

Ansible est un outil open source installable sous tous les OS et permettant de faire de l'industrialisation de configuration sur des machines virtuelles.

I Configure a Host Windows for ansible with WinRM

Prérequis :

- PowerShell 3+
- .NET 4.0 +
- WinRM ou OpenSSH

On va utiliser WinRm pour une authentification basic

a) Création d'un utilisateur local dans la machine Windows pour ansible

On va configurer un host Windows pour ansible avec le module WinRm (Service gestion WSM)

- Dans gestion d'ordinateur on va créer un utilisateur local

Rédigé par : [Artur Lambo DevOps engineer](#)

Nouvel utilisateur

Nom d'utilisateur : ansibleWinRm

Nom complet : ansibleWinRm

Description : Utilisateur local pour les tests ansible

Mot de passe :

Confirmer le mot de passe :

☐ L'utilisateur doit changer le mot de passe à la prochaine ouverture de session

☐ L'utilisateur ne peut pas changer de mot de passe

☒ Le mot de passe n'expire jamais

☐ Le compte est désactivé

Aide Créer Fermer

Username : ansibleWinRm

Password : !!Art94721805&

- Rajouter cet utilisateur dans le groupe des administrateurs

b) Vérifier la compatibilité de PowerShell et .net sur le serveur Windows

- Dans PowerShell on vérifie sa version comme dit ça ne peut fonctionner que si la version de PowerShell est sup à 3.0

```
PS C:\Users\Artur Lambo> $PSVersionTable

Name                           Value
----                           -
PSVersion                      7.4.2
PSEdition                     Core
BuildCommitId                  7.4.2
OS                             Microsoft Windows 10.0.19045
Platform                      Win32NT
PSCompatibleVersions           {1.0, 2.0, 3.0, 4.0...}
SerializationVersion           1.1.0.1
BuildManStackVersion           3.0

PS C:\Users\Artur Lambo> Get-Host | Select-Object version

Version
-----
7.4.2

PS C:\Users\Artur Lambo> _
```

- Vérifier la version de .NET installée sur l'OS voir dans la registry ou en CLI dans PowerShell

c) Vérifier que WinRm (Gestion à distance WSM) est fonctionnel sur la machine

- Si ce n'est pas le cas il existe sur la documentation officielle de Ansible un module qui permet de configurer sa prise en charge sur la machine Windows.

Rédigé par : [Artur Lambo DevOps engineer](#)

L'objectif de cette vérification est de savoir si WinRm permet la communication à distance

```
PS C:\Windows\System32> Enable-PSRemoting -SkipNetworkProfileCheck -Force
WARNING: PowerShell remoting has been enabled only for PowerShell 6+ configurations and does not af
PowerShell remoting configurations.
WinRM est déjà configuré pour recevoir des demandes sur cet ordinateur.
WinRM est déjà configuré pour la gestion à distance sur cet ordinateur.
```

PS C:\Windows\System32> Enable-PSRemoting -SkipNetworkProfileCheck -Force

Cet attribut demande à WinRm d'ignorer tous les types profils réseaux présents sur la machine.

Si les profils réseaux sont définis à public ou domaine cette attribut va les ignorer et valider la communication à distance.

PS C:\Windows\System32> Enable-PSRemoting


En saisissant cette commande sans rajouter l'attribut **SkipNetworkProfile** WinRm renverra une erreur si le profil réseau est défini à public ou domaine il faut qu'il soit défini à private

```
Message
ProviderFault
WSManFault
Message - L'exception de pare-feu WinRM ne fonctionnera pas car l'un des types de connexion réseau de cet ordinateur est défini à Public.
e ou Privé, puis recommencez.
```

On pourra soit changer le profil réseau dans les paramètres réseau soit tout simplement l'ignorer avec l'attribut **-SkipNetworkProfileCheck -Force**

Si cela ne fonctionne toujours pas on pourra soit :

- ❖ Vérifier la liste des règles de pare feu qui interagissent avec WinRm sur le trafic entrant ou sortant ensuite regarder dans les propriétés de ces règles si la connexion à distance n'est pas bloquée.
- ❖ Dans PowerShell : **gpresult /h gpo_report.html** : ce qui affiche une page html contenant toutes les stratégies de groupes appliquées à l'utilisateur et l'ordinateur.

