CMPE279 : Security
Programming Assignments 1 & 2

In these assignments, you will be improving the security posture of a socket-based client/server system, not unlike what you might find as a component of a multi-tier or microservice based application. I will provide the skeleton code for the client and the server, and you will, in each assignment, modify the code to add additional security hardening and mitigations.

You may work in teams of up to 2 people, but you can also work solo if you want.

Start by compiling and running the client and server code so you can see how they work. Next, start with assignment 1 and add the functionality described. Then move on to assignment 2 after that.

Assignment 1

In assignment 1, you will extend the server code to use privilege separation. You should accomplish this by splitting up the code into two logical parts – the part that sets up the socket and a separate part that processes the data from the client. Once you locate this split, you should fork and have the child process setuid() to drop its privileges to the "nobody" user. The server should wait for the child to exit and the child should process the connection from the client.

Submission process – Commit the code to a github repo containing a directory 'assignment1' at the root of the repo. Place all assignment 1 code/Makefiles/etc in that directory and commit to your repo (and push changes). Make sure you make the repo public as github invitations only last 7 days then expire.

Note – DO NOT use some random user ID when calling setuid. Use the well defined user ID for 'nobody'. Use the getpwnam() function to obtain the correct user ID. **DO NOT** hardcode the user ID.

In your repo, make a README.md file that lists the name(s) of the assignment team.

Each **individual team member** should submit the Canvas assignment (it will just ask for a link to your github repo's URL).

Grading – I will inspect the code to make sure the privilege separation is done properly and assign points accordingly. I might test your server code with my client.

Due – 11 Nov 2021 before midnight

Assignment 2

Starting with assignment 1 as a starting point, add in code to re-exec the server's child process after forking. You will need to determine how to pass the socket file descriptor between the forked child and the new exec'ed child, so that the new exec'ed child can process the incoming child request.

Submission process – Same as assignment 1 (create a new directory called assignment2 and place the assignment 2 files there).

Note – DO NOT create a "new" program – the server must re-exec **itself**, but you can add additional command line arguments or environment variables as needed.

Grading – Same as assignment 1.

Due – 18 Nov 2021 before midnight