

Configure - Ansible

October 2022

What is Ansible?



“[Ansible](#) is a software tool that provides simple but powerful automation for cross-platform computer support. It is primarily intended for IT professionals, who use it for application deployment, updates on workstations and servers, cloud provisioning, configuration management, intra-service orchestration, and nearly anything a systems administrator does on a weekly or daily basis. Ansible doesn't depend on agent software and has no additional security infrastructure, so it's easy to deploy.”

Ansible documentation is great and has examples that can be leveraged

We are going to deploy:

- Windows Domain Controller
- Windows Member Server
- Linux OpenVPN Server
- Linux Web Server

Overview

- Review the Files Provided on Github
- Walk through the ansible configurations
- Understand the configurations that need to be created
- Build the configurations based on your lab

Ansible - ansible.cfg

```
thepcn3rd@rutgz:~/bsidesIF/ansible$ cat ansible.cfg
[defaults]
nocows = 1
host_key_checking = False
deprecation_warnings = False
roles_path = ${PWD}/roles
library = ${PWD}/library
inventory = ${PWD}/inventory.yml

[inventory]
#enable_plugins = vmware_vm_inventory
#enable_plugins = vmware_vm_inventory, community.vmware.vmware_vm_info

[ssh_connection]
pipelining = True

[persistent_connection ]
command_timeout = 90
thepcn3rd@rutgz:~/bsidesIF/ansible$
```

nocows = 1 - Disables the ASCII Art Default Setting

host_key_checking = False (Poor Security)

deprecation_warnings = False (Poor Operational)

Inventory File

command_timeout - Necessary for slow connections
or reboots in configurations

Ansible - inventory.yml (Inventory File)

```
all:
  hosts:
    purpleVPN:
      # Ubuntu 20.04 - 1 GB - 1vCPU - 40GB SSD
      # For the VPN verify 1194/UDP is open in the networking setup...
      ansible_host: "35.91.253.148"
      ansible_port: "22"
      ansible_connection: "ssh"
      ansible_user: "{{ lightsailUser }}"
      ansible_ssh_private_key_file: "{{ lightsailPem }}"
    purpleMbr:
      # How to setup winrm: https://docs.ansible.com/ansible/latest/user_guide/windows_setup.html
      # Verify the 5986 Port is open in the networking setup...
      ansible_host: "35.90.89.40"
      ansible_port: "5986"
      ansible_user: "Administrator"
      ansible_password: "...thePassword..."
      ansible_connection: "winrm"
      ansible_winrm_server_cert_validation: "ignore"
      ansible_winrm_scheme: "https"
```

Indentation is
Important

Hosts (Associates to
the Playbooks)

