EXAMEN DE CLOUD COMPUTING

CLASS: 5 anne
FILIÈRE: IRST
MATRICULE: 950 2367

Mettre en place une architecture sécurisée type “bastion” dans le cloud AWS pour superviser des instances privées, en garantissant que seul un poste local (VM UTM) peut y accéder.

Accès sécurisé & supervision centralisée

🎯 **Objectif pédagogique :** Mettre en place une architecture sécurisée type “bastion” dans le cloud AWS pour superviser des instances privées, **en garantissant que seul un poste local (VM VirtualBox) peut y accéder**.

📌 Travail demandé :

1. **Sur la VM locale (Ubuntu UTM)** :
Générer une paire de clés SSH pour se connecter au cloud.
Tester les connexions réseau depuis la VM vers AWS.
Documenter l’environnement local utilisé (nom VM, IP locale, version Ubuntu).
Préparer les scripts de supervision à exécuter depuis la bastion (ex. uptime, netstat, df -h...).
2. **Sur AWS :**
Créer un VPC avec :
Une instance **EC2 bastion** dans un **subnet public**
Deux instances **EC2 privées** dans un **subnet privé**
Configurer la bastion box pour qu’elle soit **accessible uniquement depuis l’IP de la VM locale** (filtrage IP dans les Security Groups).
Depuis la bastion, accéder aux deux EC2 privées par **SSH interne**.
Installer et lancer des scripts de supervision simple depuis la bastion vers les EC2 privées.
Activer **fail2ban** sur la bastion pour sécuriser les accès.
(Bonus) Installer **Netdata** sur les EC2 privées et les visualiser depuis la bastion.
3. **Automatisation / Sécurité :**
Documenter l’utilisation du **SSH Agent Forwarding** pour chaîner les connexions depuis la VM locale → bastion → EC2 privée.
Créer un script pour lancer automatiquement les vérifications de santé.

🔧 Technologies

• VirtualBox + Ubuntu (local)

• VPC, Subnets privés/publics
• EC2 x3
• SSH, Fail2ban, cron
• Scripts de supervision (bash)
• (Bonus) Netdata
• SSH agent forwarding

1. Générer une paire de clés RSA (AWS-compatible)

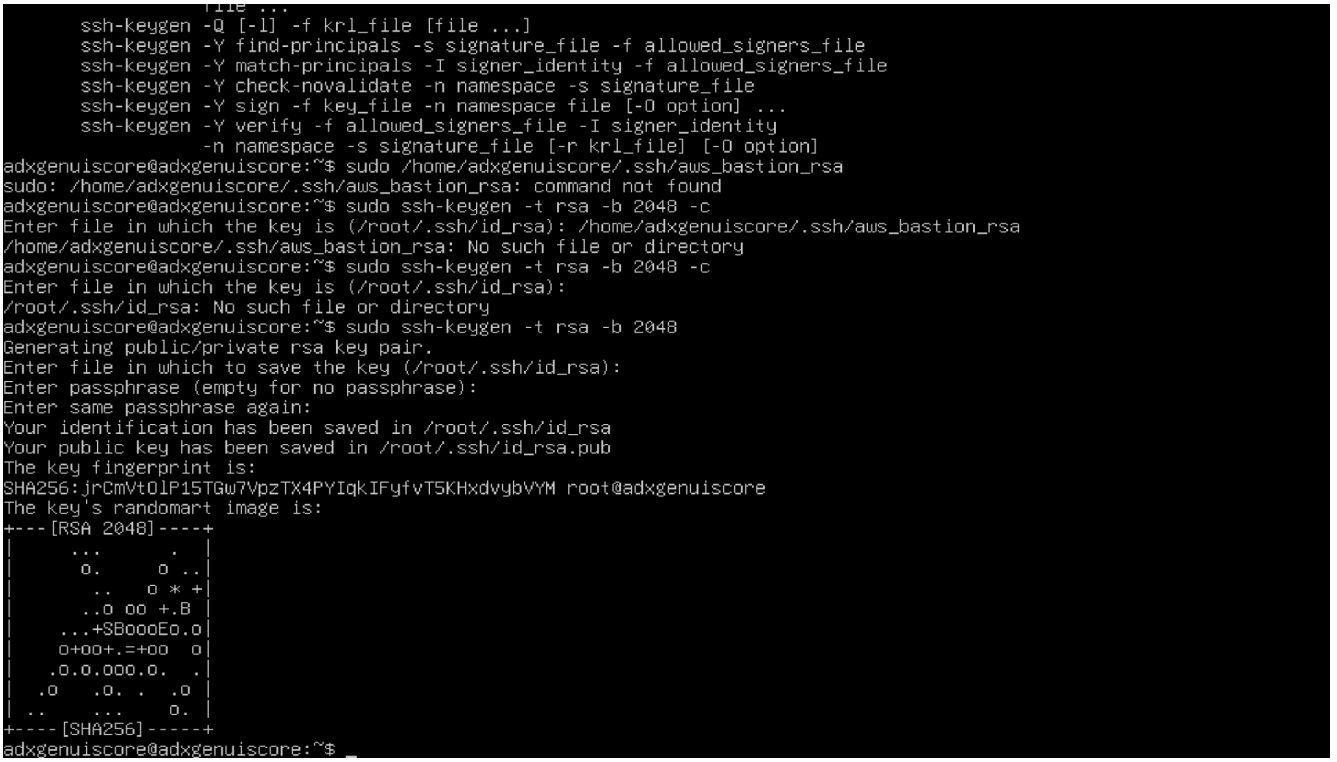
AWS accepte les clés **RSA (2048 bits ou plus)** pour les connexions SSH EC2.

