


6G7Z1009: Introduction to Computer Forensics and Security

Authentication and Hash function


Reading List

- N. Ferguson, B. Schneier, T. Kohno, Cryptography Engineering: Design Principles and Practical Applications, (1st Edition) 2010, John Wiley.
- W. Stallings, Cryptography and Network Security: Principles and Practice (5th Edition), 2010, Printice Hall (Chapter 11)
- M. Stamp, Information Security. Principles and Practice (2nd Edition), 2011, John Wiley. (Chapter 5)
- J. Erickson, Hacking: The Art of Exploitation (2nd Edition), 2008 (Chapter 7)
- Behrouz Forouzan, Cryptography and Network Security, The McGraw-Hill Companies. (Chapter 11 and 12)
- Online resources:
  - Bruce Schneier, The Skein Hash Function Family, https://www.schneier.com/skein.html


Message Authentication

- The cryptographic algorithms in previous lectures provide secrecy, or confidentiality, but not integrity. However, some cases where we may not even need secrecy but instead must have integrity.


Message Authentication

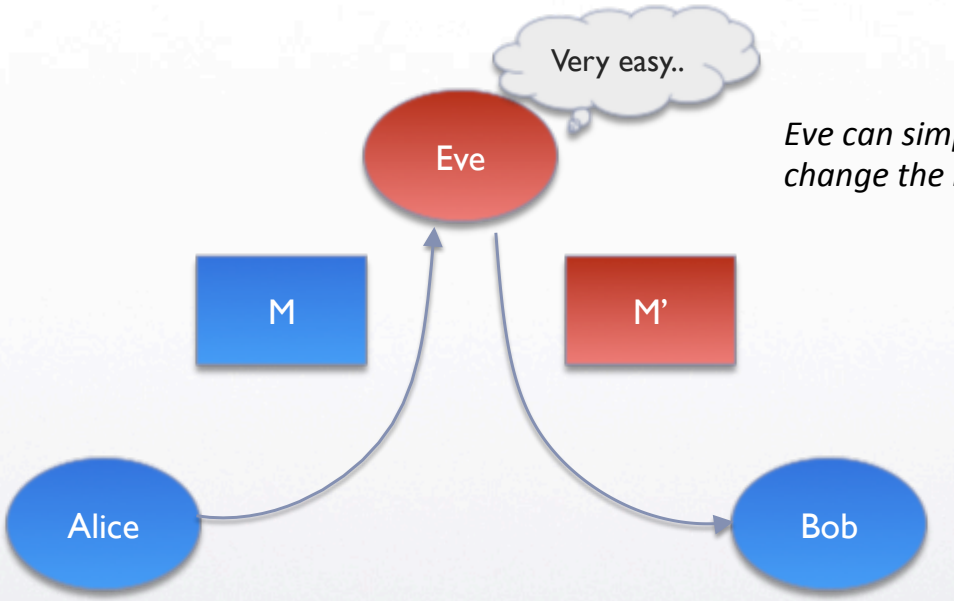
- A message, file or document is said to be authentic if it is genuine and comes from its alleged source.
- Message authentication is a procedure that allows communicating parties to verify that received messages are authentic.
- Need to verify that:
  - the contents have not been altered;
  - the source is authentic.
- May need to verify timeless of the message (it has not been artificially delayed and replayed) and its sequence relative to other messages between the parties.


Message Authentication

- Message Authentication is concerned with:
  - protecting the integrity of a message
  - validating identity of originator


