File execution hijacking

Start with access to a low-level account, student, on a compromised system.

List out what’s in their home directory with command: ls -l

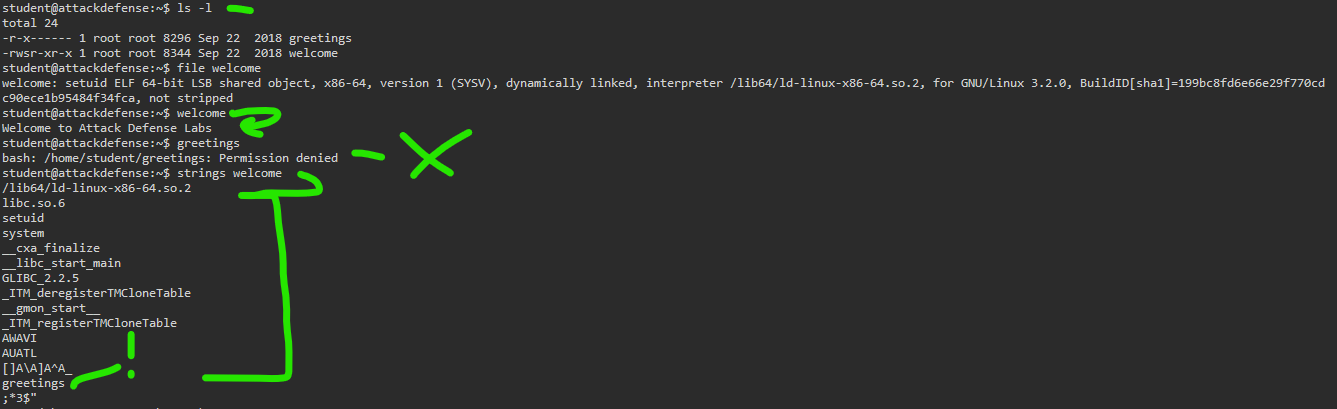
We find 2 files, “greetings” and “welcome”

We see a string echoed to the terminal when we run “welcome”

However, we lack the perms to run “greetings”.

Look at the strings that make up the file “welcome” with

Command: strings filename



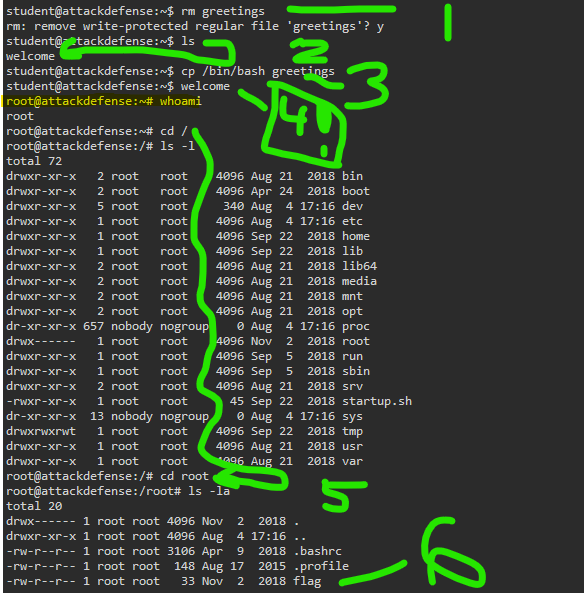
^^ In the screencap above, the second to last line calls greetings. At least, there’s a chance that whatever is going on in “welcome” calls the other file, “greetings”.

This might be useful.

1. Let’s see if we can remove the current “greetings” file and just make something else in its place.
2. List out contents, show that we only have “welcome” and the “greetings” file is indeed deleted
3. Copy the file for opening up a bash shell, and call it the target name “greetings”.

(remember that the strings within welcome indicate it will execute a file called “greetings”, so we just have to reassign something to that same name to get it to execute)

1. Run welcome, under the hood, it triggers the new “greetings”, which points to /bin/bash and opens a shell session as root. (hilit)



1. Change to root directory
2. Find the flag file