Module 21: Cryptography

CyberOps Associate v1.0

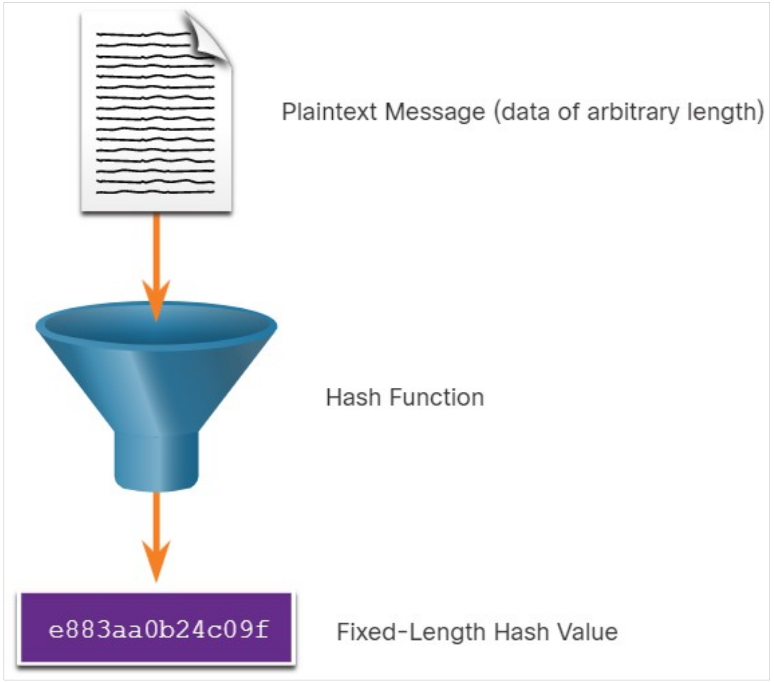
21.1 Integrity and Authenticity

Securing Communications

- Organizations must provide support to secure the data internally as well as externally.
- The four elements of securing communications are:
  - **Data Integrity** - Guarantees that the message was not altered.
  - **Origin Authentication** - Guarantees that the message is not a forgery and it actually comes from whom it states.
  - **Data Confidentiality** - Guarantees that only authorized users can read the message.
  - **Data Non-Repudiation** - Guarantees that the sender cannot repudiate, or refute, the validity of a message sent.

Functions

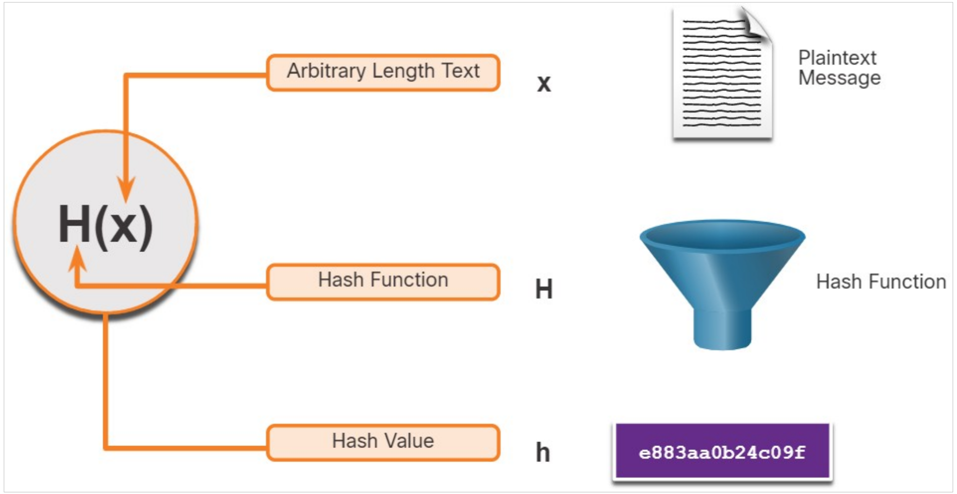
- Hashes are used to verify and ensure data integrity.
- Hashing is based on a one-way mathematical function that is relatively easy to compute, but significantly harder to reverse.
- A hash function takes a variable block of binary data, called the message, and produces a fixed-length, condensed representation, called the hash.
- The resulting hash is also sometimes called the message digest, digest, or digital fingerprint.
- With hash functions, it is computationally infeasible for two different sets of data to come up with the same hash output.
- Every time the data is changed or altered, the hash value also changes.



