Mise en place d'honeypot

1. Objectif du projet

L'objectif de ce projet est de **déployer un honeypot SSH** sur un serveur cloud (AWS) afin de :

- observer des **attaques réelles** en conditions réelles,
- analyser les **tentatives de connexion**,
- étudier les **commandes exécutées par les attaquants**,
- comprendre les **comportements post-connexion**.

Le honeypot utilisé est **Cowrie**, un honeypot SSH/Telnet largement utilisé dans la recherche en cybersécurité.

2. Environnement

- Fournisseur cloud : **AWS (EC2)**
- OS : **Debian**
- Honeypot : **Cowrie**
- Port SSH réel : `2222`
- Port honeypot Cowrie : `2223`
- Logs : **JSON (format natif Cowrie)**

3. Installation des prérequis

Mise à jour du système

```
apt-get update && apt-get upgrade -y
```

Installation des paquets nécessaires

```
apt-get install -y \
git sshpass jq tcpdump wireshark curl \
python3 python3-venv python3-dev libpython3-dev \
libssl-dev libffi-dev build-essential
```

4. Installation de Cowrie

Création de l'utilisateur dédié

```
useradd -m cowrie
```

Connexion avec l'utilisateur `cowrie`

```
sudo -u cowrie bash
cd ~
```

Clonage du dépôt Cowrie

```
git clone https://github.com/cowrie/cowrie.git
cd cowrie
```

---

5. Création de l'environnement virtuel Python

```
python3 -m venv cowrie-env
source cowrie-env/bin/activate
```

Mise à jour de pip et installation des dépendances

```
python -m pip install --upgrade pip setuptools wheel
pip install -r requirements.txt
```

Installation de Cowrie en mode editable

```
pip install -e .
```

---

6. Configuration de Cowrie

Création du fichier de configuration

```
cp etc/cowrie.cfg.dist etc/cowrie.cfg
```

Configuration du port SSH honeypot

Dans `etc/cowrie.cfg` :

```
[ssh]
listen_endpoints = tcp:2223:interface=0.0.0.0
```

Activation des logs JSON

```
[output_jsonlog]
enabled = true
```

👉 Cowrie n'utilise ici **que des logs JSON**, ce qui est le format recommandé pour l'analyse et l'intégration SIEM.

---

7. Lancement de Cowrie

⚠️ Cowrie est basé sur **Twisted**, il ne se lance pas comme un script Python classique.

Lancement en mode interactif (recommandé)

```
twistd -n cowrie
```

Alternative équivalente :

```
python3 -m twisted.scripts.twistd -n cowrie
```

Vérification de l'écoute

```
ss -lntp | grep 2223
```

---

8. Test de connexion SSH

Depuis la machine locale ou depuis la VM :

```
ssh -p 2223 root@IP_DU_SERVEUR
```

👉 La connexion fonctionne, mais :

- il s’agit d’un **shell totalement émulé**,
- aucune commande n’est exécutée sur le système réel,
- aucune connexion réseau sortante réelle n’est effectuée.

---

9. Fonctionnement du honeypot

Cowrie :

- **émule une fausse connexion SSH**,
- accepte volontairement des identifiants faibles,
- simule un système Linux crédible,
- intercepte et journalise **toutes les commandes**.

Il **ne s’agit pas du vrai serveur SMTP ou du système réel** :
l’attaquant est enfermé dans un environnement simulé.

---

10. Analyse des logs Cowrie (JSON)

Emplacement des logs

```
cd /home/cowrie/cowrie/var/log/cowrie
```

Fichier principal :

```
cowrie.json
```

Commandes exécutées par les attaquants

```
jq 'select(.eventid=="cowrie.command.input") | .input' cowrie.json
```

Exemples observés :

```
echo "test"
ssh root@smtp.agrepe.com
ls
ls /
cd /etc
```

Logins réussis

```
jq 'select(.eventid=="cowrie.login.success") |
{user:.username, pass:.password, ip:.src_ip}' cowrie.json
```

Exemple :

```
{
  "user": "root",
  "pass": "admin",
  "ip": "86.195.60.74"
}
```

Connexions entrantes

```
jq 'select(.eventid=="cowrie.session.connect") |
{ip:.src_ip, port:.src_port}' cowrie.json
```

Exemple :

```
{
  "ip": "86.195.60.74",
  "port": 48614
}
```

---

11. Redirection du port SSH (22 → 2223)

Pour attirer davantage d'attaques automatisées, le port SSH standard est redirigé vers Cowrie.

Redirection NAT avec iptables

```
iptables -t nat -A PREROUTING -p tcp --dport 22 -j REDIRECT --to-ports 2223
```

👉 Résultat :

- les bots ciblant le port 22 tombent automatiquement sur Cowrie,
- le vrai SSH reste accessible sur un port différent (ex: 2222).

12. Conclusion

Ce honeypot permet :

- d'observer des **attaques réelles en production**,
- d'analyser les **comportements post-connexion**,
- de collecter des données exploitables pour :
  - statistiques,
  - SIEM (ELK / OpenSearch),
  - recherche en cybersécurité.

Le projet constitue une **base solide** pour :

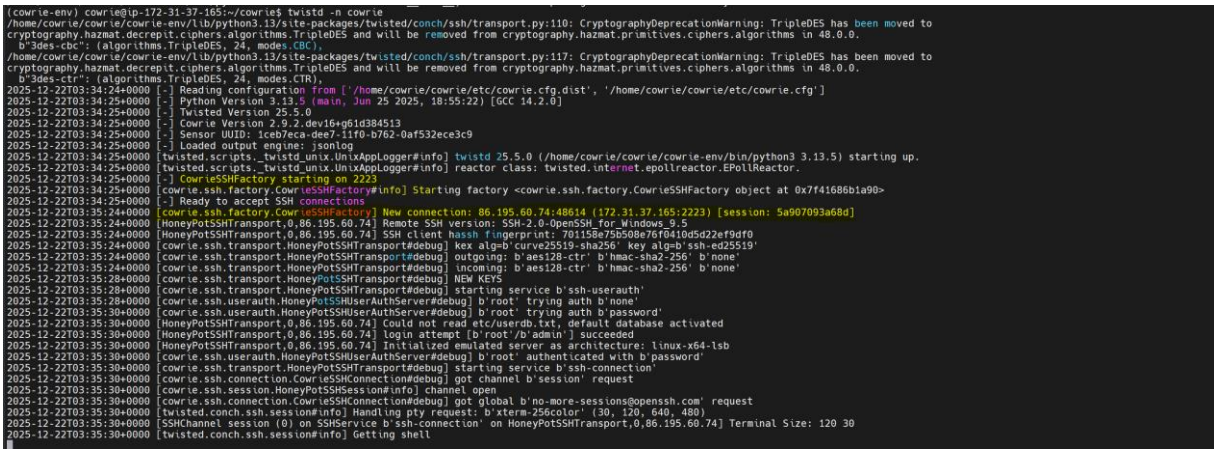
- enrichissement GeoIP / ASN,
- détection bot vs humain,
- corrélation avec d'autres honeypots (Dionaea),
- visualisation via ELK.

