



ÉCOLE MOHAMMADIA D'INGÉNIEURS

ATELIER DEVOPS / CI-CD
AUTOMATISATION DE DÉPLOIEMENT : ANSIBLE & JENKINS

Rapport d'Atelier — Déploiement WAR avec Ansible et exécution via Jenkins

Élèves :
Sami FAOUZI

Encadré par :
Asmaa RETBI

2ème année Génie Informatique
23 décembre 2025

Table des matières

1 Tentative initiale sous Windows + VirtualBox (non finalisée)	2
1.1 Installation et validation d'Ansible sur la VM Ubuntu (VirtualBox)	2
1.2 Création de l'utilisateur <code>user_sup</code> côté VM Ubuntu	3
1.3 Préparation du côté Windows (compte + droits + SSH Server)	3
1.4 Mise en place d'authentification par clé SSH (VM → Windows)	5
1.5 Limite rencontrée et décision de bascule	6
2 Préparation de l'Atelier	6
2.1 Objectif	6
2.2 Architecture Utilisée	6
3 Étape 0 : Création de la VM et Installation d'Ansible	7
3.1 Création de la Machine Virtuelle (KVM / virt-manager)	7
3.2 Installation d'Ubuntu Server	7
3.3 Installation d'Ansible sur la VM	7
4 Étape 1 : Création du Super Utilisateur (<code>user_sup</code>)	7
4.1 Sur la VM (Ansible)	7
4.2 Sur le Host Ubuntu	8
5 Étapes 2–3 : Configuration SSH (VM → Host)	8
5.1 Activation du serveur SSH sur le Host	8
5.2 Génération et copie de clé SSH	9
6 Étape 4 : Inventory Ansible + Tests	9
6.1 Création du fichier inventory	9
6.2 Test de connectivité (ping Ansible)	9
6.3 Test de validation (shell)	9
7 Étapes 5–6 : Playbook de Déploiement WAR	10
7.1 Préparation du dossier webapps (Host)	10
7.2 WAR de test	10
7.3 Playbook deploy-war.yml	10
7.4 Exécution du playbook	12
7.5 Vérification du déploiement	12
8 Étape 7 : Jenkins et Pipeline Ansible	12
8.1 Installation du plugin Ansible	12
8.2 Jenkinsfile (pipeline)	13
8.3 Résultat d'exécution (console)	14
8.4 Vérification finale du WAR (via pipeline Jenkins)	14
9 Observations et Difficultés Rencontrées	14

1 Tentative initiale sous Windows + VirtualBox (non finalisée)

1.1 Installation et validation d'Ansible sur la VM Ubuntu (VirtualBox)

Après installation des paquets nécessaires, Ansible a été vérifié via la commande `ansible -version`.

```

Ubuntu-Ansible2 [Running] - Oracle VirtualBox

File Machine View Input Devices Help

Selecting previously unselected package python3-libcloud.
Preparing to unpack .../10-python3-libcloud_3.4.1-5_all.deb
Unpacking python3-libcloud (3.4.1-5) ...
Selecting previously unselected package python3-ntlm-auth.
Preparing to unpack .../11-python3-ntlm-auth_1.5.0-1_all.deb
Unpacking python3-ntlm-auth (1.5.0-1) ...
Selecting previously unselected package python3-requests-ntlm.
Preparing to unpack .../12-python3-requests-ntlm_1.1.0-3_all.deb
Unpacking python3-requests-ntlm (1.1.0-3) ...
Selecting previously unselected package python3-selinux.
Preparing to unpack .../13-python3-selinux_3.5-2ubuntu2.1_all.deb
Unpacking python3-selinux (3.5-2ubuntu2.1) ...
Selecting previously unselected package python3-xmldict.
Preparing to unpack .../14-python3-xmldict_0.13.0-1ubuntu0.24.04.1_all.deb
Unpacking python3-xmldict (0.13.0-1ubuntu0.24.04.1) ...
Selecting previously unselected package python3-winrm.
Preparing to unpack .../15-python3-winrm_0.4.3-2_all.deb
Unpacking python3-winrm (0.4.3-2) ...
Selecting previously unselected package python3-passlib.
Preparing to unpack .../16-python3-passlib_1.7.4-4_all.deb
Unpacking python3-passlib (1.7.4-4) ...
Setting up python3-lockfile (1:0.12.2-3) ...
Setting up python3-passlib (1.7.4-4) ...
Setting up python3-ntlm-auth (1.5.0-1) ...
Setting up python3-resolvelib (1.0.1-1) ...
Setting up python3-kerberos (1.1.14-3.1build9) ...
Setting up python3-simplejson (3.19.2-1build2) ...
Setting up python3-xmldict (0.13.0-1ubuntu0.24.04.1) ...
Setting up python3-packaging (24.0-1) ...
Setting up python3-jmespath (1.0.1-1) ...
Setting up python3-dnspython (2.6.1-1ubuntu1) ...
Setting up python3-selinux (3.5-2ubuntu2.1) ...
Setting up python3-argcomplete (3.1.4-1ubuntu0.1) ...
Setting up python3-requests-ntlm (1.1.0-3) ...
Setting up ansible-core (2.16.3-0ubuntu2) ...
Setting up python3-libcloud (3.4.1-5) ...
Setting up python3-winrm (0.4.3-2) ...
Setting up ansible (9.2.0+dfsg-0ubuntu5) ...
Processing triggers for man-db (2.12.0-4build2) ...
vboxuser@Ubuntu-Ansible2:~$ ansible --version
ansible [core 2.16.3]
  config file = None
  configured module search path = ['/home/vboxuser/.ansible']
  ansible python module location = /usr/lib/python3/dist-packages/ansible
  ansible collection location = /home/vboxuser/.ansible/collections
  executable location = /usr/bin/ansible
  python version = 3.12.3 (main, Nov  6 2025, 13:44:16) [GCC 11.2.0]
  jinja version = 3.1.2
  libyaml = True
vboxuser@Ubuntu-Ansible2:~$ _

