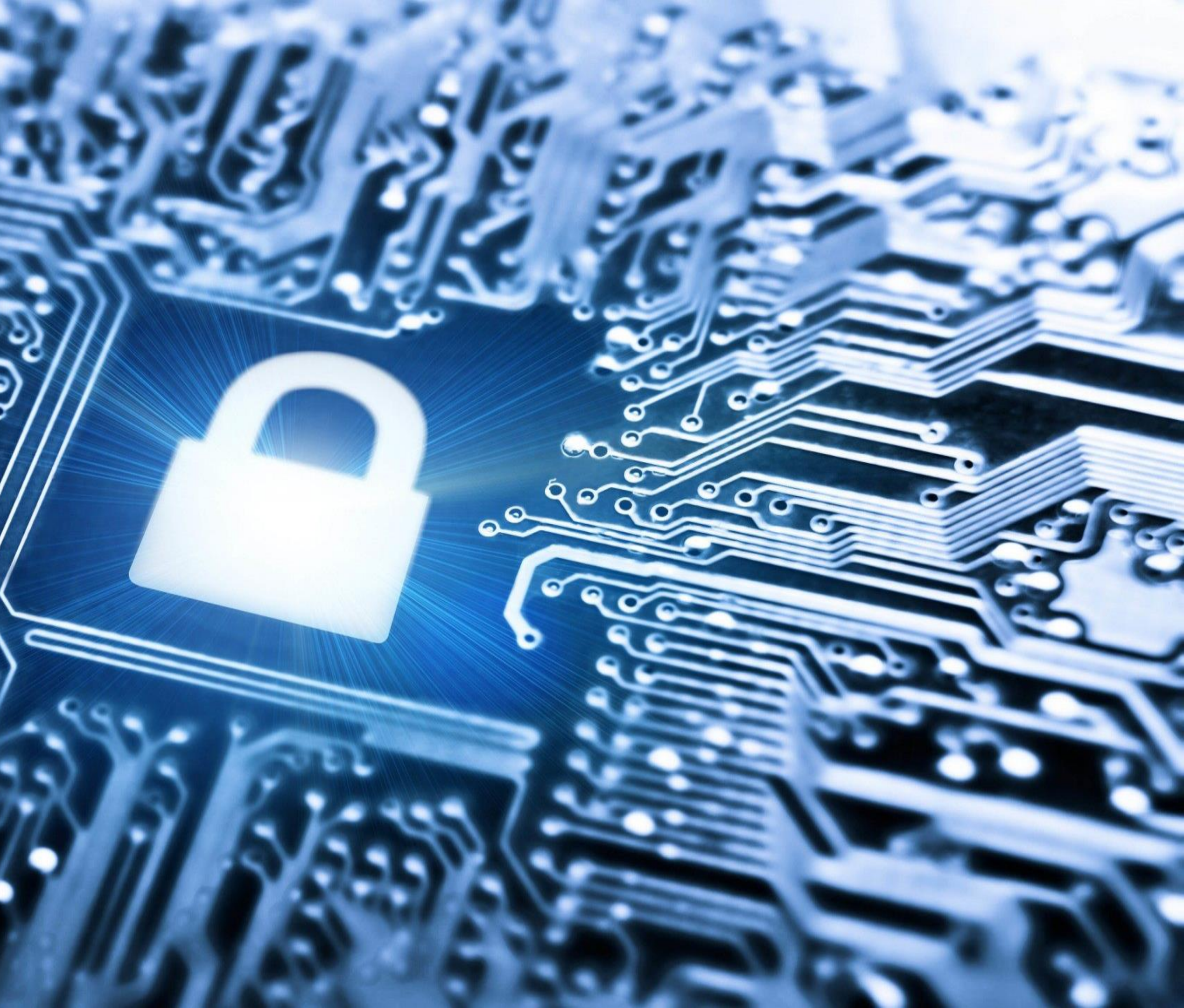
CSEN 1001

Computer and Network Security

Amr El Mougy
Reham Ayman
Abdelrahman Anwar



Lecture (5)

Public Key Cryptography

Public Key Cryptography

- Traditional **private/secret/single key** cryptography uses **one** key
- Shared by both sender and receiver
- If this key is disclosed communications are compromised
- Also is **symmetric**, parties are equal
- Hence does not protect sender from receiver forging a message & claiming is sent by sender

