What we have learned so far

- Symmetric key cryptography
- Block cipher
- Stream cipher

What we have learned so far

- Symmetric key cryptography
- Block cipher
- Stream cipher
- Key distribution
- Public key cryptography
- Cryptographic hash function
- Digital Signature

What we have learned so far

- Symmetric key cryptography
- Block cipher
- Stream cipher
- Key distribution
- Public key cryptography
- Cryptographic hash function
- Digital Signature
- Virus, Worms, Trojans
- Denial of Service Attack

Number Theory for RSA

- n and m are relatively prime if their greatest common divisor (gcd) is 1

Number Theory for RSA

- n and m are relatively prime if their greatest common divisor (gcd) is 1

Theorem

If a and n are relatively prime then there exists a unique number b such that $ab \equiv 1 \mod n$.

Number Theory for RSA

- n and m are relatively prime if their greatest common divisor (gcd) is 1

Theorem

If a and n are relatively prime then there exists a unique number b such that $ab \equiv 1 \ mod \ n$.

Fermat's Little Theorem If p is prime and $1 \leq a \leq p-1$, then $a^{p-1} \equiv 1 \ mod \ p$

Euler's generalization of Fermat's Little Theorem

- The *reduced set of residues mod n* is the set of numbers in $\{1, \ldots n\}$ that are relatively prime to n
- $\phi(n)$ is the size of the reduced set of residues mod n.
- If a and n are relatively prime, then $a^{\phi(n)} \equiv 1 \; mod \; n$

Euler's generalization of Fermat's Little Theorem

- The *reduced set of residues mod n* is the set of numbers in $\{1, \dots n\}$ that are relatively prime to n
- $\phi(n)$ is the size of the reduced set of residues mod n.
- If a and n are relatively prime, then $a^{\phi(n)} \equiv 1 \; mod \; n$
- What's used in RSA: if $n = p \cdot q$ where p and q are prime, then $\phi(n) = (p-1) \cdot (q-1)$

RSA Public Keys

- Widely used on the Internet
- Standard PKS for finance applications
- Patent in US, expired 20 Sep 2000
- not proved secure, but has withstood extensive cryptanalysis

RSA Public Keys

- Choose two random large prims p and q, Define $n = p \cdot q$
- Choose e such that e and $(p-1)\cdot(q-1)$ are relatively prime
- Compute d such that

$$ed \equiv 1 \; mod \; (p-1)\cdot(q-1)$$

  then discard p and q
- Public key is $K = (e, n)$
- Private key is $K^{-1} = (d, n)$

RSA Public Keys

- To encrypt message M using public key (e, n): $M^e \ mod \ n$
- To decrypt, compute $(M^e \ mod \ n)^d \ mod \ n = M^{e \cdot d} \ mod \ n$ which is just M
- because $e \cdot d = k(p-1)(q-1)+1$

Computing Exponentiation

We can compute a^{2^k} *mod* n in k multiplications

$$a^2 \; mod \; n = a \cdot a \; mod \; n$$

$$a^4 \; mod \; n = a^2 \cdot a^2 \; mod \; n$$

$$\vdots$$

$$a^{2^k} \; mod \; n = a^{2^{k-1}} \cdot a^{2^{k-1}} \; mod \; n$$

Given arbitrary x, we have $x = x_0 + x_1 \cdot 2 + \ldots x_k \cdot 2^k$ as the binary decomposition

Security of RSA

RSA has withstood cryptanalysis, but

- It has not been proved to be secure
- Factorization is thought not to be NP-complete
- in theory factorization can be done efficiently on a quantum computer
- it is not known that factorization is the only way to crack RSA

Cracking RSA with integer factorization

- The public key is (e, n), if we can factorize $n = p \cdot q$, we are able to compute d
- $ed \equiv 1 \ mod \ (p-1)(q-1)$ can be efficiently computed

Cracking RSA with integer factorization

- The public key is (e, n), if we can factorize $n = p \cdot q$, we are able to compute d
- $ed \equiv 1 \ mod \ (p-1)(q-1)$ can be efficiently computed
- During the RSA-155 cracking in 1999, it took 290 computers on the Internet plus a supercomputer 4 months to factor a 512 bits (155 decimal digits) integer with two large prime factors
- The required computing power was estimated at 8000 Mips-years (a Mips is a million of processor instructions per second)

Cracking RSA with integer factorization

- The public key is (e, n), if we can factorize $n = p \cdot q$, we are able to compute d
- $ed \equiv 1 \; mod \; (p-1)(q-1)$ can be efficiently computed
- During the RSA-155 cracking in 1999, it took 290 computers on the Internet plus a supercomputer 4 months to factor a 512 bits (155 decimal digits) integer with two large prime factors
- The required computing power was estimated at 8000 Mips-years (a Mips is a million of processor instructions per second)
- Early RSA key length was 1024, which has now been extended to 2048, 3072 and 4096 bits in current practice.

