Parsing HTTP Headers Report

Proof of knowing your stuff in CSE312

Guidelines

Provided below is a template you must use to write your reports for your project.

Here are some things to note when working on your report, specifically about the **General Information & Licensing** section for each technology.

- **Code Repository**: Please link the code and not the documentation. If you'd like to refer to the documentation in the **Magic** section, you're more than welcome to, but we need to see the code you're referring to as well.
- **License Type**: Three letter acronym is fine.
- **License Description**: No need for the entire license here, just what separates it from the rest.
- **License Restrictions**: What can you *not* do as a result of using this technology in your project? Some licenses prevent you from using the project for commercial use, for example.

Also, feel free to extend the cell of any section if you feel you need more room.

If there's anything we can clarify, please don't hesitate to reach out! You can reach us using the methods outlined on the course website or see us during our office hours.

Flask / Python

General Information & Licensing

Code Repository	https://github.com/pallets/flask
License Type	BSD 3-Clause "New" or "Revised" License
License Description	• A permissive license similar to the BSD 2-Clause License, but with a 3rd clause that prohibits others from using the name of the copyright holder or its contributors to promote derived products without written consent • •
License Restrictions	• Liability • Warranty •

Dispel the magic of this technology. Replace this text with some that answers the following questions for the above tech:

- How does this technology do what it does? Please explain this in detail, starting from after the TCP socket is created
- Where is the specific code that does what you use the tech for? You ***must*** provide a link to the specific file in the repository for your tech with a line number or number range.
    - If there is more than one step in the chain of calls *(hint: there will be)*, you must provide links for the entire chain of calls from your code, to the library code that actually accomplishes the task for you.
    - Example: If you use an object of type HttpRequest in your code which contains the headers of the request, you must show exactly how that object parsed the original headers from the TCP socket. This will often involve tracing through multiple libraries and you must show the entire trace through all these libraries with links to all the involved code.

*This section will likely grow beyond the page

Flask is a web application framework written in Python and depends on the Jinja template engine and the Werkzeug WSGI toolkit. Flask listens to a port on TCP socket and receives an HTTP requests and parses the data. It parses it based on the protocol format and makes it available for users in a convenient manner.

Flask:
Wrappers.py: where the request class is located
Views.py: where parsing the header request method is located(https://github.com/pallets/flask/blob/cc66213e579d6b35d9951c21b685d0078f373c44/src/flask/views.py#L135)
Testing.py: possibly where all the params are created?

Start:

- app = Flask(__name__) (our server code)
    - Ctrl click flask to go into flask library
    - https://github.com/pallets/flask/blob/066a35dd322f689ec07d7c0e82b19eacadac3c6b/src/flask/app.py#L110
- run()
    - Ctrl f for run since the description for run is ‘ `"""Runs the application on a local development server.`’
    - https://github.com/pallets/flask/blob/066a35dd322f689ec07d7c0e82b19eacadac3c6b/src/flask/app.py#L1077
- run() comment description
    - In the run description, werzuerk can be found as one of the param options
    - https://github.com/pallets/flask/blob/066a35dd322f689ec07d7c0e82b19eacadac3c6b/src/flask/app.py#L1115
    - `server. See :func:`werkzeug.serving.run_simple` for more information.`
    - `Ctrl-f and find run_simple function`
- Run_simple
    - https://github.com/pallets/flask/blob/066a35dd322f689ec07d7c0e82b19eacadac3c6b/src/flask/app.py#L1191
    - Ctrl click on run_simple leads to werkzeug library
    - https://github.com/pallets/werkzeug/blob/3115aa6a6276939f5fd6efa46282e

0256ff21f1a/src/werkzeug/serving.py#L907
    - In the parameter description it says “ :class:`~BaseHTTPServer.BaseHTTPRequestHandler` subclass to handle requests.”
    - https://github.com/pallets/werkzeug/blob/3115aa6a6276939f5fd6efa46282e0256ff21f1a/src/werkzeug/serving.py#L965
    - Ctrl-f for BaseHTTPRequestHandler
- BaseHTTPRequestHandler
    - Leads to class WSGIRequestHandler(BaseHTTPRequestHandler):
    - https://github.com/pallets/werkzeug/blob/3115aa6a6276939f5fd6efa46282e0256ff21f1a/src/werkzeug/serving.py#L148
    - Ctrl click on BaseHTTPRequestHandler
    - Leads to class BaseHTTPRequestHandler(socketserver.StreamRequestHandler):
    - https://github.com/python/cpython/blob/bf26bdf6ac04878fc720e78422991aaedb9808a1/Lib/http/server.py#L145
    - This handles all the http requests

- What does this technology (library/framework/service) accomplish for you?
    - The HTTP header parser allow the client and server to pass and receive information from an HTTP request or response

How does this technology accomplish what it does?

Parsing the request:

https://github.com/python/cpython/blob/bf26bdf6ac04878fc720e78422991aaedb9808a1/Lib/http/server.py#L266

This function is where the requests header is parsed and deciphered. In the BaseHTTPRequestHandler class, it is noted that “`The headers and data are separated by a blank line.`” (`https://github.com/python/cpython/blob/bf26bdf6ac04878fc720e78422991aaedb9808a1/Lib/http/server.py#L162`) and “`<command> is a (case-sensitive) keyword such as GET or POST,<path> is a string containing path information for the request`” (https://github.com/python/cpython/blob/bf26bdf6ac04878fc720e78422991aaedb9808a1/Lib/http/server.py#L168)

With that in mind, line 282 is where the header and data is separated by the code “requestline = requestline.rstrip('\r\n')” (https://github.com/python/cpython/blob/bf26bdf6ac04878fc720e78422991aaedb9808a1/Lib/http/server.py#L282) The lines below it are just used to figure out if the http version, request syntax, and request type are incorrect or not.

If they are correct then we proceed to line 322 where the variables command and path are created. (https://github.com/python/cpython/blob/bf26bdf6ac04878fc720e78422991aaedb9808a1/Lib/http/server.py#L322) Below that is where command is checked to see if its not a get request and if its not then we send an error.

After that, line 330 is where self.command and self.path is set (https://github.com/python/cpython/blob/bf26bdf6ac04878fc720e78422991aaedb9808a1/Lib/http/server.py#L330)

Line 336 is where the double slash is replaced with one slash to avoid open redirect attacks (https://github.com/python/cpython/blob/bf26bdf6ac04878fc720e78422991aaedb9808a1/Lib/http/server.py#L336)

Line 341 is where the headers are set (https://github.com/python/cpython/blob/bf26bdf6ac04878fc720e78422991aaedb9808a1/Lib/http/server.py#L341)

- What license(s) or terms of service apply to this technology?
    - BSD 3-Clause "New" or "Revised" License
    - The permissions for this License is Commercial use, Modification, Distribution, and Private use
    - Whereas the limitations are liability and warranty
    - This means that for our code we must meet these requirements:

Source: https://github.com/pallets/flask/blob/main/LICENSE.rst