SEOULTECH

SeoulTech UCS Lab
Ubiquitous Computing & Security Laboratory

PUBLIC KEY CRYPTOGRAPHY AND MESSAGE AUTHENTICATION

2016/03/14

Erik ML
(erik.miranda.lopez@gmail.com)

Public Key Cryptography and Message Authentication

In this lecture (Chapter 3 - page 61 to 96):

- **3.1 Approaches to Message Authentication**

a) Authentication Using Conventional Encryption

b) Message Authentication without Message Encryption

- **3.2 Secure Hash Functions**

a) Hash Function Requirements

b) Security of Hash Functions

c) Simple Hash Functions

d) The SHA Secure Hash Function

- **3.3 Message Authentication Codes**

a) HMAC

b) MACs Based on Block Ciphers

Public Key Cryptography and Message Authentication

(Continued)

- **3.4 Public-Key Cryptography Principles**

a) Public-Key Encryption Structure

b) Applications for Public-Key Cryptosystems

c) Requirements for Public-Key Cryptography

- **3.5 Public-Key Cryptography Algorithms**

a) The RSA Public-Key Encryption Algorithm

b) Diffie-Hellman Key Exchange

c) Other Public-Key Cryptography Algorithms

- **3.6 Digital Signatures**

**Image credit: 123RF*

Public Key Cryptography and Message Authentication

- **3.1 Approaches to Message Authentication**

a) Authentication Using Conventional Encryption

b) Message Authentication without Message Encryption

Image Credit: 123RF

Public Key Cryptography and Message Authentication

3.1 Approaches to Message Authentication (continued)

Message authentication is a procedure that allows communicating parties to verify that received messages are **authentic***.

*Authentic: when data is **genuine** (not altered – Data Integrity) and the receiving party can verify the **source** of the **message**

Aims of Message Authentication:

- Integrity: Verify that the contents of the message have not been altered
- Validating identity of the source
- Non-repudiation: Verify timeliness and message sequence flow

Protects against active attacks (ie falsification of data and transactions)

Public Key Cryptography and Message Authentication

3.1 Approaches to Message Authentication (continued)

a)Authentication Using Conventional Encryption

- Symmetric encryption*
  - Same key to encrypt & decrypt
  - Receiver knows sender must have created it
  - Weak against**
    - block reordering
    - Unauthorised modification
  - Therefore, **not suitable** for data authentication
    - Doesn't provide Integrity
    - Doesn't provide Authentication

*Symmetric encryption covered on Chapter 2 – Page 27

**Encryption alone without other mechanisms.

Public Key Cryptography and Message Authentication

3.1 Approaches to Message Authentication (continued)

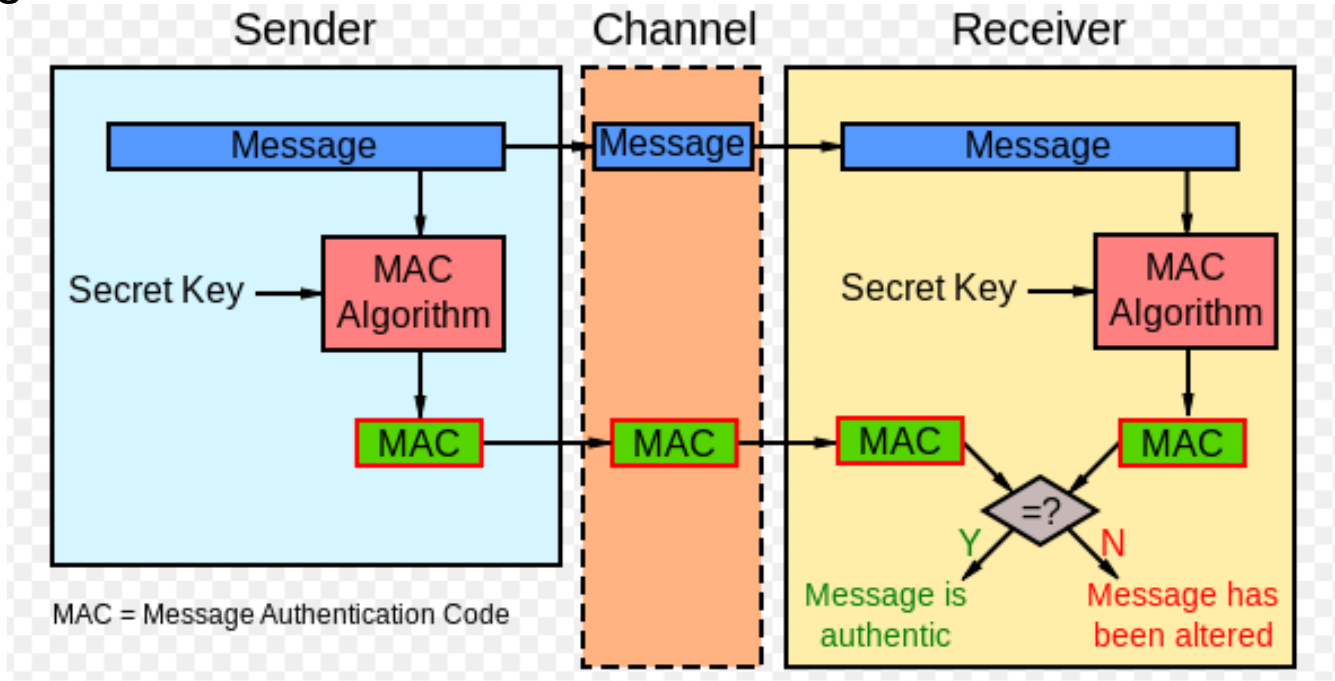
b) Message Authentication without Message Encryption