Screens

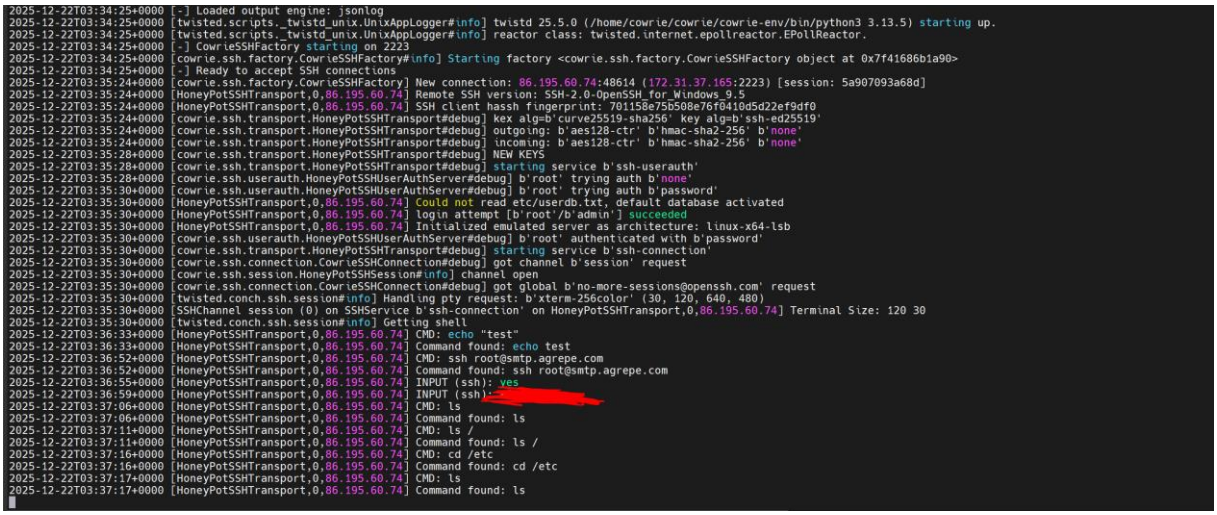
Dans cowrie.cfg

```
# JSON based logging module
#
[output_jsonlog]
enabled = true
logfile = ${honeypot:log_path}/cowrie.json
epoch_timestamp = false
```

Test connection ssh

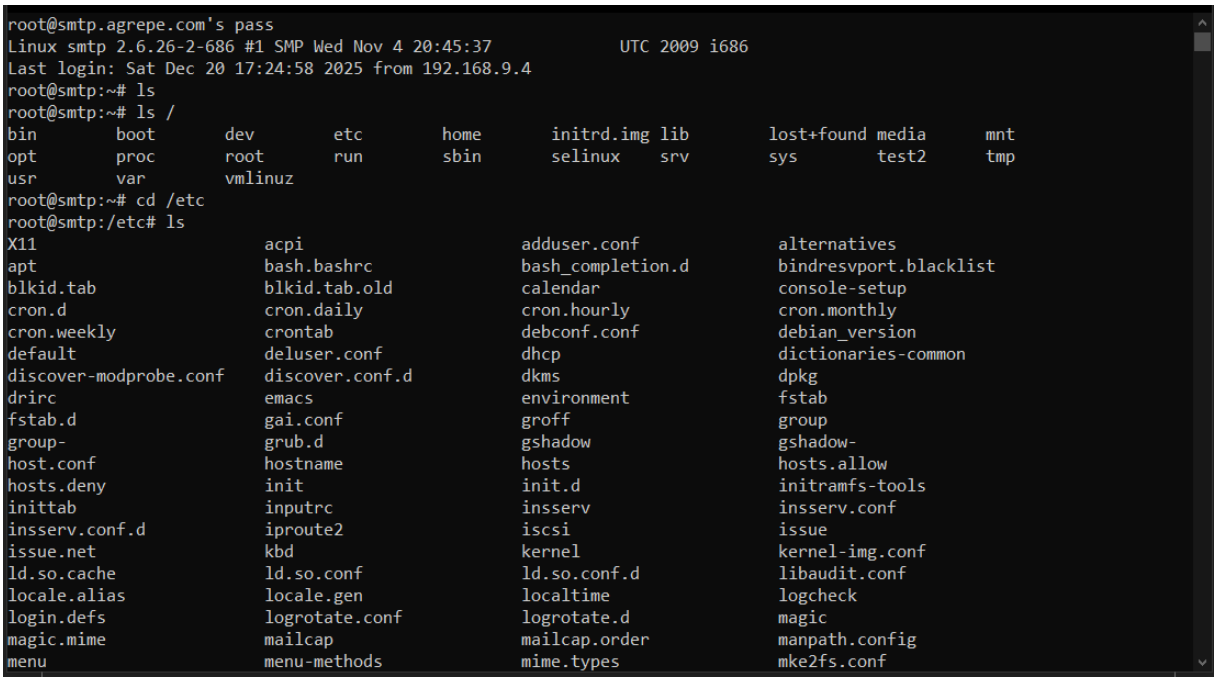


Je peux meme voir les commandes exécutés par les attaquants



Cowrie emule une fausse connection ssh

Ça ne ressemble pas du tout à mon serveur de base smtp



Ensuite pour voir l'historique des commandes tapés

D'abord

Cd /home/cowrie/cowrie/var/log/cowrie

Ensuite

jq 'select(.eventid=="cowrie.command.input") | .input' cowrie.json

Et pour login réussis

root@ip-172-31-37-165:/home/cowrie/cowrie/var/log/cowrie# jq 'select(.eventid=="cowrie.login.success") | {user:.username, pass:.password, ip:.src_ip}' cowrie.json

