IK2206 – Internet Security and Privacy

Chapter 11 – Security Handshake Pitfalls

11.1 Login only

- Consider the attacks and their implications carefully.

11.1.1 Shared key

- How can an attacker derive the shared secret from eavesdroppping?

- Why is the lack of mutual authentication significant? How does this allow an attacker to subvert protocol 11.1 ?

  Otherwise Bob authenticates Alice, but Alice does not authenticate Bob.

  If Trudy can receive packets transmitted to Bob's network address, and respond with Bob's network address, then Alice will be fooled into assuming Trudy is Bob.

- What is required of the attacker to allow her to hijack the connection after the authentication?

  If the remainder of the conversation is transmitted without cryptography protection, then Trudy can hijack the conversation after the initial exchange, assuming she can generate packets with Alice's source packet.

- Why is protocol 1.12 and 1.13 an improvement in security over 1.11?

  The random number (1.12) or the timestamp (1.13) are transferred encrypted.

- What kind of attacks will the lack of server state in 1.13 help protect against?

- What additional protection is required for 1.13 to be viable?
  - Bob has to remember all timestamps sent by Alice until they "expire". This foils that someone can use Alice's transmitted K{timestamp} to impersonate Alice.
  - If there are multiple servers for which Alice uses the same secret. Then an eavesdropper who acts quickly can misuse the transmitted encrypted timestamp field. This can be foiled by concatenating the server name with the timestamp.

11.1.2 One-Way Public Key

- Why should keys not be reused for different purposes?

The general rule is that you should not use the same key for two different purposes unless the designs for all uses of the key are coordinated to that an attacker can’t use one protocol to help break another.

This can be foiled if each type has an own structure so that it cannot be mistaken for another type of thing.

- How could a new scheme based on existing keys compromise other, pre-existing schemes?

  ?

11.2 Mutual Authentication

- Notice how having both sides authenticate complicates things.

11.2.1 Reflection Attacks

- What is a reflection attack?

  Trudy wants to impersonate Alice to Bob. Frist Trudy starts sending R2 as challenge to Bob, but when she receives the challenge R1 form Bob she cannot proceed further, because she can’t encrypt R1.

  However, Trudy has managed to get Bob to encrypt R2. Therefore, Trudy opens a second session to Bob with R1 (received challenge from Bob). Trudy receives challenge R3 and K{R1} from Bob. Trudy can’t go any further with this session.

  With K{R1} she can complete the first session because this is the answer which is expected by Bob.

- Why is the symmetry in protocol 11.8 a problem?

  There are two methods how we can fix the problem described in the answer above and both solutions do not allow symmetry:

  - Different keys – have the key used to authenticate Alice be different from the key used to authenticate Bob.
  - Different challenges – Insist that the challenge form the initiator look different form the challenge from the responder achieved by making use of odd and even number.

11.2.2 Password Guessing

- Why is impersonating Bob harder?

  ?

- What is the core of the change in protocol 11.11 from 11.8? Can you formulate a simple rule of thumb that encompasses it?

  ?

11.2.3 Public Keys

- What are the complications of using public keys?

  How does Alice knows Bob public key. Often the situation is that Alice is a human working on different workstations.

11.2.4 Timestamps

- Why should the value sent by Alice be modified by Bob before he encrypts is?

  Otherwise an attacker could use the same encrypted timestamp to impersonate as Bob.

- Why should protocol 11.13 use a timestamp, and not just a random number?

  It is possible to reduce the mutual authentication down to two messages by using timestamps instead of random numbers.

- How can protocol 11.13 be exploited by an attacker? What is needed to prevent this?

  Trudy could misuse Kab{timestamp+1} to impersonate Alice.

  It could be avoided if Bob keeps both timestamps and timestamp+1 in his replay cache.

11.3 Integrity/Encryption for data

11.3.1 Shared Secret

- Why not use use K{R} or K{R+1} as the session key?

  It is unsecure to used guessable nonces (R) as challenge. If an attacker finds out R, he is probably able to decrypt previously transmitted messages.

- What properties do we want from a session key?
  - Must be different for each session
  - Not guessable by an eavesdropper
  - Should not consist of a quantity X encrypted with K_{AB}, where X is a value that can be predicted or extracted by an intruder.

11.3.2 Two-Way Public Key Based Authentication

- How can past conversations be protected against direct attacks on Alice or Bob, i.e., if Bob's key is compromised, how can past conversations still remain secret?

  Alice and Bob, both can use their public keys to encrypt a random number and sign it. Each of them Xor's the others random value to generate a session key. An intruder must hijack both, Alice's and Bob's random number to get to know the session key. (This is almost impossible since Alice and Bob can forget about their random numbers after they agreed to a certain session key).

- Can old conversations be protected in a similar way using shared keys?

  Yes. Alice and Bob can do a Diffie-Hellman key establishment exchange. Even if Trudy overruns both Alice and Bob, she won't be able to decrypt previously recorded conversations.

