Azure Virtual Desktop

Building automated & reliable Golden Images with HashiCorp Packer

About Me

Julian Mooren
Azure Cloud Architect

- Azure Core Infrastruktur (Landing Zone, Connectivity)
- Azure Security (Governance, Identity)
- Cloud Automation (HashiCorp, Ansible, PowerShell)
- Virtual Desktop Infrastructure (Citrix, AVD, Parallels)
- Application Delivery (Load Balancing, Reverse Proxy)

MVP

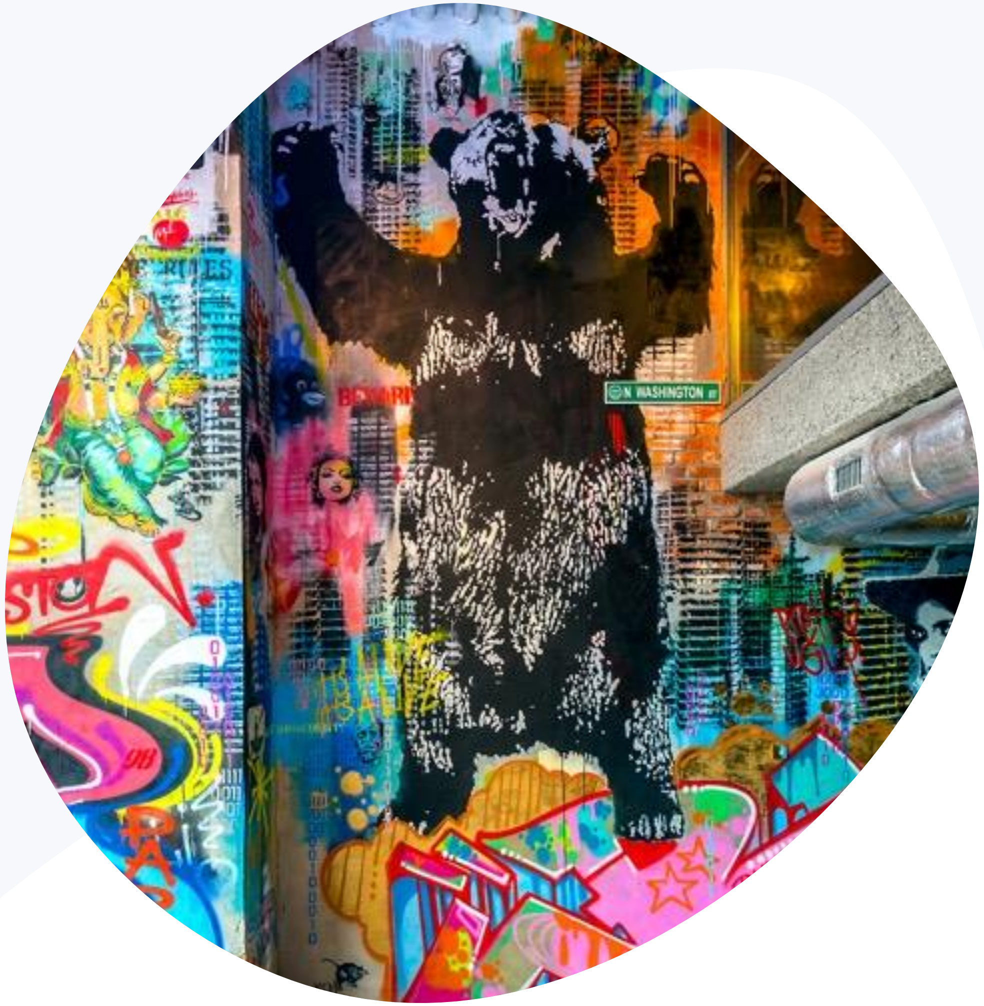
CTP
Citrix Technology Professional

Parallels
Enabled by Alludo
MEMBER
Very Important Parallels Professional
VIPP

Stuttgart

Agenda

- Why Golden Images?
- Packer 101
- Golden Image Handling
- Software Packaging
- Fasten your Deployments
- Unit Testing
- CI/CD Strategy
- Hostpool Image Update
- Repo Structure
- Ask me Anything



Why Golden Images?

Why Golden Images?

Benefits

➕ Manually built images are **time consuming and error prone**

➕ **Knowledge gaps** when building manual

➕ Virtual machines can be deployed in **just a few minutes**

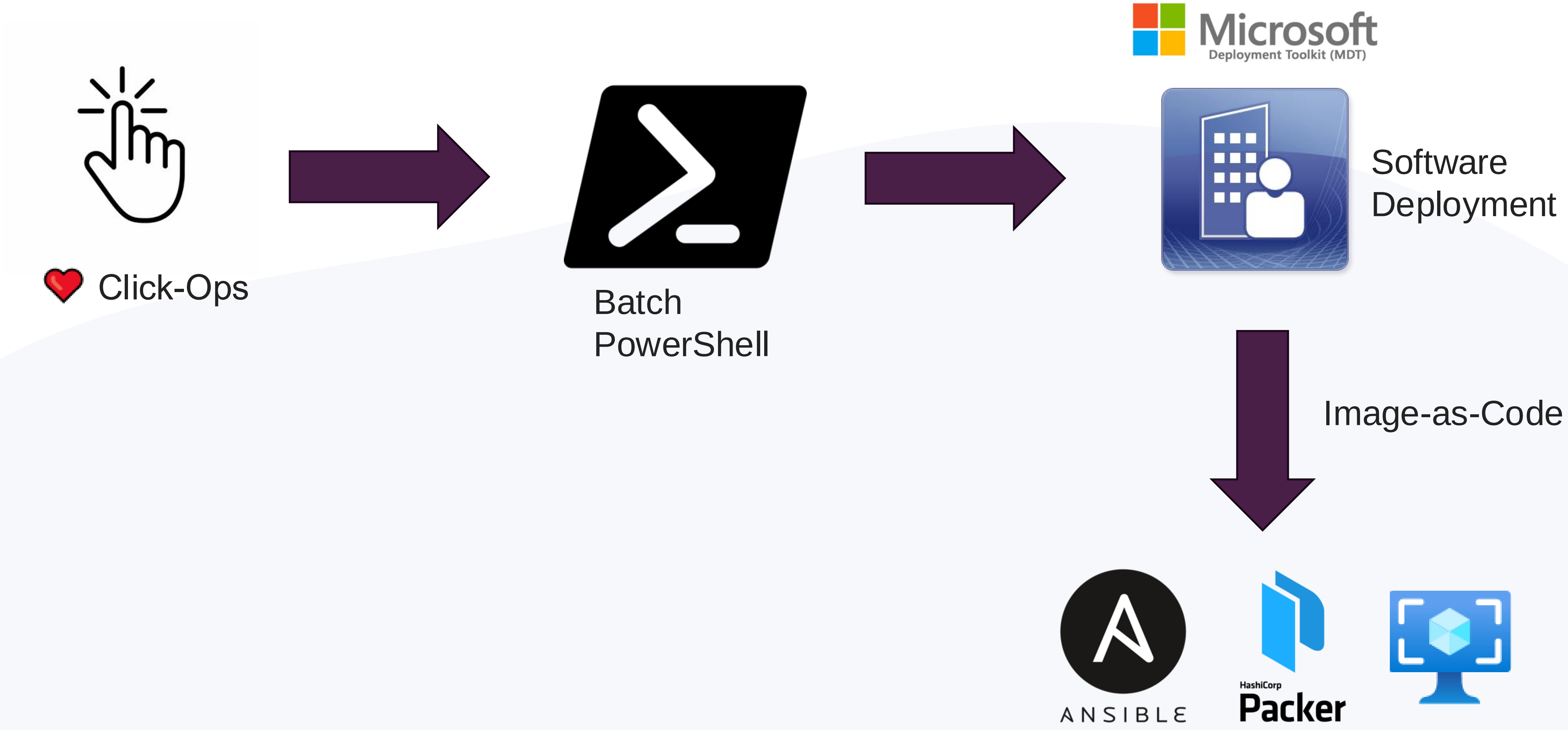
➕ **Simplifies** operating system **version upgrades**

➖ Takes more **effort** to build

➖ Application **Compatibility** Challenges

Why Golden Images?

Evolution of Image Building



❤ Click-Ops

Batch
PowerShell

Software
Deployment

Image-as-Code

Why Golden Images?

Packer vs. Azure Image Builder

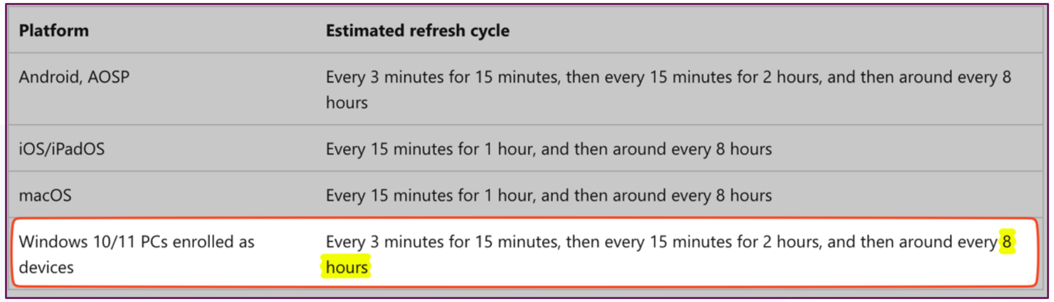
Why prefer Packer over Azure Image Builder (AIB)?

- **Packer under the hood →** Why add a middle layer? No need for Click-Ops
- AIB may not be available in your **Azure Region(s)** → Always check ✔
- AIB only supports **JSON & Bicep Templates** → No HCL Support! 😔
- **Built-In Scripts** (Language Packs, RDP Shortpath, FSLogix) are „okay“ → No Full Control ⚠
  Build your Own!

Why Golden Images?

Intune

Can we use Intune for creating Image Creation?



Platform	Estimated refresh cycle
Android, AOSP	Every 3 minutes for 15 minutes, then every 15 minutes for 2 hours, and then around every 8 hours
iOS/iPadOS	Every 15 minutes for 1 hour, and then around every 8 hours
macOS	Every 15 minutes for 1 hour, and then around every 8 hours
Windows 10/11 PCs enrolled as devices	Every 3 minutes for 15 minutes, then every 15 minutes for 2 hours, and then around every 8 hours

It is possible BUT comes with lot of **drawbacks:**

- No **offical support** from Microsoft
- Refresh Cycle up to 8 hours.... Slow as Hell 🐌🐌🐌🐌🐌🐌🐌
- No possibility to create Windows Server based Images (App requirements)
- Different story with Personal / Persistent Hostpools → Recommendation: Ansible ✅

Packer 101

Functionality & Terminology

Packer 101

General

- HashiCorp Packer is an **open-source** tool for creating Golden Images
- Available for all **major operating systems**: Windows, macOS, and Linux
- Can be used for **Windows and Linux** Image creation
- Communication via **SSH** (Linux) or **WinRM** (Windows).

Why use Packer?

- **Multi-provider portability**: Azure, AWS, GCP and VMware → No Vendor-Lock
- Source Control – easy **tracking of changes**
- **CI/CD Integration** → Pipelines publish new images directly to test or production environments.

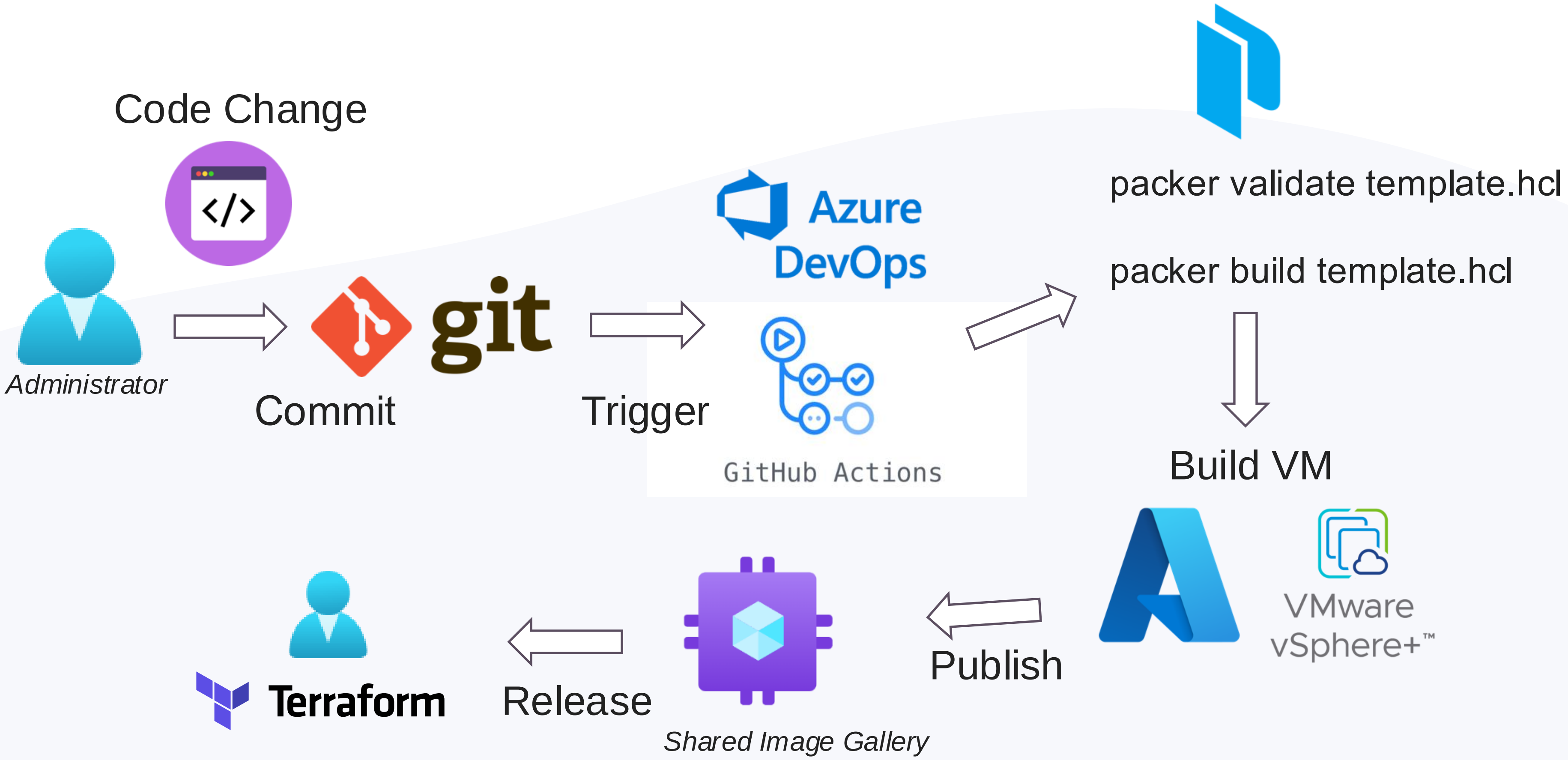
Packer 101

Terminology

- Template(s) – HCL or JSON-File with the **Build Definition**
- Command(s) – Sub-Command for **Packer Process** (validate, build)
- Build(s) – **Task** for creating an Image (pinned to source plattform)
- Provisioner(s) – Installation and configuration of **software** (scripts, reboot)
- Data Source(s) – Grab **informations** from outside the packer build environment

