GOAD Active Directory Lab Setup from a Windows host

l4dybug.medium.com/goad-active-directory-lab-setup-from-a-windows-host-dcdbfbb1ef08

Huriye Özdemir 31 января 2024 г.

Huriye Özdemir

In this blog post, I am going to explain how I set up the GOAD Active directory lab from my Windows host using VMware, along with a number of errors and how I fixed them.

GOAD (Game of Active Directory) lab is created by Orange Cyberdefense to provide pentesters a ready-to-use, vulnerable AD environment in which to practise common attack methods.

GitHub - Orange-Cyberdefense/GOAD: game of active directory

game of active directory. Contribute to Orange-Cyberdefense/GOAD development by creating an account on GitHub.

github.com

As described on the Github pages, *“the lab is intended to be installed from a ”*, but it is still possible to successfully install the lab from a Windows host. I did not want to install the lab inside a virtual Ubuntu machine, as nested virtualisation would slow down performance too much.

How my setup will look like:

Windows host: with Vagrant installed to run the VMs on VMware.

VMware Pro: with Ubuntu 22.04 VM installed to run ansible playbooks to make the AD vulnerable.

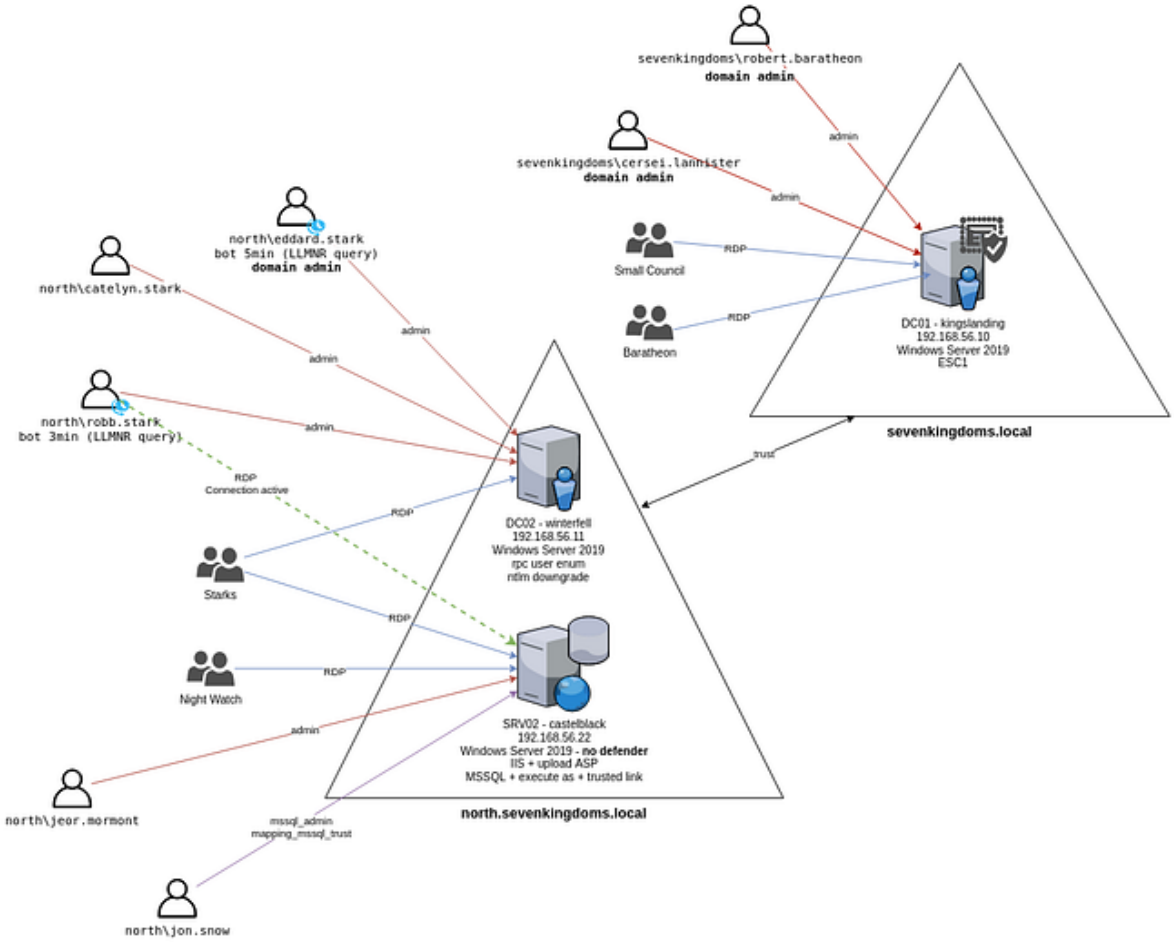
In this environment, there are two different available labs:

• GOAD : 5 vms, 2 forests, 3 domains (full goad lab)

• GOAD-Light : 3 vms, 1 forest, 2 domains (smaller goad lab for those with a smaller pc)

I am going to install **GOAD-Light** with the following VMs:

- DC01 — kingslanding
- DC02 — winterfell
- SRV02 — castelblack



Requirements

We will need Vagrant to build all virtual machines I've mentioned above. I first started to install on my Windows host.

If you do not have VMware, you can use Virtualbox and install **.**

When you successfully complete the Vagrant installation, you can check again whether the installation was successful via powershell or cmd:

```
vagrant --version
```

For the provisioning part, we have 3 ways to follow as they explained in this github page:

You can run ansible from :

I will follow the third option and use the Ubuntu VM to run the ansible scripts from this VM. I needed to configure the network settings so that all VMs can access each other on the same network. I will show you the network configuration during installation.

I installed Ubuntu 22.04 Desktop on my VMware.

Installing VMs using Vagrant

If you downloaded the GOAD project from Github, we can now run vagrant to build VMs.

Run `vagrant up` on the *..\GOAD-main\ad\GOAD-Light\providers\vmware* folder.

