Homework 1: Crypto Party

Due: Friday, February 23 @ 11:59 pm EST

Overview and instructions

This homework has 6 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

Lots of possible answers. Some examples:

- Student may want to alter their grade or rubric (attack on integrity)
- Student may try to alter someone else's grade or rubric, prevent it from being submitted (integrity, availability)
- One student may impersonate another student
- Student may forge TA evaluation outright, or submit a copy of someone else's rubric
- Grader may alter marked rubric at grading time (independent of in-person TA evaluation)
- Grader may alter report at grading time
- **Grading rubric**: 4 pts total for each attack (8 pts total for part a). Points for each attack distributed as follows:
  - 4/4: Full credit: describes who the adversary is, the operations they'd perform, and it's clear how this action connects to a security goal (they don't necessarily need to say the words "confidentiality" or "integrity" if it's obvious)
  - 3/4: Minor detail is incorrect or missing (wrong attack type, etc.)
  - 2/4: Significant missing components
  - 1/4: Incorrectly defined, missing significant components, not clear how this is an attack
  - 0/4: Missing/no credit

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.)

[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.

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)?)

Once again, lots of possible answers:

- To protect against forgery/alteration by student
  - Each TA signs rubric (or hash of rubric) with a private key; public keys known to all staff members
  - Each TA MACs rubric with a shared key, with key known to staff
  - To protect against reuse of an existing rubric, signed version must include some info about the student
  - (Good, but not solely sufficient for integrity protection) TA encrypts rubric and student submits ciphertext
- To protect against alteration by grader:
  - Student signs rubric (or hash of rubric) with public key known to staff

Grading rubric: 6 pts per operation described (so 12 pts total for part b). For each operation:

- Describe what operation does (3 pts)
  - 3/3: Full credit: describes relevant crypto mechanic, ie, what key is used, what document it operates on, and who does it, such that it's clear what is happening–should leave small comment about what's missing
  - 2/3: Minor detail missing or unclear
  - 1/3: Multiple details missing or unclear
  - 0/3: Missing/no credit
- Describe how operation prevents an attack (3 pts) (check all that apply)
  - +3: Full credit (meets criteria for next two items)
  - +1: Describes which attack this is preventing
  - +2: Describes how crypto operation prevents attack (ie, attacker can't forge signature X)
  - +1: Half credit for above item (missing some significant detail)
  - +0 Missing/no credit

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).

Problem 2: Messaging with Public Keys

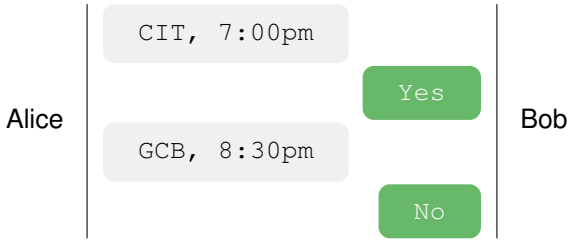
Alice and Bob use a messaging app that encrypts messages using public key cryptography.

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?

Since the encryption scheme is deterministic, the same plaintext always encrypts to the same ciphertext. After observing a large number of messages, Even can take note of the repeated ciphertexts and learn when Alice and Bob revisit a location they have visited previously, or know when a Yes/No answer is given. If Eve is also on the Blue University campus, or has other signals as to where Alice and Bob meet up after the fact, Eve could correlate the ciphertext with real-world observations and then use this info to determine the locations of future meetings.

- 3 pts total
- 3/3: Full credit: Demonstrates that Eve can recognize repeated messages or correlate them in some way, even though Eve can't decrypt.
- 2/3: Missing some important detail, or adds some incorrect detail
- 1/3: Answer does not reason about deterministic encryption or does not match problem
- 0/3: Missing/no credit
- +0 (Add to rubric and mark, but no points for now): Mentions that Eve could use out-of-band info (in-person or online stalking) to figure out where Alice and Bob are actually meeting

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.

Eve can encrypt arbitrary messages to Alice using PK_A and observe the ciphertext. This would allow Eve to learn the ciphertext of any chosen plaintext. Assuming Eve knows the message format and the names for the buildings, Eve could figure out the ciphertexts for the "yes" or "no" messages. Eve **cannot** figure out the times and places, however, unless Eve also learns Bob's public key.