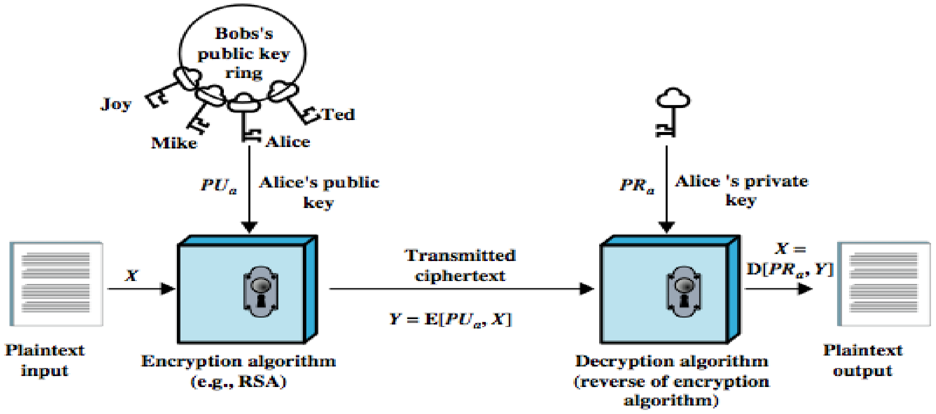
Public Key Cryptography

- Probably most significant advance in the 3000 year history of cryptography
- Uses **two** keys – a public & a private key
- **Asymmetric** since parties are **not** equal
- Uses clever application of number theoretic concepts to function
- Complements **rather than** replaces private key crypto

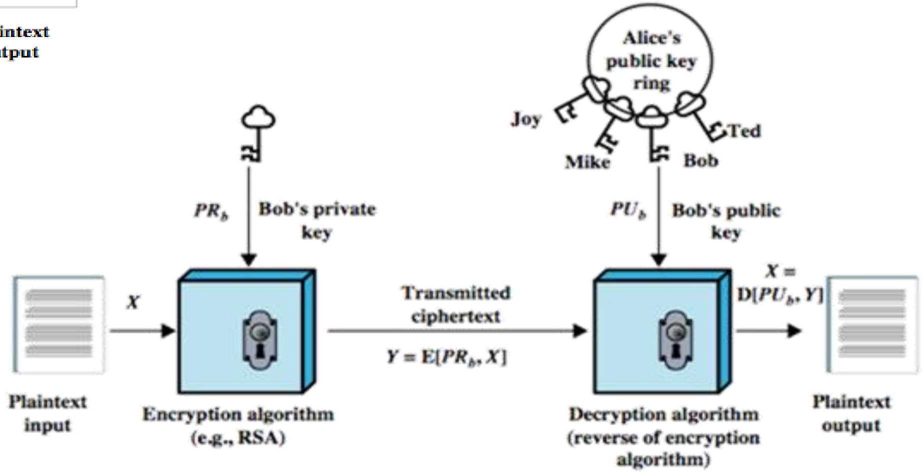
Why Public Key Cryptography

- Developed to address two key issues:
  - **Key distribution** – how to have secure communications in general without having to trust a KDC with your key
  - **Digital signatures** – how to verify a message comes intact from the claimed sender
- Public invention due to Whitfield Diffie & Martin Hellman at Stanford Uni in 1976
  - known earlier in classified community