{

"user": "root",

"pass": "admin",

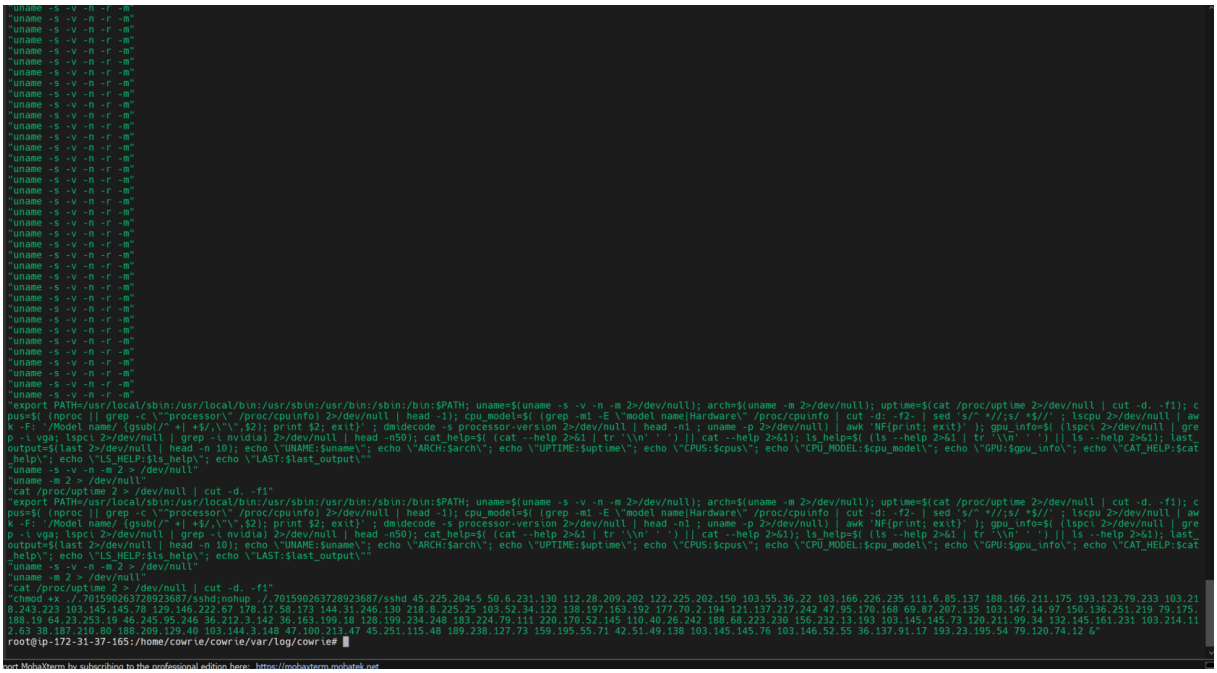
"ip": "86.195.60.74"

```
}
{
  "user": "root",
  "pass": "admin",
  "ip": "86.195.60.74"
}
```


Connexion entrante


```
root@ip-172-31-37-165:/home/cowrie/cowrie/var/log/cowrie# jq
'select(.eventid=="cowrie.session.connect") | {ip:.src_ip, port:.src_port}' cowrie.json
{
  "ip": "86.195.60.74",
  "port": 48614
}
{
  "ip": "86.195.60.74",
  "port": 62393
}
root@ip-172-31-37-165:/home/cowrie/cowrie/var/log/cowrie#
```

Apres 10-12h déjà des attaquants ont été piégés par le honeypot



Installation Grafana Loki avec Promtail (simple + dashboards)

On va juste essayer d'envoyer les logs pour les visualiser sur un dashboard propre etc

On commence par installer Loki

mkdir -p /opt/loki /var/lib/loki

cd /opt/loki

curl -L -o loki.zip https://github.com/grafana/loki/releases/latest/download/loki-linux-amd64.zip

unzip loki.zip

chmod +x loki-linux-amd64

Ensuite vérifier

./loki-linux-amd64 –version

Si je vois ça c’est bon

```
root@ip-172-31-37-165:/opt/loki# ./loki-linux-amd64 --version
loki, version 3.6.3 (branch: release-3.6.x, revision: 9385bc63)
  build user:       root@f7962624fd79
  build date:       2025-12-11T09:08:55Z
  go version:       go1.25.4
  platform:         linux/amd64
  tags:             netgo
root@ip-172-31-37-165:/opt/loki#
```

Il faut créer les dossiers de stockage maintenant

mkdir -p /var/lib/loki/{chunks,index,cache,wal}

Créer fichier de config

nano /opt/loki/loki.yaml

Et renseigner ses informations

```yaml
auth_enabled: false

server:
  http_listen_port: 3100
  grpc_listen_port: 0

common:
  path_prefix: /var/lib/loki
  storage:
    filesystem:
      chunks_directory: /var/lib/loki/chunks
      rules_directory: /var/lib/loki/rules
  replication_factor: 1
  ring:
    kvstore:
      store: inmemory

schema_config:
  configs:
    - from: 2023-01-01
      store: boltdb-shipper
      object_store: filesystem
      schema: v13
      index:
        prefix: index_
        period: 24h

limits_config:
  retention_period: 7d
  allow_structured_metadata: false
```

