Contents

Syllabus **1**
- Course Content . . . 1
- Course Structure . . . 1
- Course Grading . . . 1
  - Basic Objectives . . . 3
  - Challenge Problems . . . 3
  - Projects . . . 3
  - Final Exam . . . 4
  - Unassessed Work . . . 4
- Academic Integrity . . . 4

Syllabus

Course Content

We will cover the following topics this semester:

- Module 1: Programming Errors
- Module 2: Cryptography and Anonymous Communications
- Module 3: Networking
- Module 4: Economics and Human Behavior

There are also two "bonus" modules that we will not be covering, but you might find interesting: Digital Currencies and Fault Tolerance.

Course Structure

This is a *flipped classroom* course. That means the lectures are all pre-recorded, and class time is used for Q&A, demonstrations, exercises, and time to work on assignments.

Most assessed work has a soft deadline. You are strongly encouraged to complete the work by the due date, so that you have time for resubmissions. Class time will focus primarily on the current material, but there should be ample time on most days for questions regarding material from any point during the course.

Course Grading

This course employs **Standards-Based Grading**, rather than traditional points-based grading. Rather than accumulating points that are then used to determine the cut-offs for various letter grades, there are specific *learning objectives* for students to demonstrate mastery of. A final grade in the course is determined by how many of these objectives were met, and at what level.

There are three types of learning objectives:

1. **Basic Objectives (20 in total):** These allow the student to show that they have a basic understanding of a particular objective within a module. They are graded out of 5 points.
2. **Challenge Problems (21 in total):** There are 15 module-specific challenges. These are only available for a module once a student has successfully met at least one of that module's Basic Objectives. These cover a module's content in more detail, and across the different basic learning objectives. In addition, there are 2 project write-ups graded as challenges, and 4 Final Exam questions. They are graded on the **EMRN** scale:
   - **E:** Excellent; demonstrates mastery of the material
   - **M:** Meets Expectations; approaching mastery, with minor updates required
   - **R:** Requires Resubmission; demonstrates progress towards mastery, but additional learning and major updates are needed

- **N:** Not Assessable; does not indicate that progress has yet been made

3. **Projects (4 in total):** These are practical applications of the course concepts.

We discuss the different types of learning objectives in more detail below, as well as specific requirements and restrictions, but here is the breakdown of the minimum standards for the letter grades:

Grade	Basic Objectives	Module Challenges	Write-up Challenges	Final Exam	Total Challenges	Projects
A	5/5 on all 20	1 M from all 4 modules	M on Build-It, E on Break-It	2 M	17 M, 10 E	100% on all 4
B	5/5 on all 20	1 M from all 4 modules	M on Build-It and Break-It	1 M	12 M, 5 E	100% on all 4
C	4/5 on all 20	1 M from 3 modules	R on Build-It and Break-It	-	7 M	10% on Cryptography, 100% on other 3
D	4/5 on 14	-	-	-	-	3 projects meeting C requirement

In addition:

- There is no C-
- A C+ must meet all C requirements, at least 1 write-up must receive an M, 10 Challenge Problems must score an M, 3 Challenge Problems must score an E, and at least one Final Exam question must score an M
- A B- must meet all C requirements, with at most 1 of the following being true:
  - 5/5 on 19 Basic Objectives
  - M on 1 Write-up
  - M on 11 Challenge Problems
  - E on 4 Challenge Problems
- A B+ must meet all B requirements, at least one write-up must meet the A standard, 15 Challenge Problems must score an M, and at least 8 Challenge Problems must score an E
- An A- must meet all B requirements, with at most 1 of the following being true:
  - 1 write-up meeting the A standard
  - M on 16 Challenge Problems
  - E on 9 Challenge Problems

There is a Chrome browser extension to monitor your progress towards these goals. It also helps you track what assignments are due when; you should definitely use it. Here's how to insall it:

- Download the zip file, and expand the student_status-2026.8.0 directory.
- In Chrome, go to "Manage extensions", and enable "Developer mode".
- Then click on "Load unpacked" and select the expanded directory.
- For convenience, you might want to pin the extension so that it always appears.

When selected, the extension will open a new window that will show your current progress towards various letter grades. You must be logged into ELMS from this Chrome browser for the extension to work. It will automatically refresh every 10 minutes, though you can manually reload the data from ELMS. The source code is available from https://gitlab.cs.umd.edu/cmsc414-marsh/studentstatus.

There are also "pseudo-assignments" in the gradebook that track the following:

- Basic Objectives: Sufficient - How many of the Basic Objectives score at least a 4/5
- Basic Objectives: Completed - How many of the Basic Objectives score a 5/5

- Challenges: Meets Expectations - How many of the Module Challenges, Write-ups, and Final Exam questions score at least an M
- Challenges: Excellent - How many of the Module Challenges, Write-ups, and Final Exam questions score an E
- Challenges: Modules - How many of the Modules have at least one Challenge scoring at least an M
- Projects: Sufficient - How many of the projects meet the requirements for a C
- Projects: Completed - How many of the projects score 100%
- Write-ups: Sufficient - How many of the write-ups (there are 2) meet the requirements for a C
- Write-ups: Good - How many of the write-ups meet the requirements for a B
- Write-ups: Excellent - How many of the write-ups meet the requirements for an A

For your information here is the actual ~~contract script~~ that I am using to compute your grade. Note that you will not be able to run it, but the logic should help clarify any ambiguities from above.