Public Key Cryptography



(a) Confidentiality

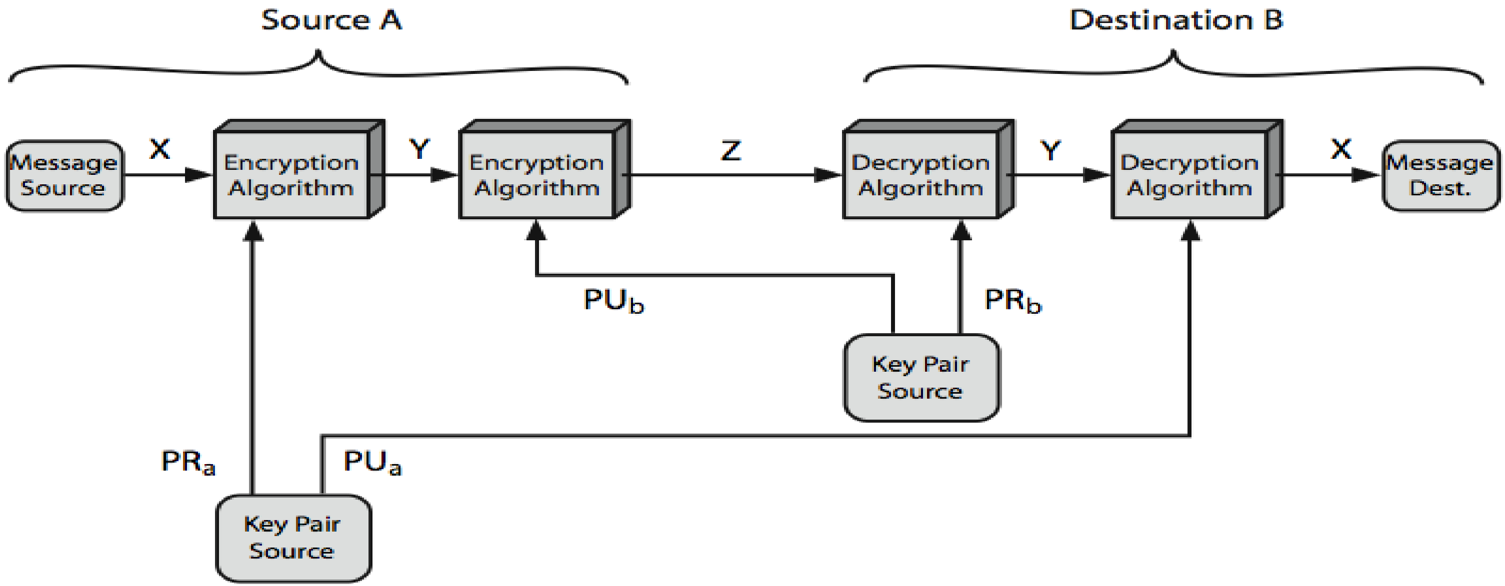


(b) Authentication

Public Key Cryptography

- **Public-key/two-key/asymmetric** cryptography involves the use of **two** keys:
  - a **public-key**, which may be known by anybody, and can be used to **encrypt messages**, and **verify signatures**
  - a **private-key**, known only to the recipient, used to **decrypt messages**, and **sign** (create) **signatures**
- Is **asymmetric** because
  - those who encrypt messages or verify signatures **cannot** decrypt messages or create signatures

Public Key Cryptosystems



Public Key Applications

- Can classify uses into 3 categories:
  - **Encryption/decryption** (provide secrecy)
  - **Digital signatures** (provide authentication)
  - **Key exchange** (of session keys)
- Some algorithms are suitable for all uses, others are specific to one

Public Key Algorithms

- RSA (Rivest, Shamir, Adleman)
  - developed in 1977
  - only widely accepted public-key encryption algorithm
  - given tech advances, need 1024 + bit keys
- Diffie-Hellman key exchange algorithm
  - only allows exchange of a secret key
- Digital Signature Standard (DSS)
  - provides only a digital signature function with SHA-1
- Elliptic curve cryptography (ECC)
  - new, security like RSA, but with much smaller keys

