Documentation Complète - Projet de Monitoring avec Grafana, Prometheus et Node Exporter

📋 Table des Matières

1. Aperçu du Projet
2. Architecture du Système
3. Structure du Projet
4. Prérequis et Installation
5. Configuration détaillée
6. Déploiement
7. Accès aux Services

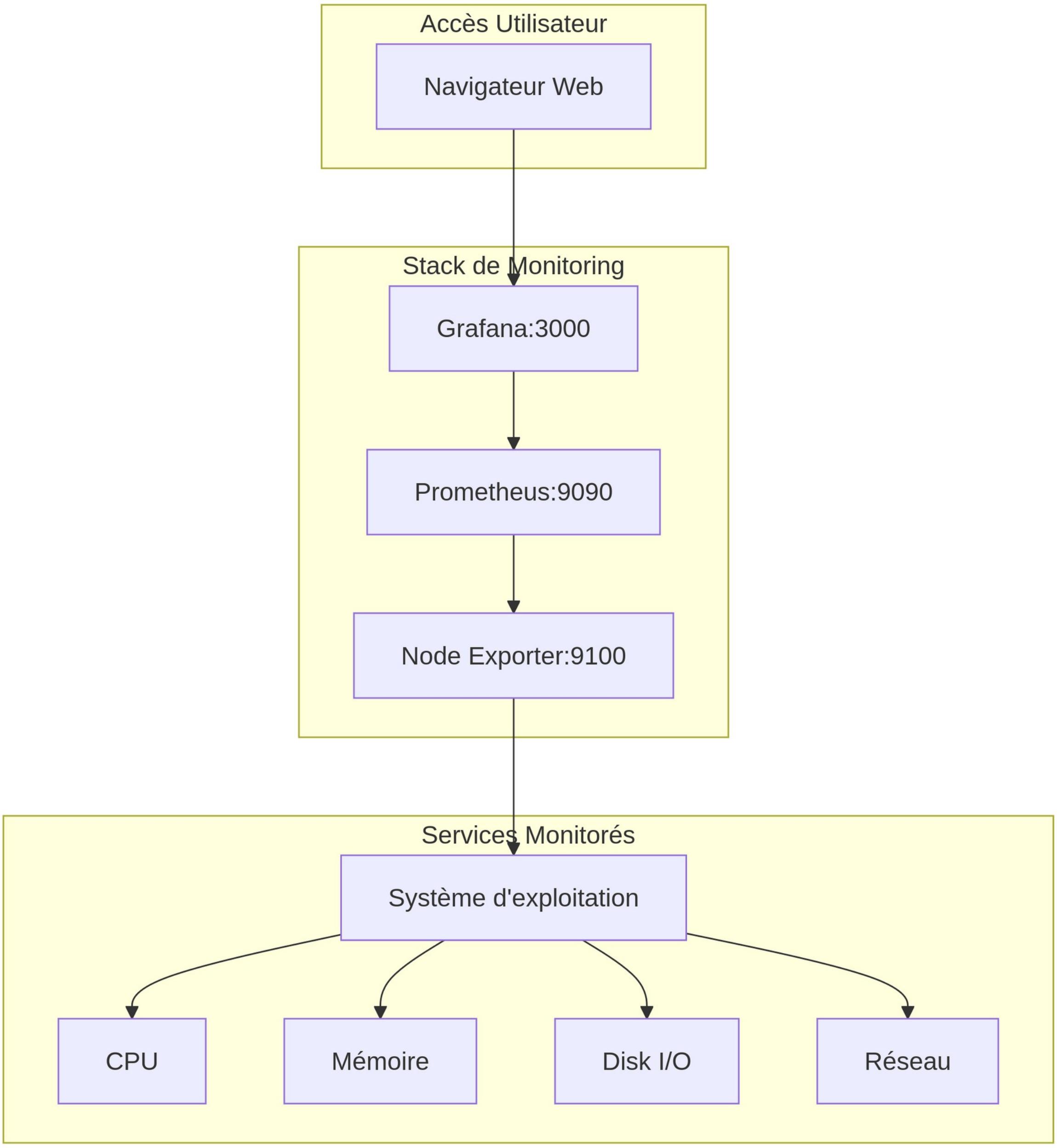
🎯 Aperçu du Projet

Ce projet déploie une stack complète de monitoring sur des serveurs AWS EC2 en utilisant Ansible. La stack comprend :

- **Grafana** : Plateforme de visualisation et de dashboard
- **Prometheus** : Système de monitoring et d'alerting
- **Node Exporter** : Collecteur de métriques système

🎯 Objectifs

- Surveillance en temps réel des performances système
- Tableaux de bord personnalisés pour les métriques AWS EC2
- Automatisation complète du déploiement
- Architecture scalable et reproductible



🏗️ Architecture du Système

🔄 Flux de données

1. **Node Exporter** collecte les métriques système
2. **Prometheus** scrape et stocke les métriques
3. **Grafana** visualise les données via des dashboards
4. **Utilisateurs** accèdent aux dashboards via le navigateur

---

📁 Structure du Projet

text

```
grafana-monitoring-ansible/
├── 📄 ansible.cfg
├── 📁 inventories/
│   └── 📁 production/
│       └── 📄 hosts
├── 📁 group_vars/
│   └── 📄 all.yml
├── 📁 playbooks/
│   └── 📄 deploy-monitoring.yml
├── 📁 roles/
│   ├── 📁 grafana/
│   │   ├── 📁 tasks/
│   │   │   └── 📄 main.yml
│   │   ├── 📁 templates/
│   │   │   ├── 📄 aws-dashboard.json.j2
│   │   │   └── 📄 grafana.ini.j2
│   │   └── 📁 files/
│   │       └── 📄 datasource.yml
│   ├── 📁 prometheus/
│   │   ├── 📁 tasks/
│   │   │   └── 📄 main.yml
│   │   ├── 📁 templates/
│   │   │   └── 📄 prometheus.yml.j2
│   │   └── 📁 handlers/
│   │       └── 📄 main.yml
│   └── 📁 node_exporter/
│       ├── 📁 tasks/
│       │   └── 📄 main.yml
│       └── 📁 handlers/
│           └── 📄 main.yml
└── 📄 README.md
```

📋 Description des Fichiers

🔧 Configuration Ansible

- **`ansible.cfg`** : Configuration globale d'Ansible
- **`inventories/production/hosts`** : Définition des serveurs cibles
- **`group_vars/all.yml`** : Variables communes à tous les hôtes