Message Authentication

- Communication without authentication

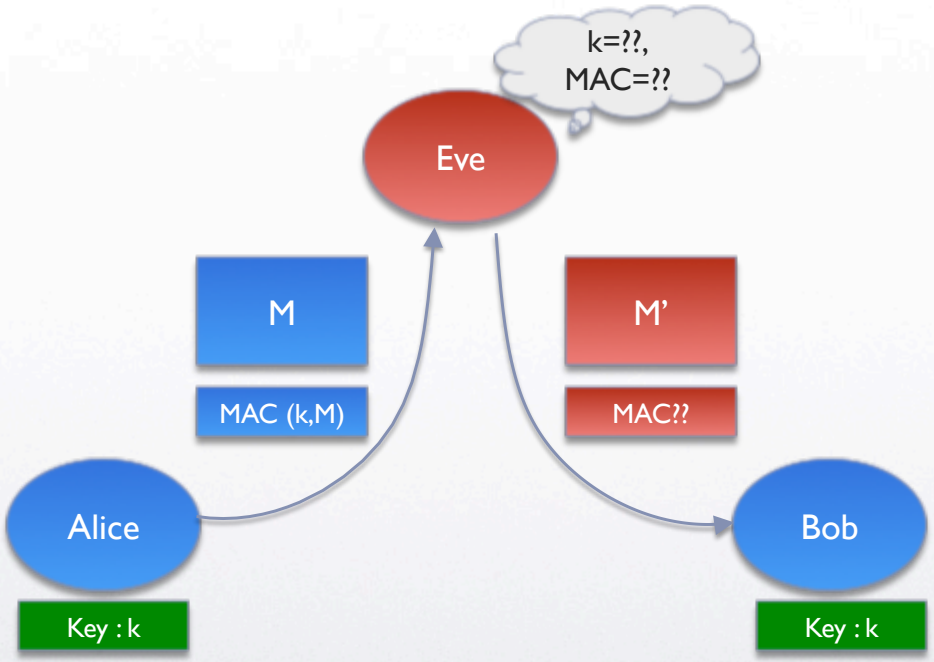


Eve can simply change the message

Shared key *k* to generate authenticate message


Message Authentication

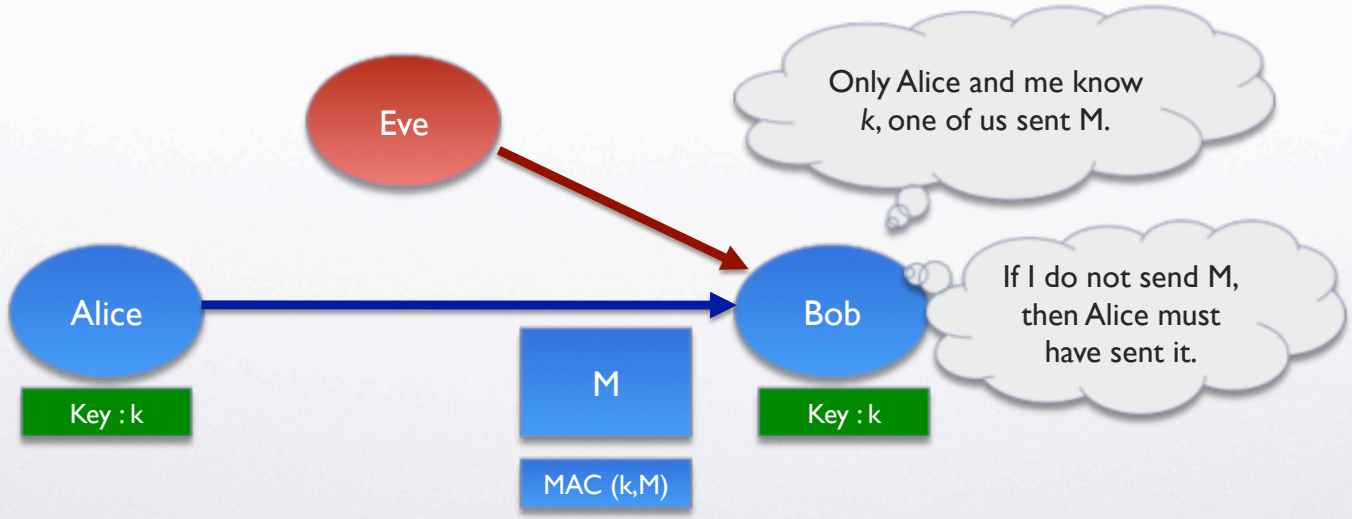
- Integrity protection with MAC



Shared key *k* to generate authenticate message

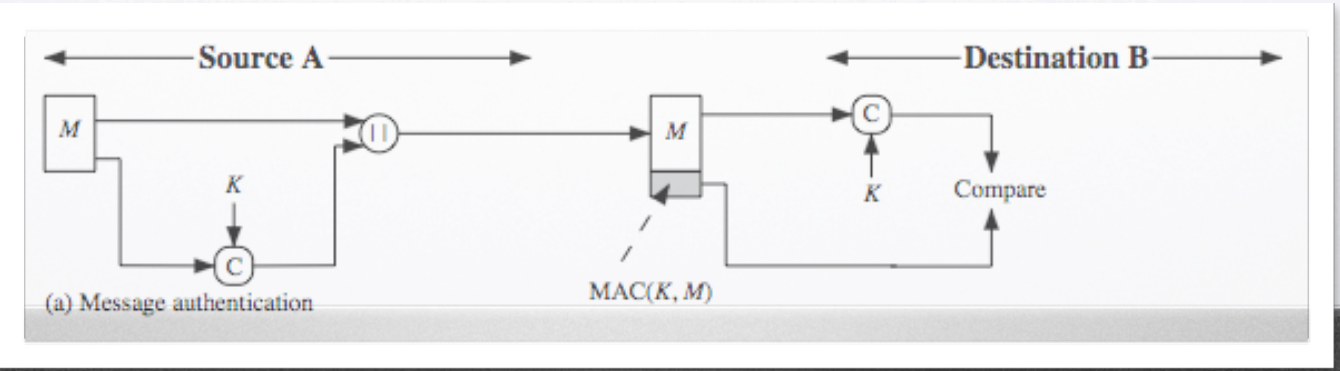

Message Authentication

- MAC Authentication
- MAC allows two or more mutually trusting parties to authenticate messages sent between members



