Name: Yun Wen Chen
EID: dc27863
CS Login: dchen
Email: dianachen@utexas.edu

Lecture 53

1. Why is it important for a digital signature to be non-reusable?

Because you don't want a someone to impersonate someone else with a copy of their digital signature.

2. Why is it the hash of the message typically signed, rather than the message itself?

Public key encryption is an expensive process, and messages could be long, whereas the hash of the message is a shorted, fixed-size value.

3. What assurance does R gain from the interchange on slide 4?

R is assured that the message is unforgeable, authentic, non-repudiable, tamperproof and non-reusable.

Lecture 54

1. What is the importance of certificate authorities?

Certificate authorities are important because they vouch for for a binding of a user's identity to their public key. A certificate binds a user's public key to their identity, but a certificate authority vouches for the authenticity of that binding.

2. In the example on slide 5, why does X sign the hash of the first message with its private key?

Because X is certifying that K_y is the legitimate public key of identity Y based off of the information that Y sent X. When Y sends X $\{Y, K_y\}$, X signs that state of Y's identity and public key. If Y later becomes compromised and tries to send messages with a different public key, the hash values of the compromised identity and public key will not match X's signed hash value of Y's identity and public key, indicating that this is not an authentic certificate.

3. Why is necessary to have a hash of Y and K_y?

You want to be able to have a hash value of Y and K_y because you want to make sure that anyone trying to impersonate Y doesn't discover the legitimate public key. Otherwise, they would be able to send certificates to anyone and be accepted.

4. What would happen if Z had a public key for X, but it was not trustworthy?

If X's public key is not trustworthy, then when Z strips the signature of X, it might decrypt to an incorrect hash value of $h(\{Y, K_y\})$. Therefore when Z computes the hash value of Y and K_y, it would not be the same as the hash value signed by X.

Lecture 55

1. What happens at the root of a chain of trust?

The root of a chain of trust contains a entity that is believed to be trusted to check credentials by everyone.

2. Why does an X.509 certificate include a "validity interval"?

The validity interval defines how long a certificate is valid for. For instance, if there is a policy where the public key of a user changes once a month, a certificate should expire at the end of the month because a subject Y would get new public key K_y such that it no longer matched the hash value of its old public key.

3. What would it mean if the hash and the received value did not match?

It means that the public key associated with the identity is not correct. That, or for the X.509, the public key associated with the identity may be expired, or not yet valid.

Lecture 56

1. What are some protocols previously discussed?

read and write properties of the BLP, read and write properties of the Chinese Wall Policy, RABC, Clark-Wilson, and other integrity models.

2. What may happen if one step of a protocol is ignored?

A user may get locked out of a message. That is, they are unable to remove their own encoding.

In addition, if a sender follows a protocol and sends the encrypted message to the receiver, who does not follow the protocol, the sender may accidentally decrypt the information and send it over the communication medium as plaintext.

3. Why must the ciphers commute in order to accomplish the task on slide 4?

Suppose S sends M to R, encrypted with a key k_1. R does not have k_1 so it cannot decrypt the message. However, R sends M encrypted by k_1 along with another layer of encryption by key k_2. Ideally, if S removed its layer of encryption, then it can deliver the message that can be decrypted by R. However, in turn, S does not know R's key, and therefore cannot "extract" its own encryption from the current message. In order for S to remove only its encoding, it must bypass R's key. If the ciphers commuted, then S would know how to properly bypass k_2 and remove it's encryption k_1.

4. Describe how an attacker can extract M from the protocol in slide 6.

Attacker observes A -> B $M \text{ XOR } K_a$
Attacker observes B -> A $(M \text{ XOR } K_a) \text{ XOR } K_b$

$(M \text{ XOR } K_a) \text{ XOR } ((M \text{ XOR } K_a) \text{ XOR } K_b) = K_b$ // K_b discovered

Attacker observes A -> $((M \text{ XOR } K_a) \text{ XOR } K_b)\ K_a == M \text{ XOR } K_b$

$(M \text{ XOR } K_b) \text{ XOR } K_b = M$ // M discovered

5. Describe how an attacker can extract K_a from the protocol in slide 6.

Attacker observes A -> B $M \text{ XOR } K_a$
Attacker observes B -> A $(M \text{ XOR } K_a) \text{ XOR } K_b$

