www.devopsgroup.com | Phone: 0800 368 7378 | e-mail: team@devopsgroup.com | 2019

PSMeetup Lightning Talk

Powershell: What you may have missed!

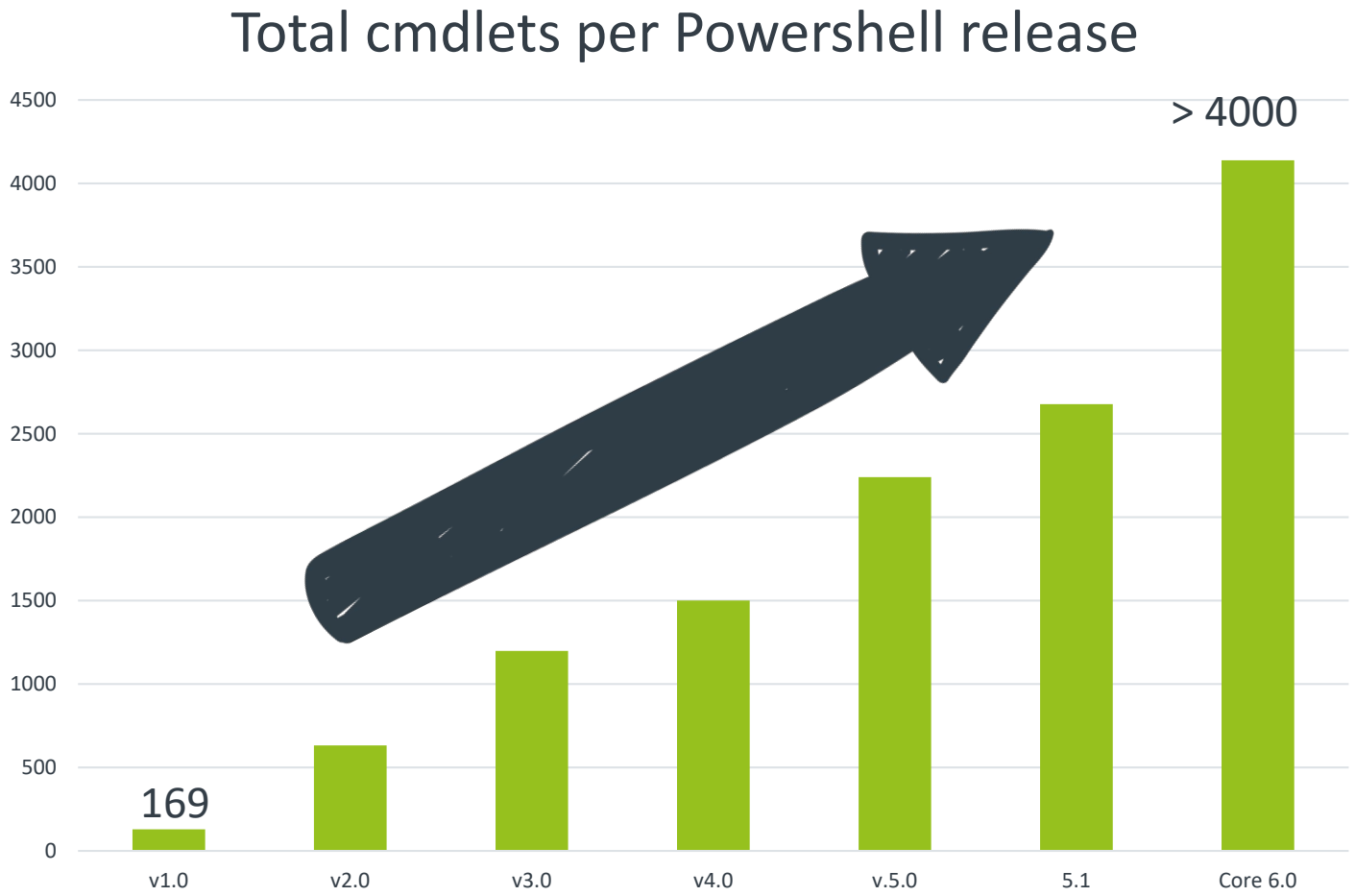
Ed Dipple

Lead CloudOps Engineer

Powershell and Me

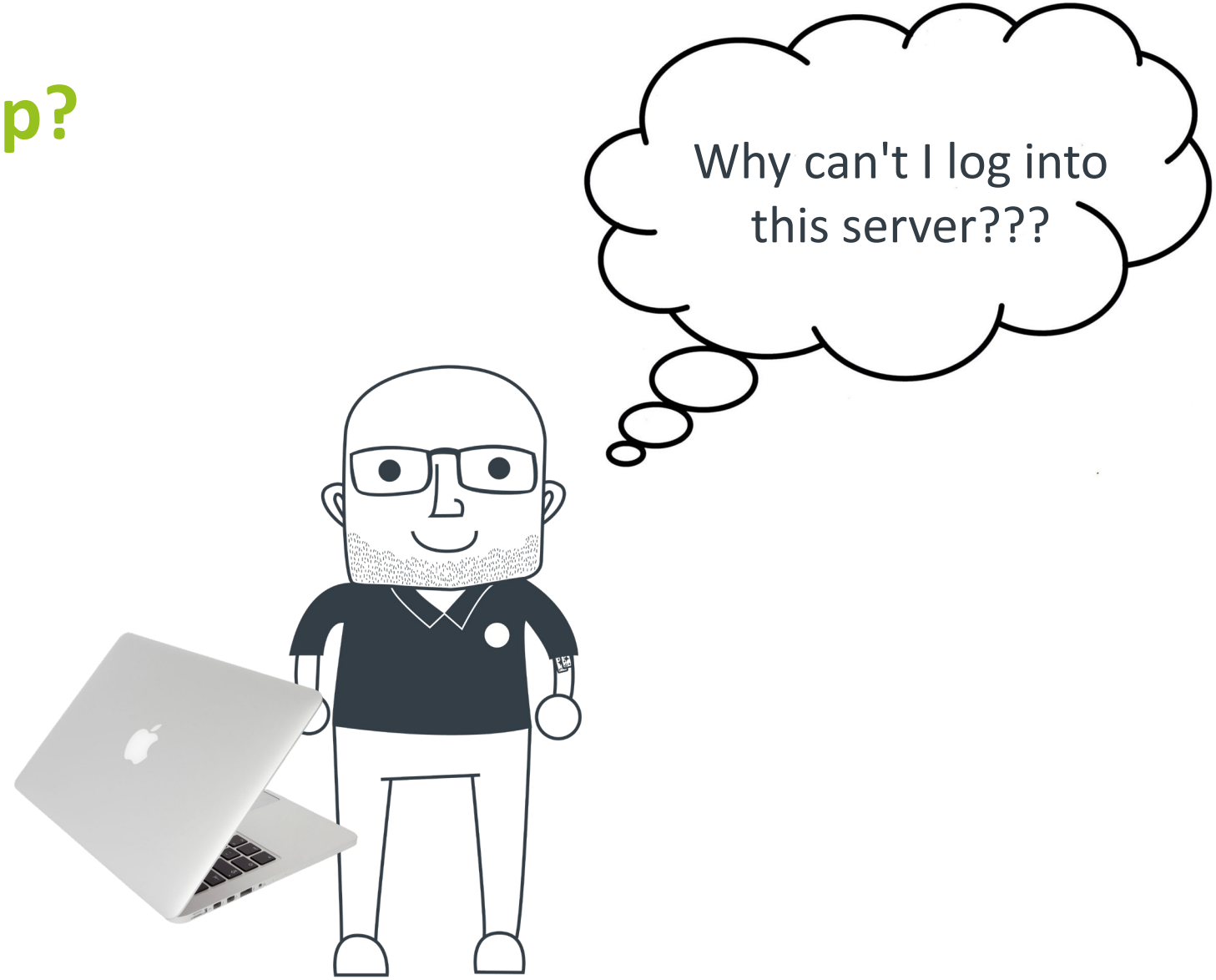
- My day job: Windows & Linux migrations to Azure and AWS
- Configuration Management (Powershell DSC)
- Querying Azure
- Azure Automation
- Avoiding using Windows Explorer
- Constantly switching between MacOS and Windows

Let's start with a graph!



It's difficult to keep up!

Is a server up?



Is a server up?

- The old school approach....

```
ping myhost.local
```

```
telnet myhost.local 3389
```

What do you get ?

```
'telnet' is not recognized as an internal or external
command, operable program or batch file
```

OR

```
telnet: unable to connect to remote host: Connection refused
```

OR

```

```

Can I remember the key combo to escape this?

