Detecting Malicious C2 Activity -SpawnAs & SMB Lateral Movement in CobaltStrike

dansec.medium.com/detecting-malicious-c2-activity-spawnas-smb-lateral-movement-in-cobaltstrike-9d518e68b64

Dan Lussier February 24, 2021

Dan

Dan Lussier

Jan 15, 2021

.

5 min read

Understanding common attack vectors and how threat actors move in your environment post-compromise is critical to identifying what kind of potential threats exist in an environment. We're going to review how threat actors use common C2 spawning features to utilize elevated privileges which allow for lateral movement via SMB, and what defenders can do to detect it.

In the last article, a lot of time was spent describing how advanced threat actors often spawn a sacrificial process. This mainly focused on same user-credential based spawning, however in many campaigns threat actors have harvested credentials and want to spawn a process as that different user context, often with elevated privileges.

SpawnAs

When a threat actor is able to successfully get a foothold on a target network some of the first steps are recon which includes identifying easy ways to get elevated privileges. There are many ways to initially get elevated privileges, but most common are Kerberoasting, ASREPRoast, and file share enumeration looking for cleartext credentials. Although these can be noisy, they're highly effective in delivering elevated privileges.

Let's take a look at spawning a process as a new user

Not a great way to show it, but you can see beacon.exe (initial payload) spawn chrome_proxy.exe (elevated with another account) with a bunch of network connections following it, if module detection was enabled that would also be tied to the new elevated process.

Now let’s take a look at a rule that would detect a threat actor utilizing CobaltStrike to spawn a new process with elevated privileges.

```
rule detect_cobaltstrike_spawnas {  meta:    author = "Dan L"    description = "Look for a cobaltstrike spawnas"    version = "1.0"    severity = "High"    mitre_TA = "TA0004 - Privilege Escalation"    mitre_T1 = "T1055"    mitre_url = ""events:// Successful Login        $e0.metadata.product_event_type = "UserLogon"        $e0.security_result.summary = "Successful login occurred"        $e0.target.user.userid != /.*$.*/ nocase        $e0.target.user.userid != /.*dwm.*/ nocase        $e0.principal.hostname = $hostname// Modules being loaded due to new session being loaded        $e1.metadata.event_type = "PROCESS_MODULE_LOAD"        $e1.principal.hostname = $hostname// A process injection must happen to spawnas        $e2.metadata.event_type = "GENERIC_EVENT"        $e2.metadata.product_event_type = "ProcessInjection"        $e2.principal.hostname = $hostnamematch:   $hostname over 1mcondition:    $e0 and #e1 > 5 and $e2}
```

In this rule there are three areas of focus which are a successful logon, a bunch of modules being loaded, and process injection. The first of these is a successful logon, this is captured via EDR/Windows Event Logs as a new user account will be utilized to spawn the process. Second a bunch of modules will load at the time of the spawn of the new process, every time a payload executes many modules are loaded with it, and finally process injection.

At this point, the threat actor has obtained a newly created beacon to their C2 with what is often a privileged account, or an account that allows for additional lateral movement in an environment.

SMB Beacon/Payload

Additional recon will often take place with this newly spawned payload due to its new user context. At this point a threat actor will want to move laterally from their current compromised asset to other assets in the environment, and one of the ways to do this in CobaltStrike is via an SMB beacon. As of this writing (early 2021) you can still utilize the default SMB beacon without a custom named pipe that drops a file called beacon.dll onto an asset without being detected by many EDR platforms. With that said, that may not be the best move for opsec purposes from a threat actor perspective.

Let’s jump into what a rule that can detect lateral movement via the SMB beacon in CobaltStrike looks like.

```
rule cobaltstrike_smb_beacon_detection {  meta:    author = "Dan L"    description = "Detects the usage of cobaltstrike, metasploit SMB Beacon"    version = "2.0"    severity = "High"    mitre_TA = "TA0008 - Lateral Movement"    mitre_T1 = "T1570"    mitre_url = ""events:          // Look for a successful user login        $e0.metadata.product_event_type = "UserLogon"        $e0.security_result.summary = "Successful login occurred"        $e0.principal.hostname = $hostname          // Look for a file launching from ADMIN$        $e1.metadata.event_type = "PROCESS_LAUNCH"        $e1.target.process.command_line = /.*\\admin\$\\.*/ nocase        $e1.principal.hostname = $hostname          // Look for an SMB SERVER Share Open        $e2.metadata.product_event_type = "SmbServerShareOpenedEtw"        $e2.target.user.userid = /.*\$/ nocase        $e2.principal.hostname = $hostnamematch:   $hostname over 10mcondition:    $e0 and $e1 and #e2 > 1}
```

In this rule, we again have 3 conditions we are trying to match on, all of these conditions will be happening on the remote device the threat actor is attempting to connect to. First a successful logon event from EDR/Windows Event Logs, second a process launch in the ADMIN$ path on the remote asset, and finally the EDR/Windows Event of a file share being opened more than once. This can take a bit of time depending on an environment, so look for all of this activity to take place over a 10 minute window.

Let’s take a look at what this looks like as a detection

Each of the events from the rule triggered when attempting SMB lateral movement via CobaltStrike (Authentication, random process in the ADMIN$ folder launched, and SmbServerShare triggered).

Once a threat actor has spawned an SMB beacon on a remote asset, they've taken the the spawnas command with a potentially elevated privileged account context and opened a persistent connection to the initial compromised asset. A large component to this is a threat actor doesn't need to generate an abundance of HTTPS/DNS traffic which could be identified (outside of the initial compromise), as it will all traverse over SMB.

BONUS: This rule detects the default beacon.dll file being written on a remote asset. This should never trigger, however it can't hurt to have it running with many ransomware groups utilizing default settings in CobaltStrike.

(**2/24/21 Updated regex pattern on $e0**)

```
rule default_cobaltstrike_smb_beacondll {  meta:    credit = "Dan L"    description =
"Identify the default beacon.dll file being written to disk during SMB beacon lateral
movement"    Version = "1.0"    severity = "High"    mitre_TA = "TA0008 - Lateral
Movement"    mitre_T1 = "T1570"    mitre_url = ""

events:

          $e0.target.file.full_path = /.*beacon.dll.*/ nocase    $e0.principal.hostname
= $hostnamematch:       $hostname over 1mcondition:     $e0}
```

One thing I'd like to note here is that SMB is not the only means to move laterally through an environment and there are many Sharp tools that can offer similar tooling with less artifacts written to disk (SharpRDP, and SharpWMI). The good news here is that the rules from the previous article should detect the launch of any Sharp tool and hand off an alert to the defenders.

Defensive > Chronicle, LimaCharlie (Robust EDR platform)
Offensive > CobaltStrike