11.3.3 One-Way Public Key Based Authentication

- What is the significance of Bob knowing that he is talking to a single party throughout the entire conversation?

  ??? A session key is established and used throughout the whole conversation. The key must remain unknown for intruders during the time of the conversation.

11.3.4 Privacy and Integrity (BRIEFLY)

What benefit does adding a sequence number give us?

It prevents from an attacker who records a message and then replays it later. With sequence numbers we can detect messages which are out-of-order.

Why do we need more than just a sequence number?
An attacker can record a message going in one direction and replays it in the other. If the same sequence number is valid in both directions, such a message could be misinterpreted. Different sequence number ranges or a direction bit could solve this issue.

What benefits do we get from key rollovers?
If we reuse sequence number during a communication (because the number is not large enough, after a while we have to start from the beginning), an attacker is able to reply an old recorded message. The best method to prevent this is to change keys periodically.

11.4 Mediated Authentication (with KDC)

11.4.1 Needham-Shroeder

- Wat are the purposes of the various "nonces"?

  The purpose of the nonce is to assure Alice that she is really talking to the KDC.

  For example, if Trudy has a stolen key of Bob and the message where Alice had previously requested a key for Bob. Trudy has just to wait until Alice makes a request to the KDC to talk to Bob. Trudy can than reply the old/stolen message, which looks like an ordinary reply form the KDC.

- What is a "ticket" here?

  It is just a pile of unintelligible bits. The ticket contains the key for the communication between Alice and Bob and the name of the communication partner.

- Why isn't it sufficient to integrity-protect message fragments?

  Otherwise a fragment could be substituted by an attacker with another fragment. If encryption (e.g. using DES) were done in CBC instead of ECB, an intruder would not be able to accomplish the replacement of fragment, because he/she cannot splice pieces of messages when using CBC.

11.4.2 Expanded Needham-Shroeder

- What attack does the expanded Needham-Shroeder prevent?

  Alice requests a none N_B from Bob, which Alice will send to the KDC, and the KDC will package N_B in the ticket to Bob. This will reassure Bob that whoever is talking to him has talked to the KDC since Bob generated N_B.

  Two additional messages at the beginning:

  - Alice -> Bob: I want to talk to you
  - Bob -> Alice: K_{Bob} $\{N_B\}$

  We can prevent the situation that if Trudy finds out Alice's key, then she can't claim to be Alice and get the KDC to give Trudy a shared key to Bob.

11.4.3 Otway-Rees

- What attack does Otway-Rees prevent?

  Eavesdropping and replay attacks

- What is the advantage compared to expanded Needham-Shroeder?

  It does mutual authentication in 5 messages.

- THe principle "the suspicuos party should always generate a challenge" is introduced here. Consider what that means. Why is this a good design principle?

  The suspicious party has to show that he can decrypt or encrypt with the key he should know.

- Why must the nonce used be unpredictable?

  Otherwise Trudy could impersonate Bob to Alice. If Alice is using a sequence number as nonce, Trudy watches and sees that Alice is currently using 007. After that Trudy sends a message to Bob claiming to be Alice with 008.

11.5 Nonce Types

- Why must not a nonce be reused?

  Otherwise an attacker could eavesdrop a nonce send from Alice to Bob or vice versa and the attacker can use the same nonce to impersonate someone.

- What types of nonces are there?

  Timestamp, large random number, sequence number

- Some protocols seen have used timestamps as nonces, what are the potential pitfalls?

  Intruder Eve has to guess the timestamp Bob will use, and she might be off by a minute or two. If the timestamp has coarse granularity, say seconds, Eve has a good chance of being able to impersonate Alice. If the timestamp has nanosecond granularity, it is likewise a random number and the protocol is secure.

11.6 Picking Random Numbers

- What is the difference between a random number and a pseudo-random number?

  A pseudorandom number generator is a deterministic algorithm. The entire sequence it will generate is determined by its initial state. In contrast a random number generator is one that truly pocks numbers unpredictably.

- How can putting a human in the loop add randomness?

  For instance: by timing their keystrokes. Such a technique is useful to obtain a few bits of randomness, perhaps to seed a pseudorandom number generator.

11.7 Performance Considerations

- What are the important metrics used for evaluating performance?
  - Number of cryptographic operations using a private key
  - Number of cryptographic operations using a public key
  - Number of bytes encrypted or decrypted using a secret key
  - Number of bytes to be cryptographically hashed
  - Number of messages transmitted

- What more than the processing is important for performance?

  It is important to consider the real-time cost of performing an authentication. (round-trip message exchanges will certainly be the dominant factor in authentication.

- How can protocols be optimized in this respect?

  The most important optimization that can be added to a protocol is the ability to cache state from a previous authentication to make it easier for the same two parties to authenticate a second time. (E.g. using tickets).

11.8 Authentication Protocol Checklist

- Understanding, not just memorizing, this checklist is a key to understanding not just this chapter, but also much of this course!