Agent Telegraf GCP

Objectifs

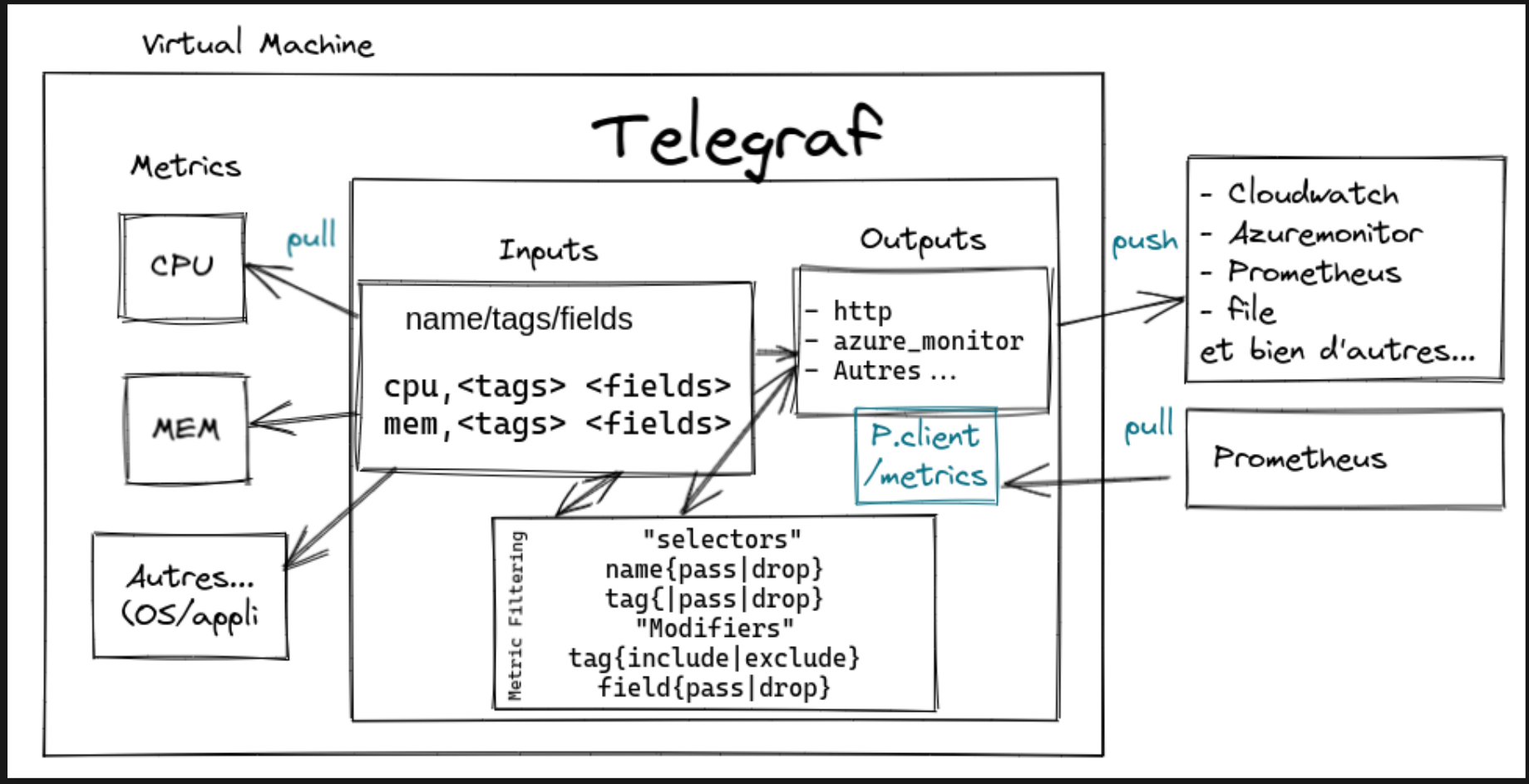
- Découvrir Telegraf
- Tester l’agent Telegraf pour envoyer les métriques dans :
  - Cloud monitoring
  - Prometheus
- Comparer l’agent natif Ops agent et Telegraf

