Haute disponibilité de la base de métrologie (Graphite) avec keepalived

Sommaire

Introduction

La base de métrologie de Shinken peut être fortement sollicitée sur des architectures supervisant un grand nombre d'éléments. Sa disponibilité est également importante puisqu'elle peut être source de données pour des outils externes à Shinken (*Grafana par exemple*) pouvant être mis à disposition.

L'objectif de cette page de documentation est d'expliquer de manière détaillée la procédure nécessaire pour la mise en place d'un cluster Graphite.

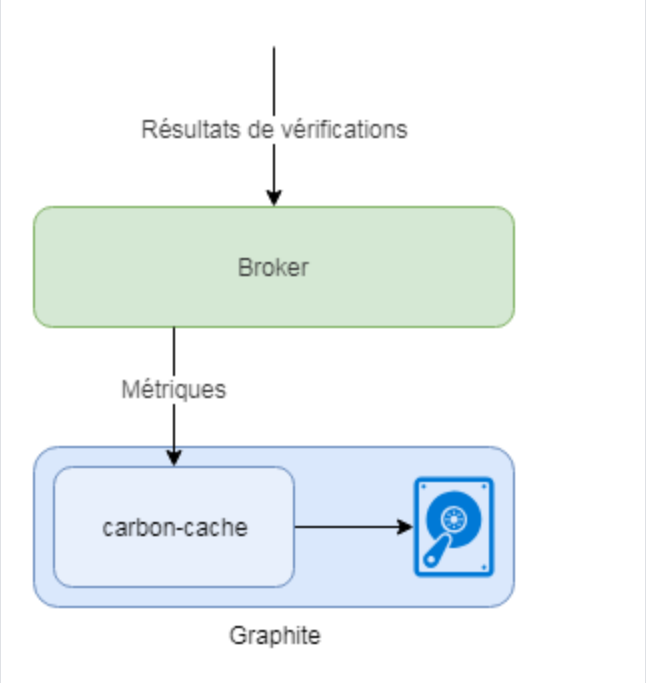
Cela permet d'augmenter la disponibilité des données et d'éviter la perte de données en cas d'incident sur la machine stockant les métriques.

Architecture mise en place

Dans une installation Shinken classique, les métriques sont stockées dans une base de données Graphite par le Broker.

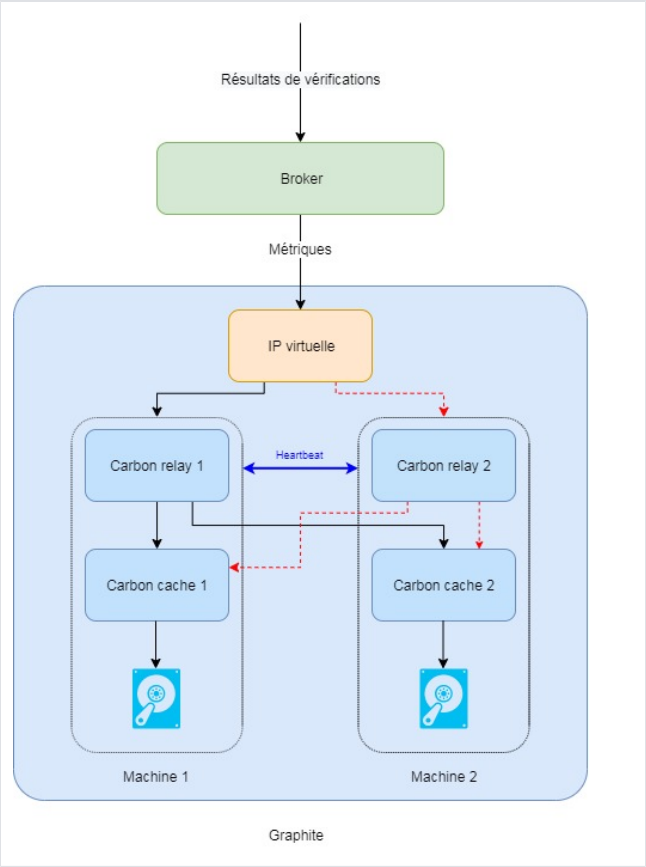
Chaque résultat de vérification contenant des données de métrologie sont analysées par le Broker. Ces métriques sont enregistrées dans Graphite par le Broker grâce au module Graphite-Perfdata.