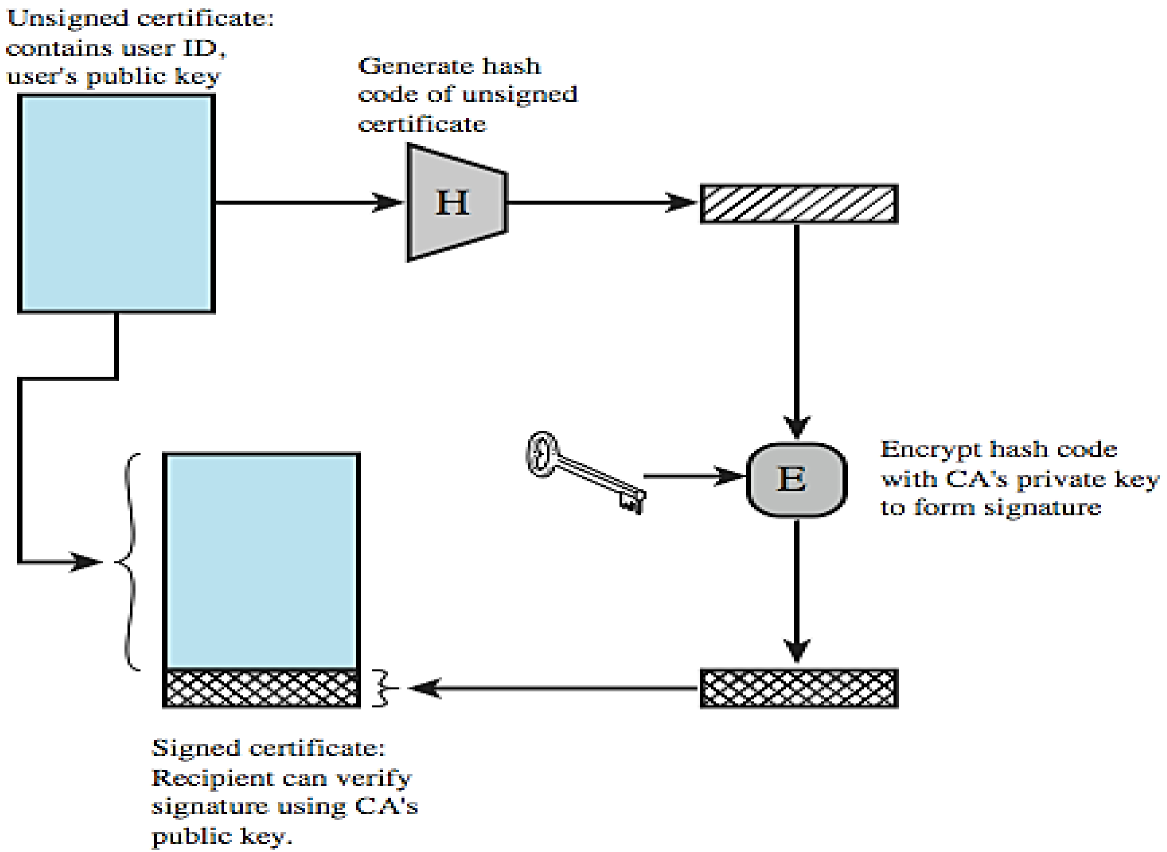
Algorithm	Encryption/Decryption	Digital Signature	Key Exchange
RSA	Yes	Yes	Yes
Elliptic Curve	Yes	Yes	Yes
Diffie-Hellman	No	No	Yes
DSS	No	Yes	No