 Si la communication est fonctionnelle on pourra vérifier l'état du service WinRm.

```
PS C:\Users\Artur Lambo> Get-Service *winrm*

Status  Name      DisplayName
-----
Running WinRM     Gestion à distance de Windows (Gestio...
```

```
PS C:\Windows\System32> Test-WSMan -ComputerName 192.168.1.13

wsid      : http://schemas.dmtf.org/wbem/wsman/identity/1/wsmanidentity.xsd
protocolVersion : http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd
productVendor : Microsoft Corporation
productVersion : OS: 0.0.0 SP: 0.0 Stack: 3.0

PS C:\Windows\System32> Test-netConnection 192.168.1.13 -Port 5985

computerName      : 192.168.1.13
remoteAddress      : 192.168.1.13
remotePort         : 5985
interfaceAlias     : Wi-Fi
sourceAddress      : 192.168.1.13
tcpTestSucceeded   : True

PS C:\Windows\System32> _
```

Voir la configuration de WinRm

```
PS C:\Users\Artur Lambo> winrm get winrm/config/Service
>>
Service
    RootSDDL = O:NSG:BAD:P(A;;GA;;;BA)(A;;GR;;;IU)S:P(AU;FA;GA;;;WD)(AU;SA;GXGW;;;WD)
    MaxConcurrentOperations = 4294967295
    MaxConcurrentOperationsPerUser = 1500
    EnumerationTimeoutms = 240000
    MaxConnections = 300
    MaxPacketRetrievalTimeSeconds = 120
    AllowUnencrypted = false
    Auth
        Basic = false
        Kerberos = true
        Negotiate = true
        Certificate = false
        CredSSP = false
        CbtHardeningLevel = Relaxed
    DefaultPorts
        HTTP = 5985
        HTTPS = 5986
    IPv4Filter = *
    IPv6Filter = *
    EnableCompatibilityHttpListener = false
    EnableCompatibilityHttpsListener = false
    CertificateThumbprint
    AllowRemoteAccess = true

PS C:\Users\Artur Lambo> winrm enumerate winrm/config/Listener
>>
Listener
    Address = *
    Transport = HTTP
    Port = 5985
    Hostname
    Enabled = true
    URLPrefix = wsman
    CertificateThumbprint
    ListeningOn = 127.0.0.1, 169.254.110.135, 169.254.132.233, 169.254.171.248, 169.254.219.87, 172.25.32.1, 192.168.1.12, 192.168.153.1, 192.168.217.1, ::1, fe80::579b:3b14:51a1:c19f%13, fe80::6677:bf84:93c5:2f36%17, fe80::7c0c:c1e8:545f:65f8%19, fe80::7d0b:cdb8:85dd:ef22%16, fe80::8b48:9b55:ba7c:7e46%30, fe80::e2b6:fe56:912d:a3c1%22, fe80::f15d:ef4c:359d:68c5%9
```

On reviendra voir cette configuration à la fin

| Directive | Valeur | Appréciation |
|--------------------|--------|---|
| Auth.Basic | False | On peut voir que l'authentification basique est à false ce qui prouve que WinRm n'a pas encore été configuré pour une authentification via un utilisateur local. |
| Auth.Kerberos | True | C'est un service qui permet de joindre une machine à un active directory. Donc dans le script ansible on peut joindre une machine dans un royaume AD via Kerberos. |
| DefaultPorts.HTTP | 5985 | Par défaut WinRm écoute sur le port 5985 http . Etant donné que le certificat n'a pas encore été installé. |
| DefaultPorts.HTTPS | 5986 | Dépend du certificat |

Rédigé par : [Artur Lambo DevOps engineer](#)

| | | |
|-------------|------|----|
| Certificate | True | ?? |
|-------------|------|----|

On a deux PC A → 192.168.1.101 et B → 192.168.1.12. On veut accéder à nos Vms installées sur le PC B depuis le PC A.

Consignes :

- ❖ les deux PC peuvent faire un ping le protocole ICMP IPV4 est actif sur les deux machines.
- ❖ On veut pouvoir industrialiser les configurations via **ansible** sur les Vms du PC B.

PC A :

- Générer une clé **ssh** pour un <utilisateur> et copier sa clé publique sur le PC B et dans le fichier **authorize_keys** de ses Vms.
-
- Créer un fichier hosts dans ansible où l'on va définir un tunnel ssh afin d'accéder aux Vms du PC B

Exemple du fichier hosts ansible

```
all:
  children:
    windows10:
      hosts:
        LAMBOFT:
          ansible_host: 192.168.1.12
          ansible_user: ansibleWinRm
          ansible_password: !vault | Unresolved tag: !vault
          $ANSIBLE_VAULT;1.1;AES256
          63623438626365613531316131346530343266663138323262303961633133316433643930353333
          3635323463343437656530363266653136306536333939650a666137333939336330636430626333
          37303861303335643936316538656365623531643437656638363535656339396238386638396464
          3964396364623161390a626333313537353139383462653439613236616234326234646161343439
          3863
          ansible_port: 5985
          ansible_connection: winrm
          ansible_winrm_scheme: http
          ansible_winrm_transport: ntlm
          ansible_winrm_server_cert_validation: ignore
          ansible_python_interpreter: 'C:\program files\Python311\python.exe'
```

