Grafana

Grafana??????????????????????????????????????Grafana
??
?????Grafana???????????????

- Installation
- Learning
- InfluxDB
- vSphere Monitoring
- Telegraf
- DB2 Monitoring
- Dashboards Setup
- Why Monitoring
- Prometheus
- Plugins
- AIX/Linux Monitoring with njmon
- Datasource

Installation

With Docker

- Run Grafana Docker image | Grafana documentation
- Configure Docker image | Grafana documentation

```
docker volume create grafana-storage

docker run -d --name=grafana -p 3000:3000 \
  -v grafana-storage:/var/lib/grafana \
  grafana/grafana
```

docker-compose.yaml

```
version: '3.8'
services:
  grafana:
    image: grafana/grafana
    container_name: grafana
    restart: unless-stopped
    ports:
      - '3000:3000'
    volumes:
      - grafana-storage:/var/lib/grafana
volumes:
  grafana-storage: {}
```

Persistent Configuration

```
docker cp grafana:/etc/grafana/grafana.ini grafana.ini
docker stop grafana
docker rm grafana

docker run -d --name=grafana -p 3000:3000 \
  -v grafana-storage:/var/lib/grafana \
  -v $PWD/grafana.ini:/etc/grafana/grafana.ini \
```

```
  grafana/grafana
```

RHEL 8

- Install on RPM-based Linux | Grafana documentation

```
cat <<EOF | sudo tee /etc/yum.repos.d/grafana.repo
[grafana]
name=grafana
baseurl=https://packages.grafana.com/oss/rpm
repo_gpgcheck=1
enabled=1
gpgcheck=1
gpgkey=https://packages.grafana.com/gpg.key
sslverify=1
sslcacert=/etc/pki/tls/certs/ca-bundle.crt
EOF

dnf makecache
dnf install grafana
```

Start the service

```
systemctl start grafana-server
systemctl status grafana-server
systemctl enable grafana-server
```

Access to the Web site

- URL: http://server-ip-address:3000
- Login: admin / admin

Learning

Dashboard Visualizing

- Visualising Latency Variance in Grafana in 2019
- [Grafana] Troubleshoot queries

MySQL Monitoring

- 2MySQL Simple Dashboard
- 2MySQL Simple Dashboard 4 MariaDB
- my2Collector
- [Video] Grafana : Install MySQL Dashboard and Collector in Grafana
- Install MySQL Dashboard and Collector

vSphere ESXi Monitoring

- [Video] Grafana Monitoring | FREE Beginner course | Collect vSphere Metrics
- How To Monitor VMware ESXi with Grafana and Telegraf
- SexiGraf - vSphere Appliance

HAProxy Monitoring

- HAProxy Monitoring (the InfluxDB Way)

Grafana Tutorials

- https://sbcode.net/grafana/
- [Video] Introduction to Grafana Variables and Templates with Prometheus
- [Video] Variables with AWS CloudWatch

Telegraf

- Custom log parsing with latest Tail Plugin, GROK and InfluxDB
- Parsing Logs into InfluxDB Using Telegraf's Logparser Plugin
- Visualising Latency Variance with ping in Grafana in 2019

- ?????Telegraf+InfluxDB+Grafana???????????

Amazon Cloudwatch

- Grafana cloudwatch | Dynamic dashboard creation with variables

Loki + Promtail

- How To Forward Logs to Grafana Loki using Promtail
- LOG SCRAPPING MADE EASY WITH GRAFANA LOKI IN WINDOWS
- Read Nginx Logs with Promtail
- [Grafana] Get logs into Loki
- Configuring Promtail
- [Grafana] Promtail Pipelines
- ??????Grafana Loki
- Grafana Loki ?????????? ELK ? EFK
- Meet Grafana LOKI, a Log Aggregation System for Everything

Plugins

Hourly Heatmap

- Plugin showcase: The hourly heatmap panel, built on Grafana's new plugin platform
- Hourly Heatmap for Grafana

Asterisk Integration

NOTE: For Grafana Cloud only.

- How to observe your Asterisk instance with Grafana Cloud
- Asterisk integration for Grafana Cloud

InfluxDB

Installation

- Install InfluxDB | InfluxDB OSS v2 Documentation (influxdata.com)
- Download: https://portal.influxdata.com/downloads/

Install Influx DB

```
# Red Hat/CentOS/Fedora
cat <<EOF | sudo tee /etc/yum.repos.d/influxdata.repo
[influxdata]
name = InfluxData Repository - Stable
baseurl = https://repos.influxdata.com/stable/\$basearch/main
enabled = 1
gpgcheck = 1
gpgkey = https://repos.influxdata.com/influxdata-archive_compat.key
EOF

yum install influxdb2
```

```
# With Docker
# --reporting-disabled □□□□□□□□
mkdir data

docker run \
    --name influxdb \
    -p 8086:8086 \
    --volume $PWD/data:/var/lib/influxdb2 \
    influxdb:2.7.4 --reporting-disabled
```

Install Influx CLI

> ? InfluxDB 2.1 ???influx CLI ? influxDB ????????

Download: Install and use the influx CLI | InfluxDB OSS 2.7 Documentation (influxdata.com)

```
# amd64
wget https://dl.influxdata.com/influxdb/releases/influxdb2-client-2.7.1-linux-amd64.tar.gz
tar xvzf path/to/influxdb2-client-2.7.1-linux-amd64.tar.gz
mv influx /usr/local/bin/
```

Start the service

```
systemctl start influxdb
systemctl status influxdb
systemctl enable influxdb
```

Set up and initialize DB (v2.7+)

1. Visit `localhost:8086` in a browser
2. Create a user, bucket and organization names.
   - Initial username
   - Password
   - Initial organization name
   - Initial bucket name
3. The **API Tokens** will be generated.
4. Copy the generated token and store it for safe keeping.

InfluxDB configuration

file: `/etc/influxdb/config.{toml,yaml,yml,json}`

```
# View the server configurations
influx server-config
```

config.toml:

- `/var/lib/influxdb/engine` ??????

```
bolt-path = "/var/lib/influxdb/influxd.bolt"
engine-path = "/var/lib/influxdb/engine"
```

Optional: With Docker

```
docker exec -it influxdb influx config create --config-name local-admin --host-url
http://localhost:8086 --org <YOUR-ORG> --token <YOUR-TOKEN --active

docker cp influxdb:/etc/influxdb2/influx-configs ./
```

```
docker exec -it influxdb influx server-config > config.yml

docker run -p 8086:8086 \
  -v $PWD/config.yml:/etc/influxdb2/config.yml \
  -v $PWD/influx-configs:/etc/influxdb2/influx-configs \
  -v $PWD/data:/var/lib/influxdb2 \
  influxdb:2.7.4
```

Set up the influx CLI (v2.7+)

> ???????? `influx` CLI ?????????????? `config create` ??????????

```
# Create config
influx config create --config-name <config-name> \
  --host-url http://localhost:8086 \
  --org <your-org> \
  --token <your-auth-token> \
  --active

influx config ls

# Enter InfluxQL shell
influx v1 shell
> show databases
> quit

# View the server configuration
influx server-config
```

Schema Design

official tutorial: [InfluxDB schema design]

Data organization

- Bucket

    - Measurement
        - Tags
        - Fields
        - Timestamp

Data Elements

- timstamp
- measurement
- tag key
- tag value
- field key
- field value

Where to store data (tag or field)

tag value

- ??????? string
- ????????
- ????????????

field value

- ??????? strings, floats, integers, or boolean
- ??? String?????????????
- ??? Integer??????????? `i`
- ??? GROUP BY ?????
- ??????????
- ???????????????? field value ????????????????
- field ????? index?????????? field???????