- Non-confidentiality: The message is in plain-text
- Reasons to not encrypt the message:
  - Reduce load
  - Quicker
  - Cheaper
- Several approaches:
  - Message Authentication Code
  - One-Way Hash Function

Public Key Cryptography and Message Authentication

3.1 Approaches to Message Authentication (continued)

- Message Authentication Code - MAC



– Drawbacks

- Weak against brute-force – at least 128bit MAC is needed
- is a many-to-one function – potentially many messages have same MAC
- Replay attacks – unless it has time stamp, one-time MAC or sequence number

•*Image source: Public domain

Public Key Cryptography and Message Authentication

- **3.1 Approaches to Message Authentication (continued)**

- **One-Way Hash Function**
  - It takes a variable length input M and converts it into a fixed-length sequence h
    - h=H(M)
  - The process is “easy” to compute but “hard” to reverse – hence one-way*
  - Sender uses a hash function to produce a message digest
  - Sender attaches the digest to the message and forward it to the receiver
  - Receiver does same calculation and compares it with the attached digest.
    - Usually the hash function is public and not keyed (unlike MAC)
  - Used for
    - Message Authentication
    - Signatures (More on this later on)

**explained in the next chapter*

Public Key Cryptography and Message Authentication

- **3.2 Secure Hash Functions**

a) Hash Function Requirements
b) Security of Hash Functions
c) Simple Hash Functions
d) The SHA Secure Hash Function

Image Credit: Foto Search

Public Key Cryptography and Message Authentication

3.2 Secure Hash Functions (continued)

a) Hash Function Requirements

•Purpose:

– To produce a message digest (aka hash value or fingerprint) of a file/message/data

•Requirements of hash function - H:

1. can be applied to any sized message M
2. produces a fixed-length output – message digest h
3. is easy to compute h=H(M) for any message
4. Preimage resistant / One way: Knowing h is infeasible to find out x - H(x)=h
5. Second preimage resistant / Weak collision resistant: given x is infeasible to find y H(y)=H(x)
6. (Strong) collision resistance: is infeasible to find any x,y