🚀 Playbooks

- **`playbooks/deploy-monitoring.yml`** : Playbook principal de déploiement

⚙️ Rôles

🎨 Grafana :

- `tasks/main.yml` : Installation et configuration
- `templates/grafana.ini.j2` : Configuration serveur
- `templates/aws-dashboard.json.j2` : Dashboard de monitoring AWS
- `files/datasource.yml` : Configuration source de données Prometheus

📊 **Prometheus** :

- `tasks/main.yml` : Installation et configuration
- `templates/prometheus.yml.j2` : Configuration des jobs de scraping
- `handlers/main.yml` : Gestionnaires de services

🖥️ **Node Exporter** :

- `tasks/main.yml` : Installation et configuration
- `handlers/main.yml` : Gestionnaires de services

⚙️ Prérequis et Installation

🖥️ Config Système

Serveur Cible

- **Système d'exploitation** : Amazon Linux 2023
- **Utilisateur** : `ec2-user` avec privilèges sudo
- **Ports ouverts** : 3000 (Grafana), 9090 (Prometheus), 9100 (Node Exporter)
- **Espace disque** : Minimum 37GB libre
- **Mémoire** : Minimum 8GB RAM

Machine de Déploiement

- **Ansible** : Version 2.9+
- **Python** : Version 3.6+
- **Accès SSH** : Clé SSH configurée

🔑 Configuration SSH

bash

```bash
# Générer une clé SSH (si nécessaire)
ssh-keygen -t rsa -b 4096 -f ~/.ssh/my_new_key

# Copier la clé publique vers le serveur
ssh-copy-id -i ~/.ssh/my_new_key.pub ec2-user@3.105.85.131
```

📦 Installation des Dépendances

bash

```bash
# Sur la machine de déploiement
sudo yum install ansible python3-pip -y

# Vérifier l'installation
ansible --version
```

```
python3 --version
```

🔧 Configuration Détaillée

🔐 Configuration Ansible (`ansible.cfg`)

ini

```
[defaults]
host_key_checking = False
inventory = inventories/production/hosts
private_key_file = ~/.ssh/my_new_key
remote_user = ec2-user
roles_path = roles

[privilege_escalation]
become = True
become_method = sudo
```

🖥️ Inventaire (`inventories/production/hosts`)

ini

```
[monitoring_servers]
3.105.85.131

[monitoring_servers:vars]
ansible_python_interpreter=/usr/bin/python3
```

📊 Variables Globales (`group_vars/all.yml`)

yaml

```
# Configuration Grafana
grafana_admin_user: "admin"
grafana_admin_password: "admin"
grafana_version: "10.2.0"
grafana_port: 3000

# Configuration Prometheus
prometheus_version: "2.47.0"
prometheus_port: 9090

# Configuration Node Exporter
node_exporter_version: "1.6.1"
node_exporter_port: 9100

# Configuration Domaine
domain_name: "3.105.85.131"
```

🚀 Déploiement

🔄 Processus de Déploiement

1. Vérification Pré-déploiement

bash

```
# Tester la connexion SSH
ansible -i inventories/production/hosts all -m ping

# Vérifier l'inventaire
ansible-inventory -i inventories/production/hosts --list
```

2. Déploiement Complet

bash

```
# Exécuter le playbook principal
ansible-playbook playbooks/deploy-monitoring.yml

# Avec verbose pour plus de détails
ansible-playbook playbooks/deploy-monitoring.yml -v
```

3. Déploiement par Rôle (si nécessaire)

bash

```
# Déployer seulement Node Exporter
ansible-playbook playbooks/deploy-monitoring.yml --tags "node_exporter"

# Déployer seulement Prometheus
ansible-playbook playbooks/deploy-monitoring.yml --tags "prometheus"

# Déployer seulement Grafana
ansible-playbook playbooks/deploy-monitoring.yml --tags "grafana"
```

📋 Étapes du Déploiement

🔄 Pré-tâches

- Mise à jour du système
- Installation des dépendances (firewalld, python3-pip)
- Démarrage et configuration du firewall

🖥️ Node Exporter

- Création de l'utilisateur dédié
- Téléchargement et installation
- Configuration du service systemd
- Ouverture du port 9100

📊 Prometheus

- Création de l'utilisateur dédié
- Téléchargement et installation
- Configuration des jobs de scraping
- Ouverture du port 9090

🎨 Grafana

- Ajout du repository officiel
- Installation du package
- Configuration via templates
- Configuration de la source de données Prometheus
- Déploiement du dashboard AWS EC2
- Ouverture du port 3000

✅ Post-tâches

- Vérification du statut des services
- Affichage des URLs d'accès

🌐 Accès aux Services

🔗 URLs des Services

Service	URL	Port	Description
Grafana	`http://3.105.85.131:3000`	3000	Interface de visualisation
Prometheus	`http://3.105.85.131:9090`	9090	Interface de requêtes
Node Exporter	`http://3.105.85.131:9100/metrics`	9100	Métriques brutes

🔐 Identifiants Grafana

- **Utilisateur** : `admin`
- **Mot de passe** : `admin`

NB : Changer le mot de passe après la première connexion si vous voulez !

📊 Dashboard AWS EC2

Le dashboard inclut les métriques suivantes :

🖥️ Métriques Système

- **Utilisation CPU** : Pourcentage d'utilisation du processeur
- **Utilisation Mémoire** : Pourcentage de mémoire utilisée

