ACN Programming Assignment-2 : WWW

README File

Group 13 **Extension 3**

Module's Overview

Main role of this project is to code a client and server which can communicate through proxy server. Proxy Extension here is showing the feature's like parental control / URL Filtering, Safesearch / Content Filtering (in google chrome browser)

URL Filtering here blocks the client to access such provided URL's and Content Filtering here shades or deletes the provided keywords from the user's requested file.

Part 1: Simple Web Client

File Name : *web_client.py*

Our simple web client is able to handle http and some https request. Client takes command line input from the user like server address, server port, file path and proxy address & proxy port (if configured) and send the get request to the server or proxy over TCP socket. If server port is 443 or server work over https then out client create SSL socket by wrapping on TCP socket and send the request on SSL socket. On receiving response from the server client parse the data and if there is external reference link inside file than client one by one fetch all the data from links.

For implementing we are using 3 libraries:

ssl	:	For https implementation
socket	:	For TCP socket implementation
re	:	For regex use

Part 2: Simple Web Proxy Server

File Name : *web_proxy.py*

Our simple web proxy server able to handle http request. When client sends a http get request, proxy server parse that request and fetch some useful details from it like server address and all. Then proxy creates new request and send it to server. On receiving response from server, proxy first parse and understand the response code and response message and then just simply forward the response to the client. For the communication all client-proxy and proxy-server connections are handle over new threads. Also it can handle and show the response code like 200, 404, 304, 301.

For implementing we are using 4 libraries:

time	:	For generating response time.
socket	:	For TCP socket implementation
threading	:	For concurrent communication
queue	:	For message passing between 2 threads.

Part 3: Simple Web Server

File Name : *web_server.py*

Our simple web server is able to handle http requests. Server listen on port 80 (default http port) and when it get connection request from client it accept connection and start the communication over new thread, means our server is capable of multiple client / concurrency. Server parse the request coming from the client or proxy then it simply check for the file in server's local directory if the file is available then it set file data as body of response otherwise it set small html code for 404 Not Found and send the response to the client.

For implementing we are using 5 libraries:

threading	:	For concurrency
socket	:	For TCP socket implementation
os	:	For file system works
time	:	For generating response time
magic	;	For getting MIME extension / content-type of the file.

Part 4: Extension to simple proxy server

File Name : *web_proxy_extend.py*

Our extended web proxy is able to handle http requests. Main role of our extended proxy is to develop the Content Filtering and URL filtering system. Where proxy takes some list of blacklisted websites as part of URL filtering and takes list of blacklisted keywords as part of Content Filtering. It than listen on 8080 port and when client request to proxy, it first parse the request and then check the host if the host is in blacklisted website list, then proxy send response of 403 Forbidden / Access Denied. If website is not in list, then it creates a new thread for proxy-server communication and send the modified request to server. On receiving response from the server proxy parse the data and check if the body contains keyword from blacklisted keyword list or not if present then it shade that word by ***** and generate a modified response and send it to client otherwise send the response simply.

Other role of extended proxy is to generate web statistics in which proxy keeps track of users session and users requested and visited websites. Then at end it plot 2 pie charts one is for the no of times particular website is visited and another one is for the no of overall websites user request through proxy.

For implementing we are using 7 libraries:

threading	:	For concurrency
socket	:	For TCP socket implementation
os	:	For file system works
time	:	For generating response time
queue	:	For message passing between 2 threads
re	:	For regex and string matching
matplotlib	:	For plotting piechart as part of web statistics

Note: All parsing is done using simple string modification and string methods. No prebuild libraries are used.

<u>**How to run code:**</u>

1. Run all client, proxy (either simple proxy or extended proxy), and server on different terminal may be on same machine or different machine.
2. Fetch the server's IP address and proxy's IP address from their terminals.
3. Enter Server IP & Port, Proxy IP & Port (in case proxy is configured), File Path (path to the file to be accessed).
4. In case you are executing extended proxy provide the blacklisted keywords and blacklisted url's separated by comma.

Note:

- All 4 modules are strictly working on http 80 port so don't feed port 443 or any URL containing https. Over this *web_client.py* will work on port 443 or https, here are some websites over which client is tested for https or port 443:
    1. Website: iith.ac.in & Port 443
    2. Website: gaia.cs.umass.edu & Port 443

- As per the TA's guideline and Assignment Document Statement "*This web proxy performs the first role (proxying) but not the second role (caching).*"
  Proxy Server not handling Conditional Get, also we are dealing with 3rd Extension where we don't need any caching and without caching there is no meaning of Conditional Get.