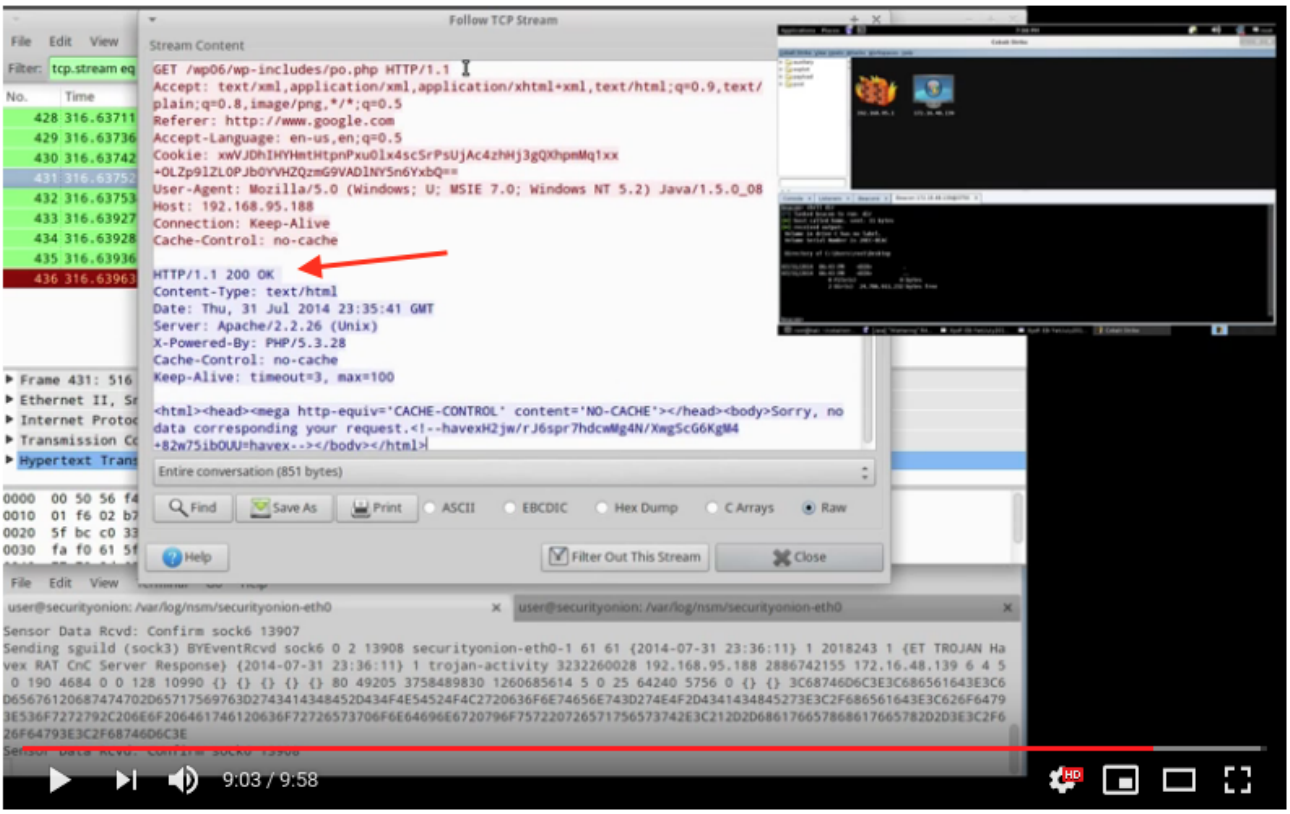
Identifying Cobalt Strike team servers in the wild

blog.fox-it.com/2019/02/26/identifying-cobalt-strike-team-servers-in-the-wild/

February 26, 2019



Energetic Bear / Crouching Yeti / Dragonfly - Threat Replication Case Study

1,068 views

13 0 SHARE SAVE

Raphael Mudge
Published on Aug 12, 2014

SUBSCRIBE 11K

This video demonstrates how to replicate an actor known by many names (Energetic Bear, Crouching Yeti, and Dragonfly). I'll show you how to modify Beacon's C2 to look like the havex trojan. We will also stand up a Java drive-by attack to deliver a Beacon. We'll analyze all of this with Snort

