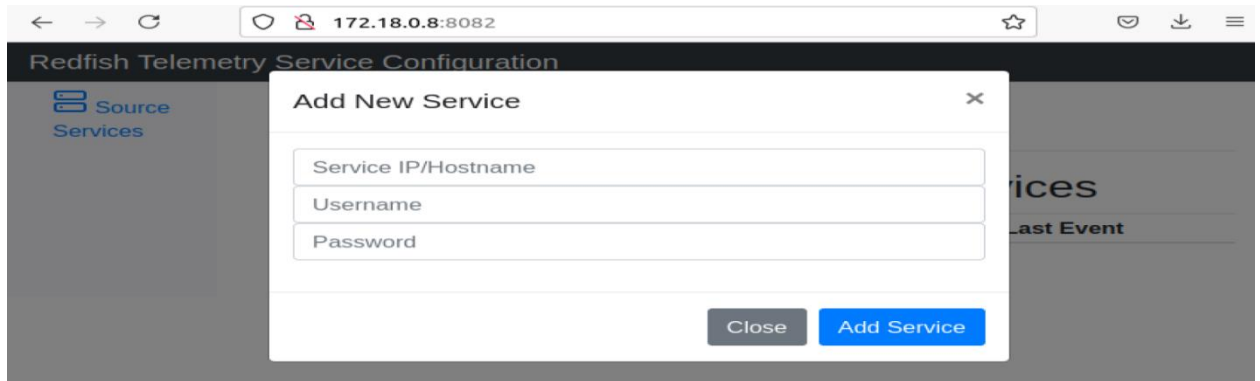
[ip1]
username=usr1
password=pwd1

[ip2]
username=usr2
password=pwd2
```

configui and dbdiscauth applications allow web user interface-based configuration. mysqlldb credentials can be configured at docker compose files. Default configui http port is 8082 (`CONFIGUI_HTTP_PORT=8082` as in docker compose file).

Section 3.1 lists the configurable variables which can be enabled through environment variables. Please note and the webservice IP is the IP of the telemetry-receiver container (or local host if running in standalone mode).





4 Setting up the Elasticsearch data ingestion pipeline

4.1 Prerequisites

Go - <https://golang.org/>

The docker compose file was tested with go1.15.6 linux/amd64. But technically any recent version should be just fine.

ActiveMQ

-activemq-5.16.0 - <https://activemq.apache.org/components/classic/download/>

Elastic Search (<https://www.digitalocean.com/community/tutorials/how-to-install-elasticsearch-logstash-and-kibana-elastic-stack-on-ubuntu-20-04>)

Kibana (<https://www.digitalocean.com/community/tutorials/how-to-install-elasticsearch-logstash-and-kibana-elastic-stack-on-ubuntu-20-04>)

4.2 Instructions to run the pipeline in docker compose

Included is a reference docker compose configuration file.

elastic-docker-pipeline-reference-unenc.yml – starts elasticsearch, kibana and ingest applications as docker containers. Docker compose configuration is not enabled to be secure or persistent and is required to be modified to suit the needs of deployment environment.

```
athena@athena-PowerEdge-R640:~/iDRAC-Telemetry-Reference-Tools$ docker-compose -f
docker-compose-files/elastic-docker-pipeline-reference-unenc.yml up -d
```

```
Creating network "docker-compose-files_elastic" with driver "bridge"
Creating es03          ... done
Creating es01          ... done
Creating activemq      ... done
Creating mysqldb       ... done
Creating es02          ... done
Creating es-ingester   ... done
Creating telemetry-receiver ... done
Creating kib01         ... done
```

```
athena@athena-PowerEdge-R640:~/iDRAC-Telemetry-Reference-Tools$ docker-compose -f
docker-compose-files/elastic-docker-pipeline-reference-unenc.yml ps
```

Name	Command	State
activemq	/bin/bash -c bin/activemq ...	Up
es-ingester	go run cmd/elkpump/elkpump ...	Up

```

es01          /tini -- /usr/local/bin/do ... Up (healthy) 9200/tcp,
9300/tcp
es02          /tini -- /usr/local/bin/do ... Up          9200/tcp,
9300/tcp
es03          /tini -- /usr/local/bin/do ... Up          9200/tcp,
9300/tcp
kib01         /usr/local/bin/dumb-init - ... Up          5601/tcp
mysqldb       docker-entrypoint.sh mysqld Up          3306/tcp,
33060/tcp
telemetry-receiver /bin/sh -c cmd/idrac-telem ... Up

```

athena@athena-PowerEdge-R640:~/iDRAC-Telemetry-Reference-Tools\$ docker network ls

NETWORK ID	NAME	DRIVER	SCOPE
cf13f260922c	bridge	bridge	local
68c0c061256b	docker-compose-files_elastic	bridge	local
4d6d8dcd18da	host	host	local

athena@athena-PowerEdge-R640:~/dell-github/iDRAC-Telemetry-Reference-Tools\$ docker network inspect docker-compose-files_elastic

```

[
  {
    "Name": "docker-compose-files_elastic",
    "Id": "68c0c061256b219815c24fa19831f4aa4798afdbcf890a4d96573b779277b438",
    "Created": "2021-06-28T18:47:39.531696162-05:00",
    "Scope": "local",
    "Driver": "bridge",
    "EnableIPv6": false,
    "IPAM": {
      "Driver": "default",
      "Options": null,
      "Config": [
        {
          "Subnet": "172.18.0.0/16",
          "Gateway": "172.18.0.1"
        }
      ]
    },
    "Internal": false,
    "Attachable": true,
    "Ingress": false,
    "ConfigFrom": {
      "Network": ""
    },
    "ConfigOnly": false,
    "Containers": {
      "09bda51f74b39e27f29696507069222df5b564d2469aa561c1e8c540d6f17d34": {
        "Name": "kib01",
        "EndpointID": "27cb8655a4e5bfff093c4afalc7793e4dd4673f66791d0cf0ed6b6e528e88c6e0",
        "MacAddress": "02:42:ac:12:00:09",
        "IPv4Address": "172.18.0.9/16",
        "IPv6Address": ""
      },
    }
  }
]

```

