dataMinds Connect 2021

Viorel Ciucu
@viorelciucu

How to setup your testing environment, the DevOps way

Our Partners

KOHERA

element 61
experience & expertise

Microsoft

cegeka
IN CLOSE COOPERATION

inetum.
realdolmen
Positive digital flow

Sparkle

arinti

Business & Decision

delaware
we commit. we deliver.

LACO KNOWLEDGE, AND MORE.

Lytix

MONIN
Database Managed Services

KPMG

U2U

ae Adapt and Enable

redgate

MYRMIDONS

What do you think?

1. Open the form
2. Provide constructive feedback
3. Be eligible for an amazing prize!

bit.ly is CASE SENSITIVE!

http://bit.ly/dMC2021_FeedbackTuesday

Get-Identity

Consultant, Automator, Database Administrator, SRE

VCI Consulting
vciconsulting.net
linkedin.com/in/cviorel
twitter.com/viorelciucu

AGENDA

DevOps way of deploying infrastructure – SQL Server Availability group

Use cases:

- Test Labs
- Demos
- New real-world environments (DEV/QA/STAGE/PROD)

WHY?

- Consistency across environments
- Continuously improve
- Compliance
- Automation ensures processes are followed

Get-Prerequisite

- PowerShell
- Active Directory
- VM Templates

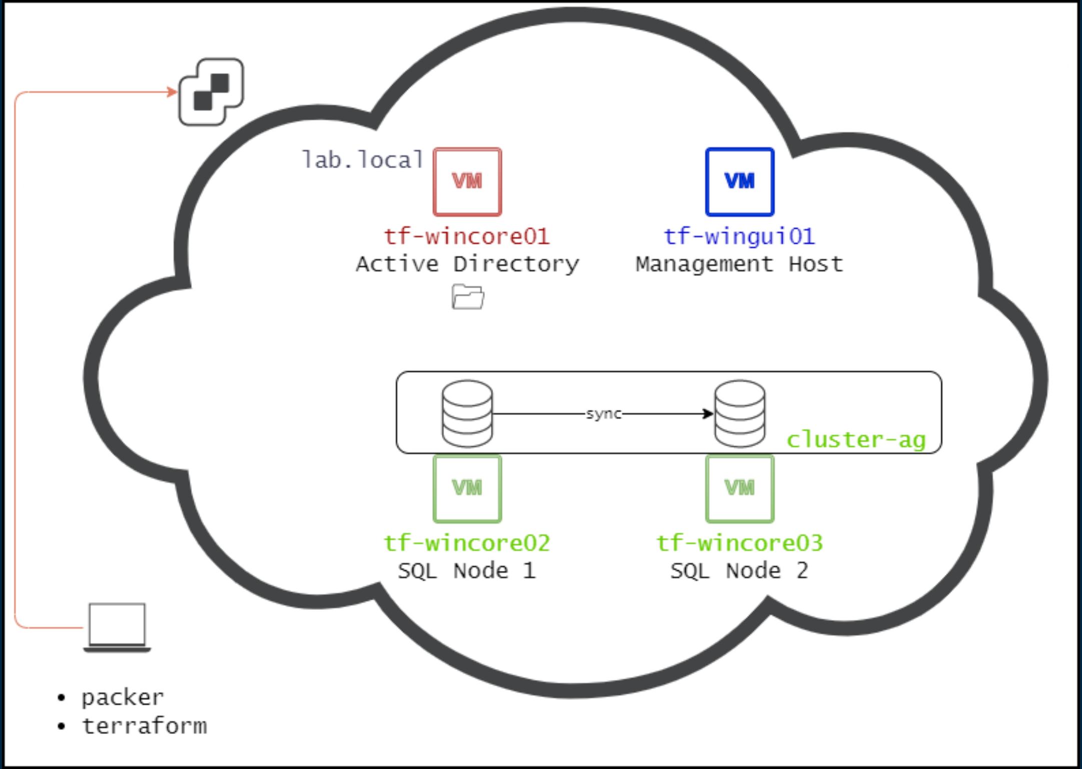
Get-Resources

https://github.com/cviorel/Presentations/tree/main/2021/dataMinds

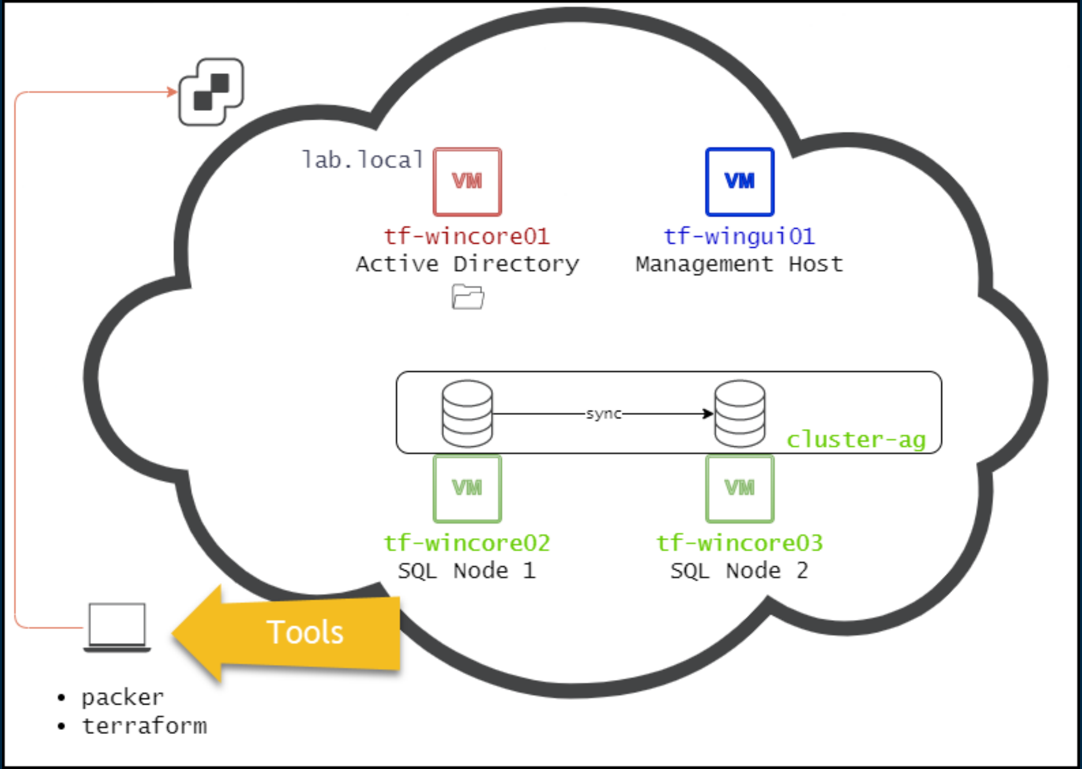
Get-Tools

- Packer - https://www.packer.io
- Terraform - https://www.terraform.io
- DBATools - https://dbatools.io

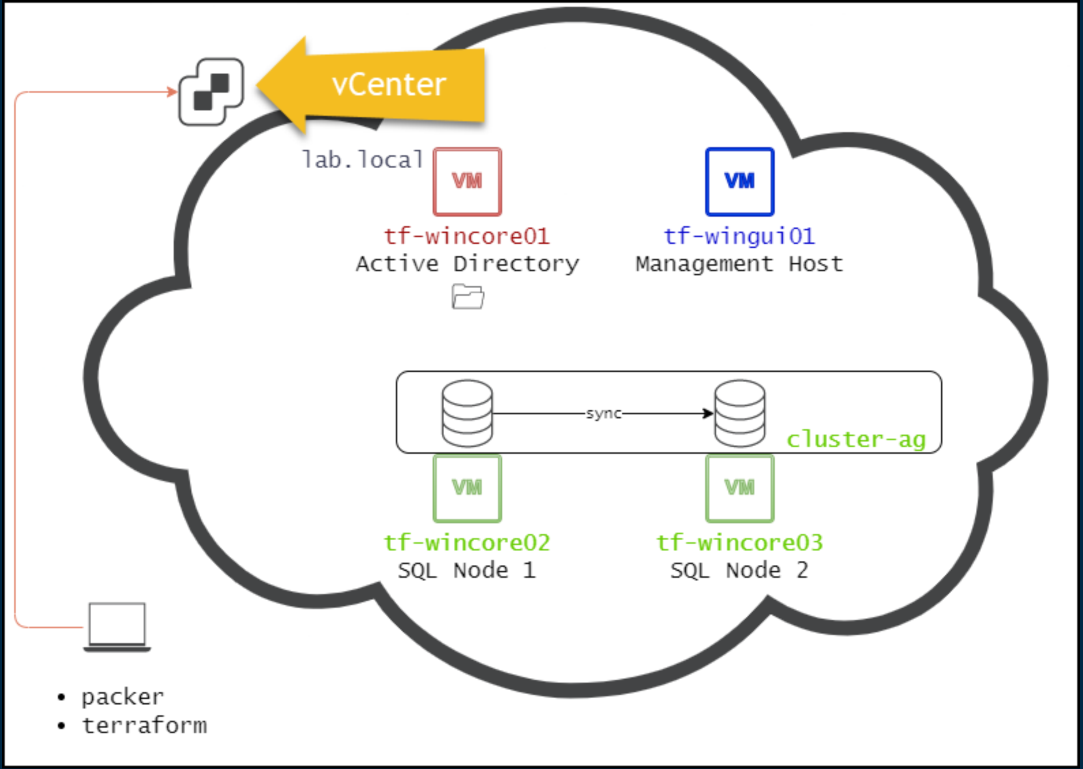
New-Infra


lab.local
VM
tf-wincore01
Active Directory
VM
tf-wingui01
Management Host
sync
cluster-ag
VM
tf-wincore02
SQL Node 1
VM
tf-wincore03
SQL Node 2
packer
terraform

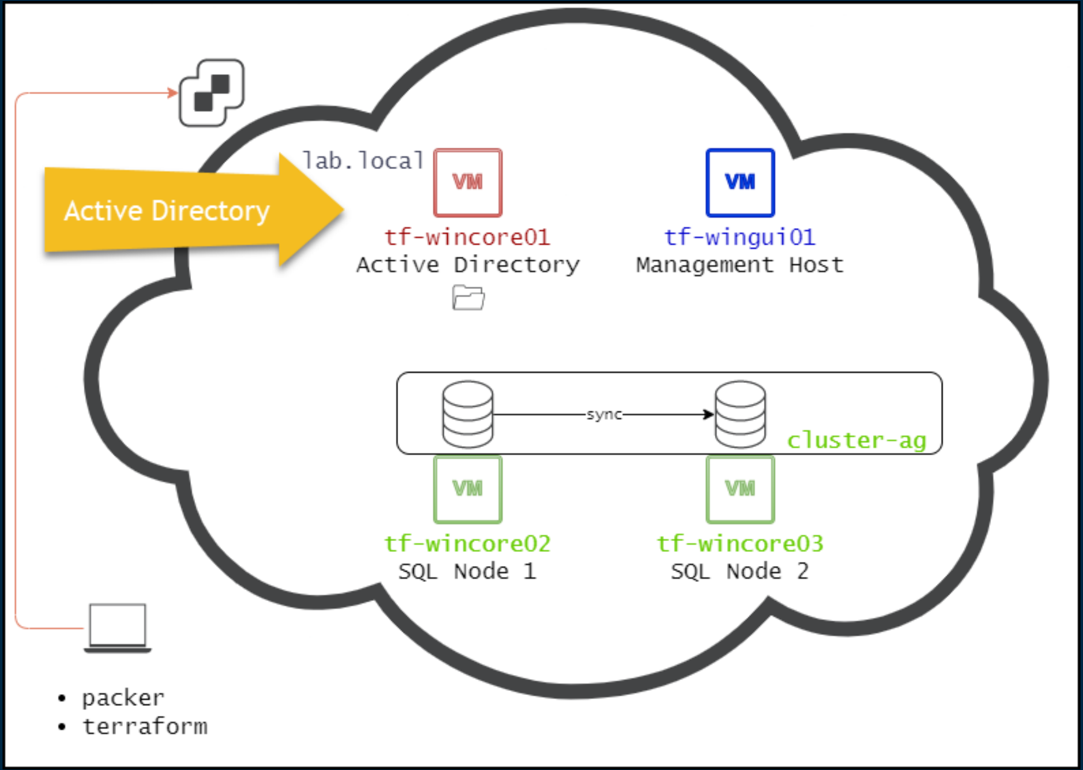
New-Infra


lab.local
VM
tf-wincore01
Active Directory
VM
tf-wingui01
Management Host
sync
cluster-ag
VM
tf-wincore02
SQL Node 1
VM
tf-wincore03
SQL Node 2
Tools
• packer
• terraform