```

FIGURE 1 – Installation d'Ansible et exécution de `ansible -version` (VM VirtualBox)

```

processing triggers for man-db (2.12.0-2ubuntu2) ...
vboxuser@Ubuntu-Ansible2:~$ ansible --version
ansible [core 2.16.3]
  config file = None
  configured module search path = ['/home/vboxuser/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3/dist-packages/ansible
  ansible collection location = /home/vboxuser/.ansible/collections:/usr/share/ansible/collections
  executable location = /usr/bin/ansible
  python version = 3.12.3 (main, Nov  6 2025, 13:44:16) [GCC 13.3.0] (/usr/bin/python3)
  jinja version = 3.1.2
  libyaml = True
vboxuser@Ubuntu-Ansible2:~$
```

FIGURE 2 – Détail : sortie de `ansible -version` confirmant l’installation

1.2 Création de l’utilisateur `user_sup` côté VM Ubuntu

Un utilisateur `user_sup` a été créé sur la VM, puis ajouté au groupe `sudo` et validé via les commandes de contrôle.

```

Changing the user information for user_sup
Enter the new value, or press ENTER for the default
  Full Name []:
  Room Number []:
  Work Phone []:
  Home Phone []:
  Other []:
Is the information correct? [Y/n] y
info: Adding new user `user_sup' to supplemental / extra groups `users' ...
info: Adding user `user_sup' to group `users' ...
vboxuser@Ubuntu-Ansible2:~$ id user_sup
uid=1001(user_sup) gid=1001(user_sup) groups=1001(user_sup),100(users)
```

FIGURE 3 – Création de `user_sup` sur la VM et vérification via `id user_sup`

```

vboxuser@Ubuntu-Ansible2:~$ sudo usermod -aG sudo user_sup
vboxuser@Ubuntu-Ansible2:~$ groups user_sup
user_sup : user_sup sudo users
vboxuser@Ubuntu-Ansible2:~$ sudo -l -U user_sup
Matching Defaults entries for user_sup on sami:
  env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin, u
User user_sup may run the following commands on sami:
  (ALL : ALL) ALL
vboxuser@Ubuntu-Ansible2:~$ _
```

FIGURE 4 – Ajout au groupe `sudo` et validation des droits (commande `sudo -l -U`)

1.3 Préparation du côté Windows (compte + droits + SSH Server)

Pour permettre la connexion distante depuis la VM Ubuntu, un compte `user_sup` a été préparé sur Windows, puis ajouté au groupe **Administrateurs**. Ensuite, le service **OpenSSH Server** a été activé et la règle pare-feu vérifiée. Enfin, le port 22 a été contrôlé en écoute.