Powershell to the rescue!

Host is up

Port is down

```
Test-Netconnection myhost.local –Port 3389


ComputerName            :  myhost.local
RemoteAddress           :  1.2.3.4
InterfaceAlias          :  Ethernet
SourceAddress           :  10.211.55.8
PingSucceeded           :  True
PingReplyDetails (RTT) :  29ms
TcpTestSucceeded        :  False
```

But wait, there's more!

- **-CommonTCPPort**
  - Can't remember the RDP port? Just write "RDP"
- **-InformationLevel Detailed**
  - Even more information available!
- **-DiagnoseRouting**
  - List all hops made during the connection attempt

Integrate with Pester!

- You can easily write Pester Integration tests that warn you of breaking network changes

```
it 'Fails if the DB isn't responding to the webserver' {
    $PortTest = Test-Netconnection myhost.local –Port 1433
    $PortTest.TCPTestSucceeded | Should Be $True
}
```

Installing a new module used to be a pain

1. Google the module name and download a zip file
2. Try and remember where Powershell expects modules to be installed
3. Manually unzip the module into the right place
4. Repeat for every module update, if you remember to do so

PowershellGet is a breath of fresh air

- Introduced in Powershell 5.0
- Install and update modules from PSGallery or any other custom repository
- All repositories are untrusted by default as a security measure

```
Register-PSRepository MyCustomRepo -SourceLocation repository.local
```

```
Install-Module MyCustomModule
```

```
Update-Module  MyCustomModule
```

There's a new Azure module in town!

- Microsoft recommend you uninstall the old Azure cmdlets and use the new one – called "Az"
- Az is cross platform (Windows/Linux/MacOS), as it uses the .NET standard library
- All functionality in AzureRM is present in Az
  - You can use the Enable-AzureRMAlias cmdlet if you prefer the old cmdlet names
  - Otherwise, most commands are just the old commands with Azure shortened to Az
- Az will continue to get support for newly introduced services
- AzureRM will only get bugfixes until 2020, no new service support

- To install the new module

```
Install-Module -Name Az -AllowClobber
```

- Interact with Azure using Az

```
Connect-AzAccount
```

```
Get-AzVM
```

```
New-AzResourceGroup -Name 'myResourceGroup' -Location 'westeurope'
```

Mocking .NET objects in Pester using Classes

- The Pester module can orchestrate Unit Tests for your Powershell code
- Pester has a concept of "mocking" cmdlets and objects during tests to prevent changing system state
  - Downloading a file
  - Running a SQL Query
  - Deleting a user from Active Directory
- A mocked cmdlet will have its functionality replaced (usually with nothing)
- What the cmdlet does instead is in the control of the test writer

Simple Mock examples

```
Mock Test-Path { $True}
```

Now Test-Path always returns True

```
Mock Test-Path { $True}
–ParameterFilter { $Path -eq "C:\config.json" }
```

Now Test-Path only returns True for a specific file

Prevent a file download

```
$wc = (New-Object System.Net.WebClient)

$wc.DownloadFile($url, $path)

# How do you make the test pass without an active
internet connection?
```

Use Powershell Classes!

- Introduced native Classes in Powershell 5
- It's possible to use Classes in Powershell 4 using inline C# code
- Classes allow you to define a custom object type
- For a complex data structures, use a Class rather than an array of Hash Tables
- In this example, the solution is to mock New-Object to produce a fake WebClient object with a fake DownloadFile function

Working example (Powershell 5)

```
Class FakeWebClient {
  DownloadFile($arg1, $arg2) { #No code in here }
}

$fakeWebClient = New-Object FakeWebClient

Mock New-Object { $fakeWebClient }
  -ParameterFilter { $TypeName –eq "System.Net.WebClient" }

New-Object System.New.WebClient # Returns the fake webclient
```

Working example (Powershell 4)

```
झsrcष हद
public class FakeWebClientv4
ठ
  public void DownloadFile(object arg1, object arg2झ
  ठ #No code here ड
ड                                                          द्ह

Add-Type -TypeDefinition $src
Mock New-Object { New-Object FakeWebClientv4 }
  -ParameterFilter { $TypeName –eq "System.Net.WebClient" }
```

Any Questions?