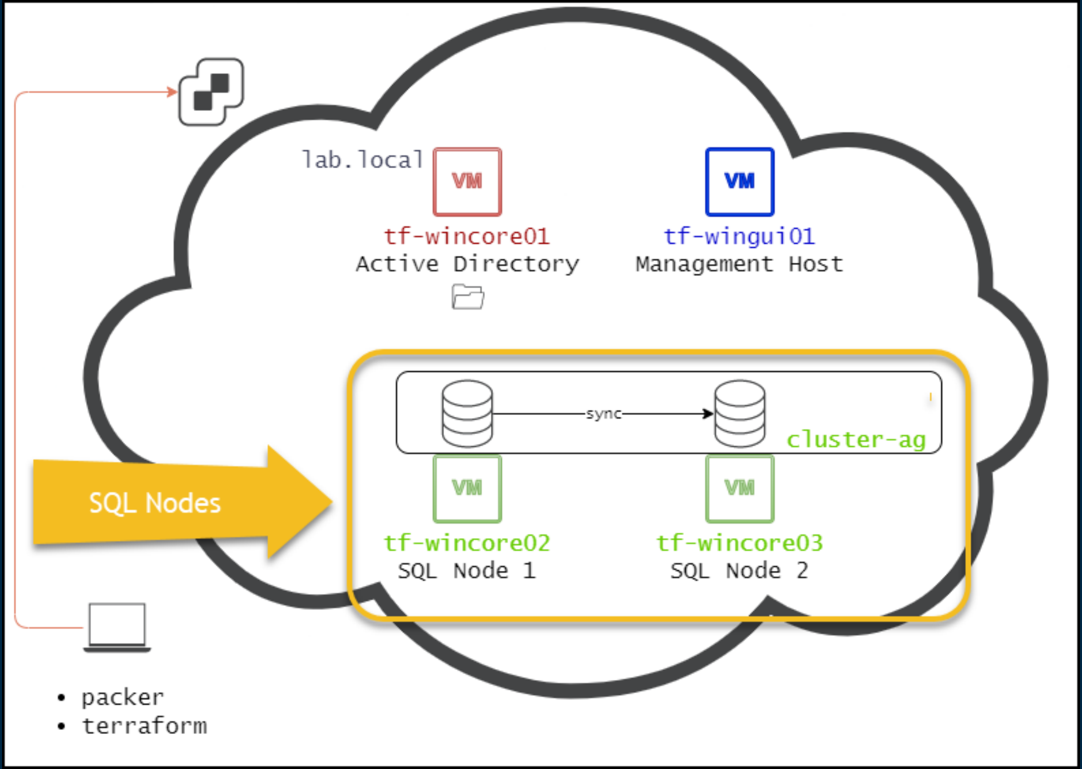
New-Infra


vCenter
lab.local
VM
tf-wincore01
Active Directory
VM
tf-wingui01
Management Host
sync
cluster-ag
VM
tf-wincore02
SQL Node 1
VM
tf-wincore03
SQL Node 2
• packer
• terraform

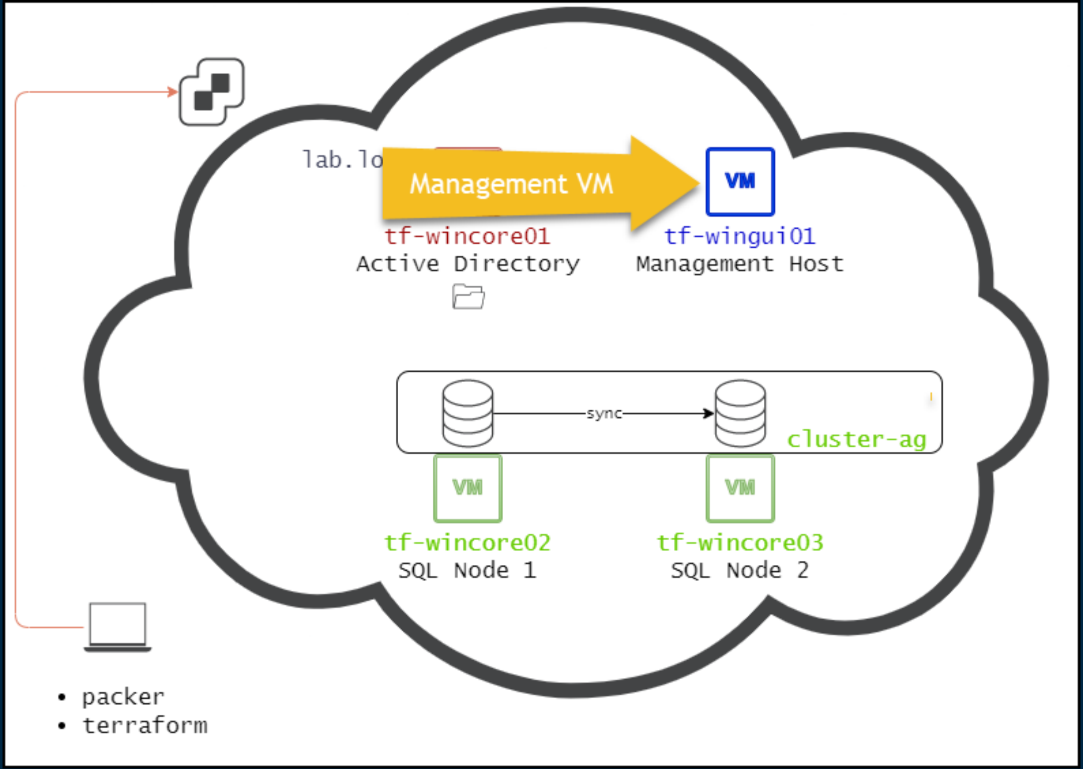
New-Infra


Active Directory
lab.local
VM
tf-wincore01
Active Directory
VM
tf-wingui01
Management Host
sync
cluster-ag
VM
tf-wincore02
SQL Node 1
VM
tf-wincore03
SQL Node 2
• packer
• terraform

New-Infra


lab.local
VM
tf-wincore01
Active Directory
VM
tf-wingui01
Management Host
sync
cluster-ag
SQL Nodes
VM
tf-wincore02
SQL Node 1
VM
tf-wincore03
SQL Node 2
• packer
• terraform

New-Infra



- packer
- terraform

Packer

I need templates!

Automate it!

Packer

Packer automates the creation of machine images

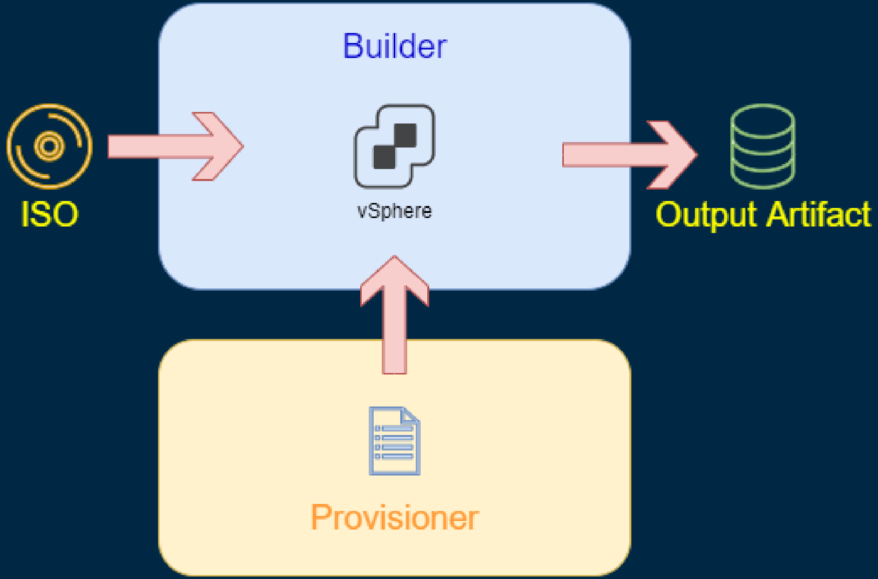
Builders

Builders are responsible for creating machines and generating images from them for various platforms.

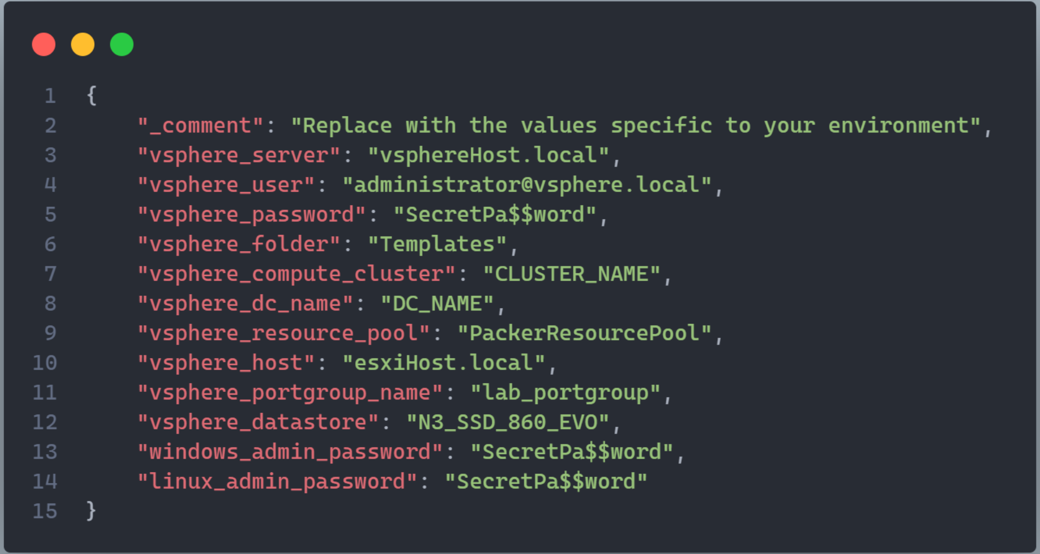
Separate builders for EC2, VMware, VirtualBox, Azure, etc.