Message Authentication Code (MAC)

- A function of the message and a secret key that produces a fixed-length value that serves as the authenticator
- Generated by an algorithm :
  - Generated from message + secret key : MAC = C(K,M)
  - A small fixed-sized block of data
  - Appended to message as a signature when sent
- Receiver performs same computation on message and checks it matches the MAC




MAC properties

- A MAC is a cryptographic checksum

  $MAC = C_K(M)$

  - condenses a variable-length message M
  - using a secret key K
  - to a fixed-sized authenticator
- A many-to-one function
  - potentially many messages have same MAC
  - but finding these needs to be very difficult


Requirements for MACS

- Considering different attacks, the MAC to satisfy the following:
  - knowing a message and MAC, is infeasible to find another message with same MAC
  - MACs should be uniformly distributed
  - MAC should depend equally on all bits of the message


Keyed Hash Functions as MACs

- Want a MAC based on a hash function
  - because hash functions are generally faster
  - crypto hash function code is widely available
- Need a hashing including a key along with message
  - But hashing internally has no key!
- Original proposal:
  - KeyedHash = Hash(Key|Message)
  - some weaknesses were found with this
  - Eventually led to development of HMAC


HMAC

- Hash-based Message Authentication Code
- Developed by Mihir Bellare, Ran Canetti, and Hugo Krawczyk in1996
- Specified as Internet standard RFC2104
- Use cryptographic hash function in combination with a secret key
- Any hash function can be used
  - eg. MD5, SHA-1, RIPEMD-160, Whirlpool
  - HMAC-MD5, HMAC-SHA1, HMAC-RIPEND-160, HMAC-Whirlpool
- HMAC-SHA1 and HMAC-MD5 are used within the IPsec and TLS protocols


Hash Function

- A Hash Function produces a fingerprint of some file/message/ data, h = H(M)
  - Produces fixed-length output h
  - Assume to be public


Requirements for Hash Function

- Can be applied to any sized message M
- Produces fixed-length output h
- Is easy to compute h=H(M) for any message M
- Given h is infeasible to find x. (H(x)=h)
- Given x is infeasible to find y. (H(y)=H(x))
- Is infeasible to find any x,y. (H(y)=H(x))


Hash Function

- A hash function takes a message of arbitrary length and creates a message digest of fixed length.
- Two most promising cryptographic hash algorithms SHA-512 and Whirlpool.
- Other hash functions such as MD5


Hash Function

- Secure Hash Function (SHA)
- SHA originally designed by NIST & NSA in 1993 was revised in 1995 as SHA-1
- US standard for use with DSA signature scheme (standard is FIPS 180-1 1995, also Internet RFC3174. The algorithm is SHA, the standard is SHS)
- based on design of MD4 with key differences and produces 160-bit hash values
- recent 2005 results on security of SHA-1 have raised concerns on its use in future applications


Hash Function

- Secure Hash Function (SHA)
- NIST issued revision FIPS 180-2 in 2002 and adds 3 additional versions of SHA
- SHA-256, SHA-384, SHA-512
- designed for compatibility with increased security provided by the AES cipher
- structure & detail is similar to SHA-1
- hence analysis should be similar but security levels are rather higher


Hash Function

- Secure Hash Function (SHA)

Characteristics of Secure Hash Algorithms (SHAs)