$(M \text{ XOR } K_a) \text{ XOR } ((M \text{ XOR } K_a) \text{ XOR } K_b) = K_b$ // K_b discovered

Attacker observes A -> $((M \text{ XOR } K_a) \text{ XOR } K_b)$ K_a == $M \text{ XOR } K_b$

$(M \text{ XOR } K_b) \text{ XOR } K_b = M$ // M discovered
$(M \text{ XOR } K_a) \text{ XOR } M = K_a$ // K_a discovered

6. Describe how an attacker can extract K_b from the protocol in slide 6.

Attacker observes A -> B $M \text{ XOR } K_a$
Attacker observes B -> A $(M \text{ XOR } K_a) \text{ XOR } K_b$

$(M \text{ XOR } K_a) \text{ XOR } ((M \text{ XOR } K_a) \text{ XOR } K_b) = K_b$ // K_b discovered

Attacker observes A -> $((M \text{ XOR } K_a) \text{ XOR } K_b)$ K_a == $M \text{ XOR } K_b$

$(M \text{ XOR } K_b) \text{ XOR } K_b = M$ // M discovered

7. Why are cryptographic protocols difficult to design and easy to get wrong?

Cryptographic protocols are difficult to design because there are many things to take into consideration simultaneously. One can design a algorithm that keeps a message decrypted throughout communications, but it could fail to consider what an observer is able to deduce as encrypted messages are passed back and forth. In addition, one can design an algorithm that is impossible to break during communications, but it might also be impossible to break for the receiver or sender. The algorithm may be flawless, but there can be subtleties in a system that an attacker may exploit.

Lecture 57

1. Explain the importance of protocols in the context of the internet.

The purpose of the internet is to open communications between parties. In order to communicate successfully and accurately, structured mechanisms have to be in place so that parties get the right message at the right time and in the right format.

2. Explain the importance of cryptographic protocols in the context of the internet.

The importance of cryptographic protocols in the context of the internet is that they are the mechanisms in the communication protocol that accomplish a information-security related goal. Though secure communication may occur on the internet, there are many hostile and insecure environments which secure information has to travel through.

3. What are the assumptions of the protocol in slide 6?

There is a public key infrastructure in place.

Each part has the others' reliable public key.

4. What are the goals of the protocol in slide 6?

For A to send the shared key K to B.
For A to let B know shared key K was from A.

For B to let A know that it received the shared key K.

5. Are the goals of the protocols in slide 6 satisfied? Explain?

Yes. A sends the proposed shared key to B with B's public key. B knows the key is sent from A because it it is signed by A's private key, which ideally, only A should know. In turn, B returns the key to A with A's public key, signed with B's private key and A knows the B received the message, because ideally, only B should have its private key.

6. How is the protocol in side 6 flawed?

A -> B : $\{\{K\}_{Ka^{-1}}\}\ K_b$
B -> A : $\{\{K\}_{Kb^{-1}}\}\ K_a$

E observes A -> B

E -> B $\{\{\ A \text{ -> } B\ \}_{Ke\text{-}1}\ \}\ K_b$
B -> E $\{\{\ A \text{ -> } B\ \}_{Kb\text{-}1}\ \}\ K_e = \{\{\{K\}_{Ka^{-1}}\}\ K_b\ \}_{Kb\text{-}1}\}\ K_e = \{\{K\}_{Ka^{-1}}\ \}\ K_e$

E strips K_e with its private key, then strips $K_{a\text{-}1}$ with A's public key to reveal K.

Lecture 58

1. Why is it important to know if a protocol includes unnecessary steps or messages?

Because ideally you would want to reduce the time that messages are in transmission. You may have a robust protocol, but it only takes one unconsidered factor to infiltrate a system, and unnecessary steps prolong the expose of messages to these kinds of attacks.

2. Why is it important to know if a protocol encrypts items that could be sent in the clear?

Often times encryption could be expensive, and if items could be sent in the clear, then it would save computation and communication time to not encrypt them.

Lecture 59

1. Why might it be difficult to answer what constitutes an attack on a cryptographic protocol?

Because between all the steps and iterations of the protocol, there are many vulnerable sources and mechanisms for attack, potentially targeted at attacking the different goals of the protocol.