Packer

```
packer build -force -var-file='json\vars.json' 'json\win2019core.json'
```



Variables



```json
{
    "_comment": "Replace with the values specific to your environment",
    "vsphere_server": "vsphereHost.local",
    "vsphere_user": "administrator@vsphere.local",
    "vsphere_password": "SecretPa$$word",
    "vsphere_folder": "Templates",
    "vsphere_compute_cluster": "CLUSTER_NAME",
    "vsphere_dc_name": "DC_NAME",
    "vsphere_resource_pool": "PackerResourcePool",
    "vsphere_host": "esxiHost.local",
    "vsphere_portgroup_name": "lab_portgroup",
    "vsphere_datastore": "N3_SSD_860_EVO",
    "windows_admin_password": "SecretPa$$word",
    "linux_admin_password": "SecretPa$$word"
}
```

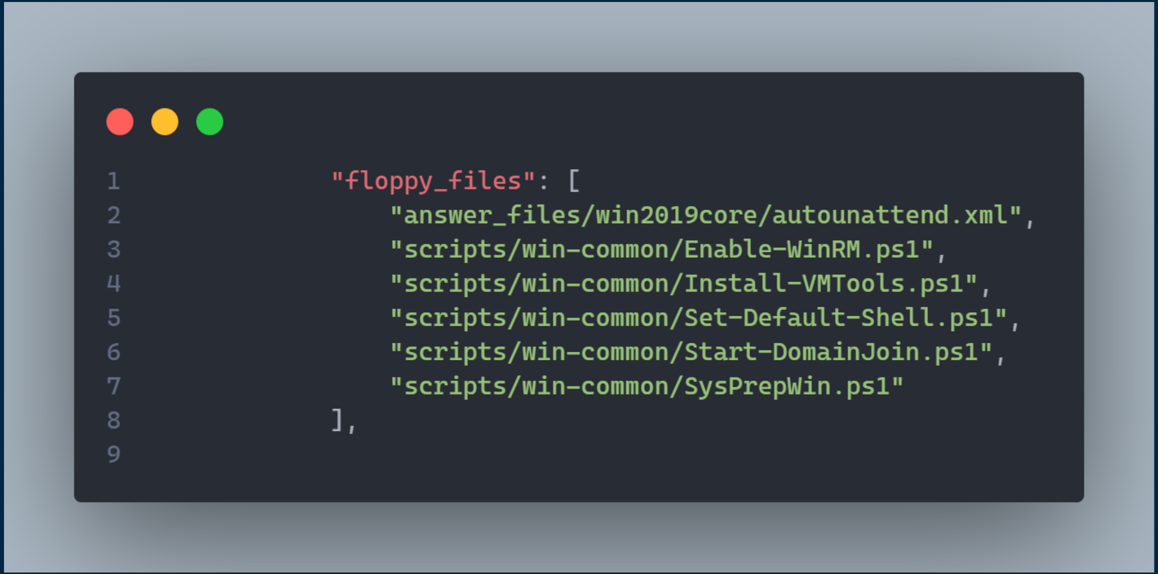
Builders

```
"builders": [
    {
        "type": "vsphere-iso",
        "username": "{{user `vsphere_user`}}",
        "vcenter_server": "{{user `vsphere_server`}}",

```

```
"iso_paths": [
    "{{user `os_iso_path`}}",
    "{{user `vmtools_iso_path`}}"
],
```

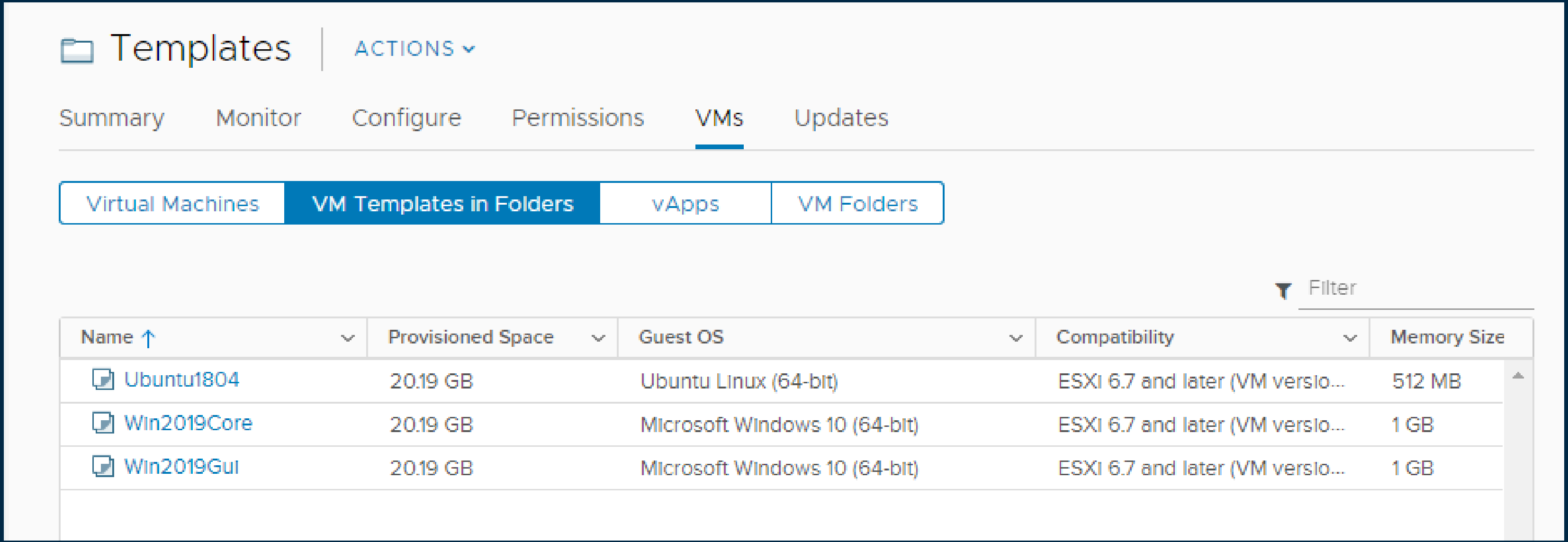
```
"network_adapters": [
    {
        "network_card": "vmxnet3"
    }
],
```

Packer


```
        "floppy_files": [
            "answer_files/win2019core/autounattend.xml",
            "scripts/win-common/Enable-WinRM.ps1",
            "scripts/win-common/Install-VMTools.ps1",
            "scripts/win-common/Set-Default-Shell.ps1",
            "scripts/win-common/Start-DomainJoin.ps1",
            "scripts/win-common/SysPrepWin.ps1"
        ],
```

PACKER DEMO

Packer



Name ↑	Provisioned Space	Guest OS	Compatibility	Memory Size
Ubuntu1804	20.19 GB	Ubuntu Linux (64-bit)	ESXi 6.7 and later (VM versio...	512 MB
Win2019Core	20.19 GB	Microsoft Windows 10 (64-bit)	ESXi 6.7 and later (VM versio...	1 GB
Win2019Gui	20.19 GB	Microsoft Windows 10 (64-bit)	ESXi 6.7 and later (VM versio...	1 GB

Terraform

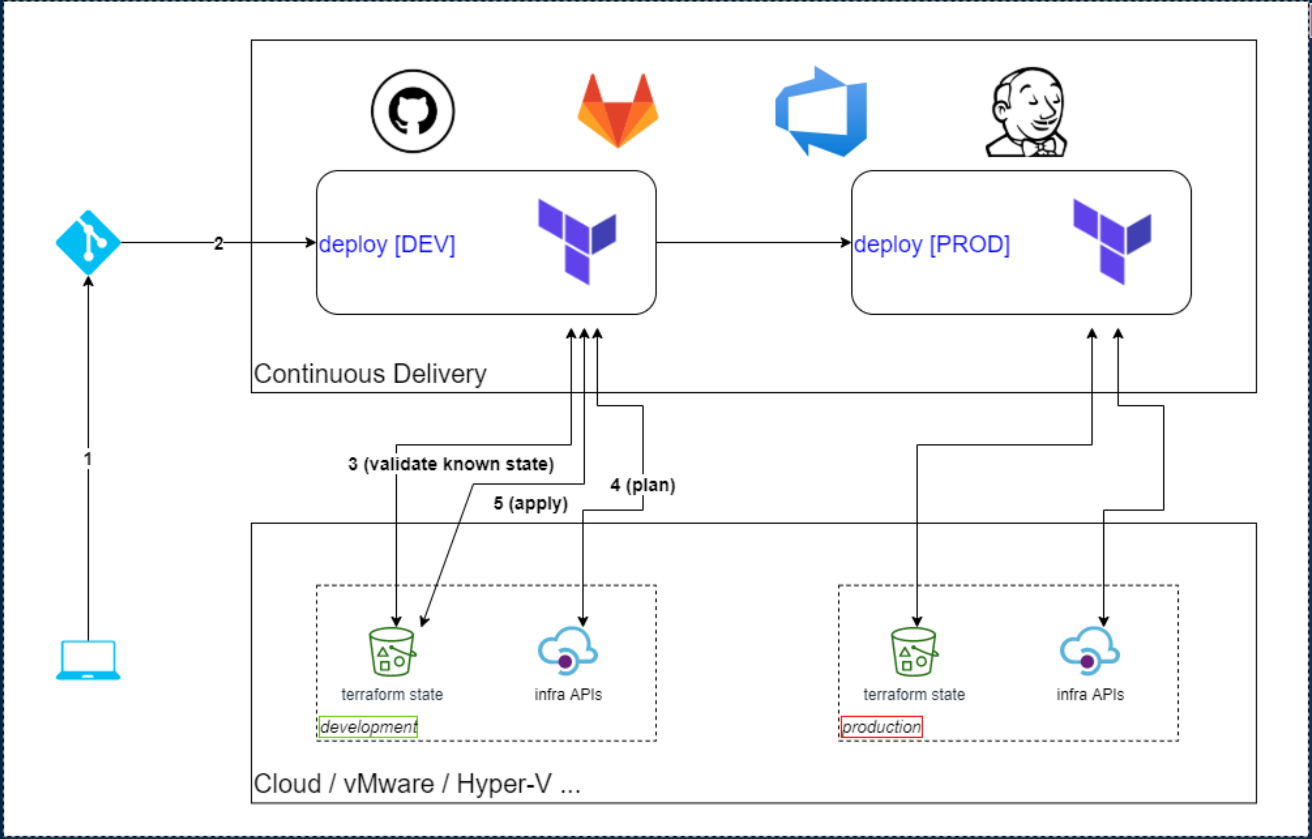
Deploy VMs from the templates

Automate it!

Terraform

- build, change and version infrastructure safely and efficiently – Infrastructure as Code (IaC)
- provision entire infrastructures that span across multiple public and private cloud providers
- manages external resources (network appliances, software as a service, platform as a service, etc.)

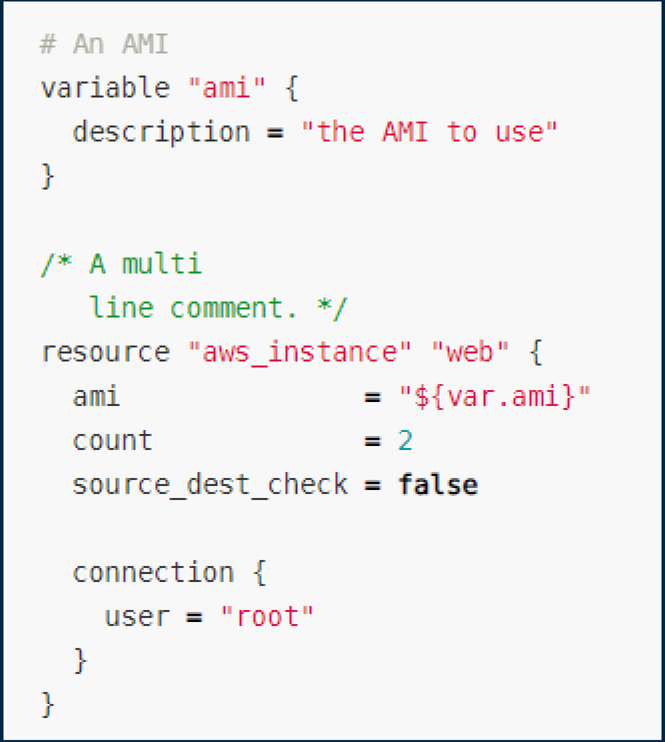
Terraform


deploy [DEV]
deploy [PROD]
Continuous Delivery
1
2
3 (validate known state)
4 (plan)
5 (apply)
terraform state
infra APIs
development
terraform state
infra APIs
production
Cloud / vMware / Hyper-V ...

