Lab: Adding Windows Host

Introduction:

Ansible is a powerful open-source automation tool that allows you to manage and configure systems, including Windows servers.

Adding a Windows host to Ansible involves configuring WinRM (Windows Remote Management) on the target machine and creating an inventory file to store connection details.

Objectives

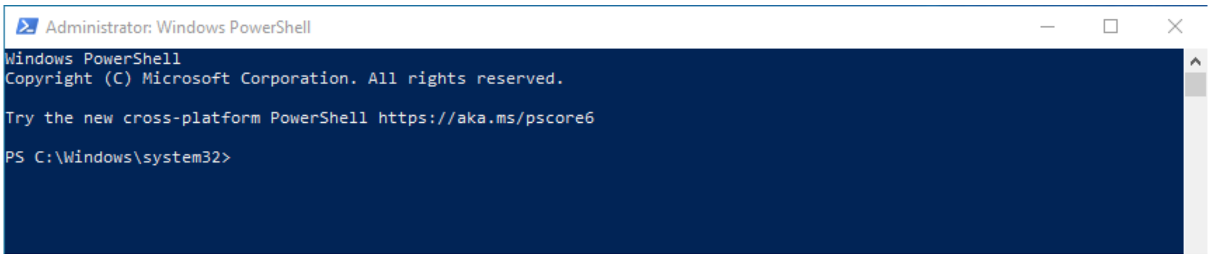
- **Configuring windows host**
- **Creating a new user for windows hosts management.**

1. Configuring windows host.

1.1 Let's install python3.11 package and install pywinrm.

1.2 In the windows machine open the powershell in administrator mode.

Output:

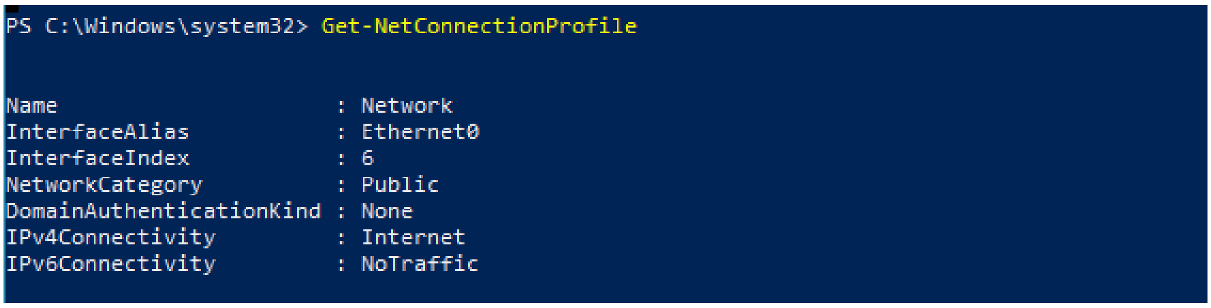


```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Windows\system32>
```

1.3 Check for your network profile.

```
# Get-NetConnectionProfile
```

Output:



```
PS C:\Windows\system32> Get-NetConnectionProfile


Name                     : Network
InterfaceAlias           : Ethernet0
InterfaceIndex           : 6
NetworkCategory          : Public
DomainAuthenticationKind : None
IPv4Connectivity         : Internet
IPv6Connectivity         : NoTraffic
```

Note: By default network profile will be in public and WinRm needs a private or Domain based Network profile.

1.4 Set then Network Profile to private by the following command.

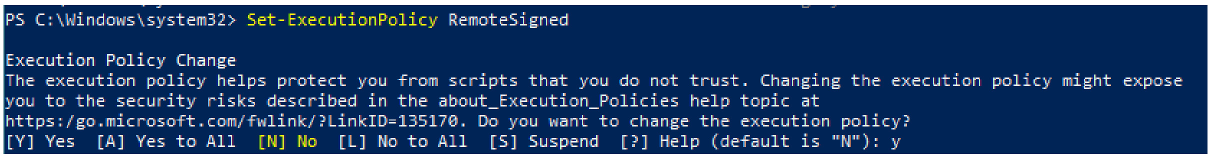
```
# Set-NetConnectionProfile -InterfaceIndex 6 -
NetworkCategory Private
```

Note: Change the no **6** to your index number.

