CS1660/CS1620/CS2660: Collaboration Policy

This document outlines the CS1660/CS1620/CS2660 policies regarding collaboration and the acceptable use of skills taught in this course. Please read this policy carefully, as it likely differs from policies in previous Computer Science courses you have taken.

By submitting any assignment in this course, you acknowledge that you have read and agree to this policy. If you have questions about the policy, please consult the course staff.

1 Introduction

Our goal is to help you learn security principles and develop the skills to identify, analyze, and discuss security threats. Collaboration is an important part of building secure systems, as well as the learning process. At the same time, by the time you leave this course, we want to make sure you have internalized the material yourself—we want every student to have worked on every project and homework and have encountered the intellectual challenges it poses.

Therefore, we have adopted a policy that generally encourages teamwork while establishing a few boundaries to help make sure you build your own understanding.

This policy is specific to CS1660, CS1620, and CS2660. If you have questions about this policy, please raise them with the instructors or TAs. The consequences of violating this policy can be severe, in accordance with Brown's Academic Code policies.

2 Assignment-specific policies

2.1 Projects

Note: Individual projects may have specific policies that differ from those listed here. Any project-specific policies will be listed on that project's handout.

Projects are submitted individually, though you are encouraged to discuss project topics with other students, so long as you are not directly revealing specific solutions to a problem. You may even work out high-level solutions together at a conceptual level, or look at parts of others' code to help them debug minor errors or environment setup issues. However, as you collaborate with others on homeworks, you must abide by the following rules:

1. **Any code you submit must be your own work (ie, written by you)**. You are responsible for understanding all code that you submit. Typing out a line of code verbatim that you see on a whiteboard, an online source, or an AI tool, when you don't know what it does is not acceptable, as this impedes your own learning process. We encourage you to use online resources, AI tools, and any other resources to *help build your understanding*, but in general you should not be copying code from them (for more info, see here).
2. **You must not directly transmit code to another student, or permit someone else to use your code directly**. To help ensure everyone follows rule #1, you should not send code, or screenshots/photos of your code, to another student via any medium (email, text, Snapchat, Tiktok, `netcat`, ...), or allow someone else to take a photo of your computer screen. Taking paper notes or whiteboard photos from a group discussion (or during TA hours) is fine, but you should not be copying any notes/photos directly into your code. If you find yourself wanting to rely on this, you should consult with the TAs or the instructors to check your understanding of the concepts involved—we can work with you to help build any skills you may be missing.

3. **When turning in your work, you must list anyone you collaborated with and what you worked on together**. Your final submission should include a file called `COLLABORATORS.{txt,md,}`. In this file, you must list the names and CS logins of anyone you worked with and include a brief description of what part(s) of the project you worked on together. Collaborating with someone on part of an assignment is not an excuse for not understanding it. We reserve the right to ask you to discuss your solutions with us if we are concerned about your work.

4. **When collaborating, your discussions should generally take one of two forms**:

   (a) **"High-level" discussions about course concepts or how to approach problems**: we encourage you to work together to figure out how to get started and develop a strategy for solving a problem. After that, you are responsible for building your own implementation and understanding how it works.
   Note: Certain projects may restrict how you can discuss your approach to problems with other students, particularly where the assignment goal is to find and exploit vulnerabilities in project code—more details will be provided where applicable.

   (b) **"Low-level" questions about programming languages, tools/environment issues, debugging techniques, dealing with errors, Linux questions, etc.**: Working in security involves using a *lot* of different tools—we don't expect you to know every detail of how they work, and we encourage you to work with your peers (and course staff!) to learn how to use them! It's always okay to ask someone about specific details of a tool, language, or library you're using (eg. *"What does this error mean?"*, *"How do I do X in a Bash script?"*). In general, for any non-assignment-specific issue you might search for online, it's also okay to ask someone else about it, too.
   For low-level issues like these, it's okay to look at someone else's code to help figure out a specific problem, or build a strategy for debugging it, so long as this does not i) directly reveal an exploit or solution, or ii) involve copying code without understanding it.

3 Homeworks

Similar to projects, you are encouraged to discuss any aspect of the homework problems with other students, so long as:

1. **The solutions you write down must be your own work (ie, written by you)**. You are responsible for understanding everything you submit—it is not sufficient to copy a solution from a whiteboard, group discussion, or homework clinic, without understanding it. As a general guideline, it's okay to work on problems with someone, but the writeup you generate should be your own (even if you're typing separately at the same table)–if you find yourself copying words or phrases verbatim, you should stop and make sure you are understanding the problem well enough to express it in your own words.