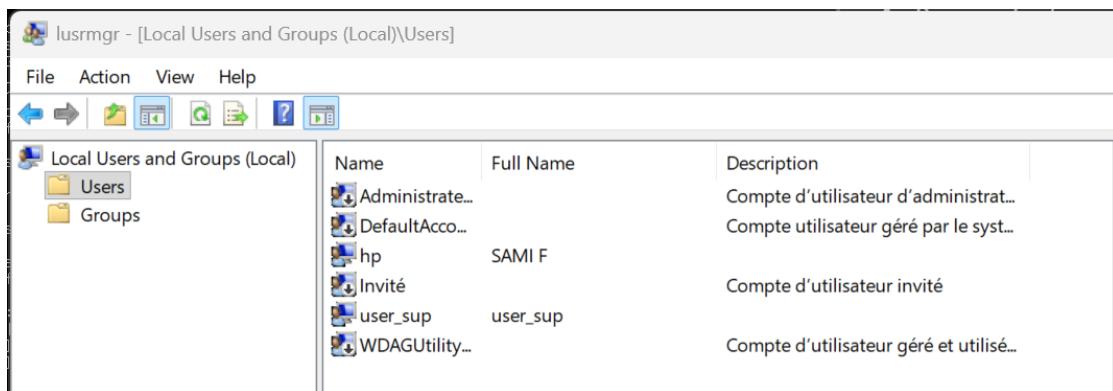


FIGURE 5 – Création du compte `user_sup` sur Windows (Local Users and Groups)