LAMBOFT : c'est le nom de notre PC B

192.168.1.12 : C'est IP du PC B.

ansibleWinRm : C'est un utilisateur local ayant les droits admins sur le PC B. son mot de passe est crypté via ansible-vault **ansible-vault encrypt_string --name 'ansible_password'**

5985 : c'est le port de WinRM en http.

```
PS C:\Users\ansibleWinRm> winrm get winrm/config/service
Service
  RootSDDL = O:NSG:BAD:P(A;;GA;;;BA)(A;;GR;;;IU)S:P(AU;FA;GA;;;WD)(AU;SA;GXGW;;;WD)
  MaxConcurrentOperations = 4294967295
  MaxConcurrentOperationsPerUser = 1500
  EnumerationTimeoutms = 240000
  MaxConnections = 300
  MaxPacketRetrievalTimeSeconds = 120
  AllowUnencrypted = false
  Auth
    Basic = true
    Kerberos = true
    Negotiate = true
    Certificate = false
    CredSSP = false
    CbtHardeningLevel = Relaxed
  DefaultPorts
    HTTP = 5985
    HTTPS = 5986
  IPv4Filter = *
  IPv6Filter = *
  EnableCompatibilityHttpListener = false
  EnableCompatibilityHttpsListener = false
  CertificateThumbprint
  AllowRemoteAccess = true
```

Winrm : C'est un protocole de communication qui permet à des hosts distants de communiquer avec des serveurs Windows.

Windows to Windows , linux to windows

En revanche Windows to linux on peut utiliser **OpenSSH**.

Ntlm : C'est une méthode d'authentification de **winrm** pour permettre à un compte local ou d'AD de se connecter sur un host Windows.

Autres méthodes d'authentications : **Kerberos**, basic authentication, via un **Certificat**.

```
vms-linux:
hosts:
  # vms-001-ubuntu:
  #   ansible_host: 192.168.153.131
  vms-002-Server:
    ansible_host: 192.168.153.132
vars:
  ansible_user: lambo
  ansible_port: 22
  ansible_connection: ssh
  ansible_ssh_common_args: '-o ProxyCommand="ssh -W %h:%p ansibleWinRm@192.168.1.12"'
  ansible_python_interpreter: '/usr/bin/python3'
```

Dans cette configuration, on a défini la liste des hosts (Vms du PC B) sur lesquels on veut déployer la configuration.

ansible_ssh_common_args : C'est argument central qui nous permettra de créer un jump host ou bastion host ssh entre le terminal ansible et les Vms du PC B

ProxyCommand :

PortForwarding : on crée un tunnel SSH pour rediriger le port 22 vers le port 3306 de la base de données.

Rédigé par : [Artur Lambo DevOps engineer](#)

ssh -L 3306:serveur_interne:3306 utilisateur@**jump_host**

on veut accéder à une base de données sur le serveur 192.168.153.132 du PC B

```
ssh -L 3306 :192.168.153.132 :3306 ansibleWinRm@192.168.1.12
```

PC B :

- Dans les Vms, on va rajouter la clé publique ssh de l'utilisateur ansible du PC A dans les authorize_keys de chaque machine.
- Configurer **WinRM** sur le PC B pour accepter les connexions sur le PC depuis un hôte distant.
- Installer l'interpréteur **Python** dans les variables d'environnements du PC B

Logique d'accès aux Vms du PC B

- Ssh [ansibleWinRM@192.168.1.12](#)
- Ssh [lambo@192.168.153.132](#) depuis le Shell de la machine B
- ssh -o ProxyCommand="ssh -W %h:%p ansibleWinRm@192.168.1.12"
[lambo@192.168.153.132](#) : cette commande nous permet d'accéder directement à la Vm en utilisant l'hôte intermédiaire via ProxyCommand

NB : Lorsqu'un playbook Ansible s'exécute sur plusieurs hôtes simultanément, il peut y avoir des problèmes de performance ou de connectivité, surtout si vous utilisez un hôte intermédiaire (jump host) pour établir les connexions SSH. Voici quelques points à vérifier et des solutions potentielles pour résoudre ces problèmes

```
ansibleWinRm@192.168.1.12's password:
fatal: [vms-002-Server]: UNREACHABLE! => {"changed": false, "msg": "Failed to connect to the host via ssh: Connection timed out during banner exchange\r\nConnection to UNKNOWN port 6553
timed out", "unreachable": true}

PLAY RECAP *****
lambo01                : ok=0    changed=0    unreachable=0    failed=0    skipped=2    rescued=0    ignored=0
vms-001-ubuntu         : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
vms-002-Server         : ok=0    changed=0    unreachable=1    failed=0    skipped=0    rescued=0    ignored=0
```