?????

```
# Measurement name with spaces
my\ Measurement fieldKey="string value"


# Double quotes in a string field value
myMeasurement fieldKey="\"string\" within a string"


# Tag keys and values with spaces
myMeasurement,tag\ Key1=tag\ Value1,tag\ Key2=tag\ Value2 fieldKey=100
```

?????

Measurement names, tag keys, and field keys ????? `_` ???

Management

Service

```
systemctl start influxdb
systemctl enable influxdb
systemctl stop influxdb
```

InfluxDB v2.7

```
# InfluxQL shell
influx v1 shell
> show databases
> quit

# User management
influx user ls
## Create user
influx user create -n <username> -p <password> -o <org-name>
## Delete user
influx user delete -i <user-id>

# Bucket management
influx bucket ls
## Creaye bucket
influx bucket create --name <bucket-name> --org <org-name> --retention <retention-period-
duration>
influx bucket delete -n <bucket-name>
## Rename the bucket
influx bucket update -i <bucket-id> --name <new-bucket-name>
## Update the retention
influx bucket update -i <bucket-id> --retention 90d

# Token management
influx auth ls
```

Case: ?????? vmware ? token ???? bucket vmware

```
# Create a user for the Org windbs
influx user create -n vmware -o windbs
influx user password -n vmware

# Create a token for the User vmware and <bucket-id>

influx auth create \
  --org windbs \
  --read-bucket 299f5d260eab27cc \
  --write-bucket 299f5d260eab27cc \
  --user vmware \
  --description "vmware's token"
```

Case: ???? Token ??? Org ?????

```
influx auth create \
  --org my-org \
  --description "my-org-all-access" \
  --all-access
```

Case: ???? Token ??? Org ? Operator ??

```
influx auth create \
  --org my-org \
  --description "my-org-operator" \
  --operator
```

InfluxDB v1.8

Create a new user and database

```
influx
> CREATE USER admin WITH PASSWORD 'adminpass' WITH ALL PRIVILEGES
> exit

influx -username admin -password adminpass
> set password for admin = 'newpass'
>
```

```
> create database mmap_nmon with duration 180d
> create user mon with password 'thisispassword'
> grant ALL on mmap_nmon to mon
> show GRANTS for mon
```

Login to InfluxDB

```
# In the shell
influx -username admin -password thisispass

# In the InfluxDB CLI
> auth
username:
password
```

Logs (v2.7+)

```
journalctl -u influxdb
journalctl -n 50 -f -u influxdb
```

Configure log level

Level: debug, info (default), error

/etc/influxdb/config.toml:

```
log-level = "info"
```

Enable the Flux query log

/etc/influxdb/config.toml:

```
flux-log-enabled = true
```

Flux (v2.x)

> 2024/1/24 ??: Flux language ??????????????? InfluxQL ? core SQL ????????

????? Flux language ????????????? Web ?? Data Explorer?

List buckets

```
buckets()
```

List all measurements in a bucket

```
import "influxdata/influxdb/schema"

schema.measurements(bucket: "example-bucket")
```

List field keys

```
import "influxdata/influxdb/schema"

schema.fieldKeys(bucket: "example-bucket")
```

List fields in a measurement

```
import "influxdata/influxdb/schema"

schema.measurementFieldKeys(
    bucket: "example-bucket",
    measurement: "example-measurement",
)
```

List tags in a measurement

```
import "influxdata/influxdb/schema"

schema.measurementTagKeys(
    bucket: "example-bucket",
    measurement: "example-measurement",
)
```

Filter by fields and tags

```
from(bucket: "example-bucket")
    |> range(start: -1h)
    |> filter(fn: (r) => r._measurement == "example-measurement-name" and r.mytagname ==
"example-tag-value")
```

```
    |> filter(fn: (r) => r._field == "example-field-name")
```

??????

```
|> first()
|> last()
|> limit(n: 3)
```

InfluxQL (v1.x)

```
> show databases
name: databases
name
----
_internal
nmon_reports
nmon2influxdb_log

> show users
user  admin
----  -----
admin true
mon   false

> use nmon_reports
Using database nmon_reports
> show measurements
name: measurements
name
----
CPU_ALL
DISKAVGRIO
DISKAVGWIO
DISKBSIZE
DISKBUSY
DISKREAD
DISKREADSERV
DISKRIO
```

```
DISKRXFER
DISKSERV
...
```

Retention Policy

```
# for current database
> show retention policies
name    duration  shardGroupDuration replicaN default
----    --------  ------------------ -------- -------
autogen 8760h0m0s 168h0m0s           1        true

# for specified database
> show retention policies on nmon2influxdb_log
name          duration shardGroupDuration replicaN default
----          -------- ------------------ -------- -------
autogen       0s       168h0m0s           1        false
log_retention 48h0m0s  24h0m0s            1        true

# Create a policy
> CREATE RETENTION POLICY "one_day_only" ON "NOAA_water_database" DURATION 1d REPLICATION 1

# Alter the policy
> ALTER RETENTION POLICY "what_is_time" ON "NOAA_water_database" DURATION 3w SHARD DURATION 2h
DEFAULT

# Delete a policy
> DROP RETENTION POLICY "what_is_time" ON "NOAA_water_database"
```

Verify the account

```
curl -G http://localhost:8086/query -u mon:thisispassword --data-urlencode "q=SHOW DATABASES"
{"results":[{"statement_id":0,"series":[{"name":"databases","columns":["name"],"values":[["mma
p_nmon"]]}]}]}
```

?????

```
SHOW MEASUREMENTS  --查詢當前資料庫中含有的表
SHOW FIELD KEYS --查看當前資料庫所有表的欄位
show field keys from <measurement-name>
```

```
SHOW series from pay --查看key數據
SHOW TAG KEYS FROM "pay" --查看key中tag key值
SHOW TAG VALUES FROM "pay" WITH KEY = "merId" --查看key中tag 指定key值對應的值
SHOW TAG VALUES FROM cpu WITH KEY IN ("region", "host") WHERE service = 'redis'
DROP SERIES FROM  WHERE ='' --刪除key
SHOW CONTINUOUS QUERIES   --查看連續執行命令
SHOW QUERIES  --查看最後執行命令
KILL QUERY  --結束命令
SHOW RETENTION POLICIES ON mydb  --查看保留數據
show series cardinality on mydb --資料庫總筆數
```

????

```
SELECT * FROM /.*/ LIMIT 1  --查詢當前資料庫下所有表的第一行記錄
select * from pay  order by time desc limit 2
select * from  db_name."POLICIES name".measurement_name --指定查詢資料庫下資料保留中的表資料 POLICIES name資料保留
```

????

```
delete from "query" --刪除表所有資料，則表就不存在了
drop MEASUREMENT "query"   --刪除表（注意會把資料保留刪除使用delete不會）
DELETE FROM cpu
DELETE FROM cpu WHERE time < '2000-01-01T00:00:00Z'
DELETE WHERE time < '2000-01-01T00:00:00Z'
DROP DATABASE “testDB” --刪除資料庫
DROP RETENTION POLICY "dbbak" ON mydb --刪除保留資料為dbbak資料
DROP SERIES from pay where tag_key='' --刪除key中的tag
```

????