In case you want to install GOAD on Virtualbox, remember to change the location accordingly.

```
PS C:\Users\HuriyeOzdemir\Documents\VMs\GOAD-main\ad\GOAD-Light\providers\vmware> vagrant up
Bringing machine 'GOAD-DC01' up with 'vmware_desktop' provider...
Bringing machine 'GOAD-DC02' up with 'vmware_desktop' provider...
Bringing machine 'GOAD-SRV02' up with 'vmware_desktop' provider...
==> GOAD-DC01: Checking if box 'StefanScherer/windows_2019' version '2021.05.15' is up to date...
==> GOAD-DC01: Verifying vmnet devices are healthy...
The host only network with the IP '192.168.56.10' would collide with
another device 'Ethernet 4'. This means that VMware cannot create
a proper networking device to route to your VM. Please choose
another IP or shut down the existing device.
```

Well, I got my very first error saying that there is a collision on network interfaces trying to use the same IP:

I checked the Vagrant file in the current folder and realised that this configuration gives the following default IPs to 3 VMs:

- GOAD-DC01: 192.168.56.10
- GOAD-DC02: 192.168.56.11
- GOAD-SRV02: 192.168.56.22

Before changing the default IPs, I am going to check the host-only networks on my VMware settings to decide which IP address I should assign.

Let me first explain how the network configuration should be. We need to have 2 different network adapters:

- to put 3 VMs and Ubuntu VM on a same network.
- to put 3 VMs and Ubuntu VM on a network.

Let's create the second adapter (NAT) after building all VMs.

For the host-only network, go to: *VMware* > *Edit* > *Virtual network editor* and check the IP address of the host only network. If you don't have a host-only network, create one.

In my case, VMnet1 is a host only network with 192.168.124.0/24 range.

Virtual Network Editor

Name	Type	External Connection	Host Connection	DHCP	Subnet Address
VMnet0	Bridged	Auto-bridging	-	-	-
VMnet1	Host-only	-	Connected	Enabled	192.168.124.0
VMnet8	NAT	NAT	Connected	Enabled	192.168.229.0

Since I am going to put all VMs to the VMnet1 network, I am going to give random IPs to 3 VMs from this network:

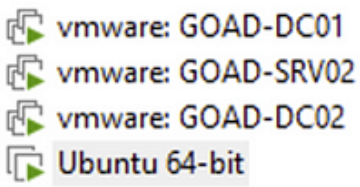
```
Vagrantfile
Vagrant.configure("2") do |config|

# Uncomment this depending on the provider you want to use
ENV['VAGRANT_DEFAULT_PROVIDER'] = 'vmware_desktop'

boxes = [
  # windows server 2019
  { :name => "GOAD-DC01",  :ip => "192.168.124.12", :box => "StefanScherer/windows_2019", :box_version => "2021.05.15", :os => "windows"},
  # windows server 2019
  { :name => "GOAD-DC02",  :ip => "192.168.124.11", :box => "StefanScherer/windows_2019", :box_version => "2021.05.15", :os => "windows"},
  # windows server 2019
  { :name => "GOAD-SRV02", :ip => "192.168.124.22", :box => "StefanScherer/windows_2019", :box_version => "2021.05.15", :os => "windows"},
]
```

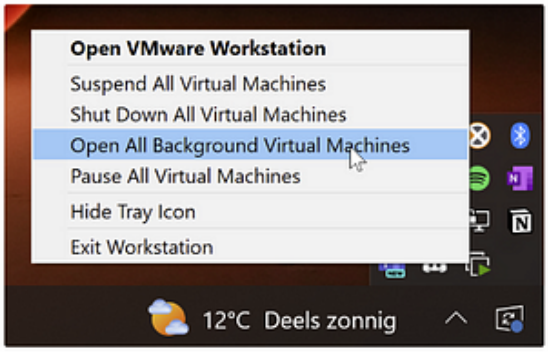
I ran the `vagrant up` command to build VMs again, and it worked without any collision:

```
Select Windows PowerShell
PS C:\Users\HuriyeOzdemir\Documents\VMs\GOAD-main\ad\GOAD-Light\providers\vmware> vagrant up
Bringing machine 'GOAD-DC01' up with 'vmware_desktop' provider...
Bringing machine 'GOAD-DC02' up with 'vmware_desktop' provider...
Bringing machine 'GOAD-SRV02' up with 'vmware_desktop' provider...
==> GOAD-DC01: Checking if box 'StefanScherer/windows_2019' version '2021.05.15' is up to date...
==> GOAD-DC01: Verifying vmnet devices are healthy...
==> GOAD-DC01: Preparing network adapters...
==> GOAD-DC01: Starting the VMware VM...
==> GOAD-DC01: Waiting for the VM to receive an address...
==> GOAD-DC01: Forwarding ports...
    GOAD-DC01: -- 5985 => 55985
    GOAD-DC01: -- 5986 => 55986
    GOAD-DC01: -- 22 => 2222
==> GOAD-DC01: Waiting for machine to boot. This may take a few minutes...
    GOAD-DC01: WinRM address: 127.0.0.1:55985
    GOAD-DC01: WinRM username: vagrant
    GOAD-DC01: WinRM execution_time_limit: PT2H
    GOAD-DC01: WinRM transport: negotiate
==> GOAD-DC01: Machine booted and ready!
==> GOAD-DC01: Configuring network adapters within the VM...
==> GOAD-DC01: Configuring secondary network adapters through VMware
==> GOAD-DC01: on Windows is not yet supported. You will need to manually
==> GOAD-DC01: configure the network adapter.
==> GOAD-DC01: Running provisioner: shell...
    GOAD-DC01: Running: ../../../../vagrant/Install-WMF3Hotfix.ps1 as C:\tmp\vagrant-shell.ps1
==> GOAD-DC01: Running provisioner: shell...
    GOAD-DC01: Running: ../../../../vagrant/ConfigureRemotingForAnsible.ps1 as C:\tmp\vagrant-shell.ps1
    GOAD-DC01: Self-signed SSL certificate generated; thumbprint: 1482615D4D4336849819D5B1C86096BCB8DFAC6C
    GOAD-DC01:
    GOAD-DC01:
    GOAD-DC01: wxf                 : http://schemas.xmlsoap.org/ws/2004/09/transfer
    GOAD-DC01: a                   : http://schemas.xmlsoap.org/ws/2004/08/addressing
    GOAD-DC01: w                   : http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd
    GOAD-DC01: lang                : en-US
    GOAD-DC01: Address             : http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
    GOAD-DC01: ReferenceParameters : ReferenceParameters
    GOAD-DC01:
    GOAD-DC01: Ok.
    GOAD-DC01:
    GOAD-DC01:
    GOAD-DC01:
==> GOAD-DC01: Running provisioner: shell...
    GOAD-DC01: Running: ../../../../vagrant/fix_ip.ps1 as C:\tmp\vagrant-shell.ps1
    GOAD-DC01:
==> GOAD-DC02: Cloning VMware VM: 'StefanScherer/windows_2019'. This can take some time...
==> GOAD-DC02: Checking if box 'StefanScherer/windows_2019' version '2021.05.15' is up to date...
==> GOAD-DC02: Verifying vmnet devices are healthy...
==> GOAD-DC02: Preparing network adapters...
==> GOAD-DC02: Fixed port collision for 5985 => 55985. Now on port 2200.
==> GOAD-DC02: Fixed port collision for 5986 => 55986. Now on port 2201.
```

```
==> GOAD-SRV02: Waiting for machine to boot. This may take a few minutes...
    GOAD-SRV02: WinRM address: 127.0.0.1:2203
    GOAD-SRV02: WinRM username: vagrant
    GOAD-SRV02: WinRM execution_time_limit: PT2H
    GOAD-SRV02: WinRM transport: negotiate
Timed out while waiting for the machine to boot. This means that
Vagrant was unable to communicate with the guest machine within
the configured ("config.vm.boot_timeout" value) time period.

If you look above, you should be able to see the error(s) that
Vagrant had when attempting to connect to the machine. These errors
are usually good hints as to what may be wrong.

If you're using a custom box, make sure that networking is properly
working and you're able to connect to the machine. It is a common
problem that networking isn't setup properly in these boxes.
Verify that authentication configurations are also setup properly,
as well.

If the box appears to be booting properly, you may want to increase
the timeout ("config.vm.boot_timeout") value.
PS C:\Users\HuriyeOzdemir\Documents\VMs\GOAD-main\ad\GOAD-Light\providers\vmware> vagrant up
Bringing machine 'GOAD-DC01' up with 'vmware_desktop' provider...
Bringing machine 'GOAD-DC02' up with 'vmware_desktop' provider...
Bringing machine 'GOAD-SRV02' up with 'vmware_desktop' provider...
==> GOAD-DC01: Checking if box 'StefanScherer/windows_2019' version '2021.05.15' is up to date...
==> GOAD-DC01: Machine is already running.
==> GOAD-DC02: Checking if box 'StefanScherer/windows_2019' version '2021.05.15' is up to date...
==> GOAD-DC02: Machine is already running.
==> GOAD-SRV02: Checking if box 'StefanScherer/windows_2019' version '2021.05.15' is up to date...
==> GOAD-SRV02: Machine is already running.
PS C:\Users\HuriyeOzdemir\Documents\VMs\GOAD-main\ad\GOAD-Light\providers\vmware>
```