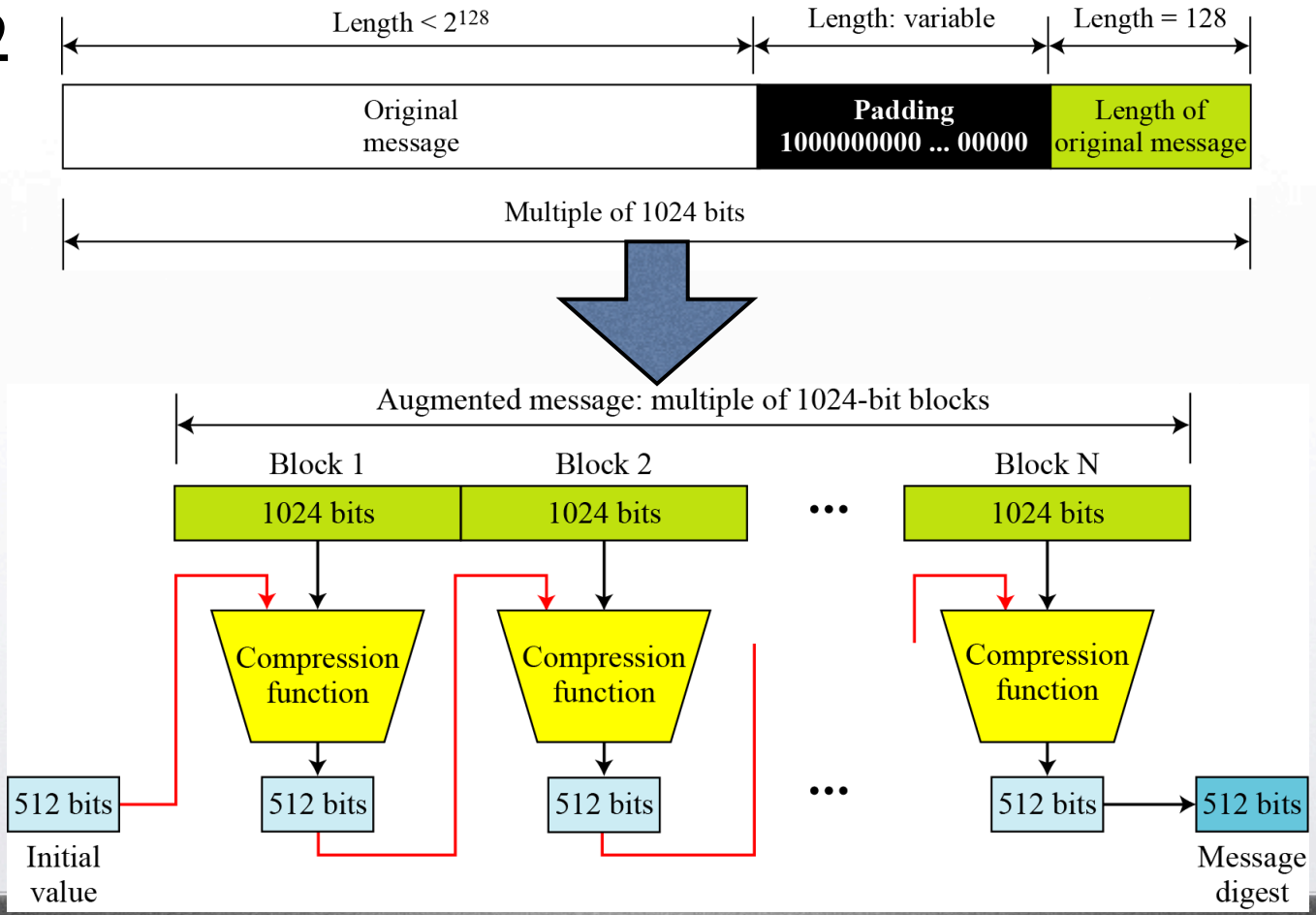
Characteristics	*SHA-1*	*SHA-224*	*SHA-256*	*SHA-384*	*SHA-512*
Maximum Message size	$2^{64} - 1$	$2^{64} - 1$	$2^{64} - 1$	$2^{128} - 1$	$2^{128} - 1$
Block size	512	512	512	1024	1024
Message digest size	160	224	256	384	512
Number of rounds	80	64	64	80	80
Word size	32	32	32	64	64


Hash Function

- SHA-512
- Step 1: Append padding bits
- Step 2: Append length
- Step 3: Initialize hash buffer
- Step 4: Process the message in 1024-bit (128-word) blocks, which forms the heart of the algorithm
- Step 5: Output the final state value as the resulting hash

Hash Function

- SHA-512




Hash Function

- SHA-512 insists that the length of the original message be less than 2^{128} bits.
- SHA-512 creates a 512-bit message digest out of a message less than 2^{128}.


Hash Function

- Examples:

- Suppose we need to send a message that is 2^{128} bits in length. How long does it take for a communications network with a data rate of 2^{64} bits per second to send this message?

- Solution:

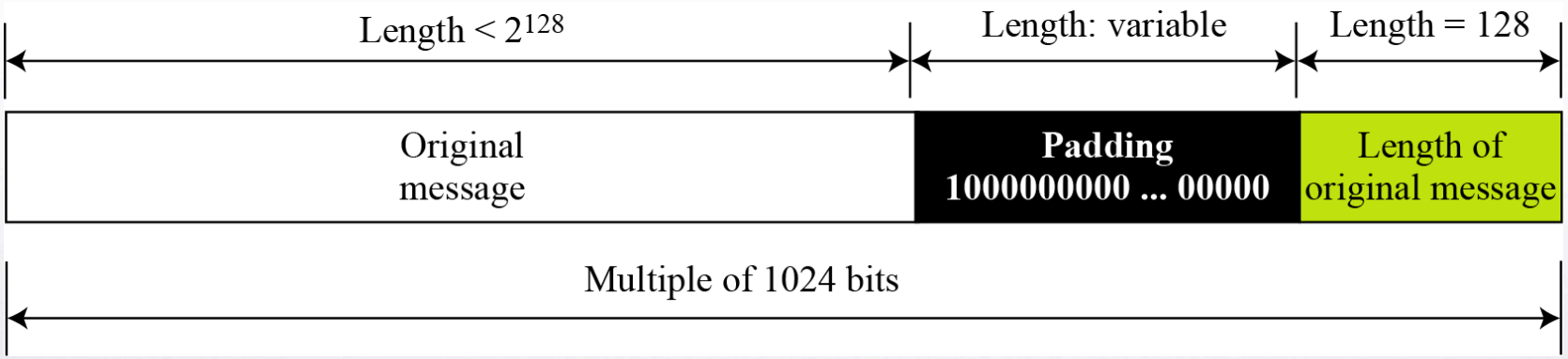
- A communications network that can send 2^{64} bits per second is not yet available. Even if it were, it would take many years to send this message. This tells us that we do not need to worry about the SHA-512 message length restriction.


