

GOAD v3 — Simple Install— Ubuntu & Troubleshooting

[Medium.com/@shanksf/goad-v3-simple-install-ubuntu-troubleshooting-7505de7e7322](https://medium.com/@shanksf/goad-v3-simple-install-ubuntu-troubleshooting-7505de7e7322)

Shanks

December 16, 2024



Shanks



GOAD v3

The new version of **GOAD** is out and it's better than v2, so I'll show you how to get the most out of it **first try**, without **no(t)** (much) troubleshooting.

This was fully tested only on a Windows 10 machine with **64GB of RAM**, using **VMware** and **Ubuntu**.

This may also work for **GOAD-light**, **GOAD-Mini**, **SSCM**, **NHA** and so on.

It's the easiest guide on the internet for using and pwning GOAD.

If you wanna use the v2, here's the tutorial for .

Leave a comment if works for you. :)

| *Main project by Mayfly:*

- PREREQUISITES
- STEPS
- TROUBLESHOOTING

PREREQUISITES

- VMWARE Workstation (Also may work with Virtualbox, just adapt)
- Kali or Ubuntu (Choose any distro you want, I'll use Ubuntu just for demonstration)
- A lot of disk space (~120GB)
- A lot of ram (At least 32GB for the main GOAD)

STEPS

After installing and setting up Ubuntu just run the following:

```

# Install vbox
sudo apt install virtualbox

# Install vagrant
wget -O- https://apt.releases.hashicorp.com/gpg | sudo gpg --dearmor -o
/usr/share/keyrings/hashicorp-archive-keyring.gpg
echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg]
https://apt.releases.hashicorp.com main" | sudo tee
/etc/apt/sources.list.d/hashicorp.list
sudo apt update && sudo apt install vagrant

# Install Vagrant plugins
vagrant plugin install vagrant-reload vagrant-vbguest winrm winrm-fs winrm-
elevated

# Add some dependencies
sudo apt install git sshpass lftp rsync openssh-client python3-venv

git clone https://github.com/Orange-Cyberdefense/GOAD.git
cd GOAD

# verify installation
./goad.sh -t check -l GOAD -p virtualbox

# install
./goad.sh -t install -l GOAD -p virtualbox

# launch goad in interactive mode
./goad.sh

```

Make sure you have enough RAM.
If you have any doubts, just use “**install**”.

```
GOAD/virtualbox/local/192.168.56.X (bed9fa-goad-virtualbox) > ?
```

```
*** Manage Lab instance commands ***
status ..... show current status
start ..... start lab
stop ..... stop lab
destroy ..... destroy lab

*** Manage one vm commands ***
start_vm <vm_name> ..... start selected virtual machine
stop_vm <vm_name> ..... stop selected virtual machine
restart_vm <vm_name> ..... restart selected virtual machine
destroy_vm <vm_name> ..... destroy selected virtual machine

*** Extensions ***
list_extensions ..... list extensions
install_extension <extension> ..... install extension (providing + provisioning)
provision_extension <extension> ..... provision extension (provisioning only)

*** Providing (Vagrant/Terraform) ***
provide ..... run only the providing (vagrant/terraform) command

*** Provisioning (Ansible) ***
provision <playbook> ..... run specific ansible playbook
provision_lab ..... run all the current lab ansible playbooks
provision_lab_from <playbook> ..... run all the current lab ansible playbooks

*** Lab Instances ***
check ..... check dependencies before creation
install ..... install the current instance (providing + provisioning)
set_as_default ..... set instance as default
update_instance_files ..... update lab instance files
disable_vagrant ..... disable vagrant user
enable_vagrant ..... enable vagrant user
list ..... list lab instances
```

Options with “?”

It'll start downloading the VMs, just be patient.