2. **When turning in your work, you must list your collaborators**. In your writeup for each problem, list the names and CS logins of anyone you worked with and include a brief description of what you worked on together. Once again, collaborating with someone on part of an assignment is not an excuse for not understanding it. We reserve the right to ask you to discuss your solutions with us if we are concerned about your work.

4 Online and AI resources

We encourage you to consult online resources, as well as AI resources (ChatGPT, Honeybear, etc.) to help write your own programs, *so long as doing so does not completely trivialize the assignment*. A big part of security work—and software development in general—is knowing *how* to use these resources to find the pieces of the "puzzle" for the problem you're trying to solve, determining *if* they're the right pieces, and then *adapting and connecting them together* to solve the problem at hand. We want you to have this experience, as it mimics how you will be solving problems and analyzing systems outside this course.

Concretely, this means that you are welcome to use online tools to look up information, but **the process of "putting the pieces together" to complete the assignment's learning goals must be your own work**.

Example code In this course, you are likely to encounter programming languages and mechanics that you have not seen before—we encourage you use online resources to help understand these, including finding code examples. While copying code outright from online sources is not generally permitted, you may use *small, generic snippets* not specifically related to any assignment. For example, *"how do I concatenate strings in PHP?"*, *"how to do X in bash?"*, or *"how to make an HTTP request from the command line?"*, are all okay, *"how to implement per-user salted hashing [project 1]"* is not. When using examples you believe are nontrivial, you should cite your source with a comment or a note in your README.

Overall, when working from code examples, you should have an understanding of *what* the code is doing and *why* it's going into your project—this will help ensure that you have the necessary conceptual understanding to use what you learned in another context. When grading, we may follow up with you about your work in order to check your understanding.

5 Outside assignment help

You are not permitted to make posts on online discussion boards, Q&A sites, or other forums asking for others to solve problems in assignments for you. Similarly, you are not permitted to consult other people (in or outside the course) to complete assignments on your behalf.

6 EdStem Discussions

We use EdStem (aka, Ed) as a platform for discussion and asking questions. As a "public" (within the course) discussion forum, Ed can be an excellent resource for providing clarifications about various course topics and assignment mechanics.

Public posts We encourage you to ask *public* clarifying questions on Ed, so long as your questions to not reveal details about assignment solutions. For example, it's okay to ask about clarifications on assignment mechanics, high-level questions, help understanding error messages or debugging strategies, or questions about any course concepts in lectures.

In general, small amounts of code can be included in public posts if it's generic enough not to reveal solutions for an assignment. For help debugging specific errors, it's often helpful (both to you, and anyone reading the post!) to create a generic example that demonstrates a problem (similar to posts you see on StackOverflow).

If you're ever unsure if it's okay to make a public post, please post privately. If we believe your post would benefit the class as a whole, we will ask if we can make it public (as an anonymous post).

Private posts Can be used for questions that might reveal your solutions, or specific issues related to grading or logistics for review by the instructors and HTAs.

Please do not request extensions or request accommodations via Ed—this information is considered sensitive and should only be sent only to the instructors at `cs1660-profs@lists.brown.edu`.

7 Protecting your work

You must ensure that your solutions will not be visible to other students. All assignment code is distributed using Github Classroom, which automatically gives you a private repository you can use for your work.

If you store your work elsewhere, such as a private version control repository, you must ensure your account is configured so your solutions are not publicly visible. (For example, GitHub, offers free private

repositories.) Leaving course projects in a place where they are visible to other students (such as in world-readable folders on the department filesystem or public GitHub repositories) constitutes a *Collaboration Policy* violation, and can result in penalties even if they are discovered after the course.

8 Responsible security practices

Students are expected to use the techniques they learn in this class in a responsible manner. Some of the techniques covered in this course for educational purposes are unethical and/or illegal to use and apply in contexts beyond the course itself. Breaking into, misusing, or harming computer systems or networks is illegal and punishable by law if done without the explicit authorization of the owner. Attacking technical systems (whether owned by Brown or by others), except as specifically assigned in this course, is a violation of Brown's Acceptable Use of IT Resources Policy and may lead to disciplinary action. The consequences of violating this policy can be severe—similar to, or greater than, a violation of the Academic Code.

If you have any questions about what kind of conduct is legal and/or ethical, please contact the instructor and/or the TA staff first.