Weaver
Fall 2017

CS 161
Computer Security

Homework 2

Due: Friday, September 22, at 11:59pm

Instructions. This homework is due **Friday, September 22, at 11:59pm**. No late homeworks will be accepted. You *must* submit this homework electronically via Gradescope (not by any other method). When submitting to Gradescope, *for each question* your answer should either be a separate file per question, or a single file with each question's answer on a separate page. This assignment must be done on your own.

Problem 1 ***True-or-False Questions*** **(5 points)**

Answer each question. You don't need to justify or explain your answer.

(a) True or False Diffie–Hellman protects against eavesdroppers but is vulnerable to man-in-the-middle attacks.

(b) True or False: Suppose there is a transmission error in a block B of ciphertext using CBC mode. This error propagates to every block in decryption, which means that the block B and every block after B cannot be decrypted correctly.

(c) True or False: The IV for CBC mode must be kept secret.

(d) True or False: Alice and Bob share a symmetric key k. Alice sends Bob a message encrypted with k stating, "I owe you \$100", using AES-CBC encryption. Assuming AES is secure, we can be confident that an active attacker cannot tamper with this message; its integrity is protected.

(e) True or False: Suppose there is an error in a block B of plaintext when encrypting using CFB mode. This error propagates to every block in encryption process, which means that after the ciphertext is received and decryption, the decrypted message of block B and every decrypted message after B will be different from the original message (that doesn't contain the error in B).

Problem 2 ***Not All Chains Are Secure*** **(15 points)**

Recall from lecture that CBC and CFB modes avoid the catastrophic failure CTR mode could have if the IV ever gets re-used (still bad, but not as bad). However, are CBC mode and CFB mode equally "secure" then? Consider the following case when the IV is *NEVER* re-used:

Bob sent Alice a message asking if she wants to study for the midterm on Monday and Alice sent back a reply to Bob. Mallory observes Alice's reply in ciphertext C and IV, and knows that M is a simple and short reply (e.g. "Yes" or "No". The reply is one of the known phrases to Mallory and all possible replies are smaller than the input size to the cipher block used).

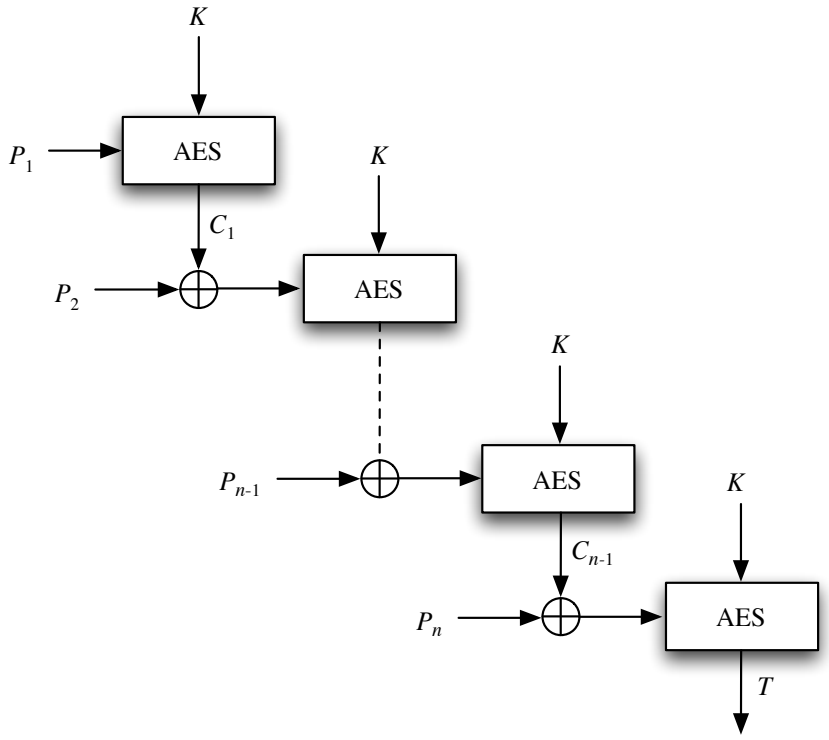
Bob is Mallory's best friend and Bob is willing to encrypt any plaintext that Mallory gives him, using the SAME key that was used in his communication with Alice but NOT the same IV.