Dans le terminal de ta VM Ubuntu :

bash
CopyEdit
```
ssh-keygen -t rsa -b 2048
```

- `-t rsa` : spécifie RSA comme type de clé.
- `-b 2048` : définit la longueur à 2048 bits (minimum accepté par AWS).

Tu peux appuyer sur **Entrée** pour accepter le chemin par défaut, ou entrer un nom personnalisé comme :



a) Test ping DNS

bash

CopyEdit

```
ping -c 4 aws.amazon.com
```

- Vérifie que la VM peut résoudre un nom de domaine et atteindre AWS.
- `-c 4` envoie seulement 4 paquets.

b) Test HTTP avec `curl`

bash

CopyEdit

```
curl -I https://aws.amazon.com
```

- Vérifie que HTTPS fonctionne depuis la VM.
- L'option `-I` affiche uniquement les en-têtes (ex. : `HTTP/2 200`).

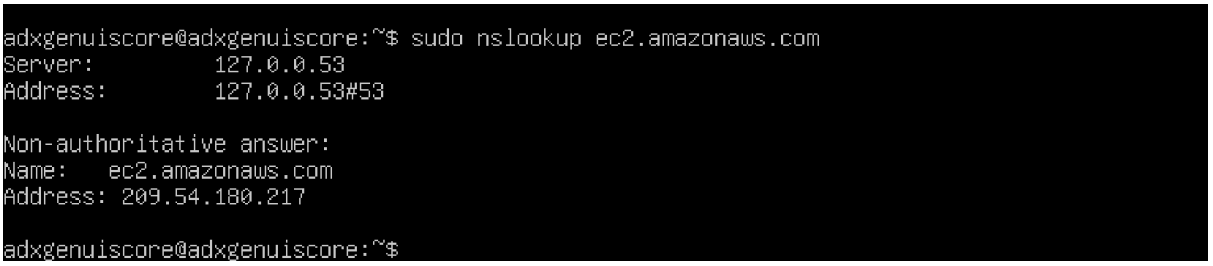
c) Test DNS directement

bash
CopyEdit
`nslookup ec2.amazonaws.com`

- Vérifie que le nom de domaine des services EC2 est bien résolu par un serveur DNS.

Si un test échoue :

- Vérifie dans VirtualBox que la VM est en mode **NAT** ou **Bridged**



- Teste aussi `ip a` pour voir si l'interface a une IP correcte.

4. Préparons les scripts de supervision à exécuter depuis la bastion

But : préparer un script Bash que tu exécutes depuis la bastion pour superviser les EC2 privées.