2. Describe potential dangers of a replay attack.

A replay attack my disrupt the flow of the protocol. The protocol may receive a replay message and thinks that it is on that step of the protocol.

3. Are there attacks where an attacker gains no secret information? Explain.

Yes. Attacks like replay and interleaving attacks may attempt to disrupt a protocol run. Though the attacker may not gain any secret information, their attack may keep the authentic senders and receivers from obtaining the secret information. In other words, the goal of a protocol attack may not be to retrieve secret messages, but rather, prohibit communication.

4. What restrictions are imposed on the attacker?

The attacker is not able to generate arbitrary messages, because then there would truly be no defense. The attacker is only able to work with all the traffic shared between parties, as well as traffic shared in the past.

5. Why is it important that protocols are asynchronous?

Protocols are asynchronous because communication between parties often run through a distributed system. That is, traffic between parties travel through systems that run independently of each other, and can't tell each other exactly what to do, rather, it can only give its next handler a value that can be understood independent of systems.

Lecture 60

1. Would the Needham-Schroeder protocol work without nonces?

No. In steps 4 and 5, A and B use the nonce to confirm the established key. Without the freshness aspect, then the keys may become asynchronous. A may send a message to B, and if B doesn't have the key, it can't decrypt it.

2. For each step of the NS protocol, answer the two questions on slide 5.

A -> S : A, B, N_a

The sender, A, is trying to tell the trusted key server, S, that it is trying to communicate with B, marked with a freshness time of N_a.

The receiver, S, is entitled to believe that at freshness time N_a, A is trying to communicate with B.

S -> A : $\{N_a, B, K_{ab}, \{K_{ab}, A\}_{Kbs}\}\ K_{as}$

The sender, trusted key server S, is trying to tell A that is has received the nonce of N_a, A's intended receiver, B, and that it has generated a shared key for A and B, K_{ab}. S signs K_{ab} and A with K_{bs} so that only B can read it. S delivers this message back to A.

The receiver, A is entitled to believe that S has given it a valid key to open communication between itself and B, and that it can deliver this shared key to B in the signed portion of the message $\{K_{ab}, A\}_{Kbs}$

A -> B : $\{K_{ab}, A\}_{Kbs}$

The sender, A is trying to tell B that it is trying to establish a secure connection with the key K_{ab} that can be shared between them. The message is signed by K_{bs}, so only B and S can access the message, which is the shared key and identity of A.

The receiver, B, is entitled to believe the K_{ab} is the shared key between A and itself.

B -> A : $\{N_b\}K_{ab}$

The sender, B, is trying to tell A that it received the key at freshness time N_b.

The receiver A, is entitled to believe that B received the shared key K_{ab}.

A -> B : $\{N_b - 1\}K_{ab}$

The sender, A is trying to tell B that it received the key at freshness time N_b. A sends $N_b - 1$ instead of just N_b because if A send $\{N_b\}K_{ab}$, it could mean that A was actually unable to access the message.

The receiver B, is entitled to believe that A is aware that it has received the share key and is reader for secure communication.

Lecture 61

1. As in slide 5, if A's keys were later changed, after having K_{as} compromised, how could A still be impersonated?

Because step 3 is not protected by a nonce, an impersonator could send an expired message to B. This expired message would be $\{K_{ab}, A\}K_{bs}$. Though A's keys were changed, the key between B and S have not. Therefore, B can strip K_{bs} and reveal the expired key K_{ab}, thinking it was fresh, and establishing communication between itself and the impostor.

2. Is it fair to ask the question of a key being broken?

Yes and no. While most cryptographic protocols are strong, some regular protocol steps may not.

3. How might you address these flaws if you were the protocol designer?

It would perhaps be advantageous if B, S, and A were able to observe the current nonces at any available time. Perhaps increasing communication between B and the secure server S could keep key values even more fresh. Perhaps S could sign the nonce of A along with the shared key and identity of A before it sent a return message to A to give to B.

Lecture 62

1. What guarantees does Otway-Rees seem to provide to A and B?

It guarantees that A and B are trying to communicate within the same session.