- 4 pts total
- 4/4: Full credit: recognizes that (1) Even can determine arbitrary ciphertexts, and (2) Eve can use this to figure out which ciphertexts have a "yes" or "no" response from Bob because Eve knows the format.
- 3/4: Missing small detail, or mentions that Eve could also figure out the locations with Alice's key
- 2/4: Missing (1) or (2)
- 1/4: Answer does not relate to problem
- 0/4: Missing/no credit

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!)

Could add a random *nonce*, a random number used only once, to the beginning or end of the message before encrypting. Assuming the number is chosen at random and not repeated, this ensures that every message encrypts to a different ciphertext.

Alternate solution (used in previous years' solutions): a high-precision timestamp could also work, but may also be possible to predict.

- 4 pts total
- 4/4: Full credit: reasonable solution involving a nonce, *precise* timestamp, or something else that behaves similarly
- 3/4: Answer is close but missing small detail (eg. precision for timestamp not mentioned)
- 2/4: Answer is in the right direction but missing major detail
- 2/4: Answer is excessively complicated or adds extra crypto
- 1/4: Answer does not relate to problem
- 0/4: Missing/no credit

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.**

False. Authentication means that we have a way to verify that a message we receive actually originated from the sender. This protocol only provides confidentiality: any user can encrypt a message using Alice's public key. The app might communicate who sent the message, but cryptographically there's no way to verify the sender.

- 4 pts total
- 4/4: Full credit: Answer (1) demonstrates a correct understanding of what authentication means, (2) explains how the current protocol does not provide authentication. (2 pts each, mark partial credit on scale as appropriate)

- 3/4: Answer is in the right direction but missing a small detail
- 2/4: Missing (1) or (2)
- 1/4: Missing significant details
- 0/4: Missing/no credit, or answers true or false without justification

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.

Sample TA Solution

Graded on completion, 5 pts per part. Answers may vary.

(a) There are two approaches that can be taken by students: arguing that exceptional access is never justified, and listing principles that they believe would make an exceptional access method acceptable.

Typically, arguing exceptional access is never justified would be done by claiming that any method that allows unintended recipients to access encrypted data necessarily constitutes a cybersecurity vulnerability, and that vulnerabilities have worse consequences (i.e., potential for significant cyberattacks, espionage, loss of trust in software services, potential compromise of journalists' ability to communicate with sources) than an inability to access encrypted data (failure to prevent and prosecute crimes).

See `https://dspace.mit.edu/bitstream/handle/1721.1/97690/MIT-CSAIL-TR-2015-026.pdf` for a justification of why exceptional access mechanisms necessarily create vulnerabilities, `https://www.washingtonpost.com/world/national-security/former-national-security-officials-urg 2015/12/15/3164eae6-a27d-11e5-9c4e-be37f66848bb_story.html` for background on national security vs. law enforcement disagreements on exceptional access, and `https://www.thirdway.org/report/weakened-encryption-the-threat-to-americas-national-security` for a pure national security argument against exceptional access.

For the second approach, below are three examples of standards that might be suggested, although there are many more that students could reasonably suggest. For additional examples, see pages 13-14 of `https://carnegieendowment.org/files/EWG__Encryption_Policy.pdf` (intended for on-device key escrow but nonetheless broadly applicable).

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

1. Software providers cannot be required to design their systems in preparation for exceptional access.

   While one-off exceptional access assistance can create temporary vulnerabilities, designing systems in a way that compromises confidentiality or integrity upfront creates persistent vulnerabilities. Additionally, this is antithetical to security by design. Although this may lead to firms redesigning their systems to make exceptional access methods impossible (i.e., by allowing the client to verify whether any keys have been changed or added, which the ghost proposal requires), these decisions would necessarily improve the overall security of these systems. For context, system redesigns to enable persistent methods of exceptional access can be required by Australia's exceptional access legislation (*the Telecommunications and Other Legislation Amendment (Assistance and Access) Act*) through Technical Capability Notices. This standard is in conflict with the view expressed by former FBI Director James Comey, who argued that "it makes more sense to address any security risks by developing intercept solutions during the design phase, rather than resorting to a patchwork solution when law enforcement comes knocking after the fact" `https://www.fbi.gov/news/speeches/going-dark-are-technology-privacy-and-public-safety-on-a-collision-course`.

2. A judge, in determining whether to allow a particular form of exceptional access, must be presented with a report on the security consequences of the proposed method by an independent cybersecurity expert. The conclusions of the report must be made public.

