

iDRAC Telemetry Reference Tools Setup Instructions (DRAFT)

This technical document describes Telemetry opensource tool setup instructions for various big database reference stacks.

Dell Server Software 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 InfluxDB stack setup instructions.

Table of contents

Revisions	2
1 Setting up Dell servers with Telemetry and Enable Telemetry reports	4
2 Assumptions and other technical considerations	5
3 Setting up the Elasticsearch data ingestion pipeline	6
3.1 Prerequisites	6
3.2 Instructions to run the pipeline in docker compose	6
3.3 Instructions to run the pipeline as standalone applications	9
3.4 Data Model and Applications	10
4 Setting up the InfluxDB data ingestion pipeline	15
4.1 Prerequisites	15
4.2 Instructions to run the pipeline in docker compose	15
4.3 Data Model and Applications	17
5 Setting up the Timescale data ingestion pipeline	19
5.1 Prerequisites	19
5.2 Instructions to run the pipeline in docker compose	19

1 Setting up Dell servers with Telemetry and Enable Telemetry reports

- 1) Install firmware version 4.00.00 or higher in PowerEdge iDRACs.
- 2) Download the telemetry utilities from-
<https://github.com/dell/iDRACTelemetry-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 and Alerts.

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  
python3 ./ConfigurationScripts/SubscriptionManagementREDFISH.py -ip $target -u $user -p $password -c y
```

(Ignore the errors)

2 Assumptions and other technical considerations

- 1) The provided docker compose files 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. It uses a message bus-based architecture. The reference toolset can be also run standalone. Please refer section 3.3 for setup instructions.

3 Setting up the Elasticsearch data ingestion pipeline

3.1 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

apache-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>)

3.2 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.

Update config.ini

Update the config.ini file with following details

- IP and credentials of idracs

```
athena@athena-PowerEdge-R640:~/toolset/telemetryservice$ docker-compose -f docker-  
compose-files/elastic-docker-pipeline-reference-unenc.yml up -d  
Creating network "telemetryservice_elastic" with driver "bridge"  
Creating es02      ... done  
Creating activemq  ... done  
Creating es01      ... done  
Creating es03      ... done  
Creating telemetryservice_authapp_1      ... done  
Creating telemetryservice_discapp_1      ... done  
Creating telemetryservice_redfishreadapp_1 ... done  
Creating telemetryservice_elasticsearchpump_1 ... done  
Creating kib01     ... done  
athena@athena-PowerEdge-R640:~/toolset/telemetryservice$ docker-compose -f docker-  
compose-files/elastic-docker-pipeline-reference-unenc.yml ps  
Name                                Command                                State  
Ports
```

```

-----
activemq /bin/bash -c bin/activemq ... Up
61616/tcp, 8161/tcp
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
telemetryservice_authapp_1 go run cmd/simpleauth/simp ... Up
telemetryservice_discapp_1 go run cmd/simpliedisc/simp ... Up
telemetryservice_elasticsearchpump_1 go run cmd/elkpump/elkpump ... Up
telemetryservice_redfishreadapp_1 go run cmd/redfishread/red ... Up

```

athena@athena-PowerEdge-R640:~/toolset/telemetryservice\$ docker network ls

NETWORK ID	NAME	DRIVER	SCOPE
1e4f95d0f3ea	bridge	bridge	local
4d6d8dcd18da	host	host	local
13626d259a09	none	null	local
65b45432b28a	telemetryservice_elastic	bridge	local

athena@athena-PowerEdge-R640:~/toolset/telemetryservice\$ docker network inspect