Ensuite lancer loki

/opt/loki/loki-linux-amd64 -config.file=/opt/loki/loki.yaml

Tester via curl si tout est ok

curl http://127.0.0.1:3100/ready

```
root@ip-172-31-37-165:/home/cowrie/cowrie/var/log/cowrie# curl http://127.0.0.1:3100/ready
Ingester not ready: waiting for 15s after being ready
root@ip-172-31-37-165:/home/cowrie/cowrie/var/log/cowrie# curl http://127.0.0.1:3100/ready
Ingester not ready: waiting for 15s after being ready
root@ip-172-31-37-165:/home/cowrie/cowrie/var/log/cowrie# curl http://127.0.0.1:3100/ready
ready
root@ip-172-31-37-165:/home/cowrie/cowrie/var/log/cowrie#
```

```
level=info ts=2025-12-23T03:05:14.715522332Z caller=ruler.go:536 msg="ruler up and running"
level=info ts=2025-12-23T03:05:14.715585424Z caller=basic_lifecycler.go:321 component=distributor msg="instance not found in the ring" instance=ip-172-31-37-165 ring=distributor
level=info ts=2025-12-23T03:05:14.715688708Z caller=ingester.go:581 component=ingester msg="recovered WAL checkpoint recovery finished" elapsed=966.481µs errors=false
level=info ts=2025-12-23T03:05:14.715696729Z caller=ingester.go:587 component=ingester msg="recovering from WAL"
level=info ts=2025-12-23T03:05:14.716282014Z caller=compactor.go:383 msg="waiting until compactor is JOINING in the ring"
level=info ts=2025-12-23T03:05:14.716295392Z caller=compactor.go:387 msg="compactor is JOINING in the ring"
level=info ts=2025-12-23T03:05:14.716351946Z caller=compactor.go:397 msg="waiting until compactor is ACTIVE in the ring"
level=info ts=2025-12-23T03:05:14.717753065Z caller=ingester.go:603 component=ingester msg="WAL segment recovery finished" elapsed=3.030018ms errors=false
level=info ts=2025-12-23T03:05:14.717770881Z caller=ingester.go:551 component=ingester msg="closing recoverer"
level=info ts=2025-12-23T03:05:14.717785706Z caller=ingester.go:559 component=ingester msg="WAL recovery finished" time=3.062233ms
level=info ts=2025-12-23T03:05:14.717866402Z caller=lifecycler.go:687 component=ingester msg="not loading tokens from file, tokens file path is empty"
level=info ts=2025-12-23T03:05:14.717917321Z caller=lifecycler.go:714 component=ingester msg="instance not found in ring, adding with no tokens" ring=ingester
level=info ts=2025-12-23T03:05:14.718556205Z caller=lifecycler.go:556 component=ingester msg="auto-joining cluster after timeout" ring=ingester
level=info ts=2025-12-23T03:05:14.718737777Z caller=wal.go:158 msg=started component=wal
level=info ts=2025-12-23T03:05:14.720063283Z caller=ingester.go:772 component=ingester msg="sleeping for initial delay before starting periodic flushing" delay=12.285035514s
level=info ts=2025-12-23T03:05:14.840777168Z caller=compactor.go:401 msg="compactor is ACTIVE in the ring"
level=info ts=2025-12-23T03:05:14.900501033Z caller=ringmanager.go:203 msg="scheduler is ACTIVE in the ring"
level=info ts=2025-12-23T03:05:14.900577214Z caller=module_service.go:82 msg=starting module=query-scheduler
level=info ts=2025-12-23T03:05:14.90068328Z caller=module_service.go:82 msg=starting module=query-frontend
level=info ts=2025-12-23T03:05:14.900706415Z caller=module_service.go:82 msg=starting module=querier
level=info ts=2025-12-23T03:05:14.900843707Z caller=loki.go:599 msg="Loki started" startup_time=300.749969ms
level=info ts=2025-12-23T03:05:17.900748603Z caller=scheduler.go:652 msg="this scheduler is in the ReplicationSet, will now accept requests."
level=info ts=2025-12-23T03:05:17.900823347Z caller=worker.go:235 component=querier msg="adding connection" addr=172.31.37.165:44029
level=info ts=2025-12-23T03:05:19.841433019Z caller=compactor.go:460 msg="this instance has been chosen to run the compactor, starting compactor"
level=info ts=2025-12-23T03:05:19.841518574Z caller=tables_manager.go:70 msg="waiting 10m0s for ring to stay stable and previous compactions to finish before starting compactor"
level=info ts=2025-12-23T03:05:24.901258356Z caller=frontend_scheduler_worker.go:106 msg="adding connection to scheduler" addr=172.31.37.165:44029
level=info ts=2025-12-23T03:05:44.721197088Z caller=recalculate_owned_streams.go:49 msg="starting recalculate owned streams job"
level=info ts=2025-12-23T03:05:44.721237389Z caller=recalculate_owned_streams.go:63 msg="detected ring changes, re-evaluating streams ownership"
level=info ts=2025-12-23T03:05:44.721246256Z caller=recalculate_owned_streams.go:52 msg="completed recalculate owned streams job"
level=info ts=2025-12-23T03:06:14.691032399Z caller=table_manager.go:187 index-store=boltdb-shipper-2023-01-01 msg="handing over indexes to shipper"
level=info ts=2025-12-23T03:06:14.691056371Z caller=table_manager.go:136 index-store=boltdb-shipper-2023-01-01 msg="uploading tables"
level=info ts=2025-12-23T03:06:14.721005496Z caller=recalculate_owned_streams.go:49 msg="starting recalculate owned streams job"
level=info ts=2025-12-23T03:06:14.721034756Z caller=recalculate_owned_streams.go:52 msg="completed recalculate owned streams job"
level=info ts=2025-12-23T03:06:44.720342538Z caller=recalculate_owned_streams.go:49 msg="starting recalculate owned streams job"
level=info ts=2025-12-23T03:06:44.720402265Z caller=recalculate_owned_streams.go:52 msg="completed recalculate owned streams job"
```

Installer Promtail

mkdir -p /opt/promtail

cd /opt/promtail

curl -L -o promtail.zip https://github.com/grafana/loki/releases/latest/download/promtail-linux-amd64.zip

unzip promtail.zip

chmod +x promtail-linux-amd64

./promtail-linux-amd64 –version

```
root@ip-172-31-37-165:/opt/promtail# unzip promtail.zip
Archive:  promtail.zip
  inflating: promtail-linux-amd64
root@ip-172-31-37-165:/opt/promtail# chmod +x promtail-linux-amd64
./promtail-linux-amd64 --version
promtail, version 3.6.3 (branch: release-3.6.x, revision: 9385bc63)
  build user:       root@f7962624fd79
  build date:       2025-12-11T09:08:55Z
  go version:       go1.25.4
  platform:         linux/amd64
  tags:             promtail_journal_enabled
root@ip-172-31-37-165:/opt/promtail#
```

Créer la config de protomail

nano /opt/promtail/promtail.yaml

```
server:
  http_listen_port: 9080
  grpc_listen_port: 0

positions:
  filename: /tmp/promtail-positions.yaml

clients:
  - url: http://127.0.0.1:3100/loki/api/v1/push

scrape_configs:
  - job_name: cowrie
    static_configs:
      - targets:
          - localhost
        labels:
          job: cowrie
          __path__: /home/cowrie/cowrie/var/log/cowrie/cowrie.json
```