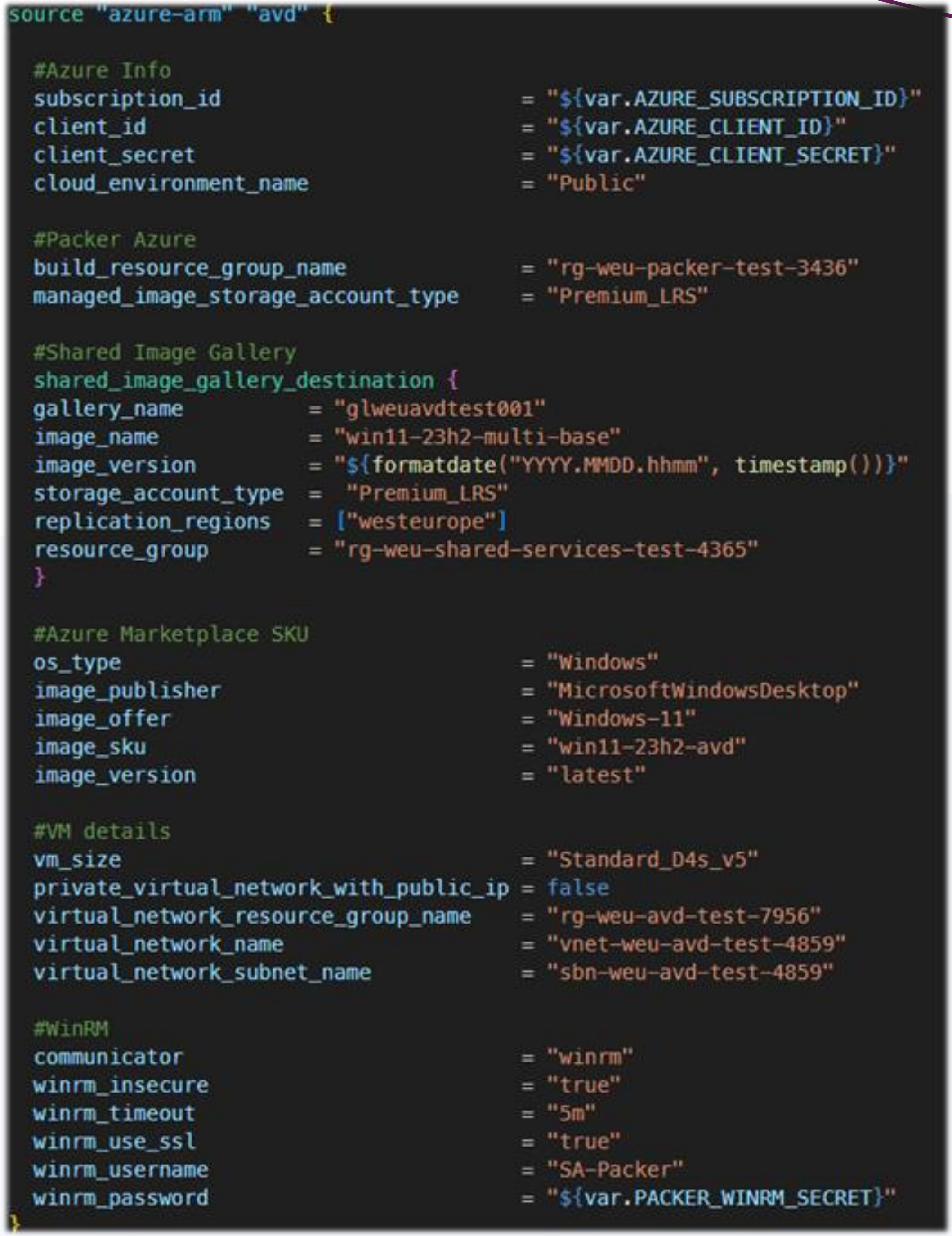
Packer 101

Architecture



Packer 101

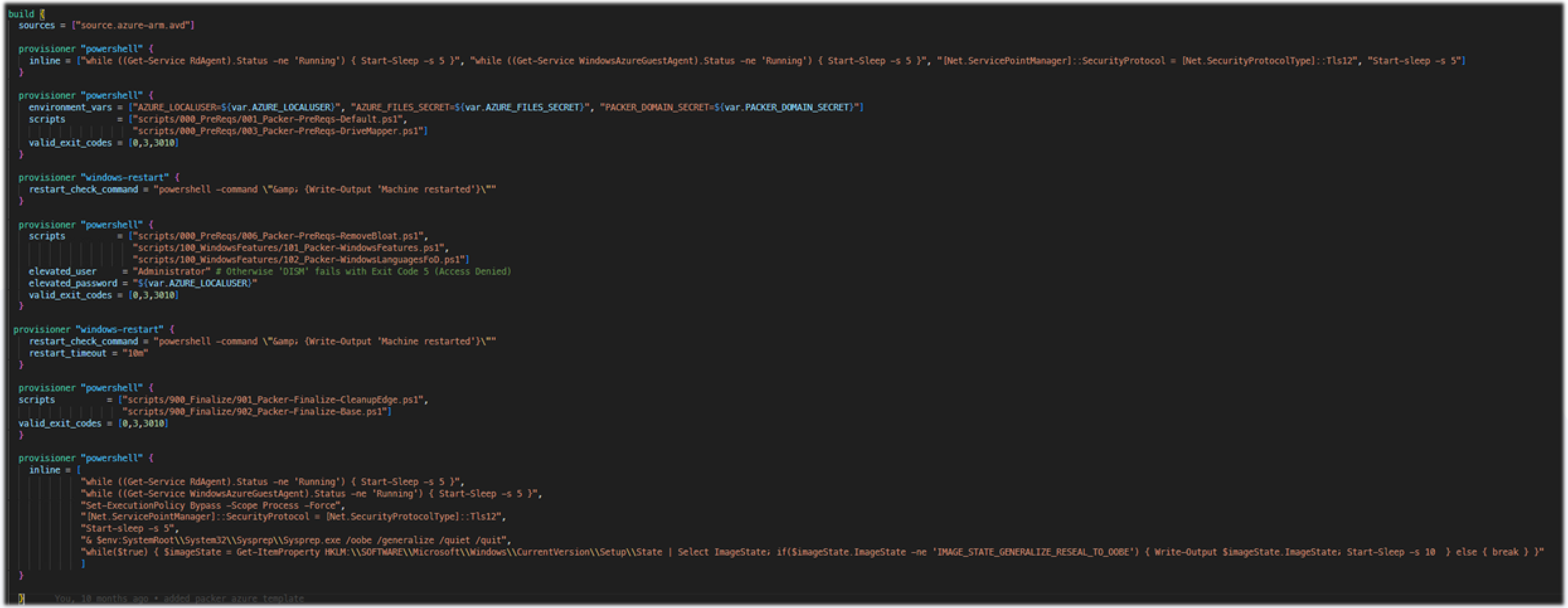
Template Structure „Source“



```
source "azure-arm" "avd" {

  #Azure Info
  subscription_id                        = "${var.AZURE_SUBSCRIPTION_ID}"
  client_id                              = "${var.AZURE_CLIENT_ID}"
  client_secret                          = "${var.AZURE_CLIENT_SECRET}"
  cloud_environment_name                 = "Public"

  #Packer Azure
  build_resource_group_name              = "rg-weu-packer-test-3436"
  managed_image_storage_account_type     = "Premium_LRS"

  #Shared Image Gallery
  shared_image_gallery_destination {
  gallery_name          = "glweuavdtest001"
  image_name            = "win11-23h2-multi-base"
  image_version         = "${formatdate("YYYY.MMDD.hhmm", timestamp())}"
  storage_account_type  =  "Premium_LRS"
  replication_regions   = ["westeurope"]
  resource_group        = "rg-weu-shared-services-test-4365"
  }

  #Azure Marketplace SKU
  os_type                                = "Windows"
  image_publisher                        = "MicrosoftWindowsDesktop"
  image_offer                            = "Windows-11"
  image_sku                              = "win11-23h2-avd"
  image_version                          = "latest"

  #VM details
  vm_size                                = "Standard_D4s_v5"
  private_virtual_network_with_public_ip = false
  virtual_network_resource_group_name    = "rg-weu-avd-test-7956"
  virtual_network_name                   = "vnet-weu-avd-test-4859"
  virtual_network_subnet_name            = "sbn-weu-avd-test-4859"

  #WinRM
  communicator                           = "winrm"
  winrm_insecure                         = "true"
  winrm_timeout                          = "5m"
  winrm_use_ssl                          = "true"
  winrm_username                         = "SA-Packer"
  winrm_password                         = "${var.PACKER_WINRM_SECRET}"
}
```

Packer 101

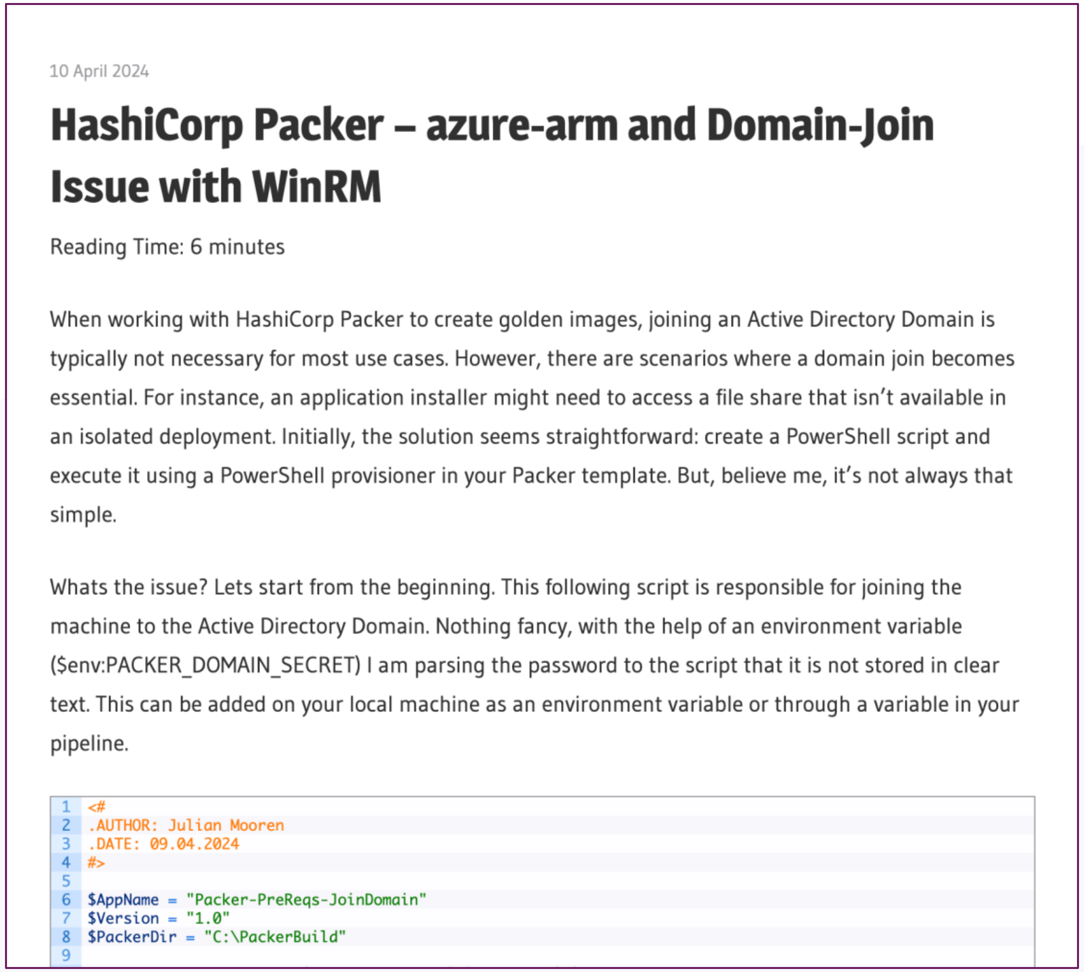
Template Structure „Build“



```
build {
  sources = ["source.azure-arm.avd"]

  provisioner "powershell" {
    inline = ["while ((Get-Service RdAgent).Status -ne 'Running') { Start-Sleep -s 5 }", "while ((Get-Service WindowsAzureGuestAgent).Status -ne 'Running') { Start-Sleep -s 5 }", "[Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12", "Start-sleep -s 5"]
  }

  provisioner "powershell" {
    environment_vars = ["AZURE_LOCALUSER=${var.AZURE_LOCALUSER}", "AZURE_FILES_SECRET=${var.AZURE_FILES_SECRET}", "PACKER_DOMAIN_SECRET=${var.PACKER_DOMAIN_SECRET}"]
    scripts          = ["scripts/000_PreReqs/001_Packer-PreReqs-Default.ps1",
                        "scripts/000_PreReqs/003_Packer-PreReqs-DriveMapper.ps1"]
    valid_exit_codes = [0,3,3010]
  }

  provisioner "windows-restart" {
    restart_check_command = "powershell -command \"&amp; {Write-Output 'Machine restarted'}\""
  }

  provisioner "powershell" {
    scripts          = ["scripts/000_PreReqs/006_Packer-PreReqs-RemoveBloat.ps1",
                        "scripts/100_WindowsFeatures/101_Packer-WindowsFeatures.ps1",
                        "scripts/100_WindowsFeatures/102_Packer-WindowsLanguagesFoD.ps1"]
    elevated_user     = "Administrator" # Otherwise 'DISM' fails with Exit Code 5 (Access Denied)
    elevated_password = "${var.AZURE_LOCALUSER}"
    valid_exit_codes = [0,3,3010]
  }

 provisioner "windows-restart" {
    restart_check_command = "powershell -command \"&amp; {Write-Output 'Machine restarted'}\""
    restart_timeout = "10m"
  }

  provisioner "powershell" {
  scripts          = ["scripts/900_Finalize/901_Packer-Finalize-CleanupEdge.ps1",
                      "scripts/900_Finalize/902_Packer-Finalize-Base.ps1"]
  valid_exit_codes = [0,3,3010]
  }

  provisioner "powershell" {
    inline = [
              "while ((Get-Service RdAgent).Status -ne 'Running') { Start-Sleep -s 5 }",
              "while ((Get-Service WindowsAzureGuestAgent).Status -ne 'Running') { Start-Sleep -s 5 }",
              "Set-ExecutionPolicy Bypass -Scope Process -Force",
              "[Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12",
              "Start-sleep -s 5",
              "& $env:SystemRoot\\System32\\Sysprep\\Sysprep.exe /oobe /generalize /quiet /quit",
              "while($true) { $imageState = Get-ItemProperty HKLM:\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Setup\\State | Select ImageState; if($imageState.ImageState -ne 'IMAGE_STATE_GENERALIZE_RESEAL_TO_OOBE') { Write-Output $imageState.ImageState; Start-Sleep -s 10  } else { break } }"
              ]
  }

}      You, 10 months ago • added packer azure template
```

Packer 101

Domain-Join during Image Creation? 👺



10 April 2024

HashiCorp Packer – azure-arm and Domain-Join Issue with WinRM

Reading Time: 6 minutes

When working with HashiCorp Packer to create golden images, joining an Active Directory Domain is typically not necessary for most use cases. However, there are scenarios where a domain join becomes essential. For instance, an application installer might need to access a file share that isn't available in an isolated deployment. Initially, the solution seems straightforward: create a PowerShell script and execute it using a PowerShell provisioner in your Packer template. But, believe me, it's not always that simple.