Public Key Cryptography and Message Authentication

3.2 Secure Hash Functions (continued)

- Collision

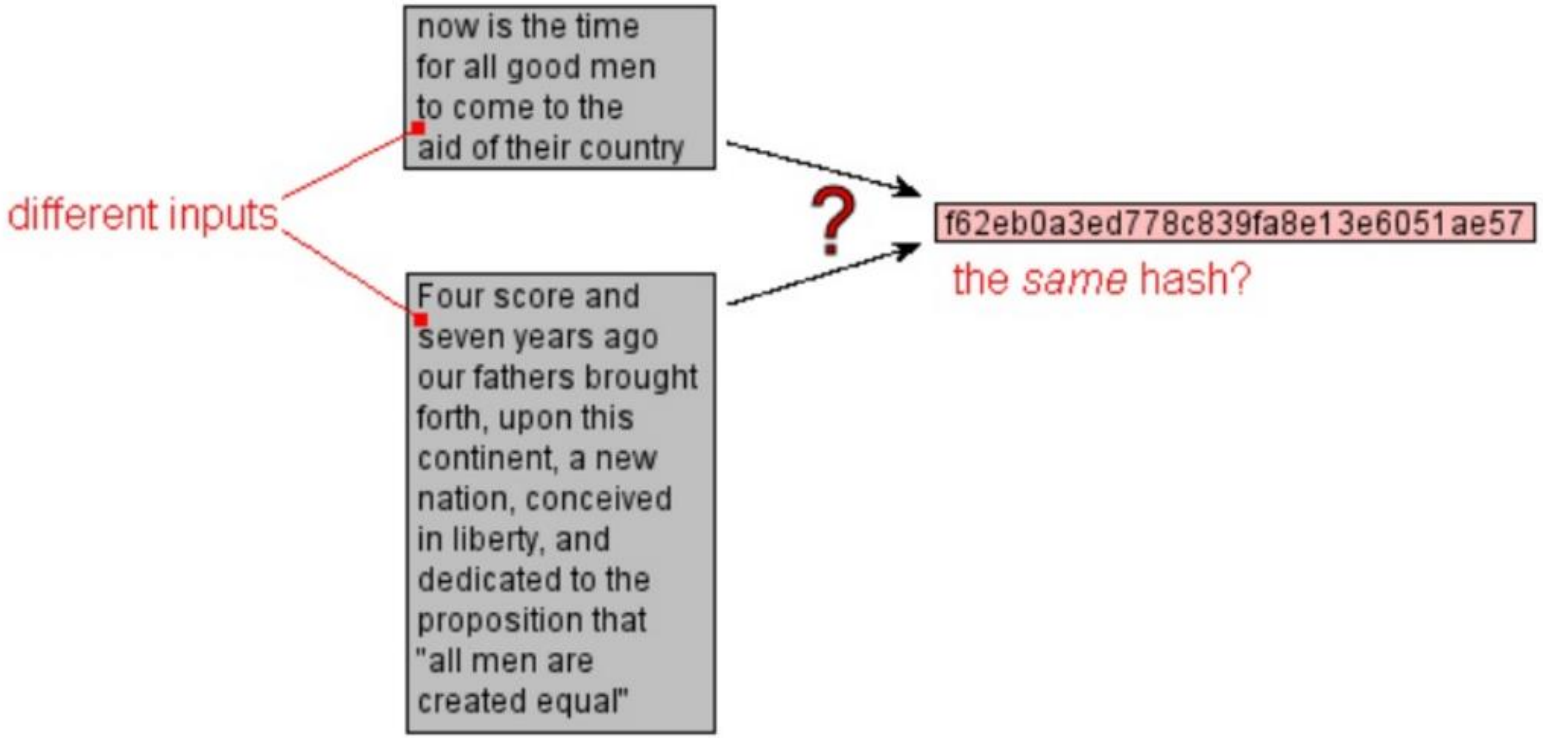


**Image credit: Sikder University of Science and Technology (Bangladesh)*

Public Key Cryptography and Message Authentication

3.2 Secure Hash Functions (continued)

b) Security of Hash Function

•Attacks:

- Brute-force
  - The level of effort depends on:

    -Length of the hash code produced.

- Cryptanalysis
  - The level of effort depends on:

    -The logical weakness in the algorithm

Public Key Cryptography and Message Authentication

3.2 Secure Hash Functions (continued)

c) Simple Hash Function

–Principles:

- The input is viewed as a sequence of n-bit blocks
- The input is processed one block at a time to produce a n-bit hash function

- Bit-by-bit exclusive OR (XOR)
  - Bitwise operator
  - The message is encrypted with a randomly generated string – secret key
  - This random secret key (ideally one–time pad) should be as long as the message and produces a ciphertext