```
"1d4798bec3ef375d55b21fc357be6058fa958792c5fd9f3f4aff3a6eb21d4e35": {
    "Name": "es03",
    "EndpointID":
"cfcae84364bc04d22313c9c1a4a4fe05a7e1ebd36ca23601e66cdeeb952541eda",
    "MacAddress": "02:42:ac:12:00:06",
    "IPv4Address": "172.18.0.6/16",
    "IPv6Address": ""
},

"1dfc694c3134f91fd5de126b05felc3d2f52948972eebfabb74e8d11f07964a8": {
    "Name": "telemetry-receiver",
    "EndpointID":
"14adb73fb7a7a7fc9962b868d8bbf0917497b3db8062944b5f909adf3752e3f3",
    "MacAddress": "02:42:ac:12:00:08",
    "IPv4Address": "172.18.0.8/16",
    "IPv6Address": ""
},

"479b719be237a8eaf936ab6956ed52a333e297a1733a6676ecd31f84a36628d7": {
    "Name": "es-ingester",
    "EndpointID":
"3e46d0b8ac5de1b07c6b26ad69e910a196cef5926f406dfdd7cc6380b1781613",
    "MacAddress": "02:42:ac:12:00:07",
    "IPv4Address": "172.18.0.7/16",
    "IPv6Address": ""
},

"5682f834dc9bfb6f5cdcf1685af215841ac7acc99dc85be9c89f87f5623de53b": {
    "Name": "es01",
    "EndpointID":
"62613a3a0cc2a26777c36e19c77011c634f89bc213e7c41fed77862ab317c7dd",
    "MacAddress": "02:42:ac:12:00:04",
    "IPv4Address": "172.18.0.4/16",
    "IPv6Address": ""
},

"6c6689a30687d50b12e25b0314441330a66acdb3f7a67f365933ac0a73a78ff1": {
    "Name": "mysqldb",
    "EndpointID":
"cc0e6f767ec3c77abfa9b91874a82a52a6144f437677d8dddb486c418b0dfaa7",
    "MacAddress": "02:42:ac:12:00:05",
    "IPv4Address": "172.18.0.5/16",
    "IPv6Address": ""
},

"ad482ca9f43b76e5d01b95825547915fc9d6b21e62d02642749fb3a81da59f7c": {
    "Name": "activemq",
    "EndpointID":
"97e07fbdae9b60b27840c092e416325f4aad4ef3dd4efa7dfc6467a02e785722",
    "MacAddress": "02:42:ac:12:00:03",
    "IPv4Address": "172.18.0.3/16",
    "IPv6Address": ""
},

"ae1c000231123e8dac35987a66ecdb12f6cf8e083ed9fe8aaf3553fb30c63781": {
```

```

        "Name": "es02",
        "EndpointID":
"f8702fe2ae7d584850708f21c435b1f26db1fd02925f2fd144f88bf47faab638",
        "MacAddress": "02:42:ac:12:00:02",
        "IPv4Address": "172.18.0.2/16",
        "IPv6Address": ""
    },
    },
    "Options": {},
    "Labels": {
        "com.docker.compose.network": "elastic",
        "com.docker.compose.project": "docker-compose-files",
        "com.docker.compose.version": "1.27.4"
    }
}
]

```

Note that Kibana IP in this case is 172.18.0.9, telemetry receiver IP is 172.18.0.8.

Please follow the instructions in section 3.4 to add source iDRAC telemetry services.

4.3 Data Model and Applications

To configure the data source and visualization dashboards please access Kibana homepage in the browser (<http://172.18.0.9:5601>)

- 1) Select Stack Management from the Management section of the tools menu.

Index named – ‘poweredge_telemetry_metrics’ is shown under the Data -> Index Management tab. Our pipeline uses a custom application (elkpump) instead of a standard logstash plugin to stash the data in to elasticsearch lucene database. Application is storing the telemetry metrics under the ‘poweredge_telemetry_metrics’ index.

Indexes can be also retrieved through command line queries from the DevTools console view.

GET /_cat/indices?v=true

health	status	index	docs.count	docs.deleted	store.size	pri.store.size	uuid	pri	rep
green	open	poweredge_telemetry_metrics	116	1557849	206.9mb	102.7mb	Fn0wnfrDTJqyb-9DuLns7A	1	1
green	open	.apm-custom-link	0	0	416b	208b	2uLRNhg4TFigKVfz36p1Kg	1	1
green	open	.kibana_task_manager_1	5	5559	1mb	544.4kb	1S5B2z7_QyGAJbr_YXAWNQ	1	1
green	open	.apm-agent-configuration	0	0	416b	208b	0noOBC1MRhee2PeCZJ5XlQ	1	1
green	open	.kibana-event-log-7.10.1-000001	1	0	11.2kb	5.6kb	gyCgu-ZoSv-DppxsmkThEQ	1	1
green	open	.async-search	0	0	541.6kb	269.2kb	j5VuQCQqTHyaf3OHJB3mTA	1	1
green	open	.kibana_1	52	11	4.3mb	2.1mb	bWwS4bQHS_yVcb67AfN5wg	1	1

GET /powerededge_telemetry_metrics/_settings

```
{
  "powerededge_telemetry_metrics" : {
    "settings" : {
      "index" : {
        "routing" : {
          "allocation" : {
            "include" : {
              "_tier_preference" : "data_content"
            }
          }
        },
        "number_of_shards" : "1",
        "provided_name" : "powerededge_telemetry_metrics",
        "creation_date" : "1611685233721",
        "number_of_replicas" : "1",
        "uuid" : "Fn0wnfrDTJqyb-9DuLns7A",
        "version" : {
          "created" : "7100199"
        }
      }
    }
  }
}
```

The data model for the data available in this index can be viewed from the index listing in the kibana console or through command line method from DevTools console

GET /powerededge_telemetry_metrics/_mapping

