

Milestone 12 - HyperV - Linked Clones and Automation 1

 Linked Clones are exceptionally handy when fast deploys are needed and when a good deal of the OS's disk footprint can remain static. They become less so when massive feature and security updates occur frequently. We are going to use a relatively painful process to create a linked clone with the goal of automating this with a couple lines of powershell at the end of this milestone.

Parent Disk

Create an ubuntu22.04-base virtual machine and prepare it for cloning.

You can access an ubuntu ISO (live server...) at:

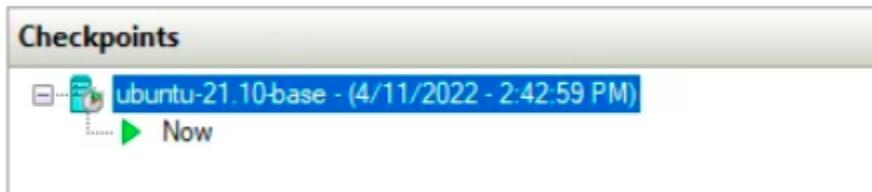
<http://192.168.7.241/isos/>

- VM Generation: 1
- Network: HyperV-WAN
- At least 2GiB ram

Take a look at the following [script](#) (from [hyperv-ubuntu-sealer](#)) which was modified for hyper v instead of vmware.

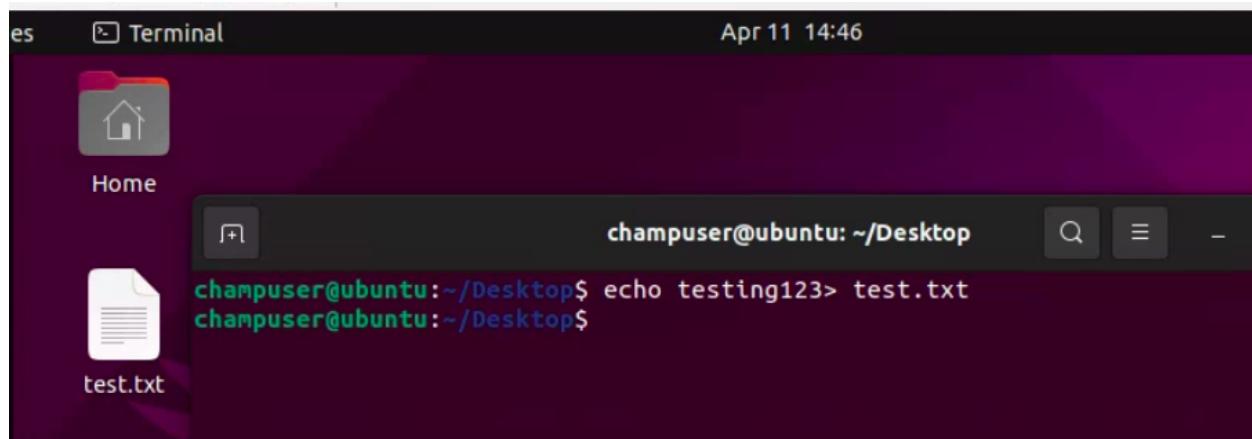
Baseline your VM using the script, adding any other software or configuration you would like to include.

Once powered off, take a checkpoint (aka snapshot)

