

CS 507 Final Project Instructions

David Heath

Fall 2026

Abstract

For your final project, you will form a team of three individuals, choose a topic in the field of cryptographically secured computation (e.g., MPC, ZK, ORAM, etc.), select research papers related to that topic, read those papers, write a literature review, and present your topic to the class.

This document provides a rubric, as well as a template for your review. Your review should be 15–25 pages long (not including your bibliography) and typeset in a single column with 11pt font. You may use the format of this document.

Your presentation will be 15–18 minutes long. You will discuss your topic, with an emphasis on *exposing* your peers to the key ideas of your exploration. 15 minutes is not enough time to give *all* technical ideas, so focus on clearly presenting high level insights.

You are not *required* to use this template, or even to use L^AT_EX, though I recommend it.

1 Summary

For your final project, you will select a handful (e.g., 1–3) of academic papers as the focus of your review, and write a 15–25 page (not including citations) review on these papers. Your goals in this review are to:

- Explain the context of your chosen topic. Why do cryptographers care about your topic? How does your topic relate to topics discussed in class? Are there any applications today? Could there be applications in the future?
- Describe the history of your chosen topic. What were the major advances in your topic? What is the current status of your topic? Do there remain unsolved problems? Are there trade-offs between different advances?
- Sketch the main technical ideas underlying your topic. Choose a construction/proof related to your topic, and sketch how it works. Here, I am expecting to see a clear explanation, but I do not necessarily need to see a full rigorous write-up. You are demonstrating that you understand the topic well.

View your reader as a knowledgeable cryptographer who may have heard of your topic, but who has not studied it in detail. Your reader wants to learn the main ideas, and they want pointers to strong follow-on materials where they can learn more.

Your review is due at midnight on **the last day of course instruction (December 9)**.

You will also present your topic. Your presentation will be 15–18 minutes long. You will **make and submit a pdf slide show**. We will reserve 3 presentation days:

- December 1, 3, and 8.

1.1 Rubric

Your first task is to *form teams* and start thinking about a final topic. **Form your team and discuss your topic with me by Friday, October 30.**

Your grade will be weighted as follows:

- 80% write-up.
- 20% presentation.

Both deliverables will be graded according to the same rubric. I am looking for the following, with equal weight:

- **Context/History.** You clearly describe the “what” and “why” of the topic. You explain why experts care about the problem, and how the topic has progressed. An A+ review will connect the topic to other areas of cryptography/computer science, e.g. by explaining how the chosen topic compares to other topics, especially those discussed in class.
- **Mastery.** You clearly describe the “how” of the topic. Technical details are laid out, and such details are clearly explained and free of errors. You clearly demonstrate that you understand ideas in the selected topic. In the presentation, you are able to answer non-trivial questions about your topic. *Note:* there is simply *no way* you will be able to explain *all* technical ideas. Focus on ideas that seem central. We prefer clear explanation of a few ideas over bad explanations of many ideas.
- **Clarity.** Your review is clear and easy to understand/You give a clear and interesting presentation. Your classmates should learn something. Your writing/slides are clear and precise. Please avoid putting “walls of text” in slides.

The rest of this document provides some tips for organizing your review, as well as potential topics of investigation.

2 Introduction

Write an introduction that is high level, clear, and reasonably short. What is the problem? What are the existing solutions? What is your review going to talk about?

It can be useful to add labels to your sections so that you can refer to them by name. As an example, I can forward reference to Section 3.

3 Preliminaries

If necessary, include a section that covers background material required to understand your topic.

3.1 Organization

I encourage you to use sections/subsections/paragraphs to organize your document.

Equations. When writing mathematics, avoid writing math inline unless it is short and clear. For example, it is acceptable to write $x + y$ inline, but for longer expressions or equations, consider breaking out designated lines:

$$\begin{array}{ll} (a + b) + c = a + (b + c) & \text{associativity} \\ a + b = b + a & \text{commutativity} \end{array}$$

4 Citations

I expect you to include professionally organized citations to peer-reviewed, reputable sources. As an example, let's cite the classic GMW paper [GMW87]. Please do use alphabetic citations, consistent with the **alpha** bibliography style in this document.

The **CryptoBib** database provides an *excellent* collection of references to high quality cryptographic references.

It is acceptable to cite non-academic resources (e.g., blog posts, web pages, GitHub projects, etc.), but your report should focus on the academic literature. As an absolute bare minimum, make sure you cite at least five academic papers.

Where to look for citations? When searching for resources, the IACR **ePrint** archive provides a high quality collection of freely accessible papers. Most (though not all) cryptographic papers appearing at top venues are on ePrint. Not all papers on ePrint are peer reviewed.

A simple Google Scholar search can be an effective way to find related works. Start from a paper that interests you. Then, (1) look at the papers the authors cite and (2) use Google Scholar to find papers that cite them. With this

methodology, you can quickly build up a web of papers and figure out which topics interest you.

Which citations are reputable? If a paper appears in one of the following journals/venues, we will consider it reputable:

- Any IACR conference, especially IACR Crypto and IACR Eurocrypt.
- The Journal of Cryptology.
- Reputable security conferences, e.g. USENIX, CCS, IEEE S&P.
- Reputable conferences on the theory of computing, e.g. STOC or FOCS.

This is *far* from an exhaustive list, and resources that appear elsewhere may also be reputable. If in doubt, just ask me.

5 Collaboration

Your project is to be completed in small teams. I therefore recommend writing your report in L^AT_EX via Overleaf. Overleaf will allow you to write the paper in parallel. Just make a new project, share with your collaborators, and, if you like, you can upload this template to your project.