```
{
  "powerededge_telemetry_metrics" : {
    "mappings" : {
      "properties" : {
        "Context" : {
          "type" : "text",
          "fields" : {
            "keyword" : {
              "type" : "keyword",
              "ignore_above" : 256
            }
          }
        },
        "ID" : {
          "type" : "text",
          "fields" : {
            "keyword" : {
              "type" : "keyword",
              "ignore_above" : 256
            }
          }
        },
        "Label" : {
          "type" : "text",
          "fields" : {
            "keyword" : {
              "type" : "keyword",
              "ignore_above" : 256
            }
          }
        }
      }
    }
  }
}
```

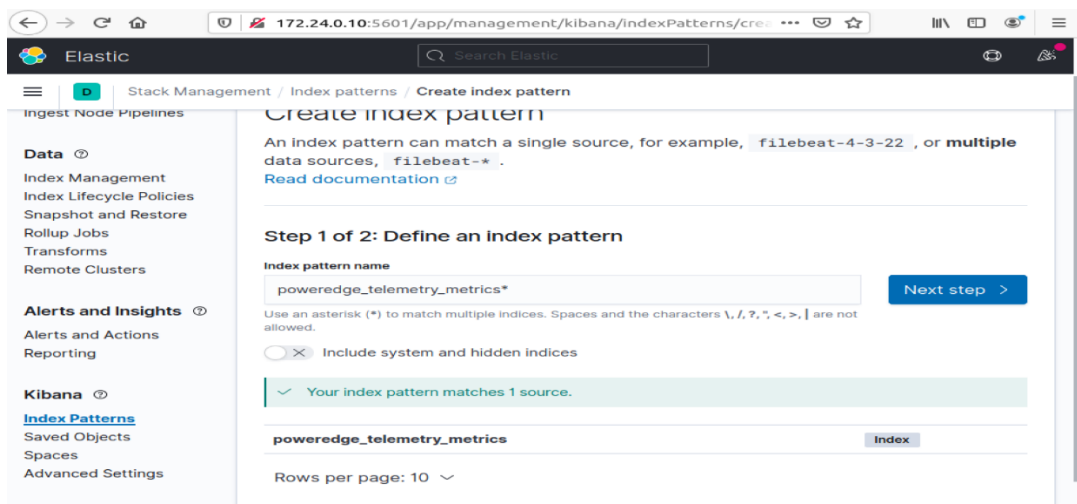
```