Prérequis

- Création des VMs sur un projet GCP
- Installation de telegraf
- Autorisations nécessaires à la VM pour communiquer avec le CSP:
- https://www.googleapis.com/auth/logging.write
- https://www.googleapis.com/auth/monitoring.write

Présentation en vidéo sur l’installation de telegraf, disponible ICI

Concepts de base Telegraf



Authentification pour envoyer dans cloud monitoring

Deux méthodes:

- Service account généré par défaut lors de la creation de la VM
- Création d’un user avec les roles nécessaires

Côté Telegraf, pas de configuration spécifique à faire

Métriques dans cloud monitoring (1/3)

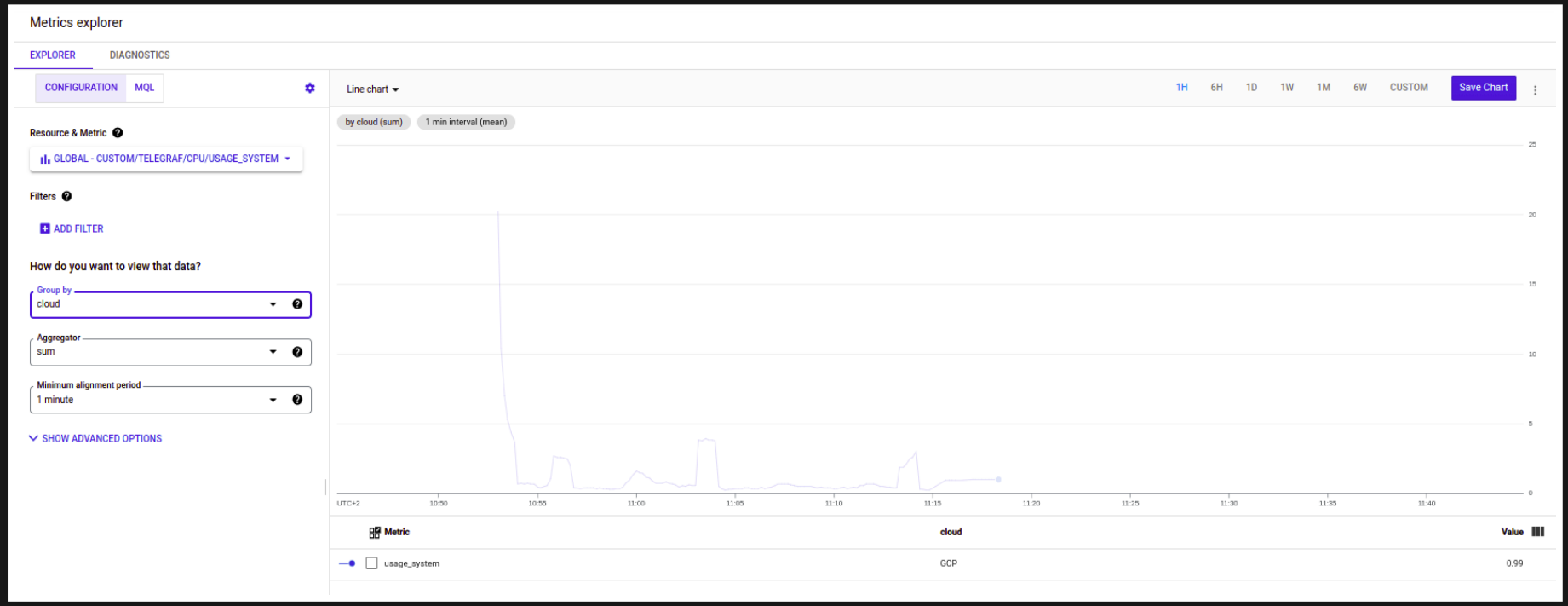
- Le plugin de type output `stackdriver` envoie les métriques vers cloud monitoring
- Seul le champ project_id est obligatoire
- Les métriques remontées seront considérées comme custom
- Les métriques seront dans le sous dossier custom