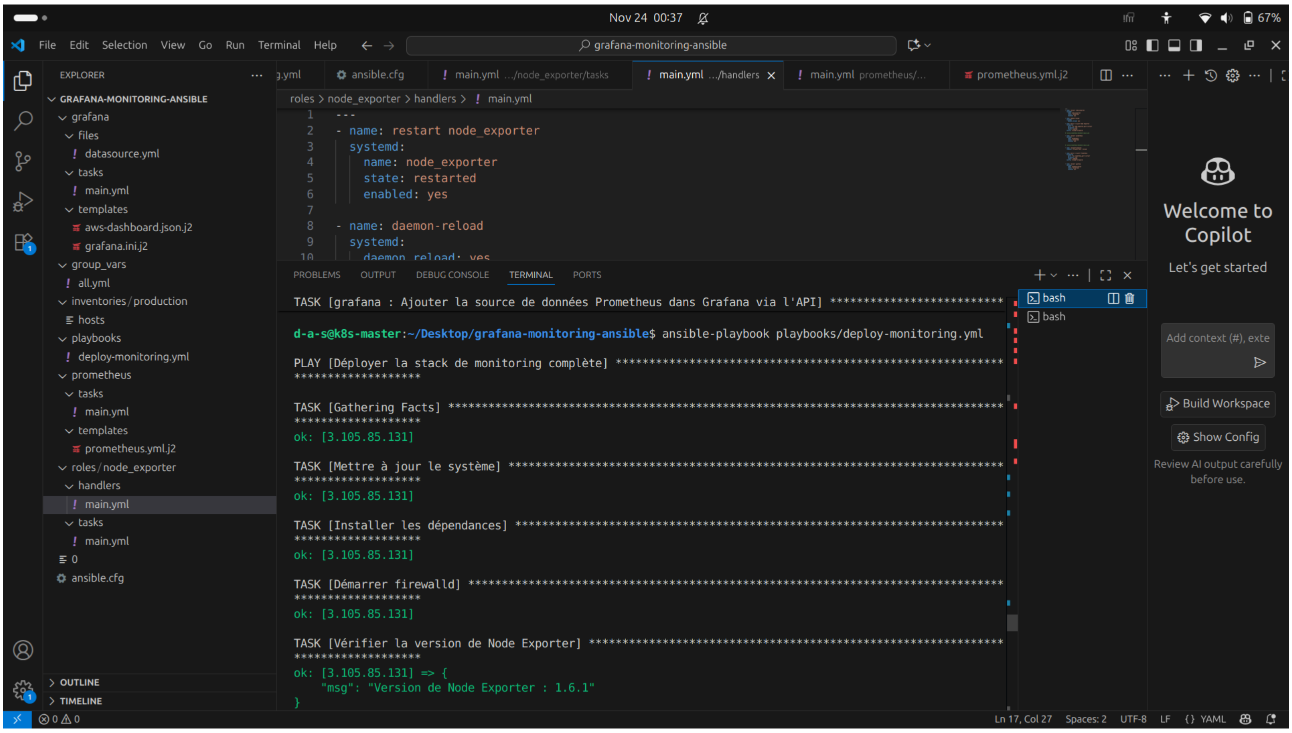
- **I/O Disque** : Taux de lecture/écriture en bytes/s
- **Trafic Réseau** : Données reçues/transmises en bytes/s

📈 Panels du Dashboard

1. **CPU Usage** : Métrique statistique avec unité en pourcentage
2. **Memory Usage** : Métrique statistique avec unité en pourcentage
3. **Disk I/O** : Série temporelle pour lecture/écriture
4. **Network Traffic** : Série temporelle pour réception/transmission

---

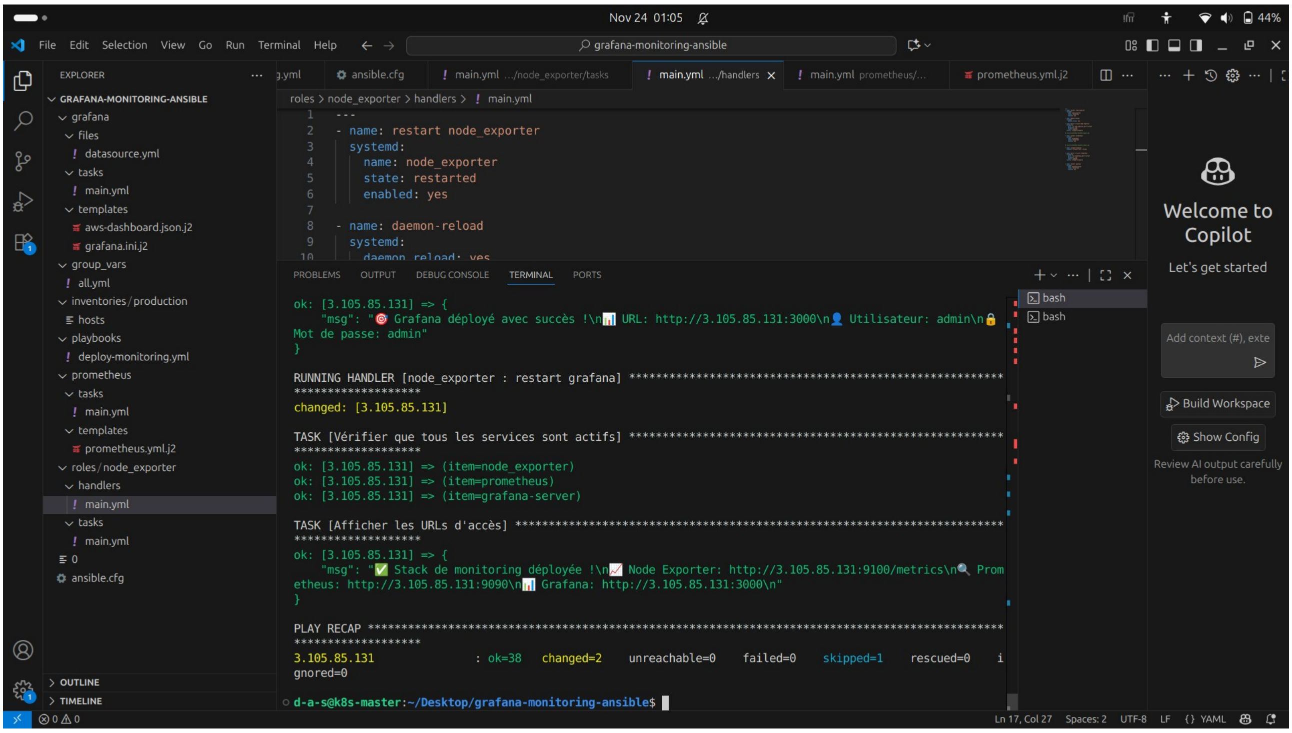
Resultats



```yaml
---
- name: restart node_exporter
  systemd:
    name: node_exporter
    state: restarted
    enabled: yes

- name: daemon-reload
  systemd:
    daemon_reload: yes
```

```
TASK [grafana : Ajouter la source de données Prometheus dans Grafana via l'API] *************************

d-a-s@k8s-master:~/Desktop/grafana-monitoring-ansible$ ansible-playbook playbooks/deploy-monitoring.yml

PLAY [Déployer la stack de monitoring complète] ***********************************************************
*******************

TASK [Gathering Facts] *************************************************************************************
*******************
ok: [3.105.85.131]

TASK [Mettre à jour le système] ****************************************************************************
*******************
ok: [3.105.85.131]

TASK [Installer les dépendances] ***************************************************************************
*******************
ok: [3.105.85.131]

TASK [Démarrer firewalld] **********************************************************************************
*******************
ok: [3.105.85.131]

TASK [Vérifier la version de Node Exporter] ****************************************************************
*******************
ok: [3.105.85.131] => {
    "msg": "Version de Node Exporter : 1.6.1"
}
```

