Net Ninny - A Web Proxy Based Service

George Abo Al Ahad

User manual

Start the program proxyserver in the terminal. After the message that reads: “please, enter a port number between 1025 and 65535: “, enter desired port number on which the web proxy will create its server socket. This is handled by the function *userInputPort(port).*

Next the command window will prompt: “Waiting for connection”. Now the proxy is set and the server socket is waiting for a request from the browser. The user has tell the local browser to use a HTTP-proxy through the browser connection settings and select the port entered previously. All major browsers allows this functionality and therefor will function with the proxy if set. Only HTTP requests will pass through the proxy, whereas HTTPS bypasses the proxy.

When accessing a page the browser will send a get request to the server side of the proxy. The url of the request will be filtered for the inappropriate words: Norrköping, Britney Spears, Paris Hilton and SpongeBob. If a bad word is encountered in the proxy will use send 30 redirect message to the browser, redirecting the user to a page stating that the url is inappropriate. The url filtering is carried out in a function called *is_inappropriate(input).* Input in this case will be the url, which will be transformed to lowercase. The function will then search for the occurrence of each word in lowercase in the url. For instance paris hilton is separated with a white space, the function will then search for the occurrence of paris and hilton. If both appears then the url is defined as inappropriate.

If the url is free of inappropriate words then the program proceeds call the *CreateSocketandConnect(),* which will connect to the port 80 of the remote server. Then the client will call its method *sendReq(get-req)* which will send the get request of the browser to the remote server. After that the Client’s method *receiveFirst()* will receive the first part of the HTTP-request from the remote server and scan its content type. If it is non-text content, it will be sent to the browser. In this process, the proxy program will receive and send packets of maximum size of 4096 bytes right away until the remote server closes connection. This is advantageous since both receive and send are best effort processes. In the case of having text based files the client-side of the proxy will buffer the content and then scan it for inappropriate information using a domestic function that is called *checkForbiddenWords(input)*. If inappropriate content is found the proxy will redirect the request to another site that states that the content is probably inappropriate.

Testing

All testing was done via ThinLinc client, with firefox web browser.
Following pages were tested:

1. http://www.ida.liu.se/~TDTS04/labs/2016/NetNinny/default.html
   Simple text based site. Works as expected.
2. http://www.ida.liu.se/~TDTS04/labs/2011/ass2/goodtest2.html
   HTML with text content. Works as expected.
3. http://www.ida.liu.se/~TDTS04/labs/2011/ass2/SpongeBob.html
   HTML with inappropriate word in the url, that is, SpongeBob. Performs a 302 redirect.
4. http://www.ida.liu.se/~TDTS04/labs/2011/ass2/badtest1.html
   Text based web page with SpongeBob in the content. Performs a 302 redirect.
5. http://stackoverflow.com/
   Gzip-encoded page, passes through the proxy even though any of the inappropriate words are present.
6. http://www.aftonbladet.se/
   Gzip-encoded page that passes through the proxy even though any of the inappropriate words are present.
7. https://www.svd.se/
   Gzip-encoded page handled as test 5 and 6.
8. http://liu.se/?l=sv
   Gzip-encoded page handled in the same fashion as previous tests.
9. https://qz.com/
   This page doesn't go through the proxy since only a http-proxy is used. It works.
10. http://www.bbc.com/
    Gzip-encoded website. The site works but the forbidden words will not be sought after because the encoding.
11. http://www.google.com
    The page is working, but forbidden words are not being filtered. This is because the site is encoded (gzip) and therefore not searched
12. http://www.youtube.com
    This page doesn't go through the proxy since only https is used? It works.
13. https://vimeo.com/player
    This page doesn't go through the proxy since only https is used? It works.
14. http://www.dailymotion.com/
    Gzip-encoded website. The site worked partly, but always showed page not found. This was corrected by changing the header by taking away the host name from the url and leaving only the path to the object requested.
15. https://www.wikipedia.org/
    Gzip-encoded website. The site works but the forbidden words will not be sought after because the encoding.

Summary

The proxy manages to filter strictly text based (pages that have *content-type: text/plain* or *text/html* in the header field) web pages that uses either HTTP 1.0 or 1.1. If the header *content-encoding* is present, the proxy will not filter the page it will simply let the page pass through. Pages that uses HTTPS will not go through the proxy and will consequently not be filtered.