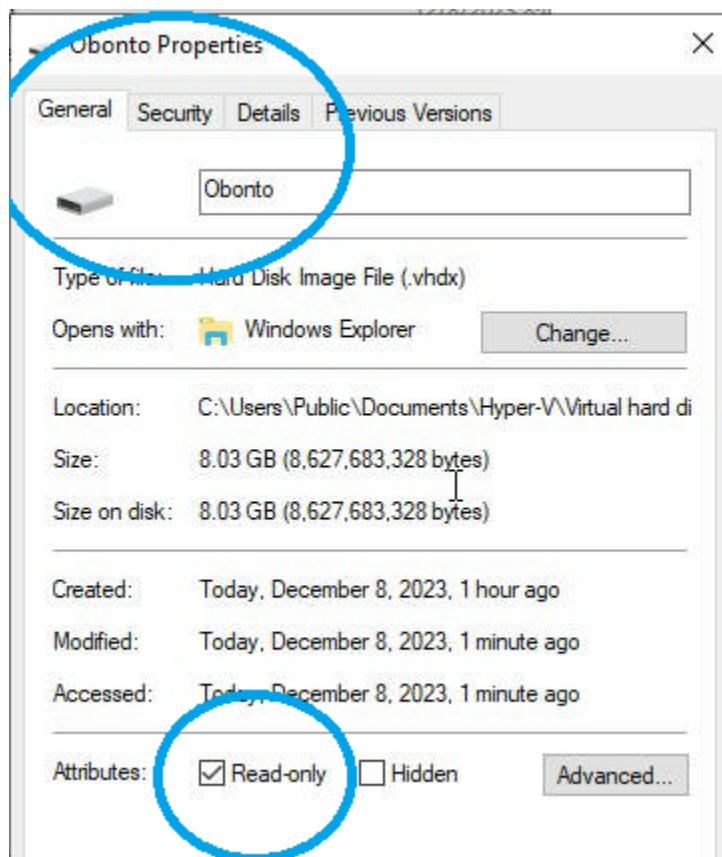


Experiment with this VM, power it on, add a file to the desktop, apply the checkpoint to the running VM or revert to the latest checkpoint, The VM should turn off and when powered on the file should go away just like a VMWare Snapshot.

Updated 11/29/23



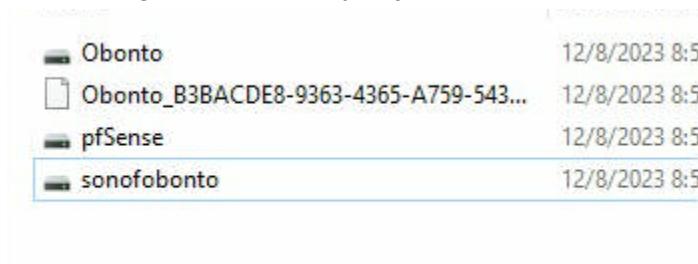
Deliverable 1. Hunt down the VHD file associated with your Ubuntu Base Image and give that file read only permissions. Provide a screenshot similar to the one below.



