Malleable C2 Profiles and You

By Michael Haag

Published: 2021-01-07 · Archived: 2026-04-05 21:55:16 UTC

5 min read

Jan 4, 2021

Like most defenders out there today we keep hearing and seeing CobaltStrike used by actors. I wanted to share some notes on some back of the napkin research I've been doing related to it in hopes that it sparks interest in others to begin investigating more.

It wasn't until about 2 years ago I realized how *powerful* CobaltStrike is. If I am this late, I can't imagine how far heavily funded adversaries are in generating their toolsets. I know some more advanced Red Teams have been using these capabilities to their fullest, but many are still using the defaults. Some of this may come as old news, but for me, this was quite profound. I'm talking about Malleable C2 Profiles. A very generic set of these profiles may be found here:

The pieces didn't connect as to what these were until I found

Simple things, like User Agent -

Press enter or click to view image in full size

```
80  ################################################
81  ## Beacon User-Agent
82  ################################################
83  ## Description:
84  ##    User-Agent string used in HTTP requests, CS versions < 4.2 approx 128 max (
85  ## Defaults:
86  ##    useragent: Internet Explorer (Random)
87  ## Guidelines
88  ##    - Use a User-Agent values that fits with your engagement
89  ##    - useragent can only be 128 chars
90  ## IE 10
91  # set useragent "Mozilla/5.0 (compatible; MSIE 10.0; Windows NT 7.0; InfoPath.3;
92  ## MS IE 11 User Agent
93  set useragent "Mozilla/5.0 (Windows NT 6.3; Trident/7.0; rv:11.0) like Gecko";
94
```

Beacon user-agent

all the fields for SMB beacons, DNS beacons — setting the pipes or DNS settings.. 🤔 This is pretty fantastic stuff! We talk about how actors blend in, here is your chance.

and I started to get really curious as to how these are being used. I wont break down each section, but here are the resources to understand what is going on:

In short of what I am getting at, if an actor or red team wants to blend in, using these malleable C2 profiles is how the job will get done. CobaltStrike itself has a lot of options, but I feel the money is on these profiles. Now, I do not have a CobaltStrike teamserver, or own a license. I am only basing my analysis on what I've analyzed and seen in the wild. However, there are cracked versions on the internet easily obtainable (probably backdoored 🤦‍♂️, but obtainable), and @BC-Security1 made the ability to import malleable profiles to Empire (more here).

ThreatExpress has been keeping their profiles up to date with each CS release.

202011 - Add CS 4.2 Reference Profile

- Add latest MalleablePE and MalleableC2 options for Cobalt Strike 4.1 and 4.2
- 4.1 Additions: `tcp_frame_header`, `smb_frame_header`, `ssh_banner`
- 4.2 Additions:
  - **global**
    - `data_jitter`
    - `headers_remove`
    - `ssh_pipename`
  - **postex**
    - `pipename`
    - `thread_hint`
    - `keylogger`
  - **stage**
    - `allocator`
    - `magic_mz_86|magic_mz_64`
    - `magic_pe`

I want to scope this blog on post exploitation.

```
post-ex {
        set spawnto_x86 "%windir%\\syswow64\\<mfpmp>.exe";
        set spawnto_x64 "%windir%\\sysnative\\<mfpmp>.exe";
        set obfuscate "[true|false]";
          # Obfuscate the permissions and content of our post-ex
          DLLs  set smartinject "[true|false]";
          # Directs Beacon to embed key function pointers (ex)
          GetProcAddress, LoadLibrary) into its same-arch post-ex
```

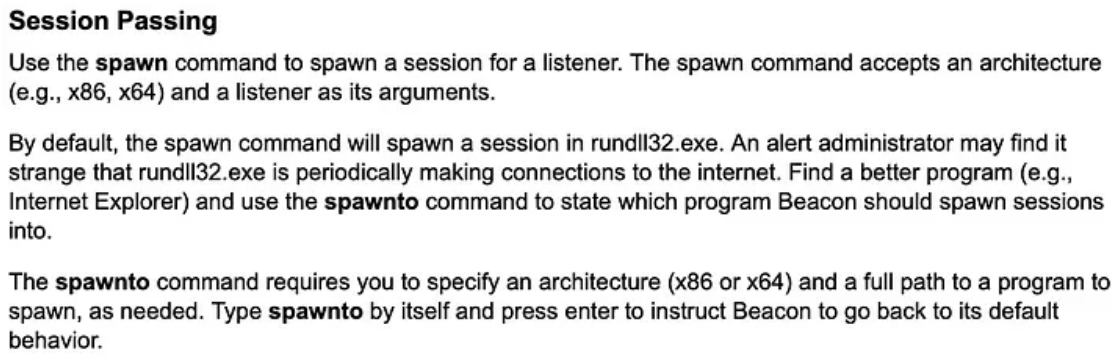
```
        DLLs.

                                                        set amsi_disable "[true|fals
        # Disable AMSI (Antimalware Scan Interface) in powerpick,
        execute-assembly and psinject before loading .NET or PS
        code
}
```

Everything else is obviously very interesting, but for some reason post-ex interests me the most.