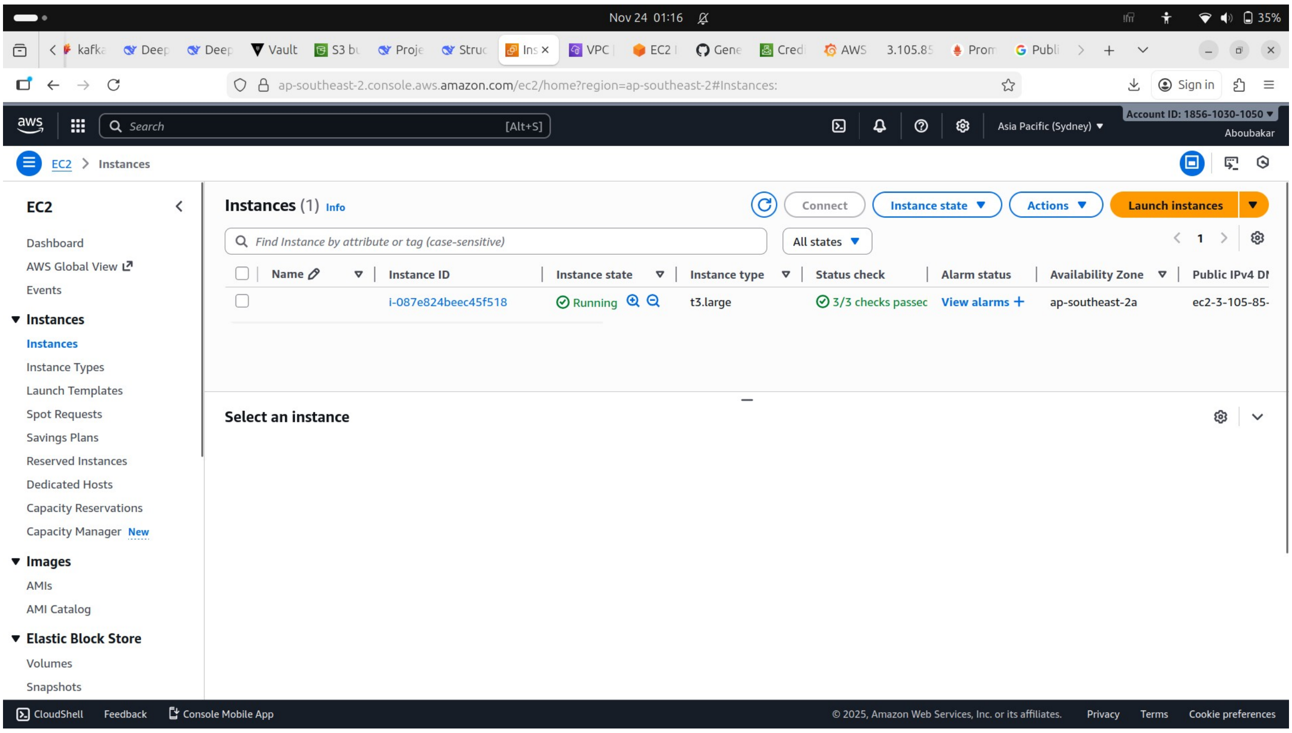
Déployer la stack de monitoring complète



```yaml
---
- name: restart node_exporter
  systemd:
    name: node_exporter
    state: restarted
    enabled: yes

- name: daemon-reload
  systemd:
    daemon_reload: yes
```

```
ok: [3.105.85.131] => {
    "msg": "🎯 Grafana déployé avec succès !\n📊 URL: http://3.105.85.131:3000\n👤 Utilisateur: admin\n🔒
Mot de passe: admin"
}

RUNNING HANDLER [node_exporter : restart grafana] *******************************************************
******************
changed: [3.105.85.131]

TASK [Vérifier que tous les services sont actifs] *******************************************************
******************
ok: [3.105.85.131] => (item=node_exporter)
ok: [3.105.85.131] => (item=prometheus)
ok: [3.105.85.131] => (item=grafana-server)

TASK [Afficher les URLs d'accès] ************************************************************************
******************
ok: [3.105.85.131] => {
    "msg": "✅ Stack de monitoring déployée !\n📈 Node Exporter: http://3.105.85.131:9100/metrics\n🔍 Prom
etheus: http://3.105.85.131:9090\n📊 Grafana: http://3.105.85.131:3000\n"
}

PLAY RECAP **********************************************************************************************
******************
3.105.85.131               : ok=38   changed=2    unreachable=0    failed=0    skipped=1    rescued=0    i
gnored=0

d-a-s@k8s-master:~/Desktop/grafana-monitoring-ansible$ 
```

Déployer la stack de monitoring complète-fin



ap-southeast-2.console.aws.amazon.com/ec2/home?region=ap-southeast-2#Instances:

Account ID: 1856-1030-1050

Aboubakar

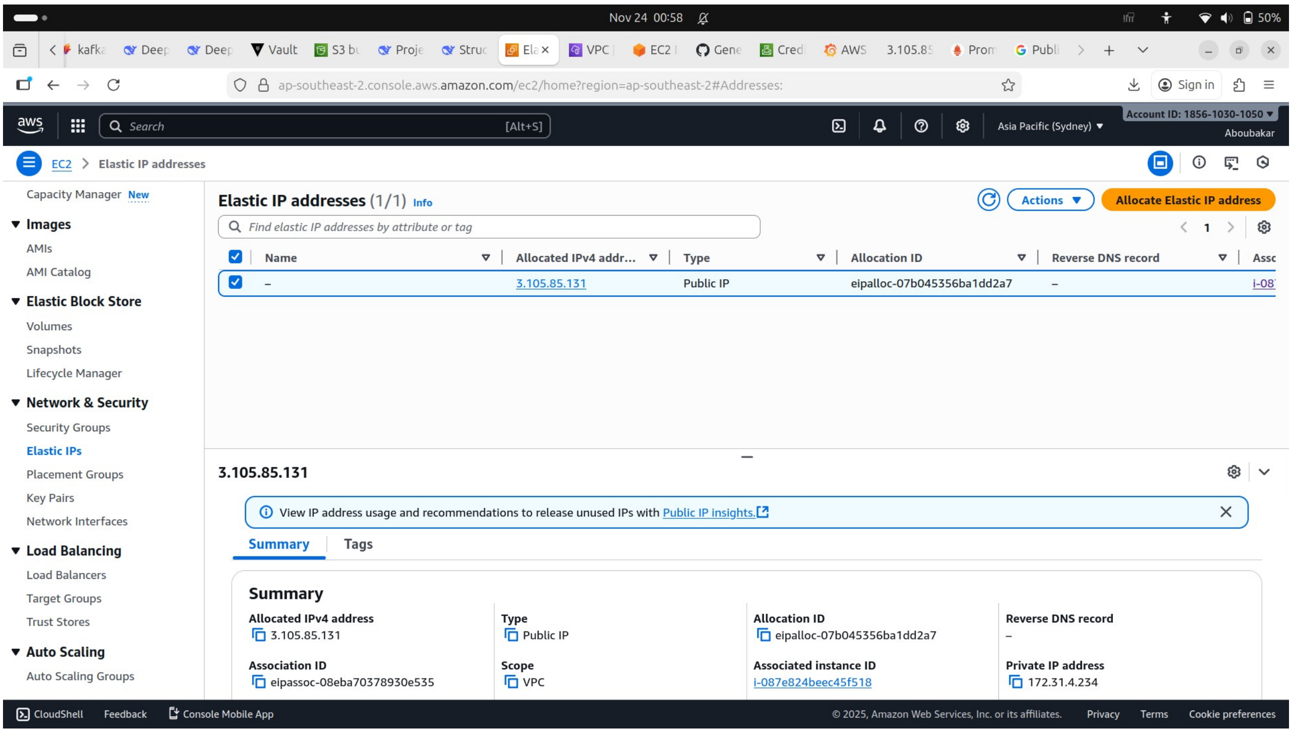
Asia Pacific (Sydney)

Instances (1)

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DI
	i-087e824beec45f518	Running	t3.large	3/3 checks passec	View alarms +	ap-southeast-2a	ec2-3-105-85-

Select an instance

EC2 Instance



Elastic IP addresses (1/1) Info

☑	Name	Allocated IPv4 addr…	Type	Allocation ID	Reverse DNS record	Assc
☑	–	3.105.85.131	Public IP	eipalloc-07b045356ba1dd2a7	–	i-08

3.105.85.131

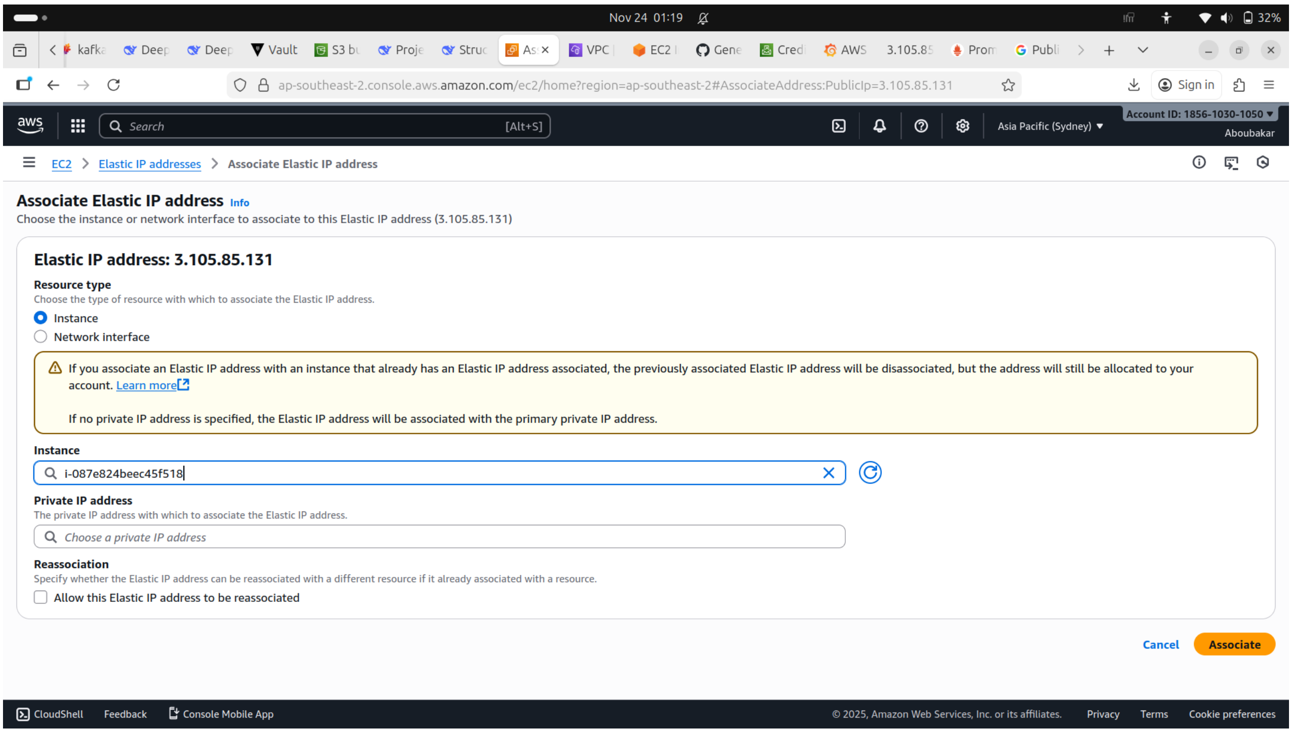
View IP address usage and recommendations to release unused IPs with Public IP insights.

Summary | Tags

Summary

Allocated IPv4 address	Type	Allocation ID	Reverse DNS record
3.105.85.131	Public IP	eipalloc-07b045356ba1dd2a7	–
Association ID	**Scope**	**Associated instance ID**	**Private IP address**
eipassoc-08eba70378930e535	VPC	i-087e824beec45f518	172.31.4.234

- Pour rendre les liens de votre instance EC2 accessible depuis internet il faut créé un Elastic IP, l'associé à votre instance avec le bouton Actions cliquer sur Associate Elastic IP Address



Associate Elastic IP address

Choose the instance or network interface to associate to this Elastic IP address (3.105.85.131)

Elastic IP address: 3.105.85.131

Resource type

Choose the type of resource with which to associate the Elastic IP address.