1.5 Set the execution policy to RemoteSigned to ensure PowerShell execution policy allows running scripts.

```
# Set-ExecutionPolicy RemoteSigned
```

Output:



```
PS C:\Windows\system32> Set-ExecutionPolicy RemoteSigned

Execution Policy Change
The execution policy helps protect you from scripts that you do not trust. Changing the execution policy might expose
you to the security risks described in the about_Execution_Policies help topic at
https:/go.microsoft.com/fwlink/?LinkID=135170. Do you want to change the execution policy?
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is "N"): y
```

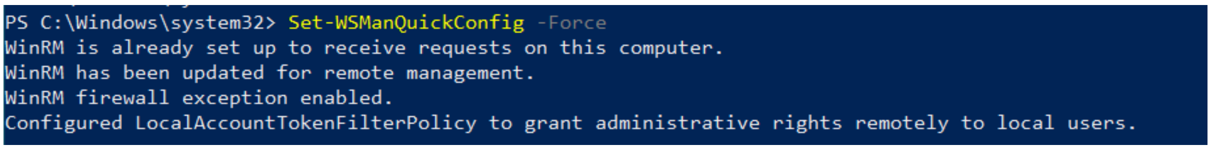
1.6 Ensure that the WinRM service is running.

```
# Start-Service WinRM
```

1.7 Enable basic remoting capabilities by the following commands.

```
# Set-WSManQuickConfig -Force
```

Output:

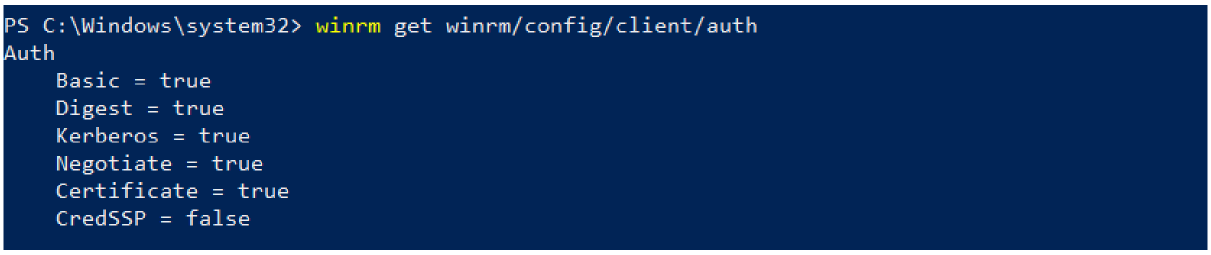


```
PS C:\Windows\system32> Set-WSManQuickConfig -Force
WinRM is already set up to receive requests on this computer.
WinRM has been updated for remote management.
WinRM firewall exception enabled.
Configured LocalAccountTokenFilterPolicy to grant administrative rights remotely to local users.
```

1.8 Check for the basic auth is true or not.

```
# winrm get winrm/config/client/auth
```

Output:

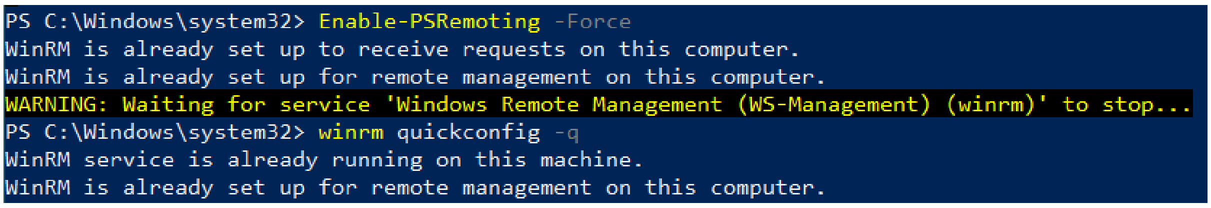


```
PS C:\Windows\system32> winrm get winrm/config/client/auth
Auth
    Basic = true
    Digest = true
    Kerberos = true
    Negotiate = true
    Certificate = true
    CredSSP = false
```