```

telemetryservice_elastic
[
  {
    "Name": "telemetryservice_elastic",
    "Id": "65b45432b28ae269b1645bf69487104d2626a5a4d0dabcb864945840a1ecda88",
    "Created": "2021-01-26T12:19:57.518320389-06:00",
    "Scope": "local",
    "Driver": "bridge",
    "EnableIPv6": false,
    "IPAM": {
      "Driver": "default",
      "Options": null,
      "Config": [
        {
          "Subnet": "172.24.0.0/16",
          "Gateway": "172.24.0.1"
        }
      ]
    },
    "Internal": false,
    "Attachable": true,
    "Ingress": false,
    "ConfigFrom": {
      "Network": ""
    },
    "ConfigOnly": false,
    "Containers": {
      "1c424cb2e8447b207d024618c01df2e96b9852f64c343438bb4341abf26a7958": {
        "Name": "kib01",
        "EndpointID":
"1aa125fd85499968611538a043cbe0f0c090eb7475b563c21321301f075d4452",
        "MacAddress": "02:42:ac:18:00:0a",
        "IPv4Address": "172.24.0.10/16",
        "IPv6Address": ""
      },
      "41a93d84ecedebb18917d6c79595513283d073acc8e7a1c7ca384603a9dc4dce": {
        "Name": "es02",

```

```
        "EndpointID":
"8edcde63093c4909399d16ab0ca587bd3c5ce26d50ef7f4fe09d113ad9f362eb",
        "MacAddress": "02:42:ac:18:00:04",
        "IPv4Address": "172.24.0.4/16",
        "IPv6Address": ""
    },
    "9165d7f79373a7d81cab1c69b6cb5bc2834c985c4e7aa92c065a39d536c4d9ee": {
        "Name": "telemetryservice_discapp_1",
        "EndpointID":
"ad3ab006a39838492322f3f6ff206ed355cfed9160f8a5324e9441075bf8f0c3",
        "MacAddress": "02:42:ac:18:00:06",
        "IPv4Address": "172.24.0.6/16",
        "IPv6Address": ""
    },
    "9bec504349ecef3614f1c46fc1298bd67d3c6b8d44ee2faa81aa081b553f234a": {
        "Name": "es01",
        "EndpointID":
"2c037932e8d5045096ad2b46db6942c4137f491c9306ee3544953a5259bf94c7",
        "MacAddress": "02:42:ac:18:00:02",
        "IPv4Address": "172.24.0.2/16",
        "IPv6Address": ""
    },
    "9cc5b47550b4d120708a03d297b48067a2264e5405b9556d4e61e9a01b40abd6": {
        "Name": "activemq",
        "EndpointID":
"beac9436d295c8f7f9ed7f064f186f3b951e670320a0c8d60ff8eca0b756b33a",
        "MacAddress": "02:42:ac:18:00:05",
        "IPv4Address": "172.24.0.5/16",
        "IPv6Address": ""
    },
    "af05c1e0723e96e739d1b77f1d2315b753b253f2077cb4bb6ce066d3dff23d91": {
        "Name": "es03",
        "EndpointID":
"8ab71852c078d081898575330883124ccea7286c890c010b3f787b8b16f7b1a1",
        "MacAddress": "02:42:ac:18:00:03",
        "IPv4Address": "172.24.0.3/16",
        "IPv6Address": ""
    },
    "c8e516347112661f5ee6aaa8ffada24a423d391c6b857b860e528fa74924c60f": {
        "Name": "telemetryservice_redfishreadapp_1",
        "EndpointID":
"50ce7731f6a47da663b41e4f4e7c9d40ca45891d3c18e8b31f56cb3469eb5547",
        "MacAddress": "02:42:ac:18:00:08",
        "IPv4Address": "172.24.0.8/16",
        "IPv6Address": ""
    },
    "c90deb7e01413c2fbc3fa84baf30355c7fd9b1c9c4b8849df2bf482abdb093ea": {
        "Name": "telemetryservice_authapp_1",
        "EndpointID":
"56ff39f4ab33048a218dcbeb57cf24b7a58c7d74f4456e0b4522274889437eac",
        "MacAddress": "02:42:ac:18:00:09",
        "IPv4Address": "172.24.0.9/16",
        "IPv6Address": ""
    },
    "e3a129b8dcaf292729e35b76823b4989d54cf8678b45262454fe39f48ea6c143": {
        "Name": "telemetryservice_elasticsearchpump_1",
        "EndpointID":
"38944a460acf430a2533d3e67b39a4d86f69b1f6acb0ab6735cf2b72449ae4c2",
        "MacAddress": "02:42:ac:18:00:07",
        "IPv4Address": "172.24.0.7/16",
        "IPv6Address": ""
    }
},
```