Métriques dans cloud monitoring (2/3)

Exemple de configuration

```
# # Configuration for Google Cloud Stackdriver to send metrics to
[[outputs.stackdriver]]
#   ## GCP Project
   project = "fluted-airline-347207"
#
#   ## The namespace for the metric descriptor
   namespace = "telegraf"
#
#   ## Custom resource type
#   # resource_type = "generic_node"
#
#   ## Additional resource labels
#   # [outputs.stackdriver.resource_labels]
#   #   node_id = "$HOSTNAME"
```

Métriques dans Cloud monitoring (3/3)



Métriques dans Prometheus (push) (1/3)

Le plugin de type output `http` envoie les données dans un message http, dans un format `prometheusremotewrite`. Le header `Authorization` associé permet l'authentification dans Rancher.

```
[[outputs.http]]
  alias = "prometheus-0"
  url = "https://rancher.ocb-test06[...]/prometheus-0[...]/v1/write"
  data_format = "prometheusremotewrite"

  [outputs.http.headers]
        Authorization = "Bearer xxx"
        Content-Type = "application/x-protobuf"
        Content-Encoding = "snappy"
        X-Prometheus-Remote-Write-Version = "0.1.0"
```

Métriques dans Prometheus (push) (2/2)

Time	__name__	cc_client	cc_prom	cloud	device	fstype	host	mode	path	prometheus	Value #A
2022-05-18 11:49:0...	disk_free	ocb-test06	cloud-app	GCP	sda1	ext4	demo-pf-918	rw	/	monitoring-app/app...	5414674432
2022-05-18 11:49:0...	disk_free	ocb-test06	cloud-app	GCP	sda15	vfat	demo-pf-918	rw	/boot/efi	monitoring-app/app...	103965696

Métriques dans Prometheus (pull)

Le plugin de type output `prometheus_client` expose les métriques sur un endpoint http (par défaut : `/metrics`). Un serveur Prometheus va ensuite récupérer les données par pulling.

```
# Configuration for the Prometheus client to spawn
[[outputs.prometheus_client]]
  ## Address to listen on.
  listen = ":9273"
```

Focus sur le plugin processor

Permet de réaliser des opérations sur les métriques définies en input.
Parmi ces opérations on a:

- Transformation
- Décoration
- Filtrage

exemple 1:

```
[[processors.clone]]
namepass = [ "cpu" ]
[processors.clone.tags]
 instance = "gcp-instance"
```

Focus sur le plugin processor

exemple 2:

```
[[processors.regex]]
     [[processors.regex.tags]]
       key = "host"
       pattern = "^(.*)$"
       replacement = "${1}"
       result_key = "instance"
```

Monitoring de Telegraf : output health (1/2)

Permet d’avoir un healtcheck http
Différents tests peuvent être réalisés :

- `compares` : ça compare la valeur d’un field à une valeur en dur
- `contains` : ça vérifie que la métrique contient bien un field

Si :

- échec : code retour 503
- succès : code retour 200

Monitoring de Telegraf : output health (2/2)

Exemple de configuration

```
[[outputs.health]]
  service_address = "http://localhost:8080"

  namepass = ["internal_agent", "internal_write"]

  [[outputs.health.compares]]
     field = "gather_errors"
     lt = 5.0

  [[outputs.health.contains]]
     field = "buffer_size"
```

Monitoring de Telegraf : input internal (1/2)

Permet d’avoir des métriques sur Telegraf lui-même :

- le nombre de métriques collectées
- le nombre de métriques écrites
- la taille limite du buffer
- la taille du buffer
- des stats sur la mémoire
- ...

Monitoring de Telegraf : input internal (2/2)

- Configuration :

```
# Collect statistics about itself
[[inputs.internal]]
## If true, collect telegraf memory stats.
# collect_memstats = true
```

- Dashboard upstream disponible

Monitoring de Telegraf : mix des deux solutions

Les deux plugins utilisés séparément ne permettent pas de valider la chaine de bout en bout.
On pourrait utiliser les deux plugins pour monitorer Telegraf.