SHOW MORE

How an anomalous space led to fingerprinting

Summary

On the 2nd of January 2019 Cobalt Strike version 3.13 was released, which contained a fix for an “extraneous space”. This uncommon whitespace in its server responses represents one of the characteristics Fox-IT has been leveraging to identify Cobalt Strike Servers, with

high confidence, for the past one and a half year. In this blog we will publish a full list of servers for readers to check against the logging and security controls of their infrastructure.

Cobalt Strike is a framework designed for adversary simulation. It is commonly used by penetration testers and red teamers to test an organization's resilience against targeted attacks, but has been adopted by an ever increasing number of malicious threat actors.

Subtle anomalies like these should not be underestimated by blue teams when it comes to combating malicious activity.

About Cobalt Strike

Cobalt Strike is a framework designed for adversary simulation. It is commonly used by penetration testers and red teamers to test an organization's resilience against targeted attacks. It can be configured using Malleable C&C profiles which can be used to customize the behavior of its beacon, giving users the ability to emulate the TTP's of in the wild threat actors. The framework is commercially and publicly available, which has also led to pirated/cracked versions of the software.

Though Cobalt Strike is designed for adversary simulation, somewhat ironically the framework has been adopted by an ever increasing number of malicious threat actors: from financially motivated criminals such as Navigator/FIN7, to state-affiliated groups motivated by political espionage such as APT29. In recent years, both red teams and threat actors have increasingly made use of publicly and commercially available hacking tools. A major reason for this is likely their ease of use and scalability. This two-sided element of pentesting suites makes it a critical avenue for threat research.

Cobalt Strike Team Servers

While the implant component of Cobalt Strike is called the "beacon", the server component is referred to as the "team server". The server is written in Java and operators can connect to it to manage and interact with the Cobalt Strike beacons using a GUI. On top of collaboration, the team server also acts as a webserver where the beacons connect to for Command & Control, but it can also be configured to serve the beacon payload, landing pages and arbitrary files.

Communication to these servers can be fingerprinted with the use of Intrusion Detection System (IDS) signatures such as Snort, but with enough customization of the beacon, and/or usage of a custom TLS certificate, this becomes troublesome. However, by applying other fingerprinting techniques (as described in the next section) a more accurate picture of the Cobalt Strike team servers that are publicly reachable can be painted.

Identifying Cobalt Strike Team Servers

One of Fox-IT's InTELL analysts, with a trained eye for HTTP header anomalies, spotted an unusual space in the response of a Cobalt Strike team server in one of our global investigations into malicious activity. Though this might seem irrelevant to a casual observer, details such as these can make a substantial difference in combating malicious activity, and warranted additional research into the set-up of the team servers. This ultimately led to Fox-IT being able to better protect our clients from actors using Cobalt Strike.