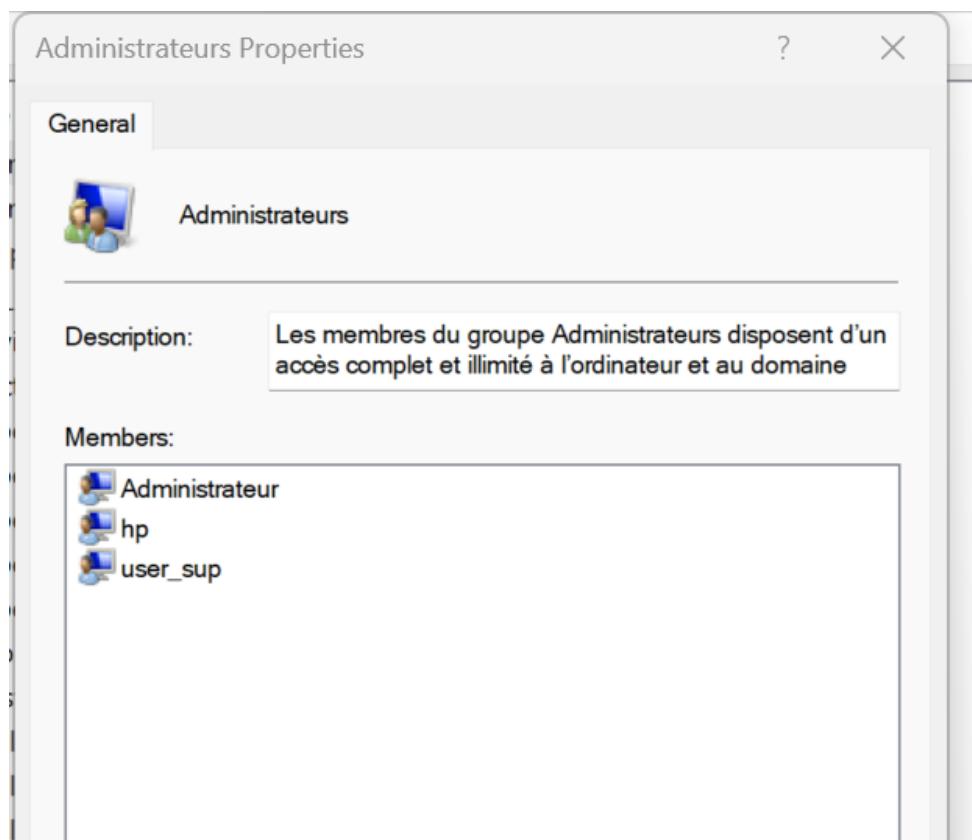


FIGURE 6 – Ajout de `user_sup` au groupe Administrateurs

```

Administrator: Windows PowerShell
PS C:\WINDOWS\system32> Start-Service sshd
>> Set-Service -Name sshd -StartupType 'Automatic'
>>
WARNING: Waiting for service 'OpenSSH SSH Server (sshd)' to start...
PS C:\WINDOWS\system32> Get-NetFirewallRule -Name *OpenSSH-Server*

Name : OpenSSH-Server-In-TCP
DisplayName : OpenSSH SSH Server (sshd)
Description : Inbound rule for OpenSSH SSH Server (sshd)
DisplayGroup : OpenSSH Server
Group : OpenSSH Server
Enabled : True
Profile : Private
Platform :
Direction : Inbound
Action : Allow
EdgeTraversalPolicy : Block
LooseSourceMapping : False
LocalOnlyMapping : False
Owner :
PrimaryStatus : OK
Status : The rule was parsed successfully from the store.
EnforcementStatus : NotApplicable
PolicyStoreSource : PersistentStore
PolicyStoreSourceType : Local
RemoteDynamicKeywordAddresses : {}
PolicyAppId :
PackageName :

```

FIGURE 7 – Activation du service SSH (sshd) et vérification des règles pare-feu

```

PS C:\WINDOWS\system32> Get-NetTCPConnection -State Listen -LocalPort 22
>>
LocalAddress          LocalPort RemoteAddress           RemotePort State      AppliedSetting
----:----              ----:---- :----:----           ----:---- ----:---- -----
:::                     22       :::                  0         Listen
0.0.0.0               22       0.0.0.0            0         Listen

```

FIGURE 8 – Preuve : le port 22 est en état Listen côté Windows

1.4 Mise en place d’authentification par clé SSH (VM → Windows)

Une clé SSH a été générée sur la VM Ubuntu, puis la clé publique a été ajoutée au fichier `authorized_keys` dans le profil `C:\Users\user_sup\.ssh`.

```

Processing triggers for man-db (2.12.0-4build2) ...
vboxuser@Ubuntu-Ansible2:~$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/vboxuser/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/vboxuser/.ssh/id_ed25519
Your public key has been saved in /home/vboxuser/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:W0Nd3tJ9gtyUpqVqfZVGWPBdqyZ65hukSF18VnYFNQ vboxuser@sami
The key's randomart image is:
+-- [ED25519 256] --+
... =*o..|
o...=.BE=o|
o + +=B *|
. ... .o..o.|
.Soo oo .|
oo =. +
..+. o
.+. .
+*o+.
+--- [SHA256] -----+
vboxuser@Ubuntu-Ansible2:~$ cat ~/.ssh/id_ed25519.pub
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIMuZXHeCRfsZ31BP6wa2qBxNpbRiFHILUcI9qUc6M2w vboxuser@sami
vboxuser@Ubuntu-Ansible2:~$
```

FIGURE 9 – Génération de clé ed25519 et affichage de la clé publique

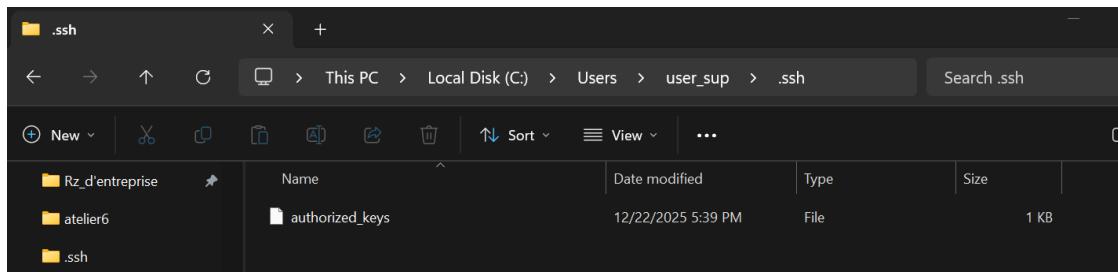


FIGURE 10 – Fichier authorized_keys créé côté Windows pour user_sup

1.5 Limite rencontrée et décision de bascule

Malgré ces étapes de préparation (*Ansible installé, compte créé, SSH activé, port 22 en écoute, clé configurée*), la suite du travail n'a pas pu être finalisée sous Windows/VirtualBox à cause de **problèmes récurrents de la VM** (instabilités et blocages répétés).

Ubuntu (solution stable), comme présenté dans la partie suivante du rapport.

2 Préparation de l'Atelier

2.1 Objectif

L'objectif de cet atelier est de mettre en place une infrastructure d'automatisation de déploiement utilisant Ansible comme outil de configuration et Jenkins pour l'orchestration via pipeline CI/CD.