● Instance

○ Network interface

If you associate an Elastic IP address with an instance that already has an Elastic IP address associated, the previously associated Elastic IP address will be disassociated, but the address will still be allocated to your account. Learn more

If no private IP address is specified, the Elastic IP address will be associated with the primary private IP address.

Instance: i-087e824beec45f518

Private IP address

The private IP address with which to associate the Elastic IP address.

Choose a private IP address

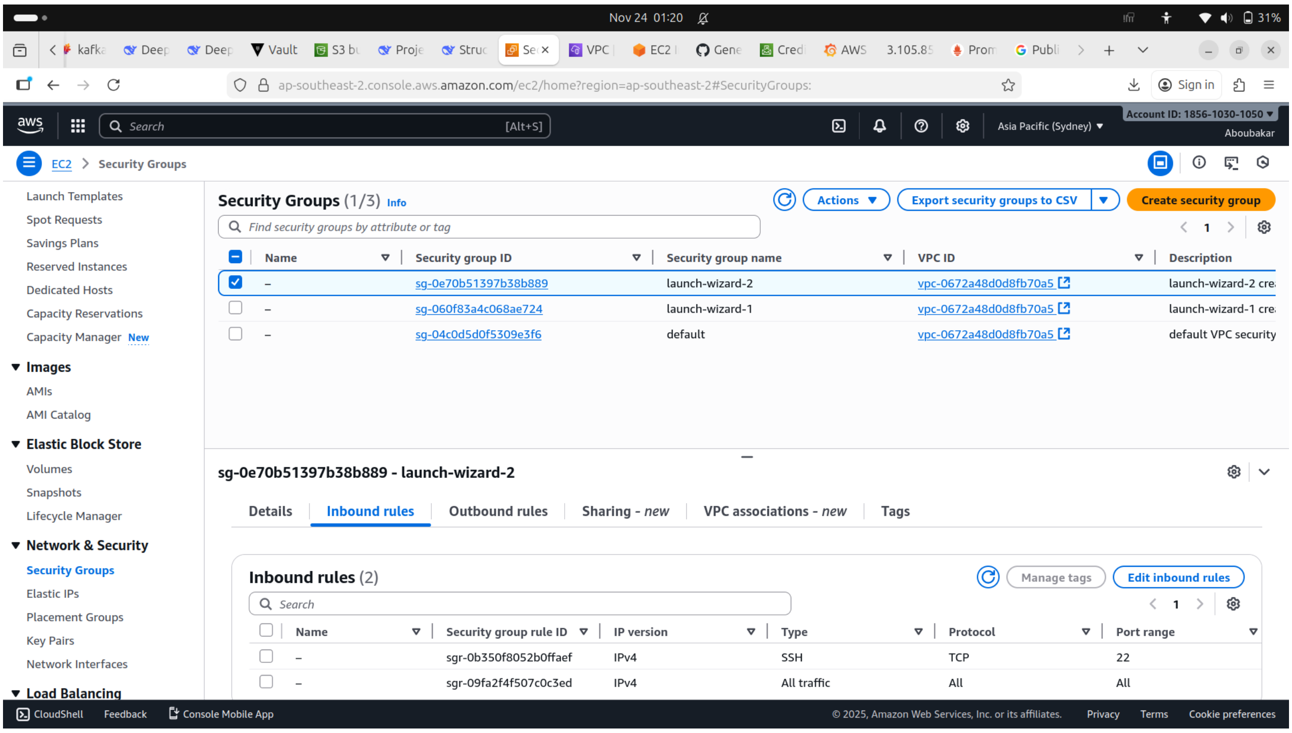
Reassociation

Specify whether the Elastic IP address can be reassociated with a different resource if it already associated with a resource.

□ Allow this Elastic IP address to be reassociated

Nov 24 01:19
ap-southeast-2.console.aws.amazon.com/ec2/home?region=ap-southeast-2#AssociateAddress:PublicIp=3.105.85.131
Asia Pacific (Sydney)
Account ID: 1856-1030-1050
Aboubakar
EC2 > Elastic IP addresses > Associate Elastic IP address
Cancel
Associate
CloudShell
Feedback
Console Mobile App
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

Choisir notre instance et cliquer sur Associate



Security Groups (1/3) Info

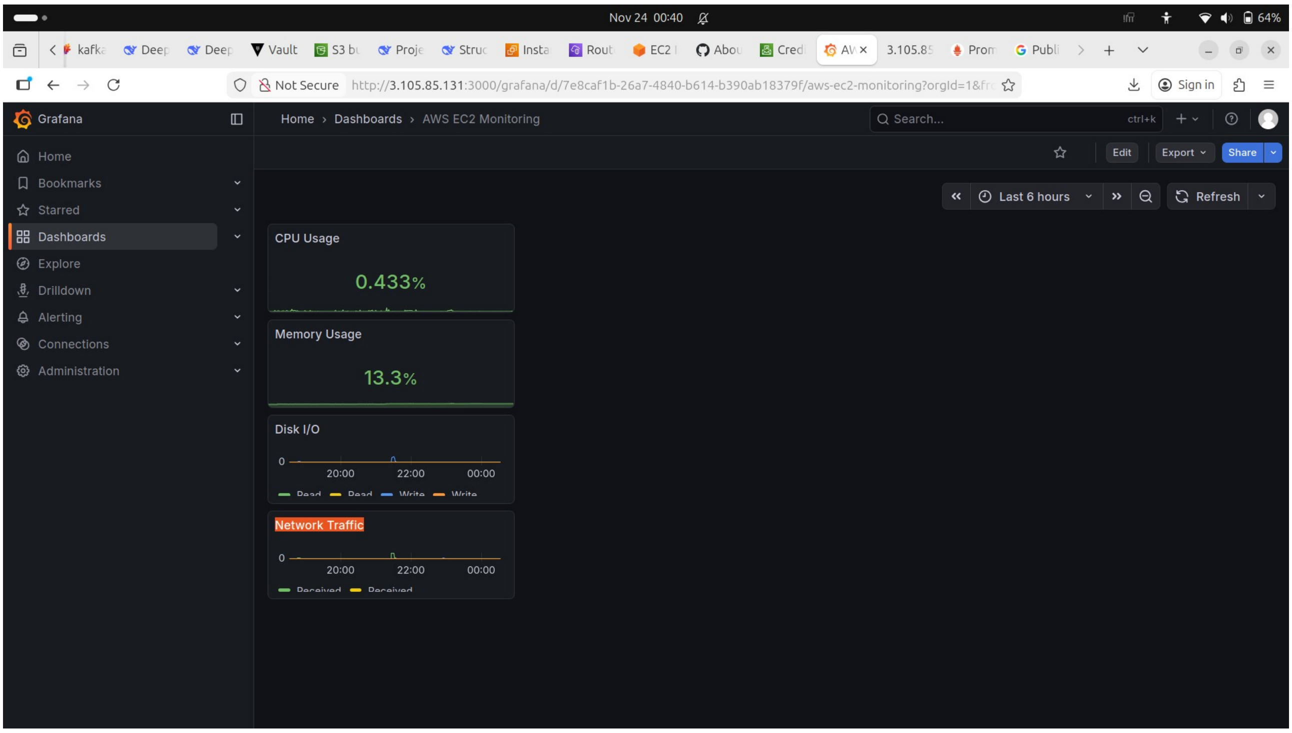
	Name	Security group ID	Security group name	VPC ID	Description
☑	–	sg-0e70b51397b38b889	launch-wizard-2	vpc-0672a48d0d8fb70a5	launch-wizard-2 crea
□	–	sg-060f83a4c068ae724	launch-wizard-1	vpc-0672a48d0d8fb70a5	launch-wizard-1 crea
□	–	sg-04c0d5d0f5309e3f6	default	vpc-0672a48d0d8fb70a5	default VPC security