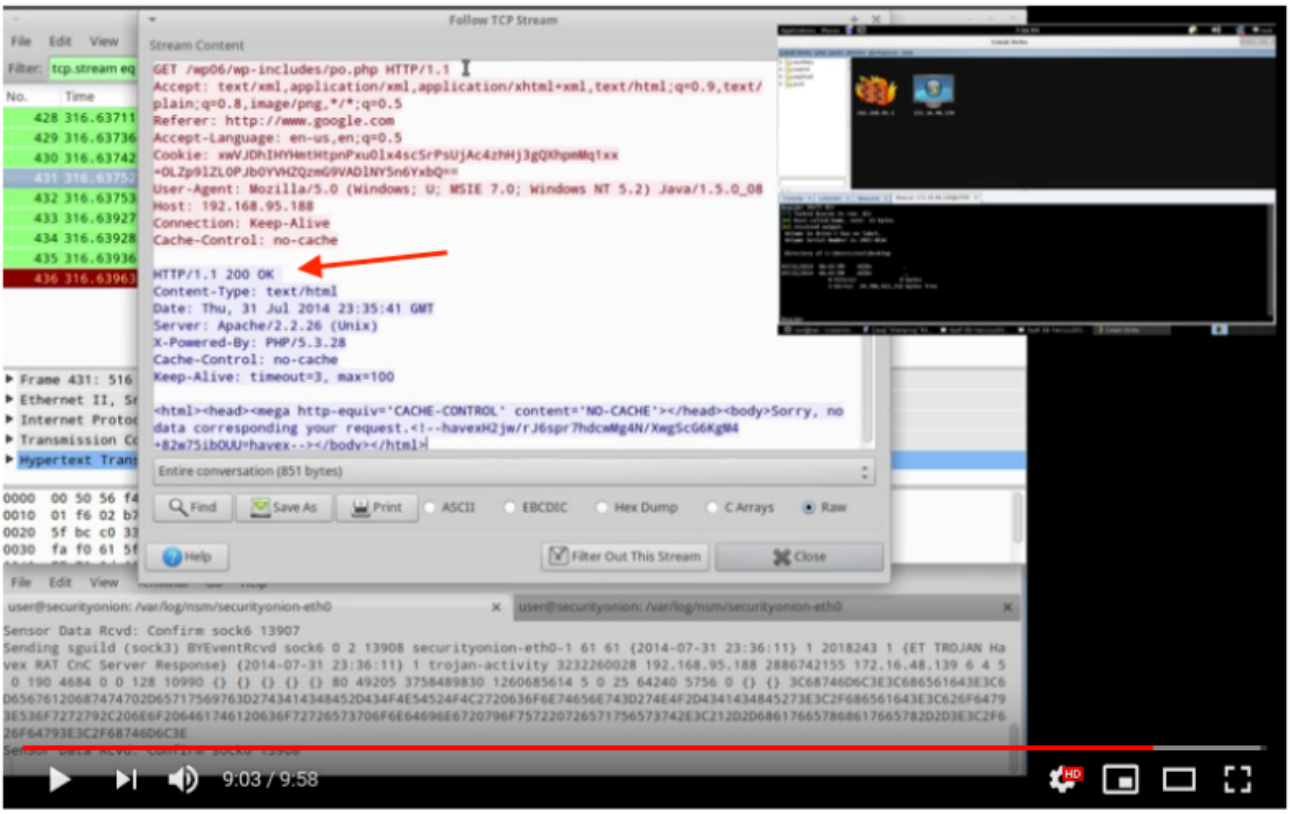
The webserver of the team server in Cobalt Strike is based on NanoHTTPD, an opensource webserver written in Java. However this webserver unintendedly returns a surplus whitespace in all its HTTP responses. It is difficult to see at first glance, but the whitespace is there in all the HTTP responses from the Cobalt Strike webserver:

```
00000000  48 54 54 50 2f 31 2e 31  20 32 30 30 20 4f 4b 20   HTTP/1.1  200 OK
00000010  0d 0a 43 6f 6e 74 65 6e  74 2d 54 79 70 65 3a 20   ..Conten t-Type:
00000020  61 70 70 6c 69 63 61 74  69 6f 6e 2f 6f 63 74 65   applicat ion/octe
00000030  74 2d 73 74 72 65 61 6d  0d 0a 44 61 74 65 3a 20   t-stream ..Date:
00000040  46 72 69 2c 20 38 20 4a  61 6e 20 32 30 31 36 20   Fri, 8 J an 2016
00000050  31 35 3a 31 37 3a 35 30  20 47 4d 54 0d 0a 43 6f   15:17:50  GMT..Co
00000060  6e 74 65 6e 74 2d 4c 65  6e 67 74 68 3a 20 30 0d   ntent-Le ngth: 0.
00000070  0a 0d 0a                                          ...
```

Extraneous whitespace after HTTP status code

Using this knowledge it is possible to identify NanoHTTPD servers, including possible Cobalt Strike team servers. We found out that public NanoHTTPD servers are less common than team servers. Even when the team server uses a Malleable C2 Profile, it is still possible to identify the server due to the "extraneous space".