Ansible connections
are established to
the hosts

Ansible variables are
great to use
everywhere

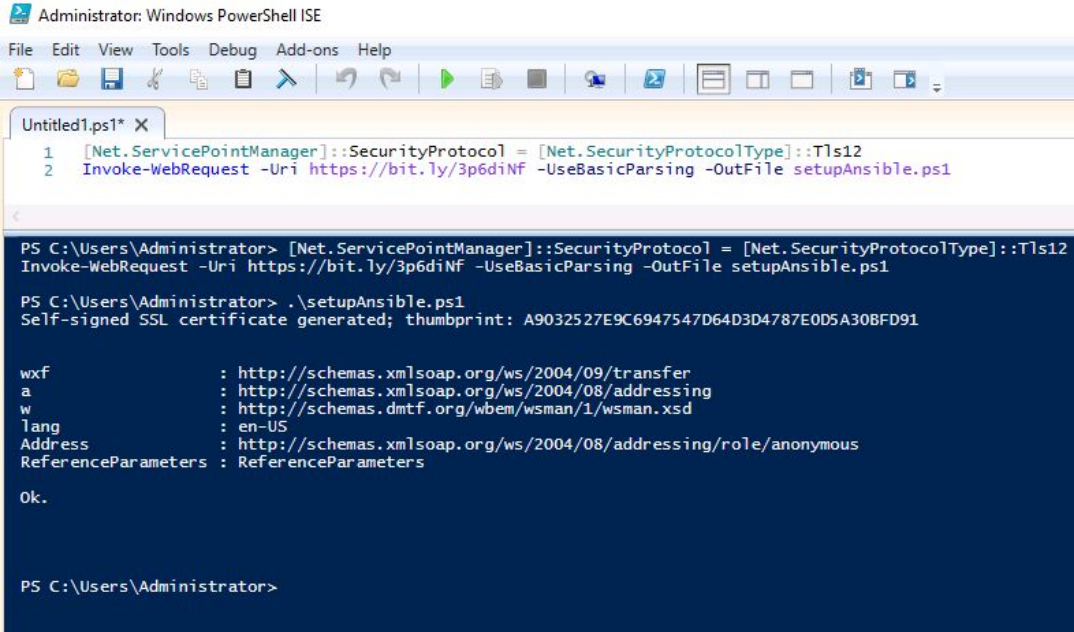
inventory.yml - Setup WinRM

```
purpleDC:
  # Install and Configure WinRM with https://raw.githubusercontent.com/thepcn3rd/IT420/main/Con
  #
  # Powershell to execute on windows host at LightSail
  # [Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12
  # Invoke-WebRequest -Uri https://bit.ly/3p6diNf -UseBasicParsing -OutFile setupAnsible.ps1
  #
  # Setup firewall in Lightsail to allow the connection over TCP 5986
  #
  ansible_host: "35.87.103.125"
  ansible_port: "5986"
  ansible_user: "Administrator"
  ansible_password: "{{ domainPass }}"
  ansible_connection: "winrm"
  ansible_winrm_server_cert_validation: "ignore"
  ansible_winrm_scheme: "https"
```

Setup Ansible on Windows by running the powershell script mentioned

The script is from ansible and they do note that it should be modified for an enterprise environment

Setup of WinRM on purpleDC and purpleMbr



```
Administrator: Windows PowerShell ISE
File Edit View Tools Debug Add-ons Help
Untitled1.ps1* X
1 [Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12
2 Invoke-WebRequest -Uri https://bit.ly/3p6diNF -UseBasicParsing -OutFile setupAnsible.ps1

PS C:\Users\Administrator> [Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12
Invoke-WebRequest -Uri https://bit.ly/3p6diNF -UseBasicParsing -OutFile setupAnsible.ps1

PS C:\Users\Administrator> .\setupAnsible.ps1
Self-signed SSL certificate generated; thumbprint: A9032527E9C6947547D64D3D4787E0D5A308FD91

wxsf      : http://schemas.xmlsoap.org/ws/2004/09/transfer
a         : http://schemas.xmlsoap.org/ws/2004/08/addressing
w         : http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd
lang      : en-US
Address   : http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
ReferenceParameters : ReferenceParameters

Ok.

PS C:\Users\Administrator>
```

To setup ansible download
and then execute the
Powershell

.\setupAnsible.ps1

Needs to be done prior to
ansible executing

Inventory.yml - vars

```
vars:
  ansible_python_interpreter: "/usr/bin/python3"
  ansibleDirectory: "/home/thepcn3rd/Ensign/ansible"
  domainName: "13lives.local"
  netbiosName: "13lives"
  # Needs to be updated to the internal IP of the Domain Controller
  dnsInternalServer: "172.26.5.109"
  # Needs to be updated to the administrator password of the domain controller
  domainPass: "...thePassword..."
  commonUser: "thepcn3rd"
  commonPass: "...aPassword..."
  lightsailUser: "ubuntu"
  lightsailPem: "/home/thepcn3rd/Ensign/ansible/keys/BSidesIF.pem"
  className: "BSides IF 2022"
```

Ansible Python Interpreter
(When python2 was a thing...)

Variables used in
playbook...