C’est ce qui envoie log cowrie à Loki

Esnuite je lance

```
root@ip-172-31-37-165:/opt/promtail# /opt/promtail/promtail-linux-amd64 -config.file=/opt/promtail/promtail.yaml
level=info ts=2025-12-23T03:12:33.588252803Z caller=promtail.go:135 msg="Reloading configuration file" sha3sum=aa2952d2f032bbb8ab97005be1ff086ebee425d3d9de6ce0db9e3be418d5ec60
level=info ts=2025-12-23T03:12:33.58982271Z caller=server.go:386 msg="server listening on addresses" http=[::]:9080 grpc=[::]:45037
level=info ts=2025-12-23T03:12:33.590001814Z caller=main.go:173 msg="Starting Promtail" version="(version=3.6.3, branch=release-3.6.x, revision=9385bc63)"
level=warn ts=2025-12-23T03:12:33.590104854Z caller=promtail.go:265 msg="enable watchConfig"
level=info ts=2025-12-23T03:12:38.59004566Z caller=filetargetmanager.go:373 msg="Adding target" key="/home/cowrie/cowrie/var/log/cowrie/cowrie.json:{job=\"cowrie\"}"
level=info ts=2025-12-23T03:12:38.590130027Z caller=filetarget.go:343 msg="watching new directory" directory=/home/cowrie/cowrie/var/log/cowrie
level=info ts=2025-12-23T03:12:38.59021426Z caller=tailer.go:147 component=tailer msg="tail routine: started" path=/home/cowrie/cowrie/var/log/cowrie/cowrie.json
ts=2025-12-23T03:12:38.590263096Z caller=log.go:168 level=info msg="Seeked /home/cowrie/cowrie/var/log/cowrie/cowrie.json - &{Offset:0 Whence:0}"
```

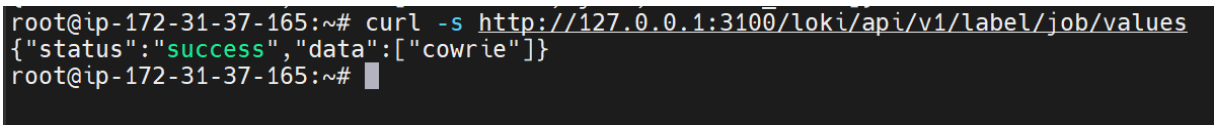
Ensuite je verifie encore via curl si tout fonctionne

curl -s http://127.0.0.1:3100/loki/api/v1/labels

```
root@ip-172-31-37-165:~# curl -s http://127.0.0.1:3100/loki/api/v1/labels
{"status":"success","data":["filename","job","service_name"]}
root@ip-172-31-37-165:~#
```

Ensuite pour vérifier que ma config cowrie a bien éte prise en compte

curl -s http://127.0.0.1:3100/loki/api/v1/label/job/values



Installation Grafana

apt install -y ca-certificates curl gnupg lsb-release

apt-get install -y apt-transport-https software-properties-common wget

curl -fsSL https://packages.grafana.com/gpg.key | gpg --dearmor -o /usr/share/keyrings/grafana.gpg

Ajouter depot grafana

echo "deb [signed-by=/usr/share/keyrings/grafana.gpg] https://packages.grafana.com/oss/deb stable main" \