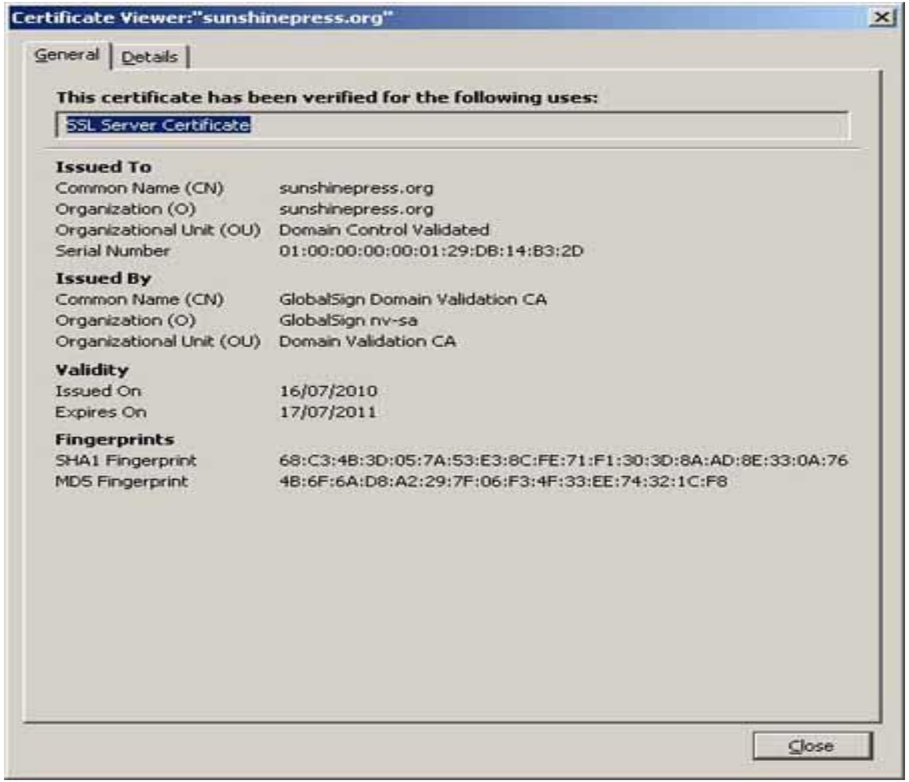
Public Key Cryptography

- Public-Key algorithms rely on two keys where:
  - It is computationally infeasible to find decryption key knowing only algorithm & encryption key
  - It is computationally easy to en/decrypt messages when the relevant (en/decrypt) key is known
  - Either of the two related keys can be used for encryption, with the other used for decryption (for some algorithms)

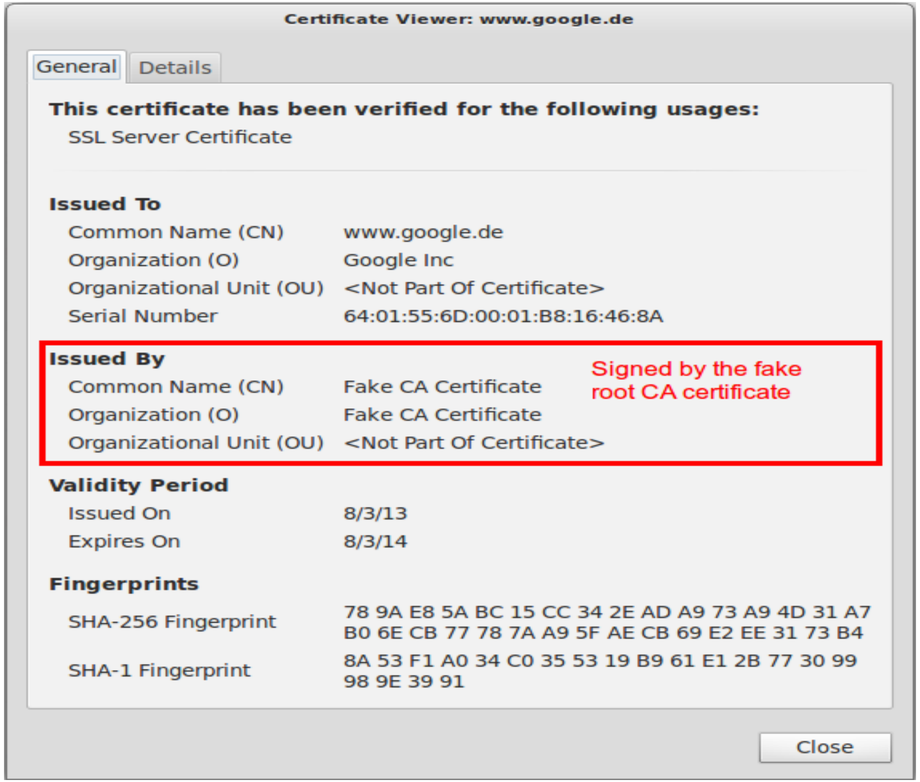
$$Y = f_k(X) \qquad easy\ if\ k\ and\ X\ are\ known$$
$$X = f_k^{-1}(Y) \qquad easy\ if\ k\ and\ Y\ are\ known$$
$$X = f_k^{-1}(Y) \qquad infeasible\ if\ Y is\ known\ but\ k\ is\ unknown$$

