Ansible Windows

Running Ansible on Windows requires WinRM to be configured, and it requires one of several different authentication options. In real environments this is typically done with HTTPS certs, but since there is no Matt-Cloud Infosec, I can just use a username and password. Furthermore, since I don't have a secure password vault, I have done some creative stuff with the registry to get this done. Group policy used to allow for creating a local account with a specific password, but that feature has been depreciated. I have used Group Policy to have a user account be created, and then a startup script will run **cosmosrm.ps1** to set the password based on a registry key.

cosmosrm.ps1

```
# script for setting ansible service account to registry key
$username = "cosmos-ansible"
$ansible_registry = "HKLM:\SOFTWARE\Cosmos\Ansible"
$password_key = "Password"
$password = (Get-ItemProperty $ansible_registry).$password_key

# This is what the thing needs to set the password
$securePassword = ConvertTo-SecureString $password -AsPlainText -Force

# Set password
$UserAccount = Get-LocalUser -Name $username
$UserAccount | Set-LocalUser -Password $securePassword

# Make it a local admin
Add-LocalGroupMember -Group "Administrators" -Member $username

# Various Ansible Settings
Set-Item -Path WSMan:\localhost\Service\Auth\Basic -Value $true
Enable-WSManCredSSP -Role Server -Force
```

The same password is in Jenkins, and the dynamic inventory file generator script adds the password to the inventory file. Like I said, not super secure, but it works.