Updated 11/29/23

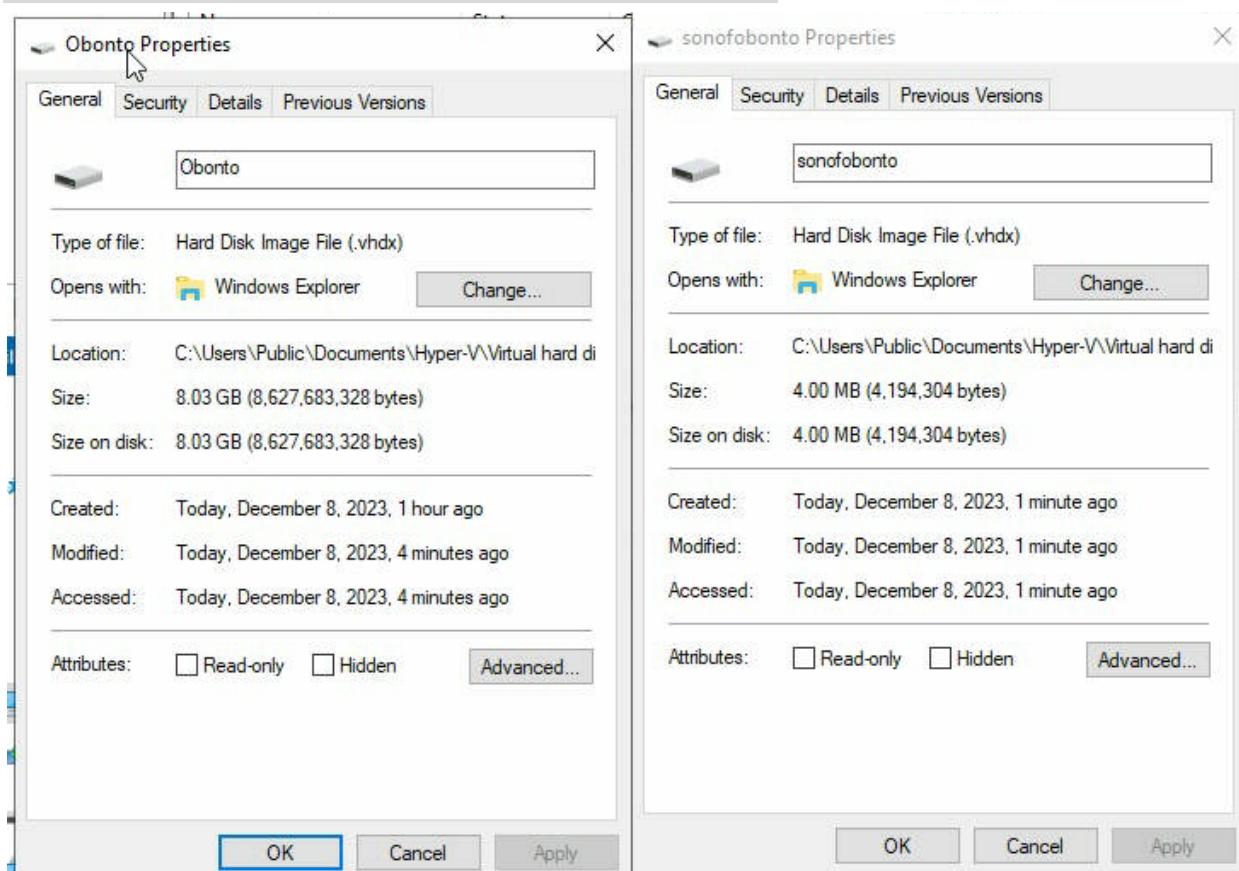
Child Disk

Figure out how to create a new virtual difference disk called sonofubuntu.vhdx with the parent value being the base disk you just created.



Create a new vm called sonofubuntu that uses the difference disk, sonofubuntu.vhdx as an existing disk

Deliverable 2. Provide a screenshot similar to the one below that shows sonofubuntu running as well as the very tiny difference disk attributes similar to the screenshot below.



💡 Now, that was pretty annoying wasn't it? Before you get mad at HyperV, linked clones aren't supported at all from the vSphere GUI, you need to write code to do the same thing with VMWare.

Automation

<https://docs.microsoft.com/en-us/virtualization/hyper-v-on-windows/quick-start/try-hyper-v-power-shell>

Figure out how to interactively use powershell commands to:

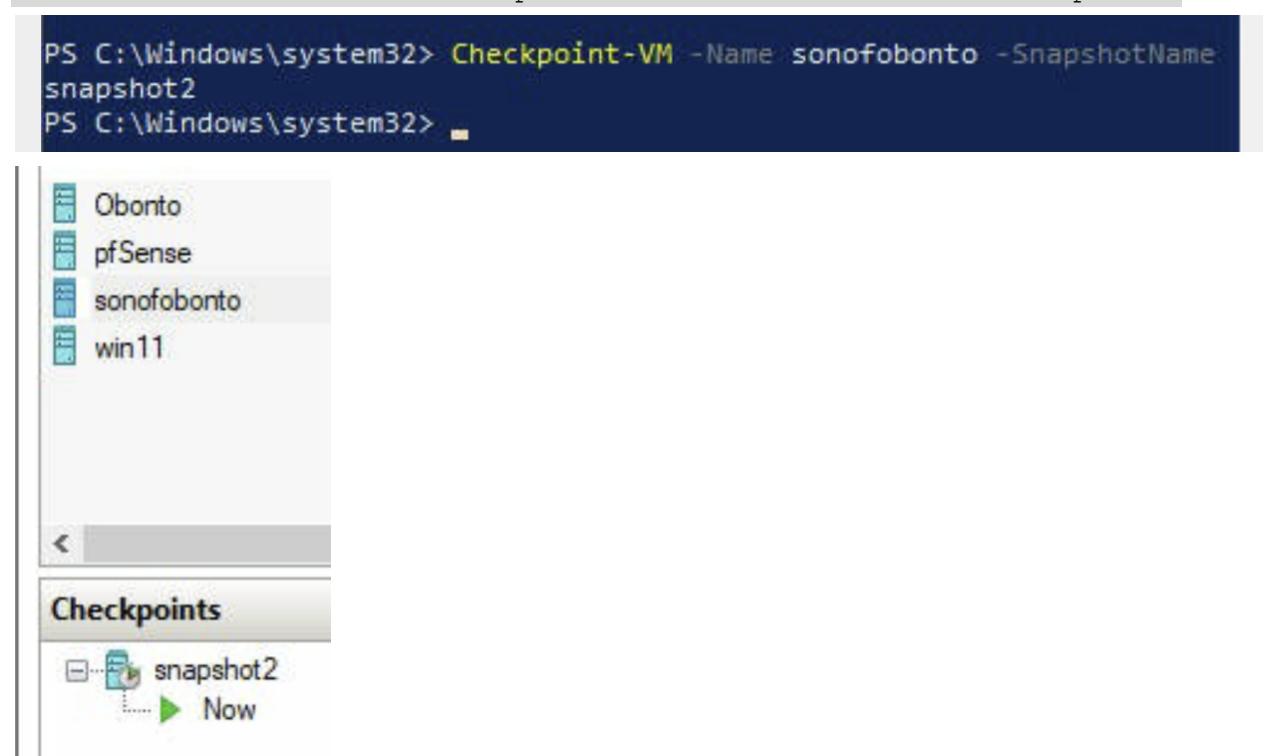
Deliverable 3. stop sonofubuntu

```
PS C:\Windows\system32> Get-VM
Name          State    CPUUsage(%) MemoryAssigned(M) Uptime
----          ----    -----        -----        -----
Obonto        Off      0             0            00:00:00
pfSense       Running  0             2048         7.18:45:35.5410000
sonofobonto   Off      0             0            00:00:00
win11         Off      0             0            00:00:00

PS C:\Windows\system32> Start-VM sonofobonto
PS C:\Windows\system32> Get-VM
Name          State    CPUUsage(%) MemoryAssigned(M) Uptime
----          ----    -----        -----        -----
Obonto        Off      0             0            00:00:00
pfSense       Running  0             2048         7.18:45:55.7710000
sonofobonto   Running  4             4096         00:00:05.6140000
win11         Off      0             0            00:00:00
```

Updated 11/29/23

Deliverable 4. take a checkpoint of sonofubuntu called snapshot1



The screenshot shows the Windows Hyper-V Manager interface. At the top, a command prompt window displays PowerShell commands to take a checkpoint of a virtual machine named 'sonofobonto' and name it 'snapshot2'. Below the command prompt is a list of virtual machines: Obonto, pfSense, sonofobonto, and win11. The 'sonofobonto' entry is highlighted. To the right, a 'Checkpoints' pane is open, showing a tree view with 'snapshot2' expanded. A green arrow points from 'Now' to 'snapshot2', indicating the current state of the VM.

```
PS C:\Windows\system32> Checkpoint-VM -Name sonofobonto -SnapshotName snapshot2
PS C:\Windows\system32>
```

- Obonto
- pfSense
- sonofobonto
- win11

Checkpoints

- snapshot2
 - Now

Updated 11/29/23

Deliverable 5. start sonofubuntu

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

Name	State	CPUUsage(%)	MemoryAssigned(M)	Uptime
Obonto	Off	0	0	00:00:00
pfSense	Running	0	2048	7.18:45:35.5410000
sonofobonto	Off	0	0	00:00:00
win11	Off	0	0	00:00:00

```
PS C:\Windows\system32> Start-VM sonofobonto
```

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

Name	State	CPUUsage(%)	MemoryAssigned(M)	Uptime
Obonto	Off	0	0	00:00:00
pfSense	Running	0	2048	7.18:45:55.7710000
sonofobonto	Running	4	4096	00:00:05.6140000
win11	Off	0	0	00:00:00

```
PS C:\Windows\system32> Checkpoint-VM -Name sonofobonto -SnapshotName  
snapshot2
```

```
PS C:\Windows\system32> Stop-VM sonofobonto
```

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

Name	State	CPUUsage(%)	MemoryAssigned(M)	Uptime
Obonto	Off	0	0	00:00:00
pfSense	Running	0	2048	7.18:47:49.3340000
sonofobonto	Off	0	0	00:00:00
win11	Off	0	0	00:00:00

```
PS C:\Windows\system32>
```