Public Key Certificates



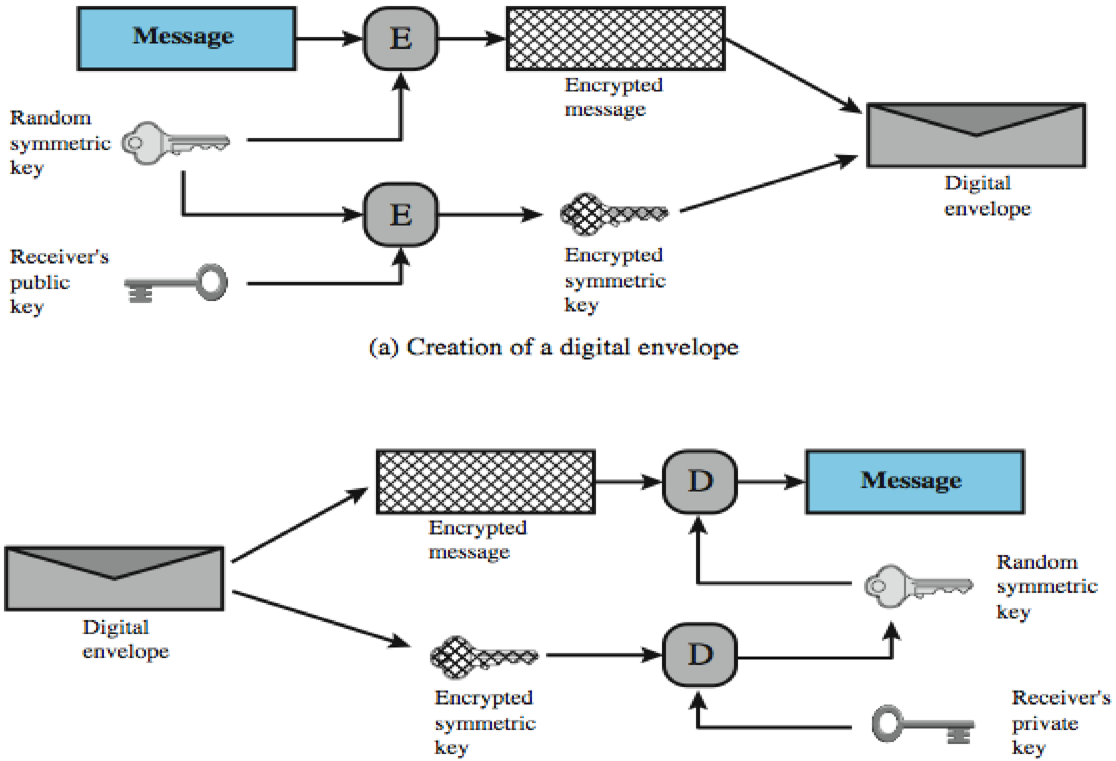


Certificate Viewer:"sunshinepress.org"
General
Details
This certificate has been verified for the following uses:
SSL Server Certificate
Issued To
Common Name (CN) sunshinepress.org
Organization (O) sunshinepress.org
Organizational Unit (OU) Domain Control Validated
Serial Number 01:00:00:00:00:01:29:DB:14:B3:2D
Issued By
Common Name (CN) GlobalSign Domain Validation CA
Organization (O) GlobalSign nv-sa
Organizational Unit (OU) Domain Validation CA
Validity
Issued On 16/07/2010
Expires On 17/07/2011
Fingerprints
SHA1 Fingerprint 68:C3:4B:3D:05:7A:53:E3:8C:FE:71:F1:30:3D:8A:AD:8E:33:0A:76
MD5 Fingerprint 4B:6F:6A:D8:A2:29:7F:06:F3:4F:33:EE:74:32:1C:F8
Close



Certificate Viewer: www.google.de
General
Details
This certificate has been verified for the following usages:
SSL Server Certificate
Issued To
Common Name (CN) www.google.de
Organization (O) Google Inc
Organizational Unit (OU) <Not Part Of Certificate>
Serial Number 64:01:55:6D:00:01:B8:16:46:8A
Issued By
Common Name (CN) Fake CA Certificate
Organization (O) Fake CA Certificate
Organizational Unit (OU) <Not Part Of Certificate>
Signed by the fake root CA certificate
Validity Period
Issued On 8/3/13
Expires On 8/3/14
Fingerprints
SHA-256 Fingerprint 78 9A E8 5A BC 15 CC 34 2E AD A9 73 A9 4D 31 A7 B0 6E CB 77 78 7A A9 5F AE CB 69 E2 EE 31 73 B4
SHA-1 Fingerprint 8A 53 F1 A0 34 C0 35 53 19 B9 61 E1 2B 77 30 99 98 9E 39 91
Close

