A Principled Technologies report: Hands-on testing. Real-world results.



Create useful data center health visualizations with Dell iDRAC Telemetry Reference toolset and Elastic Stack

Dell EMC[™] has recently released Telemetry Reference tools for the Integrated Dell Remote Access Controller (iDRAC). The Telemetry Reference toolset enables data center engineers to ingest and structure telemetry data streams for visualization with Elastic Stack software. This report serves as a how-to guide for setting up and configuring these components within your own environment.

At Principled Technologies, we set up the iDRAC Telemetry Reference toolset with a containerized Elastic Stack that comprised Elasticsearch and Kibana on a Dell EMC PowerEdge[™] R6515 with a 3rd Gen AMD EPYC[™] processor. We did not encounter any significant issues, and within a single business day, our data center technician was able to go from initial setup to creating Kibana visualizations using real telemetry data streams from multiple iDRAC sources. Additionally, to show the value of running this environment on the Dell EMC and AMD hardware, we stress-tested our telemetry environment with the Elasticsearch benchmarking tool Rally. Telemetry ingest remained uninterrupted during the stress test.

Get started quickly with iDRAC9 telemetry streaming and Elastic Stack

Using the new iDRAC Telemetry Reference toolset and an Elastic Stack container deployment, our team was able to query and visualize telemetry data starting from scratch in less than a day.

Continue to ingest telemetry data even during heavy server load

Telemetry ingest remained uninterrupted even while our environment ran a workload that stressed Elastic Stack and the system CPU.

Create useful data center health visualizations with Dell iDRAC Telemetry Reference tool and Elastic Stack

Table of contents

- Overview
- Installing and configuring Ubuntu
- Installing Docker
- Installing Docker-Compose
- Installing ActiveMQ
- Enabling the Telemetry service in iDRAC
- Setting up the iDRAC Telemetry Reference toolset
- Starting the Elastic Stack and iDRAC Telemetry Reference toolset

- Creating an index pattern in Kibana to query index
- Creating visualizations and dashboards in Kibana
- Running the Rally benchmark
- Custom reporting
- Appendix: Reference files
- Appendix: System configuration information



Dell EMC PowerEdge R6515 servers

According to Dell Technologies, these single-socket 1U servers offer the following high-level specifications:

- Up to 64 high performance 3rd Gen AMD cores
- PCIe Gen4 for high throughput connectivity
- Support for up to 10 SAS/SATA/NVMe drives
- Improved VM density and SQL performance

Learn more at https://www.dell.com/en-us/work/shop/povw/poweredge-r6515.

3rd Gen AMD EPYC processors

The latest offering from AMD, 3rd Gen EPYC processors offer increased I/O with "up to 32MB L3 cache per core," 7nm x86 technology, and new security features like Secure Encrypted Virtualization - Secure Nested Paging (SEV-SNP) and Encrypted State (SEV-ES). AMD positions the EPYC 75F3 model as being well suited for high frequency use cases such as VM density, virtualization, and VDI. For more information, visit https://www.amd.com/en/processors/epyc-7003-series.



Overview: iDRAC and Elastic Stack

In this how-to guide, we will describe the steps required to set up and use the new Dell iDRAC Telemetry tool with a containerized Elastic Stack deployment.

What is iDRAC9 Telemetry Streaming?

iDRAC9 v4.0 introduced the Telemetry Streaming feature, which allows system administrators to stream data from system hardware sensors and general health reporting to use in data analytics. (Note: Telemetry Streaming is available for Dell EMC PowerEdge systems with iDRAC Datacenter licenses.)

Collecting telemetry data via streaming is a newer and improved method compared to polling, a process that can be challenging to scale across a large data center. According to Dell EMC, the polling method is also prone to sampling errors and delays, whereas streaming is said to be more accurate and efficient.¹

Administrators can use telemetry data to predict hardware failures (thus enabling staff to proactively replace devices in danger of failing); to determine how best to improve system operations by analyzing data for key insights; and to tighten security measures, among other use cases. Because each telemetry stream in iDRAC9 contains more than 180 data points ranging from thermal data to system resource utilization, administrators will have many different ways in which to monitor their data centers. (For more information on iDRAC9 Telemetry Streaming, visit https://www. delltechnologies.com/resources/en-us/asset/whitepapers/products/software/direct-from-developmentdatacenter-telemetry.pdf.)

While iDRAC9 produces the telemetry report streams, it does not analyze or visualize data on its own. With the iDRAC Telemetry Reference toolset, however, administrators can use other components of Elastic Stack to perform those functions.

What is Elastic Stack?

Elastic Stack is a group of open-source software that enables users to format, search, analyze, and visualize realtime data from virtually any type of source. Elastic Stack consists of three components: Elasticsearch, Kibana, and an ingest tool such as Logstash.

Elasticsearch is a search and analytics engine that uses indices to store and retrieve data in JSON documents. This data can come in a variety of forms (structured, unstructured, textual, numerical, etc.), and can come from a number of sources, such as an application log or iDRAC telemetry data.

To ingest and enrich the data before it is indexed in Elasticsearch, Elastic Stack uses Logstash as their ingestion tool. In our guide, the iDRAC Telemetry Reference toolset takes care of that ingestion and enrichment step before the data is indexed.

The final piece in the Elastic Stack is Kibana, a free and open-source front end application that enables visualization and analysis of data stored in an elastic index or indices. With Kibana's built-in visualization tools, you can create dashboards that are a collection of graphs, maps, charts, and tables. We finish off our guide with instructions on how to query the data for meaningful reports and dashboards full of charts and graphs.

