Automated Windows OS Image Creation

(using some open source tools...and stuff)*

*I take no credit for the creation of these tools...only there somewhat novel use[1]

[1] I also take only partial credit for the novel use of said tools...and even that is very minimal. Seriously. I came across some tools on Github and started using them...just so we're clear.

Shoutouts and Acknowledgements

Without the work of these individuals, this presentation would not be possible.

Matt Wrock - Creator of Boxstarter, also creator of some great Packer templates

Matthew Hodgkins - For his excellent Packer templates and Cake build script

Taliesin Sisson - Packer templates and PowerShell scripts

What is this rubbish?

THAT IS AN EXCELLENT QUESTION

In a few words…

This presentation is **not** about Microsoft tools such as:

MDT (Microsoft Deployment Toolkit)

SCCM (System Center Configuration Manager)

InTune

But first, let’s do a history review!

Some background c. 1998-ish

The ~~really old~~ classic way of creating a Microsoft Windows OS image.

1. Prepare a bare-metal workstation (set BIOS settings, prepare boot media, etc)
2. Install Microsoft Windows to a physical hard disk drive
3. Install device drivers
4. Install Windows Updates
5. Install application software
6. Install more Windows updates, possibly one or more “Service Packs”
7. Customize Default Profile
8. Clean up the operating system (i.e. delete administrator profile, downloads, etc)
9. Run Sysprep utility
10. Capture Disk Image

Some +kinda old background c. 2008-ish

The ~~-really old~~ +newer classic way of creating a Microsoft Windows OS image.

~~1. -Prepare a bare-metal workstation...~~ +Create a Virtual Machine (mount ISO)
2. Install Windows to a ~~-physical~~ +virtual hard disk drive
3. Install device drivers...maybe?
4. Install Windows Updates
5. Install application software
6. Install more Windows updates, hopefully “Service Packs” are no longer a thing
7. Customize Default Profile
8. Clean up the operating system (i.e. delete administrator profile, downloads, etc)
9. Run Sysprep utility
10. Capture Disk Image +OR convert to portable format such as .WIM

See the problem?

THAT BELONGS IN A MUSEUM

Is there a better way?

INDEED.

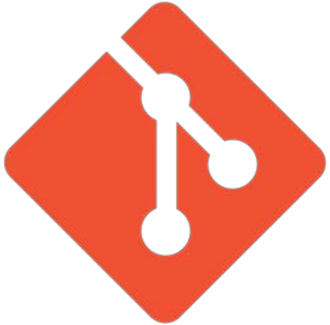
So then how?

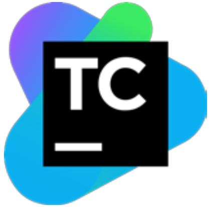
Using these tools which you may have heard of...or not.

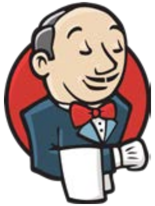
HashiCorp
Packer
Microsoft
Hyper-V

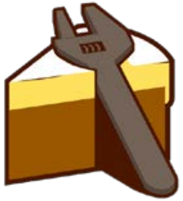
also some optional tools...