Change Paths as necessary

lightsailPem is what you
download from your
instance

dnsInternalServer is the Private IP of purpleDC, the domainPass is the administrator password of purpleDC, and the domainName and netbiosName used to create and join the domain present

Configure.yml - playbooks.yml

```
thepcn3rd@rutgz:~/bsidesIF/ansible$ cat configure.yml
# Prerequisites
# 1. Install sshpass - apt install sshpass
# 2. Setup correct permissions on pem key - chmod 400 ls.pem

## Command that you execute to run the playbooks uncommented
# ansible-playbook -i inventory.yml configure.yml
#
## Uncomment the playbook below that you want to execute
- import_playbook: playbooks/configPurpleVPN.yml
- import_playbook: playbooks/configPurpleDC.yml
- import_playbook: playbooks/configPurpleMbr.yml
- import_playbook: playbooks/configPurpleLin.yml
thepcn3rd@rutgz:~/bsidesIF/ansible$
```

```
thepcn3rd@rutgz:~/bsidesIF/ansible/playbooks$ ls -lha
total 44K
drwxrwxr-x 2 thepcn3rd thepcn3rd 4.0K Sep 17 10:24 .
drwxrwxr-x 8 thepcn3rd thepcn3rd 4.0K Sep 17 11:00 ..
-rw-r--r-- 1 thepcn3rd thepcn3rd 11K Sep 17 10:24 configPurpleDC.yml
-rw-rw-r-- 1 thepcn3rd thepcn3rd 4.5K Sep 17 10:24 configPurpleLin.yml
-rw-rw-r-- 1 thepcn3rd thepcn3rd 9.1K Sep 17 10:24 configPurpleMbr.yml
-rw-r--r-- 1 thepcn3rd thepcn3rd 1.1K Sep 17 10:24 configPurpleVPN.yml
thepcn3rd@rutgz:~/bsidesIF/ansible/playbooks$
```

Define in configure.yml the order to run the playbooks

Notes about pre-reqs and how to execute with the inventory and configure files

Configure the DC before configuring clients that join to the DC

Defined a playbook for each server

Could define roles if I was configuring multiple servers as an apache webserver (Not demonstrated)

Playbook - configPurpleDC.yml

```
thepcn3rd@rutgz:~/bsidesIF/ansible/playbooks$ cat configPurpleDC.yml
```

```
---
- name: "Configure Windows Server DC"
  # Lightsail Windows Server 2019 1GB 1vCPU $12/month
  hosts: windc
  gather_facts: yes

  tasks:
    - set_fact:
        hostname: "windc"

    - name: "Change the hostname to {{ hostname }}"
      win_hostname:
        name: "{{ hostname }}"
        register: result

    - name: "Reboot"
      win_reboot:
        when: result.reboot_required

    - name: Install the domain controller
      win_feature:
        name: AD-Domain-Services
        include_management_tools: yes
        include_sub_features: yes
        state: present
        register: result

    - name: "Install domain {{ domainName }}"
      win_domain:
        dns_domain_name: "{{ domainName }}"
        domain_netbios_name: "{{ netbiosName }}"
        safe_mode_password: "{{ commonPass }}"
        register: ad
```

“hosts” - needs to match what is defined in inventory.yml to understand the IP, how to connect, how ansible connects, and passwords. This needs to be changed to purpleDC to match what is in the inventory.yml

Note the variables are pulled from the inventory.yml. The commonPass needs to be setup for the safe mode password of the domain.

Note the reboot, the ansible execution will wait until the reboot is completed. On a slow connection this can timeout.

configPurpleDC.yml - AD Groups and Users

```
- name: Create AD Group - Coaches
  win_domain_group:
    name: "Coaches"
    scope: domainlocal

- name: Create AD Group - Athletes
  win_domain_group:
    name: "Athletes"
    scope: domainlocal
```

```
- name: Create AD User - Chanin Wibunrungrueang
  win_domain_user:
    name: chanin.w
    password: "2FVM4gIieRoa0VWJ8m"
    state: present
    groups_action: add
    groups: "Athletes"

- name: Create AD User - Phanumat Saengdi
  win_domain_user:
    name: phanumat.s
    password: "Az6IdtTg2XkKm0WoKF"
    state: present
    groups_action: add
    groups: "Athletes"
```