> /etc/apt/sources.list.d/grafana.list

apt update

apt install -y grafana

systemctl daemon-reexec

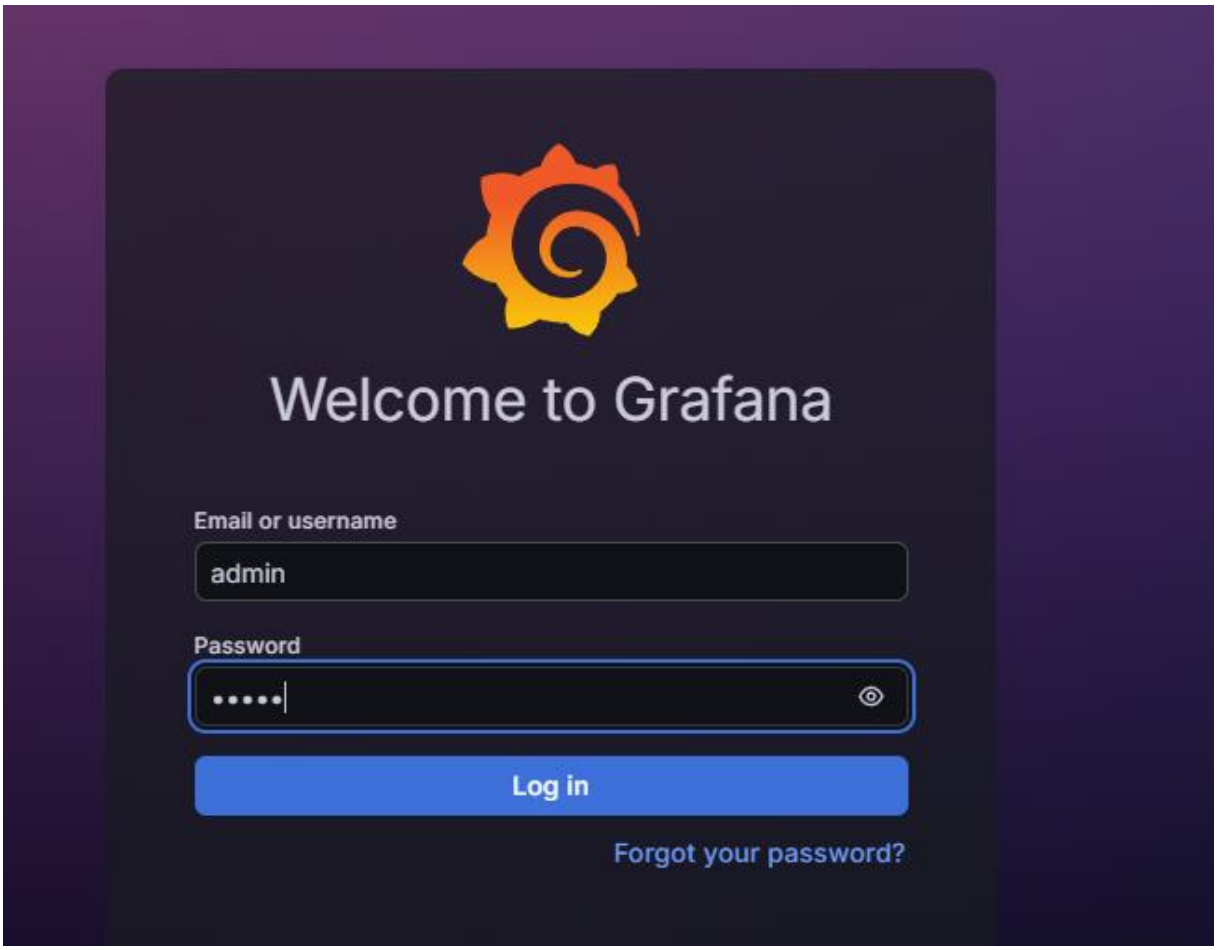
systemctl enable grafana-server

systemctl start grafana-server

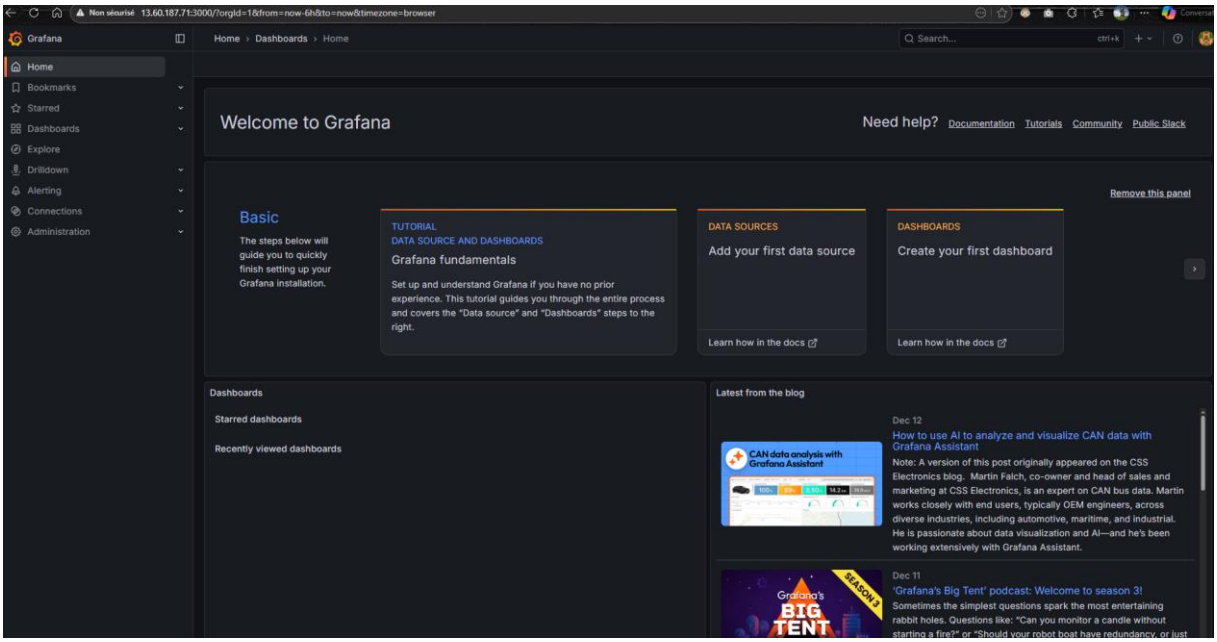
Ensuite on teste si ça fonctionne et si on a une sortie html à la commande ui suivra ça veut dire que grafana aura bien été installé

