591-04 Cybersecurity Class Syllabus

Jared Saia

1 Class Overview

In this class you will (hopefully) learn the following: basic cryptographic primitives and what can be proven about them and how to use these primitives to create algorithms which have provable security and privacy guarantees; techniques for designing distributed data structures and algorithms which have provable security and privacy guarantees even when many of the nodes in the network are controlled by an adversary; techniques for designing mechanisms for several agents which are robust even when all players behave selfishly.

In general, this class is about designing algorithms which *provably* maintain certain desirable properties even under attack. The class will be roughly divided into two parts. In the first part of the class, where we will focus on traditional cryptography, we will mostly focus on designing robust algorithms for two agents. The desirable properties we want to maintain will be e.g. privacy, authentication and identification and the attack will be by an outside adversary who is trying to break our algorithms.

In the next part of the class, we will focus on designing robust distributed algorithms and data structures. The desirable properties we want to maintain will be e.g. efficiency and correctness of the algorithms and data structures and privacy of the individual agents. The attack we will be concerned about will typically be an inside job: the adversary takes over a certain number of nodes in the network and uses them to try to break our algorithm or data structure.

Time permitting, we will spend a week or so in the class talking about designing good algorithms for economic problems. The desirable properties we want to maintain will be e.g. economic fairness. The attack will be completely internalized: we will assume that *every* agent is selfish and is trying to exploit the algorithm for its own economic benefit.

We will use several interesting mathematical techniques in the class including: probability theory (e.g. linearity of expectation, random walks and Chernoff bounds), expander graphs, number theory, Sperner's Lemma, etc.

2 Class Deliverables

The deliverables for the class will consist of a project and presentation, homeworks and a take-home final. The tentative weighting of grades will be as follows.

1. Project and presentation (30%)
2. Homeworks (30%)
3. Final (Takehome) (30%)
4. Participation (10%) (participation in class discussions via asking questions, making comments, etc.)

2.1 Project and Presentation

A significant part of this class is the class project. In this project, you will apply mathematical tools learned in this class to solve an algorithmic problem. The project must have some analytical component to it where you demonstrate mastery of mathematical tools learned in this class. I also recommend that the project have an empirical component where you do empirical tests which support or complement your analytical results (It's good to have an empirical component in case you don't get the theoretical results you're trying for).

There will be two main deliverables for the project: a paper and an in-class presentation. The paper should be no more than twelve pages in length (not including bibliography and appendix). This paper should be structured as a standard research paper in that it should have an abstract, an introduction, a related work section, a body (this could contain for example a section on algorithms, a separate section on analysis and a separate section on empirical results), and a conclusion and future work section. The presentation will be 20 minutes in-class with 10 minutes for questions and answers.

You can choose to do a project with more of an empirical or theoretical focus. One type of empirical project would involve implementing an algorithm or protocol described in class, then empirically determining how this algorithm performs (e.g. in terms of robustness and resource costs) under a certain set of attacks, and then finally comparing these empirical results with the analytical results. Another more challenging (and perhaps more interesting) project would involve simplifying an algorithm discussed in class so as to improve its resource costs empirically and/or analytically. To do this, you may need to make some additional (hopefully not too strong) assumptions. The theoretical component of such a project would be to prove that your new algorithm is still secure and the empirical component would be to implement your new algorithm to verify that it performs well empirically. Another type of project would involve designing an algorithm for a variant of some problem discussed in class (or a new security problem of your own formulation) and then proving that your algorithm is secure for some definition of security.

In general, for the class projects, I will be more excited about partial progress on a hard problem than a complete solution to an easier problem. You may work in groups of 2 or 3 on the project or you may work individually.

2.2 Homeworks

You are encouraged to work in groups of 2 or 3 people on the hws. Each group should turn in a single write-up. Make sure you put the names of all group members at the top of the hw. Some of the hws will be related to the readings and some of them will be sets of problems.

Please cite whatever sources you use for the hws.

Note: the projects and homeworks can be done in groups but the final must be done without any collaboration.