Chocolatey



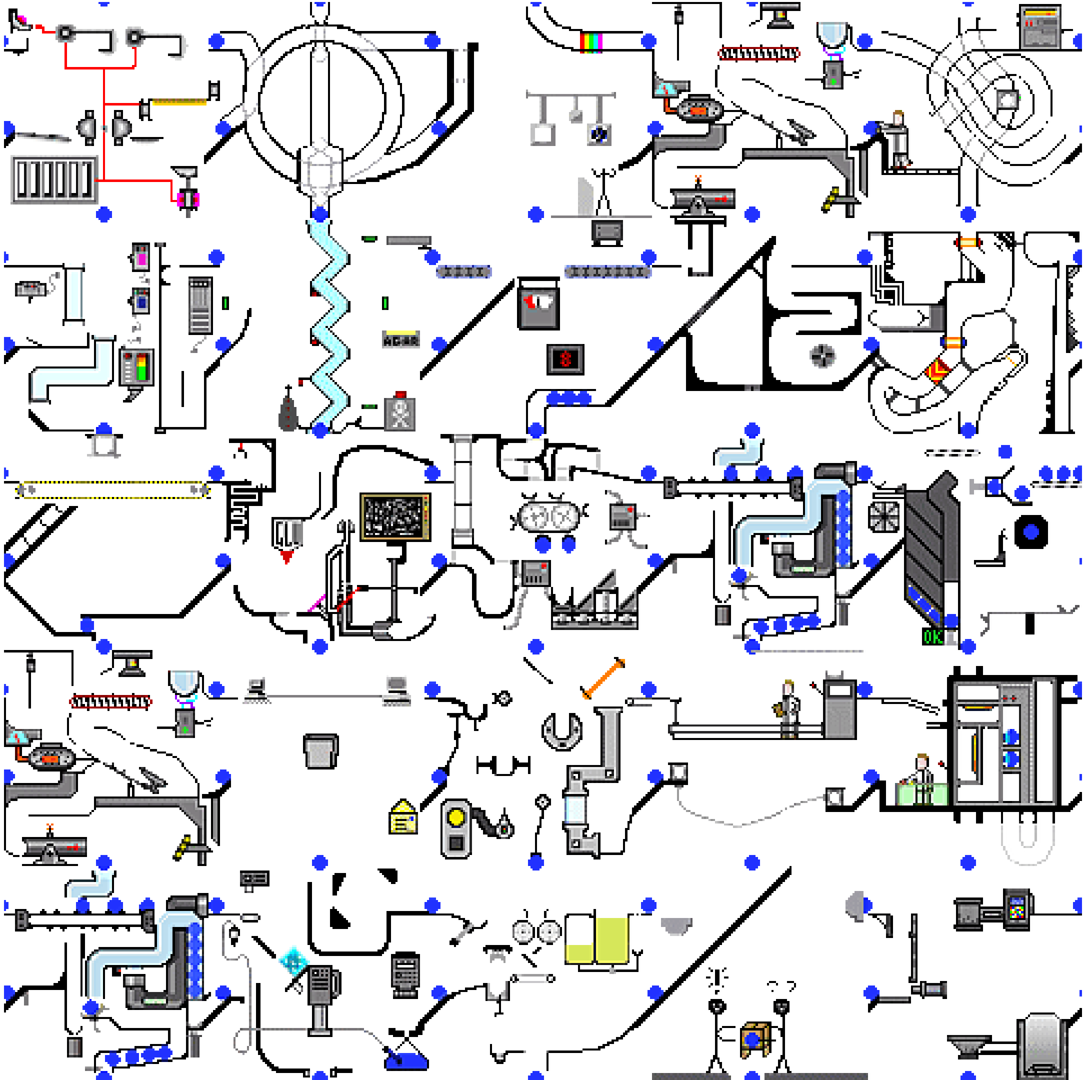






puppet
labs

=


OK

But why? Looks unnecessarily complex.

Let’s talk about Packer.

Packer is an open source tool for **creating identical machine images** for multiple platforms from a single source configuration.

Packer is lightweight, runs on every major operating system, and is highly performant, **creating machine images for multiple platforms in parallel**.

Packer **does not replace configuration management** like Chef or Puppet.

In fact, when building images, **Packer is able to use tools like Chef or Puppet** to install software onto the image.

A little info about Vagrant...

- Vagrant (also made by Hashicorp), is a tool for building and managing virtual machine environments.
- Vagrant uses “boxes” which are essentially just fancy zip archives containing a virtual machine configuration files and virtual hard disks.
- Vagrant allows simple creation of new VMs with a few simple commands.
- Packer allows you to create vagrant boxes for a variety of hypervisor environments.

Although the focus of this talk is not on Vagrant as a tool, it is part of the toolchain for building operating system images (and not just Windows OS images).