Terraform

The syntax of Terraform configurations is called HashiCorp Configuration Language (HCL)

Terraform

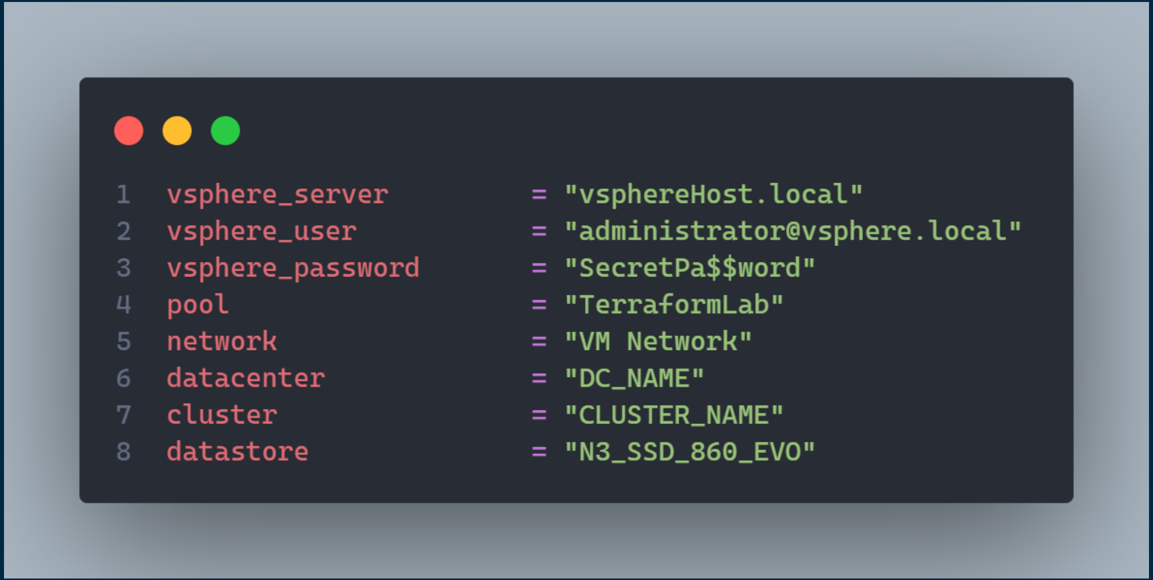


```
# An AMI
variable "ami" {
  description = "the AMI to use"
}

/* A multi
   line comment. */
resource "aws_instance" "web" {
  ami               = "${var.ami}"
  count             = 2
  source_dest_check = false

  connection {
    user = "root"
  }
}
```

Terraform

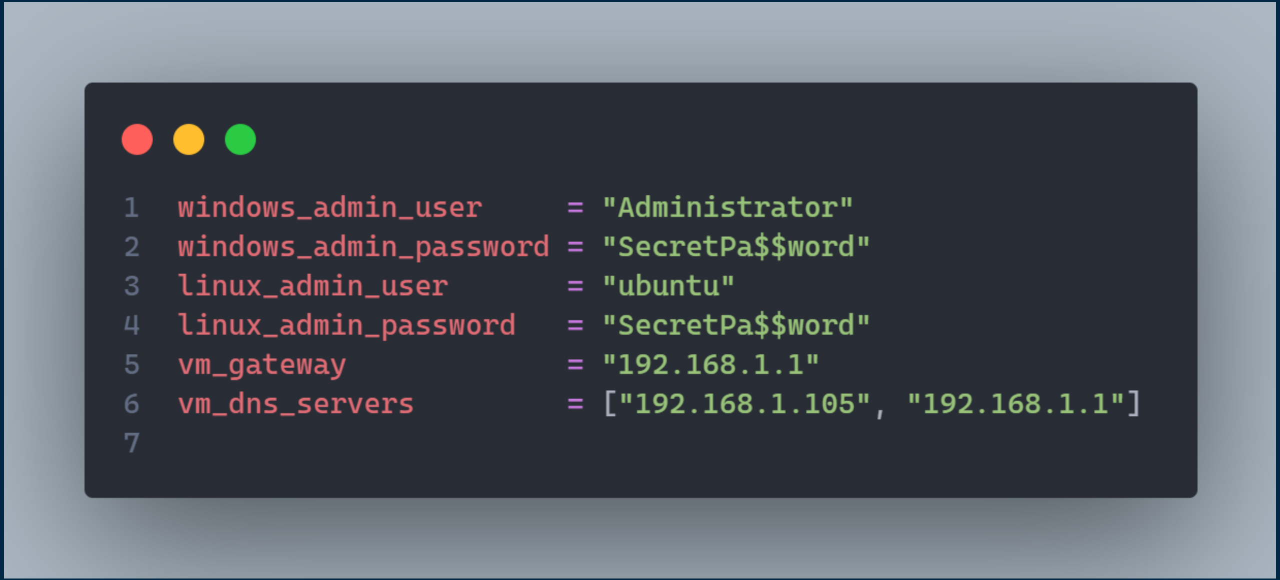
Terraform relies on plugins called "providers“
to interact with remote systems

Terraform



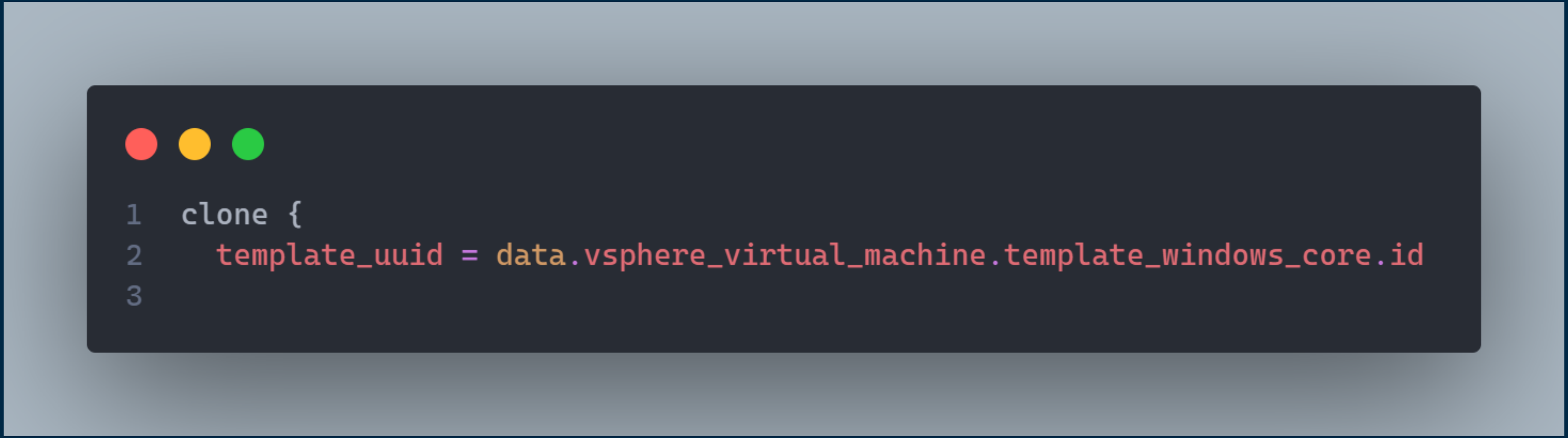
```
vsphere_server           = "vsphereHost.local"
vsphere_user             = "administrator@vsphere.local"
vsphere_password         = "SecretPa$$word"
pool                     = "TerraformLab"
network                  = "VM Network"
datacenter               = "DC_NAME"
cluster                  = "CLUSTER_NAME"
datastore                = "N3_SSD_860_EVO"
```

Terraform



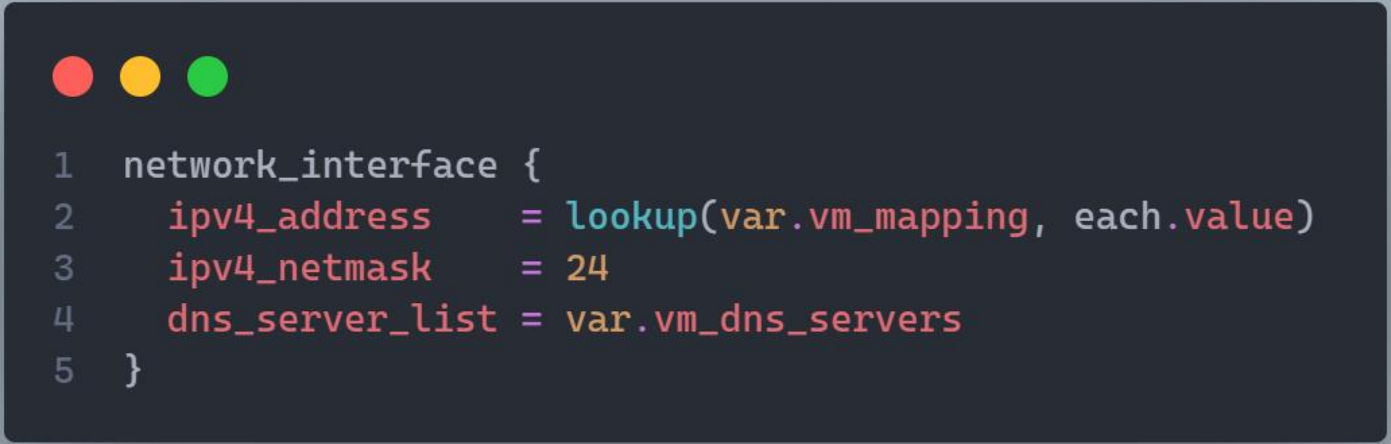
```
windows_admin_user     = "Administrator"
windows_admin_password = "SecretPa$$word"
linux_admin_user       = "ubuntu"
linux_admin_password   = "SecretPa$$word"
vm_gateway             = "192.168.1.1"
vm_dns_servers         = ["192.168.1.105", "192.168.1.1"]

```

Terraform



```
clone {
  template_uuid = data.vsphere_virtual_machine.template_windows_core.id

```

Terraform



```
network_interface {
  ipv4_address    = lookup(var.vm_mapping, each.value)
  ipv4_netmask    = 24
  dns_server_list = var.vm_dns_servers
}
```

Get-Code

Let’s see the code!

