Homework 5 - Security

Academic Honesty

Aside from the narrow exception for collaboration on homework, all work submitted in this course must be your own. Cheating and plagiarism will not be tolerated. If you have any questions about a specific case, please ask us. We will be checking for this!

NYU Tandon's Policy on Academic Misconduct:
https://engineering.nyu.edu/campus-and-community/student-life/office-student-affairs/policies/student-code-conduct#chapter-id-34265

Accepting the Assignment and Setting Up a Repo (Mandatory Step)

1. Head to **https://anubis.osiris.services/**
2. Press "LogIn" on the side bar, and log in with your NYU ID
3. Enter GitHub username (If not done yet)
4. Go to "Courses", and press "Join Class" with code **iloveos** to be registered to the class
5. Go to "Assignments", and accept the assignment
    a. Press "Create Assignment Repo" (This will create a repository on Github)

Working on the Assignment

For this assignment we will offer you **two options** for a platform to work on, or you can use your own platform. It's up to you what you use. It's **your responsibility** to turn it on time no matter what platform you choose. **To make this 100% clear**. Whether you choose **Anubis or Vagrant you are the only one responsible for turning it on time.** If Anubis or Vagrant doesn't work for you 5 minutes, 5 hours or 5 days before the deadline, it's not the Professor's responsibility or the TAs responsibility, it's **your responsibility.**

Using Anubis IDE (Option 1)

1. Select “Anubis Cloud IDE” from the Assignment
2. On the top menu bar, select “Terminal” and “New Terminal” to bring up a terminal
3. Complete your assignment in the project folder, and commit your changes for submission/grading

Notes:

- With auto save on, the IDE automatically commits your changes to GitHub every couple of minutes. Or type “autosave” in the shell to save your changes
    - Even with this option on, you are still **responsible** for saving your changes
- If you have any questions, don’t hesitate to post on Forum or email any of the TAs

Using Vagrant (Option 2)

1. Follow the instructions in the repo: **https://github.com/wabscale/cs3224-vm**
2. Pull the assignment repository from Github
3. Complete your assignment in the project folder, and commit your changes for submission/grading

Using your own environment (Option 3)

1. As always you are free to use your own linux environment
    a. Assignments are graded based on the Anubis IDE environment
2. Pull the assignment repository from Github
3. Complete your assignment in the project folder, and commit your changes for submission/grading

You must accept the assignment using Anubis for any of the options listed above.

Password Authentication

In this question, you will be making XV6 a “secure” operating system.

When a user first boots xv6, we will assume there is no password set. You should implement code that checks if a password file exists, and if it does not, prompts the user to enter a password. Because users sometimes mistype their passwords, you should ask for the password twice, and then compare them to make sure they match.

When setting a new password, once the user has entered a password, you should:

1. Generate a random **salt** value.
2. Hash the user's password using the `bcrypt()` function.
3. Write the salt and the hash out to a file.

To verify a password:

1. Ask the user to enter a password.
2. Read the salt and hash from the password file.
3. Use the `bcrypt_checkpass()` function to see if the password is correct.

A typical session might look like:

```
    cpu1: starting
    cpu0: starting
    sb: size 2000 nblocks 1941 ninodes 200 nlog 30 logstart 2
inodestart 32 bmap start 58
    No password set. Please choose one.
    Enter password: test
    Re-enter to confirm: tast
    Passwords do not match. Try again.
    Enter password: test
    Re-enter to confirm: test
    Password successfully set. You may now use it to log in.
    Enter password: test
    Password correct, logging you in.
    init: starting sh
    $
```

You do not need to restrict the number of times the user can attempt entering a password. You may assume that the password entered will **not** be longer than 72 characters.

Subsequent boot into XV6 should only ask for password and not ask to set a password. Unless `make clean` is executed, which removes the password file, then XV6 should ask to set the password again.

Hints:

- You should not let the user run any commands until they have entered the correct password. So, a reasonable place to put your code is inside of *init.c*, before the shell is started.
- You can use the `random()` function to get a random 32-bit integer for the salt. Because the salt is 16 bytes (128 bits), you will have to get multiple random integers and use them to fill out the salt value.
- Look at *bcrypt.h* for documentation on the `bcrypt()` and `bcrypt_checkpass()` functions.
- All changes should be made in *init.c*

Important Note:

Due to the interactive nature of this problem, this problem will be completely manually checked by TAs. Anubis will not build or test for this question. Please test on your own.

Rubric

We ask that you thoroughly comment your code and compile your code without errors to receive full credit for the assignment.

Some things we check but not limited to …

Password Authentication	100 Points