Deploying Elastic Stack

There are several ways to deploy Elastic Stack. Depending upon your environment, you may need to use a different method for installation and configuration. For this guide, we elected to deploy Elastic Stack and the iDRAC Telemetry Reference toolset in Docker containers using Docker-Compose to automate that process. We chose containers rather than VMs for a couple of reasons. First, in our testing, we had only a single server to work with. Running Elastic Stack and the iDRAC Telemetry Reference toolset in Containers also provide a more portable environment, allowing you to easily move your applications and data to another location if necessary. On their GitHub repository, Dell includes the Docker-Compose YAML files we used to deploy the Elastic Stack environment (https://github.com/dell/iDRAC-Telemetry-Reference-Tools/).

We start off with instructions for installing Ubuntu 20.04.1 LTS (focal) and the containerized Elastic Stack prerequisites Docker, Docker-Compose, and ActiveMQ on the Dell EMC PowerEdge R6515 server powered by a 3rd Gen AMD EPYC processor.

Installing and configuring Ubuntu 20.04.1 LTS

- 1. Boot to the installation media.
- 2. Select English.
- 3. If prompted, update to the new installer.
- 4. Select the keyboard layout, and click Done.
- 5. Configure the Network connections, and click Done.
- 6. To skip past the proxy address screen, click Done.
- 7. Leave the mirror as default, and click Done.
- 8. Configure the storage. For our testing, we used the entire disk.
- 9. Click Done.
- 10. Setup a username, hostname, and password. Click Done.
- 11. To install OpenSSH Server, check the box, and click Done.
- 12. Install any additional packages, and click Done.
- 13. When the installation is complete, select Reboot Now.
- 14. Log into the server.
- 15. Update and upgrade the OS:

sudo apt update sudo apt upgrade

16. Create a partition on the disks for Docker containers:

for i in {0..5};do sudo gdisk /dev/nvme\$in1; done

17. Create physical volumes:

for i in {0..5};do sudo pvcreate /dev/nvme\$in1; done

18. Create a virtual group:

sudo vgcreate data /dev/nvme0n1 /dev/nvme1n1 /dev/nvme2n1 /dev/nvme3n1 /dev/nvme4n1 /dev/nvme5n1

19. Create a logical volume:

sudo lvcreate -type raid10 -L 2T data

20. Create an EXT4 filesystem on the logical volume:

sudo mkfs.ext4 /dev/data/lvol0

21. Create a directory for Docker:

sudo mkdir /var/lib/docker

22. Mount the logical volume to the Docker directory:

sudo mount /dev/data/lvol0 /var/lib/docker

23. Edit /etc/fstab, and add the mounted volume for persistence across reboots.

24. Reboot.

25. Once you have installed and configured the OS, you will need to install Docker and Docker-Compose in order to deploy Elastic Stack and the iDRAC Telemetry Reference toolset within a containerized environment.

Installing Docker

1. Uninstall any Docker components that are already installed.

sudo apt-get remove docker docker-engine docker.io containerd runc

2. Install all prerequisites.

sudo apt-get install apt-transport-https ca-certificates curl gnupg-agent software-properties-common

3. Add the official Docker gpg key:

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -

4. Set up the stable repository:

```
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu \
$(lsb_release -cs) \
stable"
```

5. Update the apt index:

sudo apt-get update

6. Install Docker:

sudo apt-get install docker-ce docker-ce-cli containerd

7. To use Docker as a non-root user, add your user to the Docker group:

sudo usermod -aG docker <user>

Installing Docker-Compose

1. Download the latest release of Docker-Compose:

sudo curl -L "https://github.com/docker/compose/releases/download/1.28.0/docker-compose-\$(uname -s)\$(uname -m)" -o /usr/local/bin/docker-compose

2. Change the permissions of the binary to allow execution:

sudo chmod +x /usr/local/bin/docker-compose

The iDRAC Telemetry Reference toolset has two prerequisites: ActiveMQ and Go. Because we chose to deploy in a containerized environment, we needed only to install ActiveMQ. Note: If you are using multiple systems, you will need to install Go to build and start the required components.

Installing ActiveMQ

- 1. Download ActiveMQ from the following URL: https://activemq.apache.org/components/classic/download/apache-activemq-5.16.1-bin.tar.gz
- 2. Untar the package:

tar -xf apache-activemq-5.16.1-bin.tar.gz

Enabling the Telemetry service in iDRAC

There are several ways to enable the iDRAC telemetry streaming service, such as via iDRAC RACADM. To simplify this process, Dell has provided sample scripts on their GitHub repository. Below are the steps we used to enable telemetry reporting on several target iDRAC instances within our environment.

1. Clone the GitHub repository that contains the iDRAC Telemetry utilities:

```
wget https://github.com/dell/iDRAC-Telemetry-Scripting/archive/master.zip -0 iDRAC-
TelemetryScripting-master.zip
```

2. Unzip the file, and change directories:

```
unzip iDRAC-TelemetryScripting-master.zip
cd iDRAC-TelemetryScripting-master/
```

3. Enable telemetry reports and alerts on each iDRAC9 server:

```
python3 ./ConfigurationScripts/EnableOrDisableAllTelemetryReports.py -ip <IP ADDRESS> -u <USER> -P
<PASSWORD> -s Enabled
python3 ./ConfigurationScripts/SubscriptionManagementREDFISH.py -ip <IP ADDRESS> -u <USER> -p
<PASSWORD> -c y
```

Setting up the iDRAC Telemetry Reference toolset

1. Clone the GitHub repository that contains the iDRAC Telemetry Reference toolset:

git clone https://github.com/dell/iDRAC-Telemetry-Reference-Tools.git

2. Change directories into the iDRAC Telemetry Reference toolset:

cd iDRAC-Telemetry-Reference-Tools

3. In the config.ini file, add the iDRAC information for the systems you wish to monitor. We have included a reference in the Appendix.