Dans Graphite, le nom du démon responsable de l'enregistrement des métriques sur le disque est "carbon-cache".



Pour transformer cette architecture pour la rendre hautement disponible, on veut faire en sorte que les données soient répliquées lors de leur écriture. On utilise alors le logiciel Keepalived qui permet de créer une adresse IP virtuelle qui, en cas de problème sur le serveur principal, bascule sur le serveur secondaire automatiquement. Grâce à l'installation du démon "carbon-relay" sur le serveur principale ET le serveur secondaire, l'enregistrement des métriques n'est pas interrompu en cas de bascule.

L'architecture qu'on obtient à la fin de la procédure décrite dans cette documentation correspond à celle présente sur le schéma ci-contre.



Notre cluster Graphite est constitué d'au moins 2 machines distinctes (*id éalement 3 pour répartir les performances*).

- Le démon "*carbon-cache*" est sur toutes machine du cluster, ce qui permettra le stockage des métriques.
- Le démon "carbon-relay" est aussi sur toutes les machines du cluster, ce qui permettra en cas de bascule d'assurer la continuité de l'enregistrement des métriques.
- Sur la machine possédant la VIP, le démon "*carbon-relay*" reçoit l'ensemble des métriques envoyées par le Broker.
  Il se charge ensuite d'envoyer les ordres d'écritures des métriques aux démons "*carbon-cache*".

⚠ Le système mis en place ici est de la réplication, et non de la répartition.

Les démons "*carbon-cache*" stockent chacun l'ensemble des métriques envoyés par le Broker. Il faut donc que chaque machine qui héberge un démon "*carbon-cache*" soit capable d'enregistrer l'intégralité des métriques gérées par le Broker.

⚠ **Pour que le cluster Graphite fonctionne correctement, il est nécessaire de l'installer sur des machines utilisant le même système d'exploitation.**

Procédure de configuration

Avant de commencer, voici un résumé des différentes étapes nécessaires pour la configuration du cluster Graphite:

- Installation de Shinken
- Installation et configuration de Keepalived
- Mise en place du démon carbon-relay
- Autorisation des connexions à Graphite
- Modification de la configuration Shinken pour l'utilisation du cluster Graphite
- Redémarrage de Graphite et Shinken

Installation de Shinken

La première étape dans la mise en place d'un cluster Graphite est bien entendu l'installation de Graphite. Comme les autres dépendances de Shinken, il faut que la version de Graphite installée soit celle fournie avec l'installeur Shinken. En effet, certaines modifications ont été faites sur Graphite pour l'intégration avec Shinken, ce qui rend incompatible les versions de Graphites qui auraient pu être installées auparavant via d'autres installeurs.

Sur chaque machine qui compose le cluster Graphite, il faudra alors procéder à une installation de Shinken.

Pour tous les détails de la procédure d'installation de Shinken (*voir la page* Guide d'installation et de mise à jour).

Installation et configuration de Keepalived

Sur chaque carbon-relay

Installer Keepalived et ses dépendances sur tous les serveurs qui vont composer le cluster:

```
yum install keepalived
```

Une fois l'installation terminé, il faut aller éditer le fichier de configuration de Keepalived

/etc/keepalived/keepalived.conf

```
! Configuration File for keepalived

vrrp_instance NOM_DU_SERVEUR {
# Priorité utilisée lors de l'élection d'un nouveau maître. C'est la priorité la plus élevée qui gagne
l'élection
  priority 150
# ID du routeur virtuel. Doit être unique sur l'ensemble des noeuds
  virtual_router_id 255
# Option qui a pour conséquence qu'un master qui renaît ne reprendra pas l'IP si son SLAVE l'a
  nopreempt
# Interface d'écoute et d'échange des paquets multicast VRRP
  interface enp0s3
# L'état de santé des cibles est vérifié toutes les 5 secondes
  advert_int 5
# Interface réseau commune aux membres
  virtual_ipaddress {
        XXX.XXX.XXX.XXX/XX brd XXX.XXX.XXX.XXX dev enp0s3 scope global
  }
}
```