2.2 Architecture Utilisée

- **Machine Contrôleur (VM Ansible)** : Ubuntu Server 24.04 LTS sur KVM/-QEMU
- **Machine Cible (Host Ubuntu)** : Ubuntu Desktop (machine physique)

- IP VM : 192.168.122.61
- IP Host : 192.168.122.1 (virbr0)
- Utilisateur SSH : user_sup

3 Étape 0 : Création de la VM et Installation d'Ansible

3.1 Crédit de la Machine Virtuelle (KVM / virt-manager)

```

1 sudo apt install qemu-kvm libvirt-daemon-system libvirt-clients bridge-
  utils virt-manager -y
2 sudo systemctl enable --now libvird
3 sudo usermod -aG libvirt $USER
4 sudo usermod -aG kvm $USER

```

Listing 1 – Installation de KVM

3.2 Installation d'Ubuntu Server

- ISO utilisé : Ubuntu Server 24.04.3 LTS
- RAM allouée : 2 GB
- Disque : 25 GB
- Réseau : NAT via virbr0 (192.168.122.0/24)

3.3 Installation d'Ansible sur la VM

```

1 sudo apt update
2 sudo apt install ansible -y
3 ansible --version

```

Listing 2 – Installation Ansible sur la VM

```

user_sup@ansible-vm:~$ ansible --version
ansible [core 2.16.3]
  config file = None
  configured module search path = ['/home/user_sup/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3/dist-packages/ansible
  ansible collection location = /home/user_sup/.ansible/collections:/usr/share/ansible/collections
  executable location = /usr/bin/ansible
  python version = 3.12.3 (main, Nov  6 2025, 13:44:16) [GCC 13.3.0] (/usr/bin/python3)
  jinja version = 3.1.2
  libyaml = True

```

FIGURE 11 – Ansible installé et fonctionnel sur la VM

4 Étape 1 : Création du Super Utilisateur (user_sup)

4.1 Sur la VM (Ansible)

L'utilisateur user_sup a été créé lors de l'installation avec les droits sudo.

```
user_sup@ansible-vm:~$ groups
user_sup adm cdrom sudo dip plugdev lxd
user_sup@ansible-vm:~$ sudo -l -U user_sup
Matching Defaults entries for user_sup on ansible-vm:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/sbin\:/bin\:/snap/bin, use_pty

User user_sup may run the following commands on ansible-vm:
    (ALL : ALL) ALL
user_sup@ansible-vm:~$ _
```

FIGURE 12 – Création de user_sup avec droits sudo sur la VM

4.2 Sur le Host Ubuntu

```
1 sudo adduser user_sup
2 sudo usermod -aG sudo user_sup
```

Listing 3 – Création user_sup sur le Host

```
sami@FAOUIZI:~$ sudo -l -U user_sup
Matching Defaults entries for user_sup on FAOUIZI:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin,
    use_pty

User user_sup may run the following commands on FAOUIZI:
    (ALL : ALL) ALL
sami@FAOUIZI:~$ _
```

FIGURE 13 – Création de user_sup avec droits sudo sur la machine Host

5 Étapes 2–3 : Configuration SSH (VM → Host)

5.1 Activation du serveur SSH sur le Host

```
1 sudo apt install openssh-server -y
2 sudo systemctl enable --now ssh
3 sudo ss -tlnp | grep :22
```

Listing 4 – Installation et activation SSH

```
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
openssh-server is already the newest version (1:9.6p1-3ubuntu13.14).
0 upgraded, 0 newly installed, 0 to remove and 83 not upgraded.
Synchronizing state of ssh.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable ssh
LISTEN 0      4096          0.0.0.0:22          0.0.0.0:*      users:(("sshd",pid=17569,fd=3),("systemd",pid=1,fd=487))
LISTEN 0      4096          [::]:22            [::]:*      users:(("sshd",pid=17569,fd=4),("systemd",pid=1,fd=489))
sami@FAOUIZI:~$ _
```

FIGURE 14 – Activation du serveur SSH sur le Host (port 22)

```
user_sup@ansible-vm:~$ ssh user_sup@192.168.122.1 "ss -tlnp | grep :22"
LISTEN 0      4096          0.0.0.0:22          0.0.0.0:*
LISTEN 0      4096          [::]:22            [::]:*
```

FIGURE 15 – Preuve que SSH est actif sur le Host (écoute sur le port 22)