Starting the Elastic Stack and iDRAC Telemetry Reference toolset

We have provided a reference Docker-Compose file in the Appendix to help guide you through the basic requirements for starting up the Elastic Stack. You may need to alter certain specifications such as volume location, networks, ports, or naming schemes based on your specific environment. Note: In our reference Docker-Compose file we do not have security enabled for the Elastic Stack. Elastic recommends securing your Elastic Stack after successfully deploying your initial cluster. See the following guide for more information: https://www.elastic.co/guide/en/elasticsearch/reference/current/configuring-security.html

- 1. Navigate to the location of your Docker-Compose file.
- 2. Start up the Elastic stack and iDRAC Telemetry Reference toolset:

docker-compose -f elk.yml up -d

3. After the cluster starts successfully, check to make sure all of your containers are still running:

| aocker | container is | | | | | |
|-----------------|--|------------------------|------------|---------------------|------------------------|------------------------|
| pt@elk:~\$ dock | ter container 1s | | | | | |
| CONTAINER ID | IMAGE | COMMAND | CREATED | STATUS | PORTS | NAMES |
| 8d0d78e50c20 | docker.elastic.co/logstash/logstash:7.10.1 | "/usr/local/bin/dock…" | 4 days ago | Up 4 days | 5044/tcp, 9600/tcp | logst01 |
| 8a01240be27b | docker.elastic.co/kibana/kibana:7.10.1 | "/usr/local/bin/dumb" | 4 days ago | Up 4 days | 0.0.0.0:5601->5601/tcp | kib01 |
| f32b032920a5 | golang:1.15 | "go run cmd/redfishr…" | 4 days ago | Up 4 days | | pt_redfishreadapp_1 |
| c8cbc41afac1 | golang:1.15 | "go run cmd/simpleau…" | 4 days ago | Up 4 days | | pt_authapp_1 |
| 9be0793f7696 | golang:1.15 | "go run cmd/elkpump/…" | 4 days ago | Up 4 days | | pt_elasticsearchpump_1 |
| 15728d9127b3 | golang:1.15 | "go run cmd/simpledi…" | 4 days ago | Up 4 days | | pt_discapp_1 |
| dd960ad1ab69 | docker.elastic.co/elasticsearch/elasticsearch:7.10.1 | "/tini /usr/local…" | 4 days ago | Up 4 days (healthy) | 9200/tcp, 9300/tcp | es01 |
| 51ea4f767856 | docker.elastic.co/elasticsearch/elasticsearch:7.10.1 | "/tini /usr/local…" | 4 days ago | Up 4 days | 9200/tcp, 9300/tcp | es02 |
| 46586fc1d9f4 | docker.elastic.co/elasticsearch/elasticsearch:7.10.1 | "/tini /usr/local…" | 4 days ago | Up 4 days | 9200/tcp, 9300/tcp | es03 |
| 83d94ddc4580 | rmohr/activemg:5.10.0 | "/bin/bash -c 'bin/a" | 4 davs ago | Up 4 davs | 8161/tcp, 61616/tcp | activemo |

Figure 1: Docker output. Source: Principled Technologies.

Creating an index pattern in Kibana to query index

In order to search an index, Kibana requires you to create an index pattern. Index patterns allow you to define properties on the fields from the selected data.

1. In a web browser, enter the address for your Kibana container:

<IP ADDRESS>:5601

- 2. In the top-left, to expand the options menu, click the hamburger symbol.
- 3. Scroll down to the Management section, and click Stack Management.
- 4. On the left, under Data, click Index Management. You should see an index named poweredge_telemetry_metrics with green health, the status set to open, and multiple documents.

| Stack Management / Index Managem | nent | | | | | |
|--|---|---|-----------------------------------|-----------------------------------|--|-----------|
| Ingest © Ingest Node Pipelines Data © Index Management | Index Managemen | poweredge_telemetry_metrics summary Settings Mappings Stats Edit settings one General General General General | | | | |
| Index Lifecycle Policies Snapshot and Restore Rollup Jobs Transforms Remote Clusters | Update your Elasticsearch indices indiv Q Search Name | idually or in bulk. Le Health | Health Primaries Docs Count | • green 1 15954493 2.8nb | Status Replicas Docs Deleted Primary Storage Size | open 1 |
| Alerts and Insights ① Alerts and Actions Reporting | poweredge_telemetry_metrics geonames | • green | Aliases | none | Filmary otorage orac | |
| Kibana ① Index Patterns Saved Objects | Rows per page: 10 V | green | 90 | | | |
| Spaces Advanced Settings | | | | | | |
| Stack © License Management 8.0 Upgrade Assistant | | | | | | |
| | | | | | | ∧ Manage |

Figure 2: Index Management within Kibana. Source: Principled Technologies.

5. After verifying that the index exists, you will need to create an index pattern with which to query the index for information for visualizations. To do this, click Index Patterns under Kibana.