```

    "Options": {},
    "Labels": {
        "com.docker.compose.network": "elastic",
        "com.docker.compose.project": "telemetryservice",
        "com.docker.compose.version": "1.27.4"
    }
}
]

```

Note that Kibana IP in this case is 172.24.0.10

3.3 Instructions to run the pipeline as standalone applications

Update config.ini

Update the config.ini file with following details

- Stomp host names (activemq host name)
- IP and credentials of idracs
-

Setup:

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.

- * <go> build .\cmd\simplifiedisc\simplifiedisc.go
- * <go> build .\cmd\simpleauth\simpleauth.go
- * <go> build .\cmd\redfishread\redfishread.go
- * <go> build .\cmd\elkpump\elkpump-basic.go

Now get ActiveMQ running, the default config is fine

- * cd to the directory you unpacked or installed ActiveMQ to
- * bin\activemq start

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

- * simplifiedisc
- * simpleauth
- * redfishread
- * elkpump-basic

3.4 Data Model and Applications

To configure the data source and visualization dashboards please access Kibana homepage in the browser (<http://172.24.0.10:5601>)

- 1) Select Stack Management from the Management section of the tools menu.
- 2) 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	uuid	pri	rep
docs.count	docs.deleted	store.size	pri.store.size		
green	open	poweredge_telemetry_metrics	Fn0wnfrDTJqyb-9DuLns7A	1	1
116	1557849	206.9mb	102.7mb		
green	open	.apm-custom-link	2uLRNhg4TFigKVfz36p1Kg	1	1
0	0	416b	208b		
green	open	.kibana_task_manager_1	1S5B2z7_QyGAJbr_YXAWNQ	1	1
5	5559	1mb	544.4kb		
green	open	.apm-agent-configuration	0noOBC1MRhee2PeCZJ5X1Q	1	1
0	0	416b	208b		
green	open	.kibana-event-log-7.10.1-000001	gyCgu-ZoSv-DppxsmkThEQ	1	1
1	0	11.2kb	5.6kb		
green	open	.async-search	j5VuQCQqTHyaf3OHJB3mTA	1	1
0	0	541.6kb	269.2kb		
green	open	.kibana_1	bWwS4bQHS_yVcb67AfN5wg	1	1
52	11	4.3mb	2.1mb		

GET /poweredge_telemetry_metrics/_settings

```
{
  "poweredge_telemetry_metrics" : {
    "settings" : {
      "index" : {
        "routing" : {
          "allocation" : {
            "include" : {
              "_tier_preference" : "data_content"
            }
          }
        },
        "number_of_shards" : "1",
        "provided_name" : "poweredge_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 /poweredge_telemetry_metrics/_mapping

```
{
  "poweredge_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"
        }
      }
    }
  }
}
```