5.2 Génération et copie de clé SSH

```

1 ssh-keygen -t rsa -b 4096
2 ssh-copy-id user_sup@192.168.122.1

```

Listing 5 – Génération + copie de clé SSH

```

user_sup@ansible-vm:~$ ssh user_sup@192.168.122.1 "whoami"
user_sup
user_sup@ansible-vm:~$ _

```

FIGURE 16 – Preuve de connexion SSH de la VM vers le Host

6 Étape 4 : Inventory Ansible + Tests

6.1 Création du fichier inventory

```

1 sudo mkdir -p /etc/ansible
2 echo "[ubuntu_hosts]" > ~/inventory
3 echo "192.168.122.1" >> ~/inventory

```

Listing 6 – Crédation de l'inventory

6.2 Test de connectivité (ping Ansible)

```

1 ansible all -i ~/inventory -m ping -u user_sup

```

Listing 7 – Ping Ansible

```

some actions do not make sense in ad-hoc (include, meta, etc)
user_sup@ansible-vm:~$ ansible all -i /etc/ansible/hosts -m ping -u user_sup
192.168.122.1 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
user_sup@ansible-vm:~$ _

```

FIGURE 17 – Résultat du ping Ansible (SUCCESS)

6.3 Test de validation (shell)

```

1 ansible all -i ~/inventory -m shell -a "whoami && hostname" -u user_sup

```

Listing 8 – Commande shell via Ansible

```

user_sup@ansible-vm:~$ cat /etc/ansible/hosts
[ubuntu_hosts]
192.168.122.1 ansible_user=user_sup
user_sup@ansible-vm:~$ ansible all -m shell -a "whoami" -u user_sup
192.168.122.1 | CHANGED | rc=0 >>
user_sup
user_sup@ansible-vm:~$ -

```

FIGURE 18 – Inventory + test de validation (whoami/hostname)

7 Étapes 5–6 : Playbook de Déploiement WAR

7.1 Préparation du dossier webapps (Host)

```

1 sudo mkdir -p /opt/tomcat/webapps
2 sudo chown user_sup:user_sup /opt/tomcat/webapps

```

Listing 9 – Création du dossier webapps

7.2 WAR de test

```

1 echo "Sample WAR content" > ~/app.war

```

Listing 10 – Crédit WAR de test

7.3 Playbook deploy-war.yml

```

1 ---
2 - name: Déploiement WAR vers le serveur
3   hosts: all
4   vars:
5     war_file: /home/user_sup/app.war
6     webapps_dest: /opt/tomcat/webapps/
7
8   tasks:
9     - name: Vérifier que le WAR existe sur Ubuntu
10       local_action:
11         module: stat
12         path: "{{ war_file }}"
13         register: war_check
14
15     - name: Copier le WAR vers webapps via SSH
16       copy:
17         src: "{{ war_file }}"
18         dest: "{{ webapps_dest }}"
19
20     - name: Confirmer le déploiement
21       debug:
22         msg: "OK: WAR déployé avec succès!"

```

Listing 11 – Playbook deploy-war.yml

```

user_sup@ansible-vm:~$ cat ~/deploy-war.yml
---
- name: Deploiement WAR vers le serveur
  hosts: ubuntu_hosts
  vars:
    war_file: /home/user_sup/sample-app.war
    webapps_dest: /opt/tomcat/webapps/

  tasks:
    - name: Vérifier que le WAR existe sur Ubuntu
      local_action:
        module: stat
        path: "{{ war_file }}"
      register: war_check

    - name: Copier le WAR vers webapps via SSH
      copy:
        src: "{{ war_file }}"
        dest: "{{ webapps_dest }}"

    - name: Confirmer le déploiement
      debug:
        msg: "OK: WAR déployé avec succès!"

```

FIGURE 19 – Fichier Playbook (deploy-war.yml)

```

user_sup@ansible-vm:~$ cat ~/deploy-war.yml
---
- name: Deploiement WAR vers le serveur
  hosts: ubuntu_hosts
  vars:
    war_file: /home/user_sup/sample-app.war
    webapps_dest: /opt/tomcat/webapps/

  tasks:
    - name: Vérifier que le WAR existe sur Ubuntu
      local_action:
        module: stat
        path: "{{ war_file }}"
      register: war_check

    - name: Copier le WAR vers webapps via SSH
      copy:
        src: "{{ war_file }}"
        dest: "{{ webapps_dest }}"

    - name: Confirmer le déploiement
      debug:
        msg: "OK: WAR déployé avec succès!"