sg-0e70b51397b38b889 - launch-wizard-2

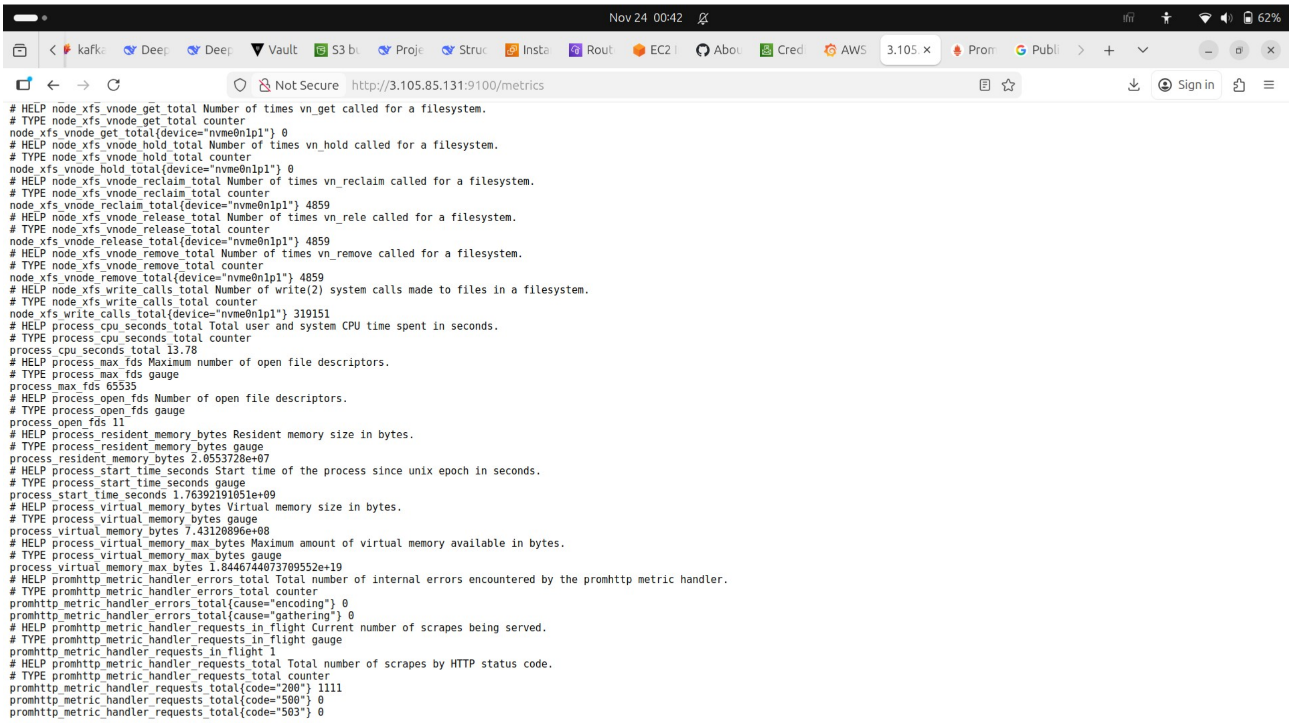
Inbound rules (2)

	Name	Security group rule ID	IP version	Type	Protocol	Port range
□	–	sgr-0b350f8052b0ffaef	IPv4	SSH	TCP	22
□	–	sgr-09fa2f4f507c0c3ed	IPv4	All traffic	All	All

Choisir le Security Groups en lien avec votre instance et permettre le trafic entrant pour rendre l' IP Elastic Publique et accéder aux applications et liens depuis internet