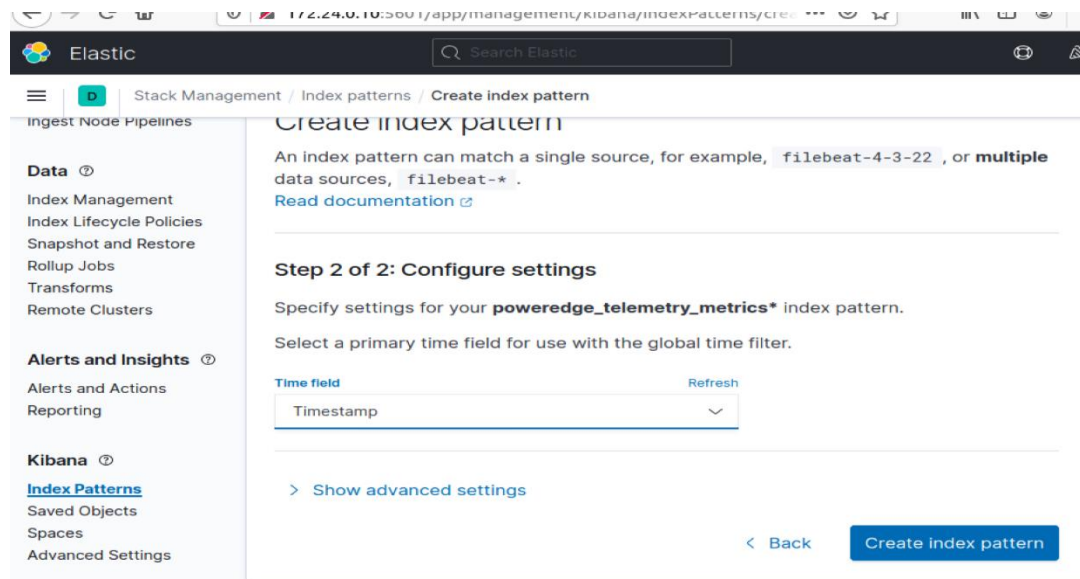
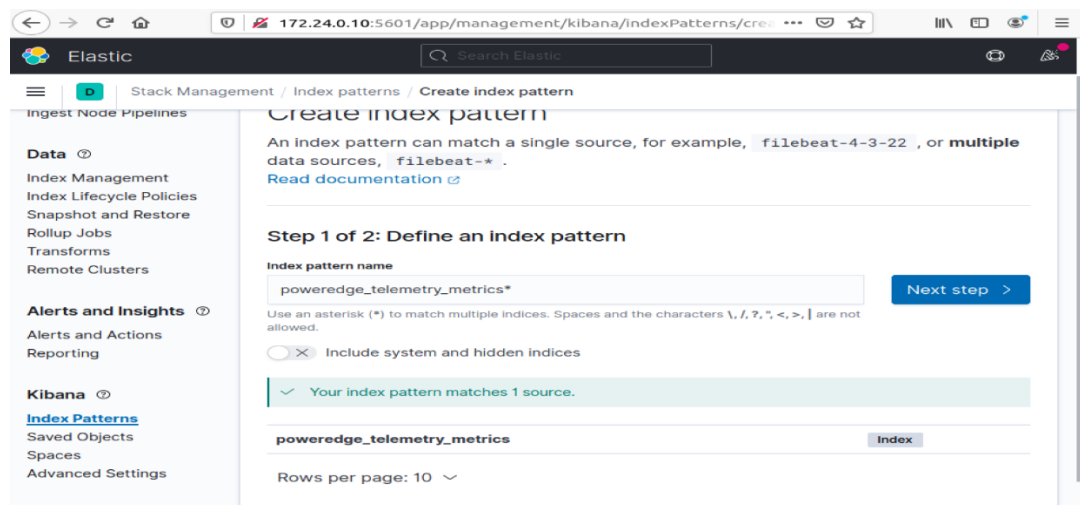
```

    },
    "ValueInt" : {