Public Key Cryptography and Message Authentication

3.2 Secure Hash Functions (continued)

- **XOR**
  - Example

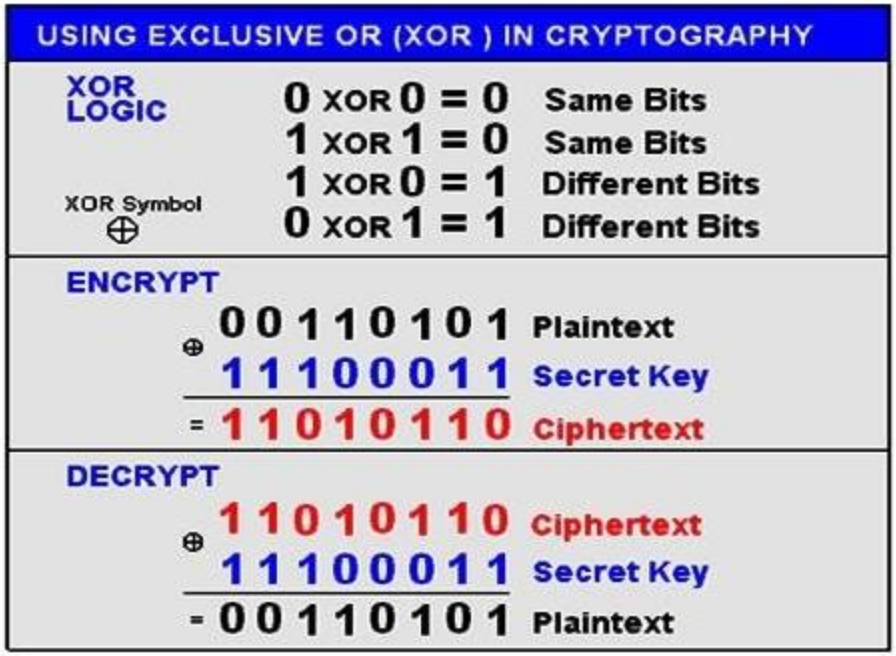


**Image Credit: wonderhowto.com*

Public Key Cryptography and Message Authentication

3.2 Secure Hash Functions (continued)

- **XOR**
  - Advantages
    - Simple to implement
    - Computationally inexpensive
  - Drawbacks
    - Impractical to have a key as long as the message
    - Not secure by itself alone (linear elements can easily be broken, given a few known plaintext - cipher text pairs, it is possible to recover the key)

Public Key Cryptography and Message Authentication

3.2 Secure Hash Functions (continued)

- **Secure Hash Function SHA**
  - Based on MD4
  - Most popular hash
  - SHA-1 - hash value 160bit (Deprecated since 2010 ie: *SSL certificates signed with SHA-1 will not be accepted by Microsoft and Google during 2016 and after)*
  - SHA-2 - hash value 256, 384 or 512bit
  - SHA-3 – hash value 256, 384 or 512bit (internal structure differs significantly from rest of SHA). Standardised 2015

Public Key Cryptography and Message Authentication

3.2 Secure Hash Functions (continued)

- **Secure Hash Function SHA**
  - Overview (SHA-2)

1. Pad message so its length is 896 and mod 1024
2. Append a 128-bit length value to message
3. Initialise 512-bit hash buffer

   Initial values:

$a = 6A09E667F3BCC908$ $e = 510E527FADE682D1$

$b = BB67AE8584CAA73B$ $f = 9B05688C2B3E6C1F$

$c = 3C6EF372FE94F82B$ $g = 1F83D9ABFB41BD6B$

$d = A54FF53A5F1D36F1$ $h = 5BE0CD19137E2179$

4. Process message in 128-word (1024-bit) chunks

   -Take initial input of hash buffer and update the content

   -Perform set of functions

   -Update hash buffer

5. Output has value is the final buffer value

Public Key Cryptography and Message Authentication

- **3.3 Message Authentication Codes**

a) HMAC

b) MACs Based on Block Ciphers

Image Credit: Care Realism

Public Key Cryptography and Message Authentication

3.3 Message Authentication Codes (continued)

- **a) HMAC**
  - Main idea is to have a MAC derived from a hash code (eg SHA-1)
    - Hash are quicker to execute compared to conventional encryption algorithm
    - Library code for hash functions widely available
- HMAC Design Objectives
  - Use free, available hash functions
  - Make it easy to replace hash functions
  - No performance degradation
  - Simple key management
  - Embedded hash function should be strong

Public Key Cryptography and Message Authentication

3.3 Message Authentication Codes (continued)

- HMAC Algorithm

$$HMAC(K, m) = H\left((K \oplus opad) \;||\; H\left((K \oplus ipad) \;||\; m\right)\right)$$