Whats the issue? Lets start from the beginning. This following script is responsible for joining the machine to the Active Directory Domain. Nothing fancy, with the help of an environment variable ($env:PACKER_DOMAIN_SECRET) I am parsing the password to the script that it is not stored in clear text. This can be added on your local machine as an environment variable or through a variable in your pipeline.

```
<#
.AUTHOR: Julian Mooren
.DATE: 09.04.2024
#>

$AppName = "Packer-PreReqs-JoinDomain"
$Version = "1.0"
$PackerDir = "C:\PackerBuild"

```

Golden Image Handling

Golden Image Handling

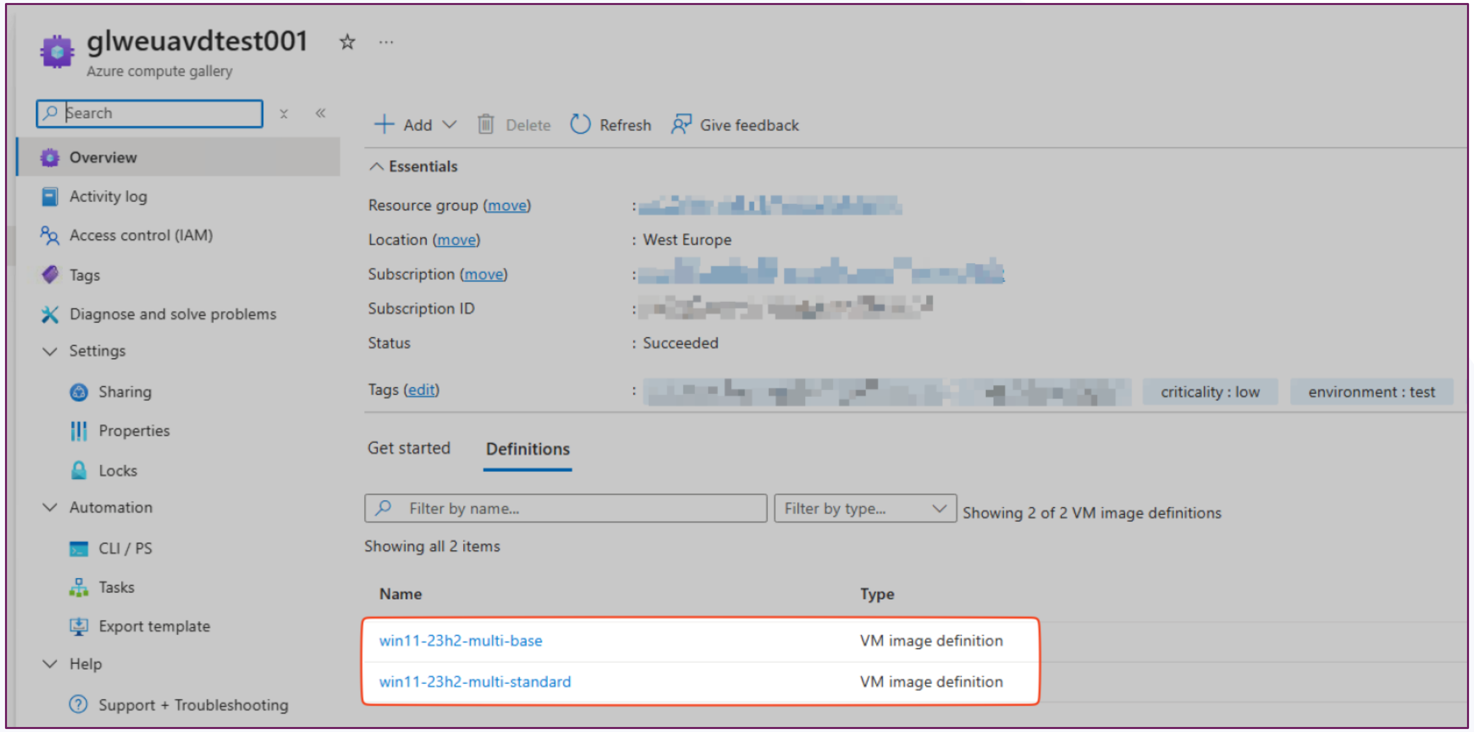
Shared Image Gallery

- Image Management Service
  - **Image Definition** (Generalized, CPU, Memory)
  - **Image Versions** (Image-Release)
- High-Availability (ZRS)
- Image Sharing between Subscriptions

<u>No billing</u> for Image Gallery
Expect:

- Storage
- Network (Egress)

Image Management without SIG?
Possible but **better together** ✅



Resource group (move) : [redacted]
Location (move) : West Europe
Subscription (move) : [redacted]
Subscription ID : [redacted]
Status : Succeeded
Tags (edit) : [redacted] criticality : low environment : test

Showing 2 of 2 VM image definitions

Showing all 2 items

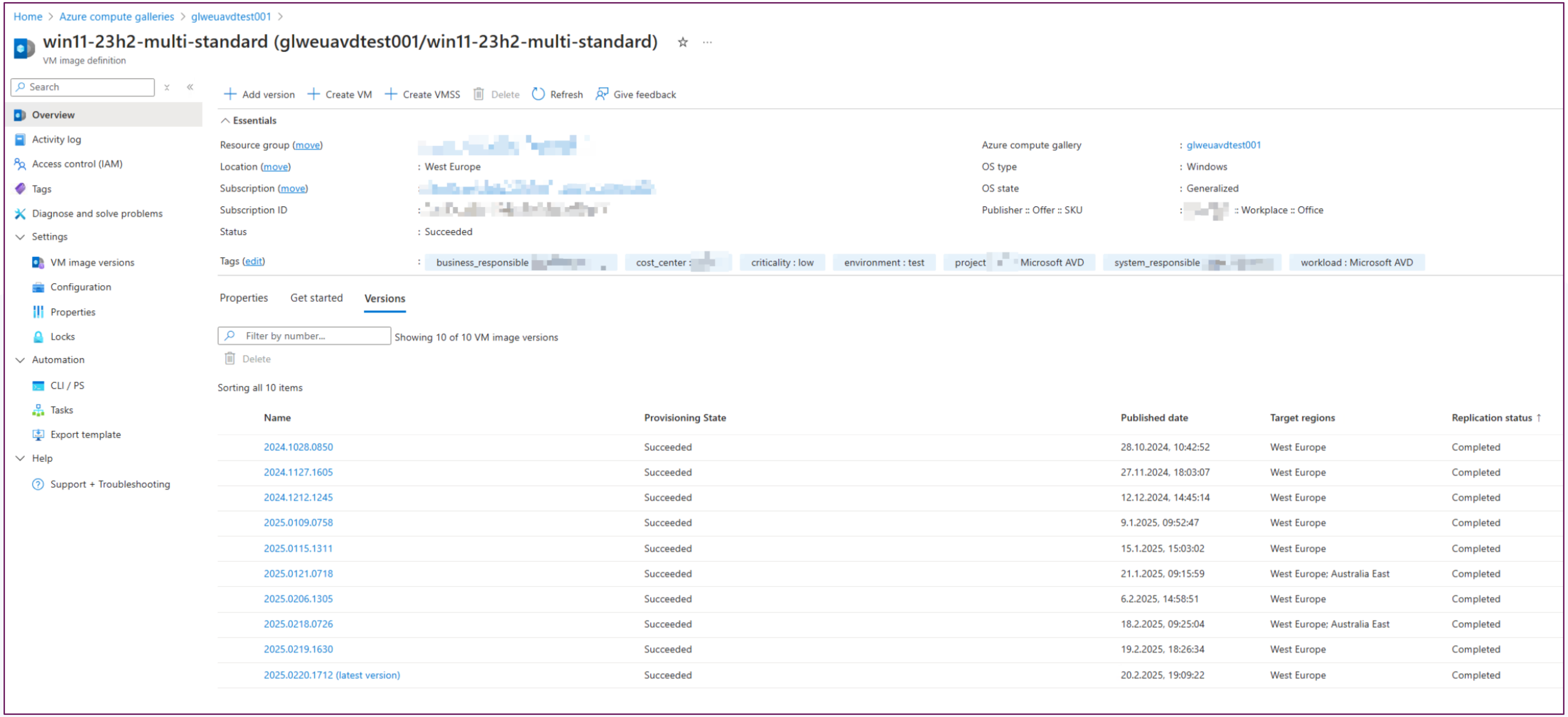
Name	Type
win11-23h2-multi-base	VM image definition
win11-23h2-multi-standard	VM image definition

Image Definitions

Golden Image Handling

Shared Image Gallery

Image Versions



win11-23h2-multi-standard (glweuavdtest001/win11-23h2-multi-standard)

VM image definition

Essentials

Resource group (move) : [redacted]
Location (move) : West Europe
Subscription (move) : [redacted]
Subscription ID : [redacted]
Status : Succeeded
Tags (edit) : business_responsible [redacted] | cost_center : [redacted] | criticality : low | environment : test | project [redacted] Microsoft AVD | system_responsible [redacted] | workload : Microsoft AVD

Azure compute gallery : glweuavdtest001
OS type : Windows
OS state : Generalized
Publisher :: Offer :: SKU : [redacted] :: Workplace :: Office

Showing 10 of 10 VM image versions

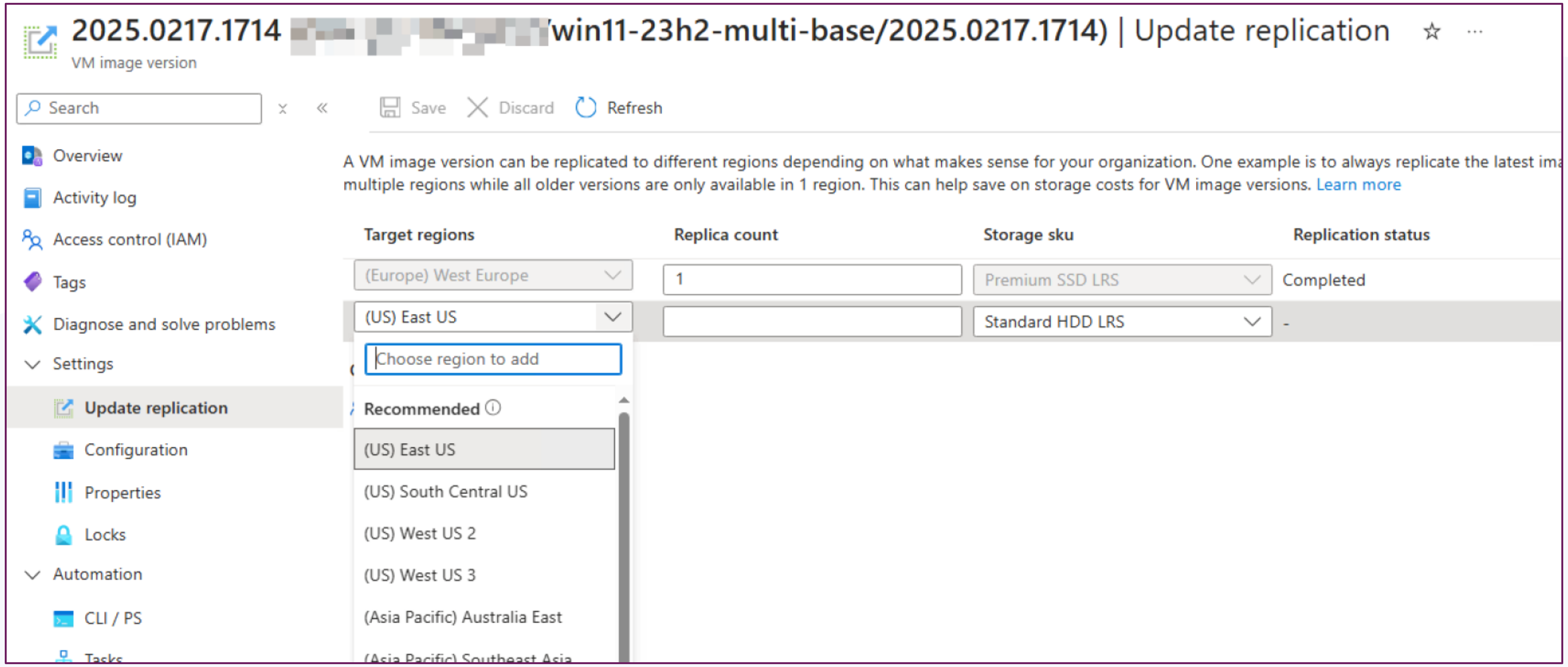
Sorting all 10 items

Name	Provisioning State	Published date	Target regions	Replication status ↑
2024.1028.0850	Succeeded	28.10.2024, 10:42:52	West Europe	Completed
2024.1127.1605	Succeeded	27.11.2024, 18:03:07	West Europe	Completed
2024.1212.1245	Succeeded	12.12.2024, 14:45:14	West Europe	Completed
2025.0109.0758	Succeeded	9.1.2025, 09:52:47	West Europe	Completed
2025.0115.1311	Succeeded	15.1.2025, 15:03:02	West Europe	Completed
2025.0121.0718	Succeeded	21.1.2025, 09:15:59	West Europe; Australia East	Completed
2025.0206.1305	Succeeded	6.2.2025, 14:58:51	West Europe	Completed
2025.0218.0726	Succeeded	18.2.2025, 09:25:04	West Europe; Australia East	Completed
2025.0219.1630	Succeeded	19.2.2025, 18:26:34	West Europe	Completed
2025.0220.1712 (latest version)	Succeeded	20.2.2025, 19:09:22	West Europe	Completed

Golden Image Handling

Shared Image Gallery

Image Replication between Regions



2025.0217.1714 [redacted]/win11-23h2-multi-base/2025.0217.1714) | Update replication

VM image version

A VM image version can be replicated to different regions depending on what makes sense for your organization. One example is to always replicate the latest ima multiple regions while all older versions are only available in 1 region. This can help save on storage costs for VM image versions. Learn more

Target regions	Replica count	Storage sku	Replication status
(Europe) West Europe	1	Premium SSD LRS	Completed
(US) East US		Standard HDD LRS	-

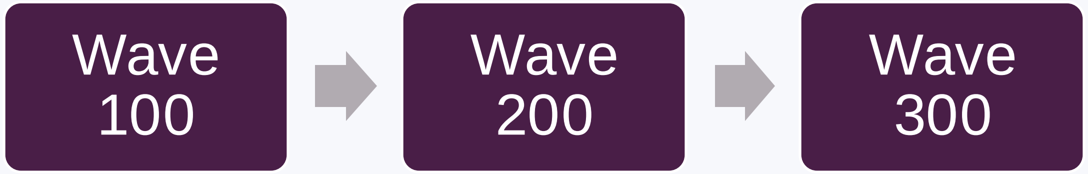
- 1 Replica Count = 20 Virtual Machines
- 100 Session Hosts = **Replica Count → <u>5</u>** 💡

Software Packaging

Software Packaging

PSAppDeployToolkit

- Use **PSAppDeployToolkit** for Application Packaging
- Create a **Packaging Template** for Team-Members → Same structure, easier to understand
- Applications will be hosted on an **Azure Storage Account** (Azure Files)
- Establish an Application & reboot **Dependency** with a **„Wave“ Script**
- **Wave Strategy** allows changing software packages **without editing** the packer template file
  → Packacking can be done by team members with no packer **knowledge**



Wave 100

Wave 200

Wave 300

PreReqs
- VC++
- .NET Runtime

Standard
- Browser
- Greenshot
- etc.

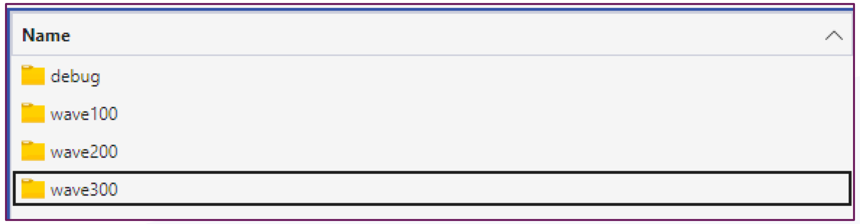
Spezial
- Outlook-Add-Ins
- Anti-Virus

💡 Documentation PSADT - [Introduction · PSAppDeployToolkit]

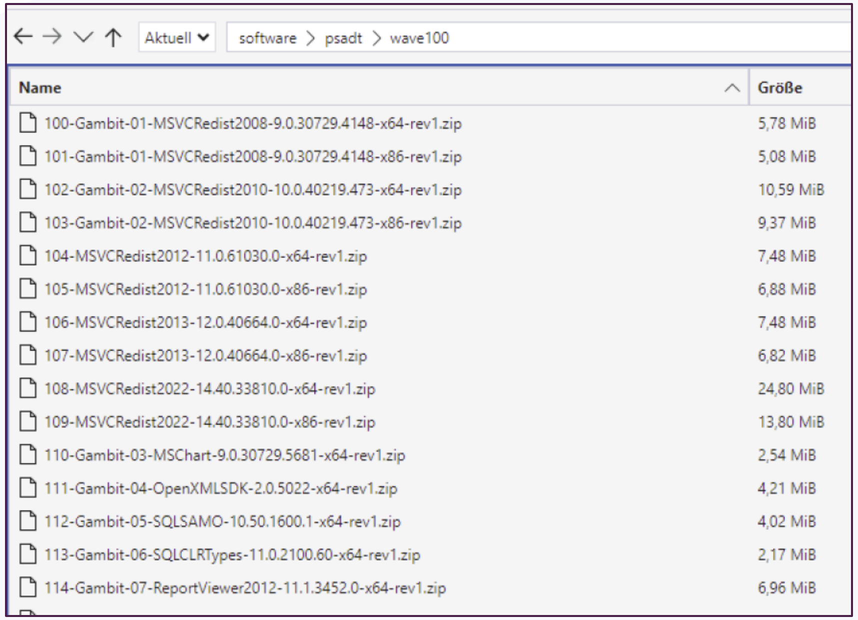
Software Packaging

PSAppDeployToolkit

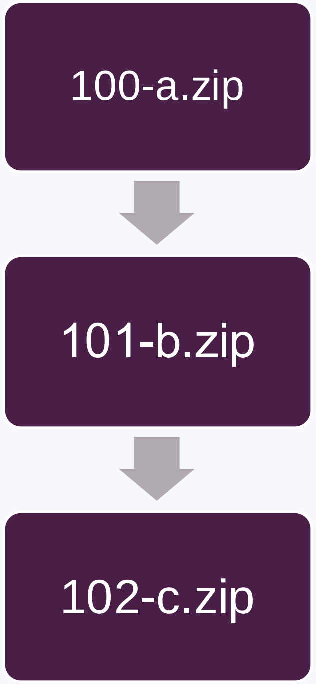
- Installation **Sequence** is controlled by the **prefix** in the PSADT bundle name