2.3 Readings

Reading papers from good conferences and journals is a critical part to doing research. Research is a social activity and reading papers is one way to stay connected with the research community. Specifically, it has the following benefits:

- It'll help you identify those problems that the research community is excited about and those problems it feels are hard
- It'll help you learn new mathematical tools which you can then apply to your own problems
- It'll keep you from redoing work that people have already done in the community

Some of the hws in this class will be related to writing up summaries of reading assignments. These will be 1-2 page summaries that you can write up in a group of 2-3 people. You will only need to turn in one summary per group. In these summaries, I'd like you to answer the following questions about the papers.

- What is the abstract problem being solved? How closely does this problem relate to a real-world problem of interest? Can you think of a new problem formulation that more closely matches an interesting real-world problem and is still simple enough that it's likely to yield to mathematical solution.
- Assumptions: What are the key assumptions being made? Are they realistic and simple to state?
- Results: What are the major results of the paper? If the paper describes an algorithm, is the result primarily theoretical, e.g "proof of concept", in that it shows the problem can be solved in theory, but the algorithm proposed is too complicated or slow to be useful. Alternatively, is there a reasonable chance that the algorithm could be used in practice? If the paper gives a negative result, how surprising and pertinent is the negative result?

- Mathematical Tools: What are the main mathematical tools used in the paper? Are there new tools introduced from other fields? Old tools used in new ways? New mathematical machinery created to solve the problem?(note: this is rare)

- Open Problems: How would you extend or improve the results of the paper? Don't just copy from the future work section of the paper. Think about oversimplifications the authors may have made. Are the algorithms overly complicated? Is it possible to improve on the resource costs achieved? Can you apply the tools used in the paper to another interesting problem?

2.4 Policies

Assignment deadlines are strict: late homework will automatically receive a grade of zero, unless reasonable cause can be shown (which is easy for one, possible for two, and very hard for three or more!); no make-up.

Collaboration is encouraged on all of the homeworks. Usual university policies for withdrawals, incompletes and academic honesty.

3 Texts

There will is no official text for this class. However good reference texts are: "Lectures on Data Security : Modern Cryptology in Theory and Practice" (Lecture Notes in Computer Science) by Ivan Damgard(Editor) , "Foundations of Cryptography I" by Oded Goldreich and "Randomized Algorithms" by Raghavan and Motwani.

We will also make use of lecture notes on the web for similar classes and from research papers. Class readings will generally consist of lecture notes and papers.

4 Class Outline

The following list includes the class topics along with readings for each topic. You should be able to get all of the papers online using either google scholar or citeseer.

Note: many of the lectures will be based on lecture notes from Jonathan Katz's class which are available here:

`http://www.cs.umd.edu/~jkatz/TEACHING/crypto_F02/lectures.html`

- Public Key Encryption and Semantic Security. Readings: roughly Lectures 1-4

- The Random Oracle Model: signature schemes and public key encryption in this model. Readings: Lectures 14-16

- Zero-Knowledge Interactive Proofs. Readings: Lectures 17-25 (minus guest lectures)

- Secret Sharing and applications *Readings: TBA*
- Byzantine Agreement and Leader Election: Readings:
  - Lectures 26 and 27
  - "Scalable Leader Election" by V. King, J. Saia, V. Sanwalani and E. Vee
  - "On Reliable Broadcast in a Radio Network" by V. Bhandari and N. Vaidya
- Secure Multi-party Computation: Readings TBA. We will also likely read some papers on Secure Voting protocols.
- Attack-Resistant Networks. Readings:
  - We will review some basic probability theory. A good background for this is Chapters 1-5 of the book "Randomized Algorithms" by Raghavan and Motwani
  - "Censorship Resistant Peer-To-Peer Content Addressable Networks" by Amos Fiat and Jared Saia
  - "Butterflies and Peer-to-Peer Networks" by Mayur Datar
  - "A Simple Fault Tolerant Distributed Hash Table" by M. Naor and U. Weider
- Game Theory: Designing mechanisms which are robust to attack by selfish coalitions. Readings: "Rental Harmony: Sperner's Lemma in Fair Division" by Francis Su
- Applications. e.g. negative databases, private information retrieval, etc.