1.9 Make sure WinRM is properly configured on your Windows host. Ansible communicates with Windows hosts over WinRM, so you need to enable it and configure it to allow connections.

```
# Enable-PSRemoting -Force
# winrm quickconfig -q
```

Output:

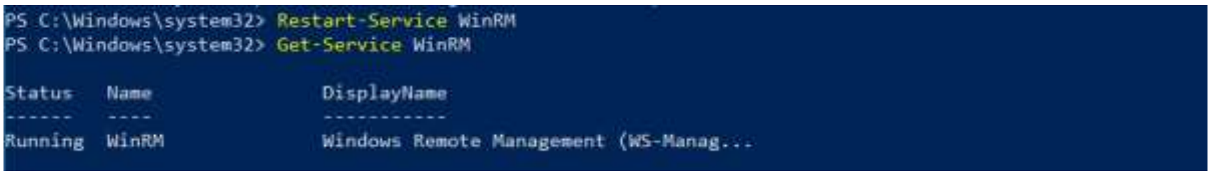


```
PS C:\Windows\system32> Enable-PSRemoting -Force
WinRM is already set up to receive requests on this computer.
WinRM is already set up for remote management on this computer.
WARNING: Waiting for service 'Windows Remote Management (WS-Management) (winrm)' to stop...
PS C:\Windows\system32> winrm quickconfig -q
WinRM service is already running on this machine.
WinRM is already set up for remote management on this computer.
```

1.10 Restart the service once and check for the state of the service.

```
# Restart-Service WinRM
# Get-Service WinRM
```

Output:



```
PS C:\Windows\system32> Restart-Service WinRM
PS C:\Windows\system32> Get-Service WinRM

Status   Name               DisplayName
------   ----               -----------
Running  WinRM              Windows Remote Management (WS-Manag...
```

2. Creating a new user for windows hosts management.

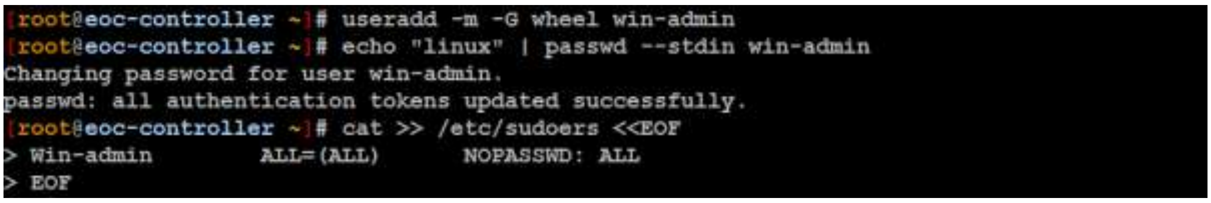
2.1 Perform this as root user.

```
# useradd -m -G wheel win-admin
```

```
# echo "linux" | passwd --stdin win-admin
```

```
# cat >> /etc/sudoers <<EOF

Win-admin          ALL=(ALL)        NOPASSWD: ALL

EOF
```

Output:

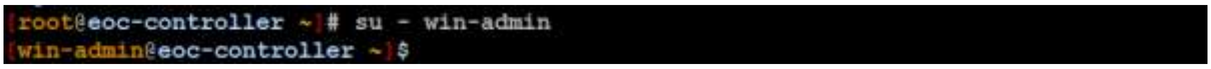


```
[root@eoc-controller ~]# useradd -m -G wheel win-admin
[root@eoc-controller ~]# echo "linux" | passwd --stdin win-admin
Changing password for user win-admin.
passwd: all authentication tokens updated successfully.
[root@eoc-controller ~]# cat >> /etc/sudoers <<EOF
> Win-admin          ALL=(ALL)       NOPASSWD: ALL
> EOF
```

