Metrics Playbook Wiki

1. 1. Table of content

Table of Contents

2. 2. How to use

```
source .envrc
ansible-playbook -i inventories/DC_TO_DEPLOY/data-metrics.yaml -C -D
playbooks/PLAYBOOK.yml -v
ansible-playbook -i inventories/DC_TO_DEPLOY/data-metrics.yaml -D
playbooks/PLAYBOOK.yml -v
```

> -C, --check don't make any changes; instead, try to predict some of the changes that may occur -D, --diff when changing (small) files and templates, show the differences in those files; works great with --check

---

3. 3. Requirements

- Python >= 3.8
- Library from `requirements.txt`

---

4. 4. Installation

4.1. 4.1. Create a virtual environment

```
virtualenv venv -p python3
```

4.2. 4.2. Configure pass

- Install pass on Ubuntu or any Debian based system :

  ```
  sudo apt update
  sudo apt install pass
  ```

- Generate GPG key key pair for pass :

  ```
  gpg --full-generate-key
  ```

- Use your token copied from vault and insert it in prompt below :

  ```
  pass insert shadow/vault.gcp.blade-group.net/token
  ```

- Use your NetBox token given by the system team:

  ```
  pass insert shadow/netbox.blade.sh/token
  ```

> If you don't have a NetBox token, ask the system team to give a read-only NetBox token.
>
> If your entry already exist but you need to change it, use the `pass edit` command. And if you need to check if the entry is correct, use the `pass` command.

4.3. 4.3. Install python libraries from `requirements.txt`

```
pip install -r requirements.txt
```

> If the `pip3` command isn't found, please use `pip3` or run the `sudo apt install python3-pip`.

4.4. 4.4. install Ansible roles and collections from `requirements.yml`

```
ansible-galaxy install -r requirements.yml --force
```

---

5. 5. TLS CA

Queries to TLS services protected by internal certificates rely on the presence of our internal CA in the system CA root trust repository. The procedure will vary depending on your distribution, but the following should work with Debian. Current valid CAs can be seen there: https://gitlab.com/blade-group/infra/ansible/-/tree/master/roles/blade.common/files/ca

```
sudo mkdir /usr/local/share/ca-certificates
sudo apt install ca-certificates
sudo cp roles/blade.common/files/ca/*.crt /usr/local/share/ca-certificates/
sudo update-ca-certificates
```

> If you're having any trouble with the CA being recognized after installation, you might need to point the Python requests module explicitly in the right direction with an environment variable `export REQUESTS_CA_BUNDLE=/etc/ssl/certs/ca-certificates.crt`. For macOS users, open the cert:

```
open roles/blade.common/files/ca/ucs-pa1-ca.crt
```

(macOS) Keychain Access is then open. You can see the cert in the list under the name "UCS PA1 CA", with a red cross on it, since it's untrusted. To trust it, double-click on the certificate, unroll the "Trust" menu and choose "Always Trust" for "When using this certificate". It'll automatically set other options to "Always Trust". You can close the window, type your password and you're ready to go. Last step, or you won't be able to interact with Vault. First, we need to find the location of openssl_cafile using

```
FILE=`/usr/bin/python3 -c 'import ssl;
print(ssl.get_default_verify_paths().openssl_cafile)'`
```

Then, copy the content of each certificate file into $FILE

```
cat /usr/local/share/ca-certificates/ucs-pa1-ca.crt | sudo tee -a $FILE
cat /usr/local/share/ca-certificates/blade-infrastructure-pki-intermediate-
ca.crt | sudo tee -a $FILE
cat /usr/local/share/ca-certificates/blade-infrastructure-pki-root-ca.crt |
sudo tee -a $FILE
```

---

6. 6. Ansible commands

A sample ad-hoc command to run on all host :

```
ansible -i inventories/YOUR_DC/data-metrics.yml -m ping all
```