Terraform

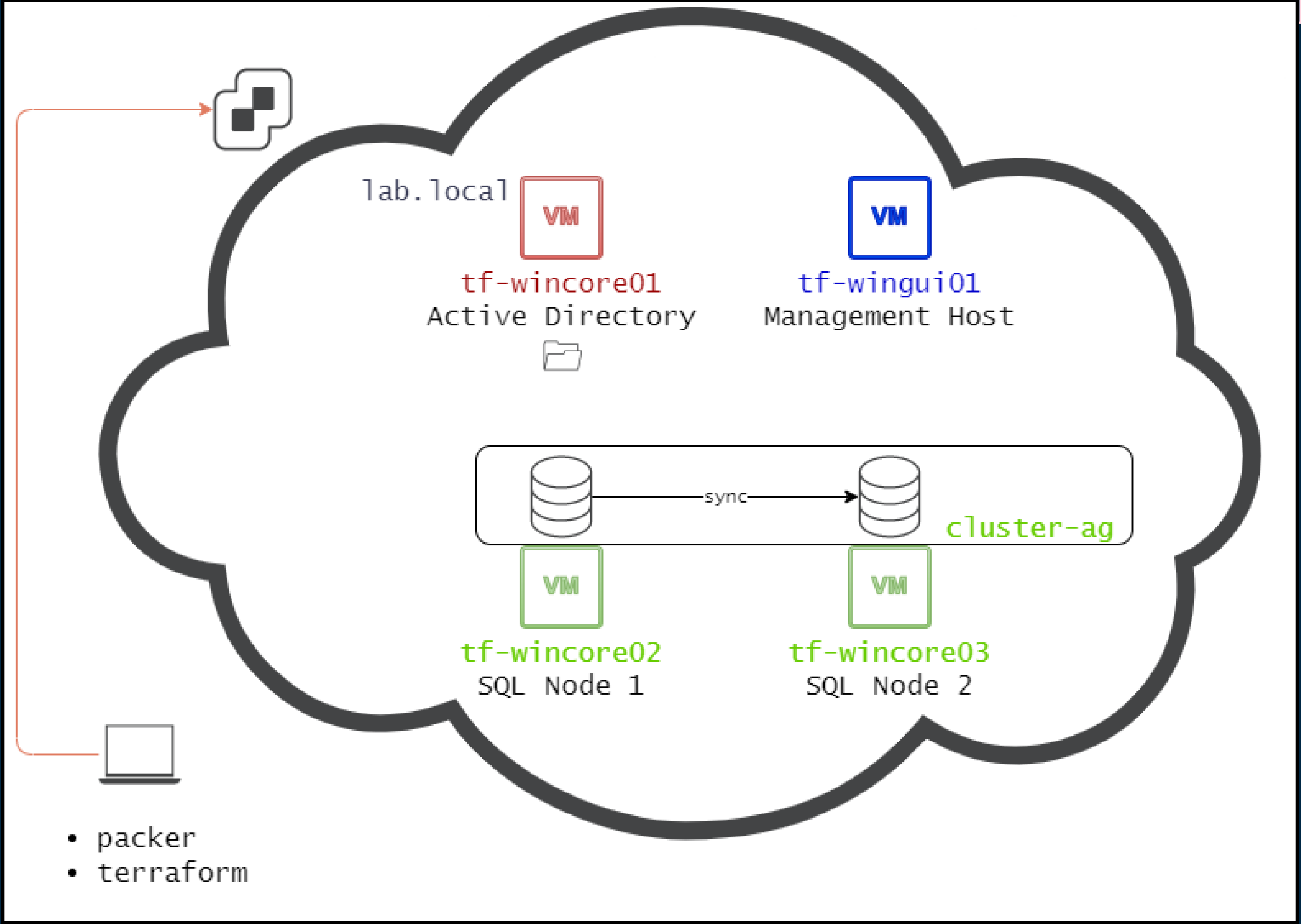
terraform plan - creates an execution plan

Terraform

terraform apply - applies the changes required to reach the desired state

TERRAFORM
DEMO

Deploy the LAB


lab.local
VM
tf-wincore01
Active Directory
VM
tf-wingui01
Management Host
sync
cluster-ag
VM
tf-wincore02
SQL Node 1
VM
tf-wincore03
SQL Node 2
• packer
• terraform

Start-Build

- Build an Active Directory
- Join the computers to the AD

Start-Build

- Create gMSAs
- Install SQL instances
- Configure SQL instances

Start-Build

- Create WFC
- Create an Availability Group

Set-Variables

00_set-variables.ps1

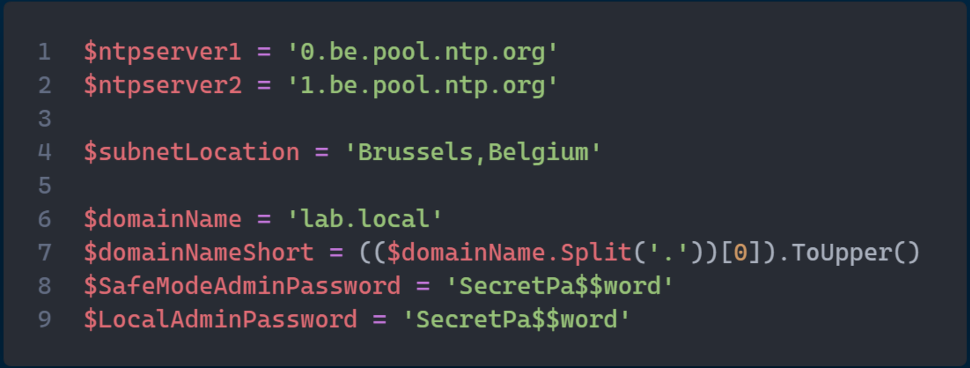
Set-Variables

dot sourcing 00_set-variables.ps1

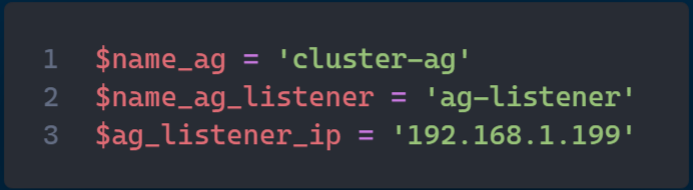
```
. .\00_set-variables.ps1
```

Variables

- Domain controller – name and IP
- SQL Nodes – name and IP
- NTP Servers
- Cluster Name & IP
- SQL Instance Name and TCP Port
- Collation
- SQL Folder Names
- AG Name
- AG Listener Name & IP



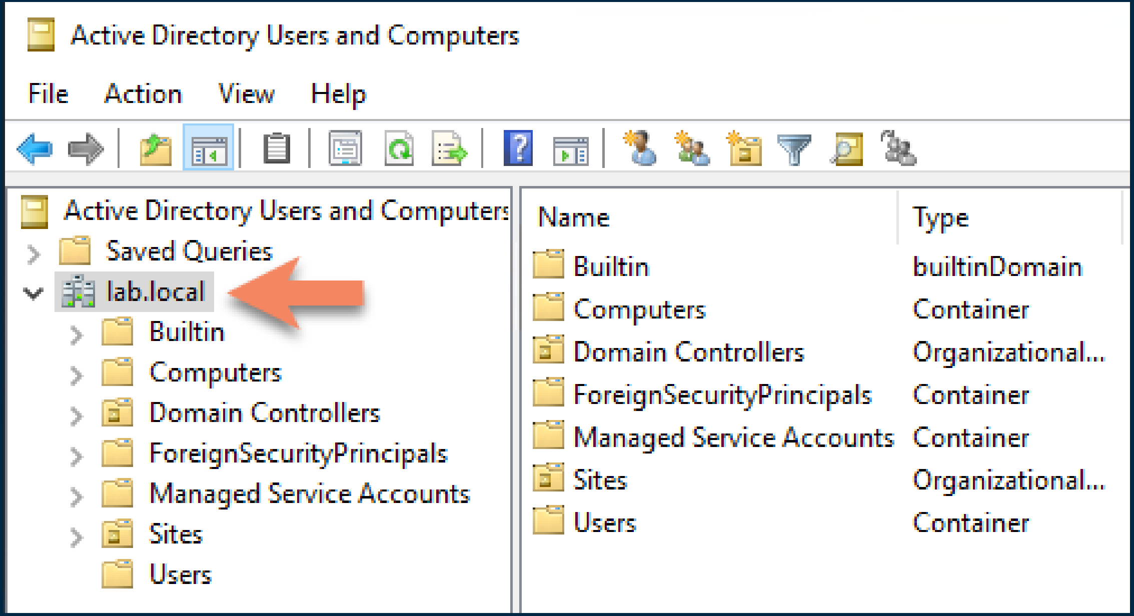
```
$ntpserver1 = '0.be.pool.ntp.org'
$ntpserver2 = '1.be.pool.ntp.org'

$subnetLocation = 'Brussels,Belgium'

$domainName = 'lab.local'
$domainNameShort = (($domainName.Split('.'))[0]).ToUpper()
$SafeModeAdminPassword = 'SecretPa$$word'
$LocalAdminPassword = 'SecretPa$$word'
```



```
$name_ag = 'cluster-ag'
$name_ag_listener = 'ag-listener'
$ag_listener_ip = '192.168.1.199'
```

Setup AD

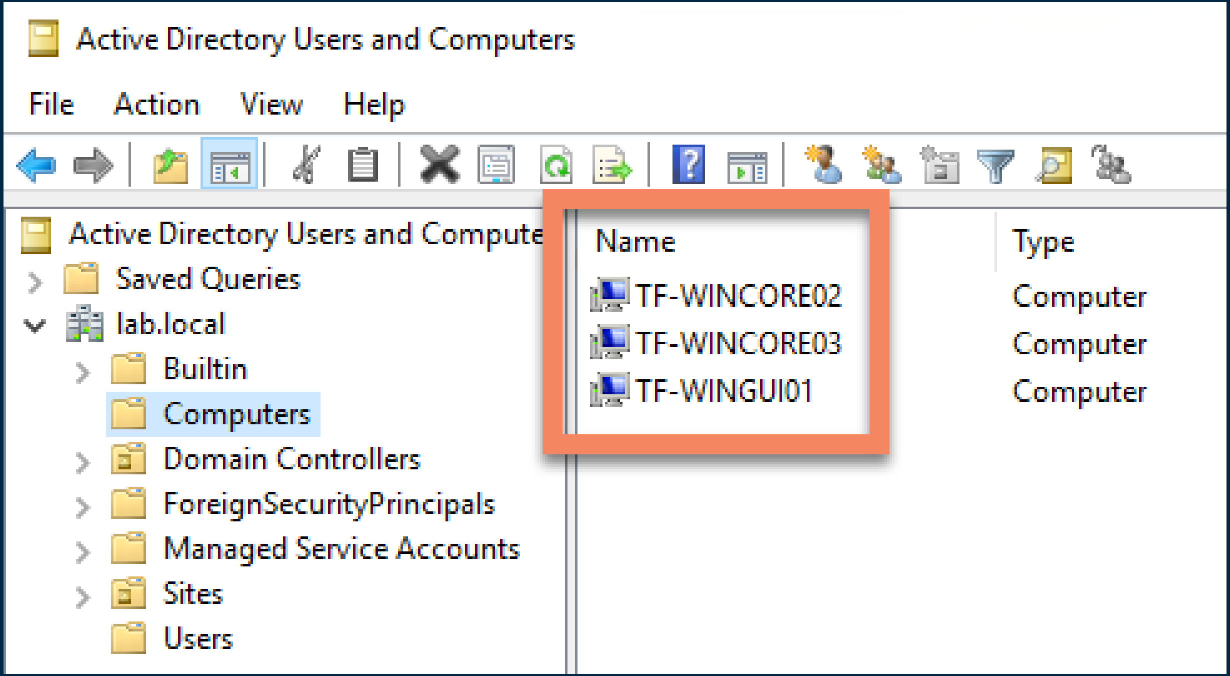
- Create forest
- Add DNS Zones
- Configure DNS Scavenging
- Configure NTP
- KdsRootKey


Active Directory Users and Computers

Active Directory Users and Computers
- Saved Queries
- lab.local
  - Builtin
  - Computers
  - Domain Controllers
  - ForeignSecurityPrincipals
  - Managed Service Accounts
  - Sites
  - Users

Name	Type
Builtin	builtinDomain
Computers	Container
Domain Controllers	Organizational...
ForeignSecurityPrincipals	Container
Managed Service Accounts	Container
Sites	Organizational...
Users	Container

