Practical exercise 2 - Task e - Group 88

e)

The easiest way to exploit this security problem is to make use of the “..” in the path parameter to go up from the root of the webserver and then jump into other files on the system that should not be available to the user. An example of this would be something like

“GET ../criticaldirectory/criticalfile/”

which would gain access to critical files in a directory outside of the webserver directory.

One possible solution to the problem is to check if the GET request includes “..” and then return a “403 Forbidden” response. This way the user cannot go outside of the directory of the webserver.

Another possible solution is to create a specific user on your unix system which is dedicated to running the webserver. We then only give this user read permissions to the directory of the webserver (like shown in lecture 3). This user will not have read/write or execute permissions on any other file on the system. When running the webserver and a malicious user of our webpage tries to access a file outside the webserver directory, the OS will forbid him/her from doing so and we can return a “403 Forbidden” response.