- mean: ???
- sum: ??
- min: ???
- max: ???
- count: ???
- difference: ??? (????????)
- non_negative_difference: ???? (????????)

```
select * from pay   order by time desc limit 2
select mean(allTime) from pay where time >= today() group by time(10m) tz('Asia/Taipei')
select * from pay tz('Asia/Taipei') limit 2
```

```
SELECT sum(allTime) FROM "pay" WHERE time > now() - 10s
select count(allTime) from pay  where time > now() - 10m  group by time(1s)

select difference("commit_sql") from "snapdb2" where time > now() - 1h limit 10
select non_negative_difference(*) from "snapdb2" where time > now() - 1h limit 10
select non_negative_difference(/commit_sql|rollback_sql/) from "snapdb2" where time > now() -
1h limit 10
```

User and Privileges

```
> CREATE USER "todd" WITH PASSWORD '123456'

> SHOW GRANTS for "todd"

> SET PASSWORD FOR "todd" = 'newpassword'

> GRANT READ ON "NOAA_water_database" TO "todd"
> GRANT ALL ON "NOAA_water_database" TO "todd"

> REVOKE ALL PRIVILEGES FROM "todd"
> REVOKE ALL ON "NOAA_water_database" FROM "todd"
> REVOKE WRITE ON "NOAA_water_database" FROM "todd"

> DROP USER "todd"
```

Covert timestamp to normal datetime

```
influx -precision rfc3339
# Or, in CLI
> precision rfc3339
```

Convert InfluxData to csv format

```
# With CLI
# There is also useful -precision option to set format of timestamp.
influx -database 'database_name' -execute "SELECT * FROM table_name" -format csv > test.csv

influx -username your_user_if_any -password "secret!" -database 'db_name' -host 'localhost' -
execute 'SELECT * FROM "db_name"."your_retention_policy_or_nothing"."your_measurement_name"
WHERE time > '\''2017-09-05'\'' and time < '\''2017-09-05T23:59:59Z'\'' AND
```

```
other_conditions_if_required' -format 'csv' > /tmp/your_measurement_name_20170905.csv

# With HTTP-API
# Samples:
# "q=SELECT * FROM \"mymeasurement\" where time > now() - 130d"
# "q=SELECT * FROM \"mymeasurement\" where (time < now() - 130d) and  (time > now() - 260d)"
curl -G 'http://localhost:8086/query' --data-urlencode "db=mydb" --data-urlencode
"epoch=#timeunit" --data-urlencode "q=SELECT * FROM \"mymeasurement\" " -H "Accept:
application/csv" >  mytargetcsv.csv
```

Tag Query

```
> show tag values from {measurement} with key={key}
> show tag values from {measurement} with key={key} where {tag-key}={tag-value}
```

Learning

- [Influxdb] influxdb????
- Use Grafana with InfluxDB v1.8 | InfluxDB OSS 1.8 Documentation (influxdata.com)
- Setting up InfluxDB v2 (Flux) with InfluxQL in Grafana | by Ivana Huckova | Medium

vSphere Monitoring

Method #1: Telegraf + InfluxDB

- VMware vSphere - Overview | Grafana Labs
- Telegraf: VMware vSphere Input Plugin

Install Telegraf

Download: https://portal.influxdata.com/downloads/

```
yum localinstall telegraf-1.18.3-1.x86_64.rpm
```

Configure Telegraf

Create a configuration file

```
telegraf config > /etc/telegraf/telegraf-vmware.conf
```

vi /etc/telegraf/telegraf-vmware.conf

Log file

```
...
[agent]
...
  logfile = "/var/log/telegraf/telegraf-vmware.log"
...
  ## If set to true, do no set the "host" tag in the telegraf agent.
  omit_hostname = true
```

Output for InfluxDB 1.x

```
# Configuration for sending metrics to InfluxDB 1.x
[[outputs.influxdb]]
    urls = ["http://10.10.2.209:8086"]
    database = "vmware"
```

```
    timeout = "0s"
    username = "admin"
    password = "dba4mis"
    retention_policy = "200d"
```

Output for InfluxDB 2.x

```
[[outputs.influxdb_v2]]
  ## The URLs of the InfluxDB cluster nodes.
  ##
  ## Multiple URLs can be specified for a single cluster, only ONE of the
  ## urls will be written to each interval.
  ##   ex: urls = ["https://us-west-2-1.aws.cloud2.influxdata.com"]
  urls = ["http://127.0.0.1:8086"]

  ## Token for authentication.
  token = "Your-Token"

  ## Organization is the name of the organization you wish to write to.
  organization = "Your-Org-Name"

  ## Destination bucket to write into.
  bucket = "Tour-Bucket-Name"

  ## Timeout for HTTP messages.
  timeout = "5s"
```

Input

????: Telegraf: VMware vSphere Input Plugin

```
###############################################################################
#                            INPUT PLUGINS                                    #
###############################################################################


## Realtime instance
[[inputs.vsphere]]
  interval = "60s"
```

```
  ## List of vCenter URLs to be monitored. These three lines must be uncommented
  ## and edited for the plugin to work.
  vcenters = [ "https://vcenter-server-ip/sdk" ]
  username = "admin@vsphere.local"
  password = "ThisPassword"

  # Exclude all historical metrics
  datastore_metric_exclude = ["*"]
  cluster_metric_exclude = ["*"]
  datacenter_metric_exclude = ["*"]
  resourcepool_metric_exclude = ["*"]

  #max_query_metrics = 256
  #timeout = "60s"
  insecure_skip_verify = true
  force_discover_on_init = true

  collect_concurrency = 5
  discover_concurrency = 5



## Historical instance
[[inputs.vsphere]]
 interval = "300s"

  vcenters = [ "https://vcenter-server-ip/sdk" ]
  username = "admin@vsphere.local"
  password = "ThisPassword"

  host_metric_exclude = ["*"] # Exclude realtime metrics
  vm_metric_exclude = ["*"] # Exclude realtime metrics

  insecure_skip_verify = true
  force_discover_on_init = true
  max_query_metrics = 256
  collect_concurrency = 3
```

