Instructions for setting up InfluxDB and Telegraf with Docker compose

VM

Go to Google cloud compute engine, launch a Linux VM(I have used Debian Linux), give it a name(influxdb) and also add a network tag influxdb to it.

Access the VM over SSH, to access it from the gcloud shell, a command like the one below may be used.

gcloud compute ssh --zone "asia-south1-c" "influxdb" --project <project-id>

Docker

Install [docker](https://docs.docker.com/engine/install/debian/) as we would need it for pulling the telgraf image.

sudo apt-get update -y

sudo apt-get install \

ca-certificates \

curl \

gnupg \

lsb-release -y

sudo mkdir -p /etc/apt/keyrings

curl -fsSL https://download.docker.com/linux/debian/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg

echo \

"deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/debian \

$(lsb\_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

sudo apt-get update -y

sudo apt-get install docker-ce docker-ce-cli containerd.io docker-compose-plugin -y

Also add the current user to the docker group.

sudo usermod -aG docker $USER

Exit and login again for the user group change to take effect. You may validate the list of users in the docker with the following command.

getent group docker

You may check [docker](https://hub.docker.com/_/telegraf?tab=tags) hub or influxdata [downloads](https://portal.influxdata.com/downloads/) portal for other options.

Directories

Create three directories, which would attached to the docker containers as volumes.

sudo mkdir /etc/influxdb2

sudo mkdir /var/lib/influxdb2

sudo mkdir /etc/telegraf

Config

Generate a sample telegraf config with cpu and memory input plugins, influxdb\_v2, and save it locally, to further modify it.

sudo sh -c 'docker run --rm telegraf:1.23.2-alpine --input-filter mem --output-filter influxdb\_v2 config > /etc/telegraf/telegraf.conf'

Compose

Create a docker compose file.

$ cat docker-compose.yml

networks:

influxdb: {}

services:

influxdb:

image: influxdb:2.3.0-alpine

networks:

- influxdb

ports:

- "8086:8086"

volumes:

- /etc/influxdb2:/etc/influxdb2

- /var/lib/influxdb2:/var/lib/influxdb2

telegraf:

environment:

- INFLUX\_TOKEN=${INFLUX\_TOKEN}

depends\_on:

- influxdb

image: telegraf:1.23.2-alpine

networks:

- influxdb

ports:

- "1619:1619"

volumes:

- /etc/telegraf:/etc/telegraf:ro

And run it.

$ docker compose up -d

[+] Running 2/2

⠿ Container influxdb-influxdb-1 Started 0.5s

⠿ Container influxdb-telegraf-1 Started 1.1s

Firewall rule

In the google cloud console, Go to VPC Network > Firewall, and create new firewall rule with name influxdb, add influxdb to target tags, and set the source as your public IP. Allow TCP ports 1619, 8086 and save the rule.

The port 1619 is for the webhook listener with in the telegraf container and 8086 is for the http port of influxdb.

Containers

We should have two running containers now.

$ docker container ls

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

32a1b0638bd2 telegraf:1.23.2-alpine "/entrypoint.sh tele…" 20 minutes ago Up 20 minutes 8092/udp, 0.0.0.0:1619->1619/tcp, :::1619->1619/tcp, 8125/udp, 8094/tcp influxdb-telegraf-1

e495f35994c2 influxdb:2.3.0-alpine "/entrypoint.sh infl…" 20 minutes ago Up 20 minutes 0.0.0.0:8086->8086/tcp, :::8086->8086/tcp influxdb-influxdb-1

Influxdb setup

Set the influxdb credentials.

bash-5.1# influx setup

> Welcome to InfluxDB 2.0!

? Please type your primary username influxdb

? Please type your password \*\*\*\*\*\*\*\*\*\*\*\*

? Please type your password again \*\*\*\*\*\*\*\*\*\*\*\*

? Please type your primary organization name influxdb

? Please type your primary bucket name influxdb

? Please type your retention period in hours, or 0 for infinite 0

? Setup with these parameters?

Username: influxdb

Organization: influxdb

Bucket: influxdb

Retention Period: infinite

Yes

User Organization Bucket

influxdb influxdb influxdb

bash-5.1# exit

API token

Get the public IP of the Linux VM.

$ curl ifconfig.co

<public-ip>

And access the InfluxDB console on the browser at http://<public-ip>:8086.

Login with the credentials, goto Load Data > API Tokens. Either create a new token or use the initial token that was generated during the setup. Set this token on the docker .env file.

$ cat .env

INFLUX\_TOKEN=<api-token>

Telegraf config

Modify certain entries in the influxdb\_v2 plugin config in telegraf.

$ sudo cat /etc/telegraf/telegraf.conf | grep -A 20 influxdb\_v2 | grep -v '#'

[[outputs.influxdb\_v2]]

urls = [ "http://influxdb-influxdb-1:8086" ]

token = "$INFLUX\_TOKEN"

organization = "influxdb"

bucket = "influxdb"

and restart telegraf

$ docker compose restart telegraf

[+] Running 1/1

⠿ Container influxdb-telegraf-1 Started 0.7s

Logs

Check if telegraf logs are fine.

$ docker container logs influxdb-telegraf-1 -f

2022-07-20T06:50:23Z I! Using config file: /etc/telegraf/telegraf.conf

2022-07-20T06:50:23Z I! Starting Telegraf 1.23.2

2022-07-20T06:50:23Z I! Loaded inputs: cpu mem

2022-07-20T06:50:23Z I! Loaded aggregators:

2022-07-20T06:50:23Z I! Loaded processors:

2022-07-20T06:50:23Z I! Loaded outputs: influxdb\_v2

2022-07-20T06:50:23Z I! Tags enabled: host=4c42a912c8d3

2022-07-20T06:50:23Z I! [agent] Config: Interval:10s, Quiet:false, Hostname:"4c42a912c8d3", Flush Interval:10s

^C

Graph

Go to Data Explorer on the InfluxDB console, create a sample graph by selecting the bucket name influxdb in the FROM box, mem in the filter and usage in the next filter. Submit and you should a graph showing the telegraf container's memory utilization.

It's also approximately matching with the docker stats.

$ docker stats --no-stream influxdb-telegraf-1

CONTAINER ID   NAME                  CPU %     MEM USAGE / LIMIT     MEM %     NET I/O           BLOCK I/O    PIDS

7020b6f13ca9   influxdb-telegraf-1   0.05%     32.05MiB / 3.838GiB   0.82%     19.8kB / 71.7kB   0B / 4.1kB   8

We can keep activating more plugins in the telegraf configuration file and restart telegraf to see those new metrics being collected by InfluxDB.

Cleanup

To teardown the containers just run the following commands.

$ cat cleanup.sh

docker compose stop

docker compose rm

sudo rm -rf /etc/influxdb2/\*

sudo rm -rf /var/lib/influxdb2/\*

$ chmod +x cleanup.sh

$ ./cleanup.sh

[+] Running 2/2

⠿ Container influxdb-telegraf-1 Stopped 0.2s

⠿ Container influxdb-influxdb-1 Stopped 0.2s

? Going to remove influxdb-telegraf-1, influxdb-influxdb-1 Yes

[+] Running 2/0

⠿ Container influxdb-influxdb-1 Removed 0.0s

⠿ Container influxdb-telegraf-1 Removed 0.0s  
  
  
Prerequisite :  
S1 - PoC on Telegraf and InfluxDB with EDR  
  
  
S2 -   
S2 - 3805 - POC - Develop WebHook Telegraf Module to post to IoT Core