Name
debug
wave100
wave200
wave300



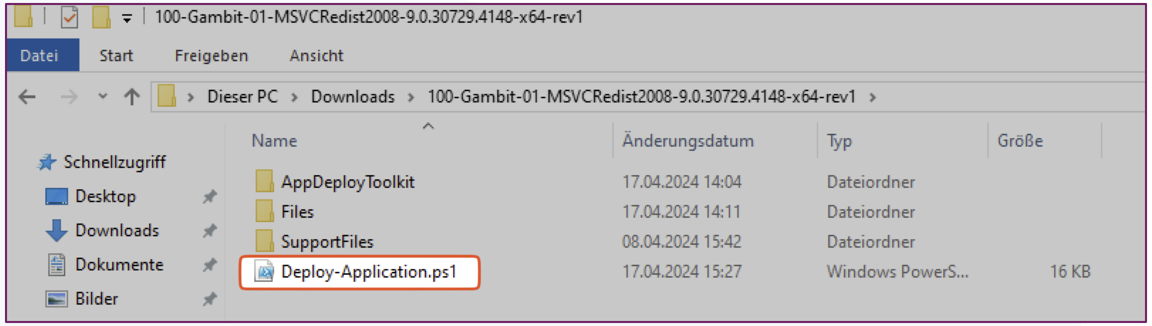
Name	Größe
100-Gambit-01-MSVCRedist2008-9.0.30729.4148-x64-rev1.zip	5,78 MiB
101-Gambit-01-MSVCRedist2008-9.0.30729.4148-x86-rev1.zip	5,08 MiB
102-Gambit-02-MSVCRedist2010-10.0.40219.473-x64-rev1.zip	10,59 MiB
103-Gambit-02-MSVCRedist2010-10.0.40219.473-x86-rev1.zip	9,37 MiB
104-MSVCRedist2012-11.0.61030.0-x64-rev1.zip	7,48 MiB
105-MSVCRedist2012-11.0.61030.0-x86-rev1.zip	6,88 MiB
106-MSVCRedist2013-12.0.40664.0-x64-rev1.zip	7,48 MiB
107-MSVCRedist2013-12.0.40664.0-x86-rev1.zip	6,82 MiB
108-MSVCRedist2022-14.40.33810.0-x64-rev1.zip	24,80 MiB
109-MSVCRedist2022-14.40.33810.0-x86-rev1.zip	13,80 MiB
110-Gambit-03-MSChart-9.0.30729.5681-x64-rev1.zip	2,54 MiB
111-Gambit-04-OpenXMLSDK-2.0.5022-x64-rev1.zip	4,21 MiB
112-Gambit-05-SQLSAMO-10.50.1600.1-x64-rev1.zip	4,02 MiB
113-Gambit-06-SQLCLRTypes-11.0.2100.60-x64-rev1.zip	2,17 MiB
114-Gambit-07-ReportViewer2012-11.1.3452.0-x64-rev1.zip	6,96 MiB



100-a.zip

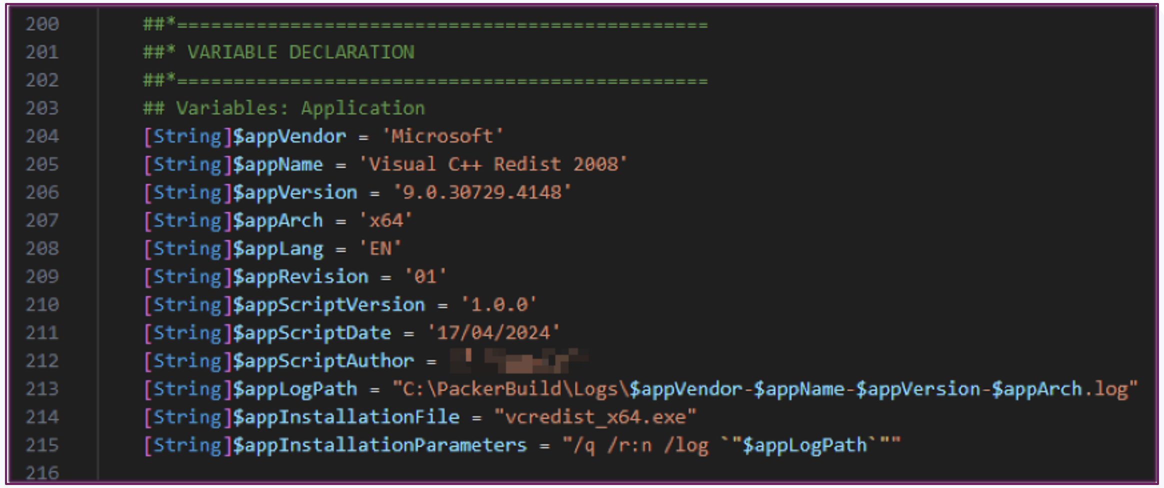
101-b.zip

102-c.zip



100-Gambit-01-MSVCRedist2008-9.0.30729.4148-x64-rev1
Datei
Start
Freigeben
Ansicht
Dieser PC > Downloads > 100-Gambit-01-MSVCRedist2008-9.0.30729.4148-x64-rev1
Schnellzugriff
Desktop
Downloads
Dokumente
Bilder

Name	Änderungsdatum	Typ	Größe
AppDeployToolkit	17.04.2024 14:04	Dateiordner	
Files	17.04.2024 14:11	Dateiordner	
SupportFiles	08.04.2024 15:42	Dateiordner	
Deploy-Application.ps1	17.04.2024 15:27	Windows PowerS...	16 KB



```
##*===============================================
##* VARIABLE DECLARATION
##*===============================================
## Variables: Application
[String]$appVendor = 'Microsoft'
[String]$appName = 'Visual C++ Redist 2008'
[String]$appVersion = '9.0.30729.4148'
[String]$appArch = 'x64'
[String]$appLang = 'EN'
[String]$appRevision = '01'
[String]$appScriptVersion = '1.0.0'
[String]$appScriptDate = '17/04/2024'
[String]$appScriptAuthor = [redacted]
[String]$appLogPath = "C:\PackerBuild\Logs\$appVendor-$appName-$appVersion-$appArch.log"
[String]$appInstallationFile = "vcredist_x64.exe"
[String]$appInstallationParameters = "/q /r:n /log `"$appLogPath`""

```

Software Packaging

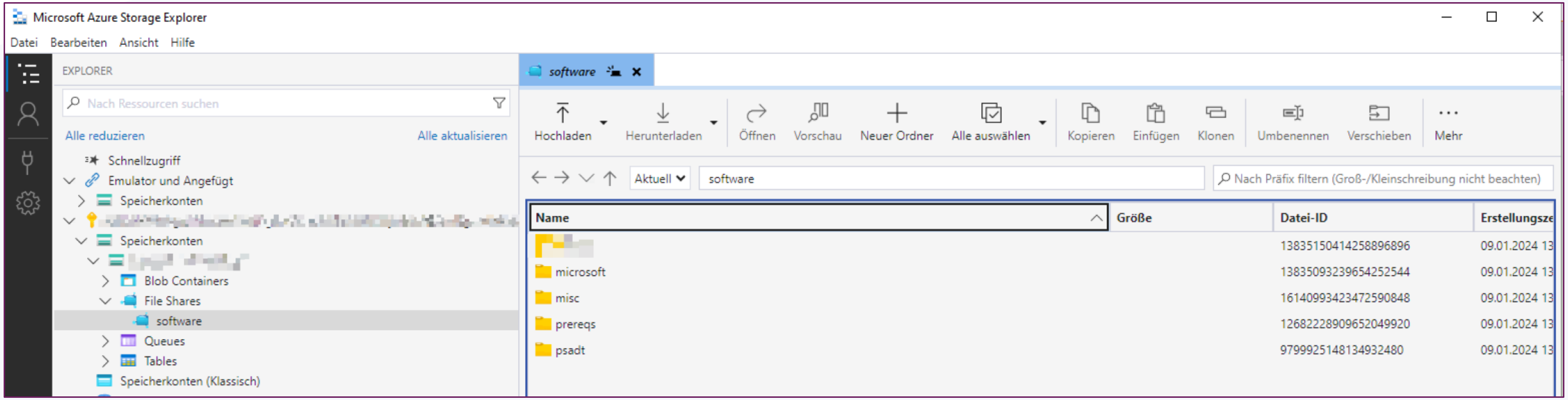
PSAppDeployToolkit

Access to Azure File Share (Storage Account) is possible in two ways:

- **Azure Storage Explorer** – Requires role assignment via Entra ID

  (Storage File Data SMB Share Contributor)

- **Windows Explorer via CIFS/SMB** – Access using an Access Key



[redacted]

[redacted]

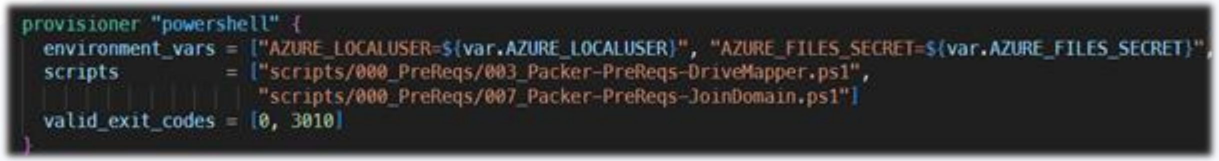
Name	Größe	Datei-ID	Erstellungsze
[redacted]		13835150414258896896	09.01.2024 13
microsoft		13835093239654252544	09.01.2024 13
misc		16140993423472590848	09.01.2024 13
prereqs		12682228909652049920	09.01.2024 13
psadt		9799925148134932480	09.01.2024 13

Software Packaging

PSAppDeployToolkit

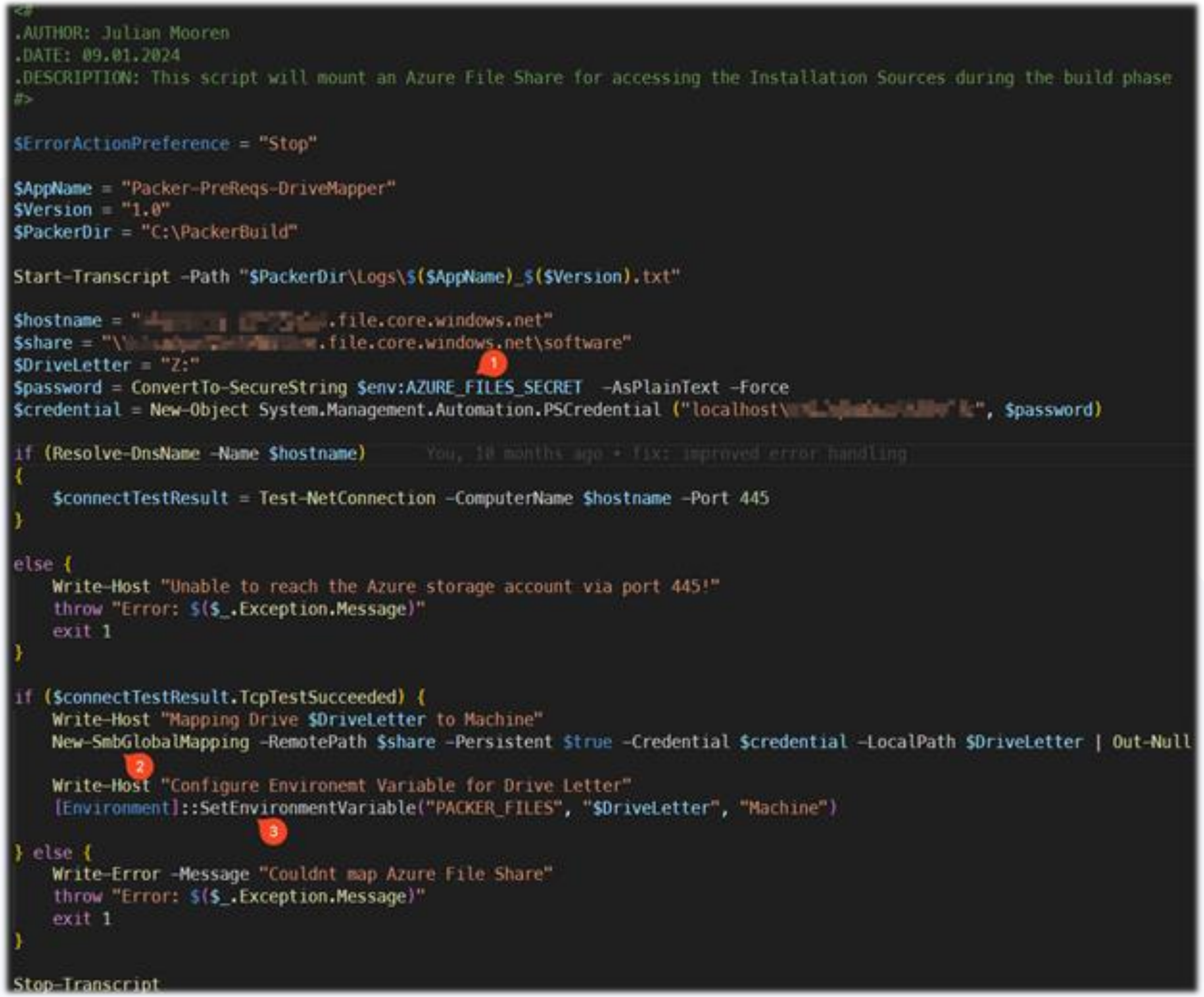
Azure Files Share is being **mapped** during Packer build process

Packer Template Snippet



```
provisioner "powershell" {
  environment_vars = ["AZURE_LOCALUSER=${var.AZURE_LOCALUSER}", "AZURE_FILES_SECRET=${var.AZURE_FILES_SECRET}",
  scripts          = ["scripts/000_PreReqs/003_Packer-PreReqs-DriveMapper.ps1",
                      "scripts/000_PreReqs/007_Packer-PreReqs-JoinDomain.ps1"]
  valid_exit_codes = [0, 3010]
}
```