Deliverable 6. switch sonofubuntu to another network

```
PS C:\Windows\system32> Get-VM | Get-VMNetworkAdapter | Connect-VMNetworkAdapter -SwitchName "LAN-INTERNAL"  
PS C:\Windows\system32> Get-VMNetworkAdapter -VMName "sonofobonto"
```

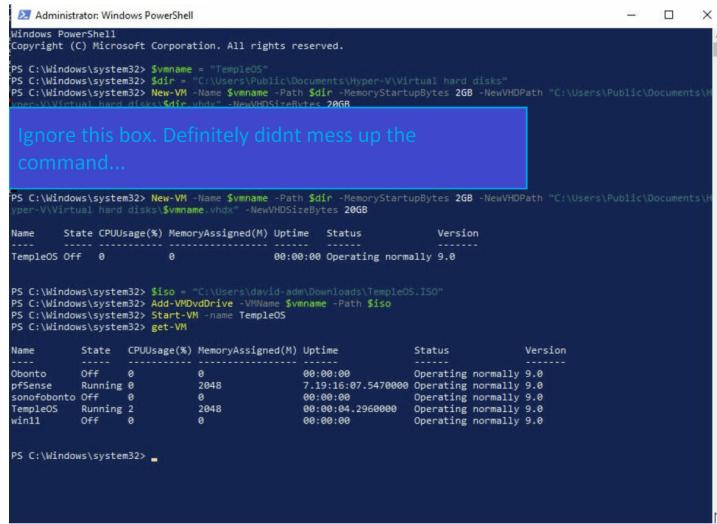
Name	IsManagementOs	VMName	SwitchName	MacAddress	Status	IPAddresses
Network Adapter	False	sonofobonto	LAN-Internal	00155D1A5C05	{}	

```
PS C:\Windows\system32> ■
```

Updated 11/29/23

Provide screenshots of your powershell commands for each task.

Deliverable 7. Create a new Base VM using an OS that is not Ubuntu. Automate the creation of a Linked clone of your new OS base image using Powershell. Provide a screenshot of your successful command(s) and a screenshot of your running OS and the virtual properties of your child disk.



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

PS C:\Windows\system32> $vmname = "TempleOS"
PS C:\Windows\system32> $dir = "C:\Users\Public\Documents\Hyper-V\Virtual hard disks"
PS C:\Windows\system32> New-VM -Name $vmname -Path $dir -MemoryStartupBytes 2GB -NewVHDPath "C:\Users\Public\Documents\Hyper-V\Virtual hard disks\$vmname.vhdx" -NewVHDSizeBytes 20GB
Ignore this box. Definitely didnt mess up the command...
PS C:\Windows\system32> New-VM -Name $vmname -Path $dir -MemoryStartupBytes 2GB -NewVHDPath "C:\Users\Public\Documents\Hyper-V\Virtual hard disks\$vmname.vhdx" -NewVHDSizeBytes 20GB
Name      State   CPUUsage(%) MemoryAssigned(M) Uptime          Status      Version
----      ----   -----        -----        -----          -----
TempleOS  Off       0           0     00:00:00  Operating normally 9.0

PS C:\Windows\system32> $iso = "C:\Users\david-admin\Downloads\TempleOS.ISO"
PS C:\Windows\system32> Add-VDVDdrive -VName $vmname -Path $iso
PS C:\Windows\system32> Start-VM -Name TempleOS
PS C:\Windows\system32> Get-VM
Name      State   CPUUsage(%) MemoryAssigned(M) Uptime          Status      Version
----      ----   -----        -----        -----          -----
Ubuntu   Off       0           0     00:00:00  Operating normally 9.0
pfSense   Running  0           2048    7.19:16.07.5470000  Operating normally 9.0
sonofobonto Off       0           0     00:00:00  Operating normally 9.0
TempleOS  Running  2           2048    00:00:04.2960000  Operating normally 9.0
win11    Off       0           0     00:00:00  Operating normally 9.0

PS C:\Windows\system32>
```

Deliverable 8. Journal, Journal, Journal. Make sure any powershell 1 liners and ps1 files are saved. Compare and contrast your experience with vCenter and adhoc pyvmomi commands with your experiences on this milestone. Provide a link to your journal entry.