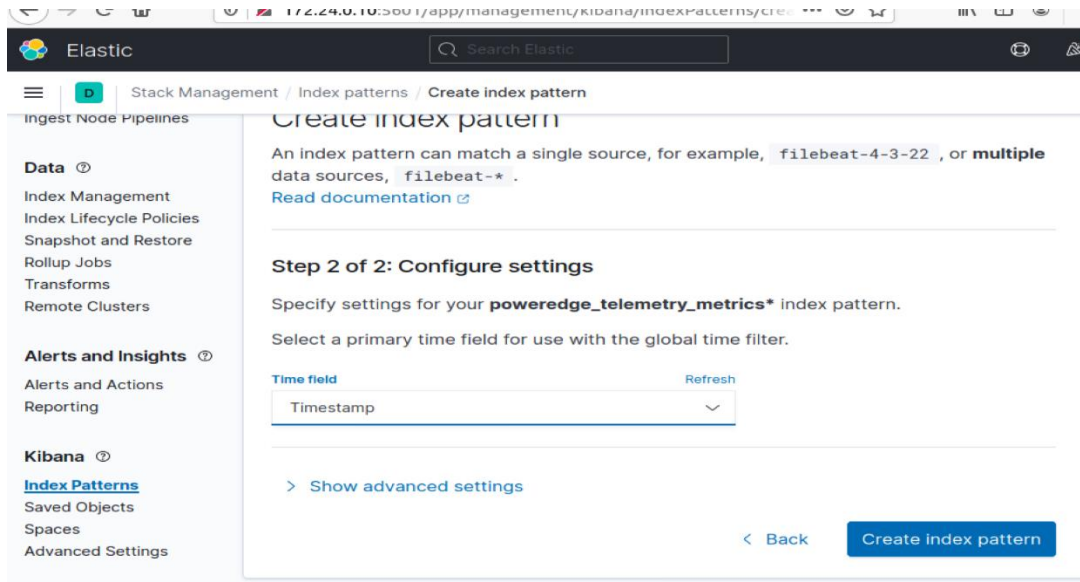
    },
    "System" : {
      "type" : "text",
      "fields" : {
        "keyword" : {
          "type" : "keyword",
          "ignore_above" : 256
        }
      }
    },
    "Timestamp" : {
      "type" : "date"
    },
    "Value" : {
      "type" : "text",
      "fields" : {
        "keyword" : {
          "type" : "keyword",
          "ignore_above" : 256
        }
      }
    },
    "ValueFloat" : {
      "type" : "long"
    },
    "ValueInt" : {
      "type" : "long"
    }
  }
}
}
}
}

```

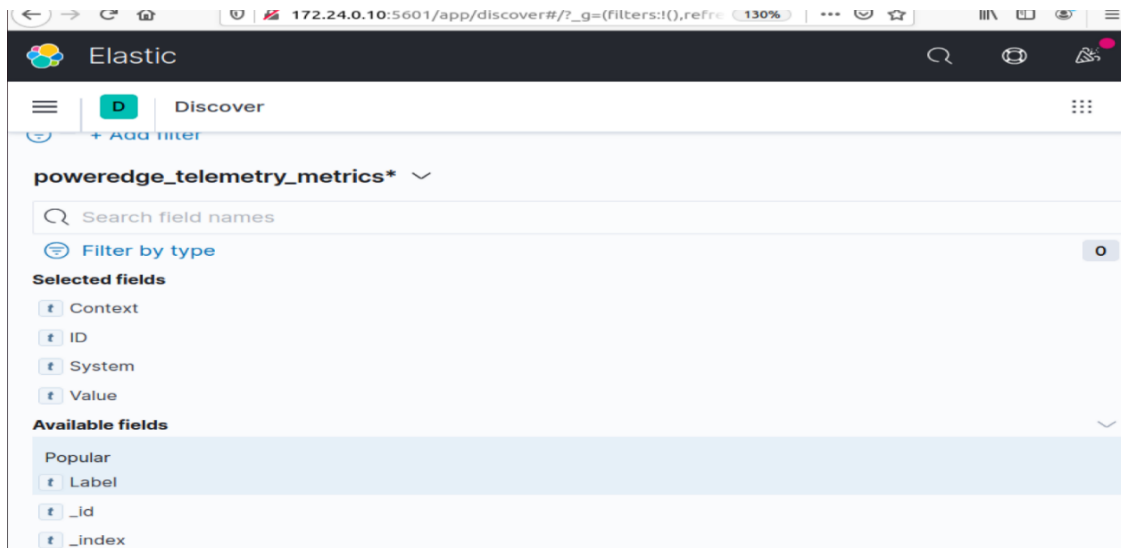
You can observe that metric values as sourced as int and float values are stored as long types in the elastic search and hence can be renamed as single value ValueAggregatable (TODO)

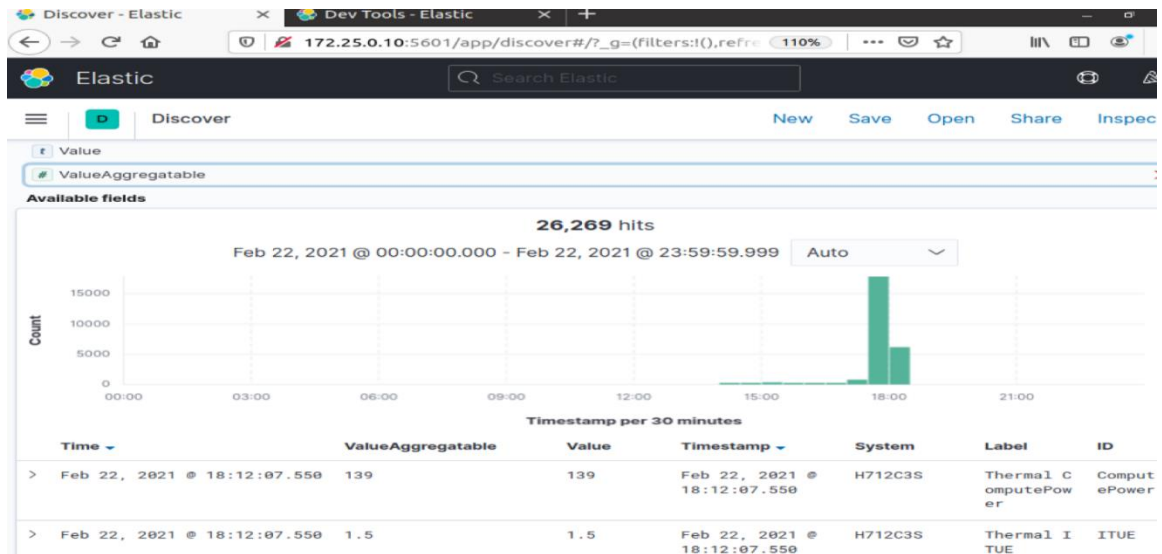
- 2) Create an index pattern for Kibana at index pattern tab.





- 3) Now from Discover tab, you can view the ingested data. You can also select the fields of interest and view the data in a tabular form with selected fields.

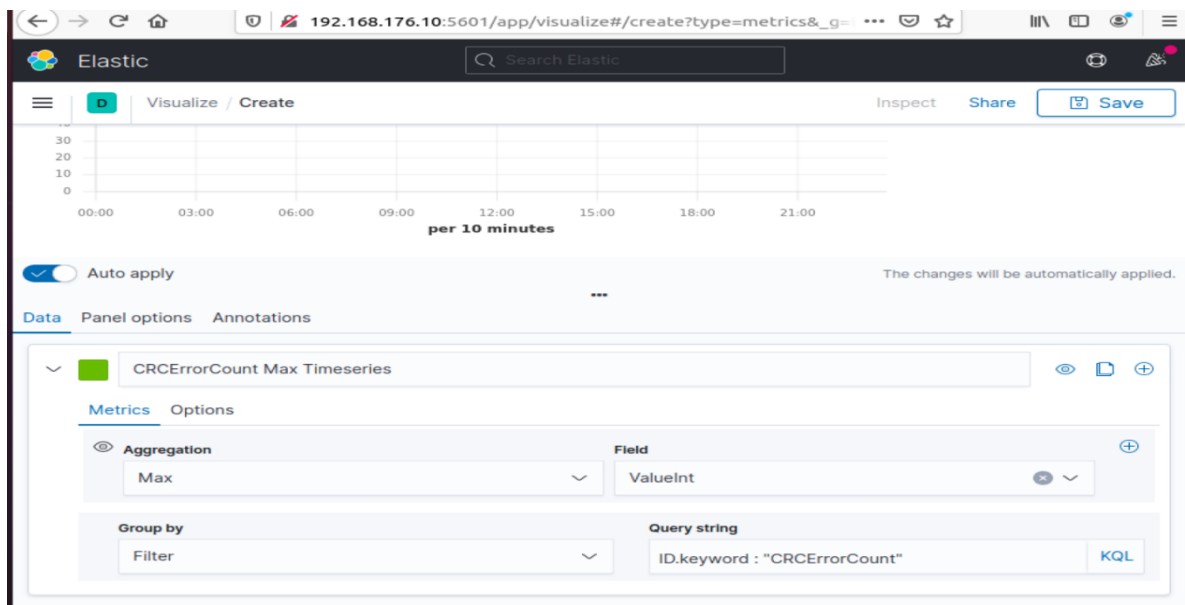


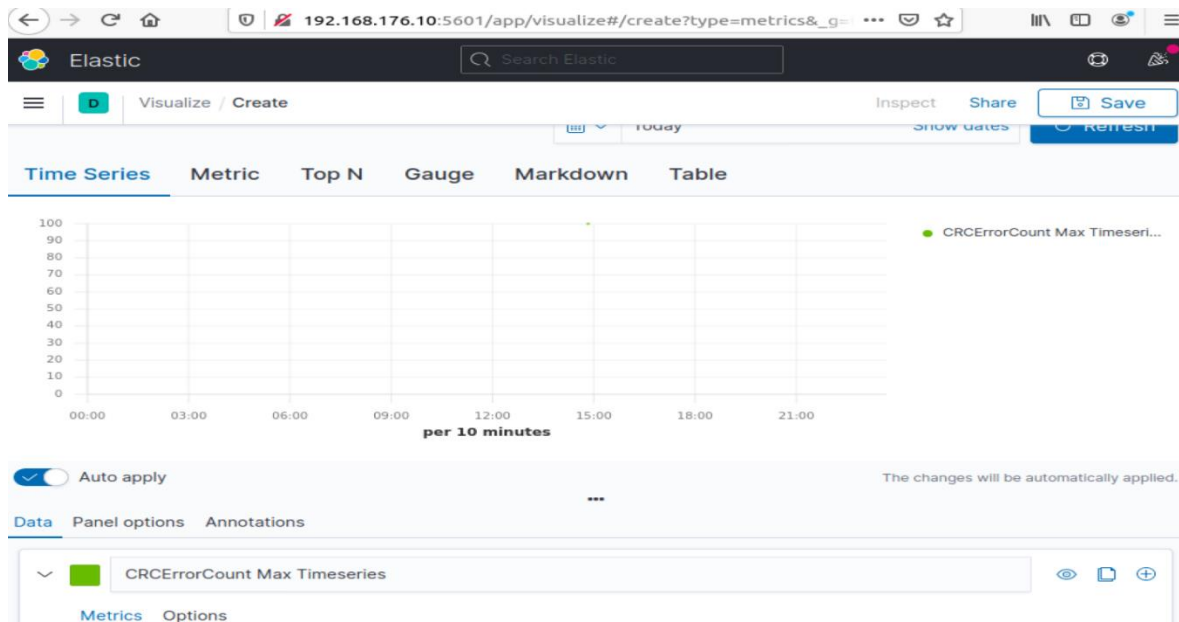


Save the discover search for future reference from visualization and dashboards.

- Now we can create charts for the dashboard with aggregable fields in data mapping. Please note that text values are indexed to be efficiently searchable in elastic search. And aggregable values (non text) can be used to configure relevant visualizations.

A timeseries label is configured as Max aggregation on CRCErrorCount below.





Kibana provides a wide variety of visualization techniques (licensed version has more variety and ML aggregates) and more charts can be created through trials and experimentations.

5 Setting up the Influxdb data ingestion pipeline

5.1 Prerequisites

Go - <https://golang.org/>

I had go version go version go1.15.6 linux/amd64. But technically any recent version should be just fine.

ActiveMQ

Download version apache-activemq-5.16.0 from
<https://activemq.apache.org/components/classic/download/>

InfluxDB

<https://docs.influxdata.com/influxdb/v1.6/introduction/installation/>

5.2 Instructions to run the pipeline in docker compose

Included is a reference docker compose configuration file.

influx-docker-pipeline-reference-unenc.yml – starts influx, grafana and ingest applications as docker containers. Docker compose configuration is not enabled to be secure or persistent and is required to be modified to suit the needs of deployment environment.