To use Vagrant to create a new VM from a “.box” template:

```
vagrant init hashicorp/windows10ent
vagrant up
```

Or, if you don’t want to use the public feed:

```
vagrant init box-name
https://source.tld/box-name.box
vagrant up
```

Source: https://www.vagrantup.com/intro/index.html

So then how does this work?

With a build template.

STUPIDGIFS.COM
H HD
HISTORY.COM

What does a “build template” look like?

Assign Variables

```
{
  "variables": {
    "vm_name": "Example-VM",
    "cpu": "4",
    "ram_size": "8192",
    "disk_size": "25000",
    "iso_url": "https://someurl.com/whereyouhostyour/Windows.iso",
    "Iso_checksum_type": "sha1",
    "iso_checksum": "acf2fce90085d0fff6e0354aaf0e15973ee04a44",
    "username" : "Administrator",
    "password" : "SomeReallyStrongPassword!@#$%^&*()"
  },
```

Specify Builder(s)

```
"builders": [
{
  "vm_name":"{{user `vm_name`}}",
  "type": "hyperv",
  "iso_url": "{{user `iso_url`}}",
  "iso_checksum_type": "{{user `iso_checksum_type`}}",
  "iso_checksum": "{{user `iso_checksum`}}",
  "Communicator":"winrm",
  "winrm_username": "{{user `username`}}",
  "winrm_password": "{{user `password`}}",
  "winrm_timeout" : "4h"
  "shutdown_command": "C:/VCU-Deploy/Packer/PackerShutdown.bat",
}],
```

Specify Provisioner(s)

```
"provisioners": [{
  "type": "powershell",
  "elevated_user": "{{user `username`}}",
  "elevated_password": "{{user `password`}}",
  "scripts": [
    "./scripts/script.ps1",
  ]
}],
{
  "type": "windows-restart",
  "restart_timeout": "2h"
},
```

Specify Post-Processor(s)

```
"post-processors": [
  {
    "type": "vagrant",
    "keep_input_artifact": true,
    "output": "/{{.Provider}}_{{ user `vm_name` }}.box"
  }
]
```

So what does this look like in action?

Prerequisites

At a bare minimum you need these two items:

- Packer (choco install packer -y)
- A hypervisor. Although our focus is on Hyper-V today, you can use:
  - Virtualbox (choco install virtualbox -y)
  - VMware (vSphere/ESXi)
  - QEMU
  - Parallels
  - Amazon EC2
  - Azure
  - etc, etc, etc...the list goes on and include cloud providers as well as community-built builders