Digital Envelopes

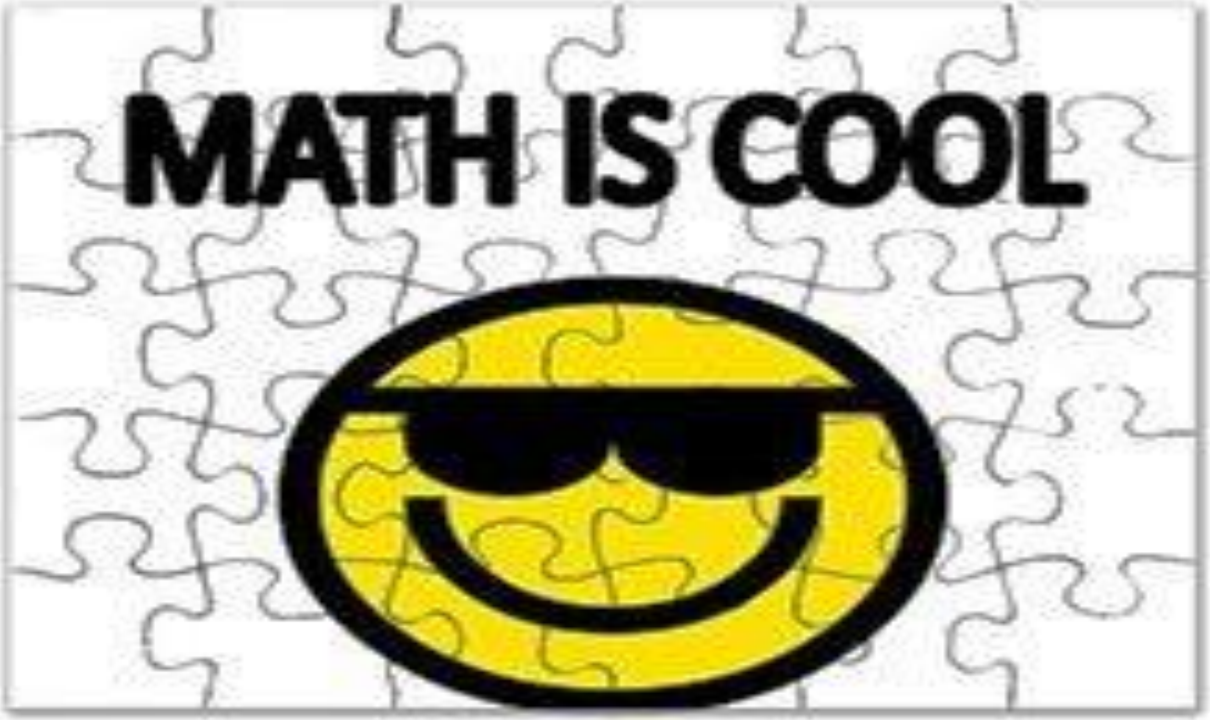


(a) Creation of a digital envelope

(b) Opening a digital envelope

Security of Public Key Cryptography

- Like private key schemes brute force **exhaustive search** attack is always theoretically possible
- But keys used are too large (>512bits)
- Security relies on a **large enough** difference in difficulty between **easy** (en/decrypt) and **hard** (cryptanalysis) problems
- More generally the **hard** problem is known, but is made hard enough to be impractical to break
- Requires the use of **very large numbers**
- Hence is **slow** compared to private key schemes


MATH IS COOL

Alice has 15 apples

She wishes to distribute them evenly over 6 friends

Remainder

RSA

- By Rivest, Shamir & Adleman of MIT in 1977
- Best known & widely used public-key scheme
- Based on exponentiation in a finite (Galois) field over integers modulo a prime
  - N.B. exponentiation takes $O((\log n)^3)$ operations (easy)
- Uses large integers (eg. 1024 bits)
- Security due to cost of factoring large numbers

RSA Key Setup

- Each user generates a public/private key pair by:
- Selecting two large primes at random: p, q
- Computing their system modulus $n=p.q$
  - note $\varnothing(n)=(p-1)(q-1)$
- Selecting at random the encryption key e
  - where $1<e<\varnothing(n)$, $\gcd(e,\varnothing(n))=1$
- Solve following equation to find decryption key d
  - $e.d \equiv 1 \bmod \varnothing(n)$
- Publish their public encryption key: $PU=\{e,n\}$
- Keep secret private decryption key: $PR=\{d,n\}$