```
athena@athena-PowerEdge-R640:~/iDRAC-Telemetry-Reference-Tools$ docker-compose -f
docker-compose-files/influx-docker-pipeline-reference-unenc.yml up -d
```

```
Creating network "docker-compose-files_influxpipeline" with driver
"bridge"
Creating activemq ... done
Creating mysqldb ... done
Creating influx ... done
Creating influx-ingester ... done
Creating telemetry-receiver ... done
Creating grafana ... done
```

```
athena@athena-PowerEdge-R640:~/iDRAC-Telemetry-Reference-Tools$ docker-compose -f
docker-compose-files/influx-docker-pipeline-reference-unenc.yml ps
```

Name	Command	State
activemq	/bin/bash -c bin/activemq ...	Up

```

grafana                /run.sh                Up
3000/tcp
influx                /entrypoint.sh influxd        Up (healthy)
8086/tcp
influx-ingester       go run cmd/influxpump/infl ... Up
mysqldb               docker-entrypoint.sh mysqld    Up
3306/tcp, 33060/tcp
telemetry-receiver    /bin/sh -c cmd/idrac-telem ... Up
athena@athena-PowerEdge-R640:~/iDRAC-Telemetry-Reference-Tools$ docker network ls

```

NETWORK ID	NAME	DRIVER	SCOPE
cf13f260922c	bridge	bridge	local
68c0c061256b	docker-compose-files_influxpipeline	bridge	local
4d6d8dcd18da	host	host	local

```

athena@athena-PowerEdge-R640:~/dell-github/iDRAC-Telemetry-Reference-Tools$ docker
network inspect docker-compose-files_influxpipeline

```

```

[
  {
    "Name": "docker-compose-files_influxpipeline",
    "Id":
"cc77b13c8b66a2a144fceed7ba17fd2a2c7a602d7b8ec728c94d31d76c573d64",
    "Created": "2021-06-28T19:23:42.185811712-05:00",
    "Scope": "local",
    "Driver": "bridge",
    "EnableIPv6": false,
    "IPAM": {
      "Driver": "default",
      "Options": null,
      "Config": [
        {
          "Subnet": "172.19.0.0/16",
          "Gateway": "172.19.0.1"
        }
      ]
    },
    "Internal": false,
    "Attachable": true,
    "Ingress": false,
    "ConfigFrom": {
      "Network": ""
    },
    "ConfigOnly": false,
    "Containers": {
      "397561a6b42a6b3ffff83a3c14e5500c50b191a00895b5f65d86babfa448366f7": {
        "Name": "grafana",
        "EndpointID":
"e70beb2a5cf4b8147347096989b0bee16be2336dd0f325788783675272456a24",
        "MacAddress": "02:42:ac:13:00:07",
        "IPv4Address": "172.19.0.7/16",
        "IPv6Address": ""
      },
      "3ba867902937dca49aa33284e65ea224f271415e427e0889e2758cafcf4c33c9": {
        "Name": "influx",
        "EndpointID":

```

```

"9336cb536e2579de99dbf73c75247bd165eaf5399bb63c2bfaaa2d3f1e31d7c",
    "MacAddress": "02:42:ac:13:00:04",
    "IPv4Address": "172.19.0.4/16",
    "IPv6Address": ""
  },

  "52ce84932b6df72f8a174401cd15557af3b86a880617f93d5a935fe86764728d": {
    "Name": "mysqldb",
    "EndpointID":
    "c796b605b8c7ae03ea5c0fa16b328f8be07617301cea8093fb3dce0b92883377",
    "MacAddress": "02:42:ac:13:00:03",
    "IPv4Address": "172.19.0.3/16",
    "IPv6Address": ""
  },

  "7508dc510394709ab334bfbb501bd0a29b588298785186365ac44787be98a60e": {
    "Name": "activemq",
    "EndpointID":
    "7a19cb16156e605e3034541ff2fd12e5bc849632e6cabf26b957a7a308c764de",
    "MacAddress": "02:42:ac:13:00:02",
    "IPv4Address": "172.19.0.2/16",
    "IPv6Address": ""
  },