003_Packer-PreReqs-DriveMapper.ps1

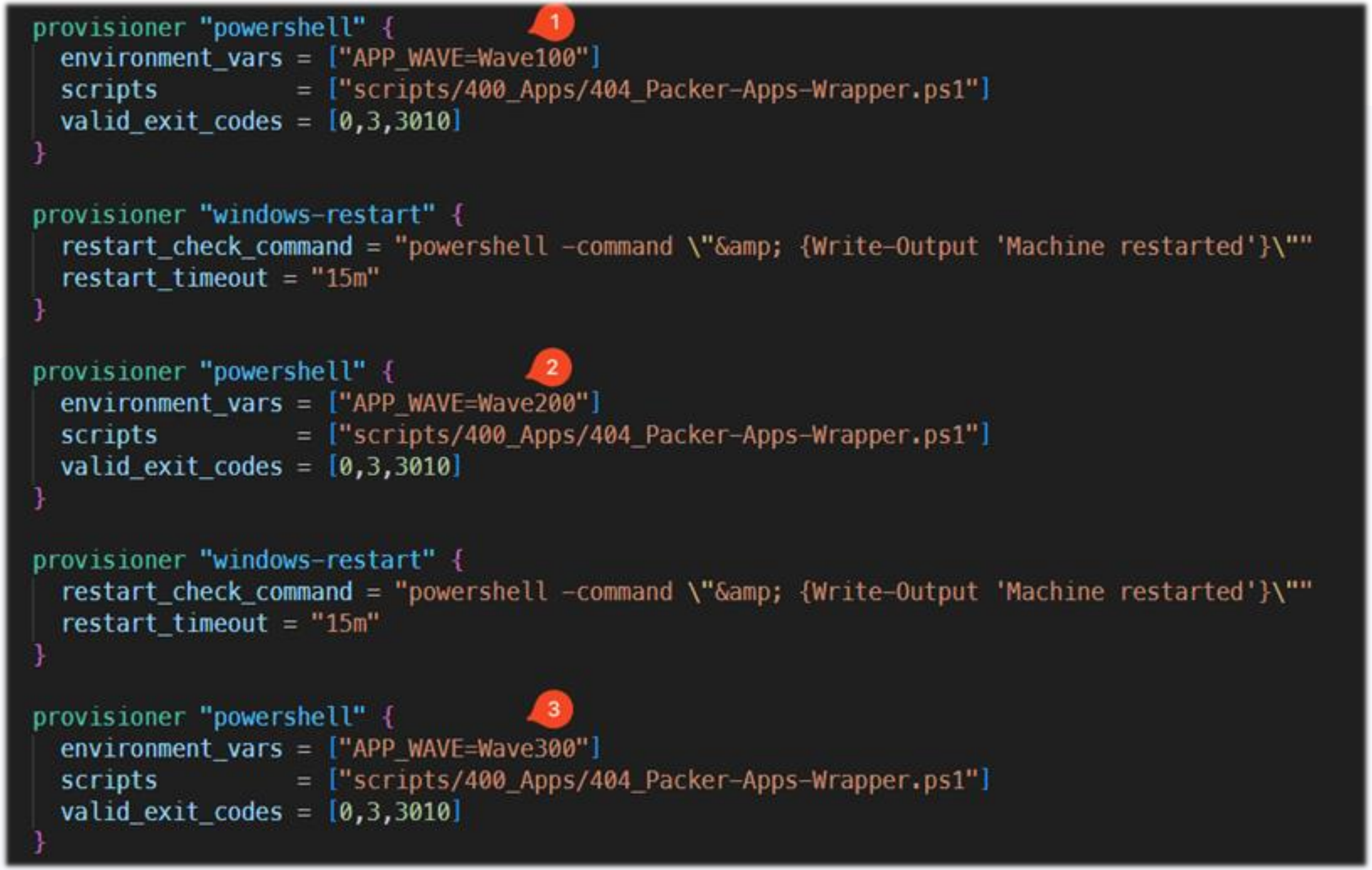


```
<#
.AUTHOR: Julian Mooren
.DATE: 09.01.2024
.DESCRIPTION: This script will mount an Azure File Share for accessing the Installation Sources during the build phase
#>

$ErrorActionPreference = "Stop"

$AppName = "Packer-PreReqs-DriveMapper"
$Version = "1.0"
$PackerDir = "C:\PackerBuild"

Start-Transcript -Path "$PackerDir\Logs\$($AppName)_$($Version).txt"

$hostname = "[redacted].file.core.windows.net"
$share = "\\[redacted].file.core.windows.net\software"
$DriveLetter = "Z:"
$password = ConvertTo-SecureString $env:AZURE_FILES_SECRET  -AsPlainText -Force
$credential = New-Object System.Management.Automation.PSCredential ("localhost\[redacted]", $password)

if (Resolve-DnsName -Name $hostname)       You, 10 months ago • fix: improved error handling
{
    $connectTestResult = Test-NetConnection -ComputerName $hostname -Port 445
}

else {
    Write-Host "Unable to reach the Azure storage account via port 445!"
    throw "Error: $($_.Exception.Message)"
    exit 1
}

if ($connectTestResult.TcpTestSucceeded) {
    Write-Host "Mapping Drive $DriveLetter to Machine"
    New-SmbGlobalMapping -RemotePath $share -Persistent $true -Credential $credential -LocalPath $DriveLetter | Out-Null

    Write-Host "Configure Environemt Variable for Drive Letter"
    [Environment]::SetEnvironmentVariable("PACKER_FILES", "$DriveLetter", "Machine")

} else {
    Write-Error -Message "Couldnt map Azure File Share"
    throw "Error: $($_.Exception.Message)"
    exit 1
}

Stop-Transcript
```

Software Packaging

PSAppDeployToolkit

Packer Template Snippet – “Waves”



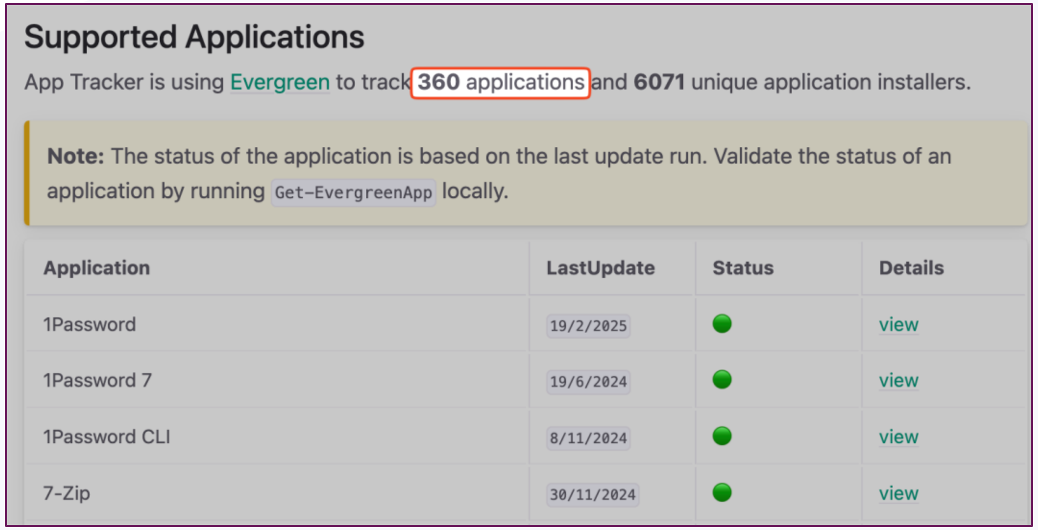
```
provisioner "powershell" {
  environment_vars = ["APP_WAVE=Wave100"]
  scripts          = ["scripts/400_Apps/404_Packer-Apps-Wrapper.ps1"]
  valid_exit_codes = [0,3,3010]
}

provisioner "windows-restart" {
  restart_check_command = "powershell -command \"&amp; {Write-Output 'Machine restarted'}\""
  restart_timeout = "15m"
}

provisioner "powershell" {
  environment_vars = ["APP_WAVE=Wave200"]
  scripts          = ["scripts/400_Apps/404_Packer-Apps-Wrapper.ps1"]
  valid_exit_codes = [0,3,3010]
}

provisioner "windows-restart" {
  restart_check_command = "powershell -command \"&amp; {Write-Output 'Machine restarted'}\""
  restart_timeout = "15m"
}

provisioner "powershell" {
  environment_vars = ["APP_WAVE=Wave300"]
  scripts          = ["scripts/400_Apps/404_Packer-Apps-Wrapper.ps1"]
  valid_exit_codes = [0,3,3010]
}
```

Software Packaging

Evergreen approach for basic applications

https://github.com/aaronparker/evergreen

- **360** Supported Applications
- Application version and download links are only pulled from **official sources**
- Do not waste your time 🕐 packaging default apps → Always deploy „latest“ (e.g. 7-Zip)

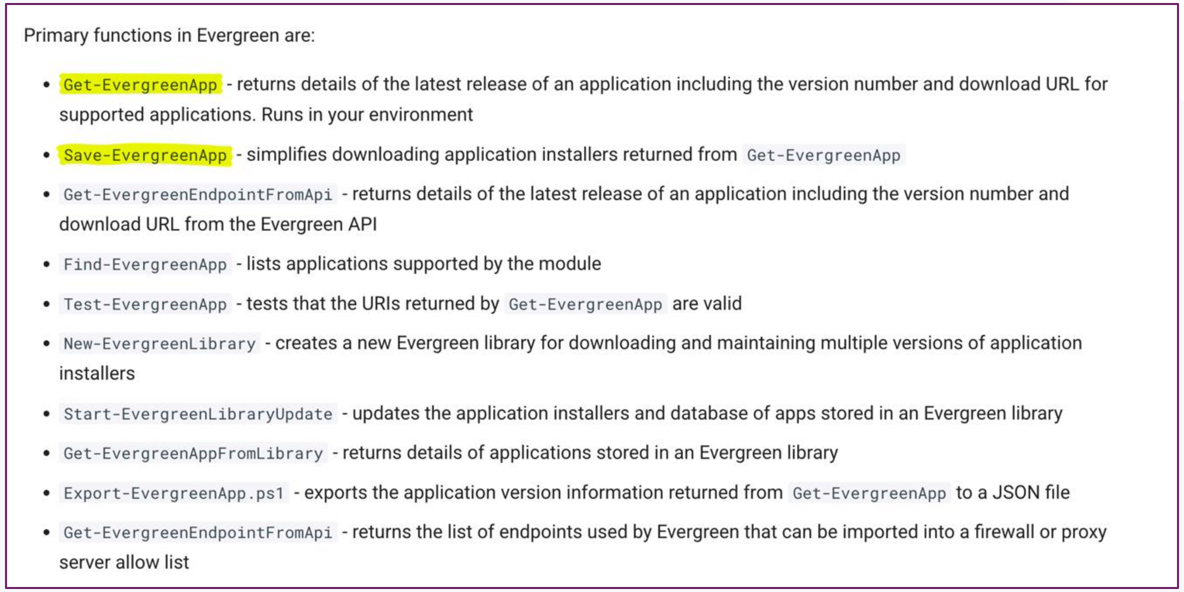


Supported Applications

App Tracker is using Evergreen to track 360 applications and **6071** unique application installers.

> **Note:** The status of the application is based on the last update run. Validate the status of an application by running `Get-EvergreenApp` locally.

Application	LastUpdate	Status	Details
1Password	19/2/2025	●	view
1Password 7	19/6/2024	●	view
1Password CLI	8/11/2024	●	view
7-Zip	30/11/2024	●	view

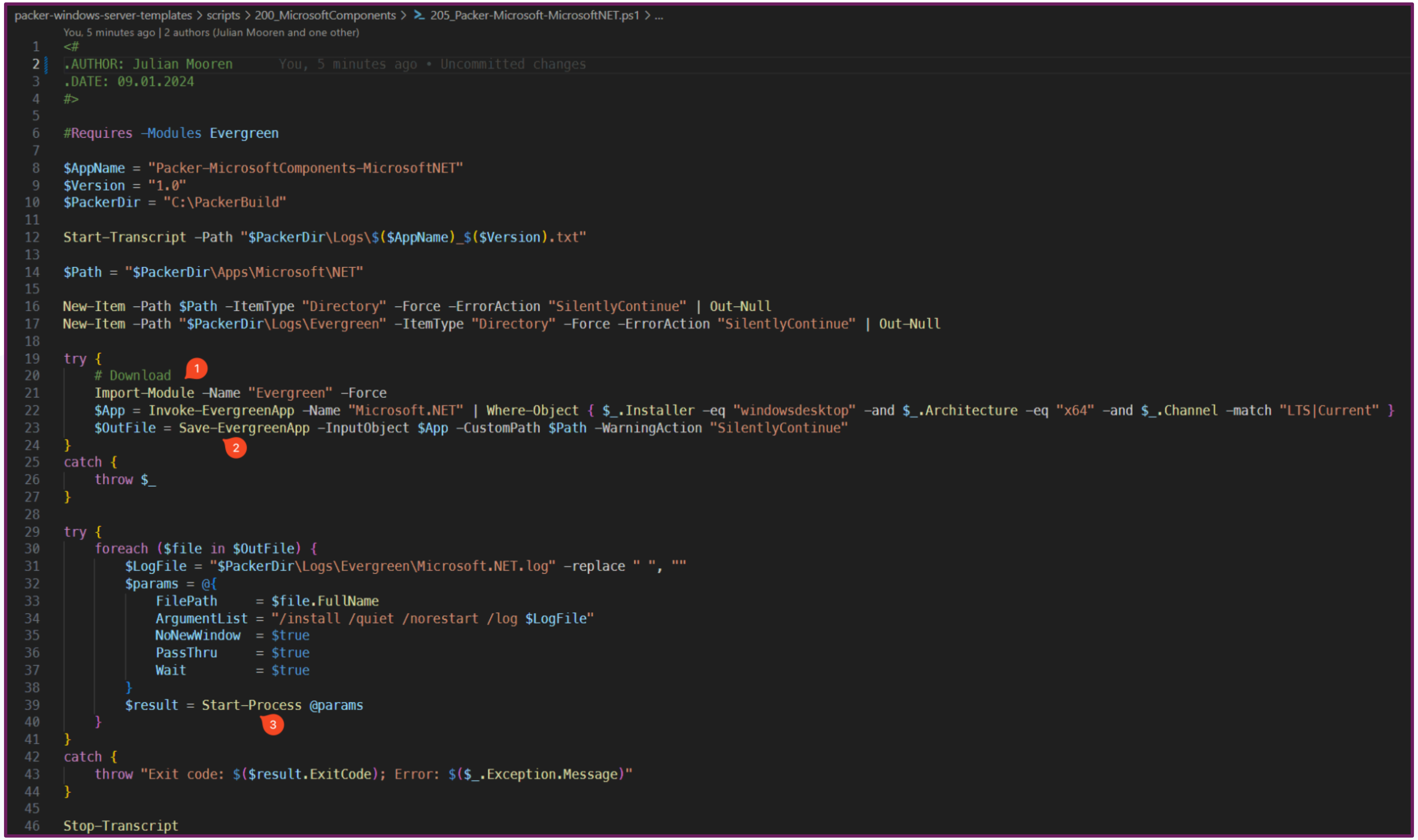


Primary functions in Evergreen are:

- `Get-EvergreenApp` - returns details of the latest release of an application including the version number and download URL for supported applications. Runs in your environment
- `Save-EvergreenApp` - simplifies downloading application installers returned from `Get-EvergreenApp`
- `Get-EvergreenEndpointFromApi` - returns details of the latest release of an application including the version number and download URL from the Evergreen API
- `Find-EvergreenApp` - lists applications supported by the module
- `Test-EvergreenApp` - tests that the URIs returned by `Get-EvergreenApp` are valid
- `New-EvergreenLibrary` - creates a new Evergreen library for downloading and maintaining multiple versions of application installers
- `Start-EvergreenLibraryUpdate` - updates the application installers and database of apps stored in an Evergreen library
- `Get-EvergreenAppFromLibrary` - returns details of applications stored in an Evergreen library
- `Export-EvergreenApp.ps1` - exports the application version information returned from `Get-EvergreenApp` to a JSON file
- `Get-EvergreenEndpointFromApi` - returns the list of endpoints used by Evergreen that can be imported into a firewall or proxy server allow list

Software Packaging

Microsoft.NET Runtime – Evergreen Example



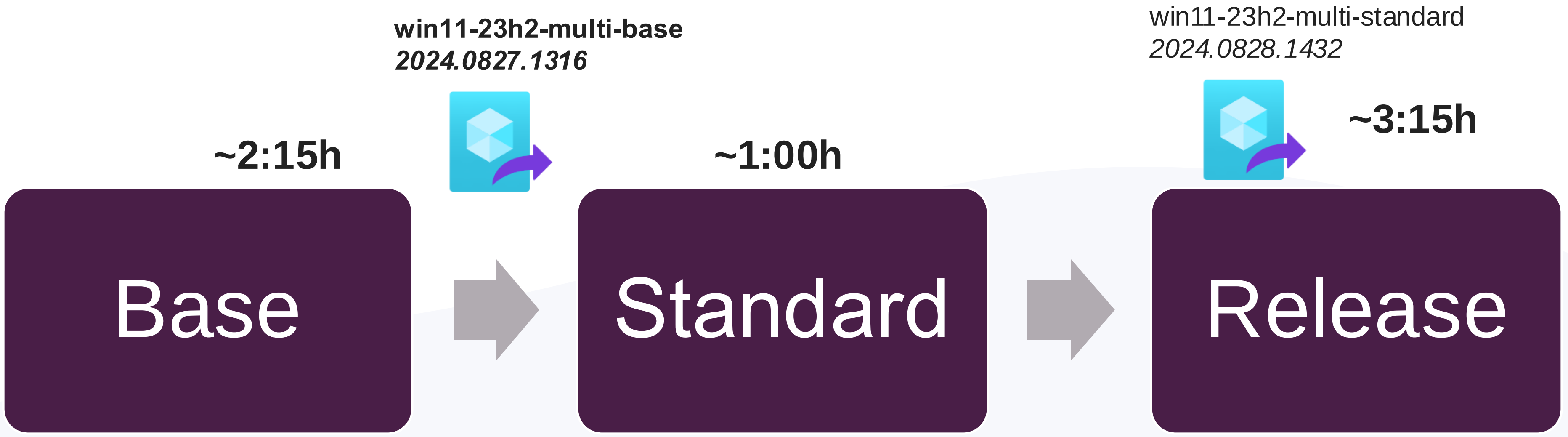
packer-windows-server-templates > scripts > 200_MicrosoftComponents > 205_Packer-Microsoft-MicrosoftNET.ps1 > ...