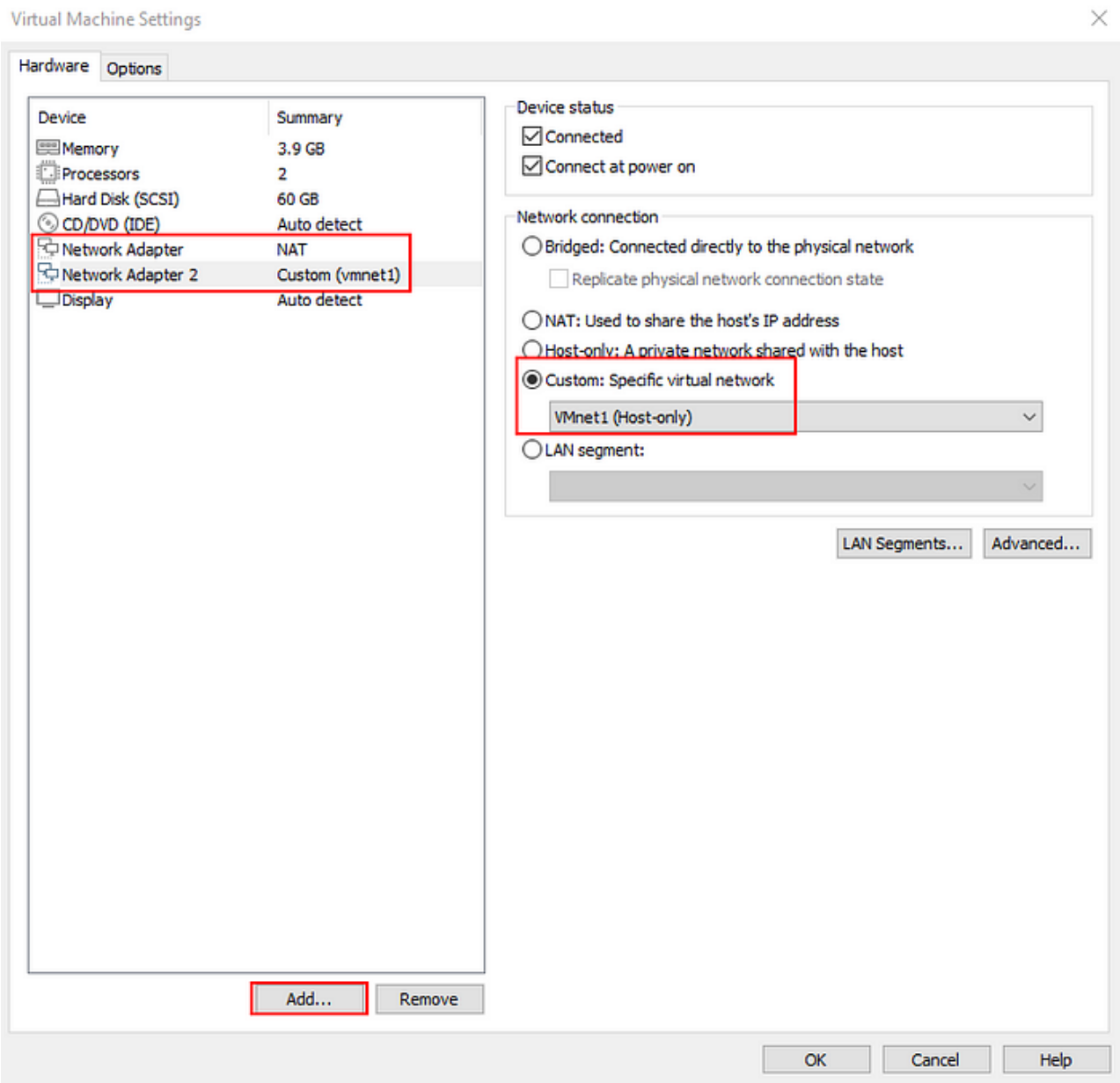
It takes some time to create the VMs and eventually I see that all machines are running. Even if they are actually running, you may not see these machines in your VMware Library. You can select “Open all background virtual machines” from the bottom right menu in the toolbar.





Now, it’s time to configure the network for all VMs. Go to VM settings of each machine and “Add” a new network adapter under Hardware tab.