Créé le fichier :

nano supervise.sh

```
  GNU nano 7.2                                                  supervise.sh
#!/bin/bash

echo "===== Supervision de  $(hostname) - $(date) ====="
echo

echo "=== Uptime ==="
uptime
echo

echo "=== Utilisation CPU (1 sec) ==="
top -bn1 | grep "Cpu(s)"|
echo

echo "=== Utilisation memoire ==="
free -h
echo

echo "=== Espace disque monte ==="
df -h --total
echo

echo "=== Connexions reseau actives (ports ecoutes) ==="
ss -tuln
echo

echo "=== Nombre de connexions etablies ==="
ss _s | grep estab
echo

echo "=== Utilisateurs connectes ==="
who
echo

echo "=== Services actifs ==="
systemctl list-units --type=services --state=running | head -15
echo

echo "=== Derniers journaux systeme ( 5 ligne) ==="
journalctl -n 5 --no-pager
echo

echo "=== Fin de supervision ==="
```

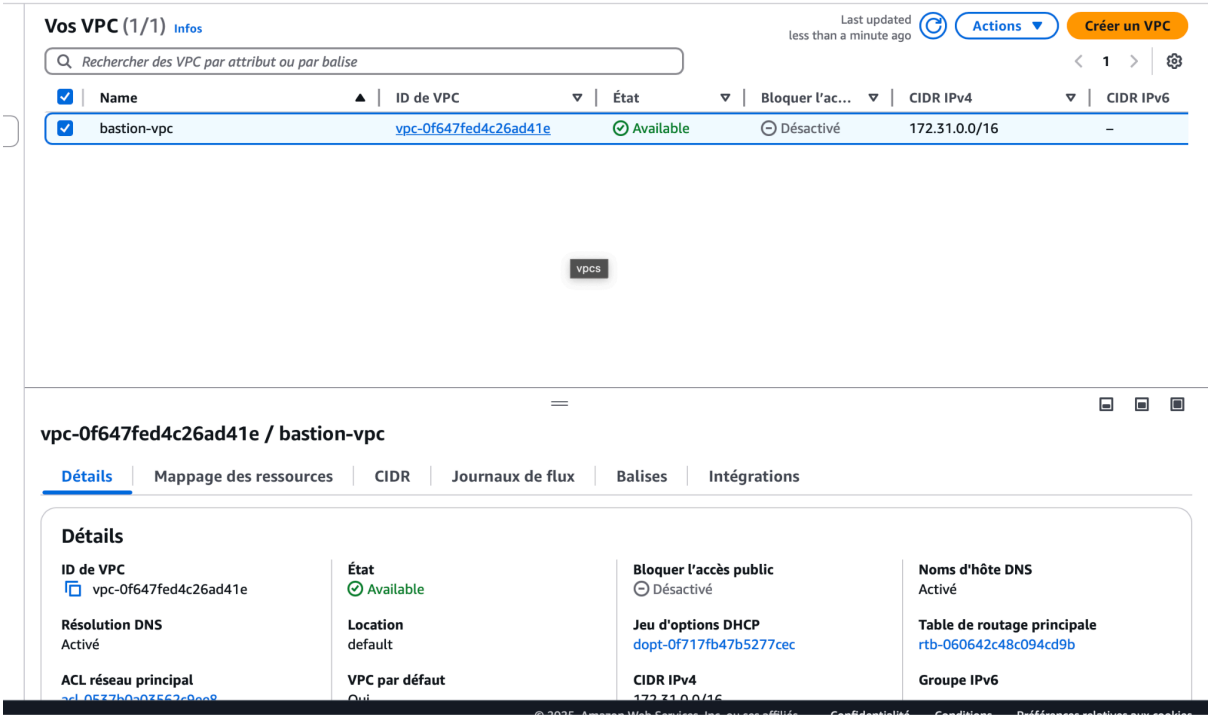
```
               total        used        free      shared  buff/cache   available
Mem:           3.8Gi       696Mi       2.7Gi       5.3Mi       578Mi       3.1Gi
Swap:          3.8Gi          0B       3.8Gi

=== Espace disque monte ===
Filesystem                         Size  Used Avail Use% Mounted on
tmpfs                              391M  1.4M  389M   1% /run
efivarfs                           256K   48K  209K  19% /sys/firmware/efi/efivars
/dev/mapper/ubuntu--vg-ubuntu--lv   30G  7.9G   21G  28% /
tmpfs                              2.0G     0  2.0G   0% /dev/shm
tmpfs                              5.0M     0  5.0M   0% /run/lock
/dev/vda2                          2.0G  191M  1.6G  11% /boot
/dev/vda1                          1.1G  6.4M  1.1G   1% /boot/efi
tmpfs                              391M   12K  391M   1% /run/user/1000
total                               36G  8.0G   26G  24% -

=== Connexions reseau actives (ports ecoutes) ===
Netid       State         Recv-Q        Send-Q                                  Local Address:Port                 Peer Address:Port        Process
udp         UNCONN        0             0                                          127.0.0.54:53                      0.0.0.0:*
udp         UNCONN        0             0                                       127.0.0.53%lo:53                      0.0.0.0:*
udp         UNCONN        0             0                                  192.168.64.4%enp0s1:68                     0.0.0.0:*
