Homework 1: Crypto Party

Due: Thursday, February 27 @ 11:59 pm EST

Overview and instructions

This homework has 5 problems:

- Problems 1–4 are required for all students
- Problems 5 is required for **CS1620/CS2660 students only**

Note on collaboration

You are welcome (and encouraged!) to collaborate with your peers, but the solutions you write down must be **your own work** (ie, written by you). You are responsible for independently understanding all work that you submit—after discussing a problem as a group, you should ensure that you are able to produce your own answers independently to ensure that you understand the problem. For more information, please see the course Collaboration Policy.

In your submission, we ask that you include a brief *collaboration statement* describing how you collaborated with others on each problem—see the next section for details.

How to submit

You will submit your work in PDF form on Gradesope. Your PDF should conform to the following requirements:

- **Do not** include any identifying information (name, CS username, Banner ID, etc.) in your PDF, since all homeworks are graded anonymously
- Each problem (where "problem" is one of the Problems 1–4 or 1–5) should start on a separate page. When you submit on Gradescope, you will be asked to mark which pages correspond to which problem
- At the start of each problem, write a brief *collaboration statement* that lists the names and CS usernames of anyone you collaborated with and what ideas you discussed together
- If you consulted any outside resources while answering any question, you should cite them with your answer

There are two separate Gradescope submissions for this assignment:

- All students should submit Problems 1–4 to the assignment labeled **"Homework 1: Problems 1–4"**
- CS1620/CS2660 students must also submit Problems 5–6 to the assignment labeled **"Homework 1: Problem 5"**. For this part, you can either make a separate PDF with problems 5, or just have one PDF and then mark the pages for these problems.
- Submissions for Problems 5 from CS1660-only students will not be graded (ie, there is no extra credit)

Problem 1: Paper threat models

Consider the following system: In some classes at Blue University, students demonstrate their programming assignments to a TA in person for grading. The TA then writes their evaluation and feedback on a paper rubric, which is then returned to the student.

After their grading meeting, students write up a report about their work (eg. a README), print it out, and then physically turn in both documents during the next lecture. We will call the student's turned-in work a *submission*, where $submission = \{report, rubric\}$. When grading, TAs read the paper submissions and use the evaluation feedback from the rubric as part of the student's grade.[1]

Question a) How is this system insecure? Specifically, describe the following (2-3 sentences each):

i. Describe at least one adversary in the system who might want to misuse it

ii. Describe at least two attacks the adversary might want to perform. When describing your attacks, try to be specific about the actions performed and use terminology we've used in class

Question b) Imagine that it's feasible for the people in the system to do cryptographic functions on the paper documents involved (ie. student encrypts submission S with some key k). How would you add cryptographic techniques (encryption, signing, hashing, etc.) to the current system? Describe **at least two** operations you could add to the system. For each one, consider the following:

i. How does the operation (encryption, signing, etc) prevent one of the attacks you described in part (a)? (It's fine if your two operations target the same attack, or target completely different attacks.)

ii. Make sure you are specific on the inputs and outputs of each operation: for example, it's not sufficient to say "use digital signatures." Instead: what document should be signed, and with what key? (And who knows key(s)?)

Note: This problem is quite open-ended and there are a lot of possible designs! Feel free to make assumptions about the parties involved (eg. "assume you can give each student a `<type of key>`"), as long as you state them in your answer.
You will be graded based on the thoroughness of the design you propose and your explanation of how it helps secure the system against the threats you describe in part (a).

[1] Fun fact: This is how programming assignments worked for many of Nick's undergraduate classes (in computer engineering). When Nick first started TAing, he graded lots of code on paper.

Problem 2: Messaging with Public Keys

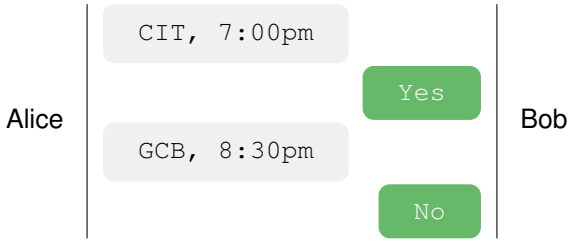
Alice and Bob use a messaging app that encrypts messages using public key cryptography.

In the app, Alice and Bob both have their own public-private key pair, (PK_A, SK_A) and (PK_B, SK_B), respectively. When Alice wants to send a message m, to Bob, Alice's app encrypts m using Bob's public key and sends the resulting ciphertext, $c = \mathsf{Enc}(PK_B, m)$, to Bob. When Bob's app receives the message, it can decrypt it using Bob's secret key, $m = \mathsf{Dec}(SK_B, c)$. Bob can send messages to Alice in a similar way (eg. Bob encrypts using PK_A, Alice decrypts using SK_A).

The exact asymmetric encryption algorithm used (eg. RSA, ECDSA, etc.) is not relevant to this problem, but you can assume that it is *deterministic*. In other words, multiple encryptions of a given plaintext always produce the same ciphertext.