  - H hash function - any of MD5, SHA-1, RIPEMD-160 can be used
  - K is the key padded out to size
  - m is the message
  - and opad, ipad* are specified padding constants

- HMAC Security
  - The security depends on the hash used
  - Attacks:
    - Brute-force on key used
    - Birthday attack

**ipad = the byte 0x36 repeated B times*
opad = the byte 0x5C repeated B times

Public Key Cryptography and Message Authentication

3.3 Message Authentication Codes (continued)

- **b) MACs Based on Block Ciphers**
  - Block ciphers encrypt the data as a whole rather than bit by bit
  - Types:
    - Cipher-Based Message Authentication Code - CMAC
    - Counter with Cipher Block Chaining-Message Authentication Code – CCM
  - Cipher-Based Message Authentication Code – CMAC
    - *"New"* method - 2006
    - Calculates message authentication codes using a block cipher coupled with a secret key
    - Uses AES and triple DES
    - Useful when a block cipher is preferred instead of a hash function
    - Block ciphers usually slower than hash functions

Public Key Cryptography and Message Authentication

3.3 Message Authentication Codes (continued)

- Counter Mode with Cipher Block Chaining-Message Authentication Code
  - aka CCM or Counter with CBC-MAC
    - Considered an Authentication Encryption mode
    - “Authenticate-then-encrypt” concept
    - Single key is used for encryption and MAC algorithm
    - Used on Wireless LAN (IEEE 802.11i standard – WPA2)

Public Key Cryptography and Message Authentication

- **3.4 Public-Key Cryptography Principles**

a) Public-Key Encryption Structure
b) Applications for Public-Key Cryptosystems
c) Requirements for Public-Key Cryptography

Image Credit: Deposit Photos

Public Key Cryptography and Message Authentication

3.4 Public-Key Cryptography Principles (continued)

- **a) Public-Key Encryption Structure**
  - Asymmetric – Uses 2 keys
  - Components
    - Plain-text
    - Encryption algorithm
    - Private & Public Key
    - Cipher-text
    - Decryption algorithm