6 Potential Topics of Investigation

Your job is to deeply investigate a topic in cryptographic secure computation. There are *many* suitable topics. The following is far from an exhaustive list, and you are encouraged to skim the literature for topics of interest.

- **MPC on asynchronous networks.** So far, the protocols we have considered assume parties have secure point-to-point channels that deliver messages within a fixed amount of time. However, the reality of the internet is much messier than this. Protocols designed for *asynchronous* networks attempt to handle a more realistic (and more challenging!) network setting. What are the challenges when designing protocols for the asynchronous setting? What changes in terms of what is possible?
- **Round optimal MPC.** When studying garbled circuits, we learned that it is possible to achieve protocols that complete in only a constant number of rounds. But how low can that constant be driven? As the number is set lower, constructing protocols becomes harder, and eventually impossible. There are many works that investigate how to construct protocols that carefully optimize the number of protocol rounds. How low can round complexity be pushed? What do the protocols that achieve the lowest number of rounds look like?

- **Optimizing oblivious transfer and correlated randomness.** We saw that OT is complete for secure computation, and that protocols like GMW consume one OT per AND gate. A naive implementation might therefore need millions of public-key operations. *OT extension* shows how to stretch a few hundred “base” OTs into millions using only symmetric-key cryptography, and more recent work generates OTs, and other useful correlations such as Beaver triples, with communication *sublinear* in the number of correlations produced. How do these constructions work, what assumptions do they rest on, and what does it actually cost to produce a billion AND gates’ worth of randomness today?
- **Honest-Majority MPC.** When a majority of parties are honest, much becomes possible: information-theoretic security, guaranteed output delivery, and protocols that cost a handful of field elements per multiplication. The three-party case with one corruption is especially well developed and underlies many deployed systems. What makes honest majority so much cheaper?
- **Fairness and Guaranteed Output Delivery.** We settled for security with abort, since an adversary who learns output first can simply stop. But the story is subtler: some functions *can* be computed with complete fairness even between two parties, abort can be made *identifiable*, and with an honest majority output delivery can be guaranteed. What exactly separates the possible from the impossible here?
- **MPC-in-the-Head and Post-Quantum Signatures.** We have seen that the IKOS paradigm allows one to transform MPC protocols into zero-knowledge proofs. A line of work has tuned the underlying protocol to shrink proofs and prover time by orders of magnitude, and the resulting proof systems are now the basis of signature schemes that remain secure against quantum attackers. Which design choices in the MPC protocol matter for the proof, and why? What is the state-of-the-art in this line? Why does this line enable post-quantum security?
- **Interactive Zero Knowledge from VOLE.** A different route to efficient ZK uses a form of correlated randomness called *vector oblivious linear evaluations*. The resulting proofs are streaming, require only a few bits of communication per gate, and scale to circuits with billions of gates. How do these protocols work? How do they trade off with other approaches to ZK?
- **Oblivious RAM.** Our protocols operate on circuits, but real-world programs use random access memory. Later in this course, we will introduce a cryptographic tool called *oblivious RAM* (ORAM), which hides the access pattern of a program to memory, and can help to implement MPC random access memory. After decades of work both the constructions and the lower bounds for ORAM are nearly tight. What are the main

constructions? What does the lower bound say? What are the remaining gaps?

- **Alternative Security Models.** Semi-honest and malicious are two points on a spectrum, and both can be the wrong model for a given deployment. *Covert* security guarantees only that a cheating party is caught with some fixed probability, say $1/2$, which is enough to deter an adversary who would be fired or sued if exposed, and it costs far less than full malicious security. *Publicly verifiable* covert security strengthens this so the honest party can prove the cheating to a third party. Other variants weaken the adversary in different ways: adversaries who are *rational* (i.e., act in their own interest, as opposed to arbitrarily), or adversaries who cannot collude. What do such relaxations buy? What do they give up? When are they justified?
- **Adaptive Security.** Our definitions force the adversary to choose whom to corrupt before the protocol begins. An *adaptive* adversary instead corrupts parties as it goes. Surprisingly, many natural protocols are not known to be adaptively secure without significant modification. Why is adaptive security so much harder, and what does it cost?
- **MPC-Friendly Primitives.** As we have seen, non-black-box cryptography is a powerful, but expensive, technique. One reason for this high cost is that it entails using a cryptographic primitive in a setting for which it was not designed. Accordingly, a line of work has pursued designing cryptographic primitives specifically with non-black-box evaluation in mind. How are these primitives designed, how are they attacked, and how much do they actually save?
- **Private Set Intersection.** Two parties each hold a set and wish to learn the intersection, and nothing else. In principle this is just secure computation of a particular function, but the function is special: it is almost entirely comparisons, there is no arithmetic, and the sets can hold millions of items. The community has developed custom protocols for this setting, the most efficient of which run within a small factor overhead of the insecure baseline, where parties exchange sets in the clear. How do such protocols work? Why are they so much cheaper than general-purpose MPC?

Other topics one might consider include:

- Private information retrieval.
- Garbled RAM.
- Function secret sharing.
- Homomorphic secret sharing.

- Threshold signature schemes.
- Privacy-preserving machine learning.
- Composable security definitions (the UC framework and related frameworks).
- Fully homomorphic encryption and its relation to secure computation.
- Garbled circuit optimizations and lower bounds.
- ... And many more.

References

- [GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In Alfred Aho, editor, *19th ACM STOC*, pages 218–229. ACM Press, May 1987.