> This ad-hoc command will run a ping command on all data-metrics related VMs.
>
> You can use the -u, --user to specify a user for the SSH connection. Example : `ansible -u root -i inventories/defra01/data-metrics.yml -m ping all`.
>
> You can also specify a role by adding the role name at the end. Example : `ansible -u root -i inventories/defra01/data-metrics.yaml -m ping device_roles_data_clickhouse`. Deploy all inventories :

```
for dc in ams1 ch1 dfr1 ny1 pa1 sv2 tx1; ansible-playbook -i
inventories/{$dc}/data-metric.yaml -D playbooks/docker.yml -vv; end
```

Get inventory list of DEFRA01 :

```
ansible-inventory -i inventories/defra01/data-metrics.yml --graph
```

---

7. 7. Using tags

8. See each playbook for any tags available.

9. 8. Variables

Ansible's variables are defined following this way:

- if a value is common to all DCs, then the variable is set in "playbook" `group_vars`, i.e. `playbooks/group_vars`.
- if a value is different for at least 1 datacenter, then it becomes part of the "inventory" `group_vars`, i.e. `inventories/ams1/group_vars`.
- if a variable is used in several groups, then it is added to the `all` group

---

10. 9. Playbooks

- ### 10.1. agent.yml

> Playbook to deploy some specific agents for the system
>
> - `telegraf` to get metrics about the system
> - `filebeat` to consume logs and push it to the ELK
>
> ⚠️ **Specificities : InfluxData chose not to keep every version of `telegraf` in their repo, so sometimes the version used is not available anymore**

```
ansible-playbook -i inventories/DC_TO_DEPLOY/data-metrics.yaml
playbooks/agent.yml -D -vv
```

- ## 10.2. clickhouse.yml

  > Playbook to deploy ClickHouse DB.

  ```
  ansible-playbook -i inventories/DC_TO_DEPLOY/data-metrics.yaml
  playbooks/clickhouse.yml -D -vv
  ```

- ## 10.3. cron.yml

  > Playbook to deploy ad-hoc crons.

  ```
  ansible-playbook -i inventories/DC_TO_DEPLOY/data-metrics.yaml
  playbooks/cron.yml -D -vv
  ```

- ## 10.4. deploy-my-clickhouse.yml

  > Playbook to deploy the deploy-my-clickhouse project, that manages ClickHouse schema update.

  ```
  ansible-playbook -i inventories/DC_TO_DEPLOY/data-metrics.yaml
  playbooks/cron.yml -D -vv
  ```

- ## 10.5. elk.yml

  > Playbook to deploy an ELK:
  >
  > - `java` on the host
  > - `elasticsearch` on the host
  > - `logstash` on the host
  > - `kibana` on the host
  > - `curator` on the host
  >
  > It also deploys `filebeat` agents on the current log servers.

```
ansible-playbook -i inventories/DC_TO_DEPLOY/data-metrics.yaml
playbooks/elk.yml -D -vv
```

- ## 10.6. external-collector.yml

> Playbook to deploy the external collectors used in the system:
>
> - the `client-log-proxy` project on 2 hosts
> - `certbot` to generate and renew the `prod.log.{datacenter}.shadow,tech` cert
> - if datacenter in HA mode: `keepalived` to balance a public IP between 2 HAProxy
> - if datacenter in HA mode: `certctl` to synchronize certs generates via
> - 1 or 2 HAProxy exposed to Internet
>
> ⚠️ **Specificities :**
>
> - since it uses git to deploy the `client-log-proxy`, you need to forward your SSH agent using `--ssh-extra-args "-o ForwardAgent=yes"`
> - differentiation is made between DC with BGP and DC without BGP
> - iptables rules, routing tables, etc. are currently set **MANUALLY**, see the configuration of another similar datacenter to see how it's done (`/etc/iptables/rules.v4`)
> - not all datacenter are HA ready (only NY1 and USSFO03 are HA ready), so the `certctl` and `keepalived` should not be played on those datacenter

```
ansible-playbook -i inventories/DC_TO_DEPLOY/data-metrics.yaml --ssh-extra-
args "-o ForwardAgent=yes"  playbooks/external-collector.yml -D -vv
```

