Cryptography Basics

Henry Wise Wood Math and Computer Science Club

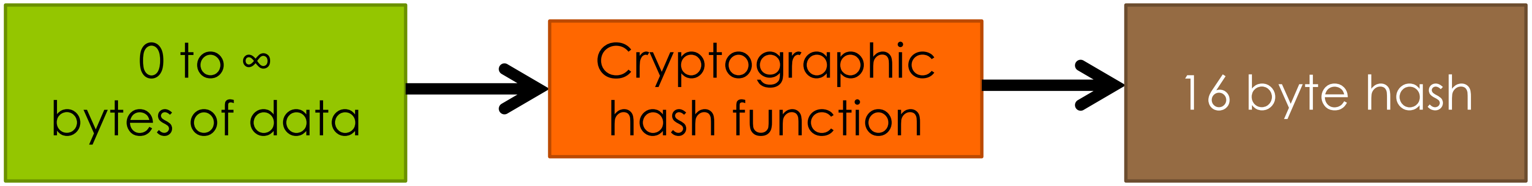
December 12, 2011

Why do I need Cryptography?

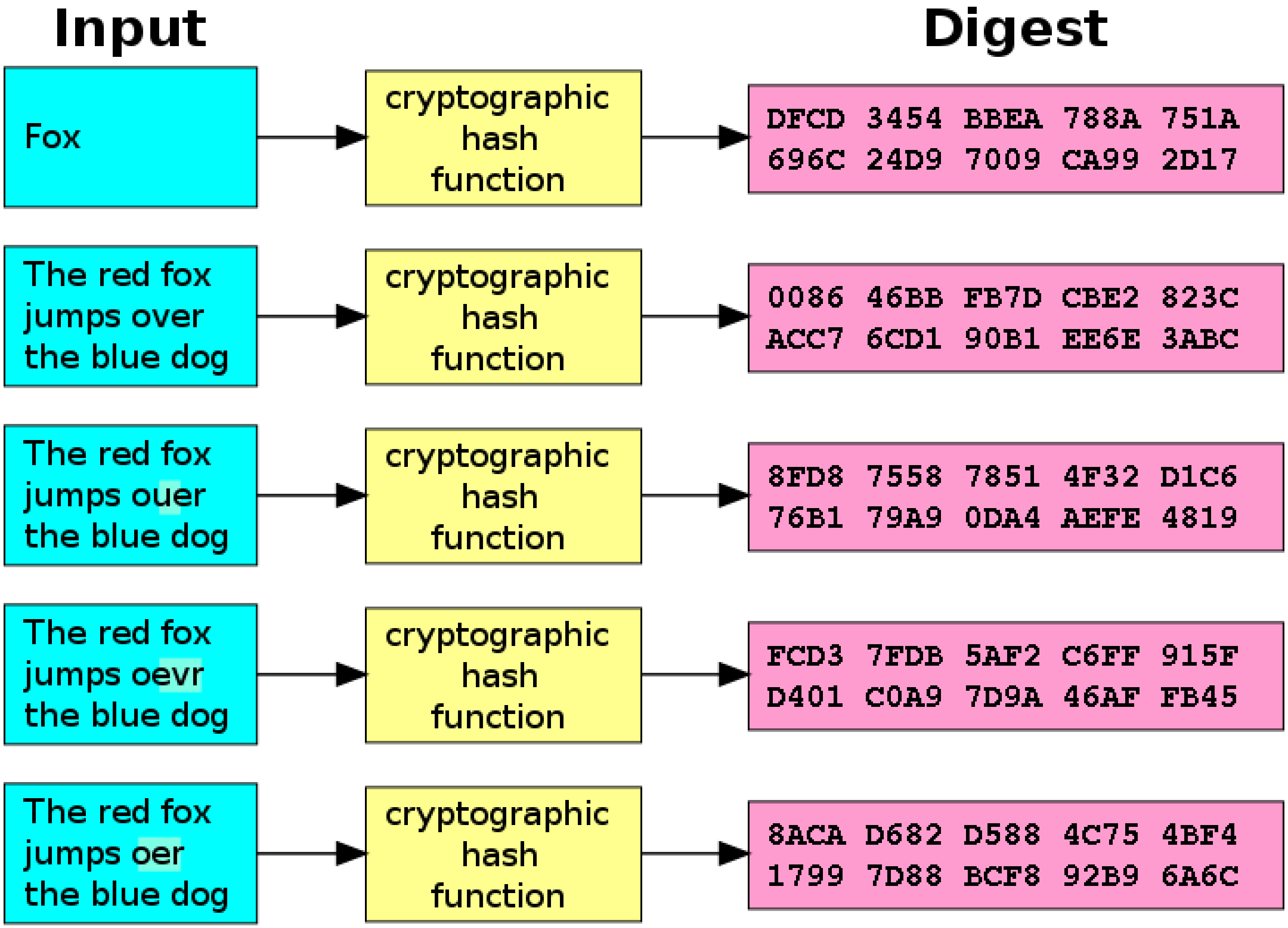
- Confidentiality
  Ensuring that only intended recipients can read a message
- Authentication / Non-repudiation
  Proving one’s identity and preventing a sender from denying that he/she sent the message
- Integrity
  Verifying that a message has not been damaged in transmission