inventory.sh

```
#!/bin/bash

# Dynamic inventory generation script ansible windows

# Function to display usage
usage() {
    echo "Windows Ansible Dynamic Inventory File Generation Script"
    echo "Usage: $0 -i IP_LIST -u JENKINS_USER -g JENKINS_GROUP -w WINDOWS_USER -p
ANSIBLE_PASSWORD [-a SERVER_SUBNET_GROUP] [-s] [-v] [-e]"
    echo "Options:"
    echo "  -i IP_LIST                Comma-separated list of IPs. Will not fail if blank, but why 0_o"
    echo "  -u JENKINS_USER           Jenkins user"
    echo "  -g JENKINS_GROUP          Jenkins primary group"
    echo "  -a SERVER_SUBNET_GROUP    Jenkins group for SSH access, need to pass something when
called"
    echo "  -w WINDOWS_USER           Windows user"
    echo "  -p ANSIBLE_PASSWORD       Password for the service account (Windows user)"
    echo "  -q                        Be quieter"
    echo "  -s                        Set variable to true if more than one IP is passed"
    echo "  -v                        Display Ansible Version"
    exit 1
}

# Initialize variables with default values
skip=false
more_than_one=false
display_version=false
allsubnet_group=missing
be_quiet=false

# Parse command line options
while getopts ":i:u:w:p:g:a:svq" opt; do
  case ${opt} in
    i ) # process option i
        IP_LIST=$OPTARG
        ;;
    u ) # process option u
        JENKINS_USER=$OPTARG
        ;;
```

```
    w ) # process option w
        WINDOWS_USER=$OPTARG
        ;;
    p ) # process option p
        ANSIBLE_PASSWORD=$OPTARG
        ;;
    g ) # process option g
        JENKINS_GROUP=$OPTARG
        ;;
    s ) # process option s
        skip=true
        ;;
    v ) # process option v
        display_version=true
        ;;
    q ) # process option q
        be_quiet=true
        ;;
    a ) # process option a
        allsubnet_group=$OPTARG
        ;;
    \? ) usage
        ;;
  esac
done
shift $((OPTIND -1))
# Check if all required options are provided
if [ -z "$JENKINS_USER" ] || [ -z "$JENKINS_GROUP" ] || [ -z "$WINDOWS_USER" ] || [ -z
"$ANSIBLE_PASSWORD" ]; then
    usage
fi

if $display_version; then
    if ! $be_quiet; then
        echo "Showing ansible version"
        ansible --version
    fi
fi
```

```
# Generate an 8-character hash from the IP list
hash=$(echo -n "$IP_LIST" | md5sum | cut -c 1-8)

if ! $be_quiet; then
    echo "IP List:"
    echo $IP_LIST
    echo $hash
fi



# Define the inventory file path with the hash
inventory_file="/var/jenkins_home/ansible-windows/.inv/inventory-$hash.yml"

if $skip; then
    IFS=',' read -ra IPS <<< "$IP_LIST"
    if [ ${#IPS[@]} -gt 1 ]; then
        more_than_one=true
    fi
fi

if $skip; then
    if ! $be_quiet; then
        echo "Single host option set"
    fi
    if $more_than_one; then
        if ! $be_quiet; then
            echo "IP list provided, inventory will be emptied"
        fi
        IP_LIST=""
    fi
fi

# Initialize the YAML inventory content
inventory_content="---
all:
  hosts:
"

# Loop through each IP in the comma-separated list
```

```
# skip if restricted user and subnet
IFS=',' read -ra IPS <<< "$IP_LIST"
for IP in "${IPS[@]}"; do
    ip_check=$(curl -s http://172.25.100.15:15010/ip_check?ip=${IP} | jq .in_subnets)
    # if this is a restricted subnet, then check the group
    if $ip_check; then
        if ! $be_quiet; then
            echo "Subnet restricted, checking group membership"
        fi
        if [ "$allsubnet_group" == "$SERVER_SUBNET_GROUP" ]; then
            if ! $be_quiet; then
                echo "IP Check Passed, adding endpoint ${IP} to inventory"
            fi
            inventory_content+="    ${IP}:
      ansible_host: ${IP}

"
        else
            if ! $be_quiet; then
                echo "Warning: User ${JENKINS_USER} not member of ${SERVER_SUBNET_GROUP}!"
                echo "Auth Check Failed for endpoint ${IP}, not adding to inventory"
            fi
        fi
    # if the subnet is not restricted, just add the endpoint to the inventory
    else
        if ! $be_quiet; then
            echo "Unrestricted subnet, adding endpoint ${IP} to inventory"
        fi
        inventory_content+="    ${IP}:
      ansible_host: ${IP}
"
    fi
done

inventory_content+="  vars:
    # windows user info
    ansible_user: ${WINDOWS_USER}
    ansible_password: '${ANSIBLE_PASSWORD}'
    ansible_become_user: ${WINDOWS_USER}
```

```
    ansible_become_pass: '${ANSIBLE_PASSWORD}'

    # ansible connection info
    ansible_connection: winrm
    ansible_winrm_transport: basic
    ansible_winrm_server_cert_validation: ignore
    ansible_winrm_scheme: http
    ansible_winrm_port: 5985

    # jenkins user info
    jenkins_user: '${JENKINS_USER}'
    jenkins_group: '${JENKINS_GROUP}'
    subnet_group_check: '${allsubnet_group}'
    SERVER_SUBNET_GROUP: '${SERVER_SUBNET_GROUP}'

    # other variables
    ansible_python_interpreter: /usr/bin/python3
"

# Write the inventory content to the file
echo "$inventory_content" > $inventory_file

# secure inventory file
if ! $be_quiet; then
    echo "Securing inventory file"
fi
chmod 700 $inventory_file

# echo inventory
if ! $be_quiet; then
    echo "Inventory file created at $inventory_file with the following content:"
    cat $inventory_file
fi
```

My primary motivation for this is so I can easily deploy the SSD Health Checker. This is done, so I am happy.

https://gitea.matt-cloud.com/matt/ansible-windows

https://jenkins.matt-cloud.com/job/Cosmos%20Windows/job/Disk%20API/

https://gitea.matt-cloud.com/matt/ansible-windows/src/branch/main/roles/storage_api

Now that it is complete, it means I have the platform and can do whatever with it now. I think this might be a good way to make a non-domain joined W11 computer de-fuckified if I ever need to be able to do that. For this, I would need a bigger powershell script and a different way to pull the password. This is just in general a good way to manage windows outside of AD, and building the infra was the biggest hurdle. Now that I have it, whipping out more playbooks is not a horribly complicated thing.

Revision #5
Created 26 October 2025 10:07:40 by Matt Anderson
Updated 5 November 2025 00:18:56 by Matt Anderson