Cryptographic Hash Operation

- Mathematically, the equation ***h= H(x)*** is used to explain how a hash algorithm operates.
- As shown in the figure, a hash function H takes an input x and returns a fixed-size string hash value h.

- A cryptographic hash function should have the following properties:
  - The input can be any length.
  - The output has a fixed length.
  - H(x) is relatively easy to compute for given x.
  - H(x) is one way and not reversible.
  - H(x) is collision free, meaning that two different input values will result in different hash values.

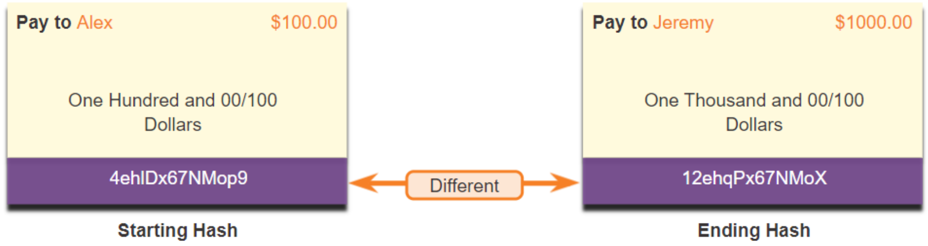


MD5 and SHA

- Hash functions are used to ensure the integrity of a message either accidentally or intentionally.
- In the figure, the sender is sending a $100 money transfer to Alex. The sender wants to ensure that the message is not altered on its way to the receiver.

There are four well-known hash functions:

- **MD5 with 128-bit digest** - A one-way function that produces a 128-bit hashed message. MD5 is a legacy algorithm.
- **SHA-1 -** Very similar to the MD5 hash functions. SHA-1 creates a **160-bit** hashed message and is slightly slower than MD5.
- **SHA-2 -** If you are using SHA-2, then **SHA-256**, **SHA-384**, and **SHA-512** algorithms should be used.



The sender wants to ensure that the message is not altered on its way to the receiver.

1. The sending device inputs the message into a hashing algorithm and computes its fixed-length hash of 4ehiDx67NMop9.
2. This hash is then attached to the message and sent to the receiver. Both the message and the hash are in plaintext.
3. The receiving device removes the hash from the message and inputs the message into the same hashing algorithm. If the computed hash is equal to the one that is attached to the message, the message has not been altered during transit. If the hashes are not equal, as shown in the figure, then the integrity of the message can no longer be trusted.

- **SHA-3 -** Next-generation algorithms and should be used whenever possible.

MD5 and SHA (Contd.)

- While hashing can be used to detect accidental changes, it cannot be used to guard against deliberate changes that are made by a threat actor.
- There is no unique identifying information from the sender in the hashing procedure.
- This means that anyone can compute a hash for any data, as long as they have the correct hash function.
- Therefore, hashing is vulnerable to man-in-the-middle attacks and does not provide security to transmitted data. To provide integrity and origin authentication, something more is required.

Note: *Hashing algorithms only protect against accidental changes and does not protect the data from changes deliberately made by a threat actor.*