Configure systemd

```
cp /usr/lib/systemd/system/telegraf.service /usr/lib/systemd/system/telegraf-vmware.service
sed -i 's/telegraf.conf/telegraf-vmware.conf/g' /usr/lib/systemd/system/telegraf-
vmware.service
```

Startup Telegraf

```
systemctl daemon-reload
systemctl start telegraf-vmware
systemctl enable telegraf-vmware
```

Configure InfluxDB

Set the retention policy

```
[root@mm-mon ~]# influx -username admin -password dba4mis
Connected to http://localhost:8086 version 1.8.5
InfluxDB shell version: 1.8.5
> show retention policies on vmware
name    duration shardGroupDuration replicaN default
----    -------- ------------------ -------- -------
autogen 0s       168h0m0s           1        true
> alter retention policy "autogen" on "vmware" duration 200d shard duration 1d
> show retention policies on vmware
name    duration  shardGroupDuration replicaN default
----    --------  ------------------ -------- -------
autogen 4800h0m0s 24h0m0s            1        true
```

Configure Grafana

1. Add a datasource for InfluxDB
   - Name: VMware
   - Type: InfluxDB
   - Database: vmware
   - Username: <InfluxDB Credential>
   - Password: <InfluxDB Credential>
2. Import the dashboards
   1. https://grafana.com/grafana/dashboards/8159
   2. https://grafana.com/grafana/dashboards/8165
   3. https://grafana.com/grafana/dashboards/8168
   4. https://grafana.com/grafana/dashboards/8162

FAQ

Q: ????? VM ????? Dashoboard?

A: ??? InfluxDB ?????? VM ? data????????? Dashboard Settings > Variables > virtualmachine > ?? Update??? Preview of values ?????? VM name?

?? InfluxDB

```
# Check all current VM names
select DISTINCT("vmname") from (select "ready_summation","vmname" from "vsphere_vm_cpu" WHERE time > now() - 10m)
```

Q: Telegraf ????

> [inputs.vsphere] Error in plugin: while collecting vm: ServerFaultCode: A specified parameter was not correct: querySpec[0].endTime

A: ??????????

```
force_discover_on_init = true
```

Q: Issue: VMware vSphere - Overview

> vCenter CPU/RAM ????????

A: ???? > Flux language syntax

? <vcenter-name> ????? vm ??

```
from(bucket: v.defaultBucket)
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["_measurement"] == "vsphere_vm_cpu")
  |> filter(fn: (r) => r["_field"] == "usage_average")
  |> filter(fn: (r) => r["vmname"] == "<vcenter-name>_vCenter")
  |> group(columns: ["vmname"])
  |> aggregateWindow(every: v.windowPeriod, fn: mean, createEmpty: false)
  |> yield(name: "mean")
```

> “ Cluster ???????? cluster name

A: ?? Dashboard > Variables > clustername > Flux language syntax

```
from(bucket: v.defaultBucket)
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["_measurement"] == "vsphere_host_cpu")
  |> filter(fn: (r) => r["clustername"] != "")
  |> filter(fn: (r) => r["vcenter"] == "${vcenter}")
  |> keep(columns: ["clustername"])
  |> distinct(column: "clustername")
  |> group()
```

Method #2: SexiGraf

- Official: http://www.sexigraf.fr/quickstart/
- OS-based: Ubuntu 16.04.6 LTS

Download the OVA appliance

- http://www.sexigraf.fr/quickstart/
- https://github.com/sexibytes/sexigraf

vCenter/vSphere Credential for monitor only

vCenter Web Client > ??? > ???? > Single Sign On: ?????? > ??

- ?????: winmon
- ??: xxxx
- ????: xxxx

vCenter Web Client > ??? > ????? > ?? > ????

- ???: vsphere.local , ?? winmon
- ??: ??
- ?????: ??

Deploy the OVA to vCenter/ESXi

??? ESXi 6.5 ????????

> Line 163: Unable to parse 'tools.syncTime' for attribute 'key' on element 'Config'.

????: ?? OVF-Tool ??? OVA ???? OVF ????

```
# Before
<vmw:Config ovf:required="true"  vmw:key="tools.syncTime" vmw:value="true"/>

# After
<vmw:Config ovf:required="false"  vmw:key="tools.syncTime" vmw:value="true"/>
```

???????????

First to Start the VM

1. SSH Credential: root / Sex!Gr@f

2. Need to manually configure the IP, Edit the `/etc/network/interfaces` .

3. Configure the hostname

```
hostnamectl set-hostname esx-mon
```

4. Configure the timezone and time server

```
timedatectl set-timezone Asia/Taipei
```

vi /etc/ntp.conf

```
#pool 0.ubuntu.pool.ntp.org iburst
#pool 1.ubuntu.pool.ntp.org iburst
#pool 2.ubuntu.pool.ntp.org iburst
#pool 3.ubuntu.pool.ntp.org iburst

# Use Ubuntu's ntp server as a fallback.
#pool ntp.ubuntu.com
```

```
# Added the local time server
server 192.168.21.86 prefer iburst
```

Restart the ntpd

```
systemctl stop ntp
systemctl start ntp

# Check the timeserver
ntpq -p
```

First to Login the Grafana Web

1. Login: admin / Sex!Gr@f
2. Add the credential to connect to the vCenter server managed: Search > SexiGraf > SexiGraf Web Admin > Credential Store
   - vCenter IP: <vCenter/ESXi IP or FQDN>
   - Username: <Username to login to vCenter/ESXi>
   - Password: <Password to login to vCenter/ESXi>

Telegraf

Installation

- [Install Telegraf | Telegraf 1.26 Documentation (influxdata.com)]

RHEL

```
cat <<EOF | sudo tee /etc/yum.repos.d/influxdb.repo
[influxdb]
name = InfluxData Repository - Stable
baseurl = https://repos.influxdata.com/stable/\$basearch/main
enabled = 1
gpgcheck = 1
gpgkey = https://repos.influxdata.com/influxdata-archive_compat.key
EOF

sudo yum install telegraf
```

Ubuntu/Debian

```
curl -s https://repos.influxdata.com/influxdata-archive_compat.key > influxdata-
archive_compat.key
echo '393e8779c89ac8d958f81f942f9ad7fb82a25e133faddaf92e15b16e6ac9ce4c influxdata-
archive_compat.key' | sha256sum -c && cat influxdata-archive_compat.key | gpg --dearmor | sudo
tee /etc/apt/trusted.gpg.d/influxdata-archive_compat.gpg > /dev/null
echo 'deb [signed-by=/etc/apt/trusted.gpg.d/influxdata-archive_compat.gpg]
https://repos.influxdata.com/debian stable main' | sudo tee
/etc/apt/sources.list.d/influxdata.list
sudo apt-get update && sudo apt-get install telegraf
```