Configure AD Groups and Users

Setup the Group Scope

Assign the users to groups

Set the Password

*The password can be secured by methods provided by ansible that I have not done for the purposes of this lab.

configPurpleMbr.yml - Prereq

```
---
- name: Configure Purple Attack Path - Windows Client
  hosts: purpleWin
  gather_facts: yes

  # Install and Configure WinRM with https://raw.githubusercontent.com
  #
  # Powershell to execute on windows host at LightSail
  # [Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocol
  # Invoke-WebRequest -Uri https://bit.ly/3p6diNf -UseBasicParsing -Ou
  #
  # Setup firewall in Lightsail to allow the connection over TCP 5986
  #
```

“hosts” - Remember to set the hosts to match what is in the yml file

Note the pre-setup of ansible is necessary

Open in Lightsail the public port of TCP 5986 and filter based on your IP

configPurpleMbr.yml - Install msi files

```
- name: Copy Firefox to c:\files
  ansible.windows.win_copy:
    src: "{{ ansibleDirectory }}/downloads/firefox103.msi"
    dest: 'c:\files\firefox103.msi'

- name: Install Firefox
  win_package:
    path: 'c:\files\firefox103.msi'
    state: present

- name: Copy Thunderbird to c:\files
  ansible.windows.win_copy:
    src: "{{ ansibleDirectory }}/downloads/thunderbird102.msi"
    dest: 'c:\files\thunderbird102.msi'

- name: Install Thunderbird
  win_package:
    path: 'c:\files\thunderbird102.msi'
    state: present
```

Copy program from ansible server to remote host

Install the MSI package, you can also remove

The path of c:\files is created above in the yml file

Everytime you execute this portion it will reinstall the msi file (You can change this behavior)

configPurpleMbr.yml - Configure to Join Domain

```
- name: Set DC as DNS
  # Change the DNS to the appropriate internal DC Address
  win_dns_client:
    adapter_names: '*'
    ipv4_addresses: "{{ dnsInternalServer }}"

- name: "Change the hostname to {{ hostname }}"
  win_hostname:
    name: "{{ hostname }}"
    register: result

# Reboot after the hostname change to be able to add to the domain
- name: Reboot
  win_reboot:
    when: result.reboot_required
```

Setup the DNS of the server to use the private IP Address of the purpleDC

Change the hostname from the default one setup. I have defined the hostname in the top of this yml file

Then reboot for the settings to take effect...

configPurpleMbr.yml - Join to Domain

```
- name: "Add computer to the domain {{ domainName }}"
  win_domain_membership:
    dns_domain_name: "{{ domainName }}"
    hostname: "{{ hostname }}"
    domain_admin_user: "administrator@{{ domainName }}"
    domain_admin_password: "{{ domainPass }}"
    state: domain
  register: domain_state
  retries: 10
  delay: 30
  until: domain_state is success

- name: Reboot system
  win_reboot:
    when: domain_state.reboot_required
```

Add the computer to the domain

Increase the delay if necessary

Reboot the System

With less than 50 lines of code we setup a domain controller, with users, groups, and joined a member server

configPurpleMbr.yml - Define RDP Users

```
- name: Add nopparat.k to the remote desktop users group and local administrators group
  ansible.windows.win_group_membership:
    name: Remote Desktop Users
    members:
      - 13lives\nopparat.k
      - 13lives\ekapol.c
      - 13lives\ekkaphon.k
    state: present

- name: Add nopparat.k to the remote desktop users group and local administrators group
  ansible.windows.win_group_membership:
    name: Administrators
    members:
      - 13lives\nopparat.k
      - 13lives\ekapol.c
      - 13lives\ekkaphon.k
    state: present
```

Based on domain users here are the accounts that are local administrators and remote desktop users