You, 5 minutes ago | 2 authors (Julian Mooren and one other)

```powershell
<#
.AUTHOR: Julian Mooren      You, 5 minutes ago • Uncommitted changes
.DATE: 09.01.2024
#>

#Requires -Modules Evergreen

$AppName = "Packer-MicrosoftComponents-MicrosoftNET"
$Version = "1.0"
$PackerDir = "C:\PackerBuild"

Start-Transcript -Path "$PackerDir\Logs\$($AppName)_$($Version).txt"

$Path = "$PackerDir\Apps\Microsoft\NET"

New-Item -Path $Path -ItemType "Directory" -Force -ErrorAction "SilentlyContinue" | Out-Null
New-Item -Path "$PackerDir\Logs\Evergreen" -ItemType "Directory" -Force -ErrorAction "SilentlyContinue" | Out-Null

try {
    # Download
    Import-Module -Name "Evergreen" -Force
    $App = Invoke-EvergreenApp -Name "Microsoft.NET" | Where-Object { $_.Installer -eq "windowsdesktop" -and $_.Architecture -eq "x64" -and $_.Channel -match "LTS|Current" }
    $OutFile = Save-EvergreenApp -InputObject $App -CustomPath $Path -WarningAction "SilentlyContinue"
}
catch {
    throw $_
}

try {
    foreach ($file in $OutFile) {
        $LogFile = "$PackerDir\Logs\Evergreen\Microsoft.NET.log" -replace " ", ""
        $params = @{
            FilePath     = $file.FullName
            ArgumentList = "/install /quiet /norestart /log $LogFile"
            NoNewWindow  = $true
            PassThru     = $true
            Wait         = $true
        }
        $result = Start-Process @params
    }
}
catch {
    throw "Exit code: $($result.ExitCode); Error: $($_.Exception.Message)"
}

Stop-Transcript
```

Fasten your Deployments

Fasten your Deployments



~2:15h

win11-23h2-multi-base
2024.0827.1316

~1:00h

win11-23h2-multi-standard
2024.0828.1432

~3:15h

Base

Standard

Release

Base

- Azure Marketplace Image
- Default Windows Customization
- Remove Bloat
- Windows Features
- Windows Language Packs

Standard

- AVD-Agent
- FSLogix
- Wave 100
- Microsoft 365
- Wave 200
- Wave 300
- Unit-Tests
- Sealing

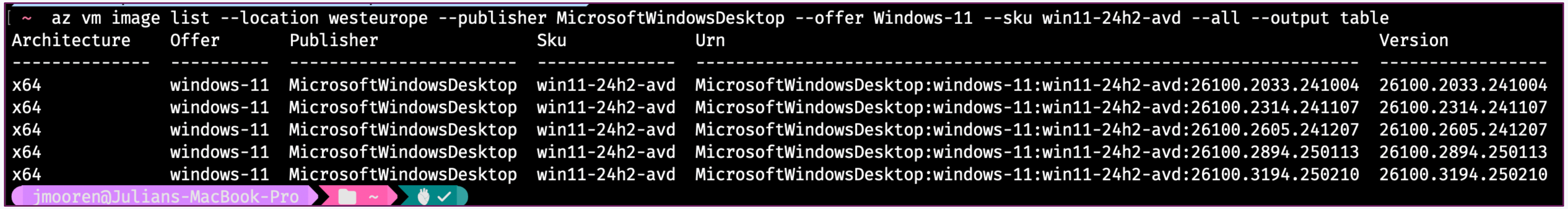
Fasten your Deployments

Recommendation

#1 Split your image build into **two layers** to deploy more efficiently ✅

#2 Freedom of Choice → Keep more than 5 versions (**Rollback Scenario**)

#3 Know whats inside your Image → ⚠ No **unplanned Change** of components (e.g FSLogix)



```
~  az vm image list --location westeurope --publisher MicrosoftWindowsDesktop --offer Windows-11 --sku win11-24h2-avd --all --output table
Architecture    Offer       Publisher                Sku             Urn                                                                 Version
--------------  ----------  -----------------------  --------------  ------------------------------------------------------------------  -----------------
x64             windows-11  MicrosoftWindowsDesktop  win11-24h2-avd  MicrosoftWindowsDesktop:windows-11:win11-24h2-avd:26100.2033.241004  26100.2033.241004
x64             windows-11  MicrosoftWindowsDesktop  win11-24h2-avd  MicrosoftWindowsDesktop:windows-11:win11-24h2-avd:26100.2314.241107  26100.2314.241107
x64             windows-11  MicrosoftWindowsDesktop  win11-24h2-avd  MicrosoftWindowsDesktop:windows-11:win11-24h2-avd:26100.2605.241207  26100.2605.241207
x64             windows-11  MicrosoftWindowsDesktop  win11-24h2-avd  MicrosoftWindowsDesktop:windows-11:win11-24h2-avd:26100.2894.250113  26100.2894.250113
x64             windows-11  MicrosoftWindowsDesktop  win11-24h2-avd  MicrosoftWindowsDesktop:windows-11:win11-24h2-avd:26100.3194.250210  26100.3194.250210
 jmooren@Julians-MacBook-Pro  ~  ✓
```

26100.2033.241004 → October 2024

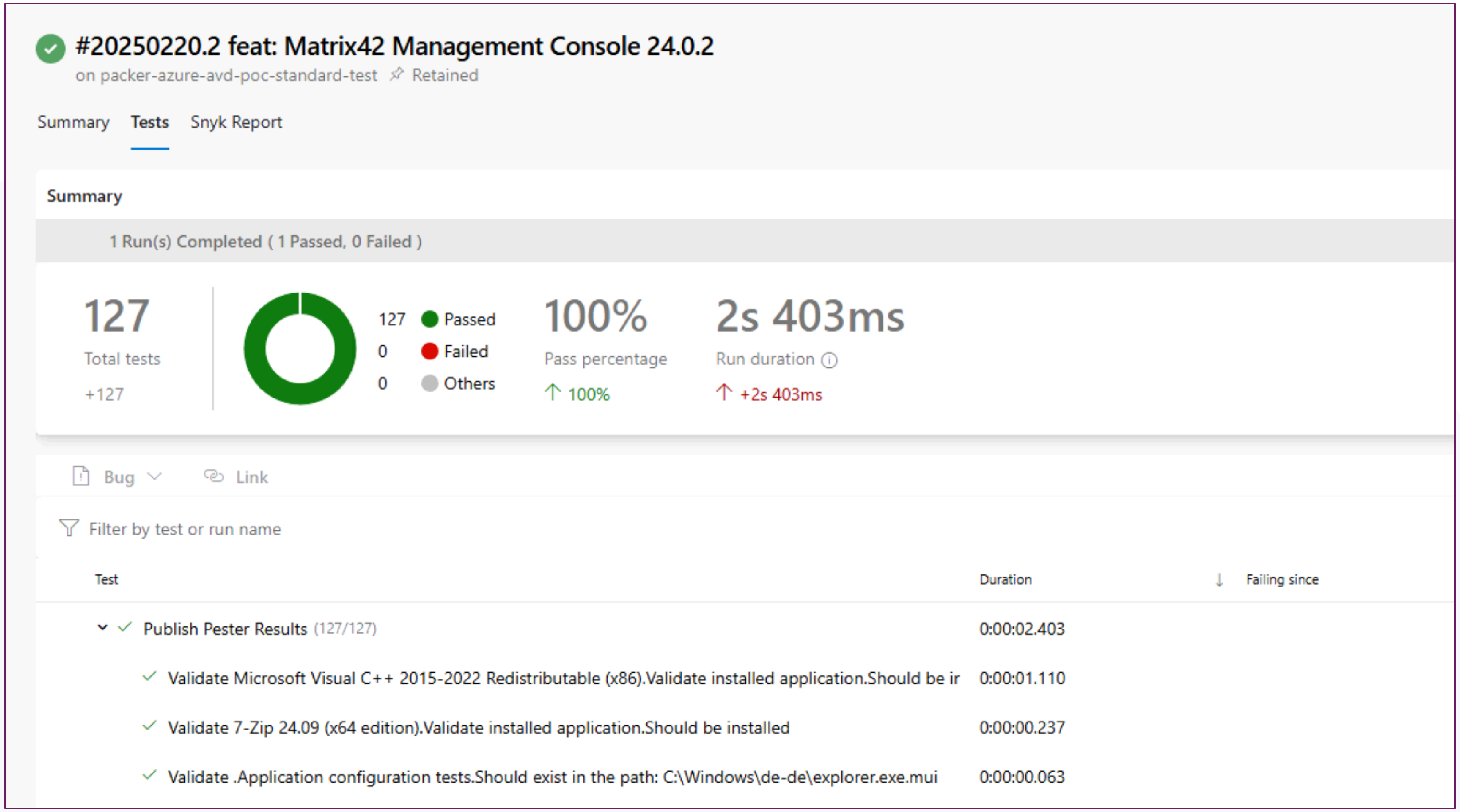
Unit Testing

Unit Tests

Unit Tests for checking Image Qualtity

Credits: Aaron Parker

- **Verify** Image before shipping it to the Dev/Test Stage 🙇
- Based on PowerShell **Pester** Framework https://pester.dev/
- Test runs inside Master VM
- **NUnitXml** Export



✓ #20250220.2 feat: Matrix42 Management Console 24.0.2

on packer-azure-avd-poc-standard-test Retained

Summary

1 Run(s) Completed (1 Passed, 0 Failed)

127 Total tests +127

127 Passed
0 Failed
0 Others

100% Pass percentage ↑ 100%

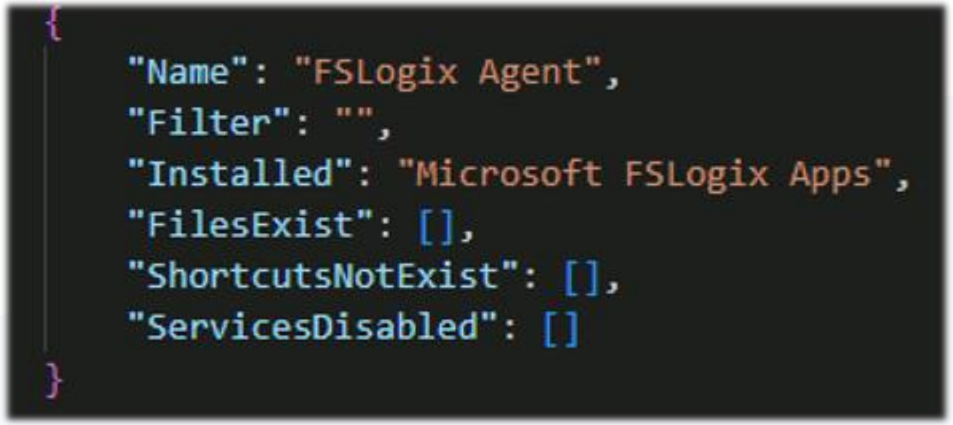
2s 403ms Run duration ↑ +2s 403ms

Test	Duration	Failing since
✓ Publish Pester Results (127/127)	0:00:02.403	
✓ Validate Microsoft Visual C++ 2015-2022 Redistributable (x86).Validate installed application.Should be ir	0:00:01.110	
✓ Validate 7-Zip 24.09 (x64 edition).Validate installed application.Should be installed	0:00:00.237	
✓ Validate .Application configuration tests.Should exist in the path: C:\Windows\de-de\explorer.exe.mui	0:00:00.063	

Unit Test

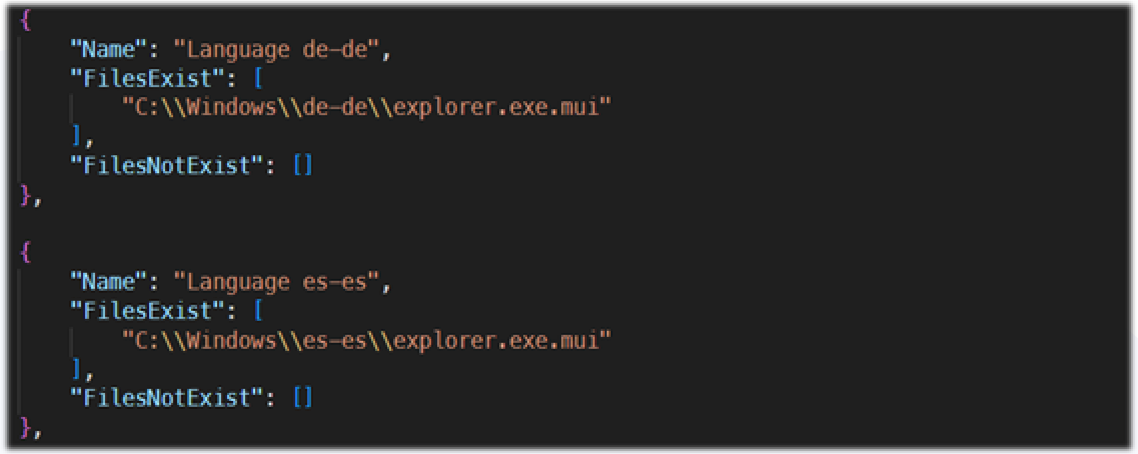
Unit Tests for checking Image Qualtity

Application Testing (Apps.json)

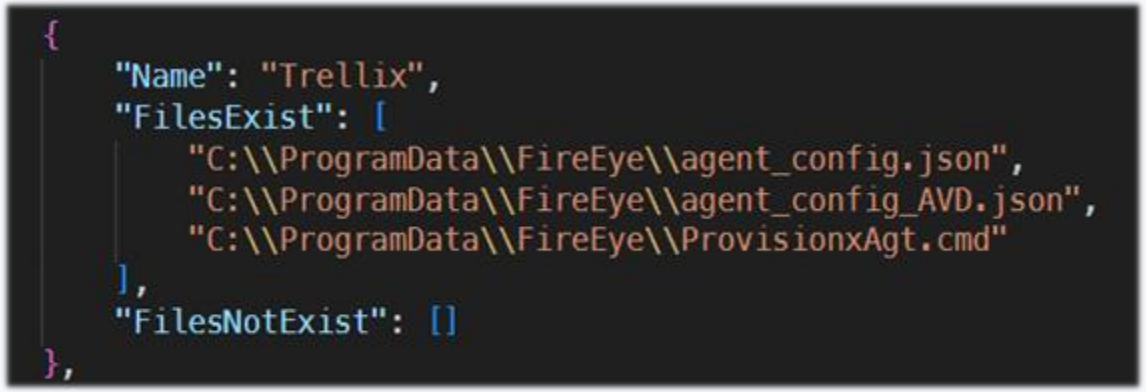


```
{
    "Name": "FSLogix Agent",
    "Filter": "",
    "Installed": "Microsoft FSLogix Apps",
    "FilesExist": [],
    "ShortcutsNotExist": [],
    "ServicesDisabled": []
}
```

File Testing (Files.json)



```
{
    "Name": "Language de-de",
    "FilesExist": [
        "C:\\Windows\\de-de\\explorer.exe.mui"
    ],
    "FilesNotExist": []
},

{
    "Name": "Language es-es",
    "FilesExist": [
        "C:\\Windows\\es-es\\explorer.exe.mui"
    ],
    "FilesNotExist": []
},
```