```

FIGURE 20 – Capture du fichier Playbook

7.4 Exécution du playbook

```
1 ansible-playbook -i ~/inventory ~/deploy-war.yml
```

Listing 12 – Exécution du playbook

```
PLAY [Déploiement WAR vers le serveur] ****
skipping: no hosts matched

PLAY RECAP ****
user_sup@ansible-vm:~$ rm ~/inventory
user_sup@ansible-vm:~$ echo "[ubuntu_hosts]" > ~/inventory
user_sup@ansible-vm:~$ echo '192.168.122.1' >> ~/inventory
user_sup@ansible-vm:~$ rm ~/inventory
user_sup@ansible-vm:~$ echo "[ubuntu_hosts]" > ~/inventory
user_sup@ansible-vm:~$ echo '192.168.122.1' >> ~/inventory
user_sup@ansible-vm:~$ cat ~/inventory
[ubuntu_hosts]
192.168.122.1
user_sup@ansible-vm:~$ ansible-playbook -i ~/inventory ~/deploy-war.yml
PLAY [Déploiement WAR vers le serveur] ****
TASK [Gathering Facts] ****
ok: [192.168.122.1]

TASK [Verifier que le WAR existe sur Ubuntu] ****
ok: [192.168.122.1 -> localhost]

TASK [Copier le WAR vers webapps via SSH] ****
changed: [192.168.122.1]

TASK [Confirmer le déploiement] ****
ok: [192.168.122.1] => {
    "msg": "Ok: WAR deploie avec succès!"
}

PLAY RECAP ****
192.168.122.1 : ok=4    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
user_sup@ansible-vm:~$ _
```

FIGURE 21 – Exécution du playbook (PLAY RECAP)

7.5 Vérification du déploiement

```
1 ls -la /opt/tomcat/webapps/
```

Listing 13 – Vérification sur le Host

```
user_sup@ansible-vm:~$ ansible all -i ~/inventory -m shell -a "ls -la /opt/tomcat/webapps/" -u user_sup
192.168.122.1 | CHANGED | rc=0 >>
total 12
drwxr-xr-x 2 user_sup user_sup 4096 Dec 22 22:18 .
drwxr-xr-x 3 root      root     4096 Dec 22 22:01 ..
-rw-rw-r-- 1 user_sup user_sup  19 Dec 22 22:18 sample-app.war
user_sup@ansible-vm:~$ _
```

FIGURE 22 – Preuve : WAR présent dans /opt/tomcat/webapps

8 Étape 7 : Jenkins et Pipeline Ansible

8.1 Installation du plugin Ansible

1. Accéder à Jenkins : <http://localhost:8080>
2. Manage Jenkins → Plugins → Available plugins
3. Rechercher Ansible et l'installer
4. Redémarrer Jenkins si nécessaire



FIGURE 23 – Plugin Ansible installé sur Jenkins

8.2 Jenkinsfile (pipeline)

```

1 pipeline {
2     agent any
3     stages {
4         stage('Build') {
5             steps {
6                 sh 'echo "WAR content" > app.war'
7             }
8         }
9         stage('Transfer') {
10            steps {
11                sh 'scp -o StrictHostKeyChecking=no app.war user_sup@192
12 .168.122.61:~/app.war'
13            }
14        }
15        stage('Deploy') {
16            steps {
17                sh 'ssh -o StrictHostKeyChecking=no user_sup@192
18 .168.122.61 "ansible-playbook -i ~/inventory ~/deploy-war.yml"'
19            }
20        }
}

```

Listing 14 – Jenkinsfile (Pipeline)