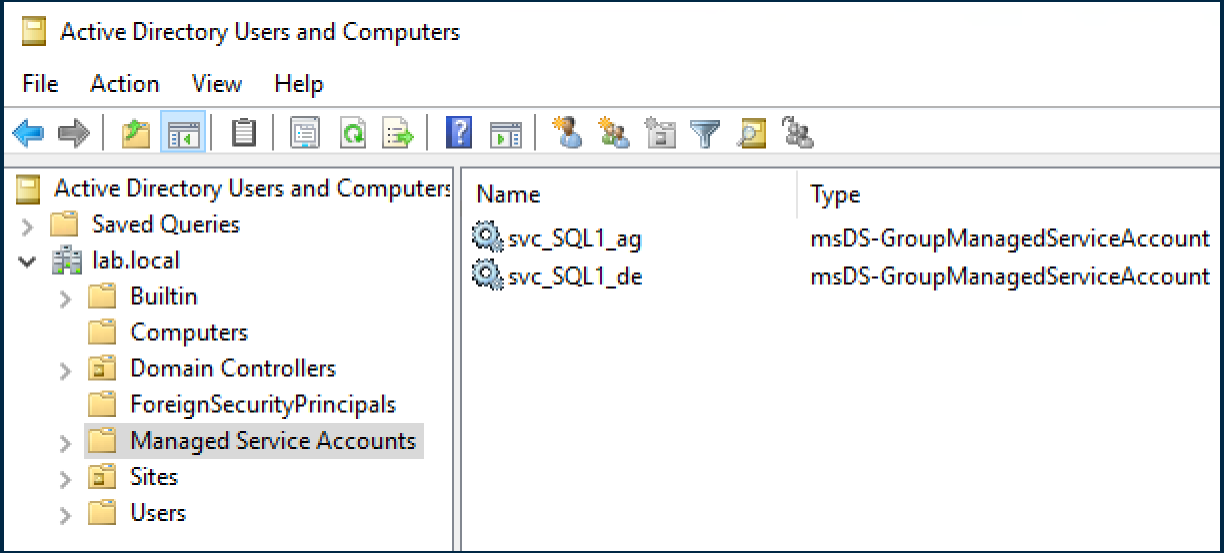
Domain join



Active Directory Users and Computers

Name	Type
TF-WINCORE02	Computer
TF-WINCORE03	Computer
TF-WINGUI01	Computer

gMSA



Active Directory Users and Computers
- Saved Queries
- lab.local
  - Builtin
  - Computers
  - Domain Controllers
  - ForeignSecurityPrincipals
  - Managed Service Accounts
  - Sites
  - Users

Name	Type
svc_SQL1_ag	msDS-GroupManagedServiceAccount
svc_SQL1_de	msDS-GroupManagedServiceAccount

Start-Configuration

Configuration management tools:

- Puppet
- Ansible
- Chef

dbatools

dbatools

- PowerShell module with over 500 SQL Server administration, best practice and migration commands included – dbatools.io

Get-Tasks

- ✓ Install SQL Server on the 2 nodes
- ✓ Configure SQL Server instances
- ✓ Perform hardening (NTFS permissions, revoke connect to guest, etc.)
- ✓ Install Ola Hallengren's Maintenance Solution
- ✓ Install WhoIsActive
- ✓ Create SQL Agent Jobs and Schedules
- ✓ Create a new TestDB database
- ✓ Create an Availability Group
- ✓ Create an Availability Group Listener
- ✓ Verify SPNs

- ✓ Install SQL Server on the 2 nodes
- ✓ Configure SQL Server instances
- ✓ Perform hardening (NTFS permissions, revoke connect to guest, etc.)
- ✓ Install Ola Hallengren’s Maintenance Solution
- ✓ Install WhoIsActive
- ✓ Create SQL Agent Jobs and Schedules
- ✓ Create a new TestDB database
- ✓ Create an Availability Group
- ✓ Create an Availability Group Listener
- ✓ Verify SPNs

Install SQL Server

```
$config = @{
    SqlInstance                 = $sqlNodes.Keys
    Version                     = 2019
    InstanceName                = "$SQLInstanceName"
    Port                        = "$SQLInstancePort"

```

Install SQL Server

```
SqlInstance                          = $sqlNodes.Keys
```

```
$sqlNodes = @{
      'tf-wincore02' = "192.168.1.82"
      'tf-wincore03' = "192.168.1.83"
}
```

Install SQL Server

```
        SQLCOLLATION          = $sqlCollation
        AGTSVCACCOUNT         = "${domainNameShort}\${AgentAccountName}`$"
        SQLSVCACCOUNT         = "${domainNameShort}\${EngineAccountName}`$"
        BROWSERSVCSTARTUPTYPE = "Disabled"
        SQLTELSVCSTARTUPTYPE  = "Disabled"

```

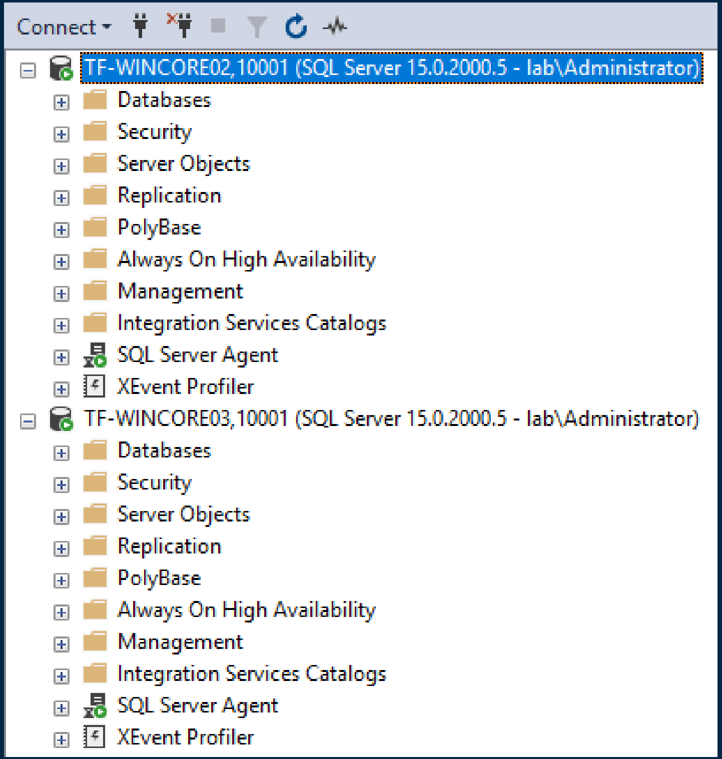
Install SQL Server

```
Feature                              = "Engine"
SaCredential                         = $saCredential
AuthenticationMode                   = "Mixed"
```

Install SQL Server

```powershell
Install-DbaInstance @config
```

Install SQL Server



Connect

TF-WINCORE02,10001 (SQL Server 15.0.2000.5 - lab\Administrator)
- Databases
- Security
- Server Objects
- Replication
- PolyBase
- Always On High Availability
- Management
- Integration Services Catalogs
- SQL Server Agent
- XEvent Profiler

TF-WINCORE03,10001 (SQL Server 15.0.2000.5 - lab\Administrator)
- Databases
- Security
- Server Objects
- Replication
- PolyBase
- Always On High Availability
- Management
- Integration Services Catalogs
- SQL Server Agent
- XEvent Profiler

- ✓ ~~Install SQL Server on the 2 nodes~~
- ✓ **Configure SQL Server instances**
- ✓ Perform hardening (NTFS permissions, revoke connect to guest, etc.)
- ✓ Install Ola Hallengren’s Maintenance Solution
- ✓ Install WhoIsActive
- ✓ Create SQL Agent Jobs and Schedules
- ✓ Create a new TestDB database
- ✓ Create an Availability Group
- ✓ Create an Availability Group Listener
- ✓ Verify SPNs

SQL Instance Configuration

- sp_configure
- Max Memory
- Add 3226 TF

SQL Instance Configuration

- Rename and disable SA
- Configure model db
- Create a utility database – DBA

SQL Instance Configuration

```
Set-DbaSpConfigure -SqlInstance $sqlInstance -ConfigName CostThresholdForParallelism -Value 50
Set-DbaSpConfigure -SqlInstance $sqlInstance -ConfigName DefaultBackupCompression -Value 1
Set-DbaSpConfigure -SqlInstance $sqlInstance -ConfigName OptimizeAdhocWorkloads -Value 1
Set-DbaSpConfigure -SqlInstance $sqlInstance -ConfigName RemoteDacConnectionsEnabled -Value 1
Set-DbaSpConfigure -SqlInstance $sqlInstance -ConfigName ShowAdvancedOptions -Value 1
```

SQL Instance Configuration

```
Set-DbaMaxMemory -SqlInstance $sqlInstance
```

SQL Instance Configuration

```
Set-DbaStartupParameter -SqlInstance "$node\$SQLInstanceName" -TraceFlag 3226 -Confirm:$false -Force
```

SQL Instance Configuration

```
New-DbaDatabase -SqlInstance $sqlInstance -Database DBA -RecoveryModel Simple -Owner 'sqladmin'
```

- ✓ ~~Install SQL Server on the 2 nodes~~
- ✓ ~~Configure SQL Server instances~~
- ✓ Perform hardening (NTFS permissions, revoke connect to guest, etc.)
- ✓ Install Ola Hallengren's Maintenance Solution
- ✓ Install WhoIsActive
- ✓ Create SQL Agent Jobs and Schedules
- ✓ Create a new TestDB database
- ✓ Create an Availability Group
- ✓ Create an Availability Group Listener
- ✓ Verify SPNs

SQL Instance Configuration

```powershell
foreach ($node in $sqlNodes.Keys) {
    (Get-DbaInstanceProtocol -ComputerName $node | Where-Object { $_.DisplayName -eq 'Named Pipes' }).Disable()
}
```

SQL Instance Configuration

```powershell
Get-DbaLogin -SqlInstance $sqlInstance -Login 'sa' | Set-DbaLogin -NewName 'sqladmin' -Disable
```

- ✓ ~~Install SQL Server on the 2 nodes~~
- ✓ ~~Configure SQL Server instances~~
- ✓ ~~Perform hardening (NTFS permissions, revoke connect to guest, etc.)~~
- ✓ Install Ola Hallengren's Maintenance Solution
- ✓ Install WhoIsActive
- ✓ Create SQL Agent Jobs and Schedules
- ✓ Create a new TestDB database
- ✓ Create an Availability Group
- ✓ Create an Availability Group Listener
- ✓ Verify SPNs

SQL Instance Configuration

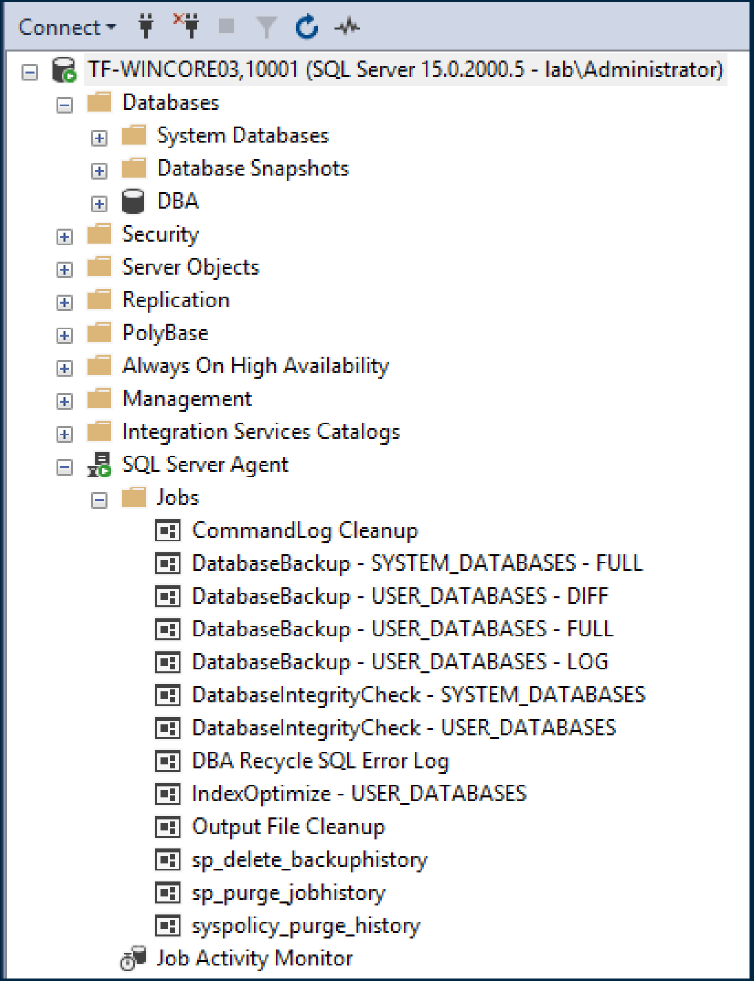
```powershell
Install-DbaMaintenanceSolution -SqlInstance $sqlInstance -ReplaceExisting -CleanupTime 48 -InstallJobs -Database DBA
Install-DbaWhoIsActive -SqlInstance $sqlInstance -Database DBA
```