Origin Authentication

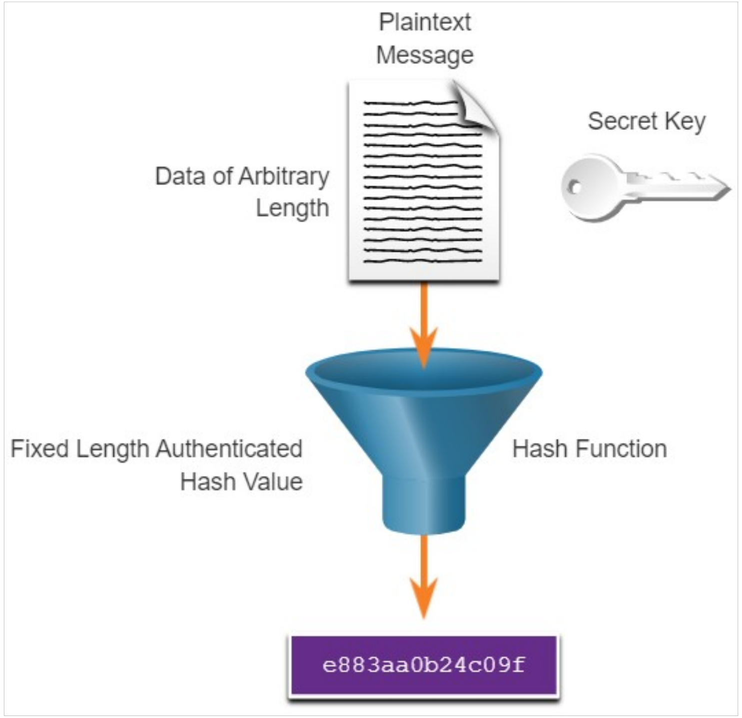
- To add origin authentication and integrity assurance, use a **Keyed-hash Message Authentication Code (HMAC)**.
- **HMAC** uses an additional secret key as input to the hash function.

Note: *Other* ***Message Authentication Code (MAC)*** *methods are also used. However,* ***HMAC*** *is used in many systems including* ***SSL****,* ***IPsec****, and* ***SSH****.*

Origin Authentication (Contd.)

HMAC Hashing Algorithm

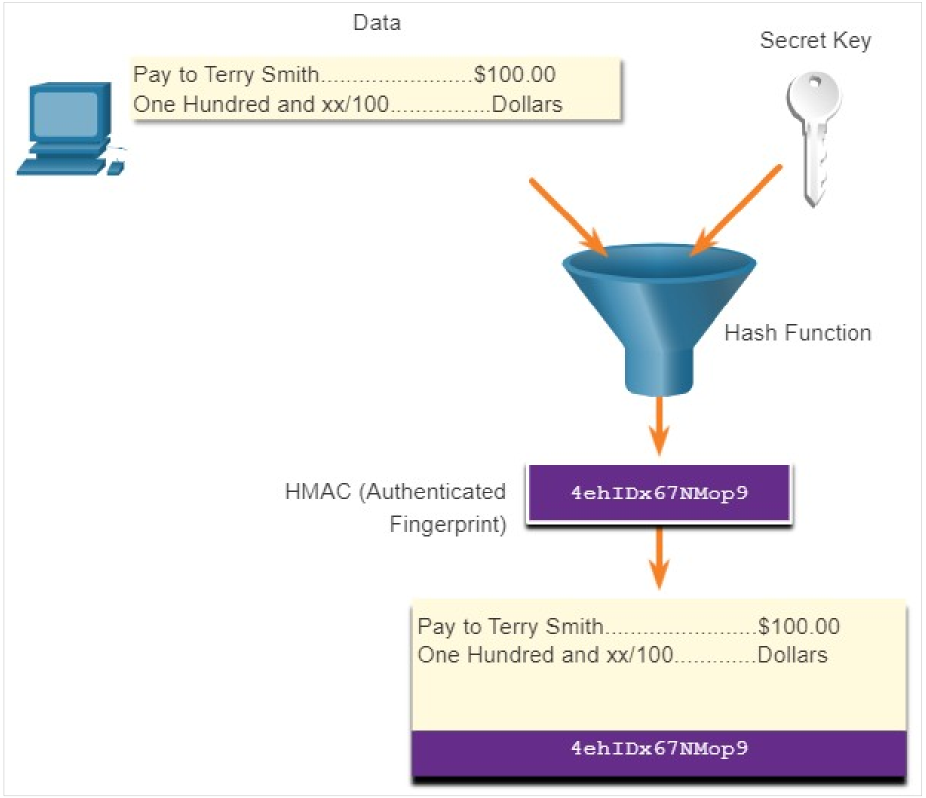
- An HMAC is calculated using any cryptographic algorithm that combines a cryptographic hash function with a secret key.
- Only the sender and the receiver know the secret key, and the output of the hash function depends on the input data and the secret key.
- Only parties who have access to that secret key can compute the digest of an HMAC function.
- If two parties share a secret key and use HMAC functions for authentication, a properly constructed HMAC digest of a message that a party has received indicates that the other party was the originator of the message.