- 6. Click Create index pattern.
- 7. Type the name of the index into the index pattern name followed by an asterisk (*) to match all indices with that name. (An asterisk is not necessary if you are only using a single index.)

| E D Stack Mar | gement / Index patterns / Create Index pattern | | | | | | | |
|---|--|---|--|--|--|--|--|--|
| Ingest ① Ingest Node Pipelines | Create index pattern | | | | | | | |
| Data () Index Management | An index pattern can match a single source, for example, filebeat-4-3-22, or multiple data sources, filebeat-*. Read documentation ⊘ | An index pattern can match a single source, for example, filebeat-4-3-22 , or multiple data sources, filebeat-+ . Read documentation (5 | | | | | | |
| Index Lifecycle Policies Snapshot and Restore Rollup Jobs | Step 1 of 2: Define an index pattern | | | | | | | |
| Remote Clusters | powersdige_telemetry_matrics! Use an asteriat (*) to match multiple indices. Spaces and the characters _/, ?, *, >, are not allowed. | Next step > | | | | | | |
| Alerts and Insights @ Alerts and Actions Reporting | Include system and hidden indices | | | | | | | |
| Kibana () | tour index patient marches i source. poweredge_telemetry_metrics mdex | | | | | | | |
| Saved Objects Spaces Advanced Settings | Rows per page: 10 $ \lor $ | | | | | | | |
| Stack @ | | | | | | | | |
| License Management 8.0 Upgrade Assistant | | | | | | | | |

Figure 3: Index pattern definition within Kibana. Source: Principled Technologies.

- 8. Click Next step.
- 9. Under Time field, from the drop-down menu, select Timestamp. Click Create index pattern.

| Stack Management / Index patterns | : / Create index pattern | | | | | | | |
|--|---|---------------------------|--|--|--|--|--|--|
| Ingest ① Ingest Node Pipelines | Create index pattern An index pattern can match a single source, for example, filebeat-4-3-22, or multiple data sources, filebeat-+. | | | | | | | |
| Data (2) Index Management Index Lifecycle Policies | An induce partient Can match a single source, for example, interest + ->-22, of multiple data sources, fileBeat + Read documentation (s | | | | | | | |
| Snapshot and Kestore Rollup Jobs Transforms Remote Clusters | Step 2 of 2: Configure settings Specify settings for your poweredge_telemetry_metrics index pattern. Select a primary time field for use with the global time filter. | | | | | | | |
| Alerts and Insights ① Alerts and Actions Reporting | Time field Befresh Timestamp V | | | | | | | |
| Kibana ① Index Patterns Saved Objects Spaces Advanced Settings | > Show advanced settings | Back Create index pattern | | | | | | |
| Stack ⑦ License Management 8.0 Upgrade Assistant | | | | | | | | |

Figure 4: Index pattern configuration within Kibana. Source: Principled Technologies.

10. Expand the left hand menu. Under the Kibana section, click Discover.

11. Expand the time range to 1 hour.



Figure 5: Kibana data discovery page with no filters applied. Source: Principled Technologies.

- 12. Verify that the data displayed include a timestamp and data from _source.
- 13. Click the plus signs next to the following fields to add them:
 - a. Context
 - b. ID
 - c. System
 - d. ValueAggregatable

Now that your data is in a more readable format, you can use the search bar and Kibana Query Language to query the index for specific results. Below is an example query you could use to see the RPM of a particular fan in your server:

System.keyword:"<IDRAC SERVICE TAG>" and Context.keyword:"System Board FanlA"



Figure 6: Kibana data discovery page with field filters. Source: Principled Technologies.

Create useful data center health visualizations with Dell iDRAC Telemetry Reference tool and Elastic Stack

| | D Discover | | | | |
|----------------------------------|---|--|--|--|--|
| | System.keyword:"FL3JFX2" and Context.keyword:"System Board Fan1A" | | | | |
| ; − + | System.keyword:"FL3JFX2" and Context.keyword:"System Board Fan1A" | | | | |
| poweredge_telemetry_metrics* 🤟 😑 | | | | | |

Figure 7: A basic query within Kibana. Source: Principled Technologies.

| Discover | | | | | | | | | | | | | New S | ave Open | Share Inspect |
|---|-----------|--------|------------------------------------|------------|------------------|--------------|-------------|--------------------|--------------------------|-----------------|-------------|---------|------------|------------|---------------|
| System.keyword:"FL3JF) | <2" and C | ontext | .keyword:" | System Boa | rd Fan1A* | | | | к | QL 😁 🗸 | Last 1 hour | | | Show dates | C Refresh |
| 🗇 🛛 + Add filter | | | | | | | | | | | | | | | |
| poweredge_telemetry_metric | :s* ∽ 1 | = | | | | | | | 1 hit | | | | | | |
| Q Search field names | | | | | | | Mar 8, 2021 | @ 09:34:50.763 | - Mar 8, 2021 @ 10 | 0:34:50.763 | Auto \sim | | | | |
| Filter by type Selected fields Context ID System ValueAggregatable Available fields | 0 | Count | 1 0.8 0.6 0.4 0.2 0 | 09:35 | 09-40 | 09:45 | 09-50 | 00:55 | 10.00 Timestamp per n | 10:05 sinute | 10-10 | 10:15 | 10:20 | 10:25 | 10:30 |
| Popular r id | | | Time 🗸 | | ID | | Context | | | System | | ValueAg | gregatable | | |
| r _index | | > | > Mar 8, 2021 0 09:36:44.000 | | :44.000 RPMReadi | 8 RPMReading | | System Board Fan1A | | FL3JFX2 | | 5,768 | | | |
| t _type | | | | | | | | | | | | | | | |
| r Label | | | | | | | | | | | | | | | |
| .score | | | | | | | | | | | | | | | |
| Timestamp | | | | | | | | | | | | | | | |

Figure 8: The resulting visualization based on the query in Figure 7. Source: Principled Technologies.

To save a query, click the Save drop-down menu beside the search bar and click Save current query. Give the query a name and description, if necessary, and click Save.

You can also use filters to further carve up the search results. To do so, click the Add filter button under the search bar. Select a field and an operator. If a value is required, enter the value, and click Save.

Creating visualizations and dashboards with Kibana

Having the capability to ingest and index data is great, but it's more useful to perform analysis on data and create visualizations from it. With Kibana, you can create visualizations specific to key areas you wish to monitor and group these with dashboards. This not only reduces the time required to evaluate the status of your hardware or software; it also provides you with insight into potential problem areas.