Hash functions

- A one-way function that takes an arbitrary amount of data and produces a fixed-length output, called a hash/digest



- A 16-byte hash has 128 bits, so there are $2^{128} \approx 3.4 \times 10^{38}$ possible hashes



Input

Digest

Fox

cryptographic hash function

DFCD 3454 BBEA 788A 751A
696C 24D9 7009 CA99 2D17

The red fox jumps over the blue dog

cryptographic hash function

0086 46BB FB7D CBE2 823C
ACC7 6CD1 90B1 EE6E 3ABC

The red fox jumps ouer the blue dog

cryptographic hash function

8FD8 7558 7851 4F32 D1C6
76B1 79A9 0DA4 AEFE 4819

The red fox jumps oevr the blue dog

cryptographic hash function

FCD3 7FDB 5AF2 C6FF 915F
D401 C0A9 7D9A 46AF FB45

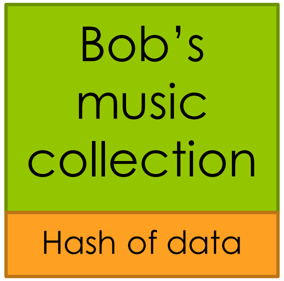
The red fox jumps oer the blue dog

cryptographic hash function

8ACA D682 D588 4C75 4BF4
1799 7D88 BCF8 92B9 6A6C

Using a hash function

- Bob wants to send data to Suzy, and he wants to make sure that she gets the correct data



Bob's music collection

Hash of data

  - So, he first generates a hash of the data and sends both the data and hash to Suzy
  - Upon receipt of the data, Suzy hashes the data and checks if the hash she generates matches the hash Bob sends
- If it matches, the data is intact. Otherwise, Suzy knows that the data has been damaged in transit and must ask Bob to send it again

Summary: Why do I need a hash function?

- ✗ Confidentiality

  Ensuring that only intended recipients can read a message

- ✗ Authentication / Non-repudiation

  Proving one's identity and preventing a sender from denying that he/she sent the message

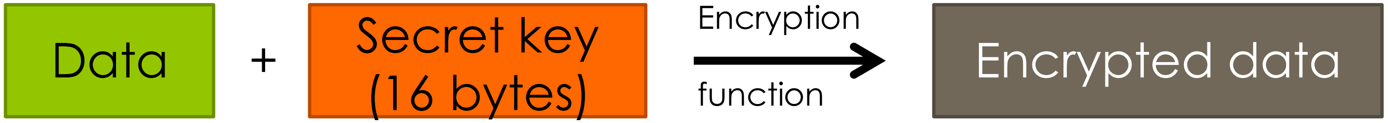
- ✓ Integrity

  Verifying that a message has not been damaged in transmission

Symmetric encryption

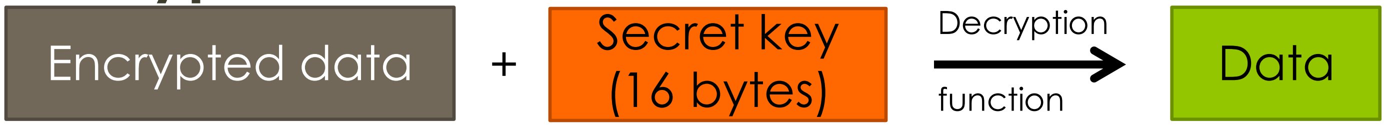
- The same secret key is used for both encryption and decryption
- Key sized is fixed – common sizes are 16 bytes (128 bits) and 32 bytes (256 bits)