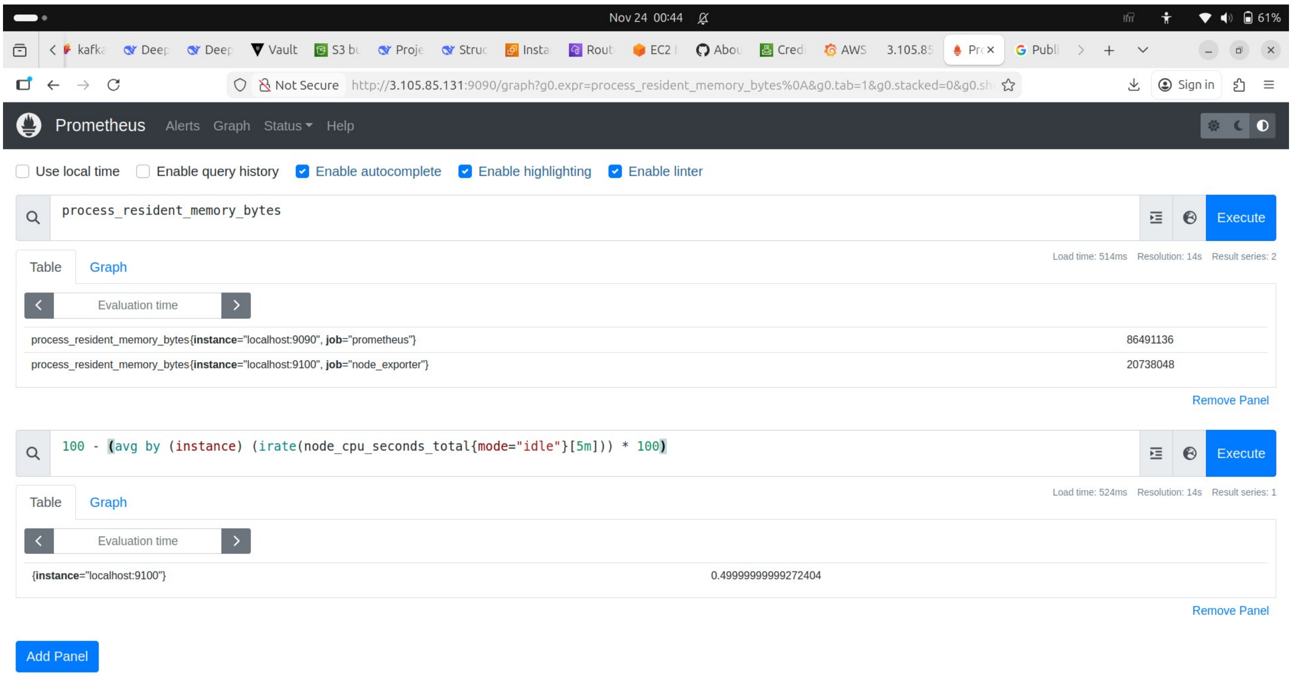


Dashboard AWS EC2 Monitoring avec les métriques



```
# HELP node_xfs_vnode_get_total Number of times vn_get called for a filesystem.
# TYPE node_xfs_vnode_get_total counter
node_xfs_vnode_get_total{device="nvme0n1p1"} 0
# HELP node_xfs_vnode_hold_total Number of times vn_hold called for a filesystem.
# TYPE node_xfs_vnode_hold_total counter
node_xfs_vnode_hold_total{device="nvme0n1p1"} 0
# HELP node_xfs_vnode_reclaim_total Number of times vn_reclaim called for a filesystem.
# TYPE node_xfs_vnode_reclaim_total counter
node_xfs_vnode_reclaim_total{device="nvme0n1p1"} 4859
# HELP node_xfs_vnode_release_total Number of times vn_rele called for a filesystem.
# TYPE node_xfs_vnode_release_total counter
node_xfs_vnode_release_total{device="nvme0n1p1"} 4859
# HELP node_xfs_vnode_remove_total Number of times vn_remove called for a filesystem.
# TYPE node_xfs_vnode_remove_total counter
node_xfs_vnode_remove_total{device="nvme0n1p1"} 4859
# HELP node_xfs_write_calls_total Number of write(2) system calls made to files in a filesystem.
# TYPE node_xfs_write_calls_total counter
node_xfs_write_calls_total{device="nvme0n1p1"} 319151
# HELP process_cpu_seconds_total Total user and system CPU time spent in seconds.
# TYPE process_cpu_seconds_total counter
process_cpu_seconds_total 13.78
# HELP process_max_fds Maximum number of open file descriptors.
# TYPE process_max_fds gauge
process_max_fds 65535
# HELP process_open_fds Number of open file descriptors.
# TYPE process_open_fds gauge
process_open_fds 11
# HELP process_resident_memory_bytes Resident memory size in bytes.
# TYPE process_resident_memory_bytes gauge
process_resident_memory_bytes 2.0553728e+07
# HELP process_start_time_seconds Start time of the process since unix epoch in seconds.
# TYPE process_start_time_seconds gauge
process_start_time_seconds 1.76392191051e+09
# HELP process_virtual_memory_bytes Virtual memory size in bytes.
# TYPE process_virtual_memory_bytes gauge
process_virtual_memory_bytes 7.43120896e+08
# HELP process_virtual_memory_max_bytes Maximum amount of virtual memory available in bytes.
# TYPE process_virtual_memory_max_bytes gauge
process_virtual_memory_max_bytes 1.8446744073709552e+19
# HELP promhttp_metric_handler_errors_total Total number of internal errors encountered by the promhttp metric handler.
# TYPE promhttp_metric_handler_errors_total counter
promhttp_metric_handler_errors_total{cause="encoding"} 0
promhttp_metric_handler_errors_total{cause="gathering"} 0
# HELP promhttp_metric_handler_requests_in_flight Current number of scrapes being served.
# TYPE promhttp_metric_handler_requests_in_flight gauge
promhttp_metric_handler_requests_in_flight 1
# HELP promhttp_metric_handler_requests_total Total number of scrapes by HTTP status code.
# TYPE promhttp_metric_handler_requests_total counter
promhttp_metric_handler_requests_total{code="200"} 1111
promhttp_metric_handler_requests_total{code="500"} 0
promhttp_metric_handler_requests_total{code="503"} 0
```

API des métriques



http://3.105.85.131:9090/graph?g0.expr=process_resident_memory_bytes%0A&g0.tab=1&g0.stacked=0&g0.sh

□ Use local time □ Enable query history ☑ Enable autocomplete ☑ Enable highlighting ☑ Enable linter

```
process_resident_memory_bytes
```

Load time: 514ms Resolution: 14s Result series: 2

```
process_resident_memory_bytes{instance="localhost:9090", job="prometheus"}    86491136
process_resident_memory_bytes{instance="localhost:9100", job="node_exporter"} 20738048
```

```
100 - (avg by (instance) (irate(node_cpu_seconds_total{mode="idle"}[5m])) * 100)
```

Load time: 524ms Resolution: 14s Result series: 1

```
{instance="localhost:9100"}    0.49999999999272404
```

Prometheus et quelques réquetes:

1 - La première requête donne la quantité de mémoire résidentielle utilisée par un processus spécifique. La mémoire résidentielle est la mémoire utilisée par un programme (ou processus) et qui réside réellement en mémoire physique (RAM), par opposition à la mémoire virtuelle (qui peut être échangée avec le disque dur)

2 – La deuxième permet de suivre la consommation mémoire de Prometheus et Node Exporter en temps réel, afin de t'assurer que ces processus n'utilisent pas de manière excessive de la mémoire.

✅ Conclusion

Cette stack de monitoring fournit une solution complète pour surveiller les performances des instances AWS EC2. L'automatisation via Ansible assure un déploiement reproductible et fiable, tandis que la modularité des rôles permet une maintenance ai