      "type" : "long"
    }
  }
}
}
}

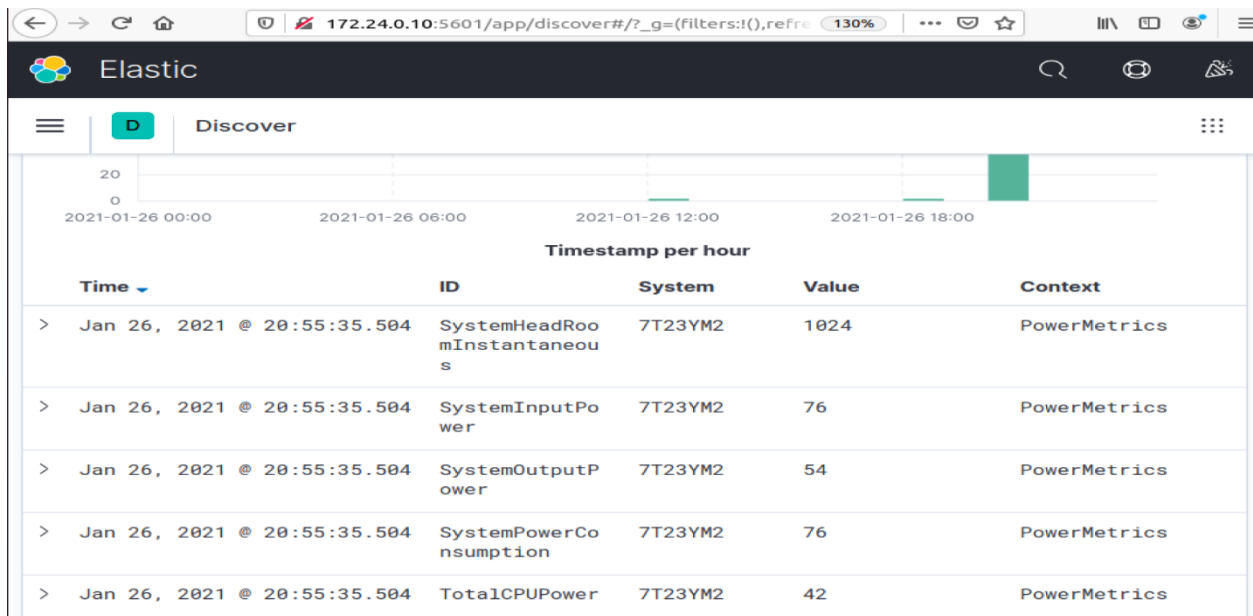
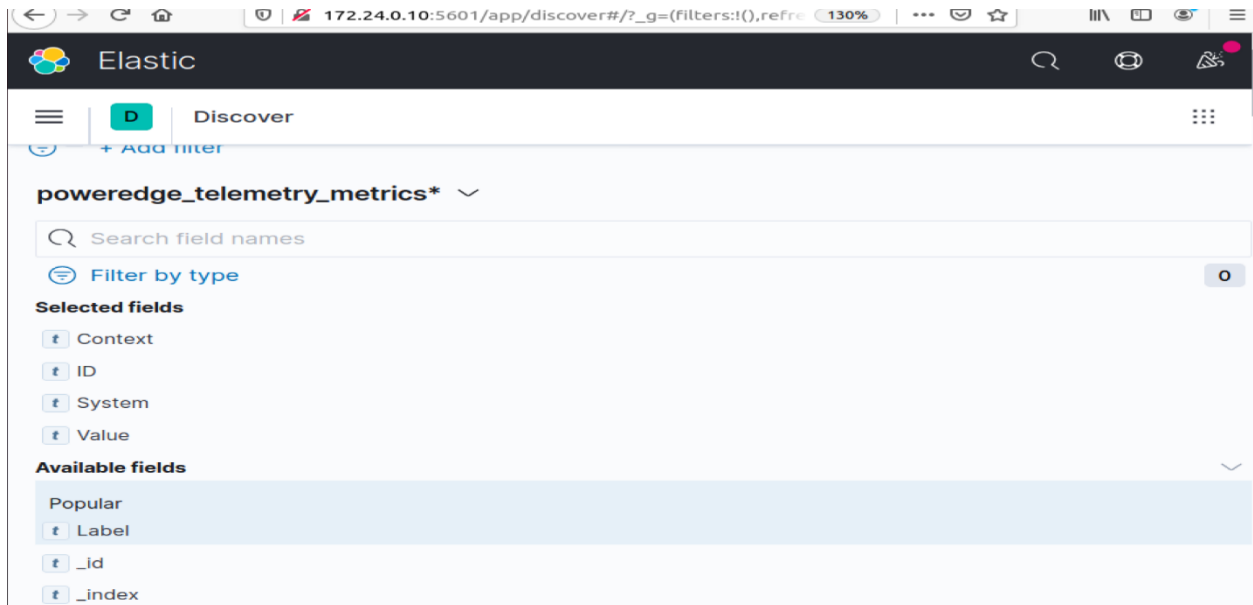
```