How could Mallory figure out the *plaintext* of Alice's reply? (Hint: Bob picks IV from a book of random numbers, so Mallory knows exactly what IV Bob is going to use when he requests Bob to encrypt his chosen plaintext)

(a) Could Malory figure it out if Alice and Bob used CBC mode?

(b) What about CFB mode?

Problem 3 ***MAC Attack*** **(20 points)**



Consider the MAC algorithm shown in the diagram above. Each P_i is the ith block of a given message. At each stage, we encrypt the XOR of the previous stage and the next message block using the key K.

This algorithm is quite similar to AES-EMAC (shown in lecture and in the lecture notes), but differs in final stage, by using the same key as in the earlier stages, not including a second invocation of AES.

Suppose Mallory observes two single-block messages, M_1 and M_2, and the corresponding tags for these, T_1 and T_2. Show that Mallory can construct a message M_3 for which Mallory knows the associated tag T_3, even though Mallory does not know K.

Problem 4 ***Finding Common Patients*** **(20 points)**

Caltopia has two hospitals: Bear Hospital and Tree Hospital, each of which has a database of patient medical records. These records contain highly sensitive patient information that should be kept confidential. For both hospitals, each medical record is a tuple (p_i, m_i), where p_i and m_i are strings that correspond to the patient's full name and medical record respectively; assume that every person in Caltopia has a unique full name. Thus, we can think of Bear Hospital's patient database as a list of tuples $(x_1, m_1), (x_2, m_2), ..., (x_n, m_n)$, where m_i is the medical information that Bear Hospital has for patient x_i. Similarly, we can think of Tree Hospital's database as a list $(y_1, m'_1), (y_2, m'_1), ..., (y_m, m'_m)$, where m'_i is a string that encodes the medical information that Tree Hospital has for the patient named y_i. Note that for a given patient, Tree Hospital and Bear Hospital might have different medical information.

The two hospitals want to collaborate on a way to identify which Caltopia citizens are patients at both hospitals. However, due to privacy laws, the two hospitals cannot share any plaintext information about patients (including their names) unless both hospitals know *a priori* that a patient has used both hospitals.

Thus, the two hospitals decide to build a system that will allow them to identify common patients of both hospitals. They enlist the help of Lady Olenna, who provides them with a trusted, third-party server S, which they will use to discover the names of patients who use both hospitals. Specifically, Bear Hospital will take some information from its patient database and transform it into a list $(x_1^*), (x_2^*), ..., (x_n^*)$ (where (x_i^*) is somehow derived from x_i (the patient's full name) and upload it to S. Similarly, Tree Hospital will take information from its patient database, transform it into a list $(y_1^*), (y_2^*), ..., (y_m^*)$, and upload this transformed list to S. Finally, S will compute a set of tuples $P = (i, j) : x_i = y_j$ of all pairs (i, j) such that $x_i^* = y_j^*$ and send P to both Bear Hospital and Tree Hospital. The two hospitals can then take their respective indices from the tuples in P to identify patients who use both hospitals.

We want to ensure three requirements with the above scheme: (1) if $x_i = y_j$, then $(i, j) \in P$, (2) if $x_i \neq y_j$, then it is very unlikely that $(i, j) \in P$, (3) even if Eve (an attacker) compromises S, she cannot learn the name of any patient at either hospital or the medical information for any patient. For this question, assume that Eve is a passive attacker who cannot conduct Chosen Plaintext Attacks; however, she does know the names of everyone in Caltopia, and there are citizens whose full names are a unique length.