Configuration

```
telegraf config > telegraf.conf
```

```
# Using filter
telegraf --input-filter exec --output-filter influxdb_v2 config > /etc/telegraf/telegraf.conf

# Test for the configuration
telegraf -config /etc/telegraf/telegraf.conf -test
```

Custom systemd

```
cp /usr/lib/systemd/system/telegraf.service /etc/systemd/system/telegraf-db2.servic
```

telegraf-db2.service:

```
## □□□□□
EnvironmentFile=-/etc/default/telegraf-db2

## □□□□□
ExecStart=/usr/bin/telegraf -config /etc/telegraf/telegraf-db2.conf $TELEGRAF_OPTS
```

Reload the daemon

```
systemctl list-unit-files --type service
systemctl daemon-reload
```

Outputs.InfluxDB v1

```
###############################################################################
#                            OUTPUT PLUGINS                                   #
###############################################################################


# Configuration for sending metrics to InfluxDB
[[outputs.influxdb]]
  urls = ["http://influxdb.server.ip.addr:8086"]
  database = "db-name"
  timeout = "0s"
  username = "db-user"
  password = "db-pass"
```

Outputs.InfluxDB v2

```
###############################################################################
#                            OUTPUT PLUGINS                                   #
###############################################################################


[[outputs.influxdb_v2]]
  urls = ["http://influxdb.server.ip.addr:8086"]
  token = "example-token"
  organization = "example-org"
  bucket = "example-bucket"
```

Inputs.exec

data_format = "influx"

???????

```
# Syntax for Line protocol
<measurement>[,<tag_key>=<tag_value>[,<tag_key>=<tag_value>]]
<field_key>=<field_value>[,<field_key>=<field_value>] [<timestamp>]

|                                                    |

<whitesoace>                                         <whitesoace>
# Example
airsensors,location=bedroom,sensor_id=MI0201 temperature=19.1,humidity=85i,battery=78i
1556813561098000000
```

> - ?? Timestamp ???????????? InfluxDB ??????(UTC)?
> - ?????Line protocol | InfluxDB OSS v2 Documentation (influxdata.com)
> - field_value ??? Integer???? i?? String???????
> - measurename, tag_key, tag_value, field_key ????????
> - ????? measurename, fileld_key, field_value?
> - ????????????????????? (\n) ????
> - ????????

Plugins

- Plugin directory | Telegraf Documentation (influxdata.com)

Scripts

Samples #1

```
#/bin/bash

devname=(`lsblk| grep 'disk'|awk '{print $1}'`)
dirname=(`lsblk| grep 'disk'|awk '{if ($7=="") print "/";else print $7}'`)
#At that time, I wanted to store these directory names in dictionary format, and later changed to variable mode, shell Of[ ] { } * @ $Special characters will drive you crazy
#declare -A devdict
devnum=`expr ${#devname[@]} - 1`
for i in `seq 0 $devnum`;do
  if [-z "${dirname[$i]}" ];then
    eval ${devname[$i]}="/"
  else
    eval ${devname[$i]}="${dirname[$i]}"
  fi
  #devdict+=([${devname[$i]}]="${dirname[$i]}")
done
#echo ${!devdict[*]}
#echo ${devdict[*]}

ioarry=`iostat -x | grep sd|awk '{print "datadir=${"$1"}@r="$4",w="$5",await="$10",svctm="$11",util="$12}'`
for i in ${ioarry[@]};do
  eval temp="${i}"
  #Replace the special character @, and the space in the shell will be truncated to two elements
  temp=${temp/@/ }
  echo "exec,${temp}"
  #Ensure that the final output is in the following format. The first character is the measurement name. If the input.exec plug-in has the configuration name "suffix", the suffix will be added automatically
  #The output format is measurement name, comma, tag keys (comma separated), space, filed keys (comma separated)
```

```
  #The data format output mismatch will lead to the failure of telegraf to parse the data and go to the influxdb. It took a long time to debug and didn't look at the hole dug by the official website
  #exec,datadir=/data/data11 r=4.1,w=6.1,await=0.83,svctm=1.35,util=1.46"
done
#echo ${devdict[@]}
```

```
[[inputs.exec]]
  ##Commands array
  commands = ["bash /appcom/telegraf/collect_iostat.sh",]
  timeout='5s'
  ##Suffix for measurements
  name_suffix="_collectiostat"
  data_format="influx"
```

Sample #2

```
#!/bin/sh
hostname=`hostname`
uptime=`awk '{print $1}' /proc/uptime`
if uptime |grep -q user ; then
load1=`uptime | grep -ohe 'up .*' | sed 's/,//g' | awk '{ print $7}'`
load5=`uptime | grep -ohe 'up .*' | sed 's/,//g' | awk '{ print $8}'`
load15=`uptime | grep -ohe 'up .*' | sed 's/,//g' | awk '{ print $9}'`
else
load1=`uptime | grep -ohe 'up .*' | sed 's/,//g' | awk '{ print $5}'`
load5=`uptime | grep -ohe 'up .*' | sed 's/,//g' | awk '{ print $6}'`
load15=`uptime | grep -ohe 'up .*' | sed 's/,//g' | awk '{ print $7}'`
fi
echo "uptime,host=$hostname uptime=$uptime,load1=$load1,load5=$load5,load15=$load15"
```

```
[agent]
interval = "5s"
round_interval = true
[[inputs.swap]]
  [inputs.swap.tags]
    metrics_source="telegraf_demo"
[[inputs.exec]]
  commands = ["/etc/telegraf/uptime.sh"]
  data_format = "influx"
```

```
  [inputs.exec.tags]
    metrics_source="telegraf_demo"
[[outputs.influxdb]]
  url = "https://influxdemo:8086"
  database = "telegraf"
```

Sample #3

```
#! /bin/bash
/usr/bin/speedtest --format json | jq '.download.bandwidth = .download.bandwidth / 125000 |
.upload.bandwidth = .upload.bandwidth / 125000'
```

```
[[inputs.exec]]
  commands = [
    "/home/rock64/speedtest.sh"
    ]
  interval = "300s"
  timeout = "60s"
```

Sample #4

```
[[inputs.exec]]
  commands = ["sh -c 'sysctl -n dev.cpu.0.temperature | tr -d C'"]
  name_override = "cpu_temp"
  timeout = "5s"
  data_format = "value"
  data_type = "float"
  [inputs.exec.tags]
    core = "core0"

[[inputs.exec]]
  commands = ["sh -c 'sysctl -n dev.cpu.1.temperature | tr -d C'"]
  name_override = "cpu_temp"
  timeout = "5s"
  data_format = "value"
  data_type = "float"
  [inputs.exec.tags]
    core = "core1"

[[inputs.exec]]
```

```
  commands = ["sh -c 'sysctl -n dev.cpu.2.temperature | tr -d C'"]
  name_override = "cpu_temp"
  timeout = "5s"
  data_format = "value"
  data_type = "float"
  [inputs.exec.tags]
    core = "core2"

[[inputs.exec]]
  commands = ["sh -c 'sysctl -n dev.cpu.3.temperature | tr -d C'"]
  name_override = "cpu_temp"
  timeout = "5s"
  data_format = "value"
  data_type = "float"
  [inputs.exec.tags]
    core = "core3"
```

Q & A

[agent] Error terminating process: operation not permitted

Causation: ? telegraf.conf ?????? agent ???????? timeout ???????? agent ???????telegraf ????? agent ???