Public Key Cryptography and Message Authentication

3.4 Public-Key Cryptography Principles (continued)

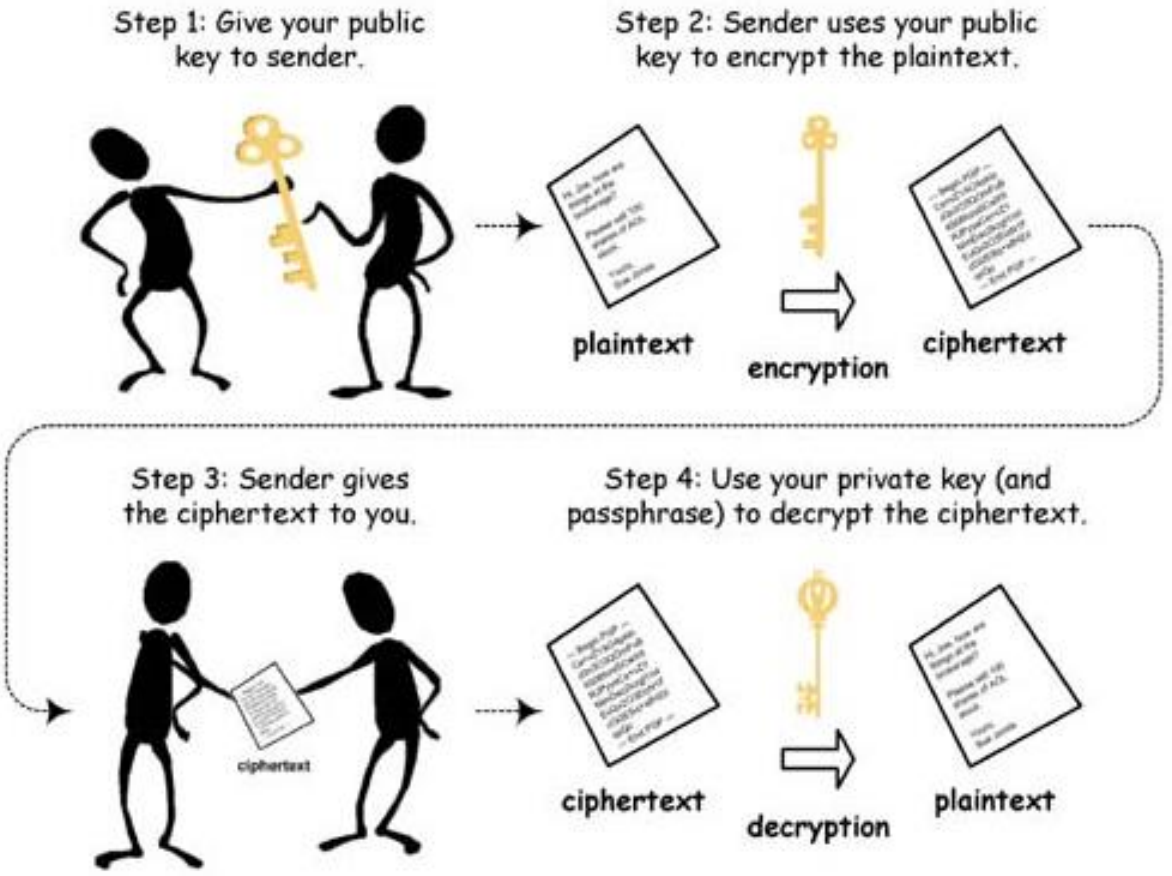


**Image credit: University of Illinois at Chicago - https://www.uic.edu/depts/accc/newsletter/adn26/figure2.html*

Public Key Cryptography and Message Authentication

3.4 Public-Key Cryptography Principles (continued)

- **b) Applications for Public-Key Cryptosystems**
  - Encryption/Decryption
  - Digital Signature (More later on – chap. 3.6)
  - Key Exchange
- **c) Requirements for Public-Key Cryptography**
  - Easy to generate pair of keys
  - Easy for the sender to encrypt message with public key
  - Easy to receiver to decrypt message with private key
  - Infeasible to determine the private key with the public key
  - Infeasible to recover the message without private key
  - Both keys can either be used to encrypt or decrypt.

Public Key Cryptography and Message Authentication

- **3.5 Public-Key Cryptography Algorithms**

a) The RSA Public-Key Encryption Algorithm
b) Diffie-Hellman Key Exchange
c) Other Public-Key Cryptography Algorithms

Image Credit: 123RF

Public Key Cryptography and Message Authentication

3.5 Public-Key Cryptography Algorithms (continued)

- **a) The RSA Public-Key Encryption Algorithm**
  - Widely used for encryption (eg SSH, TLS) and digital signatures
  - Security based on the difficulty of factoring* large integers that are product of two large prime numbers.
  - Minimum recommended key length 1024-bit
  - How to defeat it
    - Brute force
    - Factoring

**Finding what to multiply together to get an expression.*

Public Key Cryptography and Message Authentication

3.5 Public-Key Cryptography Algorithms (continued)

- Algorithm
  - Select random prime numbers *p* and *q*, and check that $p \neq q$
  - Compute modulus $n = pq$
  - Compute phi, $\Phi = (p - 1)(q - 1)$
  - Select public exponent *e*, $1 < e < \Phi$ such that $gcd(e, \Phi) = 1$
  - Compute private exponent $d = e^{-1} \bmod \Phi$
  - Public key is $\{n, e\}$, private key is *d*
- Example
  1. Select primes: $p=17$ & $q=11$
  2. Compute $n = pq = 17 \times 11 = 187$
  3. Compute $\phi(n) = (p-1)(q-1) = 16 \times 10 = 160$
  4. Select e : $\gcd(e,160)=1$; choose $e=7$
  5. Determine d: $de=1 \bmod 160$ and $d < 160$
     Value is $d=23$ since $23 \times 7 = 161 = 10 \times 160 + 1$
  6. Publish public key $KU=\{7,187\}$
  7. Keep secret private key $KR=\{23,17,11\}$

**Example credit: Yonsei University*

Public Key Cryptography and Message Authentication

3.5 Public-Key Cryptography Algorithms (continued)

- **b) Diffie-Hellman Key Exchange**
  - Use to exchange keys securely
  - Security based on difficulty of computing discrete logarithms*
  - Steps
    - 1. Alice and Bob agree on a prime number p and a base g.
    - 2. Alice chooses a secret number a, and sends Bob (g a mod p).
    - 3. Bob chooses a secret number b, and sends Alice (g b mod p).
    - 4. Alice computes ((g b mod p) a mod p).
    - 5. Bob computes ((g a mod p) b mod p).