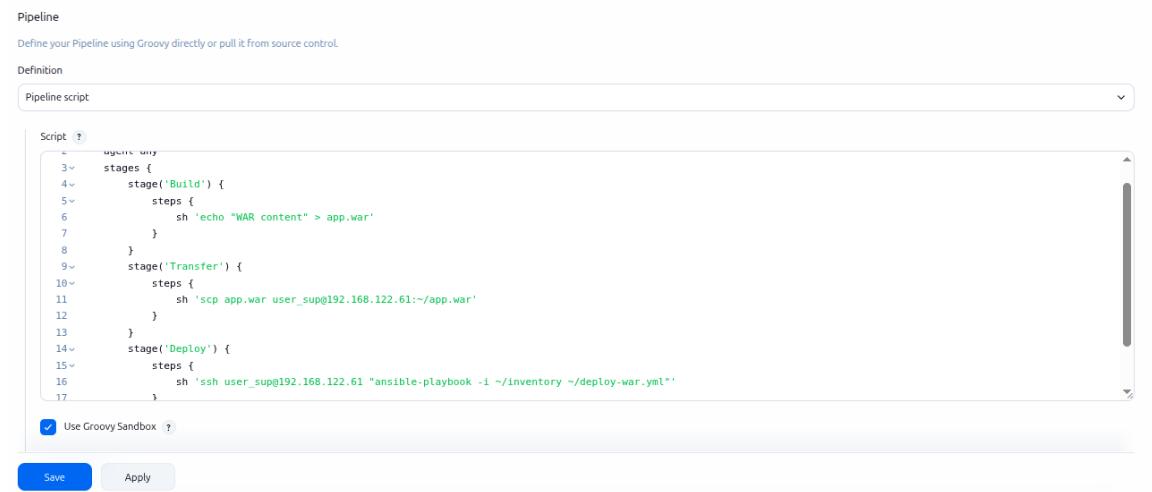


FIGURE 24 – Jenkinsfile : configuration du pipeline

8.3 Résultat d'exécution (console)

The screenshot shows the Jenkins console output for a pipeline named 'deploy-war'. The pipeline consists of several stages: Pipeline Overview, Restart from Stage, Replay, Pipeline Steps, Workspaces, and Previous Build. The main output window displays the command history and execution results. It shows the pipeline performing tasks such as echoing WAR content, copying files via SCP and SSH, gathering facts, verifying the WAR exists on Ubuntu, copying it to webapps, confirming deployment, and finally executing a recap. The output concludes with a summary of statistics: 192.168.122.1 : ok=4 changed=0 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0. The pipeline ends successfully.

```

Pipeline] sh
+ echo WAR content
[Pipeline]
[Pipeline] // stage
[Pipeline] stage
[Pipeline] {
[Pipeline] sh
+ scp -o StrictHostKeyChecking=no app.war user_sup@192.168.122.61:~/app.war
[Pipeline]
[Pipeline] // stage
[Pipeline] stage
[Pipeline] {
[Pipeline] sh
+ ssh -o StrictHostKeyChecking=no user_sup@192.168.122.61 ansible-playbook -i ~/inventory ~/deploy-war.yml

PLAY [Deploiement WAR vers le serveur] *****

TASK [Gathering Facts] *****
ok: [192.168.122.1]

TASK [Verifier que le WAR existe sur Ubuntu] *****
ok: [192.168.122.1 -> localhost]

TASK [Copier le WAR vers webapps via SSH] *****
ok: [192.168.122.1]

TASK [Confirmer le deploiement] *****
ok: [192.168.122.1] => {
    "msg": "OK: WAR deploie avec succes!"
}

PLAY RECAP *****
192.168.122.1 : ok=4    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[Pipeline]
[Pipeline] // stage
[Pipeline]
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS

```

FIGURE 25 – Console Output Jenkins : SUCCESS

8.4 Vérification finale du WAR (via pipeline Jenkins)

```

sami@FAOUIZI:~$ ls -la /opt/tomcat/webapps/
total 12
drwxr-xr-x 2 user_sup user_sup 4096 Dec 22 22:18 .
drwxr-xr-x 3 root      root     4096 Dec 22 22:01 ..
-rw-rw-r-- 1 user_sup user_sup   19 Dec 22 22:18 sample-app.war
sami@FAOUIZI:~$ 

```

FIGURE 26 – WAR présent dans webapps après exécution Jenkins

9 Observations et Difficultés Rencontrées

- VirtualBox vs KVM** : VirtualBox posait problème (conflit/contraintes). Solution : utilisation de KVM/virt-manager.
- Libvirt daemon** : service `libvirtd` non démarré initialement \Rightarrow activation manuelle.
- /etc/ansible inexistant** : création manuelle, puis utilisation d'un inventory local `~/inventory`.

4. **Format inventory** : erreurs de parsing corrigées en écrivant le fichier ligne par ligne.
5. **SSH Jenkins** : erreurs “Host key verification failed” et “Permission denied” \Rightarrow réglées via clés SSH + options SSH.

— Fin du Rapport —