HTTP Requests and Responses

Due Date: Friday, June 8th, 11:59:59 PM

In this assignment you will be writing a simple Python script to interact with a server in a couple different ways.

First your assignment should attempt to access http://cs.rpi.edu/~holzbh/CCN18/PA1_server.php , which will cause the server to return an error. Your program should not crash, and should handle the error by printing out the response code and reason. For example, a 403 might look like the following output.

An error happened, return code: 403, reason: Forbidden

Next for all modes of operation, you will use the base URL http://cs.rpi.edu/~holzbh/CCN18/PA1_server.php?homework=Su2018_1 .

When no other GET/POST parameters are provided, the server will return a word and a number. If you view these in a browser you will see they are on separate lines, however, this does not use the newline character “\n” . Your script should send a request to the link provided above, and print the reponse, for example

Response content: undoings21

Next, your script send a request the link above, but it should use an additional GET parameter, called mode with a value of `get_nonce` , a POST parameter named word containing the word from before, and a request length equal to the number in the previous response. In the example above, this means that word is undoings and the request length (including URL, headers, etc.) is 21 bytes.

If this is successful, the server will send back a long string called a nonce. Your script should store this nonce and send one final request to the server. In additional to the base URL, this time there should be a GET parameter mode with a value of `verify_nonce` and three POST variables: word, num, and hash. The first two variables are the same word and number from the original response content (so in this example undoings and 21 respectively, and hash should be set to the nonce you just received. Print the nonce as you send it, and then print Verification status: followed by the resulting entity body from the server. In this example, the output would look like:

Sending nonce: 6f4b6612125fb3a0daecd2799df6c9c299424fd920f9b308110a2c1fbd8f443
Verification status: Verified!

Make sure to test your script a few times - the server will select random values when you use the base URL. You should try sending it nonsense to see what happens as well. You may want to use Wireshark to help you debug.

Submit a single .py file via e-mail along with a README.txt describing how long you spent on the assignment, any special testing you did (input and output), and the output from your script running 3 times.

The grade breakdown will be as follows:

10 points: Error Handling
10 points: Variable names clear, sufficient comments
20 points: Works for simple input
10 points: Works on non-alphanumeric input