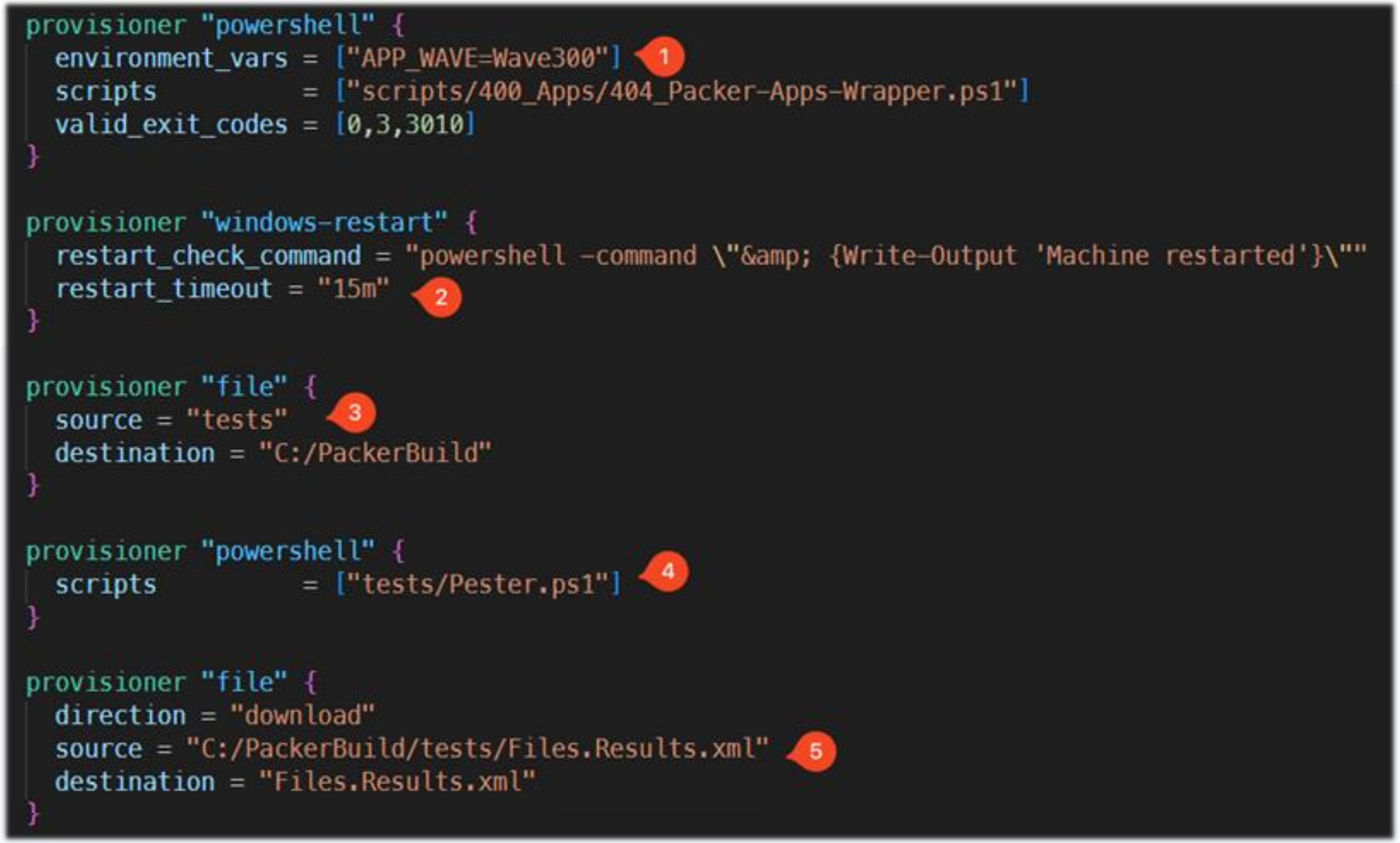


```
{
    "Name": "Trellix",
    "FilesExist": [
        "C:\\ProgramData\\FireEye\\agent_config.json",
        "C:\\ProgramData\\FireEye\\agent_config_AVD.json",
        "C:\\ProgramData\\FireEye\\ProvisionxAgt.cmd"
    ],
    "FilesNotExist": []
},
```

Unit Tests

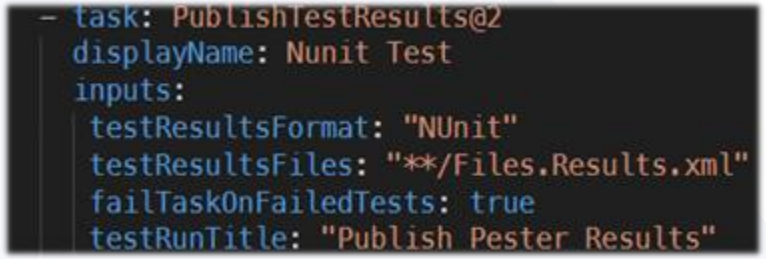
Unit Tests for checking Image Qualtity

Packer Template Snippet



```
provisioner "powershell" {
  environment_vars = ["APP_WAVE=Wave300"]
  scripts          = ["scripts/400_Apps/404_Packer-Apps-Wrapper.ps1"]
  valid_exit_codes = [0,3,3010]
}

provisioner "windows-restart" {
  restart_check_command = "powershell -command \"&amp; {Write-Output 'Machine restarted'}\""
  restart_timeout = "15m"
}

provisioner "file" {
  source = "tests"
  destination = "C:/PackerBuild"
}

provisioner "powershell" {
  scripts          = ["tests/Pester.ps1"]
}

provisioner "file" {
  direction = "download"
  source = "C:/PackerBuild/tests/Files.Results.xml"
  destination = "Files.Results.xml"
}
```

Pipeline Snippet

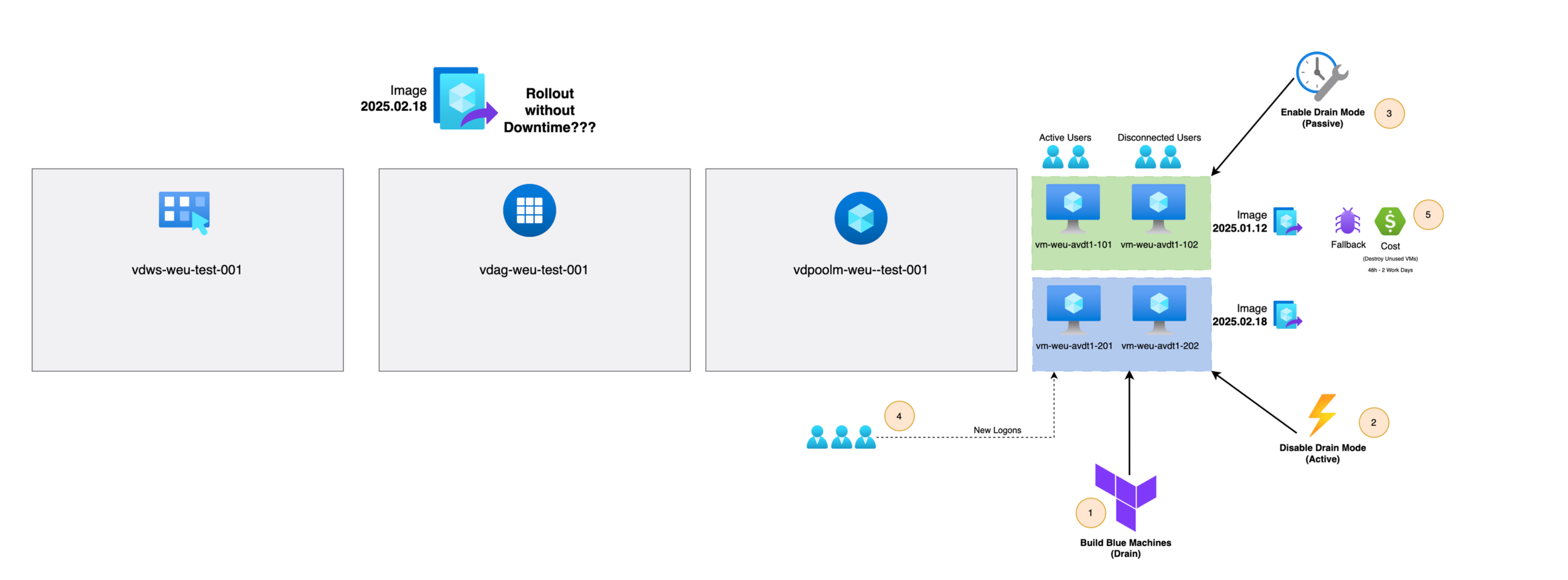


```
- task: PublishTestResults@2
  displayName: Nunit Test
  inputs:
    testResultsFormat: "NUnit"
    testResultsFiles: "**/Files.Results.xml"
    failTaskOnFailedTests: true
    testRunTitle: "Publish Pester Results"
```

Hostpool Image Update

Hostpool Image Update

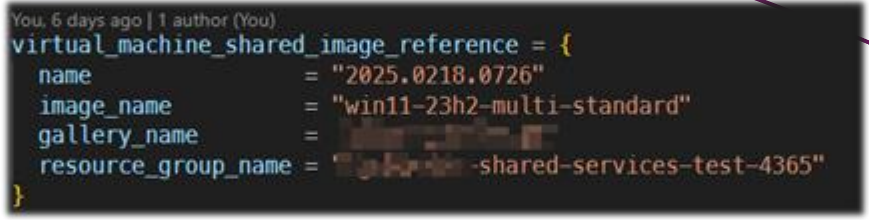
Release Strategy



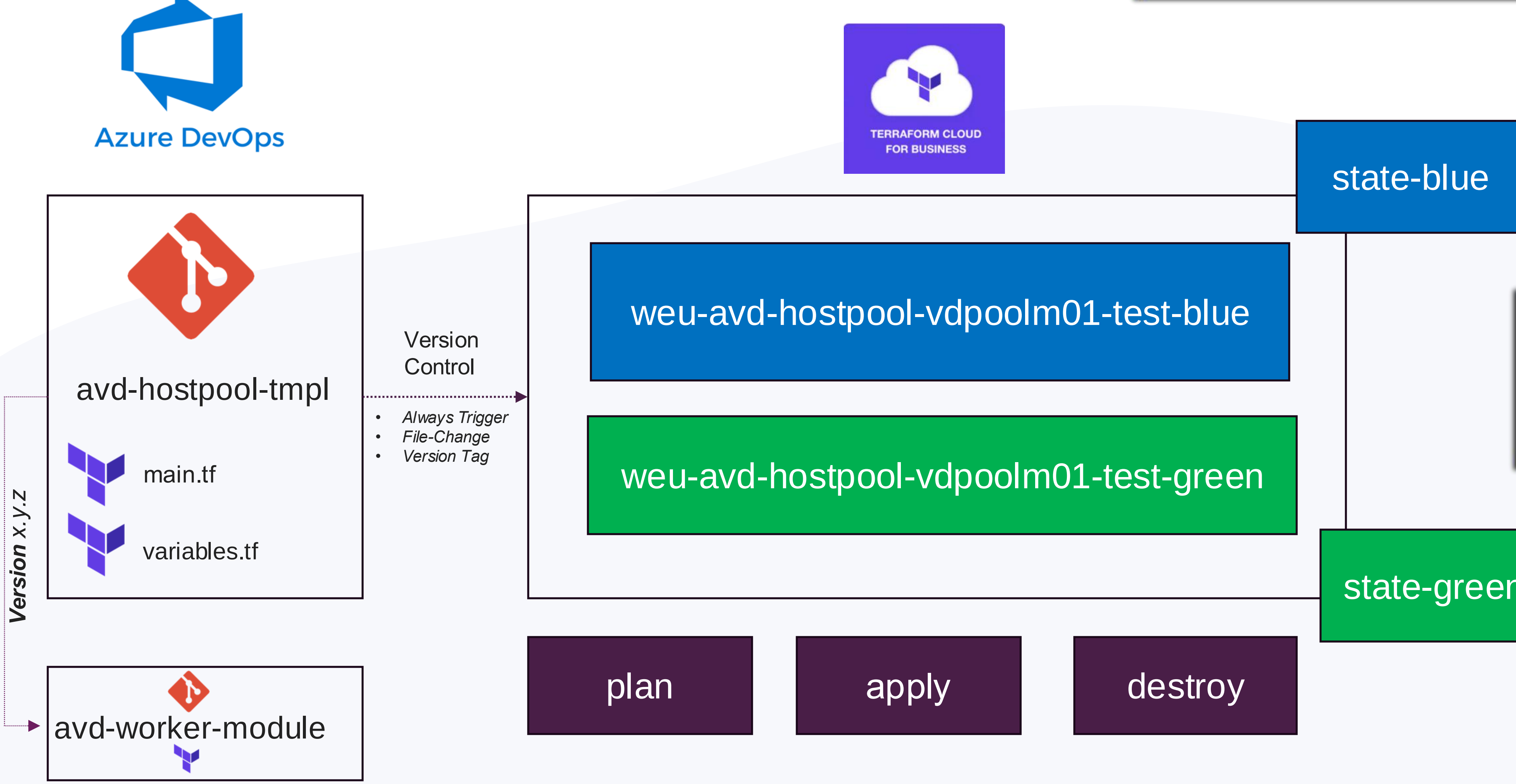
Could be done over the Portal. Goal: **End-to-End Automation** 🤖

Hostpool Image Update

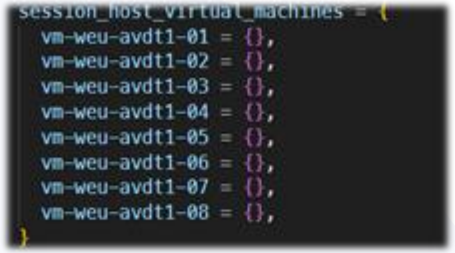
Terraform Workspace Design



```
You, 6 days ago | 1 author (You)
virtual_machine_shared_image_reference = {
  name                = "2025.0218.0726"
  image_name          = "win11-23h2-multi-standard"
  gallery_name        = [redacted]
  resource_group_name = "[redacted]-shared-services-test-4365"
}
```



settings-blue.tfvars



```
session_host_virtual_machines = {
  vm-weu-avdt1-01 = {},
  vm-weu-avdt1-02 = {},
  vm-weu-avdt1-03 = {},
  vm-weu-avdt1-04 = {},
  vm-weu-avdt1-05 = {},
  vm-weu-avdt1-06 = {},
  vm-weu-avdt1-07 = {},
  vm-weu-avdt1-08 = {},
}
```

settings-green.tfvars

CI/CD Integration

CI/CD Integration

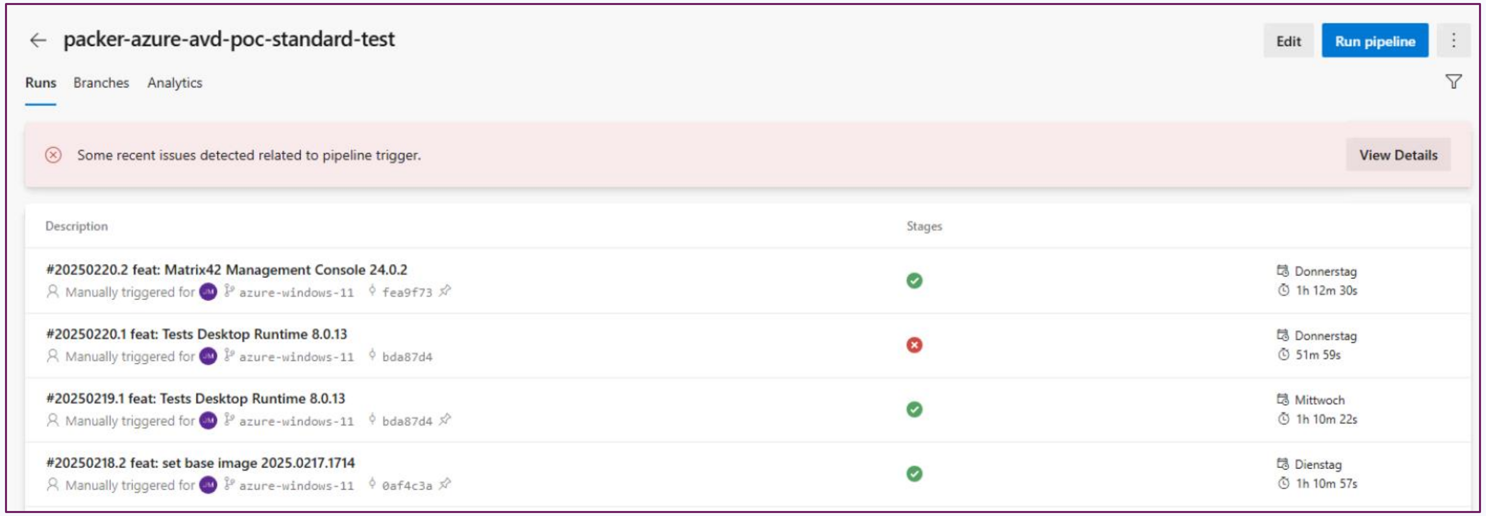
The build process is executed via DevOps Pipeline:

- packer-azure-avd-poc-**base**-test (every month → Patch Tuesday)
- packer-azure-avd-poc-**standard**-test

✅ The pipeline can run multiple times **in parallel** → Dynamic Master VMs, **needs cleanup** in Active Directoy

⚠️ If an error occurs or the pipeline is manually stopped, **orphaned resources** may remain in Azure.