The "extraneous space" was fixed in Cobalt Strike 3.13, released on January 2nd of 2019. This means that this characteristic was in Cobalt Strike for almost **7** years, assuming it used NanoHTTPD since the first version, released in 2012. If you look carefully, you can also spot the *space* in some of the author's original YouTube videos, dating back to 2014.



Energetic Bear / Crouching Yeti / Dragonfly - Threat Replication Case Study

1,068 views 13 0 SHARE SAVE

Raphael Mudge
Published on Aug 12, 2014

SUBSCRIBE 11K

This video demonstrates how to replicate an actor known by many names (Energetic Bear, Crouching Yeti, and Dragonfly). I'll show you how to modify Beacon's C2 to look like the havex trojan. We will also stand up a Java drive-by attack to deliver a Beacon. We'll analyze all of this with Snort

SHOW MORE

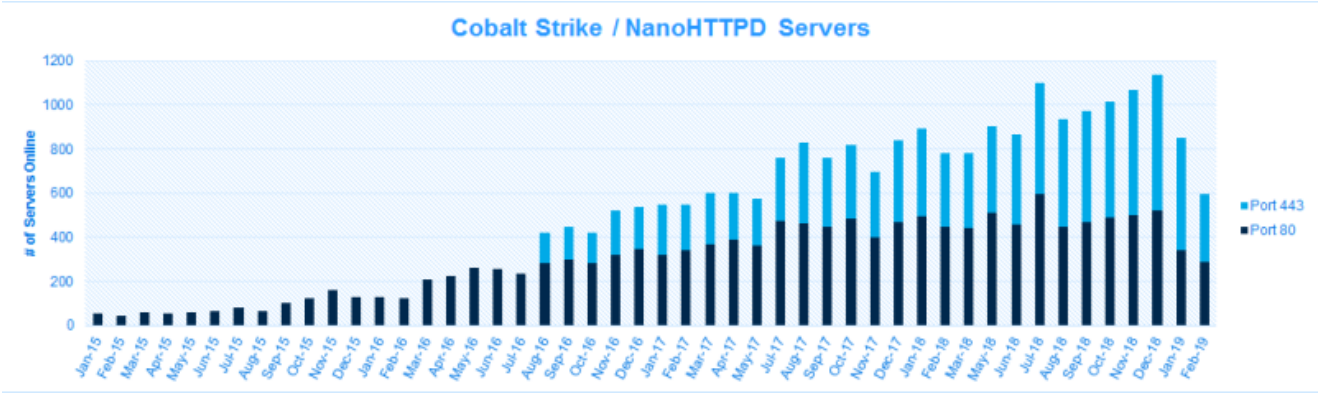
The fact that the removal of this space is documented in the change log leads us to believe that the Cobalt Strike developers have become aware of the implications of such a space in the server response, and its potential value to blue teams.

```
January 2, 2019 - Cobalt Strike 3.13
--------------
+ CS now prints console warnings, on payload staging, when kill date is past.
+ dcsync [FQDN] now runs mimikatz's dcsync with options to export all hashes
+ Added a parser to add dcsync [FQDN] hashes to credential store.
- Removed the 'mode smb' option to turn an arbitrary Beacon into an SMB Beacon
+ Refactored Beacon HTTP/HTTPS/DNS and Beacon SMB into separate binaries
+ Reworked the link management and link client for Beacon
+ Added stageless windows/beacon_reverse_tcp as a Beacon pivot listener option.
+ Removed extraneous space from HTTP status responses.
```

The change log entry highlighted above refers to the removed space being “extraneous”, in a literal sense meaning not pertinent or irrelevant. Due to its demonstrated significance as fingerprinting mechanism, this description is contested here.

Scanning and results

By utilizing public scan data, such as Rapid7 Labs Open Data, and the knowledge of how to fingerprint NanoHTTPD servers, we can historically identify the state of publicly reachable team servers on the Internet.