Hash Function

- Examples:
- How many pages are occupied by a message of 2^{128} bits?
- Solution:
- Suppose that a character is 32, or 2^6, bits. Each page is less than 2048, or approximately 2^{12}, characters. So 2^{128} bits need at least $2^{128} / 2^{18}$, or 2^{110}, pages. This again shows that we need not worry about the message length restriction.


Hash Function

- Padding and length field in SHA-512




Hash Function

• Examples: Do we need padding if the length of the original message is already a multiple of 1024 bits?

•Solutions: Yes we do, because we need to add the length field. So padding is needed to make the new block a multiple of 1024 bits.


Hash Function

- Examples: What is the minimum and maximum number of padding bits that can be added to a message?


Hash Function

- Solutions:

- The minimum length of padding is 0 and it happens when $(-M - 128) \bmod 1024$ is 0. This means that $|M| = -128 \bmod 1024 = 896 \bmod 1024$ bits. In other words, the last block in the original message is 896 bits. We add a 128-bit length field to make the block complete.


Hash Function

- Solutions:

- The maximum length of padding is 1023 and it happens when $(-|M| - 128) = 1023 \bmod 1024$. This means that the length of the original message is $|M| = (-128 - 1023) \bmod 1024$ or the length is $|M| = 897 \bmod 1024$. In this case, we cannot just add the length field because the length of the last block exceeds one bit more than 1024. So we need to add 897 bits to complete this block and create a second block of 896 bits. Now the length can be added to make this block complete.


Hash Function

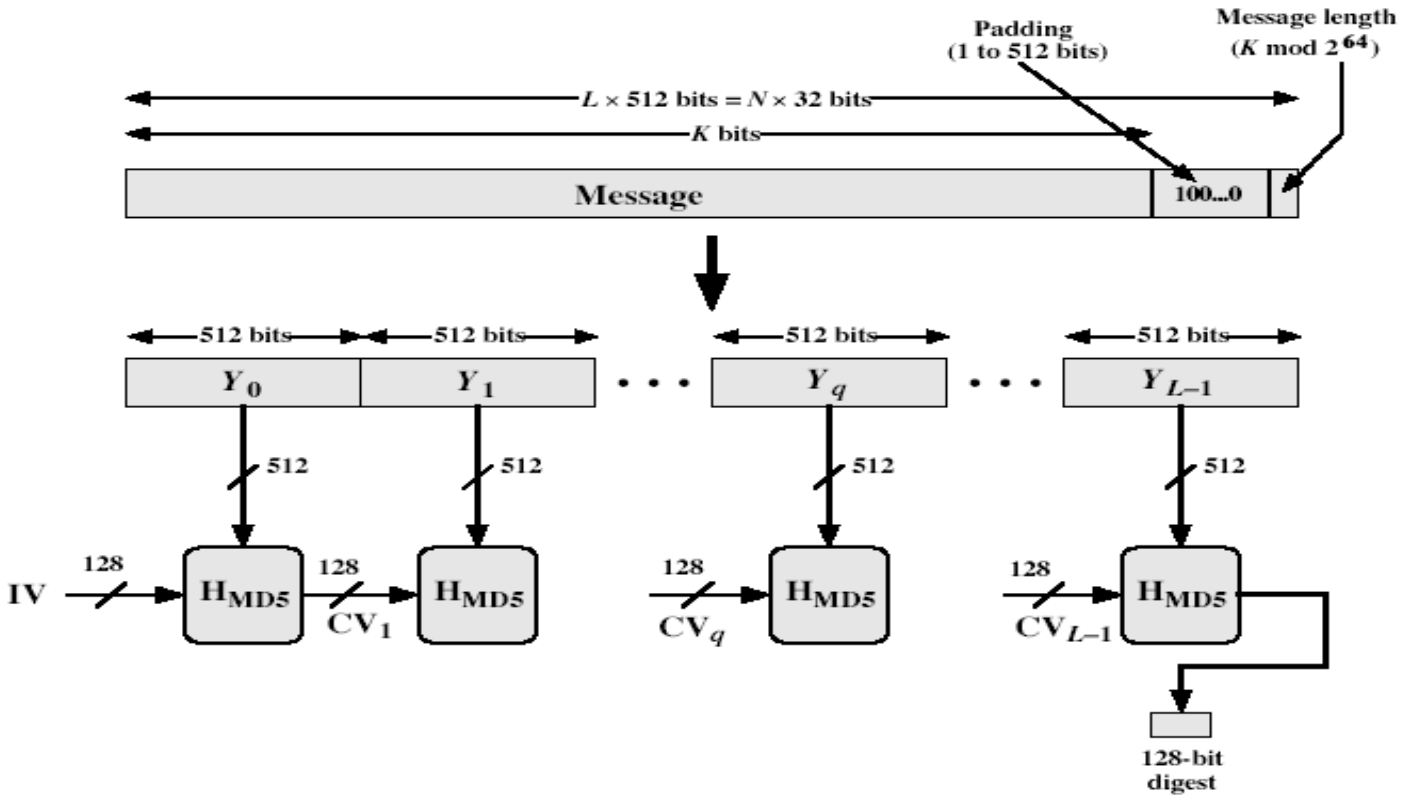
- MD5
  - Designed by Ronald Rivest
  - Latest in a series of MD2, MD4
  - Produces a 128-bit hash value
  - Until recently was the mostly widely used hash algorithm
  - Specified as an Internet standard RFC1321