1. In the left-hand pane, under the Kibana section, select Dashboard.



Figure 9: The Kibana navigation pane. Source: Principled Technologies.

- 2. Click Create dashboard.
- 3. To create a new visualization for the dashboard, click Create new.
- 4. Select the type of visualization you want. If you are new to Kibana, the Lens visualization type will be the easiest to start with.

Covering the total number of options available for creating visualizations in Kibana is outside the scope of this document. You should experiment with all of the different query and filter options as well as the different visualization types to create the most appropriate visualizations for your purposes. Below is an example of how to create a basic visualization of the average RPM of a single fan with Lens.

- 1. Drag and drop the ValueAggregatable field from the left-hand side to the middle section to begin. The X-axis should automatically populate with Timestamp, and the Y-axis should be Average of ValueAggregatable.
- 2. On the right, click the value under Y-axis.
- 3. Change the display name to Average Fan RPM, and click Close.
- 4. To filter for a specific fan, we need to add a field under Break down by section on the right side of the screen. Click Drop a field or click to add.
- 5. Select Filters.
- 6. Click the field that says "= All records" to query the index.
- 7. We will use the same query that we saved from earlier to get the average fan speed from the index for Fan 1A.

System.keyword:"<IDRAC SERVICE TAG>" and Context.keyword:"System Board FanlA"

- 8. Give the query a label. For example, Fan 1A
- 9. To add another fan to the visualization, add another filter, and repeat the query, replacing 1A with 1B.
- 10. To change the chart type, click the drop-down menu, or select one of the suggestions that Kibana provides below the chart.
- 11. When you are finished with the visualization, to save the visualization, click Save.



Figure 10: Visualization of thermal telemetry data within Kibana. Source: Principled Technologies.

- 12. Give the visualization a title and description, select Add to Dashboard after saving, and click Save and return.
- 13. Repeat the visualization creation process as many times as necessary. To save your dashboard, in the top-right, click Save.
- 14. Give the dashboard a title and description, and click Save.



Figure 11: An example of a Kibana dashboard using power usage telemetry data. Source: Principled Technologies.

Running the Rally benchmark

In order to demonstrate the capability of the iDRAC Telemetry Reference toolset to continue functioning under load with the support of the Dell EMC PowerEdge R6515 server powered by a 3rd Gen AMD EPYC processor, we produced a heavy load on the Elastic Stack and CPU by running the Rally benchmark in containers with the same track running in two terminals simultaneously. While the benchmark was running, CPU utilization was at or above 90 percent, and the iDRAC Telemetry Reference toolset continued to push data to the Elastic Stack without interruption. We used the following steps to run Rally:

- 1. On the server under test, open a terminal window.
- 2. Create two directories, one for each Rally track:

```
mkdir rally1
mkdir rally2
```

3. To add the Rally paths to your environment, run the following commands:

echo "RALLY1=[Path to rally1]" > sudo tee -a .profile echo "RALLY2=[Path to rally2]" > sudo tee -a .profile

4. To set the new environment variables, source the profile:

source .profile

- 5. Open a second terminal window.
- 6. In the first terminal, type the following (but don't start the test yet):

```
docker run --network=pt_elastic --rm -v $RALLY:/rally/.rally elastic/rally --track=nyc_taxis
--pipeline=benchmark-only --target-hosts=es01:9200
```

7. In the second terminal, type the following (but don't start the test yet):

docker run --network=pt_elastic --rm -v \$RALLY:/rally/.rally elastic/rally --track=nyc_taxis
--pipeline=benchmark-only --target-hosts=es01:9200

8. Once you have both commands queued and ready to go, hit the return key in each terminal to start the Rally benchmarks.



Figure 12: Elastic Stack CPU dashboard during the Rally benchmark test. Source: Principled Technologies.

| IndERCOM sarges_cold_line is 767256 ms.indicating that the cluster is not: Nonexampl sarges_cold_line(ist)_cold_main(ist)_fiber is the cluster ist of the cluster is 15655 ms.indicating that the cluster ist of the cluster is 15655 ms.indicating that the cluster ist of the cluster is 15655 ms.indicating that the cluster is not manning create index Renning create index | <pre>s defined tion state. Records index time stricts are be milleding. in an its a defined time state. Records index time stricts and be milleding. in a defined time trate. Records index time stricts are be milleding. Sectors time trate. Records index time stricts are be milleding. (Inter state) (Inter state) (In</pre> |
|--|---|
| | |

Figure 13: Rally benchmarking tool. Source: Principled Technologies.



Figure 14: Rally benchmarking tool. Source: Principled Technologies.