RSA Use

- To encrypt a message M the sender:
  - obtains **public key** of recipient $PU=\{e,n\}$
  - computes: $C = M^e \bmod n$, where $0 \le M < n$
- To decrypt the ciphertext C the owner:
  - uses their private key $PR=\{d,n\}$
  - computes: $M = C^d \bmod n$
- Note that the message M must be smaller than the modulus n (block if needed)

Why RSA Works

- Because of Euler's Theorem:
  - $a^{\varnothing(n)} \bmod n = 1$ where $\gcd(a,n)=1$
- In RSA have:
  - $n=p.q$
  - $\varnothing(n)=(p-1)(q-1)$
  - carefully chose e & d to be inverses $\bmod \varnothing(n)$
  - hence $e.d=1+k.\varnothing(n)$ for some k
- Hence :

$$M = C^d \bmod n = (M^e)^d \bmod n = (M^e)^d \bmod n$$

$$C^d = M^{e.d} \bmod n = M^{1+k.\varnothing(n)} \bmod n =$$

$$M^1.(M^{\varnothing(n)})^k \bmod n$$

$$\equiv M^1.(1)^k \equiv M \bmod n$$

RSA Example – Key Setup

1. Select primes: p=17 & q=11
2. Compute $n = pq$ =17 x 11=187
3. Compute $ø(n)=(p-1)(q-1)$=16 x 10=160
4. Select e: gcd(e,160)=1; choose e=7
5. Determine d: de=1 mod 160 and d < 160
   Value is d=23 since 23x7=161= 10x160+1
6. Publish public key PU={7,187}
7. Keep secret private key PR={23,187}

RSA Example – En/Decryption

- Sample RSA encryption/decryption is:
- Given message $M = 88$ (N.B. $88<187$)
- Encryption:
  - $C = 88^7 \bmod 187 = 11$
- Decryption:
  - $M = 11^{23} \bmod 187 = 88$

Primality Testing

- Often need to find large prime numbers
- Use statistical primality tests based on properties of primes
  - for which all primes numbers satisfy property
  - but some composite numbers, called pseudo-primes, also satisfy the property
- Can use a slower deterministic primality test

```
100 decimal digits, 330 bits
RSA-100 = 15226050279225333605356183781326374297180681149613
80688657908494580122963258952897654000350692006139
```

```
617 decimal digits, 2048 bits
RSA-2048 =
2519590847565789349402718324004839857142928212620403202777713783604366202070
7595556264018525880784406918290641249515082189298559149176184502808489120072
8449926873928072877767359714183472702618963750149718246911650776133798590957
0009733045974880842840179742910064245869181719511874612151517265463228221686
9987549182422433637259085141865462043576798423387184774447920739934236584823
8242811981638150106748104516603773060562016196762561338441436038339044149526
3443219011465754445417842402092461651572335077870774981712577246796292638635
6373289912154831438167899885040445364023527381951378636564391212010397122822
120720357
```

Exponentiation

- Can use the Square and Multiply Algorithm
- A fast, efficient algorithm for exponentiation
- Concept is based on repeatedly squaring base
- And multiplying in the ones that are needed to compute the result
- Look at binary representation of exponent
- Only takes $O(\log_2 n)$ multiples for number n
  - eg. $7^5 = 7^4.7^1 = 3.7 = 10 \bmod 11$
  - eg. $3^{129} = 3^{128}.3^1 = 5.3 = 4 \bmod 11$

Efficient Encryption

- Encryption uses exponentiation to power e
- Hence if e small, this will be faster
  - often choose $e=65537$ $(2^{16}+1)$
  - also see choices of $e=3$ or $e=17$

RSA Security

- Possible approaches to attacking RSA are:
  - Brute force key search (infeasible given size of numbers)
  - Mathematical attacks (based on difficulty of computing ø(n), by factoring modulus n)
  - Timing attacks (on running of decryption)

The RSA algorithm and its proof can be found in Chapter 9 of the book titled “Cryptography and Network Security”, by William Stallings.