- ✓ ~~Install SQL Server on the 2 nodes~~
- ✓ ~~Configure SQL Server instances~~
- ✓ ~~Perform hardening (NTFS permissions, revoke connect to guest, etc.)~~
- ✓ ~~Install Ola Hallengren's Maintenance Solution~~
- ✓ ~~Install WhoIsActive~~
- ✓ **Create SQL Agent Jobs and Schedules**
- ✓ Create a new TestDB database
- ✓ Create an Availability Group
- ✓ Create an Availability Group Listener
- ✓ Verify SPNs

SQL Instance Configuration

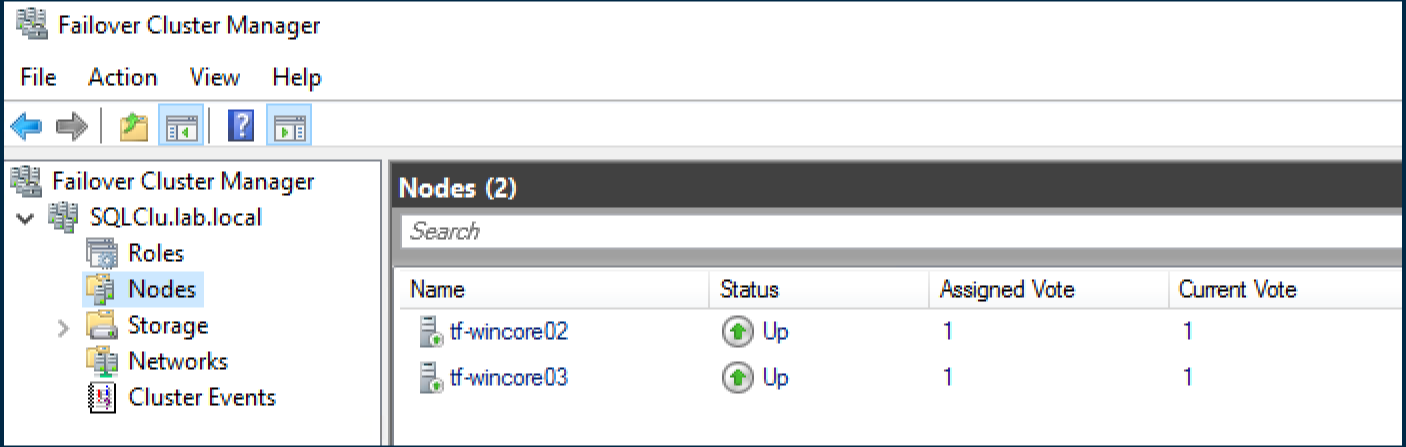
```powershell
$recycleJobName = 'DBA Recycle SQL Error Log'

New-DbaAgentJob -SqlInstance $sqlInstance -Job $recycleJobName `
    -Description 'Recycle SQL Error Log' -OwnerLogin sqladmin -Category 'DBA Reports'

New-DbaAgentJobStep -SqlInstance $sqlInstance -Job $recycleJobName `
    -StepName 'Recycle SQL Error Log' -Subsystem TransactSql -Database master -Command 'EXEC sp_cycle_errorlog'

New-DbaAgentSchedule -SqlInstance $sqlInstance -Job $recycleJobName `
    -Schedule 'Every midnight at 12:01' -FrequencyType Daily -FrequencyInterval EveryDay -StartTime 000100 -Force
```

Get-Instances



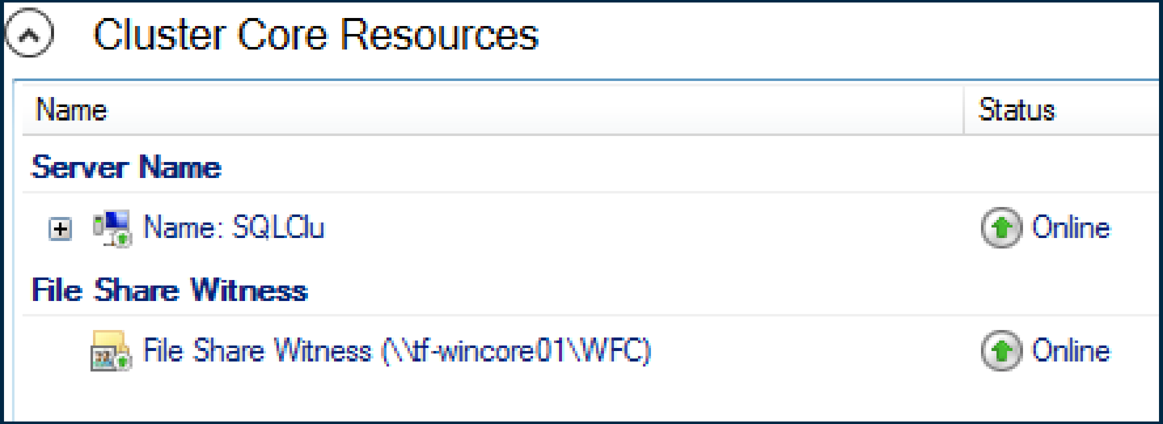
```
TF-WINCORE03,10001 (SQL Server 15.0.2000.5 - lab\Administrator)
    Databases
        System Databases
        Database Snapshots
        DBA
    Security
    Server Objects
    Replication
    PolyBase
    Always On High Availability
    Management
    Integration Services Catalogs
    SQL Server Agent
        Jobs
            CommandLog Cleanup
            DatabaseBackup - SYSTEM_DATABASES - FULL
            DatabaseBackup - USER_DATABASES - DIFF
            DatabaseBackup - USER_DATABASES - FULL
            DatabaseBackup - USER_DATABASES - LOG
            DatabaseIntegrityCheck - SYSTEM_DATABASES
            DatabaseIntegrityCheck - USER_DATABASES
            DBA Recycle SQL Error Log
            IndexOptimize - USER_DATABASES
            Output File Cleanup
            sp_delete_backuphistory
            sp_purge_jobhistory
            syspolicy_purge_history
        Job Activity Monitor
```

Create WFC



Name	Status	Assigned Vote	Current Vote
tf-wincore02	Up	1	1
tf-wincore03	Up	1	1

Create WFC



Cluster Core Resources

Name	Status
Server Name	
Name: SQLClu	Online
File Share Witness	
File Share Witness (\\tf-wincore01\WFC)	Online

- ✓ ~~Install SQL Server on the 2 nodes~~
- ✓ ~~Configure SQL Server instances~~
- ✓ ~~Perform hardening (NTFS permissions, revoke connect to guest, etc.)~~
- ✓ ~~Install Ola Hallengren's Maintenance Solution~~
- ✓ ~~Install WhoIsActive~~
- ✓ ~~Create SQL Agent Jobs and Schedules~~
- ✓ **Create a new TestDB database**
- ✓ Create an Availability Group
- ✓ Create an Availability Group Listener
- ✓ Verify SPNs

Create Availability Group

```
New-DbaDatabase -SqlInstance "$node1,$SQLInstancePort" -Database TestDB `
    -RecoveryModel Full -Owner sqladmin
Backup-DbaDatabase -SqlInstance "$node1,$SQLInstancePort" -Database TestDB `
    -FilePath C:\Temp\TestDB.bak -Type Full -IgnoreFileChecks
```

Create Availability Group

```powershell
foreach ($node in $sqlNodes.Keys) {
    Enable-DbaAgHadr -SqlInstance "$node\$SQLInstanceName" -Force -Confirm:$false
}
```

- ✓ ~~Install SQL Server on the 2 nodes~~
- ✓ ~~Configure SQL Server instances~~
- ✓ ~~Perform hardening (NTFS permissions, revoke connect to guest, etc.)~~
- ✓ ~~Install Ola Hallengren's Maintenance Solution~~
- ✓ ~~Install WhoIsActive~~
- ✓ ~~Create SQL Agent Jobs and Schedules~~
- ✓ ~~Create a new TestDB database~~
- ✓ **Create an Availability Group**
- ✓ Create an Availability Group Listener
- ✓ Verify SPNs

Create Availability Group

```powershell
$agParams = @{
    Name         = "$name_ag"
    Primary      = "$node1,$SQLInstancePort"
    Secondary    = $remaininNodesInstances
    Database     = "TestDB"
    ClusterType  = "Wsfc"
    SeedingMode  = "Automatic"
    FailoverMode = "Automatic"
    Confirm      = $false
    Verbose      = $true
}
New-DbaAvailabilityGroup @agParams
```

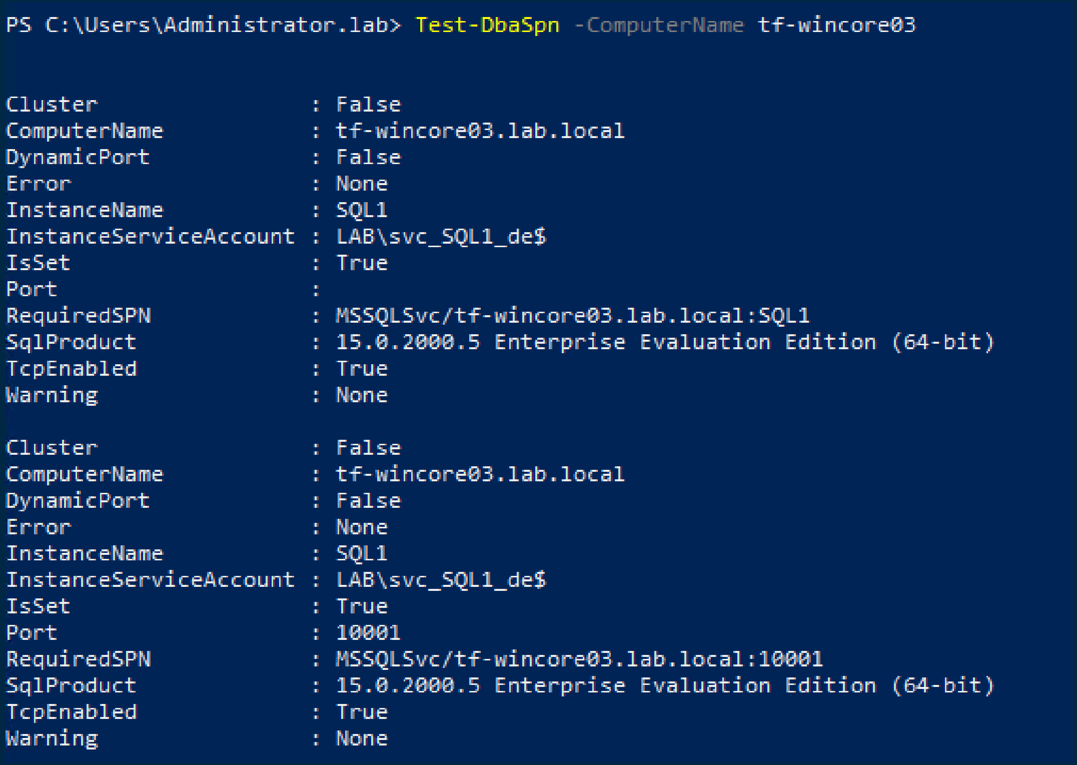
- ✓ ~~Install SQL Server on the 2 nodes~~
- ✓ ~~Configure SQL Server instances~~
- ✓ ~~Perform hardening (NTFS permissions, revoke connect to guest, etc.)~~
- ✓ ~~Install Ola Hallengren's Maintenance Solution~~
- ✓ ~~Install WhoIsActive~~
- ✓ ~~Create SQL Agent Jobs and Schedules~~
- ✓ ~~Create a new TestDB database~~
- ✓ ~~Create an Availability Group~~
- ✓ **Create an Availability Group Listener**
- ✓ Verify SPNs