Solution: ????????????? agent ????????? timeout ??????????????????

??????????? timeout ?????? agent ????????????????????????????

?? telegraf ???? agent ???????????????? timeout ???

??????agent ?? sudo ???? db2 ??????????

```
[[inputs.exec]]
    interval = "1h"
    commands = ["sudo -u db2mon sh -c '/home/db2mon/bin/collect_db2x1h.sh -d centdb -a
b_centdb'"]
    timeout = "5s"
    data_format = "influx"
```

?? telegraf ?? kill ? sudo ????????????????????? collect_db2x1h.sh???? telegraf ?? sudo ??????

```
[[inputs.exec]]
    interval = "1h"
    commands = ["/home/db2mon/bin/collect_db2x1h.sh -d centdb -a b_centdb"]
    timeout = "15s"
    data_format = "influx"
```

?????timeout ?????????? agent?????????????

> [inputs.exec] Error in plugin: exec: command timed out for command '/home/db2mon/bin/collect_db2x1h.sh -d centdb -a b_centdb'

??????????? timeout?

Error in plugin: metric parse error: expected tag at 7:20:

Causation: ??? Influxdata ???????

Solution: ??? 7 ??? 20 ????Influxdada ???

`measurement`, `tag-key1=tag-value1,tag-key2=tag-value2` `field-key1=field-value1,field-key2=field-value2,....`

- tag-key type: string
- tag-value type: string
  NOTE: ????????
- field-key type: string
- field-value type: Float | Integer | UInteger | String | Boolean
  NOTE: ??? string ??????

max-series-per-database limit exceeded: (1000000)

Causation: ????????????????? 1000000?

? InfluxDB CLI ???????????????

```
show series cardinality on <db-name>
```

Solution: ?? InfluxDB ????????? `/etc/influxdb/influxdb.conf` ??? 1000000

```
# max-series-per-database = 1000000
max-series-per-database = 2000000
```

?? InfluxDB

```
systemctl restart influxdb
```

DB2 Monitoring

Prerequisites

- InfluxDB (the host is the same as Grafana)
- Separated Linux host
  - Install and running telegraf
  - Install DB2 Client
  - Implement custom scripts

InfluxDB

?? database

```
> create database db2_mon with duration 180d
> create user mon with password 'thisispassword'
> grant read on db2_mon to mon
```

> ❝ mon ?????? Grafana ? datasource ???

Telegraf

??????? telegraf-db2.conf

```
telegraf --input-filter exec --output-filter influxdb config > /etc/telegraf/telegraf-db2.conf
```

?????

```
[agent]
...
...
  logfile = "/var/log/telegraf/telegraf-db2.log"



# Configuration for sending metrics to InfluxDB
[[outputs.influxdb]]
```

```
  urls = ["http://10.10.2.209:8086"]
  database = "db2_mon"
  timeout = "0s"
  username = "admin"
  password = "Thispassword"

[[inputs.exec]]
    interval = "300s"

    ## Commands array
    #commands = [
    #  "/tmp/test.sh",
    #  "/usr/bin/mycollector --foo=bar",
    #  "/tmp/collect_*.sh"
    #]
    commands = ["sudo -u db2mon sh -c '/home/db2mon/bin/collect_db2.v2.sh -d dcdb -a b_dcdb -u
dbuser -p dbpass'"]

    ## Timeout for each command to complete.
    timeout = "5s"

    ## measurement name suffix (for separating different commands)
    #name_suffix = "_mycollector"

    ## Data format to consume.
    ## Each data format has its own unique set of configuration options, read
    ## more about them here:
    ## https://github.com/influxdata/telegraf/blob/master/docs/DATA_FORMATS_INPUT.md
    data_format = "influx"

[[inputs.exec]]
    interval = "1h"
    commands = ["sudo -u db2mon sh -c '/home/db2mon/bin/collect_db2x1h.sh -d dcdb -a b_dcdb -u
dbuser -p dbpass'"]
    timeout = "5s"
    data_format = "influx"

### CENTDB
[[inputs.exec]]
    interval = "300s"
```

```
    commands = ["sudo -u db2mon sh -c '/home/db2mon/bin/collect_db2.v2.sh -d centdb -a
b_centdb -u dbuser -p dbpass'"]
    timeout = "5s"
    data_format = "influx"
```

??????????? telegraf-db2.service

> NOTE: ?? telegraf ????????????????????? Output ??????????? telegraf ???

```
cp /usr/lib/systemd/system/telegraf.service /etc/systemd/system/telegraf-db2.service
```

?????

```
// □□□□□
ExecStart=/usr/bin/telegraf -config /etc/telegraf/telegraf-db2.conf $TELEGRAF_OPTS
```

???????????? telegraf-db2????????

```
systemctl list-unit-files --type service
systemctl daemon-reload
```

DB2 Metrics

Bufferpool

```
-- DATA_HIT_RATIO_PERCENT: Data hit ratio, that is, the percentage of time that the database
manager did not need to load a page from disk to service a data page request.
-- INDEX_HIT_RATIO_PERCENT: Index hit ratio, that is, the percentage of time that the database
manager did not need to load a page from disk to service an index data page request.
select BP_NAME, DATA_HIT_RATIO_PERCENT, INDEX_HIT_RATIO_PERCENT
  from sysibmadm.MON_BP_UTILIZATION
  where BP_NAME not like 'IBMSYS%' and BP_NAME<>'IBMDEFAULTBP'
```

Logs (active log & archive log)

```
-- LOG_UTILIZATION_PERCENT: Percent utilization of total log space. (active logs)
-- TOTAL_LOG_USED_KB: The total amount of active log space currently used in the database
-- TOTAL_LOG_AVAILABLE_KB: The amount of active log space in the database that is not being
```

```
used by uncommitted transactions.
select DB_NAME, LOG_UTILIZATION_PERCENT, TOTAL_LOG_USED_KB, TOTAL_LOG_AVAILABLE_KB
  from sysibmadm.log_utilization
```

Connections

```
-- APPLS_CUR_CONS: Indicates the number of applications that are currently connected to the database.
-- LOCKS_WAITING: Indicates the number of agents waiting on a lock.
-- NUM_INDOUBT_TRANS: The number of outstanding indoubt transactions in the database.
select DB_NAME,APPLS_CUR_CONS,LOCKS_WAITING,NUM_INDOUBT_TRANS from sysibmadm.snapdb
```