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)

3) Create an index pattern for Kibana at index pattern tab.



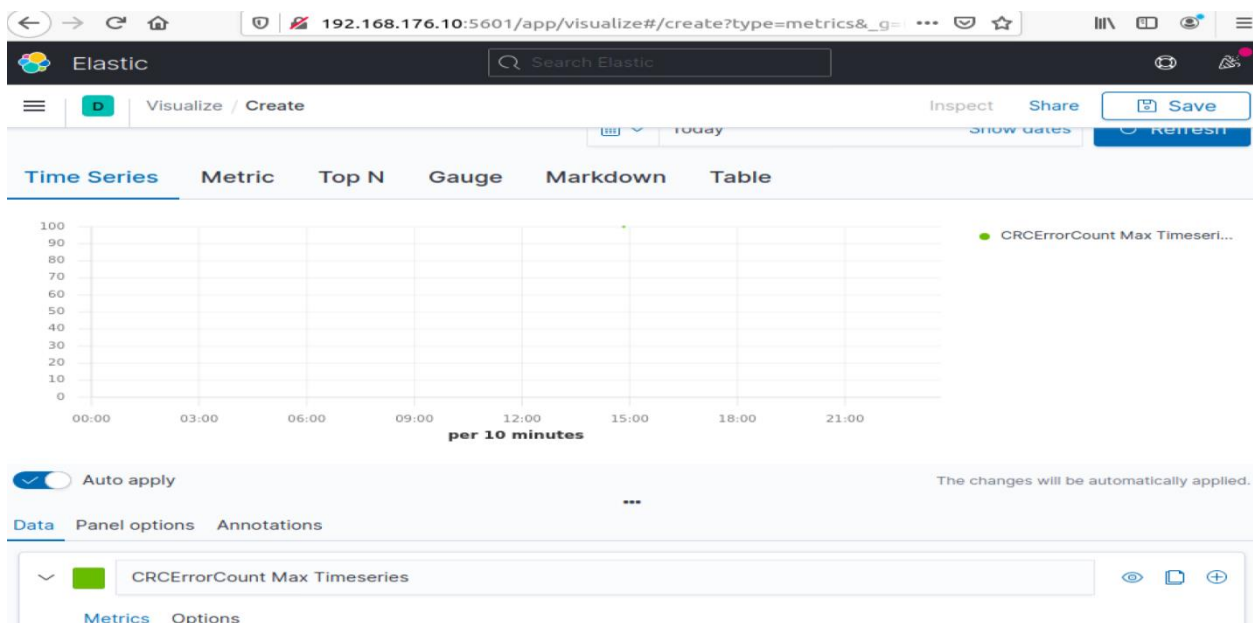
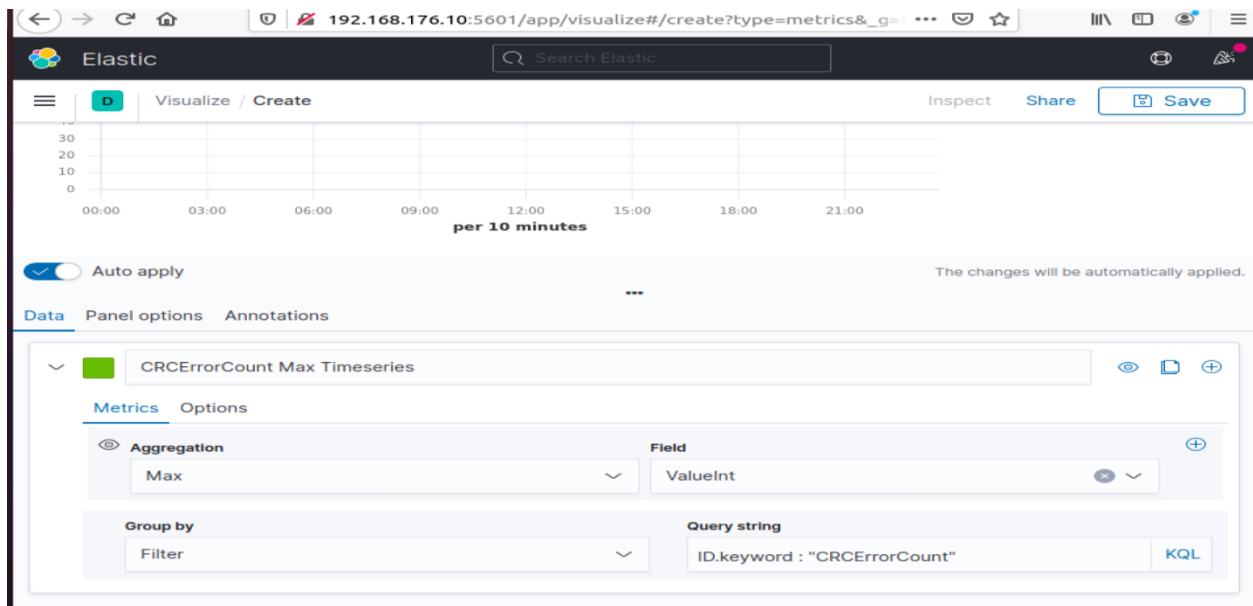
- 4) Now from Discover tab, you can view the ingested data. You can also select the fields of interest and view the data in a tabular form with selected fields.