ATTENTION à modifier les informations suivantes :

- vrrp_instance NOM_DU_SERVEUR : Le nom associé à l'adresse virtuelle
- priority 150 : 150 pour le serveur principal et une valeur inférieur pour les secondaires
- interface enp0s3 : Le nom de l'interface réseau principale du serveur (*commande "ip addr" ou "ifconfig"*)
- virtual_ipaddress XXX.XXX.XXX.XXX/XX :
    - En 1er l'adresse virtuelle qui va être créée.
    - En 2ème le masque de sous réseau.
    - En 3ème l'adresse de broadcast du sous-réseau (*se termine en général par 255*).
    - En 4ème le nom de l'interface réseau du serveur.

Exemple d'un configuration avec 1 serveur principal et 1 serveur secondaire

MASTER : /etc/keepalived/keepalived.conf

```
! Configuration File for keepalived

vrrp_instance graphite-relay {
  priority 150
  virtual_router_id 255
  nopreempt
  interface enp0s3
  advert_int 5
  virtual_ipaddress {
        192.168.1.100/24 brd 192.168.1.255 dev
enp0s3 scope global
  }
}
```

SLAVE : /etc/keepalived/keepalived.conf

```
! Configuration File for keepalived

vrrp_instance graphite-relay {
  priority 100
  virtual_router_id 255
  nopreempt
  interface enp0s3
  advert_int 5
  virtual_ipaddress {
        192.168.1.100/24 brd 192.168.1.255 dev
enp0s3 scope global
  }
}
```

Redémarrer Keepalived sur tous les serveurs du cluster en commençant par la maître :

```
systemctl restart keepalived
```

Ainsi la commande "ip addr" sur le serveur maître devrait vous renvoyer l'ip courante et l'ip virtuelle précédemment créée :

```
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:20:4a:2f brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.68/24 brd 192.168.1.255 scope global noprefixroute dynamic enp0s3
       valid_lft 75921sec preferred_lft 75921sec
    inet 192.168.1.100/24 brd 192.168.1.255 scope global secondary enp0s3
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe20:4a2f/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

Mise en place du démon carbon-relay

Une fois Shinken, Graphite et Keepalived installés, il faut procéder à la configuration de Graphite.

On commence par dire à Graphite qu'on utilise la fonctionnalité de relay via le fichier **/opt/graphite/conf/carbon.conf** (*copier le fichier d'exemple carbon.conf.example si besoin*).

Sur chaque carbon-relay

Dans la section "*[relay]*", on modifie la variable "*DESTINATIONS*" dans laquelle on décrit la liste des démons **carbon-relay** qui constituent le cluster Graphite.

/opt/graphite/conf/carbon.conf

```
[relay]
...
<autres options>
...


DESTINATIONS = adresse_carbon_cache_1:2004, adresse_carbon_cache_2:2004
```

On configure ensuite le relay pour lui spécifier les adresses des carbon-cache pour l'écriture et les adresses à utiliser pour la lecture des métriques.

Sur chaque carbon-relay

Cette configuration se fait via le fichier **/opt/graphite/conf/relay-rules.conf** (*copier le fichier d'exemple relay-rules.conf.example si besoin*):

```
[default]
default = true
destinations = adresse_carbon_cache_1:2004,adresse_carbon_cache_2:2004
read_destinations = adresse_carbon_cache_1:80,adresse_carbon_cache_2:80
```

Par défaut, l'installeur Shinken met en place le fichier d'initialisation pour le démon carbon-relay mais ne lui attribue pas les droits d'exécution. Pour pouvoir le démarrer en tant que service du système comme les autres démons, il va falloir lui donner le droit de s'exécuter.
Pour cela, lancer la commande suivant sur la machine qui va héberger le démon carbon-relay:

Donner les droits d'execution