Origin Authentication (Contd.)

Creating the HMAC Value

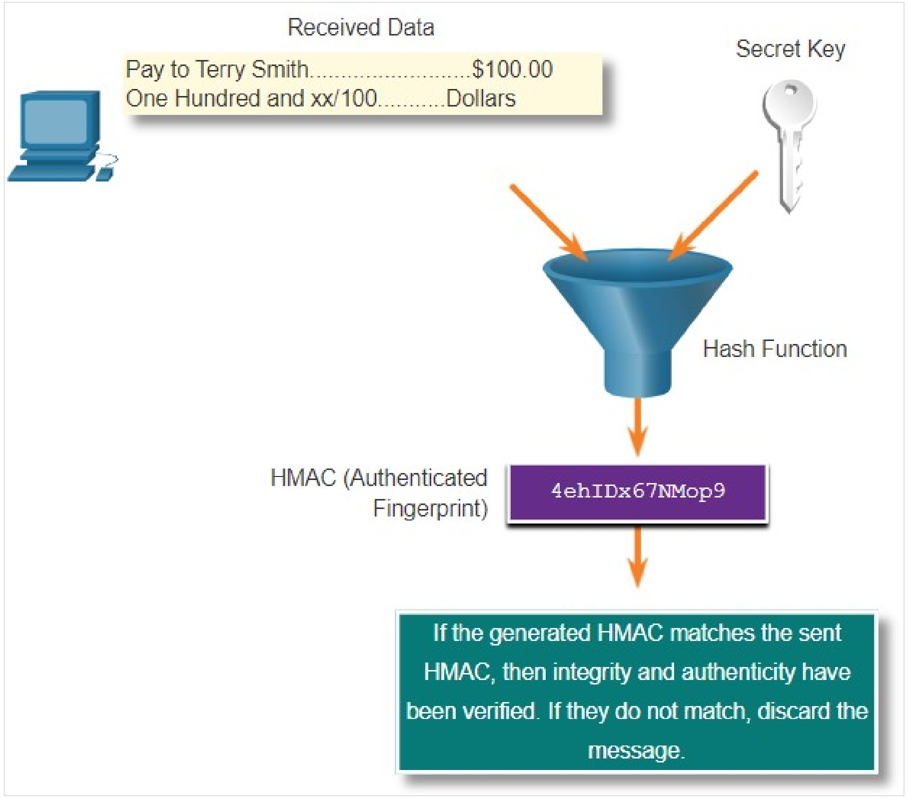
- As shown in the figure, the sending device inputs data into the hashing algorithm and calculates the fixed-length HMAC digest.
- This authenticated digest is then attached to the message and sent to the receiver.



Origin Authentication (Contd.)

Verifying the HMAC Value

- In the figure, the receiving device removes the digest from the message and uses the plaintext message with its secret key as input into the same hashing function.
- If the digest that is calculated by the receiving device is equal to the digest that was sent, the message has not been altered.
- Additionally, the origin of the message is authenticated because only the sender possesses a copy of the shared secret key. The HMAC function has ensured the authenticity of the message.



Origin Authentication (Contd.)

Cisco Router HMAC Example

- In the figure, HMACs are used by Cisco routers that are configured to use Open Shortest Path First (OSPF) routing authentication.
- R1 is sending a link state update (LSU) regarding a route to network 10.2.0.0/16:
  - R1 calculates the hash value using the LSU message and the secret key.
  - The resulting hash value is sent with the LSU to R2.
  - R2 calculates the hash value using the LSU and its secret key. R2 accepts the update if the hash values match. If they do not match, R2 discards the update.