| top - 19:26:09 Tasks: 526 tot: | up 13 day al, <u>1</u> ru | s, 1:25, nning, 5 | 3 USE 3 sleep | ing, 🌖 | stoppe | ge: 28 d, 2 | De, 23.2.1.4.2 zensis |
|------------------------------------|------------------------------|----------------------|------------------|--------------------|--------------|-----------------|--|
| MCpu(s): 84.0 (MiB Mem : 25762 | 9.9 total | 6.0 53867. | 13.8 • free, | 11736.8 | wa, used, | 0.0 hi 19202 | 6.1 si, 6.8 st 1. buffocche |
| MiB Swap: 819 | 2.0 total | 8198 | | 1.0 | used | 24410 | |
| PID USER | PR NI | VIRT | RES | SHR S | SICPU | 50HEM | TIME-COMMO |
| 2669 pt | 28 8 | 32.00 | 4.80 | 2.60 5 | 1430 | 1.9 | irotiz java Belo.21 java |
| 2482 pt 773978 pt | 20 0 | 21-1g 74.5g | 2.60 | 569328 S 1.80 S | 126.9 | 1.0 | 21:134.69 java 0:13.24 exaalu |
| 774185 pt | 20 0 | 74.50 | 1.70 | 1.6g S | 7.3 | 0.7 | 0:12.76 estally |
| 774183 pt | 20 0 | 74.50 | 1.79 | 1.69 5 | 3.0 | 0.7 | 0:33-6 erally |
| 774182 pt 774187 pt | 20 0 | 74.5g 74.5g | 1.70 | 1.6g \$ 1.6g \$ | 2.7 | 0.7 | 0:12:46 estally 0:12:69 estally |
| 774181 pt | 20 0 | 74.50 | 1.80 | 1.70 S | 2.0 | | 4:15.89 estally |
| 773968 pt | 20 0 | 74.50 | 1.70 | 1.7g S | 1.3 | 0.7 | es:22.27 escally |
| 773967 pt 773969 pt | 28 0 | 74.5g 74.5g | 1.90 | 1.9g S 1.7g S | 0.7 | 0.8 0.7 | 9:15.41 escally 9:13.16 escally |
| 773971 pt | 20 0 | 74.50 | 1.70 | 1.60 5 | 0.7 | 0.7 | atta 46 estally |
| 774184 pt | 28 8 | 74.50 | 1.70 | 1.70 \$ | 0.7 | 0.7 | 5.3.2.6 exally |
| 12 root 1217 root | 20 0 | | | 0 I 0 S | 0.3 0.3 | 0.0 | 1:21.98 rcu_schod 5:69.08 104/2/mm8 |
| 3166 root | 20 0 | 111780 | 8856 | 7040 5 | 0.3 | 0.0 | 1:32.46 containerd-shin |
| 5572 pt | 20 0 | 1389728 | 287584 | 38628 \$ | 0.3 | 0.1 | Sile.27 node |
| 764993 pt 773966 pt | 20 0 | 9872 | 4616 | 3276 R | 0.3 | 0.8 | 9:13:48 top 9:15:0:36 errally |
| 773973 pt | 20 0 | 74.50 | 1.70 | 1.70 \$ | 0.3 | 0.7 | 9:12.54 escally 8:14.27 escally |
| 1 root | 20 0 | 169320 | 11948 | 7024 S | 0.0 | e.e | 9:34-37 0143119 1:54-87 9x9temd |
| 2 root 3 root | 20 0 | | 0 | 0 S 0 I | 0.0 | 0.0 | 9:91.29 kthreadd eite cu an either a |
| 4 root | 0 -20 | | | 0 1 | 0.0 | 0.0 | 9189.88 rcu_par_go |
| 10 root | 0 -20 | | | 0 1 | 0.0 | 0.0 | olet-de molter/olet-subclus |
| 11 root 13 root | 20 0 rt 0 | | 0 | 0 S 0 S | 0.0 | 0.0 | 0:00.45 ksoftirg0/0 0:01.63 migration/0 |
| 14 root | -51 0 | | | 8 5 | 0.0 | 0.0 | 9:09.09 / 10 / 10 / 10 / 10 / 10 / 10 / 10 / |
| 16 root | | | | e s | 0.0 | 0.0 | 0-00-00 (postp/0 0-00-00 (postp/0 |
| 17 root 18 root | -51 0 rt 0 | | | 8 S | 0.0 | 0.0 | 0100.00 1dle_inject/1 000000000000000000000000000000000000 |
| 19 root 21 root | 20 0 | | | 0 S | 0.0 | 0.0 | 9:00.88 ksftird/1. 0:00.80 ksftird/1. |
| 23 root | 20 0 | | | 8 S | 0.0 | 0.0 | 9199-09 cpuhp/2 |
| 24 root 25 root | -51 0 rt 0 | | | 8 S 8 S | 0.0 | 0.0 | elee.48 jola_inject/2 ele2.27 angipration/2 |
| 26 root 28 root | 20 0 | | | 0 5 | 0.0 | 0.0 | 4:00.23 ks0fird/2 6:00.0 ks0fird/2 |
| 30 root | 20 0 | | | e s | 0.0 | 0.0 | 0:80.00 cpuhp/3 |
| 32 root | rt 0 | | | 0 5 | 0.0 | 0.0 | orea to a company and the second se |
| 33 root 35 root | 20 0 | | | 0 5 | 0.0 | 0.0 | 9:00.24 ks9fligd/3 9:00.08 ks9fligd/3 |
| 37 root | 20 0 | | | e s | 0.0 | 0.0 | 0:00,00 cpuhp/A |
| 39 root | rt e | | | 0 5 | 0.0 | 0.0 | orea constantiant/ |
| 40 root 42 root | 20 0 | | | 8 S 8 I | 8.8 | 0.0 | 9:90-33 ks9fligd/4 9:80-08 ks9fligd/4 |
| 43 root | 20 0 | | | 0.5 | 0.0 | 9.9 | 0:109.00 cpuhp/6 |
| 45 root | rt e | | | 8 5 | 0.0 | 0.0 | sterve for interview of the state of the sta |
| 48 root | 20 0 | | | 8 S 8 I | 0.0 | 0.0 | elee.as ks9ftlr0075 elee.as ks9ftlr0075 |
| 49 root | 20 0 | | | 0 5 | 0.0 | 0.0 | attéle de coube/6 attéle de trainéri/A |
| 51 root | rt e | | | e s | 0.0 | 0.0 | 0:02.08 migration/6 |
| 52 root 54 root | 20 0 | | | 0 S 0 I | 0.0 | 0.0 | 0100_10 K50TLIG075 0100_08 kworker/6100-kblockd |
| 65 root | 28 0 | | | e s | 0.0 | 0.0 | 9:69.00 cpubp/7 |
| | | | | | | | |

Figure 15: Top task manager output during the Rally benchmark test. Source: Principled Technologies.