udp         UNCONN        0             0                       [fe80::8476:41ff:fe6f:8f0]%enp0s1:546                    [::]:*
udp         UNCONN        0             0                        [fe80::d2:21ff:fe76:335a]%enp0s2:546                    [::]:*
tcp         LISTEN        0             4096                                       127.0.0.54:53                      0.0.0.0:*
tcp         LISTEN        0             4096                                    127.0.0.53%lo:53                      0.0.0.0:*
tcp         LISTEN        0             151                                        0.0.0.0:3306                      0.0.0.0:*
tcp         LISTEN        0             70                                      127.0.0.1:33060                     0.0.0.0:*
tcp         LISTEN        0             4096                                          *:22                                *:*
tcp         LISTEN        0             511                                           *:80                                *:*

=== Nombre de connexions etablies ===
Error: an inet prefix is expected rather than "_s".
Cannot parse dst/src address.

=== Utilisateurs connectes ===
adxgenuiscore tty1         2025-05-08 22:58

=== Services actifs ===
Unknown unit type or load state 'services'.
Use -t help to see a list of allowed values.

=== Derniers journaux systeme ( 5 ligne) ===
May 09 10:14:56 adxgenuiscore sudo[2373]: pam_unix(sudo:session): session opened for user root(uid=0) by adxgenuiscore(uid=1000)
May 09 10:15:01 adxgenuiscore CRON[2376]: pam_unix(cron:session): session opened for user root(uid=0) by root(uid=0)
May 09 10:15:01 adxgenuiscore CRON[2377]: (root) CMD (command -v debian-sa1 > /dev/null && debian-sa1 1 1)
May 09 10:15:01 adxgenuiscore CRON[2376]: pam_unix(cron:session): session closed for user root
May 09 10:15:47 adxgenuiscore sudo[2373]: pam_unix(sudo:session): session closed for user root

=== Fin de supervision ===
adxgenuiscore@adxgenuiscore:~$ ./supervise.sh
```

2. Mise en place sur AWS

Étape 1 : Créer un VPC personnalisé

1. Va dans **VPC > Your VPCs > Create VPC**
2. Choisis **VPC only**
3. Paramètres :
    - Nom : `bastion-vpc`
    - IPv4 CIDR : `10.0.0.0/16`
    - Laisse IPv6 désactivé
    - DNS hostnames : **Activé**
4. Clique sur **Create VPC**

ssh -i "adxgenuiscore.pem" ubuntu@ec2-18-199-84-28.eu-central-1.compute.amazonaws.com



