CS255: Intro. to Cryptography **Winter 2023**

Project 2: Short Answers

Shoaib Mohammed (shoaibmh@stanford.edu)
Lovish Chopra (lovish@stanford.edu)

Question 1. In our implementation, Alice and Bob increment their Diffie-Hellman ratchets every time they exchange messages. Could the protocol be modified to have them increment the DH ratchets once every ten messages without compromising confidentiality against an eavesdropper (i.e., semantic security)?

Answer: Yes, if we consider that the DH-ratchet is updated once every ten messages, the protocol could still be decently secure without compromising confidentiality against an eavesdropper.

In the modified protocol, Alice and Bob could keep track of the number of messages that have been exchanged, and when it turns to ten, the count is reset and a new DH-ratchet key pair is generated to derive the updated chain keys.

Let's say that an eavesdropper is able to compromise the system and obtain one of the chain keys for Alice. In that case, the eavesdropper could only get access to at most ten message keys, and hence, can decrypt only at most ten future messages. However, after that, the DH-ratchet step will be executed, the old chain keys will be discarded and the new ones will be used for further message key generation. Hence, the eavesdropper would not be able to decrypt future messages after that.

Essentially, because of the computational DH assumption, we can say that even if the DH-ratchets are incremented every ten messages, the system would still be CPA secure, and hence, won't break semantic security.

Question 2. What if they never update their DH keys at all? Please explain the security consequences of this change with regards to Forward Secrecy and Break-in Recovery.

Answer: If Alice and Bob never update their DH keys at all, our system can be subject to threat against break-in recovery. However, there is no effect on forward secrecy.

Effect on Break-in Recovery: Let's say that an adversary breaks into Alice's system and gets access to Alice's current receiving chain key from Bob. In that case, the adversary can repeatedly apply KDF to the receiving chain key to get the future receiving chain keys and message keys, and hence, can use those message keys to decrypt all messages sent by Bob to Alice. A similar argument holds for sending chain key as well. Hence, once the adversary gets access to some chain key of the user, the adversary can decrypt all future messages, violating break-in recovery.

Effect on Forward Secrecy: There is no effect on forward secrecy and the system remains secure w.r.t. forward secrecy even if the adversary gains access to the current chain key. This is because the adversary cannot reverse the output of the KDF to get the previous message keys and chain keys.

Question 3. Consider the following conversation between Alice and Bob, protected via the Double Ratchet Algorithm according to the spec:
A: Hey Bob, can you send me the locker combo?
A: I need to get my laptop
B: Sure, it's 1234!
A: Great, thanks ! I used it and deleted the previous message.
B: Did it work?
What is the length of the longest sending chain used by Alice? By Bob? Please explain.

Answer: The sending chain is used to generate the message keys, and the first chain key is generated using Diffie-Hellman output. For all the consecutive messages sent by one user to another, the same sending chain is used to derive the message keys.
Since Alice sends at most 2 consecutive messages, Alice's longest sending chain is of length 2. Since Bob sends at most 1 consecutive message, Bob's sending chain is of length 1.

Question 4. Unfortunately, in the situation above, Mallory has been monitoring their communications and finally managed to compromise Alice's phone and steal all her keys just before she sent her third message. Mallory will be unable to determine the locker combination. State and describe the relevant security property and justify why double ratchet provides this property.

Answer: Just before Alice sends her third message, Mallory would have the new DH-Ratchet derived sending chain key of the third message sent by Alice. However, by the property of the DH Ratchet algorithm that we discard previous message keys, Mallory would not really have the previous sending chain keys of Alice, or the receiving chain key corresponding to the first message received from Bob. Hence, Mallory won't be able to decrypt the locker code.

This property is called **Forward Secrecy**, which means that even if Mallory is able to obtain the current status of keys, Mallory won't be able to decrypt the previous messages in the conversation. The double ratchet algorithm provides this property by virtue of the fact that every time message exchange happens between two parties, a new chain key is created using a fresh DH-ratchet step, and the old keys are discarded. In the DH-ratchet step, a new DH-key pair is generated and new sending/receiving chain keys are derived out of them. In a conversation between Alice and Bob, it would create a 'ping-pong' behaviour as they exchange messages between each other. As a result, Mallory does not have previous keys, and hence cannot decrypt the previous messages.

Question 5. The method of government surveillance is deeply flawed. Why might it not be as effective as intended? What are the major risks involved with this method?

Answer: The goal of government surveillance is to be able to decrypt any communication that goes through. However, this method is deeply flawed because of multiple reasons:

(a) One of the major functionalities of the double ratchet algorithm is that it continues to update chain keys and message keys as messages are exchanged in order to make sure that if an adversary gets access to one of the keys, the adversary cannot decrypt the whole conversation. However, the government public key is constant. If an adversary ends up eavesdropping or compromising the government system to obtain the secret key of the government, the whole system essentially breaks, since now the adversary could decrypt every possible message that goes across in the country, losing any sign of recovery.

(b) It is dependent on the people to make sure they follow the guidelines set by the government. For instance, it is required that the sending message key is encrypted using the government's public key and attach this information in the header. However, people could use an incorrect public key due to which the government will not be able to decrypt messages. Furthermore, there are other values that need to be present in the header such as `ivGov` which is the random IV the sending message key gets encrypted with, `cGov` which is the encryption of the sending message key using AES, and `vGov` which is the public key of the government key pair object generated using ElGamal encryption scheme. If any of these fields are not present in the header or have false values, then the government will not be able to decrypt messages.

(c) One caveat that we can also see is what users can do to prevent the government from tracking them. If the government sees certain messages not decryptable, the government may want to be able to find such users since they are not following the guidelines set by the government. For this, users could hide their information by using false usernames, using VPN, or using a proxy server. This would make it difficult for the government to track people and at the same time they would be unable to decrypt communication.

Question 6. The SubtleCrypto library is able to generate signatures in various ways, including both ECDSA and RSA keys. For both the ECDSA and RSA-based signature techniques, please compare:
(a) Which keys take longer to generate (timing SubtleCrypto.generateKey)
(b) Which signature takes longer to generate (timing SubtleCrypto.sign)
(c) Which signature is longer in length (length of output of SubtleCrypto.sign)
(d) Which signature takes longer to verify (timing SubtleCrypto.verify)
Note: For this question, you can choose to conjecture what the right answer might be or run the script provided in the starter code (in folder /question6code). A well justified conjecture that may not be fully correct can still receive full credit. To run the code in /question6code, navigate to the folder in your terminal and then execute the command node q6code.js.

Answer: Based upon running the script provided in the starter code and observing the results, the following is the comparison between ECDSA and RSA-based signature techniques:

(a) RSA key takes longer to generate than ECDSA key.

(b) RSA signature takes longer to generate than ECDSA signature.

(c) RSA signature is longer in length than ECDSA signature.

(d) ECDSA signature takes longer to verify than RSA signature.