- Set one network adapter as .
- Set the second adapter as to put all on a same virtual private network.



Provisioning

Great! We have created all the virtual machines, now we need to have a vulnerable active directory environment before practicing the attack techniques!

Let's switch to Ubuntu to start provisioning part and install all requirements before running ansible.

To install python virtual environment:

```
sudo apt install gitgit  git@github.com:Orange-Cyberdefense/GOAD.git
GOAD/ansiblesudo apt install python3.8-venvpython3.8 -m virtualenv .venv
.venv/bin/activate
```

To install ansible pywinrm in the .venv:

```
python3 -m pip install python3 -m pip install ansible-core==python3 -m pip install
pywinrm
```

Install all the ansible-galaxy requirements:

```
ansible-galaxy install -r requirements.yml
```

We will use the following command to run ansible playbooks for GOAD-Light on VMware provider.

```
ansible-playbook -i ../ad/GOAD-Light/data/inventory -i ../ad/GOAD-Light/providers/vmware/inventory main.yml
```

However, we first need to modify the `../ad/GOAD-Light/providers/vmware/inventory` file and change the default IP addresses and replace the IPs we've set at the beginning.

Fixing all errors

Now it's time to run the following command to run playbooks:

```
ansible-playbook -i ../ad/GOAD-Light/data/inventory -i ../ad/GOAD-Light/providers/vmware/inventory main.yml
```

Ok, looks like we have some problems :)
The Ubuntu VM can successfully reach DC01 and DC02, but SRV02 seems to be unreachable for some reason. The first thing I checked is whether the IP address is correctly assigned to this machine.

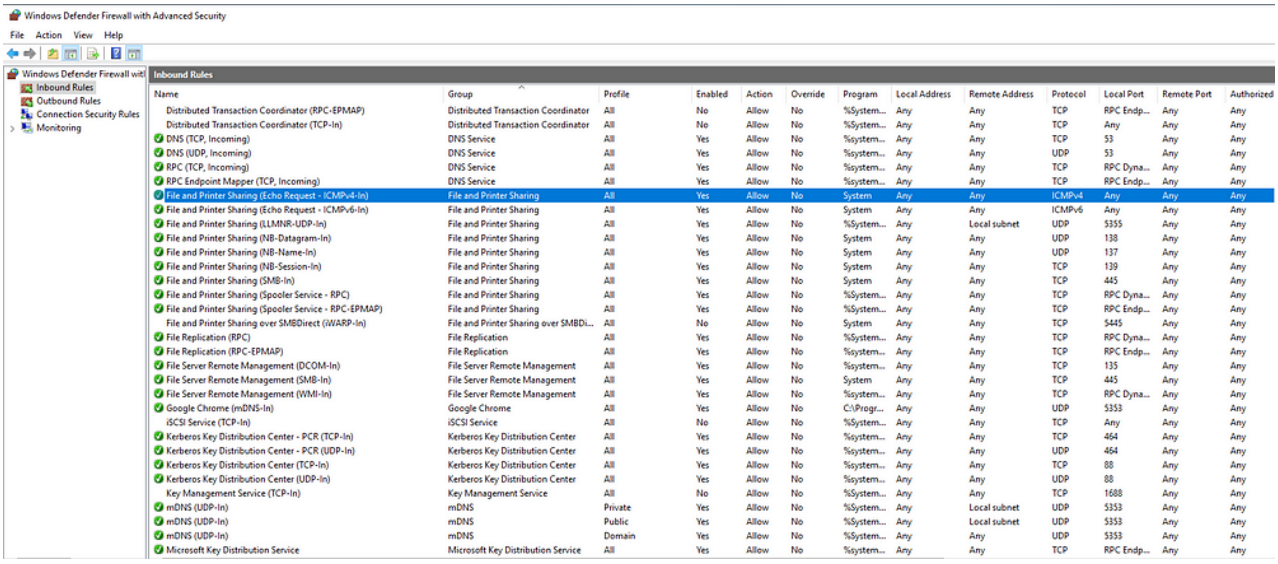
I noticed that the machine received a different IP than the one I set in the vagrant file.

I also checked the troubleshoot page of GOAD and found a solution to this problem.

I replaced the IP addresses of the SRV02 machine with the current IP address and removed the following lines in the `../ad/GOAD-Light/providers/vmware/inventory` file.