2. Are the guarantees that Needham-Shroeder provides that Otway-Rees does not or vice versa?

The Needham-Shroeder guarantees that A is able to confirm to B that is has received the key, while Otway-Rees does not. Otway-Rees guarantees that A, B, and S are trying to work within the same session, while you may used expired keys in Needham-Shroeder to impersonate a party.

3. How could you fix the flawed protocol from slide 4?

When A sends a message to B, B would have both the private key and the public key of A. Before B sends the return message to its C, it can check to see if using C's private key on A's private key would yield the same results.

Lecture 63

1. Why is the verification of protocols important?

Protocols are very difficult to design, and even widely used, seemingly perfect protocols have had flaws. Therefore, in designing a protocol, it is advantageous to verify these protocols, to make sure that compromised states cannot be reached or exist.

2. What is a belief logic?

A belief logic is a systematic way to provide reasoning for certain beliefs. Tools for belief logic include propositional and predicate logic, standard primitives and rules of reference.

3. A protocol is a program; where do you think beliefs come in?

The belief is the underlying algorithm that carries out a protocol. When one is implementing a protocol, they have an underlying algorithm that they are trying to implement in order to achieve the goals of the protocol. If the logic of the algorithm is flawed, (such as XORing messages to discover keys) then the protocol can be flawed. In addition, even if an underlying algorithm may seem logically perfect, one may discover that the implementations of the algorithm leads to flaws.

Lecture 64

1. What is modal logic?

Modal logic is a type of formal logic that uses propositional and predicate logic. Primitives define formulas. Inference rules lay out the rules for inferring a new fact, typically backed by the facts that a system already has in its database or history. Together, privatives, inference rules, and background information allow for the deduction of new facts.

2. Explain the intuition behind the message meaning inference rule.

A believes that A and B share a key K that no one else knows. Therefore, if A receives a message encrypted with K, then A can believe the message X came from B, ideally, the only other party that knew the secret key. In other words, if n parties know a key k that no one else knows, then it can be inferred that anytime a message is intentionally encrypted with k, it would have came from one of the n party members.

3. Explain the intuition behind the nonce verification inference rule.

A believes X is fresh. A also believes that B has said X. Therefore, A believes that B believes X. In other words, if a party says a message that another party independently believes is fresh, then that party can believe that the declaring party also thinks its fresh.

4. Explain the intuition behind the jurisdiction inference rule.

B has jurisdiction over X, meaning B can believe X. A believes B, so A would believe X by transitivity. An example would be: if person A conducts a set of studies X, then a person B would believe the studies so long as he believes the person A.

5. What is idealization and why is it needed?

Idealization is essentially mapping the transactions of the protocol to a belief logic. If a protocol was a program, then idealization would be analogous to the in-line comments that explain the intentions of that execution. For instance, to the untrained eye, merge sort looks like an absolutely ridiculous way to sort items in order. However, the logic behind the steps executing merge sort support the goal of sorting items in order.

Lecture 65

1. Why do you think plaintext is omitted in a BAN idealization?

Because principals should know their own information. That is, A should know without a doubt that it wants to communicate with B, and that the nonce it created is accurate to itself.

2. Some idealized steps seem to refer to beliefs that will happen later in the protocol. Why would that be?

Because freshness and time play a role in a protocols. That is, for time-sensitive steps in the protocol, such as session identifier and nonces, it requires a step to be taken earlier in the protocol in order to be effective and correct later in the protocol.

3. One benefit of a BAN proof is that it exposes assumptions. Explain that.

In order to write a systematic proof, meaning that you have to use primitives, functions, and some sort of computation, you have to declare all the necessary variables and assumptions. Therefore, as you are working on a BAN proof, you might stumble upon a step that requires an assumption that you may have no intuitively declared from the beginning. In turn, if this assumption is poor, then it may be a vulnerability to your logic and in turn, the system. For instance, step 3 of the Needham-Shroeder assumes that the key that A sends to B encrypted by the key shared between B and S is always fresh. However, as we have analyzed the flaws of Needham-Shroeder, we know that this assumption is bad because the K isn't always fresh. As a result, doing a BAN proof on the Needham-Shroeder protocol exposed an assumption that may not be all that safe to the security goals of the protocol.