The graphs shows a steady growth of Cobalt Strike (NanoHTTPD) webservers on port 80 and 443 which is a good indication of the increasing popularity of this framework. The decline since the start of 2019 is most likely due to the “extraneous space” fix, thus not showing up in the scan data when applying the fingerprint.

In total Fox-IT has observed **7718** unique Cobalt Strike team server or NanoHTTPD hosts between the period of 2015-01 and 2019-02, when based on the current data (as of 26 Feb 2019) from Rapid7 Labs HTTP and HTTPS Sonar datasets.

The table below contains several examples of Cobalt Strike team servers, used by malicious threat actors:

IP Address	First seen	Last seen	Actor
95.128.168.227	2018/04/24	2018/05/22	APT10
185.82.202.214	2018/04/24	2018/09/11	Bokbot
206.189.144.129	2018/06/05	2018/07/03	Cobalt Group

The full list of Cobalt Strike team servers identified using this method can be found on the following Fox-IT GitHub Repository.

Do note that possible legitimate NanoHTTPD servers are listed here and that some IP addresses may have been rotated and reused swiftly, for example due to being part of Amazon or Azure cloud infrastructure.

Therefore we recommend to investigate connections to these IP addresses within the corresponding time ranges. A starting point is to verify whether requested URI matches a Cobalt Strike beacon checksum, or by using historical DNS data using passive DNS. Going beyond this can be done in various ways and we challenge readers to use their investigative creativity.

Please also note that this list contains servers of both legitimate and illegitimate operations, since these cannot be distinguished easily. Fox-IT recognizes the merit of building and distributing offensive tooling, particularly for security testing purposes. In our opinion the benefits of publishing this list (allowing everyone to detect unwanted attacks retroactively) outweigh the downsides, which could include potentially affecting ongoing red team operations. We believe that we all have an interest in raising the bar of security operations, and therefore increasing visibility across the board will inform a higher level of operational security and awareness on all sides.

Network IDS Signatures

Fox-IT developed a Snort rule for network detection. The rule checks for the “extraneous space” in the HTTP header. Please note that this detection rule only works to detect plaintext HTTP traffic to and from Cobalt Strike Team servers with the Cobalt Strike version up until release 3.13. Nevertheless, this is still a valuable detection rule, considering threat actors tend to use pirated and cracked- and therefore inherently unsupported- versions.


```
alert tcp any any -> any any (msg:"FOX-IT – Trojan – Possible CobaltStrike C2 Server"; \
flow:to_client; \
content:"HTTP/1.1 200 OK |0d0a|"; fast_pattern; depth:18; \
content:"Date: "; \
pcre:"/^HTTP/1.1 200 OK \r\nContent-Type: [^\r\n]{0,100}\r\nDate: [^\r\n]{0,100} GMT\r\n(Content-Length: \d+\r\n)\r\n/"; \
threshold:type limit, track by_dst, count 1, seconds 600; \
classtype:trojan-activity; priority:2; \
```

sid:21002217; rev:3;)

view raw

cobaltstrike-extraspace.rules

hosted with ❤ by GitHub

Conclusion

- Organizations are encouraged to use the published list with Cobalt Strike team servers IP addresses to retroactively verify whether they have been targeted with this tooling by either a red team or an adversary in the recent past. The IP addresses can be checked with e.g. firewall and proxy logs, or on aggregate against SIEM data. To minimize the amount of false positives, the reader is urged to take the corresponding first and last seen dates into consideration.
- For the 'red team readers' of this blog looking for ways to avoid their Cobalt Strike team server being both publicly available and easy to fingerprint, see the Cobalt Strike Team Server Population Study blog for a detailed set of mitigations. Furthermore, Red Teams are encouraged to critically examine their toolsets in use or rely on their Blue Team, for potential tell-tales and determine the appropriate way to apply and mitigate such findings for both Red and Blue team purposes.

Watch this space (pun intended) for further analysis on this subject.