- ## 10.7. flattenizer.yml

> Playbook to deploy the `flattenizer` app with different configurations.

```
ansible-playbook -i inventories/DC_TO_DEPLOY/data-metrics.yaml
playbooks/flattenizer.yml -D -vv
```

- ## 10.8. influxdb.yml

> Playbook to deploy an InfluxDB.
>
> ⚠️ **Specificities : InfluxData chose not to keep every version of `influxdb` in their repo, so sometimes the version used is not available anymore**

```
ansible-playbook -i inventories/DC_TO_DEPLOY/data-metrics.yaml
playbooks/influxdb.yml -D -vv
```

- ## 10.9. internal-collector.yml

> Playbook to deploy the internal collectors of the system:
>
> - the `shadow-telegraf` project
> - `keepalived` to assign the VIP and perform the load balancing between `shadow-telegraf` processes
>
> It also deploys `filebeat` agents on the current log servers.
>
> ⚠️ **Specificities :**
>
> - an "all facts gathering" is needed to make sure all involved hosts' facts are set
> - differentiation is made between DC with BGP and DC without BGP

```
ansible-playbook -i inventories/DC_TO_DEPLOY/data-metrics.yaml
playbooks/internal-collector.yml -D -vv
```

- ## 10.10. job.yml

> Playbook to deploy some one-shot jobs that may be useful.

```
ansible-playbook -i inventories/DC_TO_DEPLOY/data-metrics.yaml
playbooks/job.yml -D -vv
```

- ## 10.11. kafka.yml

> Playbook to deploy:
>
> - a Kafka cluster (Zookeeper + Kafka)
> - required topics
>
> ⚠️ **Specificities :**
>
> - an "all facts gathering" is needed to make sure all involved hosts' facts are set
> - the topic parts is delegated to localhost, because there is no need doing it from remote hosts

```
ansible-playbook -i inventories/DC_TO_DEPLOY/data-metrics.yaml
playbooks/job.yml -D -vv
```

- ## 10.12. scraper.yml

> Playbook to deploy some in-house projects used to scrape some APIs.

```
ansible-playbook -i inventories/DC_TO_DEPLOY/data-metrics.yaml
playbooks/job.yml -D -vv
```

- ## 10.13. stream-metrics.yml

> Playbook to deploy the `stream-metrics` project that contains several workers.
>
> ⚠️ **Specificities :**
>
> - an "all facts gathering" is needed to make sure all involved hosts' facts are set
> - only containers are deployed, so some plays are duplicated to be able to deploy the same worker several times on the same host without conflicts (container name, path to local data, etc.)

```
ansible-playbook -i inventories/DC_TO_DEPLOY/data-metrics.yaml
playbooks/stream-metrics.yml -D -vv
```

- ## 10.14. system.yml

> Playbook to deploy to set the DNS configuration. This should not be used anymore.

```
ansible-playbook -i inventories/DC_TO_DEPLOY/data-metrics.yaml
playbooks/system.yml -D -vv
```

- ## 10.15. worker.yml

> - `telegraf` to consume data from Kafka and insert it into InfluxDB, it also performs some processing to optimize InfluxDB series cardinality
> - `s3-worker` to consume data from Kafka and put it into a S3 object storage for long term needs
> - `alerter` to consume data from Kafka and flags Shadows that seems to be abusing the service (long-time used, high GPU or CPU usage, etc.)
>
> ⚠️ **Specificities : some datacenters have a 2nd `telegraf` worker deployed to handle the load**

```
ansible-playbook -i inventories/DC_TO_DEPLOY/data-metrics.yaml
playbooks/worker.yml -D -vv
```