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

- 5) 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 aggregatable values (non text) can be used to configure relevant visualizations.

A timeseries label is configured as Max aggregation on CRCErrCount 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.

4 Setting up the InfluxDB data ingestion pipeline

4.1 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/>

4.2 Instructions to run the pipeline in docker compose

Included is a reference docker compose configuration file.

influx-docker-pipeline-reference-unenc.yml

Update config.ini

Update the config.ini file with following details

- IP and credentials of idracs

```
athena@athena-PowerEdge-R640~/toolset/telemetryservice: docker-compose -f docker-compose-
files/influx-docker-pipeline-reference-unenc.yml up -d
Creating network "docker-compose-files_influxpipeline" with driver "bridge"
Creating influx ... done
Creating activemq01 ... done
Creating docker-compose-files_redfishreadapp_1 ... done
Creating docker-compose-files_influxpump_1 ... done
Creating docker-compose-files_discapp_1 ... done
Creating docker-compose-files_authapp_1 ... done
Creating grafana ... done
```

```
athena@athena-PowerEdge-R640~/toolset/telemetryservice: docker-compose -f docker-compose-
files/influx-docker-pipeline-reference-unenc.yml ps
```

Ports	Name	Command	State
-------	------	---------	-------

```

-----
activemq01                               /bin/bash -c bin/activemq ... Up
61616/tcp, 8161/tcp
docker-compose-files_authapp_1          go run cmd/simpleauth/simp ... Up
docker-compose-files_discapp_1          go run cmd/simpdisc/simp ... Up
docker-compose-files_influxpump_1       go run cmd/influxpump/infl ... Up
docker-compose-                           go run cmd/redfishread/red ... Up
files_redfishreadapp_1
grafana                                  /run.sh Up 3000/tcp
influx                                  /entrypoint.sh influxd Up (healthy)
8086/tcp