Custom reporting

Dell has added the ability to create custom report definitions for iDRAC9 telemetry data. With this new capability, data center technicians can use the Redfish API to create, edit, and delete custom reports. These custom reports allow users to set different metric values, collection intervals, and collection functions that are unique to their organization's interests or concerns. For example, your technician could create a report that collects relevant thermal metrics via temperature and fan speed readings. We were able to successfully upload a custom report following the steps below:

1. Create a JSON document with the relevant fields as in the example below.

```
{
    "Id": "TemperatureAndRPMReading",
    "Name": "All Temperature and RPM Readings",
    "Description": "A report with all relevant
thermal metrics - Temperature reading and FAN
RPM readings",
    "MetricReportDefinitionEnabled": true,
    "MetricReportDefinitionType": "Periodic",
    "MetricReportHeartbeatInterval":
"PTOHOMOS",
    "SuppressRepeatedMetricValue": false,
    "ReportTimespan": "PTOH1MOS",
    "ReportUpdates": "Overwrite",
    "Schedule": {
        "RecurrenceInterval": "PTOH1MOS"
    },
    "Metrics": [
            {"MetricId": "TemperatureReading"},
            {"MetricId": "RPMReading"}
    ],
    "Metrics@odata.count": 2
```

2. To create the custom report, run the following command:

```
curl -s -k -u <username>:<password>
-X POST https://<iDRACIP>/redfish/v1/
TelemetryService/MetricReportDefinitions -H
'Content-Type: application/json' -d @<custom
file path>
```

Note: At the time of writing, the Dell iDRAC Telemetry Reference toolset does not support ingestion of custom reports. However, the Dell team has told us that support for custom reporting is on the development roadmap for the tool.





Appendix: Reference files

iDRAC configuration file

Below are the full contents of the config.ini file we used when setting up the iDRAC Telemetry Reference toolset.

config.ini

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

```
[Services]
Types=iDRAC,iDRAC,iDRAC,iDRAC,iDRAC
IPs=IDRAC_IP01,IDRAC_IP02,IDRAC_IP03,IDRAC_IP04,IDRAC_IP05
```

[IDRAC_IP01] username=<USERNAME> password=<PASSWORD>

[IDRAC_IP02] username=<USERNAME> password=<PASSWORD>

[IDRAC_IP03] username=<USERNAME> password=<PASSWORD>

[IDRAC_IP04] username=<USERNAME> password=<PASSWORD>

[IDRAC_IP05] username=<USERNAME> password=<PASSWORD>

Docker-Compose reference file

Below are the contents of the file we used when setting up Docker-Compose.

```
elk.yml
   version: '3.5'
   services:
     es01:
       image: docker.elastic.co/elasticsearch/elasticsearch:7.10.1
       container name: es01
       environment:
         - node.name=es01
         - cluster.name=es-docker-cluster
         - discovery.seed hosts=es02,es03
         - cluster.initial master nodes=es01,es02,es03
         - bootstrap.memory lock=true
       ulimits:
         memlock:
           soft: -1
           hard: -1
       volumes:
         - <PATH TO ES01 DATA>:/usr/share/elasticsearch/data
       networks:
         - elastic
       healthcheck:
         test: curl http://localhost:9200 >/dev/null; if [[ $$? == 52 ]]; then echo 0; else echo 1; fi
         interval: 30s
         timeout: 10s
         retries: 5
     es02:
       image: docker.elastic.co/elasticsearch/elasticsearch:7.10.1
       container name: es02
       environment:
         - node.name=es02
          - cluster.name=es-docker-cluster
         - discovery.seed hosts=es01,es03
         - cluster.initial master nodes=es01,es02,es03
         - bootstrap.memory_lock=true
       ulimits:
         memlock:
           soft: -1
           hard: -1
       volumes:
         - <PATH TO ES02 DATA>:/usr/share/elasticsearch/data
       networks:
         - elastic
     es03:
       image: docker.elastic.co/elasticsearch/elasticsearch:7.10.1
       container name: es03
       environment:
         - node.name=es03
          - cluster.name=es-docker-cluster
          - discovery.seed hosts=es01,es02
         - cluster.initial master nodes=es01,es02,es03
          - bootstrap.memory_lock=true
       ulimits:
         memlock:
           soft: -1
           hard: -1
```

```
volumes:
   - <PATH TO ES03 DATA>:/usr/share/elasticsearch/data
 networks:
   - elastic
kib01:
 image: docker.elastic.co/kibana/kibana:7.10.1
 container name: kib01
 depends_on: {"es01": {"condition": "service_healthy"}}
 environment:
   ElasticSEARCH URL: http://es01:9200
   ElasticSEARCH HOSTS: http://es01:9200
 ports:
   - 5601:5601
 networks:
   - elastic
logst01:
 image: docker.elastic.co/logstash/logstash:7.10.1
 container name: logst01
 depends on:
   - kib01
 environment:
   http.host: 0.0.0.0
   monitoring.elasticsearch.hosts: http://es01:9200
   xpack.monitoring.collection.enabled: "true"
 networks:
   - elastic
activemq:
 image: rmohr/activemq:5.10.0
 container name: activemq
 hostname: activemq
 volumes:
   - <PATH TO ACTIVEMQ DIRECTORY>/conf/activemq.xml:/opt/activemq/conf/activemq.xml
 working_dir:
   /opt/activemq/
 networks:
    - elastic
discapp:
 image: golang:1.15
 volumes:
   - <PATH TO iDRAC-Telemetry-Reference-Tools>:/go/src/github.com/telemetryservice
 working dir:
   /go/src/github.com/telemetryservice
 command:
   go run cmd/simpledisc/simpledisc.go
 networks:
   - elastic
 depends on:
   - "activemq"
authapp:
 image: golang:1.15
 volumes:
   - <PATH TO iDRAC-Telemetry-Reference-Tools>:/go/src/github.com/telemetryservice
 working dir:
   /go/src/github.com/telemetryservice
 command:
   go run cmd/simpleauth/simpleauth.go
 networks:
   - elastic
 depends on:
   - "activemq"
```

```
redfishreadapp:
 image: golang:1.15
  volumes:
   - <PATH TO iDRAC-Telemetry-Reference-Tools>:/go/src/github.com/telemetryservice
 working dir:
   /go/src/github.com/telemetryservice
 command:
   go run cmd/redfishread/redfishread.go
  networks:
   - elastic
  depends on:
   - "activemq"
elasticsearchpump:
  image: golang:1.15
  volumes:
   - <PATH TO iDRAC-Telemetry-Reference-Tools>:/go/src/github.com/telemetryservice
 working dir:
   /go/src/github.com/telemetryservice
 command:
   go run cmd/Elasticpump/Elasticpump-basic.go
 networks:
   - elastic
  depends on: {"es01": {"condition": "service healthy"}}
  depends on:
   - "activemq"
  environment:
   ElasticSEARCH_URL: http://es01:9200
```