Public Key Infrastructure

- Do you "trust" other people's public key?

Active Attackers

An active attacker Mallory (M), typically can

- listen in on transmissions
- block messages from reaching the intended recipient
- modify plaintext parts of messages
- send fake messages

Mallory's attack on public keys

1. $A \rightarrow B$: Hey Bob, I've got a really juicy secret to tell you, what's your public key?
2. $B \rightarrow A(M)$: Hi Alice, its K_B, Regards, Bob
3. $M \rightarrow A$: Hi Alice, its K_M, Regards, Bob
4. $A \rightarrow B(M)$: $\{Secret\}_{K_M}$
5. Mallory decrypts the message, and get *Secret*
6. $M \rightarrow B$: $\{Secret\}_{K_B}$

Diffie/Hellman's solution: A secure online directory D serving Public key requests

1. Each user trusts D
2. Each user has a shared key with D
3. n users can establish n^2 secure channels

Kohnfelder's solution: Signed certificates for offline name-key binding validation

Real World Certificates

A document containing a certified statement, especially as to the truth of something

- Birth certificates
- Marriage certificates
- Degree certificates
- Doctors certificates

Public Key Certificates

The contained information

1. **Subject:** name of person/entity holding the key
2. **Public Key:** key value, (e, N) in the case of RSA
3. **Certificate Authority Name:** a name N, e.g., verisign/Symantec, Comodo, pingan
4. Signed using N's private key

Certificate Distribution Methods

Certificate gets integrity and verifiability from the signature, so does not need secure storage/transmission. Can be distributed

1. Along with the signed document
2. As part of a protocol (e.g., SSL/TLS)
3. using directory Services (e.g., X.500, LDAP)
4. on web-pages
5. person-to-person, or by email

Public Key Certificate Standards

Aspects to be standardized

1. Certificate Syntax
2. Certificate Semantics
3. Rules for Operation of certificate infrastructure
4. Legal Issues, Liability

X.509 certificate structure

1. Certificate Version
2. Certificate Serial Number
3. CA's signature algorithm ID
4. CA's X.500 name
5. Validity period
6. Subjects Public Key information (e.g., Algorithm Identifier, Public key value)
7. (Optional) Issuer Unique identifier, subject unique identifier, Extension fields

Single Certificate Authority

Fine within a single, centralized company

1. Who can everybody trust?
2. Who is qualified of verifying everyone's identity?
3. How to do identity verification over a distance?
4. Monopoly (pay to get a certificate)
5. Single point of vulnerability for all applications

Sub Certification Authorities

- A root CA, delegating its rights to sign certificates to others (which may delegate further, to some depth)
- A (delegation) certificate says "Key x belongs to RA, and RA is authorized by me as a sub-certification authority"
- E.g., Central government $\rightarrow$ Provencial government $\rightarrow$ local/city/district government
- X.509 uses BasicConstraints extension

Certificate Chain Logic

If

1. I believe K_0 belongs to CA, and I trust CA as a certification authority
2. "K_1 belongs to RA1, which is my subCA", signed K_0
3. "K_2 belongs to RA2, which is my subCA", signed K_1
4. "K_3 belongs to Alice", signed K_2

Therefore, I believe K_3 belongs to Alice.

Certificate Chain with Constraints

Name constraints: what names an RA is allowed to certify

E.g., JNU may certify names of the form

1. { Country = China, Organization = JNU }
2. $*$@$*.jnu.edu.cn$

Multiple CA's

Since in practice, there is nobody that everybody trusts, we have multiple organizations setting themselves up as CA's

The compete/pay browser/OS vendors to include their keys as trusted keys in their software

Some of the default Signers in Chrome 40 and IE 10

Trusted Root CA Certificate	Expiration
AAA Certificate Services	12/31/28
ABA.ECOM Root CA	07/09/09
AC Raíz Certicámara S.A.	04/02/30
AC1 RAIZ MTIN	11/03/19
ACCVRAIZ1	12/31/30
ACEDICOM Root	04/13/28
...	...

Revocation

Keys do get compromised, smartcards lost, employees leave the company, etc.
So we need to be able to undo the effects of a certificate.
Approaches:

1. Certificate Revocation Lists (X.509)
2. Online revalidation