```
chmod +x /etc/init.d/carbon-relay
```

A la fin de cette étape de configuration, le relais est correctement configuré en ce qui concerne l'écriture des données.

Pour la lecture des métriques, il faut encore autoriser la lecture des métriques, ce qui est l'objectif de l'étape suivante.

Autorisation des connexions à Graphite

Par défaut, Graphite n'autorise la lecture des métriques seulement depuis une interface locale, pour des raisons de sécurité. Il faut alors sur chaque machine qui héberge un démon carbon-relay, modifier les réglages d'Apache pour autoriser la lecture sur des machines externes.

Sur chaque carbon-relay, on trouve dans le fichier **/etc/httpd/conf.d/graphite.conf** la ligne suivante:

/etc/httpd/conf.d/graphite.conf

```
  <VirtualHost 127.0.0.1:80>
```

Cette ligne permet de dire à Apache qu'il faut écouter les requêtes de lecture des métriques sur l'interface locale (*127.0.0.1*) et le port 80 seulement.

Pour écouter ces requêtes sur toutes les interfaces réseau de la machine, on remplacera cette ligne par la suivante:

/etc/httpd/conf.d/graphite.conf

```
  <VirtualHost *:80>
```

On redémarre ensuite Apache pour prendre en compte les modifications:

```
systemctl restart httpd
```

Modification de la configuration Shinken pour l'utilisation du cluster Graphite

Comme vu dans la section présentant l'architecture haute disponibilité qu'on est en train de mettre en place, le Broker, et en particulier son module Graphite-Perfdata, doit maintenant envoyer ses informations au démon carbon-relay au lieu de les envoyer directement à un démon carbon-cache.

Il faut donc modifier la configuration actuelle de Shinken pour envoyer les métriques sur le démon carbon-relay plutôt qu'au démon carbon-cache.

Pour cela, on modifie la configuration du module Graphite-Perfdata dans le fichier **/etc/shinken/modules/graphite.cfg**:

```
   #   Metrology server parameters   #

   #  Graphite writer ( carbon ) address
   #  IP address or FQDN of carbon-cache or carbon-relay instance to send metrics to
   #
   #           Default : localhost
   #
   broker__module_graphite_perfdata__writer__host      addresse_carbon_relay

   #  Graphite writer ( carbon ) port
   #  tcp port of carbon-cache or carbon-relay instance to send metrics to
   #
   #           Default : 2003
   #
   broker__module_graphite_perfdata__writer__port      2013
```

On change ici le port 2003 (*port par défaut du carbon-cache*) vers le port 2013 (*port par défaut du carbon-relay*).

Il faut également modifier les réglages de Shinken pour lui spécifier d'envoyer les requêtes de lecture des métriques sur le carbon-relay au lieu de les envoyer directement sur un démon carbon-cache.

Cette modification se fait directement dans les réglages du module WebUI, responsable de l'interface de Visualisation. Dans **/etc/shinken/modules /webui.cfg**, on modifie la section suivante:

/etc/shinken/modules/webui.cfg

```
[...]


#======== Metrology access =========
# Multi-realm graphite parameter
graphite_backends        *:addresse_ip_virtuelle


[...]
```

Redémarrage de Graphite et Shinken

A ce stade, la configuration de Graphite a été modifiée pour utiliser un démon carbon-relay. On a aussi effectué des modifications de configuration au niveau de Shinken pour prendre en compte ce changement d'architecture au niveau de Graphite.

La dernière étape est de redémarrer les différents démons pour prendre en compte ces changements de configuration.

- On commence par stopper le démon carbon-cache sur le serveur shinken:

  ```
  service carbon-cache stop
  ```

- On démarrer le démon carbon-relay sur tous les serveurs composants le cluster:

  ```
  service carbon-cache start
  ```

- On redémarrer le démon carbon-cache sur tous les serveurs composants le cluster:

  ```
  service carbon-cache restart
  ```

- On redémarre ensuite l'Arbiter pour prendre en compte les modifications de configuration de Shinken:

  ```
  service shinken-arbiter restart
  ```

Vérification du firewall (*est ce que vous avez accès*)

Si vous n'arrivez pas à vous connecter au carbon-cache / carbon-relay vérifier que le port est ouvert dans votre firewall.

Si vous avez firewalld (*firewall par défaut de la Redhat / AlmaLinux*)

Vérifier que les ports sont ouvert

Voici comment vérifier que les ports des carbon-relay / carbon-cache sont ouvert sur leurs machines :

```
firewall-cmd --list-ports
```

Exemple de résultat

```
80/tcp 7763/tcp 7765/tcp 7766/tcp 7767/tcp 7768/tcp 7769/tcp 7770/tcp 7771/tcp 7772/tcp 7773/tcp 7777/tcp
7780/tcp 50000/tcp
```

Dans cet exemple, aucun des ports de carbon-relay (*2013/tcp*) et carbon-cache (*2004/tcp*) ne sont présents, ils sont donc bloqués.

Sur les serveurs carbon-relay

Si le port du **carbon-relay** n'est pas ouvert, il faut alors autoriser le trafic vers ce port.

Cela peut être fait avec les commandes suivantes :

```
firewall-cmd --add-port=2013/tcp
firewall-cmd --runtime-to-permanent
```

Sur les serveurs carbon-cache

Si le port du **carbon-cache** n'est pas ouvert, il faut alors autoriser le trafic vers ce port.

Cela peut être fait avec les commandes suivantes :

```
firewall-cmd --add-port=2004/tcp
firewall-cmd --runtime-to-permanent
```

Sur les serveurs keepalived

Vérifier les autorisations sur firewalld

Pour vérifier que les démons **keepalived** puissent communiquer, lancer la commande suivante

```
firewall-cmd --list-all
```

La ligne **rich rules** indiquera si le protocol "vrrp" est autorisé.

Voici deux exemples de sorties,

- sur le premier l'autorisation est en place

Autorisation keepalived présente

```
public (active)
  target: default
  icmp-block-inversion: no
  interfaces: enp0s3 enp0s8
  sources:
  services: cockpit dhcpv6-client ssh
  ports: 80/tcp 7765/tcp 7766/tcp 7767/tcp 7768/tcp 7769/tcp 7770/tcp 7771/tcp 7772/tcp 7773/tcp 7777/tcp
7780/tcp 50000/tcp 3000/tcp
  protocols:
  forward: no
  masquerade: no
  forward-ports:
  source-ports:
  icmp-blocks:
  rich rules:
        rule protocol value="vrrp" accept
```

- sur le deuxième, elle ne l'est pas :

Autorisation keepalived absente

```
public (active)
  target: default
  icmp-block-inversion: no
  interfaces: enp0s3 enp0s8
  sources:
  services: cockpit dhcpv6-client ssh
  ports: 80/tcp 7765/tcp 7766/tcp 7767/tcp 7768/tcp 7769/tcp 7770/tcp 7771/tcp 7772/tcp 7773/tcp 7777/tcp
7780/tcp 50000/tcp 3000/tcp
  protocols:
  forward: no
  masquerade: no
  forward-ports:
  source-ports:
  icmp-blocks:
  rich rules:
```

Autoriser les démons keepalived à communiquer sur firewalld

Si le protocole n'est pas autorisé par le firewall, il faut alors autoriser le trafic pour **keepalived.**

Cela peut être fait avec les commandes suivantes :

```
firewall-cmd --add-rich-rule='rule protocol value="vrrp" accept' --permanent
firewall-cmd --reload
```

Comportement de Shinken avec un cluster Graphite

La haute disponibilité de Graphite est donc gérée entièrement par Keepalived, puis Graphite s'occupe de la réplication des données entre les démons carbon-cache et de la lecture des données.

Cette modification d'architecture est invisible pour les utilisateurs.

Outils externes (*Grafana*) et relation UUID nom

En cas d'utilisation d'outils externes (*comme Grafana par exemple*) pour consulter les métriques, il faut également

- configurer la récupération des données de l'inventaire sur les serveurs carbon-cache
- autoriser les connexions au serveur d'inventaire depuis les serveurs carbon-cache

(*voir la page* Base de métrologie (Graphite) *section* Correspondance ID Nom de l'élément)