Source: https://www.packer.io/docs/builders/

Very-Simple-Packer-Build-Image.ps1

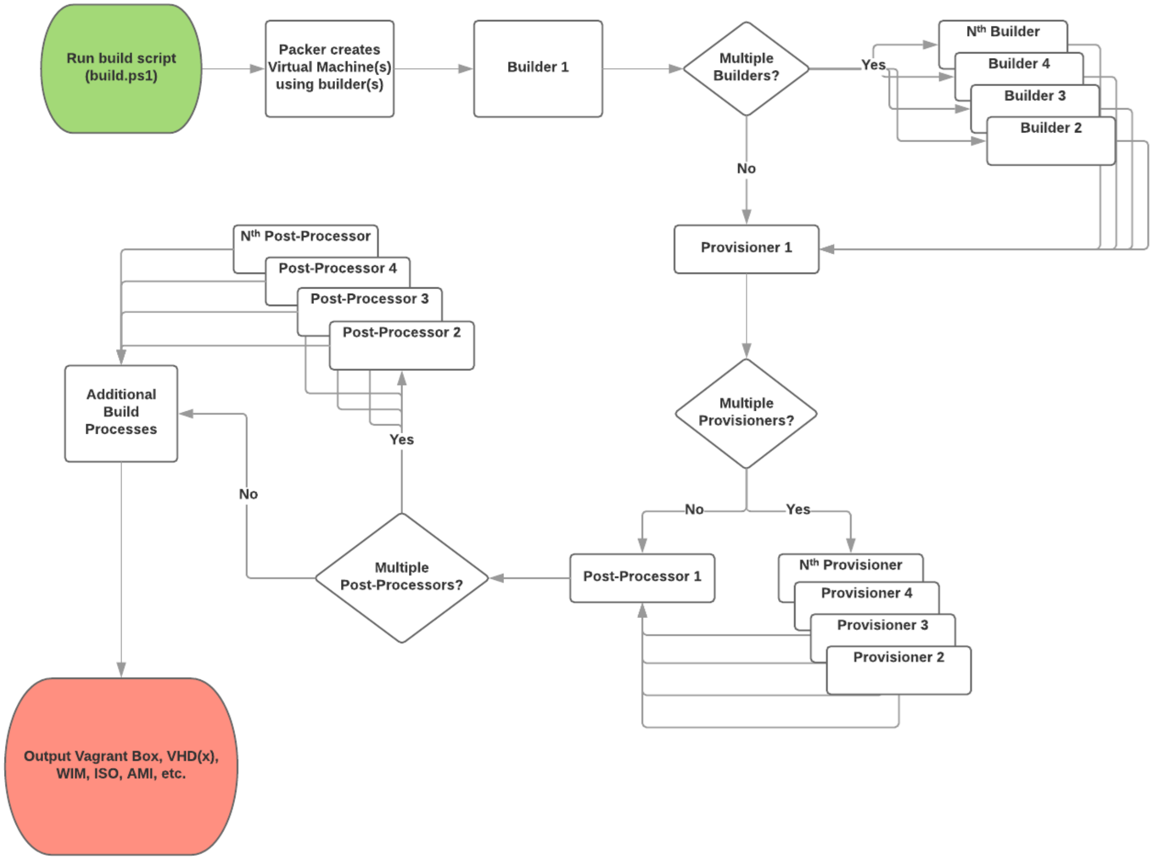
```
#ensure chocolatey is installed
if(!($env:ChocolateyInstall) -or !(Test-Path "$env:ChocolateyInstall")){
    iex ((new-object net.webclient).DownloadString("http://chocolatey.org/install.ps1"))
}
#force reinstall Packer
choco install packer -y -f

#check for Hyper-V, if enabled run the build
if($hyperv.State -eq "Enabled"){
     packer.exe build .\windows10.json --provider=hyperv
}
Else {Write-Output “You need to enable Hyper-V first!”}
```

DEMO TIME

Do you have a flowchart?

PACKER BUILD PROCESS



So what about a bare-metal image?

Converting a VHD to a WIM using PowerShell

Once a VM has been exported to a Vagrant box from Packer, we use the remaining .VHD(x) to convert to a .WIM automatically using a PowerShell script something like this :

```
#first we specify some variables about paths to things
$mount = "F:\Mount"
$wimPath = "F:\WIM"
$wimFile   = "F:\WIM\baseimage.wim"
$vhd   = "F:\Provisioning\Packer\Packer-Templates\OUTPUT-(LCC-win10L-Ops)-base\Virtual Hard Disks\LCC-win10L-Ops-
base.vhdx"
$imageName = "Base-Windows-10-Image"
#ensure our mount path and wim path both exist
if (!(Test-Path $mount)){New-Item -Type Directory -Path $mount -Force}
if (test-path $wimPath){
    Remove-Item "F:\WIM" -Force
    New-Item -Type Directory -Path $wimPath -Force
}
else {New-Item -Type Directory -Path $wimPath -Force}
#next we mount the VHD file as a Windows Image to our temporary mount folder and capture to a WIM file
Mount-WindowsImage -ImagePath $vhd -Path $mount -Index 1
New-WindowsImage -CapturePath $mount -Name $imageName -ImagePath $wimFile -Description $imageName -Verify
Dismount-WindowsImage -Path $mount -Discard
#delete temp mount path
Remove-Item $mount -force
```

Putting it all together (building the cake)

Q&A

THANK YOU.