Create Availability Group

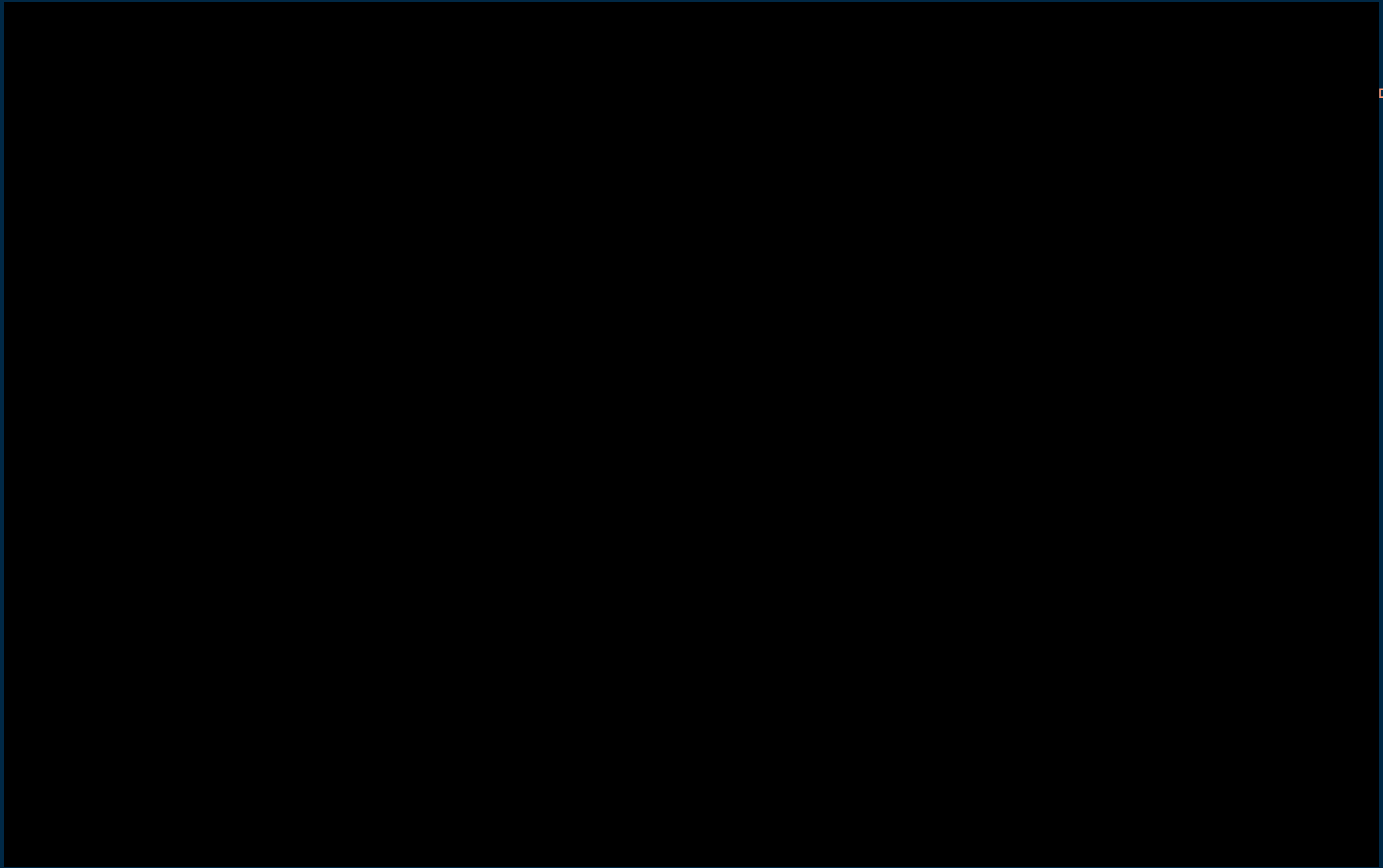
```
Add-DbaAgListener -SqlInstance "$node1,$SQLInstancePort" `
    -Name $name_ag_listener -AvailabilityGroup $name_ag `
    -IPAddress $ag_listener_ip -SubnetMask $netmask `
    -Port $SQLInstancePort -Verbose
```

- ✓ ~~Install SQL Server on the 2 nodes~~
- ✓ ~~Configure SQL Server instances~~
- ✓ ~~Perform hardening (NTFS permissions, revoke connect to guest, etc.)~~
- ✓ ~~Install Ola Hallengren's Maintenance Solution~~
- ✓ ~~Install WhoIsActive~~
- ✓ ~~Create SQL Agent Jobs and Schedules~~
- ✓ ~~Create a new TestDB database~~
- ✓ ~~Create an Availability Group~~
- ✓ ~~Create an Availability Group Listener~~
- ✓ Verify SPNs

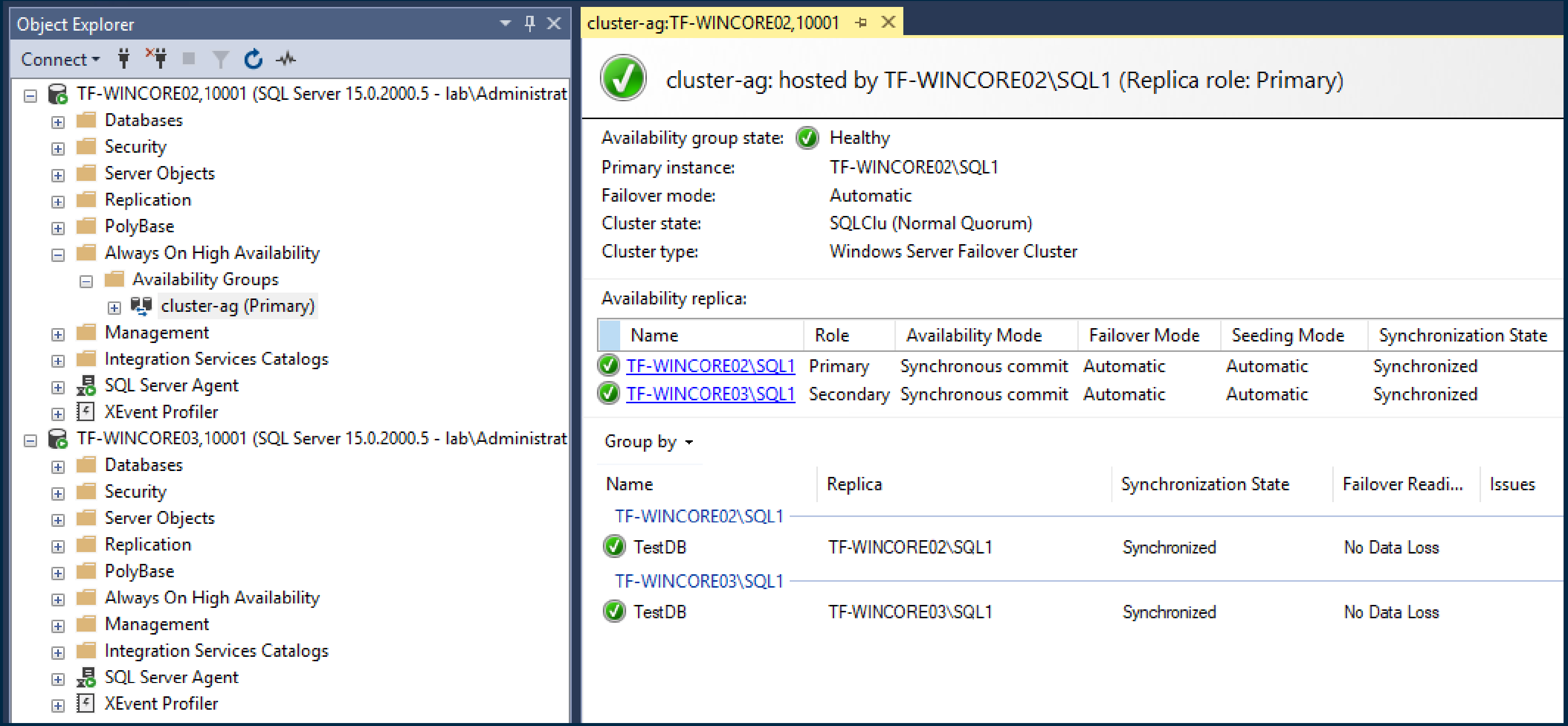
SPNs



```
PS C:\Users\Administrator.lab> Test-DbaSpn -ComputerName tf-wincore03


Cluster                 : False
ComputerName            : tf-wincore03.lab.local
DynamicPort             : False
Error                   : None
InstanceName            : SQL1
InstanceServiceAccount  : LAB\svc_SQL1_de$
IsSet                   : True
Port                    :
RequiredSPN             : MSSQLSvc/tf-wincore03.lab.local:SQL1
SqlProduct              : 15.0.2000.5 Enterprise Evaluation Edition (64-bit)
TcpEnabled              : True
Warning                 : None

Cluster                 : False
ComputerName            : tf-wincore03.lab.local
DynamicPort             : False
Error                   : None
InstanceName            : SQL1
InstanceServiceAccount  : LAB\svc_SQL1_de$
IsSet                   : True
Port                    : 10001
RequiredSPN             : MSSQLSvc/tf-wincore03.lab.local:10001
SqlProduct              : 15.0.2000.5 Enterprise Evaluation Edition (64-bit)
TcpEnabled              : True
Warning                 : None
```



- ✓ ~~Install SQL Server on the 2 nodes~~
- ✓ ~~Configure SQL Server instances~~
- ✓ ~~Perform hardening (NTFS permissions, revoke connect to guest, etc.)~~
- ✓ ~~Install Ola Hallengren's Maintenance Solution~~
- ✓ ~~Install WhoIsActive~~
- ✓ ~~Create SQL Agent Jobs and Schedules~~
- ✓ ~~Create a new TestDB database~~
- ✓ ~~Create an Availability Group~~
- ✓ ~~Create an Availability Group Listener~~
- ✓ ~~Verify SPNs~~



Object Explorer

Connect ▾

- TF-WINCORE02,10001 (SQL Server 15.0.2000.5 - lab\Administrat
  - Databases
  - Security
  - Server Objects
  - Replication
  - PolyBase
  - Always On High Availability
    - Availability Groups
      - cluster-ag (Primary)
  - Management
  - Integration Services Catalogs
  - SQL Server Agent
  - XEvent Profiler
- TF-WINCORE03,10001 (SQL Server 15.0.2000.5 - lab\Administrat
  - Databases
  - Security
  - Server Objects
  - Replication
  - PolyBase
  - Always On High Availability
  - Management
  - Integration Services Catalogs
  - SQL Server Agent
  - XEvent Profiler

cluster-ag:TF-WINCORE02,10001

cluster-ag: hosted by TF-WINCORE02\SQL1 (Replica role: Primary)

Availability group state:	✓ Healthy
Primary instance:	TF-WINCORE02\SQL1
Failover mode:	Automatic
Cluster state:	SQLClu (Normal Quorum)
Cluster type:	Windows Server Failover Cluster

Availability replica:

Name	Role	Availability Mode	Failover Mode	Seeding Mode	Synchronization State
✓ TF-WINCORE02\SQL1	Primary	Synchronous commit	Automatic	Automatic	Synchronized
✓ TF-WINCORE03\SQL1	Secondary	Synchronous commit	Automatic	Automatic	Synchronized

Group by ▾

Name	Replica	Synchronization State	Failover Readi...	Issues
TF-WINCORE02\SQL1				
✓ TestDB	TF-WINCORE02\SQL1	Synchronized	No Data Loss	
TF-WINCORE03\SQL1				
✓ TestDB	TF-WINCORE03\SQL1	Synchronized	No Data Loss	

Get-Resources

https://learn.hashicorp.com/
https://www.packer.io/docs
https://www.terraform.io/docs/index.html
https://docs.dbatools.io/
https://dbatools.io/blog/

Thank you!

SQL Community is amazing!

Do you have any questions?

viorel.ciucu@vciconsulting.net

vciconsulting.net

VCI Consulting

THANK YOU