Preparation

Prior to the official opening of the course, there is some brief introductory material that describes in some detail the course scope, content, and the expected background of learners. To help learners decide whether they have the right background, we have put together a qualifying quiz; failure to do well on the quiz (scoring a 70% or higher) suggests that the student may not be well-prepared to take the course without additional study.

Weeks and Themes

The core content of the course is as follows, with one topic per week:

1 **Low-level, memory-based attacks**, including stack smashing, format string attacks, stale memory access attacks, and return-oriented Programming (ROP)

2 **Defenses against memory-based attacks**, including stack canaries, non-executable data (aka W+X or DEP), address space layout randomization (ASLR), memory-safety enforcement (e.g., SoftBound), control-flow Integrity (CFI)

3 **Web security**, covering attacks like SQL injection, Cross-site scripting (XSS), Cross-site request forgery (CSRF), and Session hijacking, and defenses that have in common the idea of *input validation*

4 **Secure design**, covering ideas like threat modeling and security design principles, including organizing ideas like *favor simplicity*, *trust with reluctance*, and *defend in depth*; we present real-world examples of good and bad designs

5 **Automated code review with static analysis and symbolic execution**, presenting foundations and tradeoffs and using static *taint analysis* and *whitebox fuzz testing* as detailed examples

6 **Penetration testing**, presenting an overview of goals, techniques, and tools of the trade

Assessment

There will be an quiz each week covering the material presented that week. These quizzes will be due after two weeks at 8am ET (i.e., just prior to the release of that week's material).

There will also be three hands-on projects.

1 **Buffer overflow attacks**: The lab walks you through how a buffer overflow occurs, and how it can be exploited.

2 **Web application security**: The lab asks you to find and exploit common vulnerabilities in web applications, like SQL injection and cross-site scripting

3 **Static analysis for finding security bugs**: The lab will give you some experience using tools that aim to find security flaws automatically

Students will carry out the work of these projects at home, and to show that they have done so, will take a project-specific quiz.