Basic Objectives

Each module is divided into a number of distinct learning objectives, of which there are 20 in total. An objective is assessed as *Completed*, *Sufficient*, or *Not Completed*. As an example, within the Programming Errors module, one objective would be Stack Vulnerabilities and another would be SQL Injection.

For each basic learning objective, there is a quiz containing a set of 5 questions, randomly selected from a question bank. A student must get ***4*** correct to successfully complete the objective at the C level (*Sufficient*), and ***all 5*** correct to complete the objective at the B or A level (*Completed*). Multiple attempts are permitted.

Challenge Problems

Once a threshold of basic objectives have been met for a module, that module's challenge problems become available. These are more in-depth questions about the material, that allow a student to show an understanding of the material. Most of these are free-response questions.

*You **must** cite all of your sources, including course materials. Anything you quote must be appropriately indicated (with quotation marks or block quotes), with citations. Your submission must not be substantially quotations – you must demonstrate independent thought.* We do not specify a citation format, as long as it is clear. **Please note that LLMs provide unreliable information**, regardless of how convincingly they do so. If you are going to use an LLM as a research tool in your response, you ***must*** ensure that the information is correct and addresses the actual question asked. The LLM response must be treated as any other external reference: indicate what you are quoting or paraphrasing, and cite the LLM, *including the prompt or prompts used*. An LLM ***cannot*** be the sole source of information: If you are going to use an LLM, you must also include supporting citations. **If you are using an LLM to improve your writing, after your submission you *must* add your *initial text* and any *prompts to* and *responses from* the LLM.**

There are also additional challenge problems, associated with the team project and the final exam. These will be discussed separately. In total, there are 21 challenge problems (including project and exam problems).

Because learning is a process, students may resubmit challenge problems as they gain mastery of a topic.

Projects

There are three individual projects in the course. For each, there is a due date, at which point students' submissions will be scored, and feedback provided. You may also submit projects before the due date to receive feedback. You may resubmit these as many times as you need to. **In order to pass the course with at least a C, all of the individual projects *must* receive perfect scores, except for Cryptography.** While this might sound intimidating, experience shows that most students have either minor errors or did not grasp a key concept of the project. By revisiting the project until it works, you solidify your understanding

of the material. Due to the added complexity of the Cryptography project, there is a lower requirement for that one to receive a C (but not a B or higher).

For the first two projects, there are in-person follow-ups, in which you will have 15 minutes (plus any ADS accommodations for extra time) to solve a related problem. These must be done in-person, and cannot be completed over Zoom. You will not have access to notes or other references during these follow-ups. *Your project is not considered to be 100% completed until you have completed the follow-up.*

When resubmitting a project, you must also include a brief discussion of what enabled you to make progress. This can range from "I was unable to devote sufficient time to the project before the initial deadline" to "I fundamentally misunderstood how X behaved, and have since learned…" No excuse needs to be given, merely an acknowledgement that more time was needed.

There is also a two-phase team project, structured as a build-it/break-it. In the first phase, teams design and implement a system. In the second phase, students *individually* attack other teams' phase-one implementations, looking for vulnerabilities.

The build-it phase is scored in two parts:

- **Basic functionality according to the specification:** As with the individual projects, this must (eventually) receive a perfect score for the students on the team to receive a C- or better.
- **Secure Design Document:** This is assessed as a Challenge Problem

The break-it phase is an evaluation of the build-it submission for some team the student was not a member of. It is assessed as a Challenge Problem. The targets for the break-it phase will be those build-it submissions made at the nominal deadline that reach a predetermined threshold on the basic functionality tests.

The project challenges are assessed on the same scale as the module challenges.

*You may **not** use Generative AI, such as ChatGPT, to produce your code, and you may not upload the code we provide to any such system (this would be a copyright violation). If you are using code you find elsewhere, you must cite it in the comments.*

Final Exam

The final exam is a set of four questions that you might be asked on a typical interview for a computer security-related position. Each of these is assessed as a separate Challenge Problem, using the same 4-point scale. Students who do not need the additional challenges to reach their final grade goal are not required to take the final exam.

Students will not be able to resubmit final exam questions. The final exam time is also the absolute deadline for any resubmissions of prior auto-graded work. Challenge Problem resubmissions must be made by the day before the final, though we ask that you complete all of your resubmissions well before this.

Unassessed Work

There are also a series of in-class exercises and homework mini-projects. These are for your benefit as a way to reinforce concepts from the lectures, and are likely (in some cases) to help with the projects.

Academic Integrity

You are expected to adhere to the University's academic integrity policy. All work that is not explicitly stated to be a team effort must be done by you alone. *You have 24 hours after the deadline of any graded work to self-report lapses in academic integrity*, which we can then hopefully resolve without a reference to the Office of Student Conduct.

Posting of any project/exercise implementation (even after the course is over) in a publicly available online location (e.g., github or sourcehut) is prohibited under the Code of Academic Integrity (facilitation of academic dishonesty). Any student responsible for this will be reported

to the Office of Student Conduct and risks the sanction of an "XF" in the course. You may post your code on a public site such as a GitHub or SourceHut private repository to share with potential employers, as long as access is appropriately restricted.