Hash Function

- MD5 implementation
  - Pad the message so its length (in bits) is 448 mod 512.
  - Append a 64-bit length value to the message
  - Initialise 4 word (128-bit) MD buffer (A,B,C,D)
  - process message in 16-word (512) blocks
  - Output hash value is the final buffer value

Hash Function

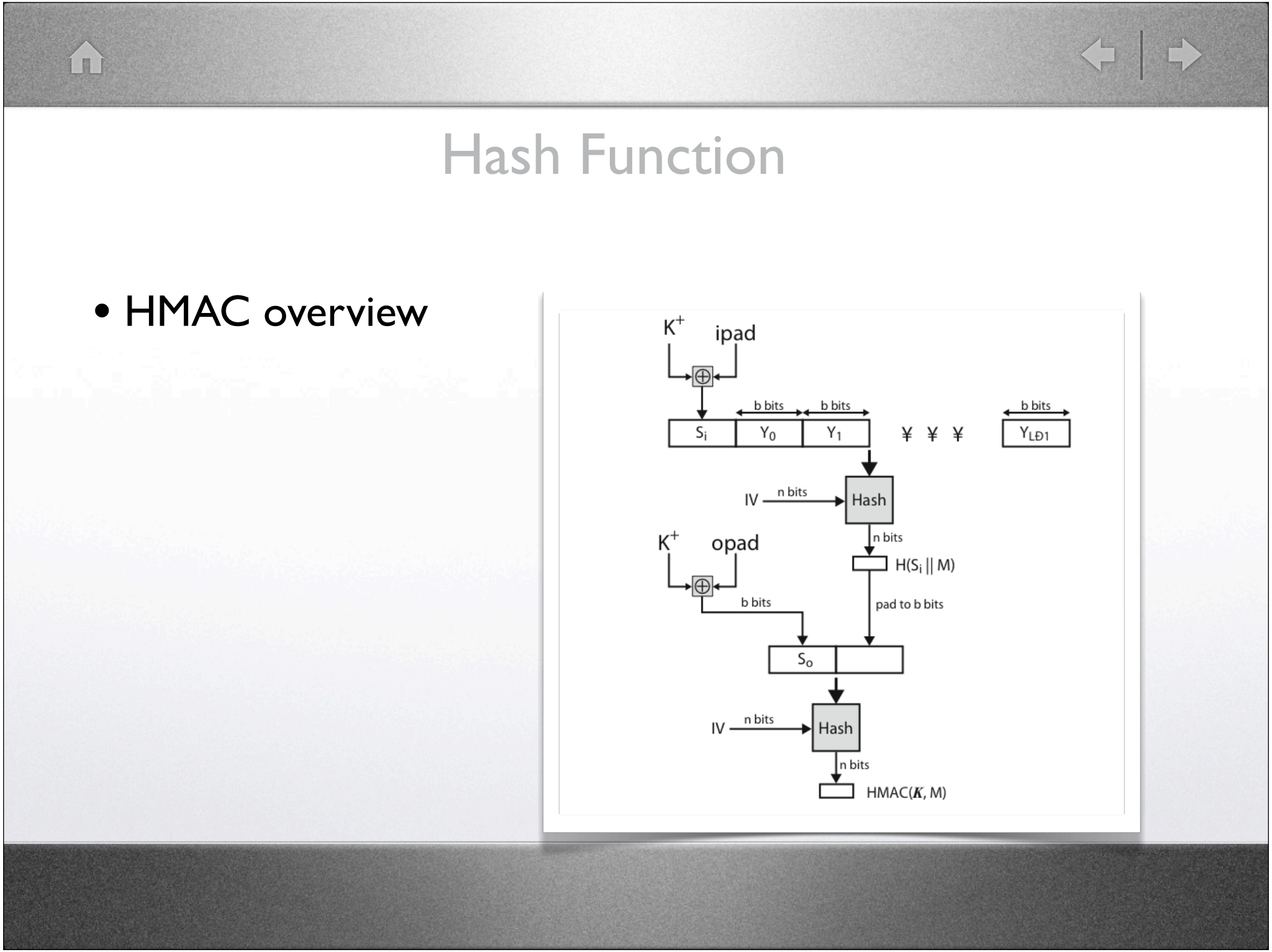
- MD5 implementation




Hash Function

- keyed hash as MAC (HMAC)
- Specified as Internet standard RFC2104
- Uses hash function on the message:
- HMACK = Hash[(K^+ XOR opad) ||