Encryption



Data + Secret key (16 bytes) → Encryption function → Encrypted data

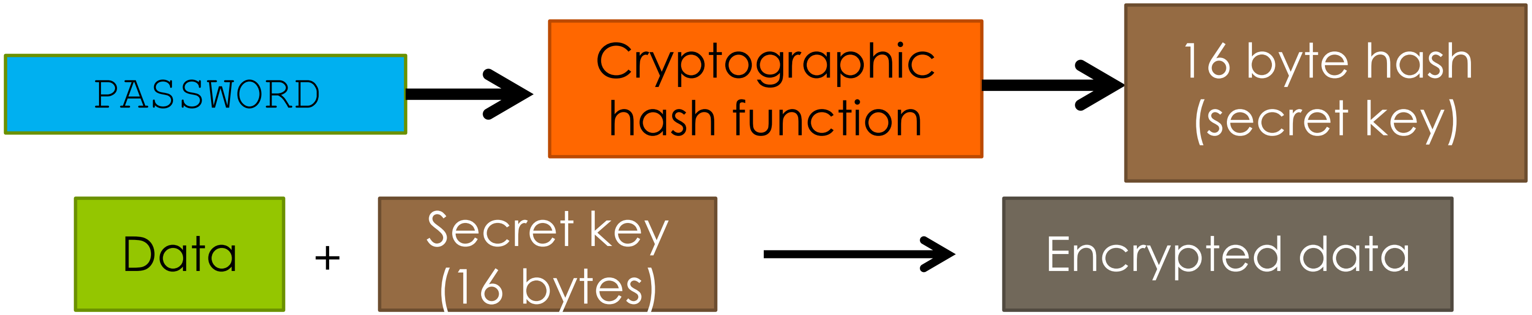
Decryption



Encrypted data + Secret key (16 bytes) → Decryption function → Data

Symmetric encryption with passwords

- How do we convert a password into a fixed-length key?



PASSWORD → Cryptographic hash function → 16 byte hash (secret key)

Data + Secret key (16 bytes) → Encrypted data

- Slower hash functions are more secure because they make brute-force attacks hard
- The password is usually hashed multiple times to make it slower

Summary: Why do I need a symmetric encryption?

- ✓ Confidentiality

  Ensuring that only intended recipients can read a message
- ✗ Authentication / Non-repudiation

  Proving one's identity and preventing a sender from denying that he/she sent the message
- ✗ Integrity

  Verifying that a message has not been damaged in transmission

Postal problem

- Alice needs to send a secret message to Bob through the mail
- Alice has never met Bob
- Alice has a lockable iron box
- Bob has a padlock and key
- The postal service will read her message unless it is locked inside the iron box
- How can Alice and Bob accomplish their goal?

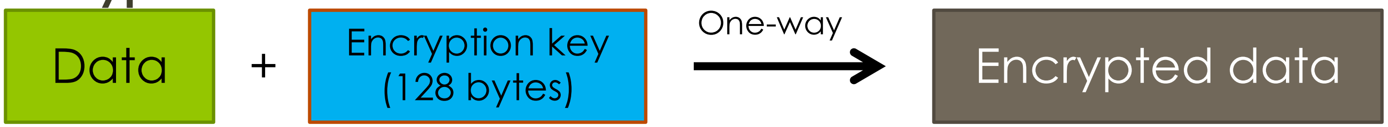
Postal problem solution

- Bob sends Alice his lock but keeps the key
- Alice places her message in the iron box and locks it with Bob’s lock
- Bob receives the box and unlocks it

Asymmetric encryption

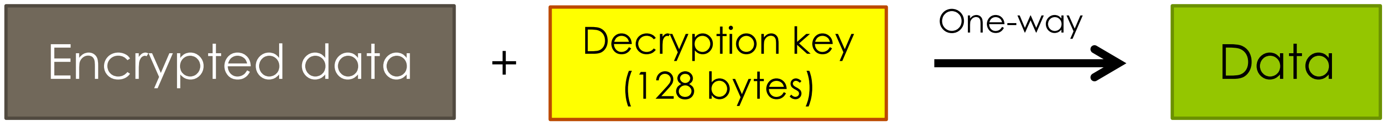
- Different keys are used for encryption and decryption.
- The keys are mathematically related, but it is unfeasible to derive one key from the other
- Common key sizes are 128 bytes (1024 bits), 256 bytes (2048 bits) and 512 bytes (4096 bits)

Encryption



Data + Encryption key (128 bytes) One-way → Encrypted data

Decryption



Encrypted data + Decryption key (128 bytes) One-way → Data

- The encryption key is like Bob’s padlock, and the decryption key is like Bob’s key
- Bob sends Alice his encryption key. Alice encrypts her message with it and sends it back to Bob. Bob decrypts the message with his decryption key.