Étape 2 : Créer 2 sous-réseaux

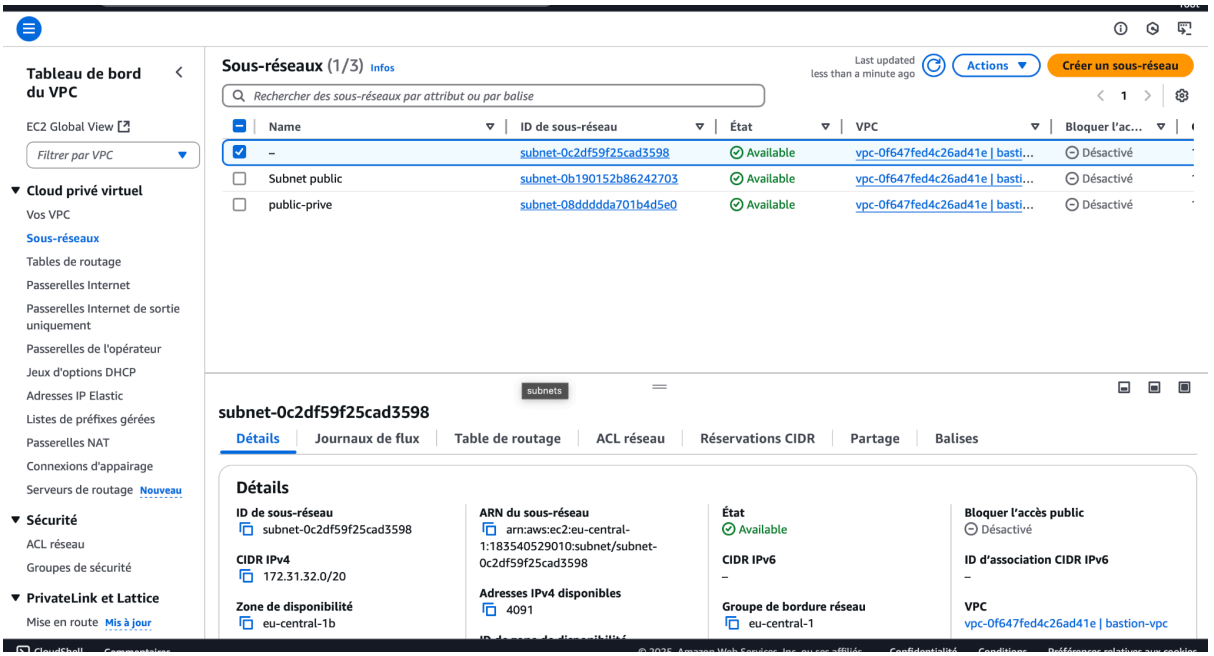
a) Subnet public (pour la bastion)

1. Va dans **Subnets > Create subnet**
2. Sélectionne le VPC `bastion-vpc`
3. Nom : `public-subnet`
4. AZ : ex `eu-west-3a`
5. CIDR : `10.0.1.0/24`

6. Crée le subnet

b) Subnet privé (pour les 2 EC2 privées)