curl http://127.0.0.1:3000/login

Ensuite on accede via navigateur par defaut login/mdp admin/admin



Ça nous propose de changer mdp je le fais

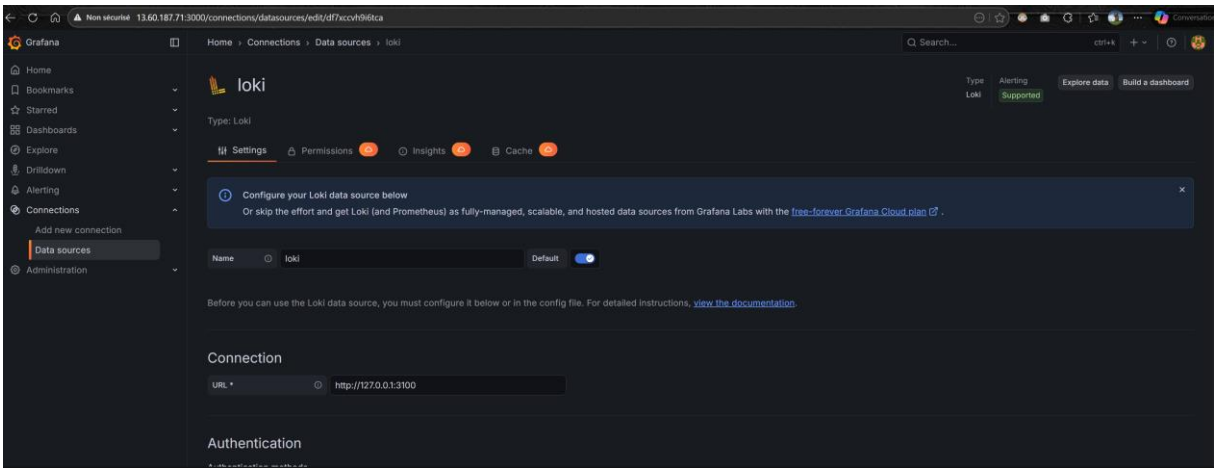


Avant d'aller plus lin il faut garder en tete que le chemin est comme telle

Cowrie → Promtail → Loki → Grafana

Relier grafana au reste

Connections > Data sources > Add data source > Loki

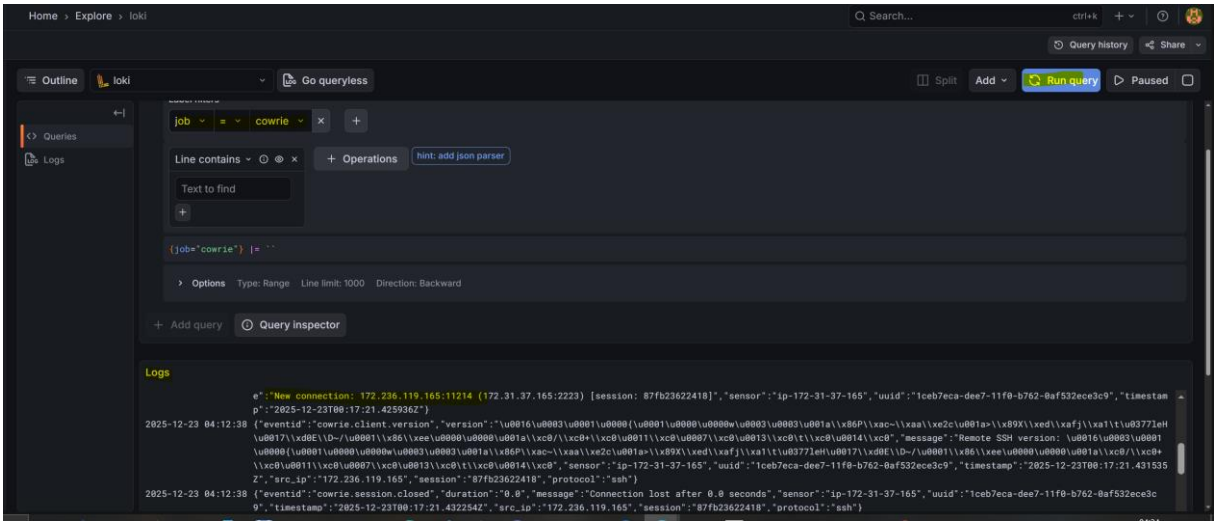


On valide

Pour voir les logs

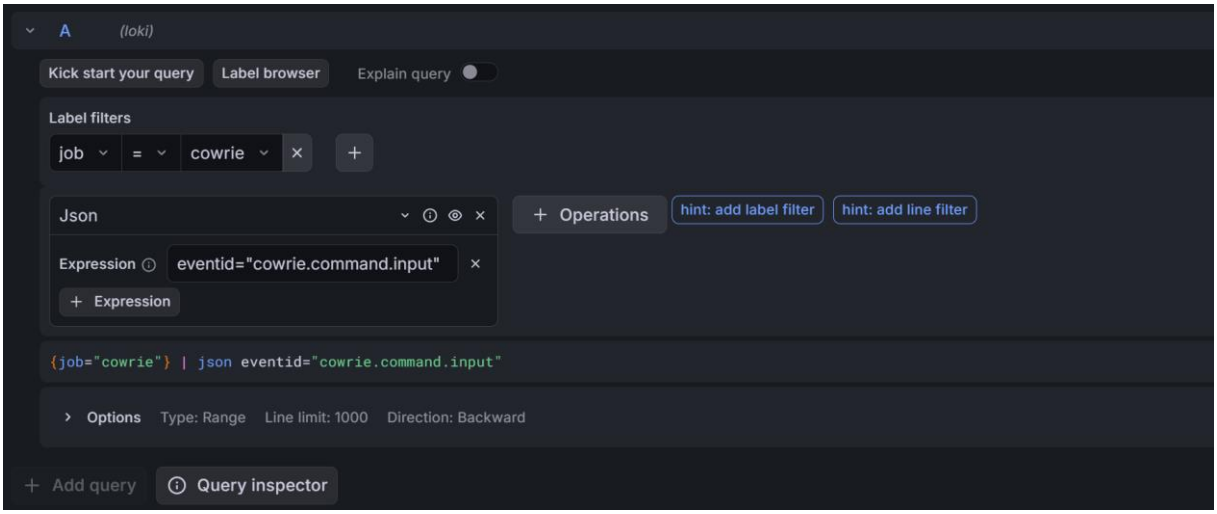
Il faut aller dans explorer > Loki

Saisir le label filter « job = cowrie » puis lancer via « Run query »