    Both Alice and Bob can use this number as their key.

    Note that p and g need not be protected.

Public Key Cryptography and Message Authentication

- **Example**
  - Alice and Bob agree on p = 23 and g = 5
  - Alice chooses a = 6 and sends 5^6 mod 23 = 8
  - Bob chooses b = 15 and sends 5^{15} mod 23 = 19
  - Alice computes 19^6 mod 23 = 2
  - Bob computes 8^{15} mod 23 = 2

Discovering the shared secret given g, p, g a mod p and g b mod p would take longer than the lifetime of the universe, using the best known algorithm. This is called the discrete logarithm problem

Public Key Cryptography and Message Authentication

3.5 Public-Key Cryptography Algorithms (continued)

- How to beat Diffie-Hellman
  - Brute-force: Infeasible with large prime numbers
  - Man-in-the-Middle: Algorithm does NOT authenticate

Public Key Cryptography and Message Authentication

3.5 Public-Key Cryptography Algorithms (continued)

c) Other Public-Key Cryptography Algorithms

– Elliptic-Curve Cryptography - ECC

- Uses smaller keys compared to RSA with equal security
- Quicker
- Used on low computational devices

– Digital Signature Standard – DSS

- Provides only digital signature function
- No encryption
- Cannot be used for key exchange
- US government standard for authentication of electronic documents

Public Key Cryptography and Message Authentication

- **3.6 Digital Signatures**

Image Credit: Dreams Time

Public Key Cryptography and Message Authentication

3.6 Digital Signatures (continued)

- At a glance:
  - Alice creates a digital signature with her private key
  - The signature is attached to the message
  - Bob uses Alices's public key to verify the signature
  - The whole message might or not be encrypted (usually no)

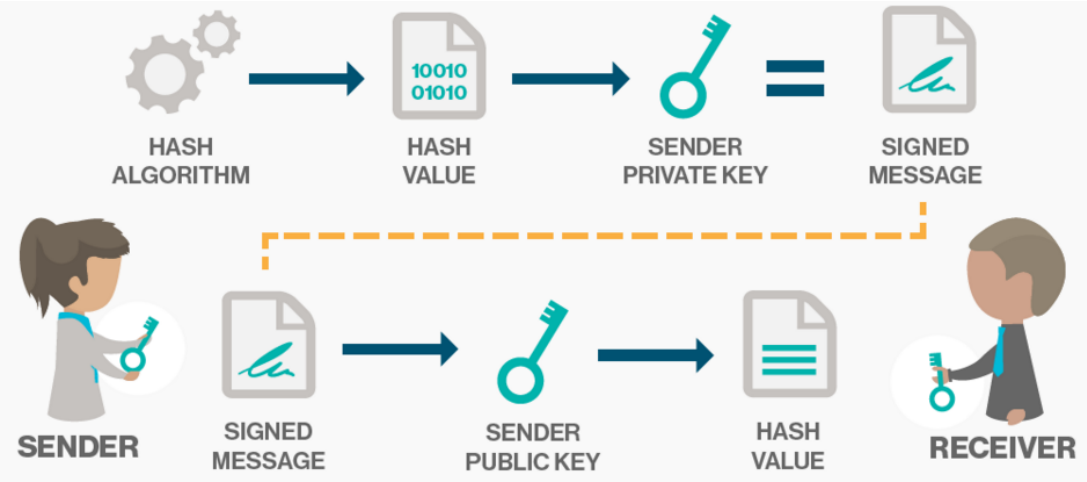


Image credit: SearchTarget.com

Public Key Cryptography and Message Authentication

3.6 Digital Signatures (continued)

- Properties:
  - Provides Integrity
  - Provides Authentication
  - Provides Non-repudiation
  - Does NOT provide confidentiality

Public Key Cryptography and Message Authentication

- **Resume**
  - **Message Authentication**
    - Symmetric encryption – not suitable for Message Authentication
    - Hash functions – XOR & SHA
    - MAC
      - HMAC
      - MAC Based on Block Ciphers
  - **Public-Key Encryption**
    - RSA
    - Diffie-Hellman
    - Elliptic-Curve Cryptography & Digital Signature Standard
  - **Digital Signatures**

Image Credit: Ris EU

Q & A

Image Credit: Public Domain