Solution :

- Rajouter dans le playbook **serial : 1** → exécute les tâches sur les hôtes de manière séquentielle et non pas simultanées car de base ansible s'exécute de façon simultanée sur tous les hôtes.
- Rajouter l'entrée suivante dans le fichier de conf d'ansible

```
[ssh_connection]
ssh_args = -o ControlMaster=auto -o ControlPersist=60s -o ConnectTimeout=30s
```

II Organisation du fichier d'inventaire

On va scinder les directives liées aux configurations linux des configurations Windows. Sachant que les playbooks sont joués à partir d'un jump host Windows.

Jump Host Windows 10

```
all:
  children:
    # Groupe pour la machine physique Windows 10
    jump_hosts:
      hosts:
        LAMBOFT:
          ansible_host: 192.168.1.12
      vars:
        ansible_user: ansibleWinRm
        ansible_password: !vault |
        ansible_port: 5985
        ansible_connection: winrm
        ansible_winrm_scheme: http
        ansible_winrm_transport: ntlm
        ansible_winrm_server_cert_validation: ignore
        ansible_python_interpreter: 'C:\Program Files\Python311\python.exe'
```

Linux

```
linux_servers:
  hosts:
    vms-001-ubuntu:
      ansible_host: 192.168.153.131
    vms-002-Server:
      ansible_host: 192.168.153.132
    vms-003-centos7:
      ansible_host: 192.168.153.133

  vars:
    ansible_user: lambo
    ansible_port: 22 # Port SSH standard pour Linux
    ansible_connection: ssh
    ansible_ssh_common_args: '-o ProxyCommand="ssh -W %h:%p
ansibleWinRm@192.168.1.12" -o ControlMaster=auto -o ControlPersist=60s'
    ansible_python_interpreter: '/usr/bin/python3'
    become: yes
    become_method: sudo
```

Vérifier bien que l'interpréteur python se trouve au bon endroit dans linux.

Rédigé par : [Artur Lambo DevOps engineer](#)

III Création des rôles pour Windows et Linux

a. ROLES LINUX

b. ROLES WINDOWS

On va créer un rôle ansible pour les windows via la commande

```
ansible-galaxy init <role>
```

03 modules sont à notre disposition pour créer des rôles Ansible dans Windows

- **win_chocolatey** : Qui se base sur le dépôt chocolatey afin de télécharger les sources des modules que l'on souhaite utiliser sur windows.
- **win_package** : Installe les paquets de type .msi ou .exe
- **win_command** | **win_shell** : Pour une installation manuelle ici on a déjà le paquet et on souhaite lancer une commande.

Exemple : si on a déjà installé chocolatey on peut utiliser Win_command pour exécuter la commande suivante

```
choco install nodejs-lts --version=18.20.4
```

Pour l'installation de nodejs-lts

On va utiliser les 03 modules si possibles

- Installation de chocolatey sur Windows

```
- name: Installation de l'outil chocolatey sur windows
  win_shell: |
    if (-not (Get-Command choco -ErrorAction SilentlyContinue)) {
      Set-ExecutionPolicy Bypass -Scope Process -Force;
      [System.Net.ServicePointManager]::SecurityProtocol =
[System.Net.ServicePointManager]::SecurityProtocol -bor 3072;
      iex ((New-Object
System.Net.WebClient).DownloadString('https://chocolatey.org/install.ps1'));
    }
  args:
    creates: C:\ProgramData\chocolatey\bin\choco.exe
```

Ce Shell va vérifier si chocolatey n'existe pas sur la Vm et si ce n'est pas le cas il va le créer

Creates : Cette directive est utilisée pour ne pas réexécuter l'installation de chocolatey si elle a déjà été faite et aussi vérifie la présence de **choco.exe**.

1. Configuration de IIS sur le slave Windows et installation des outils de complémentation du projet web

PB

```
fatal: [LAMBOFT]: UNREACHABLE! => {
  "changed": false,
  "msg": "ntlm: HTTPConnectionPool(host='192.168.1.12', port=5985): Max
retries exceeded with url: /wsman (Caused by
ConnectTimeoutError(<urllib3.connection.HTTPConnection object at
0x7f8825e4f500>, 'Connection to 192.168.1.12 timed out. (connect
timeout=30)'))",
  "unreachable": true
}
```