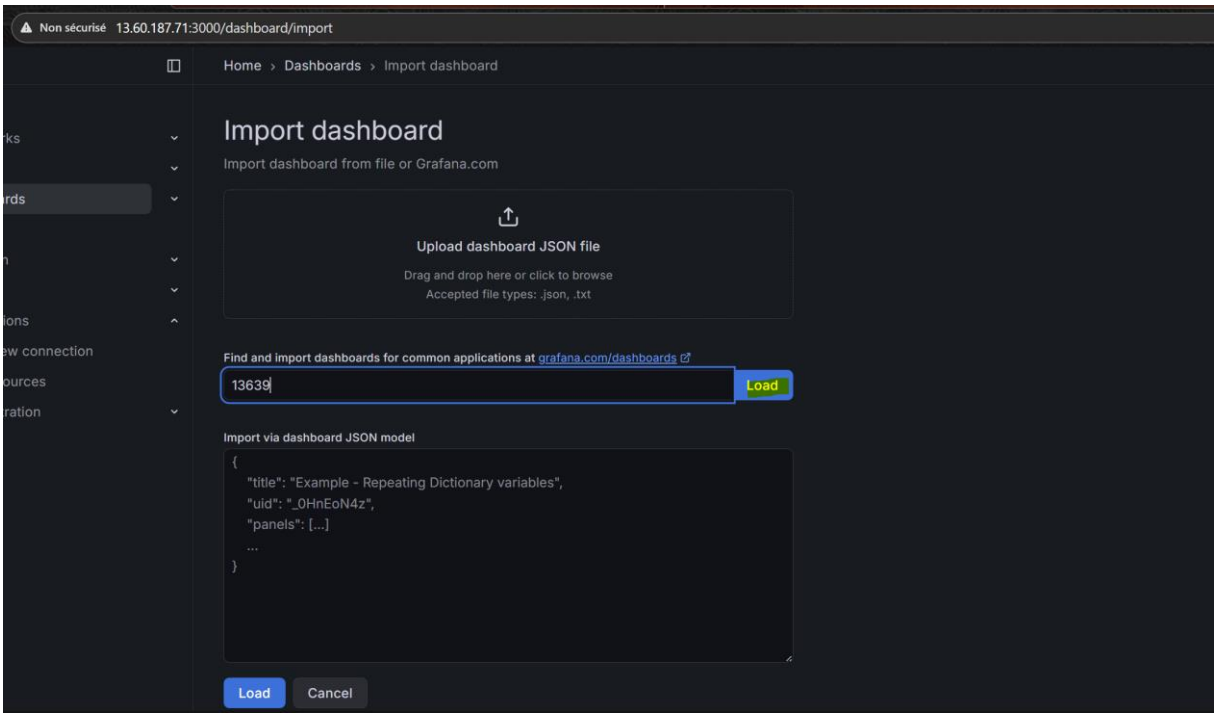
On peut maintenant voir les logs

Pour filtrer



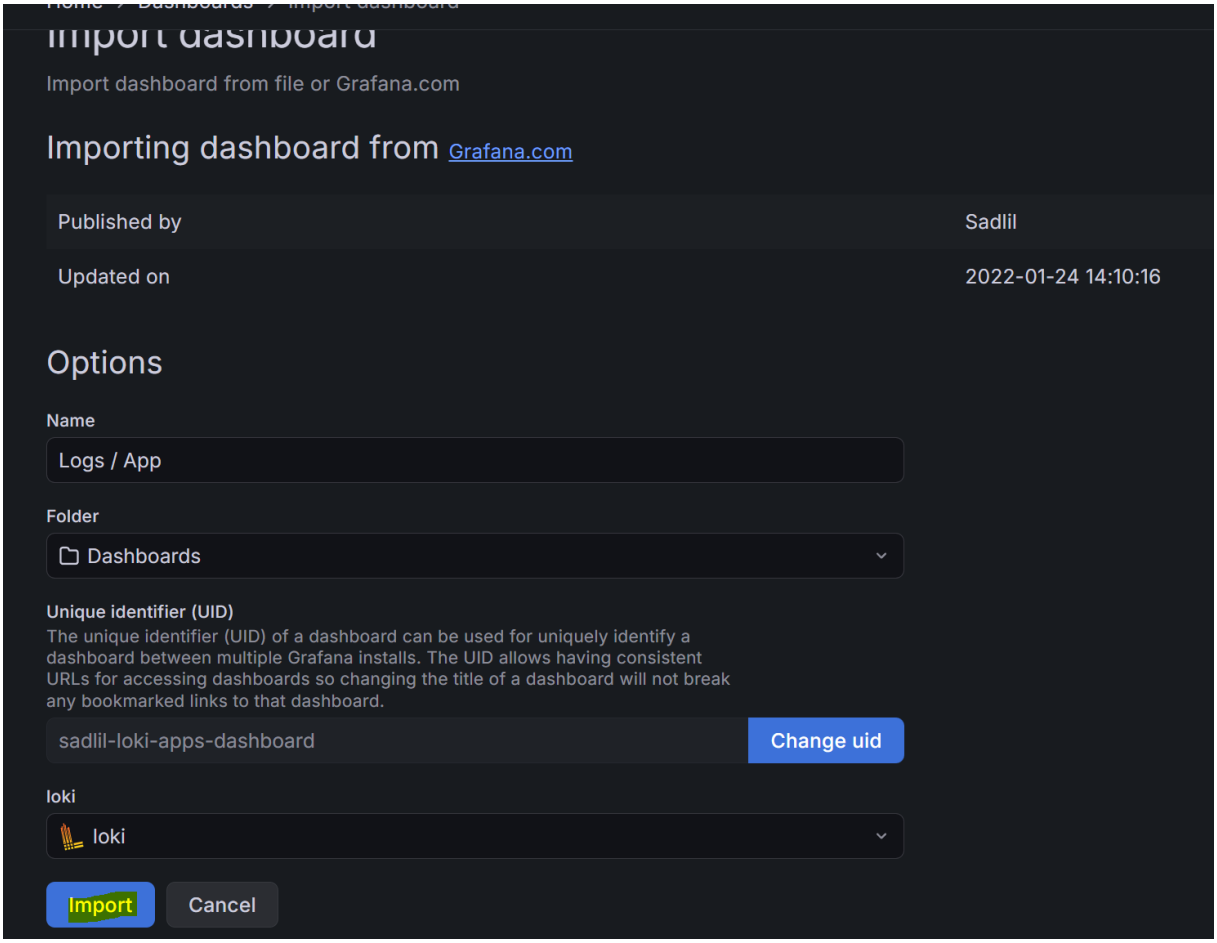
Créer un dashboard

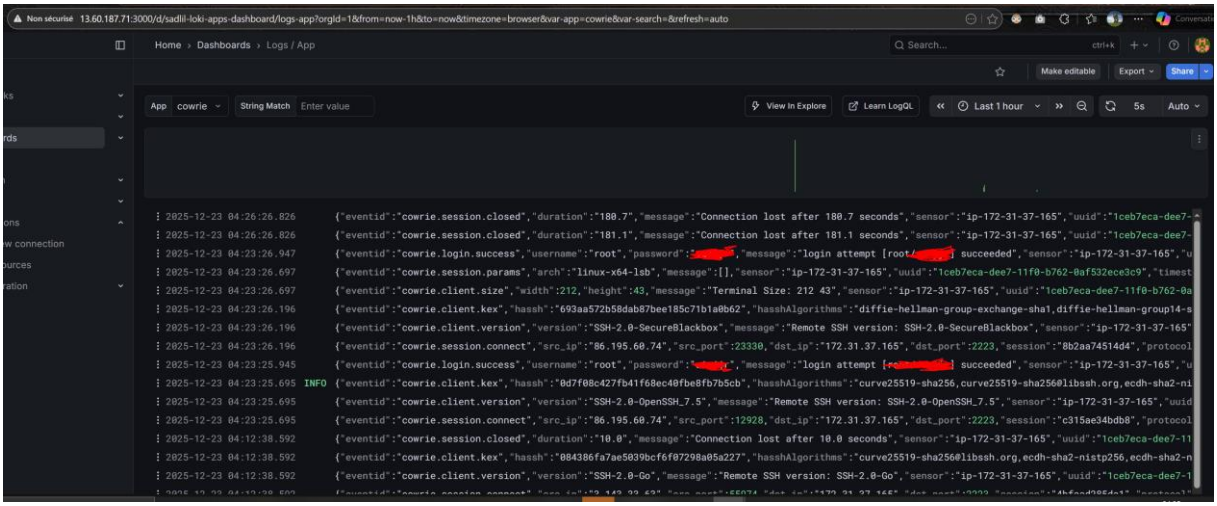
Dashboard > Create dashboard > Import dashboard > ID : 15156



Cliquer sur le load a coter de l'ID

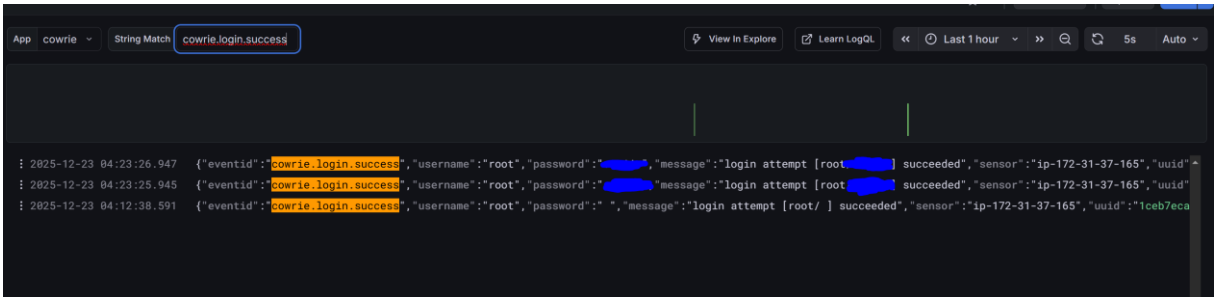
Esnuite dans menu deoulant sélectionner notre loki





Ensuite pour filtrer il faut aller dans String match et valider

cowrie.login.success



Les trois filtres qui nous intéressent

- `cowrie.login.success`
- `cowrie.command.input`
- `cowrie.session.connect`