  "96c633454f19f52324970d562c8e36b21e0474afb0ea8194b258c8298b176f41": {
    "Name": "telemetry-receiver",
    "EndpointID":
    "9309cd520ab3c8f17a37bc1c68f773dd03de4e521c3e3cb5111f284734979230",
    "MacAddress": "02:42:ac:13:00:06",
    "IPv4Address": "172.19.0.6/16",
    "IPv6Address": ""
  },

  "eff7679a19274cb662f5693c2cd211f06ccf5cfb3126b59b8b24203cbd0add60": {
    "Name": "influx-ingester",
    "EndpointID":
    "2bfef225098634f5ad5348a070f4562ae0b85f264611f8ec317585e258005c4d",
    "MacAddress": "02:42:ac:13:00:05",
    "IPv4Address": "172.19.0.5/16",
    "IPv6Address": ""
  }
},
"Options": {},
"Labels": {
  "com.docker.compose.network": "influxpipeline",
  "com.docker.compose.project": "docker-compose-files",
  "com.docker.compose.version": "1.27.4"
}
}
]

```

Note that Grafana IP in this case is 172.19.0.7, telemetry receiver IP is 172.19.0.6.

Please follow the instructions in section 3.4 to add source iDRAC telemetry services.

5.3 Data Model and Applications

To configure the data source and visualization dashboards please access Grafana homepage in the browser (<http://172.19.0.7:3000>)

InfluxDB data source can be added from Add data source option and once the data source is added dashboards and alerts can be created for the specific use case of interest

The screenshot shows the Grafana web interface for configuring an InfluxDB data source. The page title is "Data Sources / InfluxDB" with the subtitle "Type: InfluxDB". The "InfluxDB" data source is selected and its status is "Default" with a toggle switch turned on. The "Query Language" is set to "InfluxQL". Under the "HTTP" section, the "URL" is "http://192.168.112.3:8086", the "Access" is "Server (default)", and there is a "Whitelisted Cookies" section with a "New tag" button and an "Add" button. The "Auth" section includes "Basic auth" (disabled), "TLS Client Auth" (disabled), "Skip TLS Verify" (disabled), and "Forward OAuth Identity" (disabled). The "Custom HTTP Headers" section has an "Add header" button. The "InfluxDB Details" section contains a "Database Access" warning box, a "Database" field set to "poweredge_telemetry_metrics", a "User" field, a "Password" field set to "configured" with a "Reset" button, an "HTTP Method" dropdown set to "GET", a "Min time interval" field set to "10s", and a "Max series" field set to "1000".

Data Sources / InfluxDB
Type: InfluxDB

InfluxDB Default ☒

Query Language
InfluxQL

HTTP
URL:
Access: [Help](#)
Whitelisted Cookies: [Add](#)

Auth
Basic auth: ☐ With Credentials: ☐
TLS Client Auth: ☐ With CA Cert: ☐
Skip TLS Verify: ☐
Forward OAuth Identity: ☐

Custom HTTP Headers
[+ Add header](#)

InfluxDB Details
Database Access
Setting the database for this datasource does not deny access to other databases. The InfluxDB query syntax allows switching the database in the query. For example: `SHOW MEASUREMENTS ON _internal` Or `SELECT * FROM "_internal"..<database> LIMIT 50`
To support data isolation and security, make sure appropriate permissions are configured in InfluxDB.

Database:
User:
Password: [Reset](#)
HTTP Method:
Min time interval:
Max series:

6 Setting up the Timescale data ingestion pipeline

6.1 Prerequisites

Go - <https://golang.org/>

I had go version go version go1.15.6 linux/amd64. But technically any recent version should be just fine.

ActiveMQ

Download version apache-activemq-5.16.0 from
<https://activemq.apache.org/components/classic/download/>

TimescaleDB 2.0

(<https://docs.timescale.com/latest/getting-started/installation>)

Follow through out to setup the memory and WAL parameter tunings.

A great summary of timescale data model capabilities are outlined in the timescale onlinedocumentation -

<https://docs.timescale.com/latest/introduction/data-model>

6.2 Instructions to run the pipeline in docker compose

Included is a reference docker compose configuration file.

timescale-docker-pipeline-reference-unenc.yml – starts timescale/postgres, grafana and ingest applications as docker containers. Docker compose configuration is not enabled to be secure or persistent and is required to be modified to suit the needs of deployment environment.

```
athena@athena-PowerEdge-R640:~/iDRAC-Telemetry-Reference-Tools$ docker-compose -f  
docker-compose-files/timescale-docker-pipeline-reference-unenc.yml up -d
```

```
Creating activemq    ... done  
Creating mysqldb     ... done  
Creating timescale   ... done  
Creating grafana     ... done  
Creating timescale-ingester ... done  
Creating telemetry-receiver ... done
```

```
athena@athena-PowerEdge-R640:~/iDRAC-Telemetry-Reference-Tools$ docker-compose -f  
docker-compose-files/timescale-docker-pipeline-reference-unenc.yml ps
```


Ports	Name	Command	State
61616/tcp, 8161/tcp	activemq	/bin/bash -c bin/activemq ...	Up
3000/tcp	grafana	/run.sh	Up
3306/tcp	mysqldb	docker-entrypoint.sh mysqld	Up
	telemetry-receiver	/bin/sh -c cmd/idrac-telem ...	Up
	timescale	/bin/sh -c cmd/initialize_ ...	Up
	timescale-ingester	go run cmd/timescalepump/t ...	Up

athena@athena-PowerEdge-R640:~/iDRAC-Telemetry-Reference-Tools\$ docker network ls

NETWORK ID	NAME	DRIVER	SCOPE
cf13f260922c	bridge	bridge	local
68c0c061256b	docker-compose-files_timescalepipeline	bridge	local
4d6d8dcd18da	host	host	local

athena@athena-PowerEdge-R640:~/dell-github/iDRAC-Telemetry-Reference-Tools\$ docker network inspect docker-compose-files_timescalepipeline