Grab a coffee ☕ Read a book 📖 Watch an anime 🎬 Go for a run 🏃

```
==> GOAD-DC01: Importing base box 'StefanScherer/windows_2019'...
==> GOAD-DC01: Matching MAC address for NAT networking...
==> GOAD-DC01: Checking if box 'StefanScherer/windows_2019' version '2021.05.15' is up to date
==> GOAD-DC01: Setting the name of the VM: GOAD-DC01
==> GOAD-DC01: Clearing any previously set network interfaces...
==> GOAD-DC01: Preparing network interfaces based on configuration...
    GOAD-DC01: Adapter 1: nat
    GOAD-DC01: Adapter 2: hostonly
==> GOAD-DC01: Forwarding ports...
    GOAD-DC01: 5985 (guest) => 55985 (host) (adapter 1)
    GOAD-DC01: 5986 (guest) => 55986 (host) (adapter 1)
    GOAD-DC01: 22 (guest) => 2222 (host) (adapter 1)
==> GOAD-DC01: Running 'pre-boot' VM customizations...
==> GOAD-DC01: Booting VM...
==> 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: Checking for guest additions in VM...
    GOAD-DC01: The guest additions on this VM do not match the installed version of
    GOAD-DC01: VirtualBox! In most cases this is fine, but in rare cases it can
    GOAD-DC01: prevent things such as shared folders from working properly. If you see
    GOAD-DC01: shared folder errors, please make sure the guest additions within the
    GOAD-DC01: virtual machine match the version of VirtualBox you have installed on
    GOAD-DC01: your host and reload your VM.
    GOAD-DC01:
    GOAD-DC01: Guest Additions Version: 6.1.22
    GOAD-DC01: VirtualBox Version: 7.0
==> GOAD-DC01: Configuring and enabling network interfaces...
==> 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-she
```

Downloading and setting up the machines

```

PLAY RECAP ****
dc01 : ok=11    changed=2      unreachable=0    failed=0
dc02 : ok=31    changed=14     unreachable=0    failed=0
dc03 : ok=8     changed=3      unreachable=0    failed=0
srv02 : ok=18    changed=10     unreachable=0    failed=0
srv03 : ok=12    changed=8      unreachable=0    failed=0

[*] Lab successfully provisioned in 22:43:43
GOAD/virtualbox/local/192.168.56.X (bed9fa-goad-virtualbox) > status
[*] CWD: /workspace/bed9fa-goad-virtualbox/provider
[*] Running command : vagrant status
Current machine states:

GOAD-DC01           running (virtualbox)
GOAD-DC02           running (virtualbox)
GOAD-DC03           running (virtualbox)
GOAD-SRV02          running (virtualbox)
GOAD-SRV03          running (virtualbox)

This environment represents multiple VMs. The VMs are all listed above with their current state. For more information about a specific VM, run `vagrant status NAME`.

```

Machines are all set up, use “status” to check if they’re running

Awesome, the labs are all installed and running.

```

shanks@akagami:~/GOAD$ nxc smb 192.168.56.0/24
SMB      192.168.56.22  445    CASTELBLACK      [*] Windows 10 / Server 2019
orth.sevenkingdoms.local) (signing:False) (SMBv1:False)
SMB      192.168.56.11  445    WINTERFELL      [*] Windows 10 / Server 2019
rth.sevenkingdoms.local) (signing:True) (SMBv1:False)
SMB      192.168.56.12  445    MEEREN         [*] Windows Server 2016 Standard
ain:essos.local) (signing:True) (SMBv1:True)
SMB      192.168.56.23  445    BRAAVOS        [*] Windows Server 2016 Standard
ain:essos.local) (signing:False) (SMBv1:True)
SMB      192.168.56.10  445    KINGSLANDING   [*] Windows 10 / Server 2019
sevenkingdoms.local) (signing:True) (SMBv1:False)
Running nxc against 256 targets ━━━━━━━━━━━━━━━━━━━ 100% 0:0
shanks@akagami:~/GOAD$ 