2.2 Switch the user.

```
# su - win-admin
```

Output:

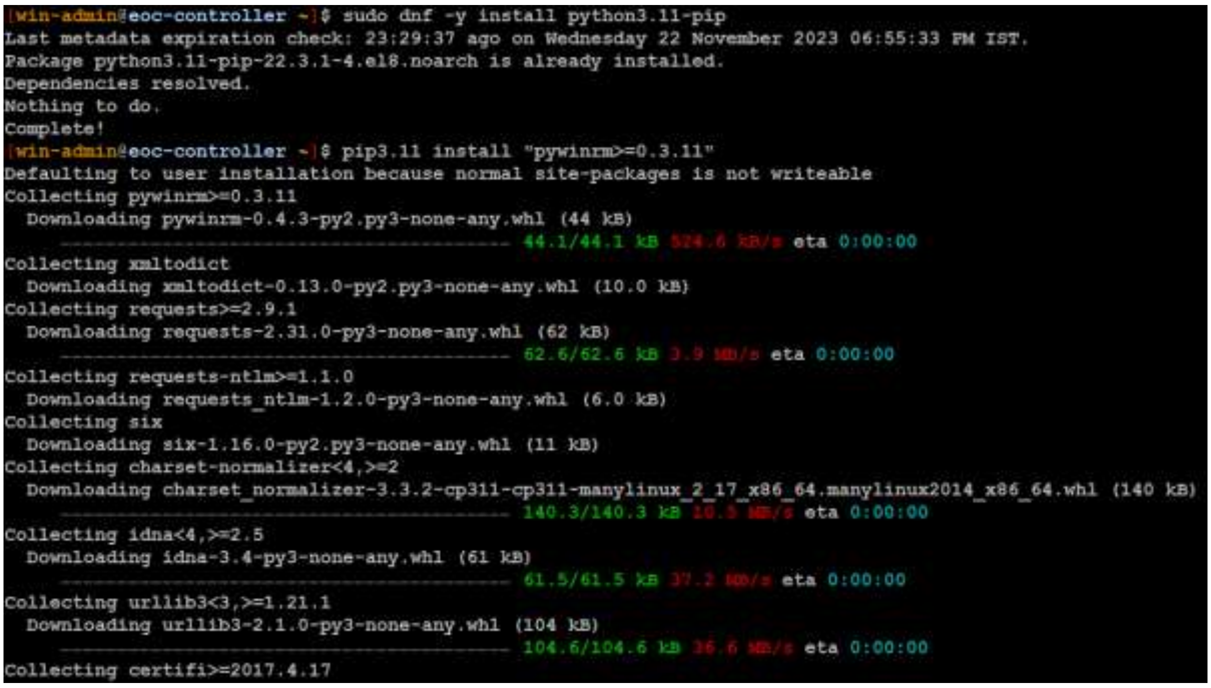


```
[root@eoc-controller ~]# su - win-admin
[win-admin@eoc-controller ~]$
```