```
[
  {
    "Name": "docker-compose-files_timescalepipeline",
    "Id": "a451c1969bcd6a2dbf9892a63aeb9f5e7b47f204672c5dc0f50d09bf0a835fa0",
    "Created": "2021-05-14T19:16:58.162137338-05:00",
    "Scope": "local",
    "Driver": "bridge",
    "EnableIPv6": false,
    "IPAM": {
      "Driver": "default",
      "Options": null,
      "Config": [
        {
          "Subnet": "192.168.192.0/20",
          "Gateway": "192.168.192.1"
        }
      ]
    },
    "Internal": false,
```

```
    "Attachable": true,
    "Ingress": false,
    "ConfigFrom": {
      "Network": ""
    },
    "ConfigOnly": false,
    "Containers": {

      "008d03e7e66130c9883f2b58600fde5416040d2278336c73127a8128383bcad8": {
        "Name": "activemq",
        "EndpointID":
"5c34a46026eb58fb301936bf48bfd2adc3a9b94496a6df339d2d4132eeb4ff49",
        "MacAddress": "02:42:c0:a8:c0:04",
        "IPv4Address": "192.168.192.4/20",
        "IPv6Address": ""
      },

      "1a44797f4ddd7f2a499f6aad59fa684d4c2c54472fd2a5b59c810761d14dd776": {
        "Name": "telemetry-receiver",
        "EndpointID":
"ca494fb40d6c3161a503f6c65f686a482e316e512ca97e5ebedcf2a9ac6eefc",
        "MacAddress": "02:42:c0:a8:c0:07",
        "IPv4Address": "192.168.192.7/20",
        "IPv6Address": ""
      },

      "2592245fa1a34f67266afec1027a152e4e9fe9cfd26dbb01a0ac184be62f4a54": {
        "Name": "timescale-ingester",
        "EndpointID":
"6c551b0d7f4b57a41c0004cedbc988296006cc3eb5c64c78d714f06586703bd3",
        "MacAddress": "02:42:c0:a8:c0:06",
        "IPv4Address": "192.168.192.6/20",
        "IPv6Address": ""
      },
    },
  },
}
```

```

"4b8ecc84f78f04625613d023c18cb7792527bc8d1d8d0975abb7cee6d86a76b3": {
    "Name": "grafana",
    "EndpointID":
"2075cbe6a00268c6f05a05c6d452b7520a720a9f86a3601d6d0967268f8c0200",
    "MacAddress": "02:42:c0:a8:c0:02",
    "IPv4Address": "192.168.192.2/20",
    "IPv6Address": ""
},

"e3b908f848df5056d124d6cad6ccb3d39c7e324578874b72c250ef49d5d3c91e": {
    "Name": "mysqldb",
    "EndpointID":
"37b0e49456ef8dd4d270c07de5d6cf6aa5d9331a285881a9b7359ec4def1cae3",
    "MacAddress": "02:42:c0:a8:c0:03",
    "IPv4Address": "192.168.192.3/20",
    "IPv6Address": ""
}
},
"Options": {},
"Labels": {
    "com.docker.compose.network": "timescalepipeline",
    "com.docker.compose.project": "docker-compose-files",
    "com.docker.compose.version": "1.27.4"
}
}
]

```

Note that Grafana IP in this case is 192.168.192.2, telemetry receiver IP is 192.168.192.7.

Please follow the instructions in section 3.4 to add source iDRAC telemetry services.

6.3 Data Model and Applications

Postgres data source can be added from Add data source option and once the data source is added dashboards and alerts can be created for the specific use case of interest.

The screenshot shows the Grafana interface for configuring a PostgreSQL data source. The page title is "Data Sources / PostgreSQL" with a sub-label "Type: PostgreSQL". A "Settings" tab is active. The configuration includes a "Name" field set to "PostgreSQL" and a "Default" toggle. The "PostgreSQL Connection" section contains fields for "Host" (localhost:5432), "Database" (poweredge_telemetry_metrics), "User" (postgres), "Password" (configured), and "SSL Mode" (disable). A "Reset" button is next to the password field. The "Connection limits" section has "Max open" (unlimited), "Max idle" (2), and "Max lifetime" (14400). The "PostgreSQL details" section includes "Version" (12), "TimescaleDB" (disabled), and "Min time interval" (1m). A "User Permission" warning box is present. At the bottom, a green bar indicates "Database Connection OK", and buttons for "Save & Test", "Delete", and "Back" are visible.

Data Sources / PostgreSQL
Type: PostgreSQL

Settings

Name: PostgreSQL Default

PostgreSQL Connection

Host: localhost:5432
Database: poweredge_telemetry_metrics
User: postgres Password: configured Reset
SSL Mode: disable

Connection limits

Max open: unlimited
Max idle: 2
Max lifetime: 14400

PostgreSQL details

Version: 12
TimescaleDB: Help
Min time interval: 1m

User Permission

The database user should only be granted SELECT permissions on the specified database & tables you want to query. Grafana does not validate that queries are safe so queries can contain any SQL statement. For example, statements like `DELETE FROM user;` and `DROP TABLE user;` would be executed. To protect against this we Highly recommend you create a specific PostgreSQL user with restricted permissions.

✓ Database Connection OK

Save & Test Delete Back

7 Setting up the Prometheus data ingestion pipeline

7.1 Prerequisites

Go - <https://golang.org/>

I had go version go version go1.15.6 linux/amd64. But technically any recent version should be just fine.

ActiveMQ

Download version apache-activemq-5.16.0 from
<https://activemq.apache.org/components/classic/download/>

Prometheus

(<https://prometheus.io/docs/prometheus/latest/installation>)

7.2 Instructions to run the pipeline in docker compose

Included is a reference docker compose configuration file.

prometheus-docker-pipeline-reference-unenc.yml – starts prometheus, grafana and ingest applications as docker containers. Docker compose configuration is not enabled to be secure or persistent and is required to be modified to suit the needs of deployment environment.

