11210/IIS500200 Network Security, Fall 2023

Homework 2_3

111061548 游鎮遠

1.(a)

- It can be applied to a block of data of any size.
  Yes. Since we are using a fixed value of n and the summation of all the elements of x, hence H can be applied to a block of data of any size. All the data needs to be summed up and modulo function needs to be applied.

- It produces a fixed-length output.
  Yes. Since the value of n is fixed, the modulo function will always result in the range of 0 to n-1. Hence, the function will always produce fixed length outputs.

- It is relatively easy to compute for any given input.
  Yes. The hash function is definitely a relatively easy function using just summation and modulo. The only constraint will be summing a large quantity of numbers if data set is large. However the implementation of hardware and software using modulo n is very practical.

- Preimage resistance.
  No. This hash function does not perform well with Preimage Resistance. Since the value of n is fixed, the results will be very limited. So finding a data set x which will result in a particular code will not be very hard. So it is not infeasible to find x such that H(x)=h. This shows that this function lacks strong Preimage Resistance.

- Secondary preimage resistance.
  No. For Secondary Preimage Resistance, this hash function is also less than ideal. An attacker can find a different input with a known input relatively easily, whose square is the same as the result modulo n. This shows that this function lacks strong Secondary Preimage Resistance.

- Collision resistance.
  No. This hash function is also problematic for Collision Resistance. This is because it is relatively easy for an attacker to find two different input sequences whose squares sum to the same result modulo n. This violates

the requirement of Collision Resistance because it should be difficult to find different inputs with the same hash value.

1.(b)

$$h = (\sum_{i=1}^{t} a_i^2)\ mod\ n,\ \ M = (120, 145, 224, 657), n = 981$$

$$(120^2 + 145^2 + 224^2 + 657^2)\ mod\ 981 = \ 263$$

2.

The process:

i. $A \rightarrow B: A\ and\ NonceA$

A sends its identity(A) and a nonce(NonceA) to B.

ii. $B \rightarrow A: B, NonceB\ and\ (J, NonceA)K_b$

B sends its identity(B), a nonce(NonceB), and an encrypted message containing J, and NonceA encrypted with B's key (K_b) to A.

iii. $A \rightarrow B: (NonceB)J$

A decrypts the message using J and responds with NonceB encrypted with J.

iv. $B \rightarrow A: B, (NonceA, NonceB, S)K_b$

B decrypts the message using K_b, and both parties confirm their identities by exchanging nonces. B generates a session key (S) and sends it along with the exchanged nonces encrypted with K_b.

A and B share the established session key S for subsequent data exchange.

3.

In a communication system with a replay window, the receiver keeps track of received packets within a certain range to prevent replay attacks.

(a) Incoming packet with sequence number 99:

Since 99 is outside the current replay window (111 to 430), the receiver will likely reject the packet as it falls outside the acceptable range. The receiver will not process or accept the packet.

The window parameters remain the same: 111 to 430.

(b) Incoming packet with sequence number 420:

420 is within the current replay window (111 to 430), so the receiver will process and accept the packet.

After processing the packet, the window parameters may shift. The window might move forward, and the new window could be something like 112 to 431 or a similar range, depending on the specific design of the replay window mechanism.

(c) Incoming packet with sequence number 566:

Similar to scenario (a), 566 is outside the current replay window (111 to 430). The receiver will likely reject the packet.

The window parameters remain the same: 111 to 430.

In summary:

For packets within the window, the receiver processes and accepts them.

For packets outside the window, the receiver typically rejects them to prevent replay attacks.

After processing a packet within the window, the window may advance.

The specific behavior can depend on the design of the replay protection mechanism and the security requirements of the system.

4.

- IPSec:

  IPSec works on network layer of OSI model. Since it works on network layer, it provides security of data which travelled between the two clients.

  IPSec provides the data integrity, authentication and confidentiality over the internet.

  A single IPSec tunnel secure all the communication between the device.

  IPSec provide two security protocol for protecting the data.

  - Authentication Header(AH):

    Authentication header protect data from the IP header to the transport header against the replay and cut and post attack.

  - Encapsulating Security Payload(ESP):

    ESP encapsulate the data and works between the IP header and TCP header. ESP provides security against the eavesdropping, replay, cut-and paste kind of attack.

    Thus, inserting the bogus packet into the actual data attacker will not make any affect on security of a system which has IPSec security.

- SSL:

  SSL work on application layer of the OSI model.

  SSL is used to provide the security on web-based communication over the internet.

  SSL is a standard security technology for establishing an encrypted link between the client and the server.

  All browsers have the capability to use secure web using SSL protocol.

  Since the SSL works only on application layer it does not look at the other part of the security. Thus, by inserting the bogus packet an attacker can make some effect on the communication between the clients.

5.

No, the email containing phrase " Homework 2 is out" may be spread on several packets. A stateless packet filter is unable to keep in mind a state from prior packets.