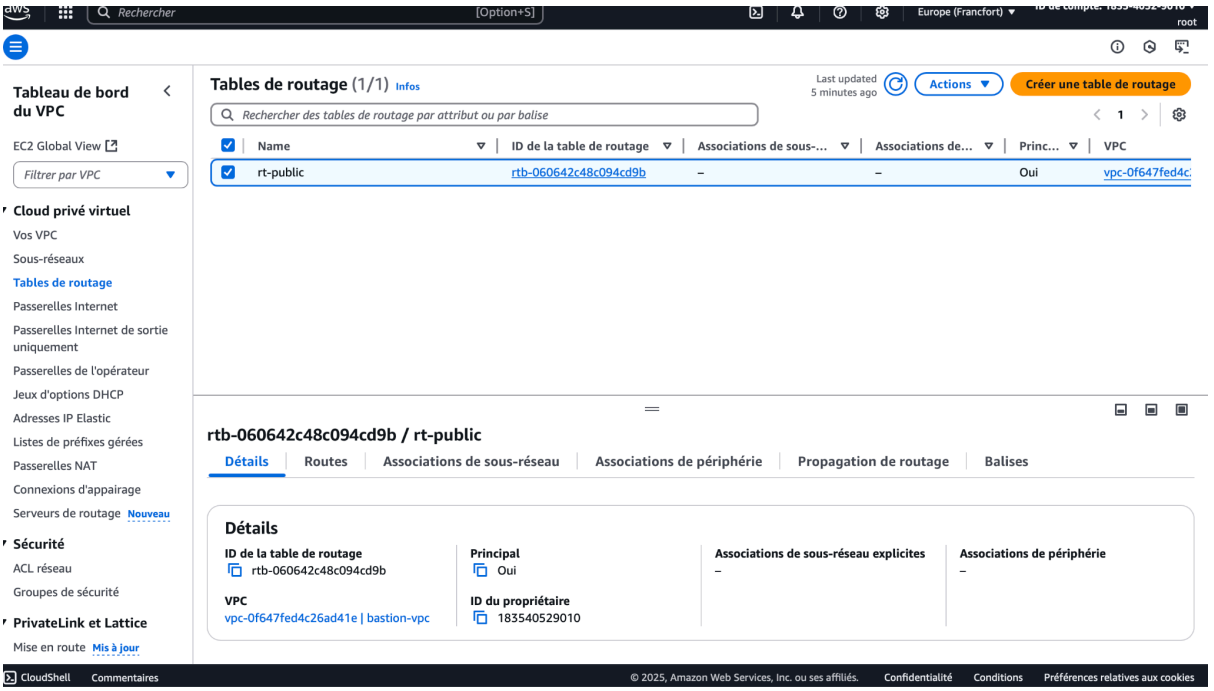
1. Nom : `private-subnet`
2. AZ : même ou autre que le public
3. CIDR : `10.0.2.0/24`
4. Crée le subnet



Étape 4 : Table de routage pour le public

1. Va dans **Route Tables > Create**
2. Nom : `rt-public`
3. Associe-la à `bastion-vpc`
4. Édite les routes :
    - Destination : `0.0.0.0/0`
    - Target : l'**Internet Gateway**
5. Clique sur **Subnet associations** > Associer à `public-subnet`

Le subnet privé n'a **pas de route vers Internet**



Étape 7 : Connexion Bastion → EC2 privées

1. Depuis ta VM locale :

   bash

   CopyEdit

   ```
   ssh -i ~/.ssh/aws_bastion_rsa ubuntu@<IP-PUBLIQUE_BASTION>
   ```

2. Sur la bastion, connecte-toi aux instances privées :

   bash

   CopyEdit

   ```
   ssh ubuntu@10.0.2.10
   ```

   ```
   ssh ubuntu@10.0.2.11
   ```

Tu dois copier la **même clé publique** sur `/home/ubuntu/.ssh/authorized_keys` dans les EC2 privées si ce n'est pas fait.

Étape 8 : Installer scripts de supervision sur la bastion

Depuis la bastion :