```
athena@athena-PowerEdge-R640:~/iDRAC-Telemetry-Reference-Tools$ docker-compose -f
docker-compose-files/prometheus-docker-pipeline-reference-unenc.yml up -d
```

```
Creating network "docker-compose-files_prometheuspipeline" with driver
"bridge"
Creating activemq    ... done
Creating grafana     ... done
Creating prometheus  ... done
Creating mysqldb     ... done
Creating prometheus-ingester ... done
Creating telemetry-receiver ... done
```

```
athena@athena-PowerEdge-R640:~/iDRAC-Telemetry-Reference-Tools$ docker-compose -f
docker-compose-files/prometheus-docker-pipeline-reference-unenc.yml ps
```

Name	Command	State
Ports		


```

activemq          /bin/bash -c bin/activemq ... Up
61616/tcp, 8161/tcp
grafana           /run.sh Up 3000/tcp
mysqldb           docker-entrypoint.sh mysqld Up
3306/tcp, 33060/tcp
prometheus        /bin/prometheus --config.f ... Up 9090/tcp
prometheus-ingester go run cmd/prometheuspump/ ... Up
telemetry-receiver /bin/sh -c cmd/idrac-telem ... Up

```

athena@athena-PowerEdge-R640:~/iDRAC-Telemetry-Reference-Tools\$ docker network ls

NETWORK ID	NAME	DRIVER	SCOPE
cf13f260922c	bridge	bridge	local
68c0c061256b	docker-compose-files_prometheuspipeline	bridge	local
4d6d8dcd18da	host	host	local

athena@athena-PowerEdge-R640:~/dell-github/iDRAC-Telemetry-Reference-Tools\$ docker network inspect docker-compose-files_prometheuspipeline

```

[
  {
    "Name": "docker-compose-files_prometheuspipeline",
    "Id": "6301419378a7e3da153899750ead81c8f4fbba4b3d971cf895b6ca0dd2fd2bf8",
    "Created": "2021-06-28T19:50:08.901957363-05:00",
    "Scope": "local",
    "Driver": "bridge",
    "EnableIPv6": false,
    "IPAM": {
      "Driver": "default",
      "Options": null,
      "Config": [
        {
          "Subnet": "172.20.0.0/16",
          "Gateway": "172.20.0.1"
        }
      ]
    },
    "Internal": false,
    "Attachable": true,
    "Ingress": false,
    "ConfigFrom": {

```

```
        "Network": ""
    },
    "ConfigOnly": false,
    "Containers": {

        "05898f35d11de7ce5d02313640e08da7c63d36855e711f0793024f7691edaeec": {
            "Name": "activemq",
            "EndpointID":
            "fb4fc668c6c7bf603f80e60f9eee213dfd06ed6842e28c08265ab682fc23c446",
            "MacAddress": "02:42:ac:14:00:03",
            "IPv4Address": "172.20.0.3/16",
            "IPv6Address": ""
        },

        "17fe44e410a8f116820268274a49b4098223ab120476f0801d1d5694a7b37b5b": {
            "Name": "prometheus-ingester",
            "EndpointID":
            "ae375f0ec45a7d28c8d4e490cb021add114db074312122f2aac0221baef1e141",
            "MacAddress": "02:42:ac:14:00:06",
            "IPv4Address": "172.20.0.6/16",
            "IPv6Address": ""
        },

        "8bb5db8c19404c5da5100ebdc562de8345c9098e3cac7879c1c9707c415a8ec4": {
            "Name": "mysqldb",
            "EndpointID":
            "6a48ca12fd82b861f399f6ec60a858c8472bb4a863b14df3a64d8582f75d9891",
            "MacAddress": "02:42:ac:14:00:05",
            "IPv4Address": "172.20.0.5/16",
            "IPv6Address": ""
        },

        "dae51c68b9764e1fb6e53247b4c4c30e85096a21b2df9c546678d15bd3bec6e4": {
            "Name": "grafana",
```

```

        "EndpointID":
"68c4ab2b5094ee784c4bad06f44537ec9a0592ac0ac65d568069027c43090611",
        "MacAddress": "02:42:ac:14:00:02",
        "IPv4Address": "172.20.0.2/16",
        "IPv6Address": ""
    },

    "db9a3bd4520efec037eaff5fb8b4df59d9a2c92388809c838f68d4a03e57ed68": {
        "Name": "prometheus",
        "EndpointID":
"c43e37a690dfeeb8ea81fb8e1525c7b2f4a66f96a9610625f2d29aaecdd24d72",
        "MacAddress": "02:42:ac:14:00:04",
        "IPv4Address": "172.20.0.4/16",
        "IPv6Address": ""
    },

    "e74b37bd09a6392af2fafd2b670c6f47d35ec8c7bc681b9ce0b8f5d54959fcb8": {
        "Name": "telemetry-receiver",
        "EndpointID":
"438b85b2eeddca1b31f9de520de6f6c2027f8a4b02b2380431a33e5f46d18f27",
        "MacAddress": "02:42:ac:14:00:07",
        "IPv4Address": "172.20.0.7/16",
        "IPv6Address": ""
    }
},
"Options": {},
"Labels": {
    "com.docker.compose.network": "prometheuspipeline",
    "com.docker.compose.project": "docker-compose-files",
    "com.docker.compose.version": "1.27.4"
}
}
]

```


Note that Grafana IP in this case is 172.20.0.2, telemetry receiver IP is 172.20.0.7

Please follow the instructions in section 3.4 to add source iDRAC telemetry services.

7.3 Data Model and Applications

To configure the data source and visualization dashboards please access Grafana homepage in the browser (<http://172.20.0.2:3000>)

Prometheus data source (**172.20.0.4:9090**) can be added from Add data source option and once the data source is added dashboards and alerts can be created for the specific use case of interest.

© 172.28.0.3:3000/data-sources/edit/ 1/ 40% ...

Data Sources / Prometheus

Type: Prometheus

Settings Dashboards

Configure your Prometheus data source below

Or skip the effort and get Prometheus (and Loki) as fully managed, scalable and hosted data sources from Grafana Labs with the [free forever Grafana Cloud plan](#)

Name Prometheus Default

HTTP

URL http://172.28.0.2:9090

Access Server (default) Help

Whitelisted Cookies New tag (enter key to add) Add

Auth

Basic auth	With Credentials
TLS Client Auth	With CA Cert
Skip TLS Verify	
Forward OAuth Identity	

Custom HTTP Headers

+ Add header

Scrape interval 15s

Query timeout 30s

HTTP Method POST

Misc

Disable metrics lookup

Custom query parameters Example: max_source_resolution=5m&timeout=10

Exemplars

+ Add

✓ Data source is working

Save & Test Delete Back