Fill in your solutions below. Your solution can use the cryptographic hash SHA-256 and/or AES with one of the three block cipher encryption modes discussed in class; keep in mind that Eve can also compute SHA-256 hashes and use AES with any block cipher mode. You can assume that Bear Hospital and Tree Hospital share a key k that is not known to anyone else. You *cannot* use public-key cryptography or modular arithmetic.

(a) In the collaboration scheme described above, how should Bear Hospital compute x_i^* (as a function of x_i)? How should Tree Hospital compute y_i^* (as a function of y_i)? Specifically, your solution should define a function F that Bear Hospital will use to

transform x_i into x_i^*, and if relevant, a function G that Tree Hospital will use to transform y_i into y_i^*.

(b) Explain why requirement (1) is met by your solution, i.e., explain why it is guaranteed that if $x_i = y_j$, then $x_i^* = y_j^*$ will hold. Explain your answer in one or two sentences.

(c) Explain why requirement (2) is met by your solution, i.e., if $x_i \neq y_j$, explain why it is unlikely that $x_i^* = y_j^*$. Explain your answer in one or two sentences.

(d) Explain why requirement (3) is met by your solution, i.e., if S is compromised by Eve, then the information known to S does not let Eve learn any patient information (neither the names of patients at a particular hospital nor the medical history for any patient). Explain your answer in one or two sentences.

Problem 5 ***Photo Retrieval System*** **(20 points)**

CalPix, a new social photo-sharing site, lets users upload photos to its servers and share them with their friends. For privacy, the service has implemented a form of access control: upon uploading a photo, it lets a user decide which other users can view it.

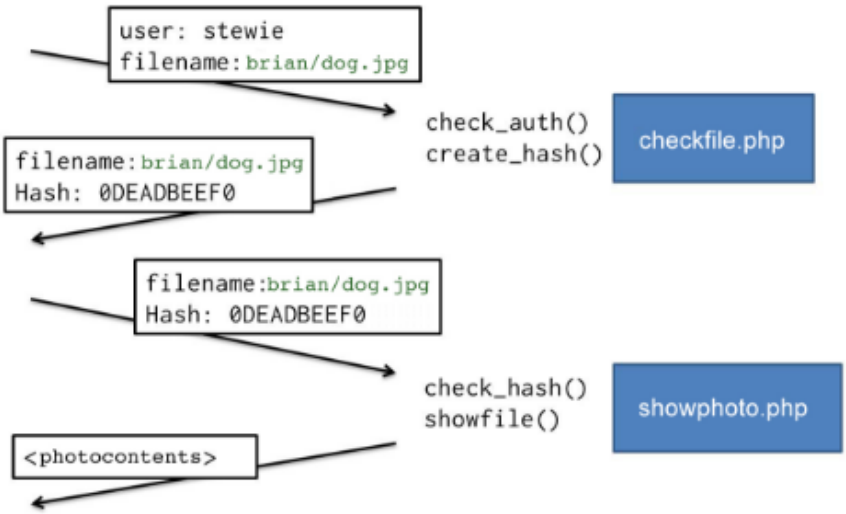
Photos are retrieved via calls to the `http://calpix.com/showphoto.php` page, which checks whether the given user should be able to view the photo, and if so, sends the photo file's contents.

The script `showphoto.php` accepts two URL parameters: `filename`, which is the filename of the photo to show, and `hash`, which is an authenticator. Photos are grouped into subdirectories by username, so the `filename` parameter is actually the path to the file, relative to the base photo directory (for example, `someuser/somephoto.jpg`).

Before showing any photo, `showphoto.php` checks the `hash` value sent. This hash value is the first five bytes (10 hex digits) of a SHA-256 digest of the concatenation of the filename and a secret encryption key (known only to the server).[1]

The page `http://calpix.com/checkfile.php` looks up the current logged-in user's identity and checks whether the user is allowed to access a particular photo file. If so, `checkfile.php` generates the correct hash value and redirects to `showphoto.php`. `checkfile.php` is able to generate the correct hash value since it runs on the server side and thus has access to the secret encryption key.

The diagram below sketches this protocol. User `stewie` has access rights for the file `brian/dog.jpg`.