Asymmetric encryption: practical considerations

Cipher	Speed (on 1.8 GHz Core 2 Duo)	Time to process 1GB file	Cipher name	Number of operations to crack
Symmetric encryption	100 MB/s	10 sec	128-bit AES	2^{128}
Symmetric decryption	100 MB/s	10 sec	128-bit AES	
Asymmetric encryption	1 MB/s	16.67 min	3072-bit RSA	2^{128}
Asymmetric decryption	0.02 MB/s (20 KB/s)	13.89 hrs	3072-bit RSA	

- It is unfeasible to encrypt large amounts of data with asymmetric encryption
- Usually, asymmetric encryption is only used to encrypt a key for symmetric encryption
- Hackers can exploit this to bring down websites
- Notice that key size does not indicate security level

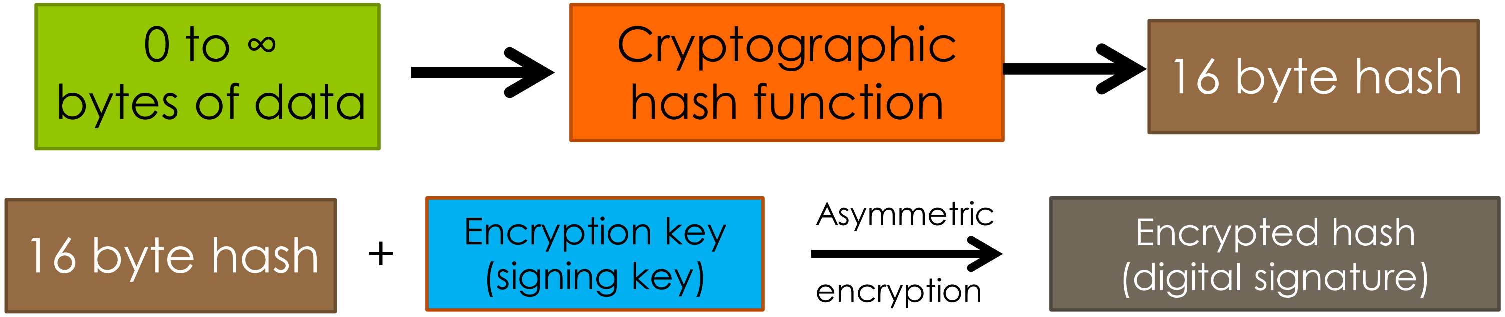
Digital signatures

- A digital signature proves the authenticity of a message
- There are 2 keys in a digital signature scheme, a signing key and verification key
- Only the signing key can be used to sign messages, and only the verification key can be used to verify messages
- The signing key and verification key are mathematically related, but it is unfeasible to derive one from the other

Digital signatures

- Hash functions + asymmetric encryption = digital signature

Signing



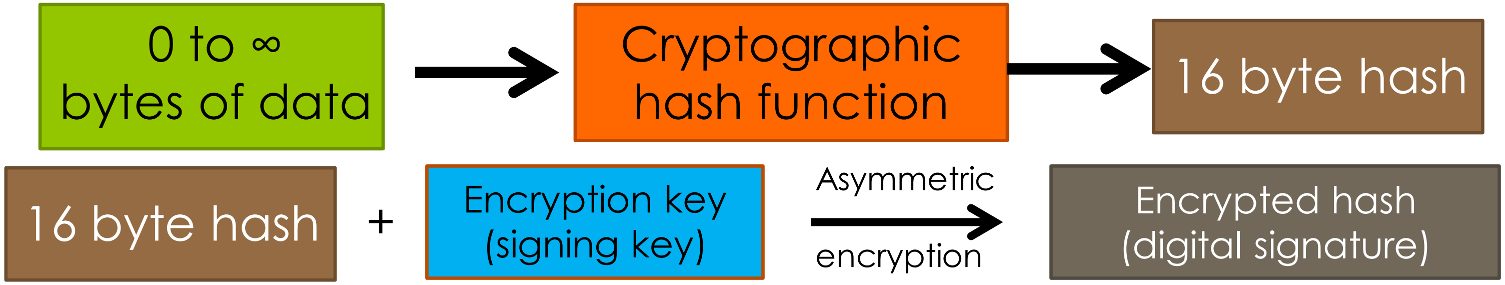
0 to ∞ bytes of data → Cryptographic hash function → 16 byte hash

16 byte hash + Encryption key (signing key) Asymmetric → encryption Encrypted hash (digital signature)