bash

CopyEdit

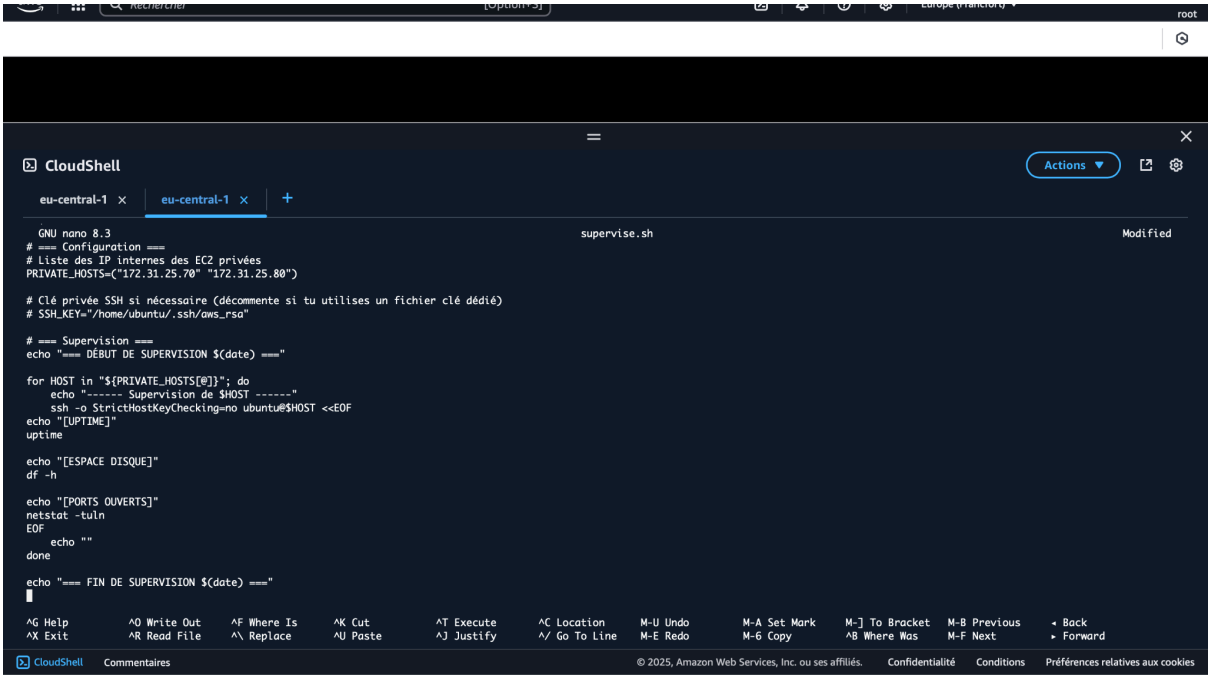
```
nano supervise.sh
```

Colle le script (identique à celui que tu as préparé en local), et rends-le exécutable :

bash

CopyEdit

```
chmod +x supervise.sh
./supervise.sh
```



Étape 9 : Activer Fail2ban sur la bastion

bash

CopyEdit

```
sudo apt update && sudo apt install -y fail2ban

sudo systemctl enable --now fail2ban
```

Optionnel : éditer `/etc/fail2ban/jail.local` pour ajuster les règles (ex. pour sshd).

Installer Netdata sur les EC2 privées

Depuis la bastion, SSH dans chaque EC2 privée :

bash

CopyEdit

```
bash <(curl -Ss https://my-netdata.io/kickstart.sh)
```

Netdata écoute sur le port 19999 (en local), donc tu peux y accéder en **SSH port forwarding** si nécessaire.

✅ Résumé de l'architecture

text

CopyEdit

```
[ VM Ubuntu Locale ]
        |
     SSH (clé RSA)
        |
  [ Bastion EC2 Publique ] ----> [ EC2 privée 1 ]
              |                  [ EC2 privée 2 ]
              |
          Scripts de supervision + fail2ban
```

2. Sur AWS

✅ Créer le VPC

- CIDR VPC : `10.0.0.0/16`
- Subnet public : `10.0.1.0/24`
- Subnet privé : `10.0.2.0/24`

✅ Lancer les instances EC2

- Bastion (Ubuntu) dans le **subnet public**
- Deux EC2 privées (Ubuntu) dans le **subnet privé**

✅ Sécuriser la Bastion Box (Security Group)