[1] You may find this scheme peculiar, but in fact this example was inspired by a real life example, with the only significant change being that we're specifying the hash function as SHA-256, whereas in reality the function was MD5. We didn't want students distracted by possible weaknesses in MD5, which don't play a role here.

(a) Among the following primitives developed in class: (signatures, PRNGs, encryption, MAC, hashing, certificates), which primitive's usage and functionality does the above hashing scheme aim to achieve? Which of the three CIA(Confidentiality, Integrity and Authentication) goals discussed in lecture does the above hashing scheme aim to provide? State your answers to these two questions and explain your choices in one or two sentences.

(b) Why do we need a secret key as an input to the hash function?

Problem 6 ***Why do RSA signatures need a hash?*** **(20 points)**

This question explores the design of standard RSA signatures in more depth. To generate RSA signatures, Alice first creates a standard RSA key pair: (n, e) is the RSA public key and d is the RSA private key, where n is the RSA modulus. For standard RSA signatures, we typically set e to a small prime value such as 3; for this problem, let $e = 3$.

To generate a standard RSA signature S on a message M, Alice computes $S = H(M)^d \bmod n$. If Bob wants to verify whether S is a valid signature on message M, he simply checks whether $S^3 = H(M) \bmod n$ holds. Analogous to RSA encryption, d is a private key known only to Alice and $(n, 3)$ is a publicly known verification key that anyone can use to check if a message was signed using Alice's private signing key.

For this question we'll now explore why RSA signatures use a hash function to compute the signatures. Suppose RSA signatures skipped using a hash function and just used M directly, so the signature S on a message M is $S = M^d \bmod n$. In other words, if Alice wants to send a signed message to Bob, she will send (M, S) to Bob, where $S = M^d \bmod n$ is computed using her private signing key d.

(a) With this simplified RSA scheme, how can Bob verify whether S is a valid signature on message M? In other words, what equation should he check, to confirm whether M, S was validly signed by Alice? You don't need to justify your answer; just show the equation.

(b) Mallory learns that Alice and Bob are using the simplified (hash-less) signature scheme described above and decides to trick Bob. Mallory wants to send some (M, S) to Bob that Bob will think is from Alice, even though Mallory doesn't know the private key. Explain how Mallory can find M, S such that S will be a valid signature on M.

You should assume that Mallory knows Alice's public key n, but not Alice's private key d. She can choose both M and S freely. The message M does not have to be chosen in advance and can be gibberish.

Hint: If Mallory chooses M and then tries to find a corresponding S, she'll be at a dead-end, because finding S requires inverting a one-way function (cubing modulo n), and we know that is hard without knowledge of the trapdoor (the private key d). So instead, she should ...

(c) Sameer is holding an auction. Alice and Bob will submit signed bids to the auctioneer Sameer, signed using this simplified RSA signature scheme. The message M is an integer that is their bid (in dollars), and they will send just their signature on M, signed using this simplified RSA scheme. Sameer will accept whichever bid is highest and expect that person to pay up however much they bid.

Mallory wants to mess with Bob (her rival). So, when Bob forms his bid M and sends Sameer the signed bid $S = M^d \bmod n$, Mallory intercepts the message from Bob containing S. Mallory would like to tamper with S to form a new signature S' that corresponds to a bid for $64\times$ as much as Bob's original bid, to force Bob to

win the auction and pay through the nose for it. More precisely, she'd like to find a value S' such that S' is a valid signature on $64M$, so she can replace S with S' and forward the result onto Sameer. Help Mallory out: show how she can compute such an S'.

(Assume that M is small enough that $64M < n$, so $64M$ does not wrap around modulo n.)

(d) Are your attacks in parts (b) and (c) possible against the real RSA signature scheme (the one that includes the cryptographic hash function)? Why or why not?

Problem 7 ***Feedback*** **(0 points)**

Optionally, feel free to include feedback. What's the single thing we could do to make the class better? Or, what did you find most difficult or confusing from lectures or the rest of class, and what would you like to see explained better?