This configuration does not show that I added prayut.c as a local administrator

configPurpleLin.yml - Install Packages Ubuntu

```
- name: "Install Dependencies for GetSimple CMS"
  apt:
    pkg:
      - curl
      - unzip
      - zip
      - apt-transport-https
      - ca-certificates
      - software-properties-common
      - apache2
      - php
      - php-gd
      - php-xml
      - php-curl
      - php-zip
      - libpam-google-authenticator
```

This installs packages necessary for the webserver and php software

libpam-google-authenticator is necessary for the bypassing MFA technique

configPurpleLin.yml - Configure apache

```
- name: "Start the apache2 web server on GetSimple CMS (Remember AWS firewall blocks access)"
  systemd:
    state: started
    name: apache2

- name: "Enable the apache2 web server on GetSimple CMS to start on reboot"
  systemd:
    name: apache2
    enabled: yes
    masked: no

- name: "Enable the Apache2 module rewrite"
  # ansible-galaxy collection install community.general
  community.general.apache2_module:
    state: present
    name: rewrite

- name: "Restart apache2 after module change"
  ansible.builtin.service:
    name: apache2
    state: restarted
```

Starts the apache server and enables it, in the event the server is restarted

Enables the mod_rewrite module for Apache

After the module is installed Apache needs to be restarted

configPurpleLin.yml - Setup of GetSimple CMS

```
- name: "Copy GetSimple CMS files to server"
  ansible.builtin.copy:
    src: "{{ ansibleDirectory }}/downloads/getSimpleCMS-Custom.zip"
    dest: /var/www/html/cms.zip
    mode: '0644'

- name: "Copy index.html file to server"
  ansible.builtin.copy:
    src: "{{ ansibleDirectory }}/downloads/index.html"
    dest: /var/www/html/index.html
    mode: '0644'
    group: www-data
    owner: www-data

- name: "Extract the GetSimple CMS web server files"
  ansible.builtin.unarchive:
    src: /var/www/html/cms.zip
    dest: /var/www/html
    remote_src: yes
    group: www-data
    owner: www-data

- name: "Remove the cms.zip file"
  ansible.builtin.file:
    path: /var/www/html/cms.zip
    state: absent
```

Pre-configured the GetSimple CMS PHP Website and created cms.zip

Copied the cms.zip and an index.html file to redirect to the cms folder where the CMS is located

Extracts the archive of cms.zip

Removes the archive of the cms.zip from /var/www/html (Good habit to remove zip archives)

Configure.yml - Execution Order and Execution

```
##  
## Uncomment the playbook below that you want to execute  
- import_playbook: playbooks/configPurpleVPN.yml  
- import_playbook: playbooks/configPurpleDC.yml  
- import_playbook: playbooks/configPurpleMbr.yml  
- import_playbook: playbooks/configPurpleLin.yml
```

```
thepcn3rd@rutgz:~/Ensign/ansible$ cat executeConfigure.sh  
#!/bin/bash  
  
ansible-playbook -i inventory.yml configure.yml
```

In the configure.yml file is where the playbooks are referenced in the execution order that you need.

Then execute the command with the inventory and configure file. (If any errors re-execute)

Downloads and SupportingScripts Folder

Downloads folder contains msi packages, the software is now out-dated

- Firefox
- Thunderbird
- OpenJDK 17 from Microsoft
- Libre Office 7.35
- "registrymodifications.xcu" - Used by Libre Office for Settings of Macros and History of Files
- "pwshProcessODT.ps1" - Used by the scheduled task to execute the macros sent
- 7-Zip - Can be used in a dll persistence activity outside of Purple Attack Path
- openvpn-install.sh - Script to configure openvpn
- getSimpleCMS-Custom.zip - Customized working version of GetSimple CMS

SupportingScripts

- buildADUsers.py - Used to generate the ansible yml to build the users, not the most efficient way of conducting this activity but was most efficient for what was created.

Keys Folder

Folder needs the SSH Key that is necessary to authenticate to your LightSail created Ubuntu Servers

Remember: You can create a custom key for the various instances of servers that you have...

The key if named something unique needs to be updated in the inventory.yml file for access to the Ubuntu Servers that you create