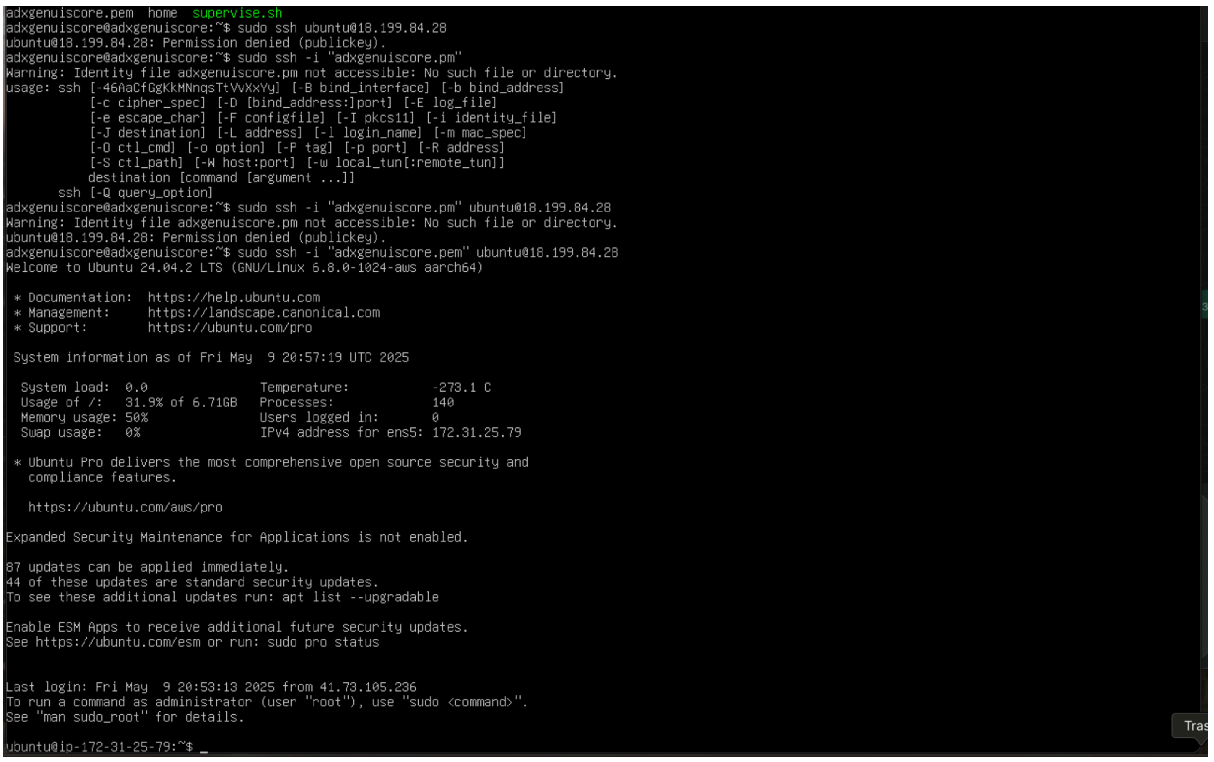
- Port 22 autorisé **uniquement depuis l'IP publique de la VM locale**
  - À trouver avec : `curl ifconfig.me`
- Bloquer tout autre accès non autorisé.

✅ Connexion SSH locale → Bastion

bash

CopyEdit

```
ssh -i ~/.ssh/id_rsa ubuntu@<bastion-public-ip>
```



✅ SSH interne Bastion → EC2 privées

- Sur Bastion : placer la **même clé privée** (`id_rsa`) ou transférer une autre autorisée sur les EC2 privées.
- Exemple :

  bash

  CopyEdit

  ```
  ssh -i ~/.ssh/id_rsa ubuntu@ip
  ```

✅ Scripts de supervision depuis Bastion vers EC2 privées

Exécuter le script avec :

bash

CopyEdit

```
ssh ubuntu@10.0.2.10 'bash -s' < ./scripts/supervision.sh
```

✅ Activer Fail2Ban sur la Bastion

bash

CopyEdit

```
sudo apt update && sudo apt install fail2ban -y

sudo systemctl enable fail2ban

sudo systemctl start fail2ban
```

Netdata sur EC2 privées

Installation :

bash

CopyEdit

```
bash <(curl -Ss https://my-netdata.io/kickstart.sh)
```

Visualiser depuis Bastion :

- Se connecter à l'IP privée : `http://10.0.2.10:19999`
- Utiliser un tunnel SSH pour l'accès :

bash

CopyEdit

```
ssh -L 19999:localhost:19999 ubuntu@ip
```

3. Automatisation / Sécurité

✅ SSH Agent Forwarding

Sur ta VM locale :

bash

CopyEdit

```
eval "$(ssh-agent -s)"

ssh-add ~/.ssh/id_rsa

ssh -A ubuntu@<bastion-public-ip>

# Puis depuis Bastion :

ssh ubuntu@ip
```

✅ Script de supervision automatique (depuis Bastion)

Créer `check_all.sh` :

bash

CopyEdit

```
#!/bin/bash
```

```
for ip in 10.0.2.10 10.0.2.11; do
  echo "Checking $ip..."
  ssh ubuntu@$ip 'bash -s' < ./scripts/supervision.sh
done
```