  Hash[(K^+ XOR ipad)||M)]]
- Where K^+ is the key padded out to size and opad, ipad are specified padding constants
- Overhead is just 3 more hash calculations than the message needs alone



Hash Function

- HMAC overview


Hash Function vs MAC algorithm

- Hash function
  - Condenses arbitrary message to fixed size by processing message in blocks through some compression function either custom or block cipher based
  - Usually assume that the hash function is public and not keyed (while MAC which is keyed) and hash used to detect changes to message
  - Can use in various ways with message
  - Most often to create a digital signature


Hash Function vs MAC algorithm

- Message Authentication Code (MAC)
  - fixed sized authenticator for some message
  - to provide authentication for message
  - by using block cipher mode or hash function


Digital Signature

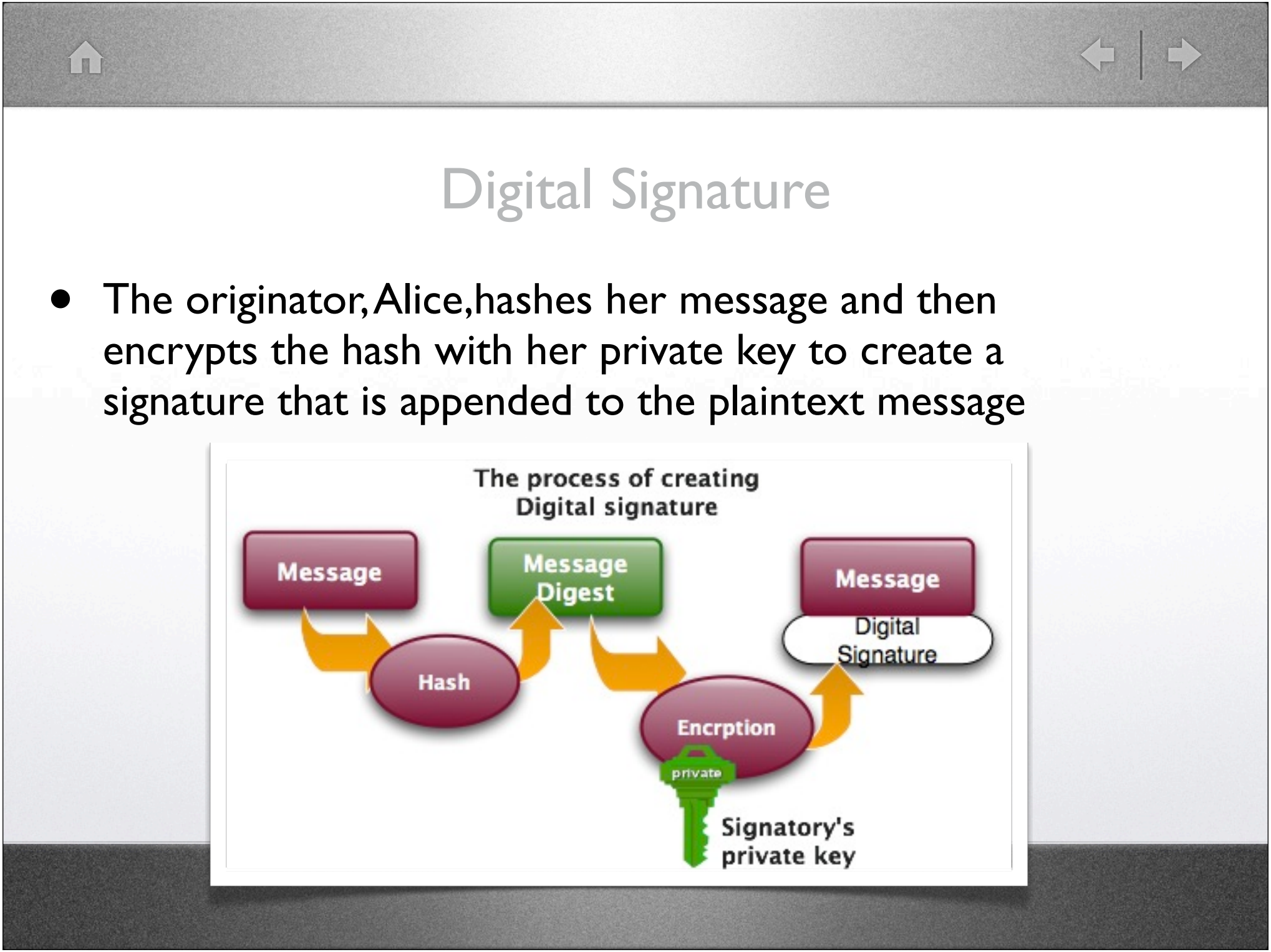
• Another way to provide message integrity and message authentication is a digital signature. A MAC uses a secret key to protect the digest; a digital signature uses a pair of private-public keys.