- Only a person who has the correct encryption key will be able to produce the encrypted hash
- However anyone with the decryption key will be able to decrypt the encrypted hash
- By successfully decrypting the hash, this proves the identity of the signer

Digital signatures

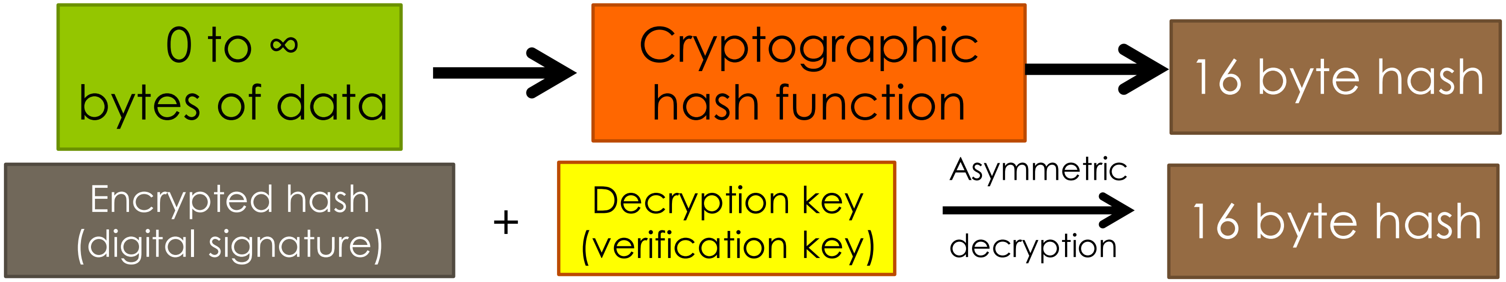
Signing



0 to ∞ bytes of data → Cryptographic hash function → 16 byte hash

16 byte hash + Encryption key (signing key) → Asymmetric encryption → Encrypted hash (digital signature)

Verification



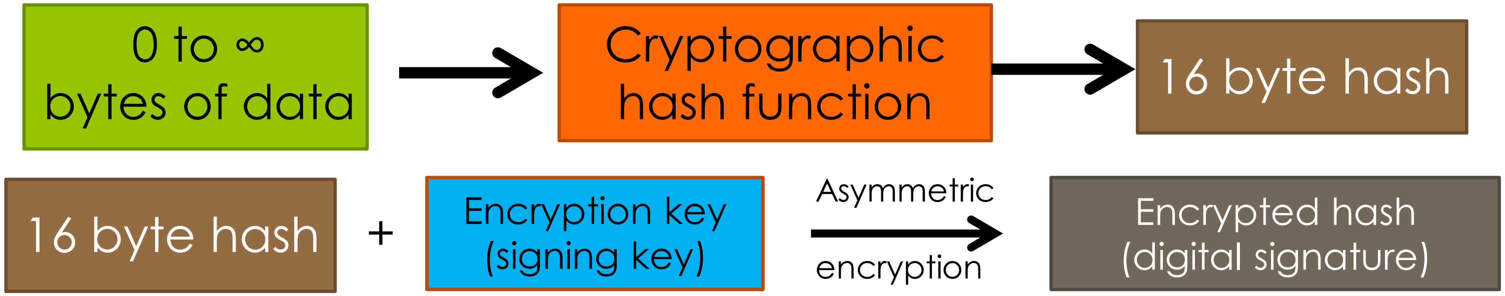
0 to ∞ bytes of data → Cryptographic hash function → 16 byte hash

Encrypted hash (digital signature) + Decryption key (verification key) → Asymmetric decryption → 16 byte hash

If the hashes match, verification is successful

Digital signatures FAQ

Signing



0 to ∞ bytes of data → Cryptographic hash function → 16 byte hash

16 byte hash + Encryption key (signing key) → Asymmetric encryption → Encrypted hash (digital signature)

- Q. Why not just encrypt the whole data instead of the hash?
- A. 2 reasons:
  - It is too slow for practical use (1GB takes 16hrs)
  - Hashing ensures integrity while encryption alone does not

Summary: Why do I need a digital signature?

- ✗ Confidentiality
  Ensuring that only intended recipients can read a message
- ✓ Authentication / Non-repudiation
  Proving one's identity and preventing a sender from denying that he/she sent the message
- ✓ Integrity
  Verifying that a message has not been damaged in transmission

Future of cryptography

- New CPUs, such as the Intel Core i7, have hardware AES encryption/decryption, allowing speeds of over 1 GB/s
- A quantum computer, if one could ever be built, would permanently break most asymmetric encryption and digital signature algorithms