CHAPTER 31

Network Security

Solutions to Odd-Numbered Review Questions and Exercises

Review Questions

1. A ***nonce*** is a large random number that is used ***only once*** to help distinguish a fresh authentication request from a repeated one.
3. Both the ***Needham-Schroeder*** and the ***Otway-Rees*** protocols use a ***KDC*** for user authentication.
5. The ***Kerberos TGS*** issues a ticket for the real server and provides the session key between the sender and the receiver.
7. A ***certification authority (CA)*** is a federal or state organization that binds a public key to an entity and issues a certificate.
9. A ***frequently-changed password*** is more secure than a ***fixed password*** but less secure than a ***one-time password.*** However, a one-time password needs more effort from the system and the user. The system needs to check if the password is fresh every time the user tries to use the password. The user needs to be careful not to use the pervious one. A more frequently changed password can be used as an alternative. One solution is that the system initializes the process of changing the password by sending the new password, through a secure channel, and challenging the user to be sure that the right user has received the new password.

Exercises

11.

a. The algorithm meets the first criteria (***one-wayness***). It is not possible to find the original numbers if the digest is given. For example, if we know the digest is 76, we cannot find the original ten numbers. They can be any set of 10 numbers.

b. The algorithm does not meet the second criteria (***weak collision***). If the digest is given, we can create 10 numbers that hash to the same digest. For example, Eve, without knowing the original set of numbers, can intercept the digest of **51** and create the set {12, 23, 45, 12, 34, 56, 9, 12, 34, 14} and send it with the digest **51** to Bob. Bob is fooled and believes that the set is authentic.

c. The algorithm does not meet the third criteria (***strong collision***). If the digest is given, we can create at least two sets of 10 numbers that hash to the same digest. For example, Alice can create two sets {12, 23, 45, 12, 34, 56, 9, 12, 34, 14} and {12, 23, 45, 16, 34, 56, 9, 12, 34, 10} that both hash to **51**. Alice can send the first set and the digest to Bob, but later she can claimed that she sent the second set.

13. The possible number of digests is 2^N because each bit can be in one of the two values (0 or 1).

15. The second and third criteria for a hashing function are closely related to the solution found in problem 14. In the problem we try to related the number of people at the party to the number of days in a year. In a hashing function, we can relate the number of possible messages to the number of possible digests. To understand the problem assume that there are only 10 possible messages (number of people at the party) but there are 365 possible digests.

    a. If a particular digest is given (a particular birthday), the probability that Eve can find one of the ten messages (one of the ten people in the party) is 0.027 (2.7 percent).This is related to the weak collision. The probability is very weak. That is why it is called ***weak collision***.

    b. The probability that Alice can create two or more messages with the same digests is the probability of finding two or more people with the same birthday in a party. If the number of possible messages is 10 and the number of possible digest is 365, this probability is 0.117 or (11 percent). That is why this criterion is called ***strong collision***. The probability is higher. It is more probable that Alice can find two or messages with the same digest than Eve can find a message with a given digest.

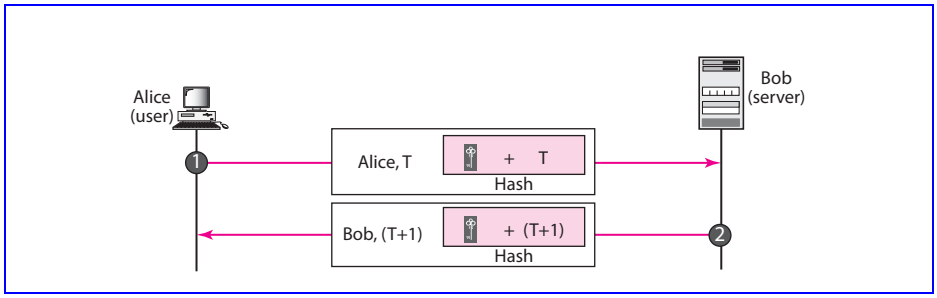
    The above discussion leads us to the point that we should worry more about the second criterion that the first. To decrease the probability of both criteria, we need to increase the number of possible digests and the number of possible messages. We need to increase the number of bits in a digest and impose a minimum number of bits on messages.

17. The whole idea of a sophisticated hash function such as ***SHA-1*** is that the partial digest of each block is dependent on the partial digest of the previous block and the message on the current block. Each block mingles and mixes the bits in a such a way that changing even one bit in the last block of the message may changed the whole final digest.

19. It is normally both. The entity authentication (based on the PIN) is needed to protect the person and the bank in case the money card is stolen. The message authentication is normally needed for the entity authentication.

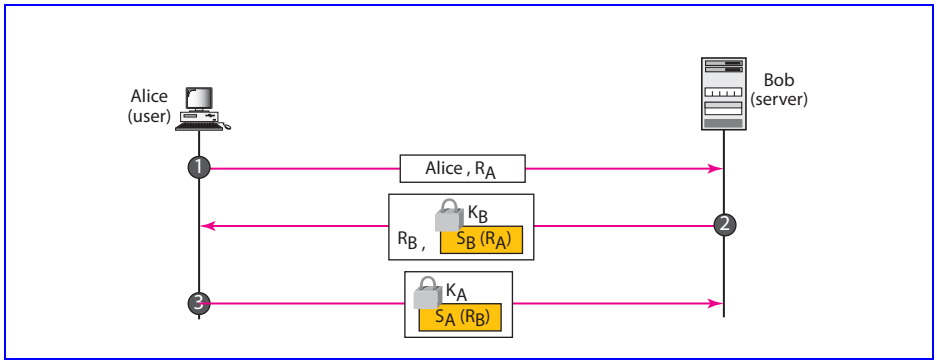
21. Figure 31.1. shows one scheme. Note that the scheme forces Bob to use the timestamp which is related to the timestamp used by Alice (T+1), this ensures that the two messages belongs to the same session.

Figure 31.1 *Solution to Exercise 21*



23. Figure 31.2 shows one simple scheme. Note that in the second message, Bob signs the message with his private key. When Alice verifies the message using Bob's public key, Bob is authenticated for Alice. In the third message, Alice signs the message with her private key. When Bob verifies the message using Alice's public key, Alice is authenticated for Bob.

Figure 31.2 *Solution to Exercise 23*



25. The ***timestamp*** definitely helps. If Alice adds a timestamp to the password before encrypting, the university, after decrypting, can check the freshness of the plaintext. In other words, adding a timestamp to a password, is like creating a new password each time.

27. If the ***KDC*** is down, nothing can take place. KDC is needed to create the session key for the two parties.

29. If the ***trusted center*** is down, Bob cannot obtain his certificate. Bob still can use his public key if the other party does not ask for a certificate.

31. See Figure 31.3. The shaded area shows the encryption/decryption layer.

Figure 31.3 *Solution to Exercise 31*