```

Check with nxc if you can reach them

Running you’ll see that all the machines are working! 🙌

Really, that’s all of it. Just start hacking it. Have fun! 🎉

TROUBLESHOOTING

Usually you won’t have problems, but if you do and happens to be about ‘INTEL VT-x/EPT or AMD-V/RVI’ here’s the solution:

```

GOAD-DC01: 22 (guest) => 2222 (host) (adapter 1)
==> GOAD-DC01: Running 'pre-boot' VM customizations...
==> GOAD-DC01: Booting VM...
There was an error while executing 'VBoxManage', a CLI used by Vagrant
for controlling VirtualBox. The command and stderr is shown below.

Command: ["startvm", "734544e2-36fb-4d8e-98af-11449818249b", "--type", "headless"]

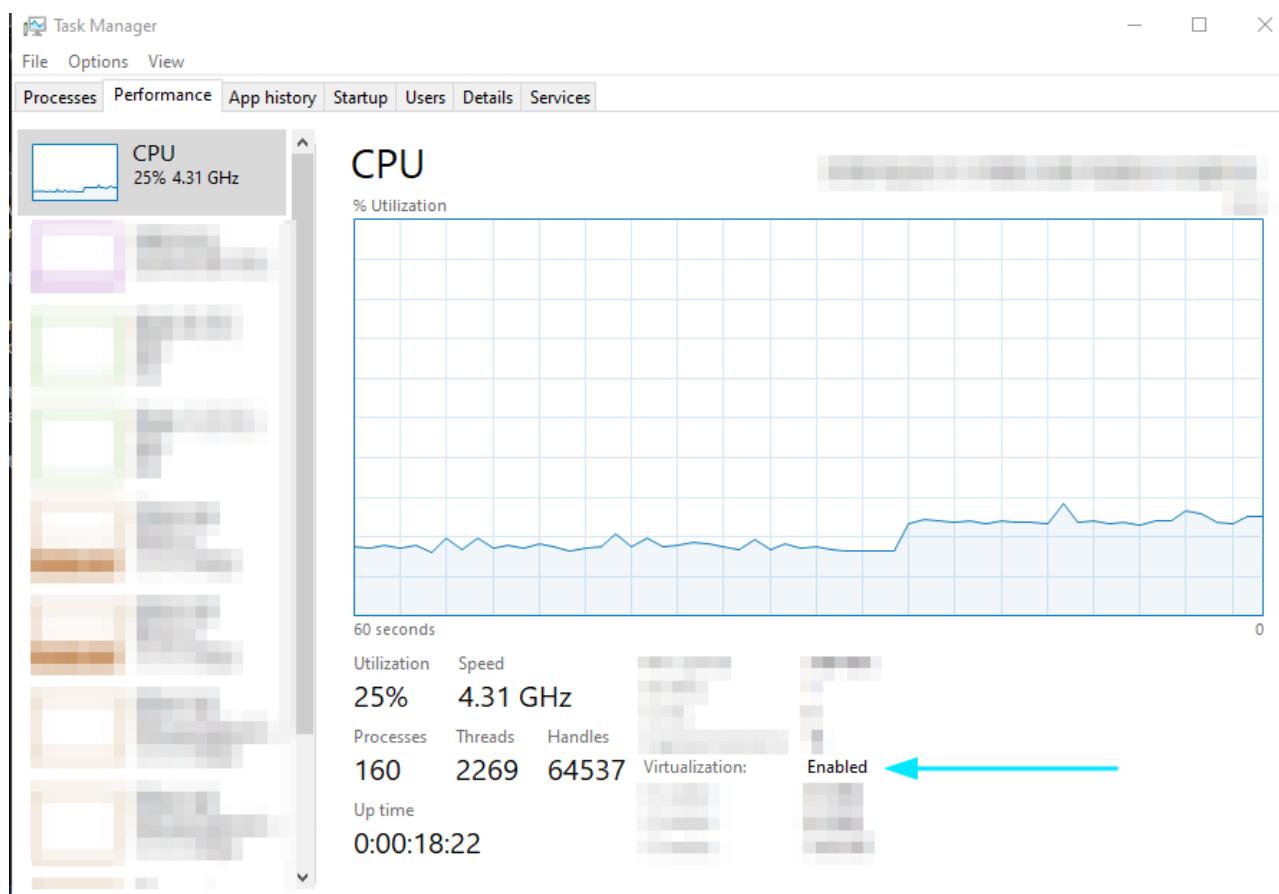
Stderr: VBoxManage: error: AMD-V is not available (VERR_SVM_NO_SVM)
VBoxManage: error: Details: code NS_ERROR_FAILURE (0x80004005), component ConsoleWrap, interface IConsole
[-] Providing error stop