2.3 Install the required packages.

```
# sudo dnf -y install python3.11-pip
# pip3.11 install "pywinrm>=0.3.11"
```

Output:



```
[win-admin@eoc-controller ~]$ sudo dnf -y install python3.11-pip
Last metadata expiration check: 23:29:37 ago on Wednesday 22 November 2023 06:55:33 PM IST.
Package python3.11-pip-22.3.1-4.el8.noarch is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[win-admin@eoc-controller ~]$ pip3.11 install "pywinrm>=0.3.11"
Defaulting to user installation because normal site-packages is not writeable
Collecting pywinrm>=0.3.11
  Downloading pywinrm-0.4.3-py2.py3-none-any.whl (44 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 44.1/44.1 kB 524.6 kB/s eta 0:00:00
Collecting xmltodict
  Downloading xmltodict-0.13.0-py2.py3-none-any.whl (10.0 kB)
Collecting requests>=2.9.1
  Downloading requests-2.31.0-py3-none-any.whl (62 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 62.6/62.6 kB 3.9 MB/s eta 0:00:00
Collecting requests-ntlm>=1.1.0
  Downloading requests_ntlm-1.2.0-py3-none-any.whl (6.0 kB)
Collecting six
  Downloading six-1.16.0-py2.py3-none-any.whl (11 kB)
Collecting charset-normalizer<4,>=2
  Downloading charset_normalizer-3.3.2-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (140 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 140.3/140.3 kB 10.5 MB/s eta 0:00:00
Collecting idna<4,>=2.5
  Downloading idna-3.4-py3-none-any.whl (61 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 61.5/61.5 kB 37.2 MB/s eta 0:00:00
Collecting urllib3<3,>=1.21.1
  Downloading urllib3-2.1.0-py3-none-any.whl (104 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 104.6/104.6 kB 36.6 MB/s eta 0:00:00
Collecting certifi>=2017.4.17
```

2.4 Create an Ansible **config file**.

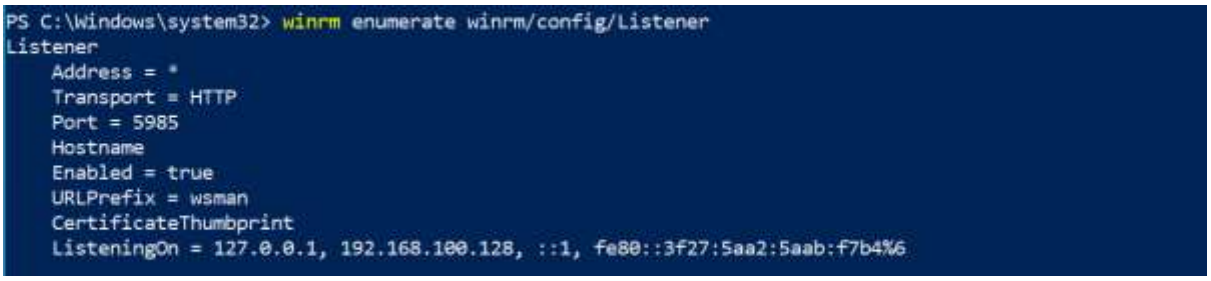
```
# cat > .ansible.cfg <<EOF
[defaults]
inventory = ~/org-infra
roles_path = ~/roles
remote_user = ansible
ansible_python_interpreter=/usr/bin/python3.11

[privilege_escalation]
become=True
become_method=winrm
become_user=Administrator
become_ask_pass=False

[winrm_connection]
ansible_winrm_transport = basic
ansible_winrm_server_cert_validation = ignore
EOF
```

2.5 Configure the host's **inventory**.

```
# cat > org-infra << EOF
[windows_hosts]
windows_machine ansible_host=192.168.100.128
ansible_user=ansible ansible_password=linux
ansible_port=5985 ansible_connection=winrm
ansible_winrm_server_cert_validation=ignore
ansible_winrm_transport=ntlm
EOF
```

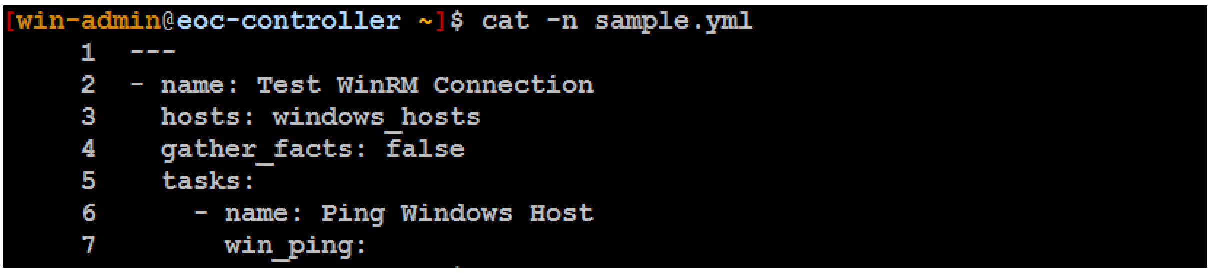
Note: Change the bold ones with your configuration, get the port no by running the command "**winrm enumerate winrm/config/Listener**" in the powershell