   Although it is possible for firms to protest exceptional access orders, it is likely that they will consistently claim that methods are too risky while governments are likely to consistently claim that the risk is minimal. This may to lead to situations in which judges are forced to weigh well-defined national security consequences against contested cybersecurity risks. When the judge cannot evaluate the cybersecurity risks, they could inadvertently allow exceptional access orders where there is significant risk, or vice versa. An independent expert would provide the judge with unbiased advice on the consequences of the order, allowing them to evaluate its proportionality more effectively. Additionally, publicising the expert's finders would help to either assuage public concerns about the security of the products they use (if the orders are reasonable) or put pressure on government agencies to proactively suggest methods that maintain the cybersecurity of these systems (if the orders are reckless).

3. Requests for exceptional access must allow sufficient time for secure methods of retrieving the information to be developed.

   This would prevent access methods from being rushed, limiting the likelihood of serious vulnerabilities being introduced. Although the rhetorical examples typically used when justifying exceptional access are time-sensitive (i.e., a bomb will go off and the location is on an encrypted hard drive), situations like this are unlikely (even the case examples provided by Comey in the article cited earlier are all after the fact evidence gathering) and the risk caused by rushing exceptional access is tremendous. Given that a large part of the concern about exceptional access is that vulnerabilities may be created by developer error, shortening the timespan for implementation greatly increases the risk that serious vulnerabilities are introduced. Additionally, rushed implementation may compromise the validity of review processes both internally and by independent evaluators (such as in standard 1).

The key area of contention between these standards and the GCHQ principles is that the GCHQ principles leave the door open to changing the design of systems to make exceptional access easier. In fact, their ghost proposal is only possible if companies can be forced to actively change the design of existing messaging apps (including Signal and WhatsApp).

(b) I do not think that this is a valid argument. As seen with the Cisco-Webex vulnerability discussed in the second reading, there are independent security researchers working to find vulnerabilities with motivation of finding these first to receive a bounty. Although the government could find hacks into software and devices allowing them access to important user-data, it would only be a matter of time until another security researcher finds the same vulnerability used by the government and discloses this to the company who will then fix the bug. Therefore, it is not viable for the government to assume

that hacking will always work. (4 sentences)

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.

Graded on completion, +5 for including a screenshot that looks reasonable.

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?

Sample TA Solution

a) (4 pts) In a dictionary attack, an attacker tries password possibilities that are likely to succeed (such as common words or short sequences of letters). PwdHash combines an easy password with a domain-specific string and then hashes it; this generates a pseudorandom string that is difficult to guess. If the attacker doesn't know about PwdHash, then this would be hard to guess. However, due to the open design principle, this is not a valid assumption to make. If the attacker knows that the user uses PwdHash, then a dictionary attack would be possible (since the hash function is public, the attacker knows the domain, and "`balloons`" is easy).

- 4/4 pts: Recognizes that this makes dictionary attacks harder, unless the attacker knows Alice is using PwdHash.
- 3/4 points: Correct intuition, but something important missing
- 2/4 points: Correct intuition, but multiple important things missing
- 1/4 points: Incorrect intuition, major components misssing
- 0/4 points: No credit/missing

b) (4 pts) If the attacker does not know that Alice is using PwdHash, then the attacker will have to brute force long, random strings (assuming the output from PwdHash is long) which will be difficult. However, if the attacker does knows that Alice is using PwdHash, all they have to do is compute the brute force attack by running passwords through PwdHash before passing them to the database hash function, which will reduce the length of passwords needed to brute-force a small password like "`balloons`".

- Same grading scale as above. For full credit, must mention that attack is hard if attacker doesn't know Alice is using PwdHash, but easy if they do.

c) (4 pts) PwdHash is domain-specific, so it will generate a different hash of the password on the bank.co website; thus, `bank.co` will not get Alice's `bank.com` password.

- Same grading scale as above. For full credit, must mention that the hash for `bank.co` is different because a different domain is used.

d) (4 pts)

Users' passwords would have the same or less entropy if they are using PwdHash. This is because a hash function is deterministic: each password is only going to have one output, which has the same entropy. If user's password happened to have a higher entropy than the output space of the hash function, the PwdHash could even reduce entropy.

- Same grading scale as above. For full credit, must mention that the entropy of the password does `not` increase because the input the space is still the same.