Transactions

```
-- □□□□□□□□
-- COMMIT_SQL_STMTS: The total number of SQL COMMIT statements that have been attempted.
-- ROLLBACK_SQL_STMTS: The total number of SQL ROLLBACK statements that have been attempted.
select COMMIT_SQL_STMTS, ROLLBACK_SQL_STMTS from sysibmadm.snapdb
```

Custom scripts

Learning

Archive Log Monitor

- Generating Log Archive Activity Histograms

Dashboards Setup

Variables

Get the value from a tag

```
> show tag values from "snapdb" with key="db"
```

Get the value from a field

```
> select DISTINCT("vmname") from (select "ready_summation","vmname" from "vsphere_vm_cpu")
```

Query Editor

InfluxQL with InfluxDB v2

```
SELECT * FROM cpu WHERE time >= $__timeFrom AND time <= $__timeTo
SELECT * FROM cpu WHERE $__timeFilter(time)
SELECT $__dateBin(time) from cpu
```

Time Series: vSphere Cluster CPU Usage

```
SELECT mean("usage_average")
FROM "vsphere_host_cpu"
WHERE ("clustername" =~ /^$clustername$/ AND "cpu" = 'instance-total') AND $timeFilter
GROUP BY time($inter), "clustername", "cpu" fill(none)
```

Gauge: vSphere Datastore Status

```
SELECT mean("used_latest") * (100 / mean("capacity_latest"))
FROM "vsphere_datastore_disk"
WHERE ("source" =~ /^$datastore$/) AND $timeFilter
GROUP BY time($inter) fill(none)
```

Bar gauge: vSphere Datastore Usage Capacity

```
SELECT last("used_latest") * (100 / last("capacity_latest"))
FROM "vsphere_datastore_disk"
WHERE ("source" =~ /^$datastore$/) AND $timeFilter
GROUP BY time($inter) , "source"  fill(none)
```

Stat: Uptime

```
SELECT last("uptime_latest") AS "Uptime"
FROM "vsphere_host_sys"
WHERE ("vcenter" =~ /^$vcenter$/ AND "clustername" =~ /^$clustername$/) AND $timeFilter
GROUP BY time($inter) fill(null)
```

Flux with InfluxDB v2

Time Series: vSphere Cluster CPU Usage

```
from(bucket: v.defaultBucket)
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["_measurement"] == "vsphere_host_cpu")
  |> filter(fn: (r) => r["_field"] == "usage_average")
  |> filter(fn: (r) => r["cpu"] == "instance-total")
  |> filter(fn: (r) => r["clustername"] =~ /${clustername:regex}/)
  |> group(columns: ["clustername"])
  |> aggregateWindow(every: v.windowPeriod, fn: mean, createEmpty: false)
  |> yield(name: "mean")
```

Gauge: vSphere Datastore Status

```
from(bucket: v.defaultBucket)
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["_measurement"] == "vsphere_datastore_disk")
  |> filter(fn: (r) => r["_field"] == "capacity_latest" or r["_field"] == "used_latest")
  |> filter(fn: (r) => r["source"] =~ /${datastore:regex}/)
  |> group()
  |> pivot(rowKey:["_time"], columnKey: ["_field"], valueColumn: "_value")
  |> map(fn: (r) => ({ r with  _value: float(v: r.used_latest) / float(v: r.capacity_latest) *
100.0 }))
  |> group(columns: ["source","_field"])
  |> aggregateWindow(every: v.windowPeriod, fn: mean, createEmpty: false)
```

Bar gauge: vSphere Datastore Usage Capacity

```
from(bucket: v.defaultBucket)
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["_measurement"] == "vsphere_datastore_disk")
  |> filter(fn: (r) => r["_field"] == "capacity_latest" or r["_field"] == "used_latest")
  |> filter(fn: (r) => r["source"] =~ /${datastore:regex}/)
  |> pivot(rowKey:["_time"], columnKey: ["_field"], valueColumn: "_value")
  |> map(fn: (r) => ({ r with  _value: float(v: r.used_latest) / float(v: r.capacity_latest) *
100.0 }))
  |> group(columns: ["source","_field"])
  |> aggregateWindow(every: v.windowPeriod, fn: mean, createEmpty: false)
```

Stat: Uptime

```
from(bucket: v.defaultBucket)
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["_measurement"] == "vsphere_host_sys")
  |> filter(fn: (r) => r["_field"] == "uptime_latest")
  |> filter(fn: (r) => r["vcenter"] =~ /${vcenter:regex}/)
  |> filter(fn: (r) => r["clustername"] =~ /${clustername:regex}/)
  |> group(columns: ["clustername"])
  |> aggregateWindow(every: v.windowPeriod, fn: mean, createEmpty: false)
  |> yield(name: "mean")
```

Why Monitoring

Why Monitoring

Sample #1

- Know when things go wrong
  - Detection & Alerting
- Be able to debug and gain insight
- Detect changes over time and drive technical/business decisions
- Feed into other system/processes (e.g. security, automation)

Sample #2

- Monitoring is used to assess performance of services & applications
- Timely detection of issues and preventing failures
- Capacity planning

Presentation Videos

- Infrastructure and application monitoring using Prometheus by Marco Pas

Prometheus

Installation

Download: Download | Prometheus ?select Operating system: linux, Architecture: amd64

```
tar xzf prometheus-2.43.0.linux-amd64.tar.gz
mv prometheus-2.43.0.linux-amd64 /opt/prometheus
```

?????

```
cd /opt/prometheus/
./prometheus --config.file="prometheus.yml"
```

???? (???????):

- http://localhost:9090/metrics
  ?????????????
- http://localhost:9090/
  ?? Graph?? Expression ?? `promhttp_metric_handler_requests_total` ??? Execute ????????

Configuration

?????????

```
./prometheus --config.file="prometheus.yml" \
    --storage.tsdb.path="/data/prometheus" \
    --storage.tsdb.retention.time=30d
```

- `--storage.tsdb.path`:
  Where Prometheus writes its database. Defaults to data/.
- `--storage.tsdb.retention.time`:
  When to remove old data. Defaults to 15d. Overrides storage.tsdb.retention if this flag is set to anything other than default.
- `--storage.tsdb.retention.size`:
  The maximum number of bytes of storage blocks to retain. The oldest data will be removed first. Defaults to 0 or disabled. Units supported: B, KB, MB, GB, TB, PB, EB. Ex: "512MB". Based on powers-of-2, so 1KB is 1024B. Only the persistent blocks are

deleted to honor this retention although WAL and m-mapped chunks are counted in the total size. So the minimum requirement for the disk is the peak space taken by the wal (the WAL and Checkpoint) and chunks_head (m-mapped Head chunks) directory combined (peaks every 2 hours).

????(????)