networks: elastic: driver: bridge



Appendix: System configuration information

Table 1: Detailed information on the system we tested.

| System configuration information | Dell EMC PowerEdge R6515 | | | | | |
|--|----------------------------|--|--|--|--|--|
| BIOS name and version | Dell 2.0.3 | | | | | |
| Non-default BIOS settings | N/A | | | | | |
| Operating system name and version/build number | Ubuntu 20.04.1 LTS | | | | | |
| Date of last OS updates/patches applied | 02/27/2021 | | | | | |
| Power management policy | System Profile=Performance | | | | | |
| Processor | | | | | | |
| Number of processors | 1 | | | | | |
| Vendor and model | AMD EPYC 7313 | | | | | |
| Core count (per processor) | 16 | | | | | |
| Core frequency (GHz) | 3.0 | | | | | |
| Stepping | Model 1 Stepping 1 | | | | | |
| Memory module(s) | | | | | | |
| Total memory in system (GB) | 256 | | | | | |
| Number of memory modules | 16 | | | | | |
| Vendor and model | Hynix HMA82GR7CJR8N-XN | | | | | |
| Size (GB) | 16 | | | | | |
| Туре | DDR-4 | | | | | |
| Speed (MHz) | 2,933 | | | | | |
| Speed running in the server (MHz) | 2,933 | | | | | |
| Storage controller | | | | | | |
| Vendor and model | PERC H730p Mini | | | | | |
| Cache size (GB) | 2 GB | | | | | |
| Firmware version | 25.5.7.0005 | | | | | |
| Local storage 1 | | | | | | |
| Number of drives | 1 | | | | | |
| Drive vendor and model | Micron MTFDDAK240TCB | | | | | |
| Drive size (GB) | 240 | | | | | |
| Drive information (speed, interface, type) | 6Gbps, SATA, SSD | | | | | |

| System configuration information | Dell EMC PowerEdge R6515 | | | | | | |
|--|----------------------------|--|--|--|--|--|--|
| Local storage 2 | | | | | | | |
| Number of drives | 6 | | | | | | |
| Drive vendor and model | Dell Express Flash PM1725b | | | | | | |
| Drive size (GB) | 1,600 | | | | | | |
| Drive information (interface and type) | NVMe SSD | | | | | | |
| Network adapter | | | | | | | |
| Vendor and model | Broadcom BCM5720 | | | | | | |
| Number and type of ports | 2 x 1Gb | | | | | | |
| Driver version | 21.60.8 | | | | | | |
| Cooling fans | | | | | | | |
| Vendor and model | Dell 16W7D-A01 | | | | | | |
| Number of cooling fans | 6 | | | | | | |
| Power supplies | | | | | | | |
| Vendor and model | Dell 06V43GA00 | | | | | | |
| Number of power supplies | 2 | | | | | | |
| Wattage of each (W) | 550 | | | | | | |

1 Dell EMC, "Transforming Datacenter Analytics with iDRAC9 Telemetry Streaming," accessed March 15, 2021, https://www.delltechnologies.com/resources/en-us/asset/white-papers/products/software/direct-from-developmentdatacenter-telemetry.pdf.

This project was commissioned by Dell Technologies.





Principled Technologies is a registered trademark of Principled Technologies, Inc. All other product names are the trademarks of their respective owners.

DISCLAIMER OF WARRANTIES; LIMITATION OF LIABILITY:

Principled Technologies, Inc. has made reasonable efforts to ensure the accuracy and validity of its testing, however, Principled Technologies, Inc. specifically disclaims any warranty, expressed or implied, relating to the test results and analysis, their accuracy, completeness or quality, including any implied warranty of fitness for any particular purpose. All persons or entities relying on the results of any testing do so at their own risk, and agree that Principled Technologies, Inc., its employees and its subcontractors shall have no liability whatsoever from any claim of loss or damage on account of any alleged error or defect in any testing procedure or result.

In no event shall Principled Technologies, Inc. be liable for indirect, special, incidental, or consequential damages in connection with its testing, even if advised of the possibility of such damages. In no event shall Principled Technologies, Inc.'s liability, including for direct damages, exceed the amounts paid in connection with Principled Technologies, Inc.'s testing. Customer's sole and exclusive remedies are as set forth herein.