Level 1.4 - Malware signature in compressed HTTP

HTTP responses can be compressed, which of course changes the response's hash.

Password

`q0glgtu5ir`

Instructions

1. Implement scanning compressed TCP streams
    - Hint: HTTP headers can help you identify compressed responses
2. While it is possible to decompress zip files into the file system, and inspect them there,
    - Writing un-scanned files to the file system is not advisable
    - Instead, the `gzip` module can be used to decompress gzip in memory (i.e. from a variable holding the data, no need to have a file)
    - Bonus: what impact will this have on performance?
3. Detect whenever the response is compressed, and if it is, decompress it in memory and then check for the malware signature.

Notes

GUIDING QUESTIONS Use the guiding questions below to identify the malicious trigger and structure your code's detection:

- How can we identify if the data is compressed in a HTTP response?
    - Is there a certain keyword or header that indicates its compressed?

- How can we decompress the data in memory without downloading the file?

To submit

Submit a ZIP `nids.zip` containing all `.py` files.