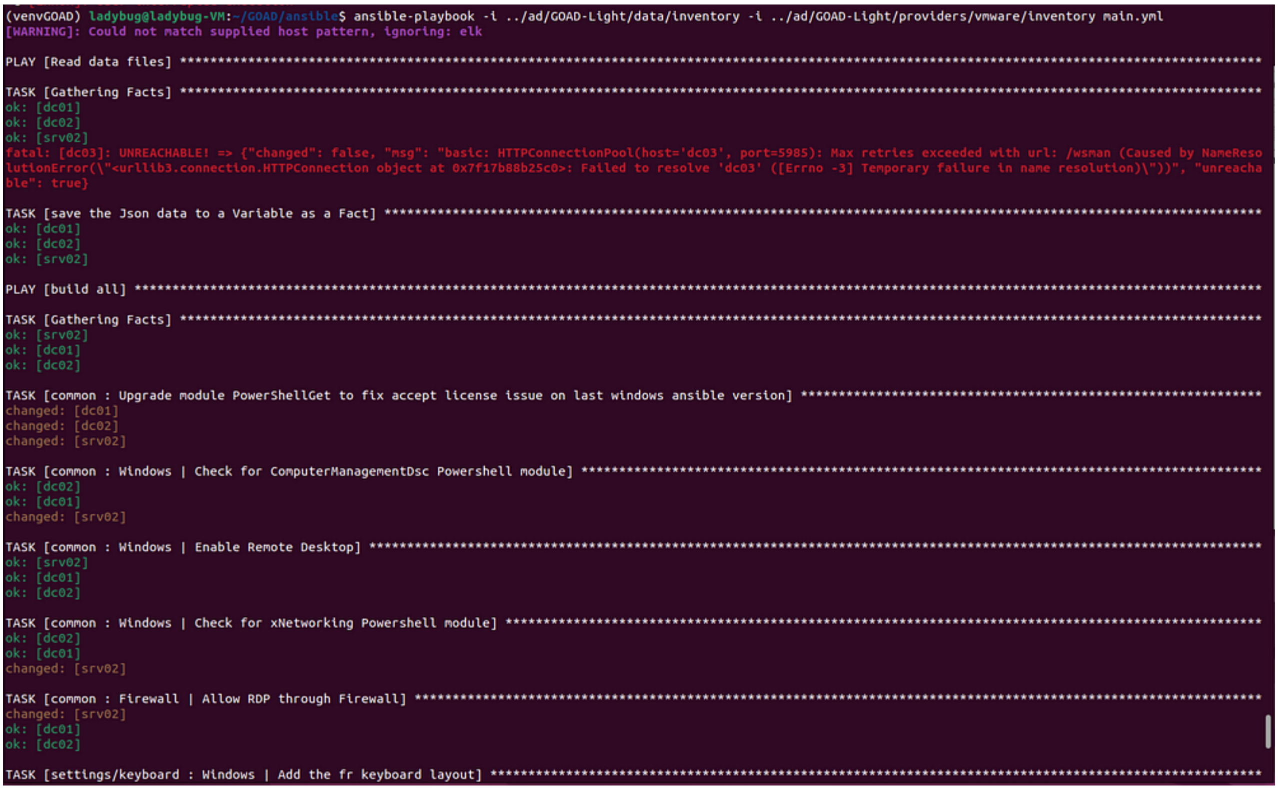
```
; sevenkingdoms.local
; ---------------------------------------------
dc01 ansible_host=192.168.124.12 dns_domain=dc01 dict_key=dc01
; ---------------------------------------------
; north.sevenkingdoms.local
; ---------------------------------------------
dc02 ansible_host=192.168.124.11 dns_domain=dc01 dict_key=dc02
srv02 ansible_host=192.168.124.131 dns_domain=dc02 dict_key=srv02

[all:vars]
; domain_name : folder inside ad/
domain_name=GOAD-Light

force_dns_server=no
dns_server=x.x.x.x
two_adapters=yes

; adapter created by vagrant and vmware (uncomment if you use vmware)
nat_adapter=Ethernet0
domain_adapter=Ethernet1

; winrm connection (windows)
ansible_user=vagrant
ansible_password=vagrant
ansible_connection=winrm
ansible_winrm_server_cert_validation=ignore
ansible_winrm_operation_timeout_sec=400
ansible_winrm_read_timeout_sec=500
ansible_winrm_transport=basic
ansible_port=5985

; proxy settings (the lab need internet for some install, if you are behind a proxy you should set the proxy here)
enable_http_proxy=no
ad_http_proxy=http://x.x.x.x:xxxx
ad_https_proxy=http://x.x.x.x:xxxx
```

I was also unable to ping this machine to see if it was accessible from the Ubuntu VM. I then checked the firewall rules on SRV02 and compared all the rules with the other machines (DC01 and DC02). There were some differences in the rules that prevented me from sending a ping request. I enabled the file and printer sharing rules and made all the rules the same as the other machines.



Let’s run ansible-playbook again!

`ansible-playbook -i ../ad/GOAD-Light/data/inventory -i ../ad/GOAD-Light/providers/vmware/inventory main.yml`



```
(venvGOAD) ladybug@ladybug-VM:~/GOAD/ansible$ ansible-playbook -i ../ad/GOAD-Light/data/inventory -i ../ad/GOAD-Light/providers/vmware/inventory main.yml
[WARNING]: Could not match supplied host pattern, ignoring: elk

PLAY [Read data files] *************************************************************************************************************************

TASK [Gathering Facts] *************************************************************************************************************************
ok: [dc01]
ok: [dc02]
ok: [srv02]
fatal: [dc03]: UNREACHABLE! => {"changed": false, "msg": "basic: HTTPConnectionPool(host='dc03', port=5985): Max retries exceeded with url: /wsman (Caused by NameReso
lutionError(\"<urllib3.connection.HTTPConnection object at 0x7f17b88b25c0>: Failed to resolve 'dc03' ([Errno -3] Temporary failure in name resolution)\"))", "unreacha
ble": true}

TASK [save the Json data to a Variable as a Fact] **********************************************************************************************
ok: [dc01]
ok: [dc02]
ok: [srv02]

PLAY [build all] *******************************************************************************************************************************

TASK [Gathering Facts] *************************************************************************************************************************
ok: [srv02]
ok: [dc01]
ok: [dc02]

TASK [common : Upgrade module PowerShellGet to fix accept license issue on last windows ansible version] ***************************************
changed: [dc01]
changed: [dc02]
changed: [srv02]

TASK [common : Windows | Check for ComputerManagementDsc Powershell module] *********************************************************************
ok: [dc02]
ok: [dc01]
changed: [srv02]

TASK [common : Windows | Enable Remote Desktop] ************************************************************************************************
ok: [srv02]
ok: [dc01]
ok: [dc02]

TASK [common : Windows | Check for xNetworking Powershell module] ******************************************************************************
ok: [dc02]
ok: [dc01]
changed: [srv02]

TASK [common : Firewall | Allow RDP through Firewall] ******************************************************************************************
changed: [srv02]
ok: [dc01]
ok: [dc02]

TASK [settings/keyboard : Windows | Add the fr keyboard layout] ********************************************************************************
```

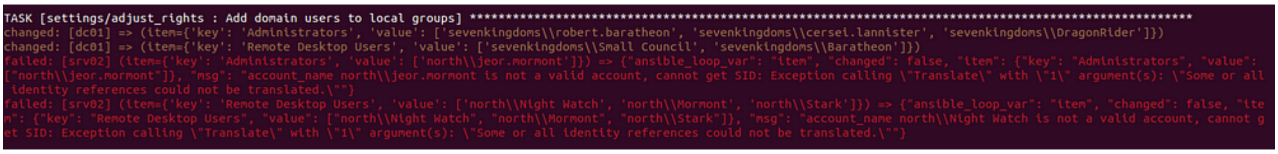
All servers are reachable by Ubuntu now. You can ignore the error for DC03, if you want to setup GOAD-Light. We just need 2 DCs and 1 server for this setup.

