CITS3007 Secure Coding
Project tips

Unit coordinator: Arran Stewart

General tips

Break down the problem

1. read and write score records
2. do so from a function that uses `seteuid()`

Suggestions:

- Sketch out a solution to (1) in some other language, if you're not confident in C
- Prove to your satisfaction it works
- $\rightarrow$ convert implementation to C

Labs

Week 10 labs We look at *testing in C*, and will consider functions useful for the project

We will look at a way of creating “mock” files, handy for testing functions that operate on files.

Labs

Week 10 labs You don't have to use them, but:

- we will provide a *header file* of some useful functions (`curdle.h`)
- a skeleton for your `adjust_score.c` file

If you want to use the header file, you can submit code that `#include`'s `curdle.h`.

Labs

Week 10 labs We will provide some *test* cases

- **good** - scores files you should be able to pass
- **bad** - some invalid scores files
- **ugly** - valid, but contain unusual or “edge” case data

Scores file

- We know that a record in the scores file is always 21 characters long.
  - So a valid file will always be some multiple of 21 bytes
- It's easiest to treat the file as a binary blob in which we can **seek** (move to a position), read a blob of binary data, and write a blob of binary data

Error handling

- Your program isn't allowed to *crash* (e.g. segfault) or hang, but just bailing out with e.g. `exit(1)` is fine.
- Many C and POSIX functions (e.g. `open()`, see `man 2 open`, and `lseek()`, see `man lseek` can fail
- Good practice is to always handle these in *some* way, else our program is now in an unknown state
  - In real programs, we might want to `return` an error condition to our caller
  - In the project, it'll be fine to print an error message and exit

Wiki in MS Teams

- I'll create a Wiki of frequently asked questions in MS Teams

More testing

I suggest you *fuzz* your program using `afl-fuzz` or another fuzzer – I'll post in Teams FAQ on doing this.