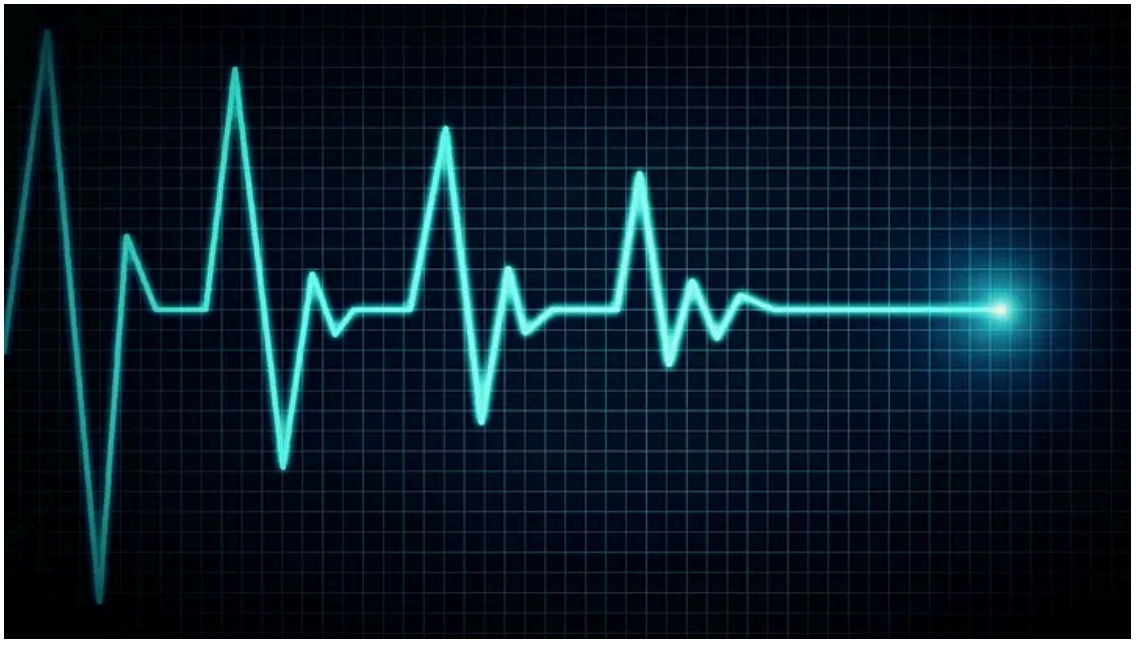
Enterprise Scale Threat Hunting: C2 Beacon Detection with Unsupervised ML and KQL — Part 1

By Mehmet Ergene

Published: 2021-05-23 · Archived: 2026-04-05 17:04:07 UTC

This blog is part one of a two-part series focused on C2 beacon detection.

Press enter or click to view image in full size



> Beacons or beaconing is the practice of sending short and regular communications from one host to another. As used in malware, this is mostly used to communicate to an external host that a compromised internal host is active, functioning and ready for further instructions. **Not all beacons are malicious in nature.** There are many benign use cases of beaconing behaviour, such as system time services, software update services, etc.[1]

In the world of malware, a beacon doesn't have to use regular intervals. As seen in many C2 frameworks, it is possible to use jitter for communication. Using jitter makes it difficult to detect beacons, and hence the C2 traffic.

Jitter Usage in C2 Beacons

For example, 60 seconds of sleep with 10% jitter results in a uniformly random sleep between 54 and 66 seconds for PoshC2 and Empire, or a uniformly random sleep between 54 and 60 seconds for Cobalt Strike.*[2]*

I've come across few good methods for detecting network beacons(including the ones having jitter) by using statistical analysis with KQL. Unfortunately, none of them worked for me because my data was a little bit huge. Also, I wasn't able to detect some beacons explained below. So, I decided to develop a high-performing query that can analyze a large amount of data within timeout thresholds and detect almost all kinds of beacons with high precision.

In this blog series, I'll explore the approaches and problems in beacon detection, how we can solve them, and develop a beacon detection mechanism with KQL in Azure Sentinel and Microsoft 365 Defender by using Sysmon, Process network and Firewall/Proxy events.

Statistics 101: Standard Deviation

> In statistics, the **standard deviation** is a measure of the amount of variation or dispersion of a set of values. A low standard deviation indicates that the values tend to be close to the mean (also called the expected value) of the set, while a high standard deviation indicates that the values are spread out over a wider range.
>
> A useful property of the standard deviation is that unlike the variance, **it is expressed in the same unit as the data.**

Relation Between Beacon Jitter and Standard Deviation

Let's say we have a Cobalt Strike beacon with a 15 minutes sleep and 25% jitter. It will have a random sleep between 675s and 900s. This means:
Average beacon sleep = 787.5s
Max standard deviation = 112.5s (not percentage)

Get Mehmet Ergene's stories in your inbox

Join Medium for free to get updates from this writer.

Remember me for faster sign in

By analyzing the **time deltas** between each consecutive connection of a source-destination pair, we can detect beacons with the help of standard deviation. Depending on the data source we have, we can also use **sent bytes** information to perform the same calculation. **However**, the standard deviation is not enough for proper detection. For example, if you set a threshold for standard deviation like 100, you will miss the beacon that has 15 minutes of sleep with 25% jitter if the standard deviation of the time deltas exceeds 100 seconds(Since the sleep duration calculated randomly, the standard deviation of the time deltas can be anything between 0–112.5s). The same applies to the **sent bytes.**

Outliers and False Negatives

Beacon detection assumes that the computer or the beacon is up and running for the duration of the data you analyze. Let's have a look at an example:

You analyze 24h of data to detect beacons. If a computer is turned off for 3 hours during this 24h period, you will have a spike in your time delta calculation which may result in false negatives. When using the **sent bytes** for the analysis, the beacon can send a fairly large amount of data occasionally and this may result in false negatives as well.

In order to detect beacons properly, we need to use other statistical values or calculations. We also need to eliminate situations that can cause false negatives.

In part two, I'll explain how we can solve these problems and apply an optimal method in Azure Sentinel and Microsoft 365 Defender by leveraging the KQL functions. I hope the result will be quite close(or maybe better) to the open-source project RITA. Stay tuned and follow me on Twitter | LinkedIn to get more updates! Finally, If you see something wrong, please let me know (I'm not expert at statistics).

References

1. https://www.activecountermeasures.com/threat-hunting-simplifying-the-beacon-analysis-process/
2. https://blog.fox-it.com/2020/01/15/hunting-for-beacons/
3. https://en.wikipedia.org/wiki/Standard_deviation