All playbooks completed but we have another issues :

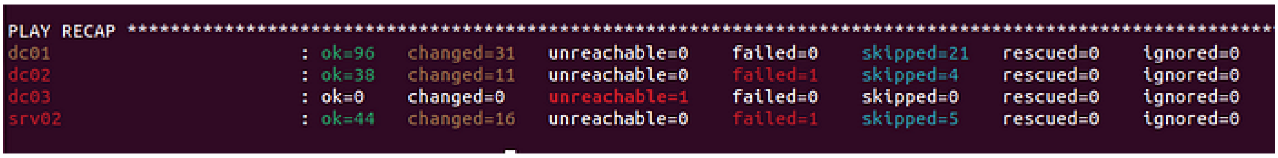
Ansible failed to;

- Set a password policy for DC02
- Add a domain user to local groups (invalid user?)

```
TASK [password_policy : set password policy] ***************************************************************************************************
changed: [dc01]
fatal: [dc02]: FAILED! => {"changed": true, "debug": [], "error": [{"category_info": {"activity": "Set-ADDefaultDomainPasswordPolicy", "category": "ResourceUnavailable", "categ
ory_id": 21, "reason": "ADServerDownException", "target_name": "north.sevenkingdoms.local", "target_type": "ADDefaultDomainPasswordPolicy"}, "error_details": null, "exception":
 {"help_link": null, "hresult": -2146233088, "inner_exception": {"help_link": null, "hresult": -2146233088, "inner_exception": null, "message": "Unable to find a default server
 with flags: 'Writable | ADWS | ReturnDnsName'.", "source": null, "type": "Microsoft.ActiveDirectory.Management.ADException"}, "message": "Unable to find a default server with
Active Directory Web Services running.", "source": "Microsoft.ActiveDirectory.Management", "type": "Microsoft.ActiveDirectory.Management.ADServerDownException"}, "fully_qualifi
ed_error_id": "ActiveDirectoryServer:1355,Microsoft.ActiveDirectory.Management.Commands.SetADDefaultDomainPasswordPolicy", "output": "Set-ADDefaultDomainPasswordPolicy : Unable
 to find a default server with Active Directory Web Services running.\r\nAt line:19 char:1\r\n+ Set-ADDefaultDomainPasswordPolicy -Identity $Domain -AuthType Negotia ...\r\n+ ~
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\r\n    + CategoryInfo          : ResourceUnavailable: (north.sevenkingdoms.local:ADDefaultDomainPasswordPol
icy) \r\n[Set-ADDefaultDomainPasswordPolicy], ADServerDownException\r\n    + FullyQualifiedErrorId : \r\nActiveDirectoryServer:1355,Microsoft.ActiveDirectory.Management.Command
s.SetADDefaultDomainPasswordPolicy\r\n \r\n", "pipeline_iteration_info": [0, 1], "script_stack_trace": "at <ScriptBlock>, <No file>: line 19", "target_object": {"AddedPropertie
s": {"Comparer": {}, "Count": 0, "IsReadOnly": false}, "ComplexityEnabled": null, "LockoutDuration": null, "LockoutObservationWindow": null, "LockoutThreshold": null, "MaxPassw
ordAge": null, "MinPasswordAge": null, "MinPasswordLength": null, "ModifiedProperties": {"Comparer": {}, "Count": 0, "IsReadOnly": false}, "PasswordHistoryCount": null, "Proper
tyCount": 0, "PropertyNames": {"Count": 0, "IsReadOnly": true, "IsSynchronized": false, "SyncRoot": {}}, "RemovedProperties": {"Comparer": {}, "Count": 0, "IsReadOnly": false},
 "ReversibleEncryptionEnabled": null}}], "host_err": "", "host_out": "", "information": [], "output": [], "result": {}, "verbose": [], "warning": []}
```



```
TASK [settings/adjust_rights : Add domain users to local groups] *******************************************************************************
changed: [dc01] => (item={'key': 'Administrators', 'value': ['sevenkingdoms\\robert.baratheon', 'sevenkingdoms\\cersei.lannister', 'sevenkingdoms\\DragonRider']})
changed: [dc01] => (item={'key': 'Remote Desktop Users', 'value': ['sevenkingdoms\\Small Council', 'sevenkingdoms\\Baratheon']})
failed: [srv02] (item={'key': 'Administrators', 'value': ['north\\jeor.mormont']}) => {"ansible_loop_var": "item", "changed": false, "item": {"key": "Administrators", "value":
["north\\jeor.mormont"]}, "msg": "account_name north\\jeor.mormont is not a valid account, cannot get SID: Exception calling \"Translate\" with \"1\" argument(s): \"Some or all
 identity references could not be translated.\""}
failed: [srv02] (item={'key': 'Remote Desktop Users', 'value': ['north\\Night Watch', 'north\\Mormont', 'north\\Stark']}) => {"ansible_loop_var": "item", "changed": false, "ite
m": {"key": "Remote Desktop Users", "value": ["north\\Night Watch", "north\\Mormont", "north\\Stark"]}, "msg": "account_name north\\Night Watch is not a valid account, cannot g
et SID: Exception calling \"Translate\" with \"1\" argument(s): \"Some or all identity references could not be translated.\""}
```



```
PLAY RECAP *********************************************************************************************
dc01                       : ok=96   changed=31   unreachable=0    failed=0    skipped=21   rescued=0    ignored=0
dc02                       : ok=38   changed=11   unreachable=0    failed=1    skipped=4    rescued=0    ignored=0
dc03                       : ok=0    changed=0    unreachable=1    failed=0    skipped=0    rescued=0    ignored=0
srv02                      : ok=44   changed=16   unreachable=0    failed=1    skipped=5    rescued=0    ignored=0
```

First recap with 2 failed tasks

We ran the main.yml under ansible folder to run ALL playbooks. We can also run the playbooks one by one. I checked the inventory file that **ad-data.yml** file is trying to set password policy for DCs. Then I tried to run it again to fix the error:

```
ansible-playbook -i ../ad/GOAD-Light/data/inventory -i ../ad/GOAD-Light/providers/vmware/inventory ad-data.yml
```

I got the same error again and again.

I found this issue previously posted on github:

According to the recommended solution:

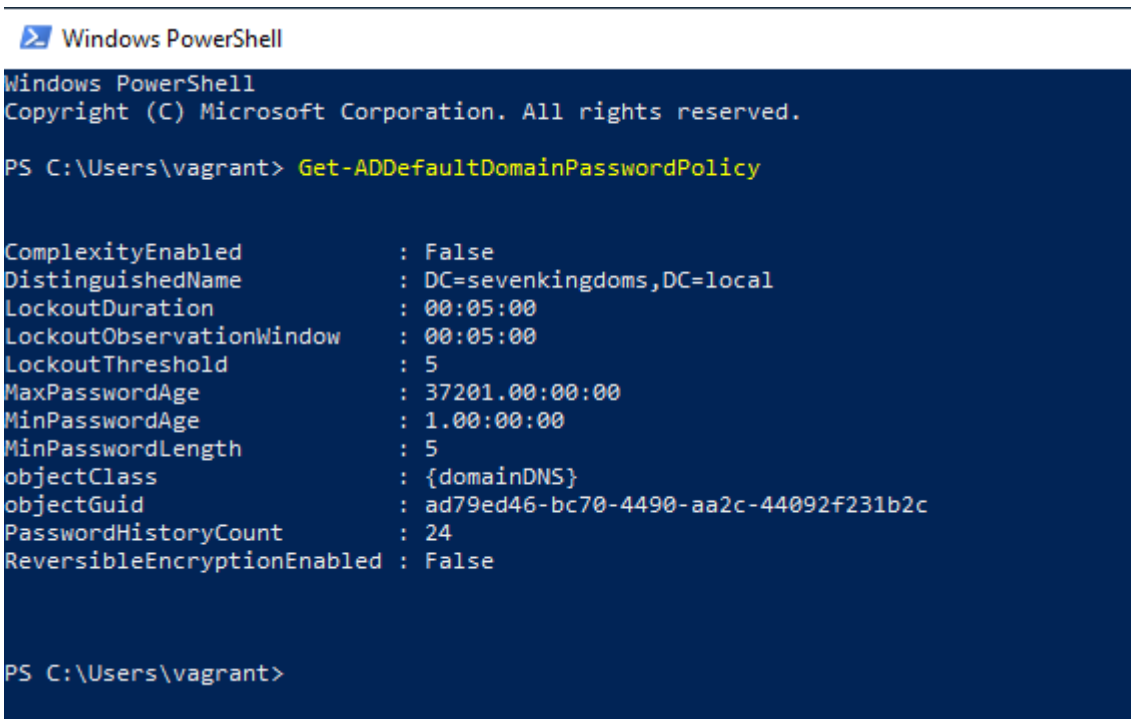
- We should check if the DNS server of DC02 is correctly set as the IP address of DC01.
- Disable the NAT network during domain installation.

I checked DNS server and disabled NAT networks on all VMs and ran the playbook again.

Unfortunately it did not work… I gave up setting it with ansible and decided to set it manually on the DC02.

First, I decided to check the password policy configuration on DC01 before setting DC02. I was able to run the following command on powershell of DC01 successfully.

```
ADDefaultDomainPasswordPolicy
```

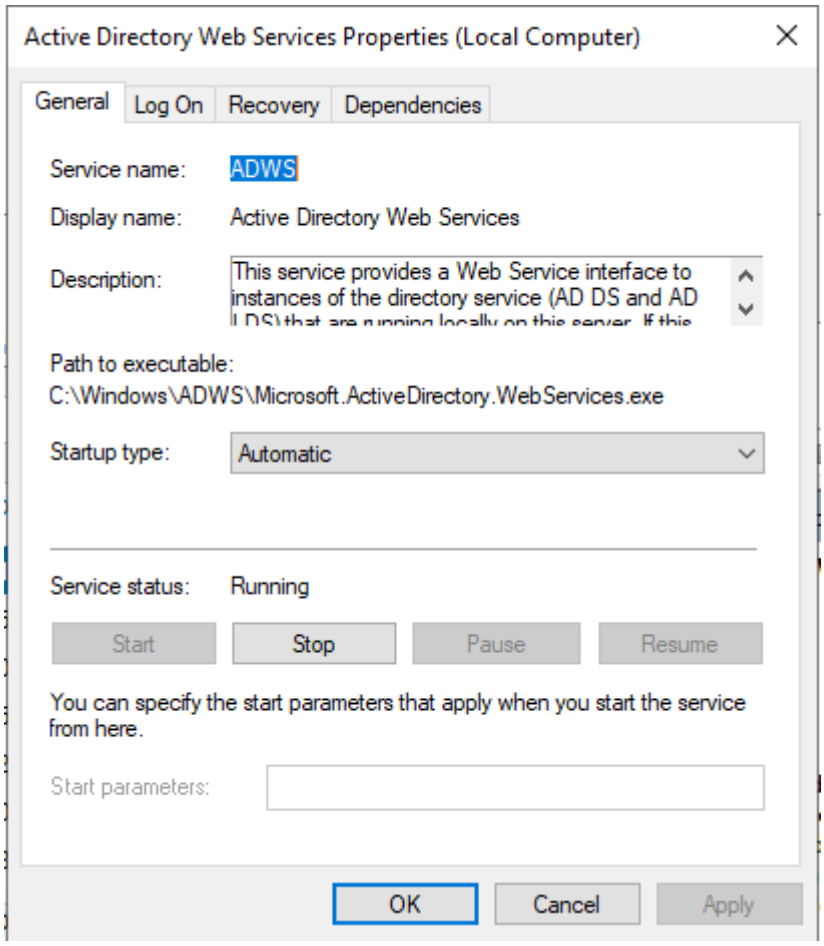


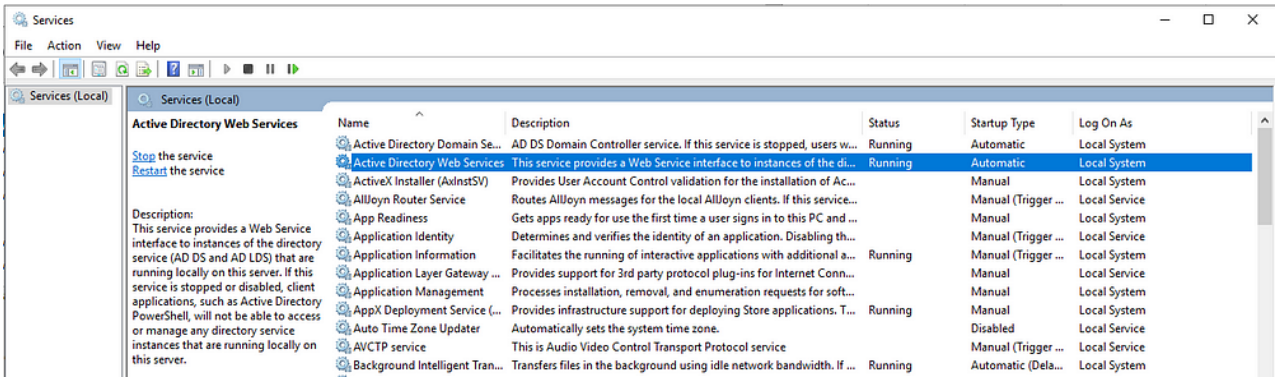
Password policy of DC01

However, when I tried to check the password policy on DC02, I got this error: ADWS (Active Directory Web Services) is not running.

```
PS C:\Users\robb.stark>  Get-ADDefaultDomainPasswordPolicy
Get-ADDefaultDomainPasswordPolicy : Unable to find a default server with Active Directory Web Services running.
At line:1 char:2
+  Get-ADDefaultDomainPasswordPolicy
+  ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : ResourceUnavailable: (NORTH:ADDefaultDomainPasswordPolicy) [Get-ADDefaultDomainPasswordPolicy], ADServerDownException
    + FullyQualifiedErrorId : ActiveDirectoryServer:1355,Microsoft.ActiveDirectory.Management.Commands.GetADDefaultDomainPasswordPolicy
```

Then I understood **why ansible did not work** on Ubuntu. I checked “Services” and realised that ADWS was disabled and not running. Then, I changed the startup type to Automatic and started the service.