```

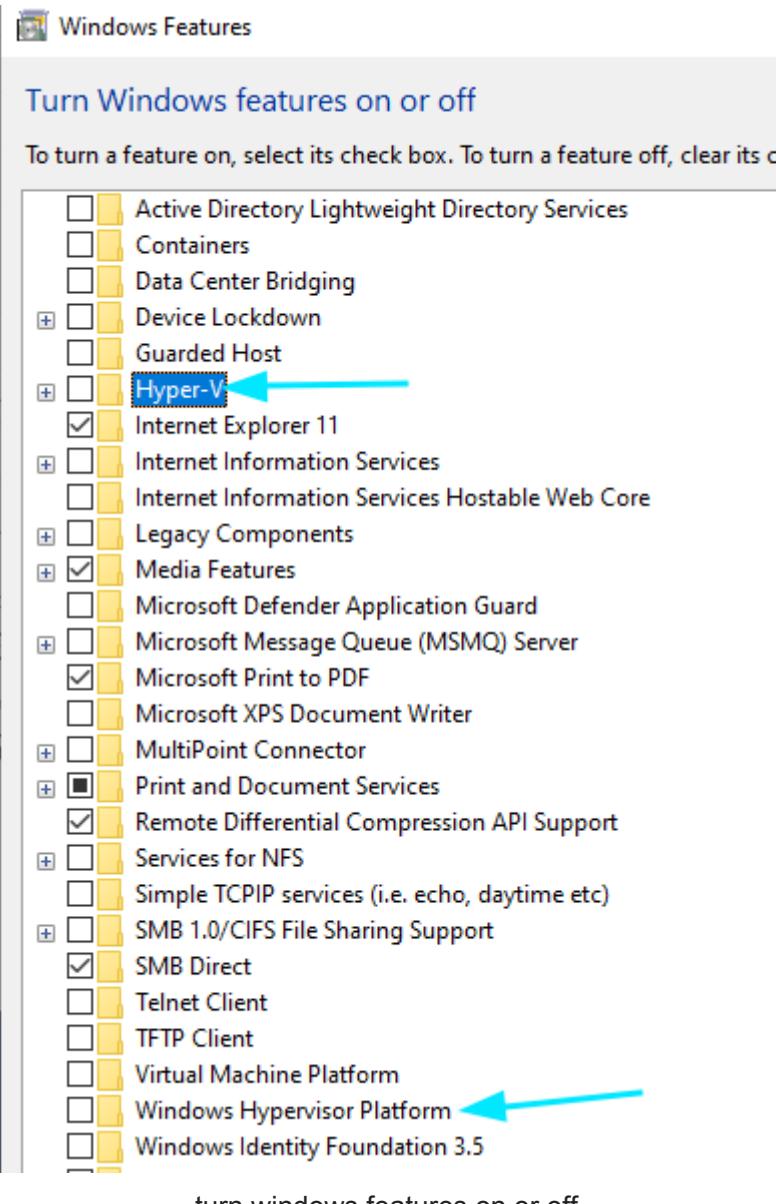
AMD-V error

First of, you need to **enable Virtualization** in your **BIOS**. You can google search on how to do that for your motherboard.

You can check if you need to do this by going to your **CPU tab** using Task Manager like so:



After enabling it, you need to **turn off any hypervisor** you have, you can do that by using the “**Turn Windows features on or off**”, cmd or Powershell if some doesn't work. Don't forget to **restart** your computer.



turn windows features on or off

```
C:\Users\Administrator>bcdedit /set hypervisorlaunchtype off
The operation completed successfully.

C:\Users\Administrator>
```

cmd option

bcdedit / hypervisorlaunchtype

```
PS C:\Users\Administrator> Disable-WindowsOptionalFeature -Online -FeatureName Microsoft-Hyper-V-All
Do you want to restart the computer to complete this operation now?
[Y] Yes [N] No [?] Help (default is "Y"):
```

Powershell option

Disable-WindowsOptionalFeature -Online -FeatureName Microsoft-Hyper-V-

Go to your services and **stop/disable all related to Hyper-V**.

HV Host Service	Manual (
Hyper-V Data Exchange Service	Manual (
Hyper-V Guest Service Interface	Manual (
Hyper-V Guest Shutdown Service	Manual (
Hyper-V Heartbeat Service	Manual (
Hyper-V Host Compute Service	Manual (
Hyper-V PowerShell Direct Service	Manual (
Hyper-V Remote Desktop Virtualization Service	Manual (
Hyper-V Time Synchronization Service	Manual (
Hyper-V Virtual Machine Management	Automatic
Hyper-V Volume Shadow Copy Requestor	Manual (

disable all

Then, go to your Ubuntu and enable the option below.

Virtual Machine Settings

Device	Summary
Memory	32 GB
Processors	2
Hard Disk (SCSI)	200 GB
CD/DVD 2 (SATA)	Using file C:\Users\Administra...
CD/DVD (SATA)	Using file autoinst.iso
Floppy	Using file autoinst.flp
Network Adapter	NAT
Network Adapter 2	Custom (VMnet1)
USB Controller	Present
Sound Card	Auto detect
Display	Auto detect

Processors

Number of processors:

Number of cores per processor:

Total processor cores: 2

Virtualization engine

Virtualize Intel VT-x/EPT or AMD-V/RVI

Virtualize CPU performance counters

Enable the virtualize option

By doing those, you can start your Ubuntu normally and install GOAD.