RedHat 8

???????

```
useradd -s /sbin/nologin --system prometheus
mkdir /etc/prometheus /data/prometheus
```

????

```
tar xvf prometheus-*.tar.gz
cd prometheus-*/
cp prometheus promtool /usr/local/bin/
cp -r prometheus.yml consoles/ console_libraries/ /etc/prometheus/

chown -R prometheus.prometheus /etc/prometheus
chmod -R 0755 /etc/prometheus
chown prometheus.prometheus /data/prometheus
```

?????: `/etc/systemd/system/prometheus.service`

```
[Unit]
Description=Prometheus Time Series Collection and Processing Server
Wants=network-online.target
After=network-online.target

[Service]
User=prometheus
Group=prometheus
Type=simple
ExecReload=/bin/kill -HUP $MAINPID
EnvironmentFile=/etc/sysconfig/prometheus
ExecStart=/usr/local/bin/prometheus $OPTIONS

[Install]
```

```
WantedBy=multi-user.target
```

?????: `/etc/sysconfig/prometheus`

```
OPTIONS="
  --config.file /etc/prometheus/prometheus.yml \
  --storage.tsdb.path /data/prometheus/ \
  --storage.tsdb.retention.time=30d \
  --web.console.templates=/etc/prometheus/consoles \
  --web.console.libraries=/etc/prometheus/console_libraries \
"
```

????

```
systemctl daemon-reload
systemctl start prometheus.service
systemctl enable prometheus.service
```

Monitor to Linux node

Linux Monitoring with Node Exporter

- Node Exporter for Prometheus Dashboard EN v20201010
- Node Exporter Full
- Node Exporter Server Metrics
- Node Exporter for Prometheus Dashboard based on 11074
- Node Exporter Quickstart and Dashboard
- [Video] Setting up Prometheus and Grafana for monitoring your servers
- MONITORING LINUX HOST METRICS WITH THE NODE EXPORTER

On Linux target

Node Exporter Installation

Download: Download | Prometheus

```
tar xzf node_exporter-1.5.0.linux-amd64.tar.gz
mv node_exporter-1.5.0.linux-amd64 /opt/node_exporter
```

```
chown -R root.root /opt/node_exporter

cd /opt/node_exporter
./node_exporter
# Ctrl + C to exit
```

Set up node_exporter as service

```
# Create a user
useradd -r -c "Node Exporter" -s /sbin/nologin node_exporter

# Create a service file
cat <<EOF>/etc/systemd/system/node_exporter.service
[Unit]
Description=Node Exporter

[Service]
User=node_exporter
EnvironmentFile=/etc/sysconfig/node_exporter
ExecStart=/opt/node_exporter/node_exporter $OPTIONS

[Install]
WantedBy=multi-user.target
EOF
```

```
# Create the file /etc/sysconfig/node_exporter
echo '#OPTIONS=""' > /etc/sysconfig/node_exporter

# Start the node exporter
systemctl daemon-reload
systemctl start node_exporter.service
```

On Prometheus Server

prometheus.yml:

```
scrape_configs:

  # Linux Nodes
  - job_name: linux
```

```
    # Override the global default and scrape targets from this job every 5 seconds.
    scrape_interval: 15s

    static_configs:
      - targets: ['linux-node-ip:9100']
```

Monitor to MySQL

- MySQL Overview
- MySQL Server Exporter
- [Video] Setting up Prometheus and Grafana for monitoring your servers
- Prometheus ??Mysql????Grafana???

Monitor to AIX

- nimon working with Prometheus

Monitor to RabbitMQ

prometheus.yml:

```
scrape_configs:

  # RabbitMQ Nodes
  - job_name: rabbitmq

    # Override the global default and scrape targets from this job every 5 seconds.
    scrape_interval: 15s

    static_configs:
      - targets: ['rmq01:15692', 'rmq02:15692', 'rmq03:15692']
```

Monitor to Containers

- MONITORING DOCKER CONTAINER METRICS USING CADVISOR
- [Video] Node Application Monitoring with cAdvisor Prometheus and Grafana | part 1
- ?????cAdvisor

Plugins

Install plugin on local Grafana

Option 1: with grafana-cli

```
# Internet network is required
# reference to https://grafana.com/docs/grafana/latest/administration/cli/#plugins-commands
grafana-cli plugins install marcusolsson-hourly-heatmap-panel
```

Option 2: manually unpack the .zip file

```
unzip my-plugin-0.2.0.zip -d YOUR_PLUGIN_DIR/my-plugin
```

> By default the plugin_dir is `/var/lib/grafana/plugin`s?

Restart the Grafana

```
systemctl stop grafana-server
systemctl start grafana-server
```

Q & A

Q: ??? marcusolsson-hourly-heatmap-panel-1.0.0 plugin???? Visualization ?????
A: ?? grafana-cli ??? UI ??? Plugins ??????? plugin???????????

AIX/Linux Monitoring with njmon

nimon (NOT njmon) + InfluxDB + Grafana

- njmon = JSON output
- nimon = njmon but straight to InfluxDB

> NOTE: as of version 78, the njmon and nimon have been merged into one binary file.
>
> Using the option `-J` (nimon mode) or `-I` (nimon mode).

njmon

Download: http://nmon.sourceforge.net/pmwiki.php?n=Site.Njmon

InfluxDB

Create a new database for njmon

```
create database aix_njmon with duration 180d
create user mon with password 'thisispassword'
grant ALL on aix_njmon to mon
show GRANTS for mon
```

Grafana

Dashboards

- njmon for AIX Large Set v66
  - Plugin Clock: https://grafana.com/grafana/plugins/grafana-clock-panel/
  - Plugin Pie Chart: https://grafana.com/grafana/plugins/grafana-piechart-panel/
- njmon for AIX Simple Six PLUS Copy V78

    - Plugin Clock: https://grafana.com/grafana/plugins/grafana-clock-panel/
- njmon Linux Simple Six Plus v67
    - Plugin Clock: https://grafana.com/grafana/plugins/grafana-clock-panel/

AIX/Linux

Cron job:

```
## Gathing AIX/Linux performance data with njmon
# Running forever, in case the process is killed the cron job will restart it every one hour.
# -i : the hostname of InfluxDB
# -x : the DB name in InfluxDB
# -y : the DB user
# -z : the DB password
3 * * * * /usr/local/bin/njmon -I -s 60 -k -i <ip-or-hostname-to-InfluxDB> -x <db-name> -y
<db-user> -z <db-pass> > /dev/null 2>&1
```

Datasource

InfluxDB

InfluxDB v2 + InfluxQL (*Recommend*)

- Name: InfluxDBv2_InfluxQL-<BUCKET-NAME> (*Recommend*)
- Query Language: InfluxQL
- HTTP.URL: <URL-TO-InfluxDB-Server>
- Custom HTTP Headers
  - Header: Authorization
  - Value: Token <*INFLUX-API-TOKEN*>
- Database: <*BUCKET-NAME*>

InfluxDB v2 + Flux

- Name: InfluxDBv2_Flux-<*BUCKET-NAME*> (*Recommend*)
- Query Language: Flux
- HTTP.URL: <*URL-TO-InfluxDB-Server*>
- Organization: <*ORG-NAME*>
- Token: <*INFLUX-API-TOKEN*>
- Default Bucket: <*BUCKET-NAME*>