🛠️ Packer builds should be executed in a **dedicated resource group** for easier cleanup.

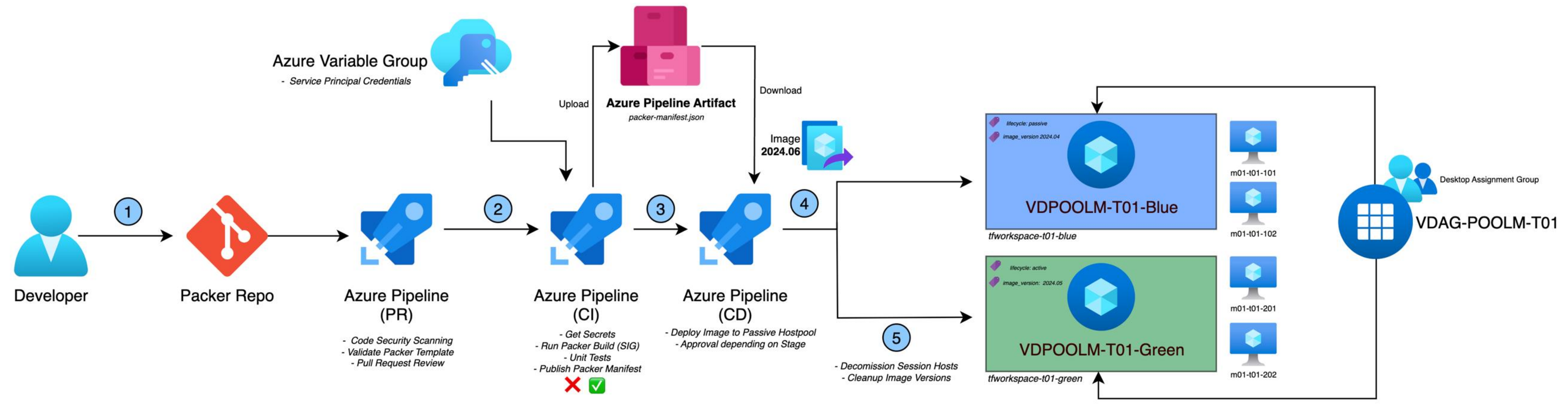


Some recent issues detected related to pipeline trigger.

Description	Stages	
#20250220.2 feat: Matrix42 Management Console 24.0.2 Manually triggered for azure-windows-11 fea9f73	✓	Donnerstag 1h 12m 30s
#20250220.1 feat: Tests Desktop Runtime 8.0.13 Manually triggered for azure-windows-11 bda87d4	✗	Donnerstag 51m 59s
#20250219.1 feat: Tests Desktop Runtime 8.0.13 Manually triggered for azure-windows-11 bda87d4	✓	Mittwoch 1h 10m 22s
#20250218.2 feat: set base image 2025.0217.1714 Manually triggered for azure-windows-11 0af4c3a	✓	Dienstag 1h 10m 57s

CI/CD Integration

Deloyment Process Overview (Image Release)



Azure Variable Group
- Service Principal Credentials

Upload

Azure Pipeline Artifact
packer-manifest.json

Download

Image
2024.06

1 Developer → Packer Repo

Azure Pipeline (PR)
- Code Security Scanning
- Validate Packer Template
- Pull Request Review

2 Azure Pipeline (CI)
- Get Secrets
- Run Packer Build (SIG)
- Unit Tests
- Publish Packer Manifest

❌ ✅

3 Azure Pipeline (CD)
- Deploy Image to Passive Hostpool
- Approval depending on Stage

4

5
- Decomission Session Hosts
- Cleanup Image Versions

lifecycle: passive
image_version 2024.04
VDPOOLM-T01-Blue
tfworkspace-t01-blue
m01-t01-101
m01-t01-102

lifecycle: active
image_version: 2024.05
VDPOOLM-T01-Green
tfworkspace-t01-green
m01-t01-201
m01-t01-202

Desktop Assignment Group
VDAG-POOLM-T01

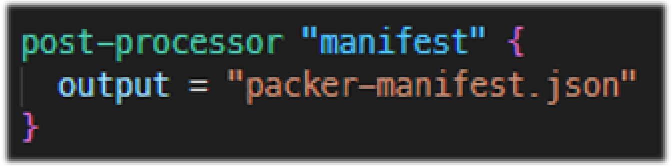
CI/CD Integration

Artifacts

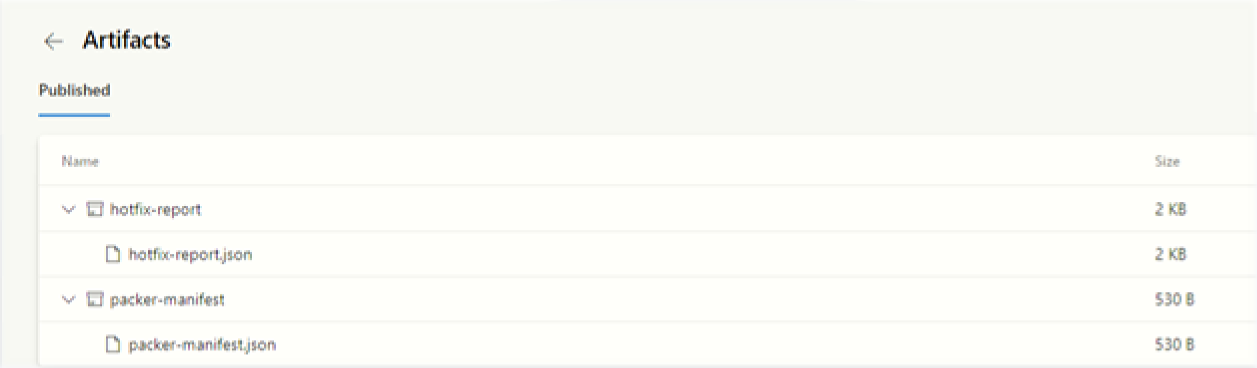
Pipeline artifacts provide the following information:

- hotfix-report.json
- packer-manifest.json

template.hcl

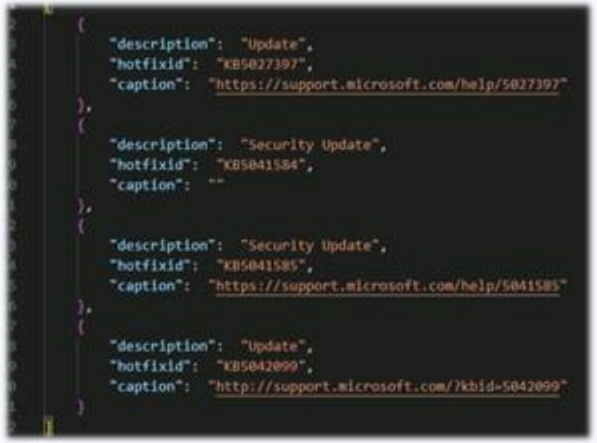


```
post-processor "manifest" {
  output = "packer-manifest.json"
}
```



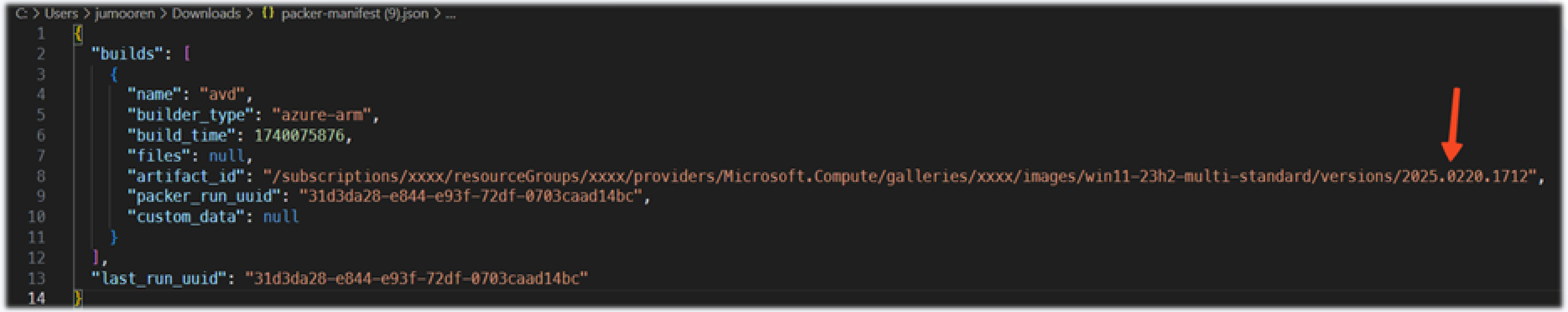
Name	Size
hotfix-report	2 KB
hotfix-report.json	2 KB
packer-manifest	530 B
packer-manifest.json	530 B

hotfix-report.json



```
[
    {
        "description":  "Update",
        "hotfixid":  "KB5027397",
        "caption":  "https://support.microsoft.com/help/5027397"
    },
    {
        "description":  "Security Update",
        "hotfixid":  "KB5041584",
        "caption":  ""
    },
    {
        "description":  "Security Update",
        "hotfixid":  "KB5041585",
        "caption":  "https://support.microsoft.com/help/5041585"
    },
    {
        "description":  "Update",
        "hotfixid":  "KB5042099",
        "caption":  "http://support.microsoft.com/?kbid=5042099"
    }
]
```

packer-manifest.json

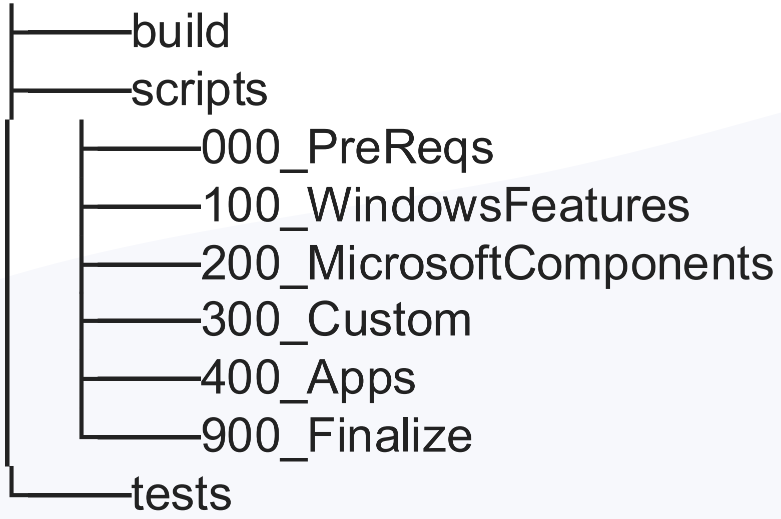


C: > Users > jumooren > Downloads > {} packer-manifest (9).json > ...

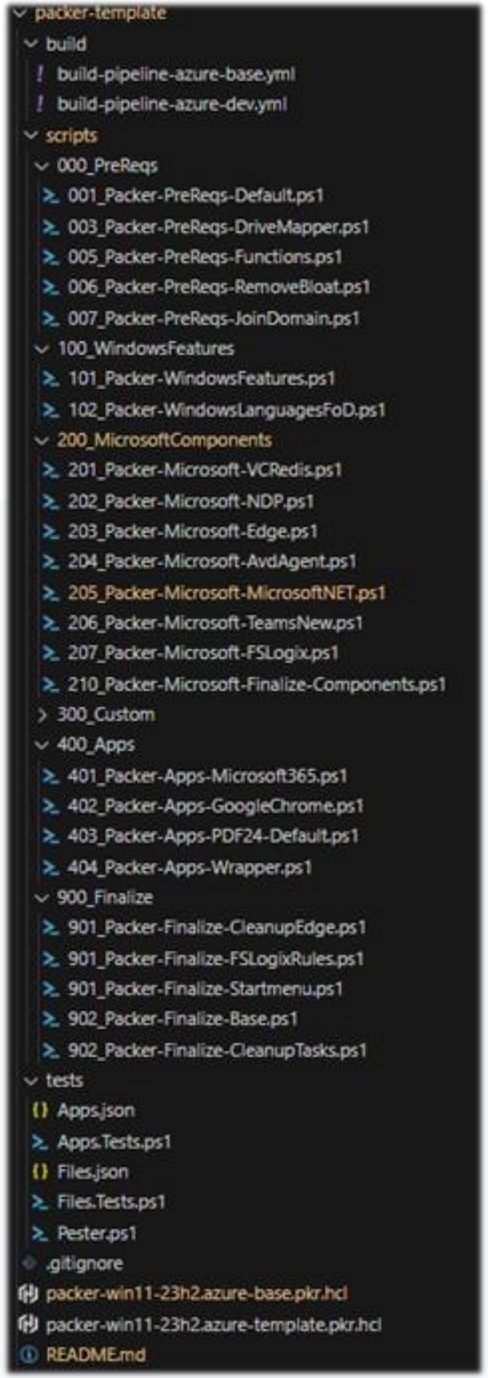
```
{
  "builds": [
    {
      "name": "avd",
      "builder_type": "azure-arm",
      "build_time": 1740075876,
      "files": null,
      "artifact_id": "/subscriptions/xxxx/resourceGroups/xxxx/providers/Microsoft.Compute/galleries/xxxx/images/win11-23h2-multi-standard/versions/2025.0220.1712",
      "packer_run_uuid": "31d3da28-e844-e93f-72df-0703caad14bc",
      "custom_data": null
    }
  ],
  "last_run_uuid": "31d3da28-e844-e93f-72df-0703caad14bc"
}
```

Packer Repo Structure

Repo Structure



```
├───build
├───scripts
│   ├───000_PreReqs
│   ├───100_WindowsFeatures
│   ├───200_MicrosoftComponents
│   ├───300_Custom
│   ├───400_Apps
│   └───900_Finalize
└───tests
```



```
packer-template
  build
    build-pipeline-azure-base.yml
    build-pipeline-azure-dev.yml
  scripts
    000_PreReqs
      001_Packer-PreReqs-Default.ps1
      003_Packer-PreReqs-DriveMapper.ps1
      005_Packer-PreReqs-Functions.ps1
      006_Packer-PreReqs-RemoveBloat.ps1
      007_Packer-PreReqs-JoinDomain.ps1
    100_WindowsFeatures
      101_Packer-WindowsFeatures.ps1
      102_Packer-WindowsLanguagesFoD.ps1
    200_MicrosoftComponents
      201_Packer-Microsoft-VCRedis.ps1
      202_Packer-Microsoft-NDP.ps1
      203_Packer-Microsoft-Edge.ps1
      204_Packer-Microsoft-AvdAgent.ps1
      205_Packer-Microsoft-MicrosoftNET.ps1
      206_Packer-Microsoft-TeamsNew.ps1
      207_Packer-Microsoft-FSLogix.ps1
      210_Packer-Microsoft-Finalize-Components.ps1
    300_Custom
    400_Apps
      401_Packer-Apps-Microsoft365.ps1
      402_Packer-Apps-GoogleChrome.ps1
      403_Packer-Apps-PDF24-Default.ps1
      404_Packer-Apps-Wrapper.ps1
    900_Finalize
      901_Packer-Finalize-CleanupEdge.ps1
      901_Packer-Finalize-FSLogixRules.ps1
      901_Packer-Finalize-Startmenu.ps1
      902_Packer-Finalize-Base.ps1
      902_Packer-Finalize-CleanupTasks.ps1
  tests
    Apps.json
    Apps.Tests.ps1
    Files.json
    Files.Tests.ps1
    Pester.ps1
  .gitignore
  packer-win11-23h2.azure-base.pkr.hcl
  packer-win11-23h2.azure-template.pkr.hcl
  README.md
```

Questions?

Ask me Anything 🤓