- internal pour récupérer les métriques internes de Telegraf
- health pour vérifier que le plugin internal fonctionne comme attendu

Problème connu avec telegraf (1/2)

Description

- des métriques en out-of-order (quand le buffer telegraf est rempli)

```
E! [agent] Error writing to outputs.stackdriver: rpc error: code = Inva
```

Problème connu avec telegraf (2/2)

Description

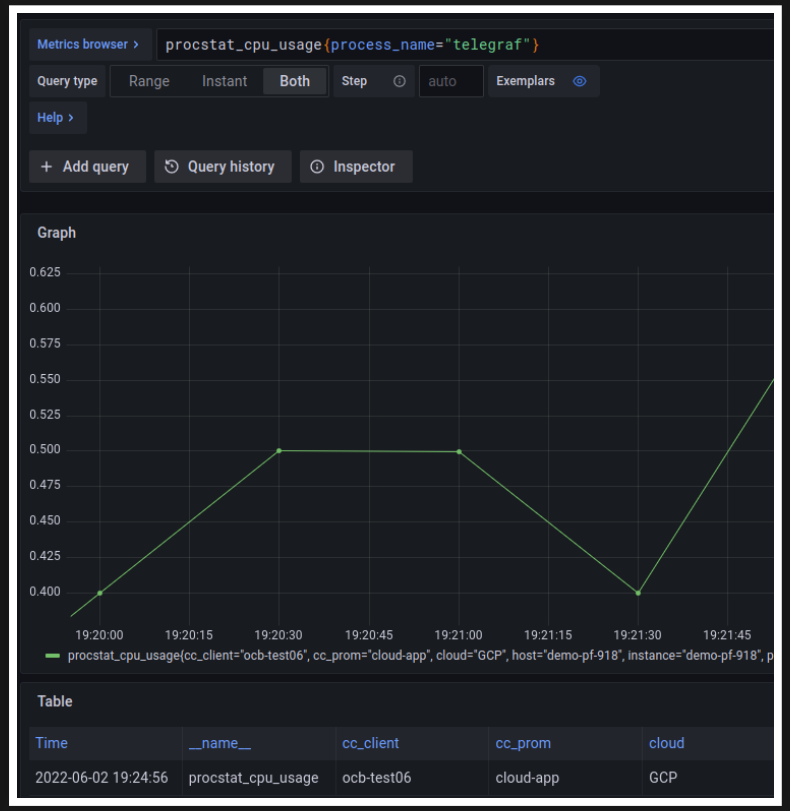
- Toutes les métriques remontées par telegraf sont considérées comme custom
- Les métriques se retrouvent dans le sous dossier custom
- Une limite de 10000 métriques custom est imposée par google

Limitations

- la limite imposée sur les custom metrics
- Supporte pas des valeurs de type string dans les métriques
- Des logs d’erreur sur “out of order”.

Ressources utilisées (CPU/RAM)

- usage mémoire = 14% ==> 250 MB
- usage cpu = 0,6%



Comparaison entre Telegraf et l'Ops agent

	Ops agent	Telegraf
Linux	⊕	⊕
Windows	⊕	⊕
Empreinte CPU	faible : 2% 1 CPU (linux)	faible
Empreinte mémoire	faible : 204 MB (linux)	faible
Outside GCP	AWS	⊕
Facilité de configuration / installation	⊕	⊕
Etendue de supervision	⊕	⊕
Filtrage des métriques	⊕	⊕
Configuration flexible : tags custom, transformation des métriques, suppression de tags ...	⊖	⊕
Métriques OS	⊕	⊕
Métriques applicatives	⊕	⊕
Configuration échantillonnage métriques	⊕	⊕
Documentation	⊕	⊕
limitations	pas de custom métriques avec opentelemtry collector	perte des métriques out of order, toutes les métriques qont considérées comme custom