Answer the following questions based on this scenario. Your responses to each question should be no more than 1–2 sentences each.

Question a) Suppose that Alice and Bob exchange short messages about times and places to meet on Blue University's campus. Their messages always have the same format and use consistent names to refer to places, like this:



Suppose that Eve can observe the encrypted messages sent by Alice and Bob, but can't modify or decrypt them (we call this type of attacker a "passive eavesdropper"). If Eve knows the message format and observes a large number of encrypted messages, what could Eve learn about Alice and Bob's meetings?

Question b) The messaging app has a way to search for users and obtain their public keys. Eve manages to find Alice on the app and downloads her public key, PK_A. Does this help Eve reveal any new information about Alice and Bob's messages? **Explain your reasoning** and what (if anything) is revealed.

Question c) Imagine you are a developer for the messaging app. How could you modify the protocol to prevent the attacks described in parts (a) and (b)? Assume that you can't send any extra messages, or change which cryptographic functions are used. (*Hint*: you don't need to change a lot!)

Question d) *(Independent of your answer for part (c))* **True or false**: Since this protocol uses public key cryptography, a user (eg. Alice) can *authenticate* messages they receive from another user (eg. Bob). **Explain your reasoning.**

Problem 3: Exceptional Access

Ever since strong cryptography became accessible to consumers, law enforcement agencies have been advocating for "exceptional access" to secure systems, or mechanisms built-into cryptographic protocols that allow law-enforcement to gain access to encrypted data in certain circumstances (eg. search warrants, public safety, etc.). This has been a matter of public debate, as many cybersecurity experts have argued that enabling exceptional access would reduce overall security and challenge users' fundamental right to privacy.

To provide one high-profile example, in 2015, Apple refused to create a software update that would allow the FBI to unlock an iPhone that was part of a criminal investigation[2].

Two cryptographers from GCHQ (the UK's equivalent of the NSA) have proposed a method of exceptional access that they believe does not compromise the integrity of encryption in the following article: `https://www.lawfareblog.com/principles-more-informed-exceptional-access-debate`. Please read over this article and then consider the following questions.

These are open questions and will be graded for completeness or your responses and justification of your claims. Your answers should be at most 3–4 sentences.

Question a) The article discusses several principles that can be used to evaluate exceptional access methods. What standards would need to be followed for you to find an exceptional access reasonable? If you believe exceptional access is never acceptable, please justify why. How does your answer differ from the GCHQ principles (from the article above), if at all?

Question b) It is argued that if companies don't provide a mechanism for the government to access communications between users, then the only option left for government agencies is hacking, eg. finding or exploiting known vulnerabilities in a system until they get what they want. However, relying on vulnerabilities for this purpose can incentivize governments to avoid disclosing them to the public so they remain useful to law enforcement. Do you find hacking a compelling mechanism for providing exceptional access? Explain your reasoning.

[2] `https://en.wikipedia.org/wiki/Apple%E2%80%93FBI_encryption_dispute`

Problem 4: Lab: Burp Suite

This problem is a short "lab" designed to give you practice using Burp Suite, a set of tools for testing web applications, which you may find useful for the next project.

To complete the lab, follow the instructions here: `https://hackmd.io/@cs1660/burp-suite-lab`

At the end of the lab, you'll be asked to take a screenshot showing you completed the final task, which involves sending a request that causes a specific change in a website. To receive credit for this problem, just include this screenshot in your submission.

Problem 5: (CS1620/CS2660 only) Better Passwords via a Browser Extension

Only CS1620/CS2660 students are required to complete this problem.

PwdHash (`https://crypto.stanford.edu/PwdHash/`) is a browser extension that transparently converts a user's password into domain-specific values. For example, if a user visits `bank.com` and types in the plaintext password `iampassword`, when the user submits the login form, PwdHash instead causes the browser to send hash(`iampassword` ++ `bank.com`) as the password, where ++ is the string concatenation operator and hash is some cryptographic hash function. (PwdHash knows the domain that the user is visiting on because, as a browser extension, it has direct access to the URLs a user visits.)

Consider the following questions. For each part, your answers should be around 50 words each.

Question a) (4 pts) Alice, a user with the PwdHash extension installed, likes using the same password "`balloons`" for every single website. Does PwdHash prevent Alice's password from being broken by a dictionary attack? Explain.

Question b) (4 pts) Alice uses `bank.com` for their online banking operations. Suppose `bank.com`'s database is stolen. Does PwdHash protect Alice's password from being cracked by a brute-force attack? Explain.

Question c) (4 pts) Suppose Alice visits a fake website set up by Eve, a web attacker that controls a website that looks identical to `bank.com` but is actually hosted at `bank.co`. However, Alice doesn't notice the difference and, submits their real password for `bank.com` to the fake website. Does PwdHash protect Alice's password from being stolen and used by Eve? Explain.

Question d) How does PwdHash affect the overall entropy of its users' passwords?