Digital Signature

- Digital Signature: ensure the origin and the integrity of a message via hashing.
- The originator of a message uses a signing key (private key) to sign the message and send the message and its digital signature to a recipient
- The recipient uses a verification key (public key) to verify the origin of the message and that it has not been tampered with while in transit


Digital Signature

- Digital Signature: ensure the origin and the integrity of a message via hashing.
- The originator of a message uses a signing key (private key) to sign the message and send the message and its digital signature to a recipient
- The recipient uses a verification key (public key) to verify the origin of the message and that it has not been tampered with while in transit

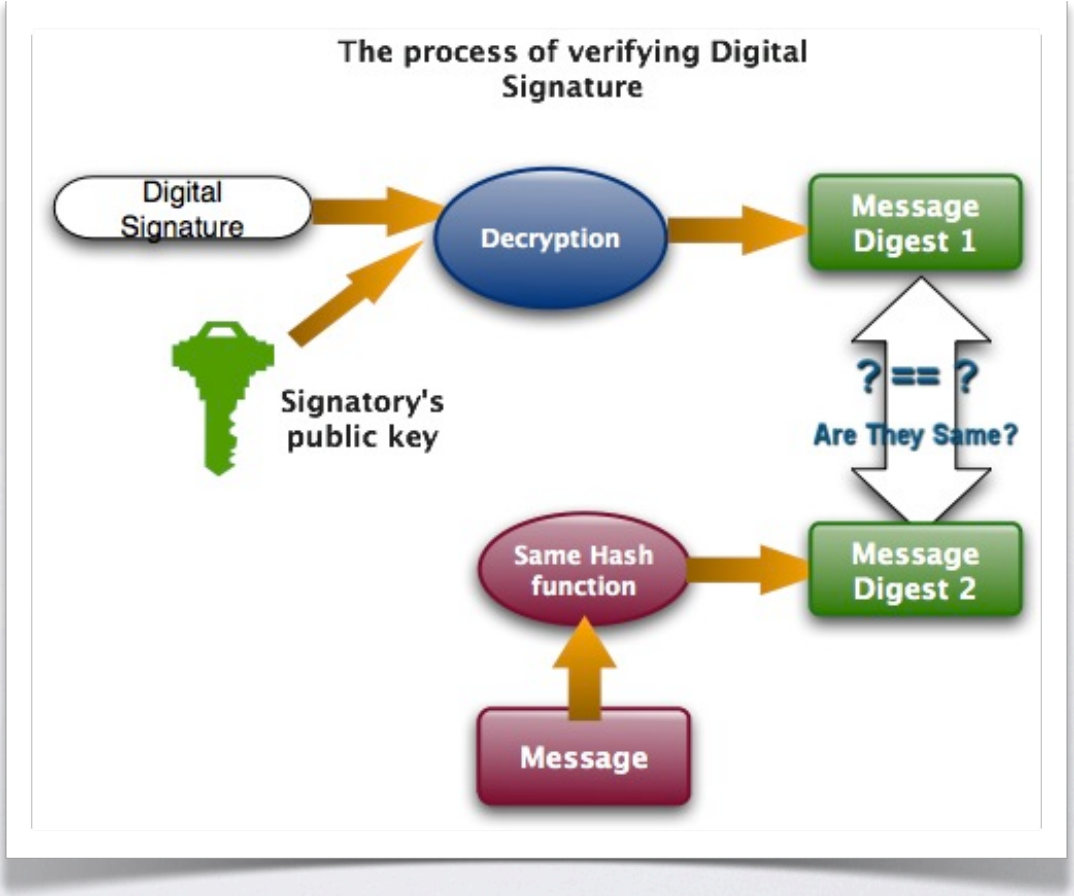


Digital Signature

- The originator, Alice, hashes her message and then encrypts the hash with her private key to create a signature that is appended to the plaintext message

Digital Signature

- The recipient, David, uses Alice's public key to verify the origin of the message by decrypting the




Digital Signature

- Problems of Digital signature
  - Trust issue on the real identity of the signer
  - Forging signatures when attackers intercepts message and private key
  - Digital signature only ties a message to a person to a private key, not to a person


Digital Signature

- Problems of Digital signature
  - A digital signature does not provide privacy
  - If there is a need for privacy, another layer of encryption/ decryption must be applied.
  - There are some attacks
  - Solution --- move to PKI


Summary

- Message authentication code
- Hash functions
- Digital signature