Now, we can try to run ansible playbook for setting the password policy again!

```
ansible-playbook -i ../ad/GOAD-Light/data/inventory -i ../ad/GOAD-Light/providers/vmware/inventory ad-data.yml
```

```
TASK [password_policy : set password policy] *****
changed: [dc02]
changed: [dc01]
```

It worked!

I ran all playbooks to see the final table again and this was the recap table after running the main.yml:

```
PLAY RECAP *********************************************************************************************************************
dc01                       : ok=77   changed=13   unreachable=0    failed=0    skipped=16   rescued=0    ignored=0
dc02                       : ok=68   changed=14   unreachable=0    failed=0    skipped=14   rescued=0    ignored=0
dc03                       : ok=0    changed=0    unreachable=1    failed=0    skipped=0    rescued=0    ignored=0
srv02                      : ok=68   changed=17   unreachable=0    failed=1    skipped=8    rescued=0    ignored=0
```

Interestingly, the second error I mentioned above (invalid user) was gone but I had another error about installing the mssql database for SRV02.

```
TASK [mssql : Install the database] ********************************************************************************************
fatal: [srv02]: FAILED! => {"attempts": 3, "changed": true, "cmd": "c:\\setup\\mssql\\sql_installer.exe /configurationfile=c:\\setup\\mssql\\sql_conf.ini /IACCEPTSQLSERVERLICENSETERMS /MEDIAPATH=c:\\setup\\mssql\\media /QUIET /INSTANCEDIR=C:\Program Files\Microsoft SQL Server\", "delta": "0:00:21.79
6475", "end": "2024-01-29 12:15:43.0065067", "msg": "non-zero return code", "rc": 22263234158, "start": "2024-01-29 12:15:21.2100317", "stderr": "", "stderr_lines": [], "stdout": "Microsoft (R) SQL Server Installer\r\nCopyright (c) 2019 Microsoft.  All rights reserved
.\r\n\r\nDownloading install package...\r\n\r\n\r\nOperation finished with result: Failure\r\n\r\nOops...\r\n\r\nUnable to install SQL Server (setup.exe).\r\n\r\n      Exit code (Decimal): -2068643838\r\n      Exit message: No features were installed during the set
up execution. The requested features may already be installed. Please review the summary.txt log for further details.\r\n\r\n  SQL SERVER INSTALL LOG FOLDER\r\n      c:\\Program Files\\Microsoft SQL Server\\150\\Setup Bootstrap\\Log\\20240129_041525\r\n\r\n", "stdo
ut_lines": ["Microsoft (R) SQL Server Installer", "Copyright (c) 2019 Microsoft.  All rights reserved.", "", "Downloading install package...", "", "", "Operation finished with result: Failure", "", "Oops...", "", "Unable to install SQL Server (setup.exe).", "
    Exit code (Decimal): -2068643838", "      Exit message: No features were installed during the setup execution. The requested features may already be installed. Please review the summary.txt log for further details.", "", "  SQL SERVER INSTALL LOG FOLDER", "
    c:\\Program Files\\Microsoft SQL Server\\150\\Setup Bootstrap\\Log\\20240129_041525", ""]}
```

I checked the inventory file to find the related ansible file for this error: server.yml

```
Open ▾   inventory   ~/GOAD/ad/GOAD-Light/providers/vmware   Save
87 ; usage : adcs.yml
88 [adcs_customtemplates]
89 dc03
90
91 ; install iis with default website asp upload on 80
92 ; usage : servers.yml
93 [iis]
94 srv02
95
96 ; install mssql
97 ; usage : servers.yml
98 [mssql]
99 srv02
100
101 ; install mssql gui
102 ; usage : servers.yml
103 [mssql_ssms]
104 srv02
105
106 ; install webdav
107 [webdav]
108 srv02
109
110 ; install elk
111 ; usage : elk.yml
112 [elk_server]
113
114 ; add log agent for elk
115 ; usage : elk.yml
116 [elk_log]
117
118 ; allow computer update
119 ; usage : update.yml
120 [update]
121 srv02
122
123 ; disable update
Plain Text ▾   Tab Width: 8 ▾   Ln 104, Col 6   INS
```

I realised that sometimes servers are not properly running while ansible playbooks are trying to configure the server. I constantly checked the servers and Server manager to fix the possible errors.

I ran this file again to fix the database error hopefully.

```
ansible-playbook -i ../ad/GOAD-Light/data/inventory -i ../ad/GOAD-Light/providers/vmware/inventory servers.yml
```

```
PLAY RECAP *********************************************************************************************************
dc01                       : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
dc02                       : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
dc03                       : ok=0    changed=0    unreachable=1    failed=0    skipped=0    rescued=0    ignored=0
srv02                      : ok=38   changed=16   unreachable=0    failed=0    skipped=6    rescued=0    ignored=0

ladybug@ladybug-VM:~/GOAD/ansible$
```

recap of servers.yml

Finally it worked and all failed tasks were fixed successfully!

Double-check all playbooks with main.yml:

```
ansible-playbook -i ../ad/GOAD-Light/data/inventory -i ../ad/GOAD-Light/providers/vmware/inventory main.yml
```

```
PLAY RECAP *********************************************************************************************************
dc01                       : ok=96   changed=18   unreachable=0    failed=0    skipped=21   rescued=0    ignored=0
dc02                       : ok=95   changed=31   unreachable=0    failed=0    skipped=18   rescued=0    ignored=0
dc03                       : ok=0    changed=0    unreachable=1    failed=0    skipped=0    rescued=0    ignored=0
srv02                      : ok=107  changed=28   unreachable=0    failed=0    skipped=19   rescued=0    ignored=0
```

the last recap of main.yml

Thanks for being patient and reading until the last step!