What's the point of *spawn_to*?

The idea is, which is great, allows the operator to select what process to spawn beacon into. Upon its use, a process will spawn. In practice, the process will sometimes look out of place by having a non-standard parent process, no command line arguments or maybe a network connection.

From Beacon help:

> Press enter or click to view image in full size
>
> 
> **Session Passing**
>
> Use the **spawn** command to spawn a session for a listener. The spawn command accepts an architecture (e.g., x86, x64) and a listener as its arguments.
>
> By default, the spawn command will spawn a session in rundll32.exe. An alert administrator may find it strange that rundll32.exe is periodically making connections to the internet. Find a better program (e.g., Internet Explorer) and use the **spawnto** command to state which program Beacon should spawn sessions into.
>
> The **spawnto** command requires you to specify an architecture (x86 or x64) and a full path to a program to spawn, as needed. Type **spawnto** by itself and press enter to instruct Beacon to go back to its default behavior.
>
> https://www.cobaltstrike.com/help-beacon

There is a lot of ways to remove OpSec with CobaltStrike — maybe it's not widely known or understood, but Beacon has a lot of options for configuration.

I've seen others on Twitter begin to publicly share scans of CobaltStrike Team Servers. I don't believe it's entirely public how some are gathering these data points 🤷, but I am glad it's happening.

Get Michael Haag's stories in your inbox

Join Medium for free to get updates from this writer.

Remember me for faster sign in

Some great references:

- Awesome-CobaltStrike-Defence — https://github.com/MichaelKoczwara/Awesome-CobaltStrike-Defence

- RiskIQ — https://www.riskiq.com/blog/labs/cobalt-strike/ and https://www.riskiq.com/blog/external-threat-management/jarm-incident-response/
- RecordedFuture — https://www.recordedfuture.com/cobalt-strike-servers/
- SANS — https://isc.sans.edu/diary/Threat+Hunting+with+JARM/26832
- Salesforce — https://engineering.salesforce.com/easily-identify-malicious-servers-on-the-internet-with-jarm-e095edac525a
- 🔥 NMAP NSE — https://github.com/whickey-r7/grab_beacon_config

There you go, lots of options and details there.

Lists released:

All the scanning and public lists lead to this small set of binaries being used to spawn into:

Press enter or click to view image in full size

	spawnto	count
2	WUAUCLT.exe	1
3	WerFault.exe	3
4	batchexe	2
5	cmstp.exe	1
6	compact.exe	1
7	dllhost.exe	3
8	eventvwr.exe	1
9	gpresult.exe	2
10	gpupdate.exe	16
11	iexplore.exe	1
12	lsass.exe	1
13	mavinject.exe	3
14	mstsc.exe	6
15	net.exe	2
16	regsvr32.exe	1
17	rundll32.exe	401
18	svchost.exe	8
19	w32tm.exe	1
20	wusa.exe	2

Note: rundll32.exe is the default.

How can defenders use this information? Why is this important?

We can use the spawn_to values as a jump off point for detection coverage by looking for and asking the following:

- Is it normal for <spawn_to value> to have no command line arguments?
- What is the default/normal process lineage for <spawn_to value>?
- Does the <spawn_to value> make network connections?
- Is it normal for <spawn_to value> to load jscript, vbscript, Amsi.dll, and clr.dll?

Does our EDR/AV product provider visibility to answer these questions? This should be looked into.

Should we go and generate new coverage for the list above? It's a good start, however, I recommend baselining all of system32.

If threat actors of all sorts can get their hands on CobaltStrike, it is in your threat model. Spawn_to is only one of the many things Beacon can do (more here). We have only scratched the surface. I only hope that as organizations buy and use CobaltStrike, they are taking the time to understand it and build out detection capabilities. I also wish more teams publicly released or talked about how they are using CobaltStrike so that more defenders can learn from its use.

I hope you have found this interesting and use this as a way to begin (or continue) your journey in understanding more about CobaltStrike and its many uses.