Output:



```
PS C:\Windows\system32> winrm enumerate winrm/config/Listener
Listener
    Address = *
    Transport = HTTP
    Port = 5985
    Hostname
    Enabled = true
    URLPrefix = wsman
    CertificateThumbprint
    ListeningOn = 127.0.0.1, 192.168.100.128, ::1, fe80::3f27:5aa2:5aab:f7b4%6
```

2.6 Create a sample playbook to ping the hosts.

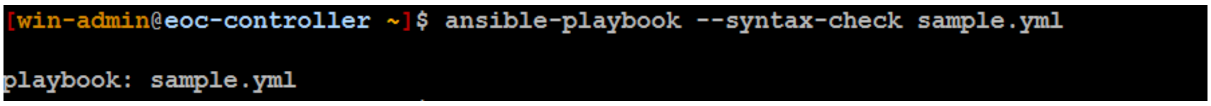


```
[win-admin@eoc-controller ~]$ cat -n sample.yml
     1  ---
     2  - name: Test WinRM Connection
     3    hosts: windows_hosts
     4    gather_facts: false
     5    tasks:
     6      - name: Ping Windows Host
     7        win_ping:
```

2.7 Let's verify the syntax of the playbook **sample.yml.**

```
# ansible-playbook --syntax-check sample.yml
```

Output:

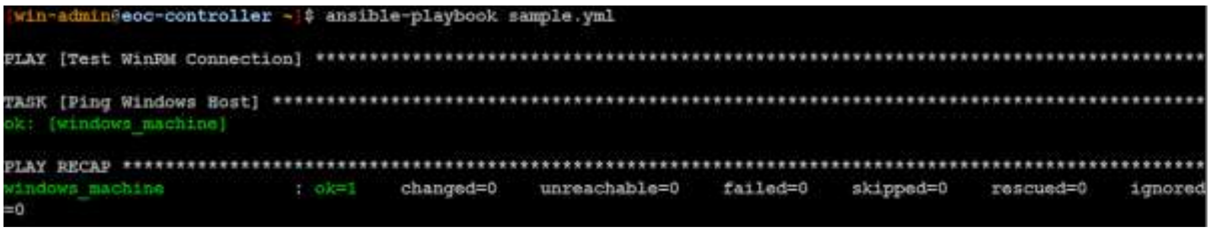


```
[win-admin@eoc-controller ~]$ ansible-playbook --syntax-check sample.yml

playbook: sample.yml
```

2.8 Run the playbook **sample.yml** and check the hosts.

```
# ansible-playbook sample.yml
```

Output:

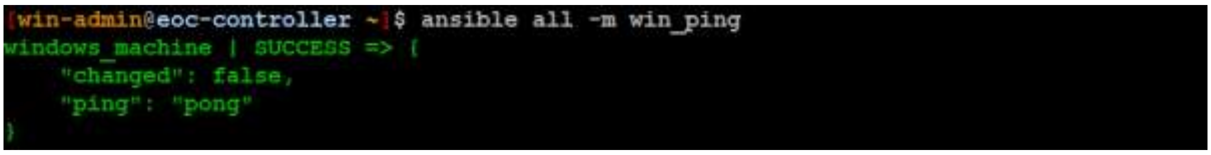


```
[win-admin@eoc-controller ~]$ ansible-playbook sample.yml

PLAY [Test WinRM Connection] *******************************************************************

TASK [Ping Windows Host] ***********************************************************************
ok: [windows_machine]

PLAY RECAP *************************************************************************************
windows_machine            : ok=1    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored
=0
```

2.9 Ping using the module directly.

```
# ansible all -m win_ping
```

Output:



```
[win-admin@eoc-controller ~]$ ansible all -m win_ping
windows_machine | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
```