```

athena@athena-PowerEdge-R640~/toolset/telemetryservice:~/telemetry-opensource-home/telemetry-reference-tools\$ docker inspect docker-compose-files_influxpipeline

```

[
  {
    "Name": "docker-compose-files_influxpipeline",
    "Id": "2ccf9680801448d0c77d9aa1a856423b355786104d50498ec765943fa5af38a7",
    "Created": "2021-04-06T20:08:14.182802852-05:00",
    "Scope": "local",
    "Driver": "bridge",
    "EnableIPv6": false,
    "IPAM": {
      "Driver": "default",
      "Options": null,
      "Config": [
        {
          "Subnet": "192.168.112.0/20",
          "Gateway": "192.168.112.1"
        }
      ]
    },
    "Internal": false,
    "Attachable": true,
    "Ingress": false,
    "ConfigFrom": {
      "Network": ""
    },
    "ConfigOnly": false,
    "Containers": {
      "0310d0dd408eaca6c4b0dlbb72f4c550701a4aa107b82d564b6a0779fa0646e3": {
        "Name": "docker-compose-files_discapp_1",
        "EndpointID":
"916a794cc535c808a87638e4d67d8c040037935002f940b6fbbba4653035bc329",
        "MacAddress": "02:42:c0:a8:70:04",
        "IPv4Address": "192.168.112.4/20",
        "IPv6Address": ""
      },
      "8b72452e97aabd671d293d4cfd2f1f37ba199549fc6b2365e4ab01db84da411e": {
        "Name": "activemq01",
        "EndpointID":
"ba4a99299ce4e3b574a462b4cee234966823a2c3a379f97928977966956c52d2",
        "MacAddress": "02:42:c0:a8:70:02",
        "IPv4Address": "192.168.112.2/20",
        "IPv6Address": ""
      },
      "97f6bf1266699b931ac72e30a2977cb34768cd3bce4a1893c7db26b18212b933": {
        "Name": "grafana",
        "EndpointID":
"09b1e0643d5ebcfaebce3c8ecccc2561f74c42c34e472ec65ae0cfeef2491ce7",
        "MacAddress": "02:42:c0:a8:70:08",
        "IPv4Address": "192.168.112.8/20",

```



```

        "IPv6Address": ""
    },
    "ae717d5417f668fb66c65d2f018317453c0500d1914713e8d7416277025eaf67": {
        "Name": "influx",
        "EndpointID":
"ef92c87ea5fe8ec3d79602c023f1f12b8428027ce63bd61ce90fd724500c3b57",
        "MacAddress": "02:42:c0:a8:70:03",
        "IPv4Address": "192.168.112.3/20",
        "IPv6Address": ""
    },
    "c5e17d59615b066ba5e163e7671fe857c1fdc0a2135463feb960c50391dc273f": {
        "Name": "docker-compose-files_authapp_1",
        "EndpointID":
"f4eaed22c293817c0ecd25a6801928ab01f76549197834124c898505397f630b",
        "MacAddress": "02:42:c0:a8:70:07",
        "IPv4Address": "192.168.112.7/20",
        "IPv6Address": ""
    },
    "d2778be129bbfb27b540fd453bcc6b4f9ecb0e0ec29e632410f20649ecdccfd1": {
        "Name": "docker-compose-files_redfishreadapp_1",
        "EndpointID":
"7485dbc9aba901efaf1df8a872d08bea5c6301c15640f5b66719ece7c2c1a1e5",
        "MacAddress": "02:42:c0:a8:70:06",
        "IPv4Address": "192.168.112.6/20",
        "IPv6Address": ""
    },
    "e32198358d06a16a92aedbf1c7a8b38558b3742bed6f4951712af776bef0f554": {
        "Name": "docker-compose-files_influxpump_1",
        "EndpointID":
"c98192ccfba25a3ec46c74708f372a2befbbe3a20ea4571cae9367a00f33480c",
        "MacAddress": "02:42:c0:a8:70:05",
        "IPv4Address": "192.168.112.5/20",
        "IPv6Address": ""
    }
},
"Options": {},
"Labels": {
    "com.docker.compose.network": "influxpipeline",
    "com.docker.compose.project": "docker-compose-files",
    "com.docker.compose.version": "1.27.4"
}
}
]

```

4.3 Data Model and Applications

To configure the data source and visualization dashboards please access Kibana homepage in the browser (<http://192.168.112.8:3000>)

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.

Data Sources / InfluxDB

Type: influxDB

W Settings

☐ Default

Query Language

InfluxQL

HTTP

Auth

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 10`

5 Setting up the Timescale 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/>

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

5.2 Instructions to run the pipeline in docker compose

Included is a reference docker compose configuration file.

timescale-docker-pipeline-reference-unenc.yml

Update config.ini

Update the config.ini file with following details

- IP and credentials of idracs