CSE 589 -- Part VIIII

Five or six weeks later, she asked me if I had deciphered the manuscript… I told her that I had.

"Without the key, sir, excuse me if I believe the thing impossible."

"Do you wish me to name your key, madame?"

"If you please."

I then told her the key-word which belonged to no language, and I saw her surprise. She told me it was impossible, for she believed herself the only possessor of that word which she kept in her memory and which she had never written down.

I should have told her the truth -- that the same calculation which had served me for deciphering the manuscript had enabled me to learn the word -- but on a caprice it struck me to tell her that a genie had revealed it to me. This false disclosure fettered Madame d'Urfe to me. That day I became the master of her soul, and I abused my power.

From the autobiography
of Casanova (1757)

Cryptography

Cormen, Leiserson, Rivest, Chapter 33
Cryptography Theory and Practice,
by Douglas Stinson
Applied Cryptography
by Bruce Schneier

Outline

Overview
Perfect Secrecy
Public Key Cryptography
Mathematical Background to RSA
RSA
- Implementation Details
- Security Provided

Digital Signatures
Attacks Against RSA

Cryptography

Goal:
- to be able to communicate securely over a channel, any medium for communication between two parties.
  - Telephone, radio waves, Internet, satellite communication, etc.
- of immense commercial importance, particularly with increasing use of Internet for commercial purposes.

Security Risks of Internet Communication

Eavesdropping
- intermediaries listen in on private conversations
- Solution: encryption (public or private-key)

Manipulation
- intermediaries change information in a private communication
- Solution: methods for preserving data integrity (one-way hash fn's and MACs)

Impersonation
- a sender or receiver communicates under false identification
- Solution: authentication (signatures, etc.)

Terminology

A sender wants to send a message to a receiver securely -- wants to make sure no eavesdropper can read the message.

ciphertext

Plaintext ----> Encryption ----------> Decryption ----> Plaintext

M **E(M)** **C=E(M)** **D(C)** **M=D(C)**

plaintext -- original message

encryption -- process of disguising message so as to hide its substance

ciphertext -- encrypted message

decryption -- process of turning ciphertext back into plaintext.

cryptography -- art and science of keeping messages secure

cryptanalysis -- art and science of breaking ciphertext

Cryptography: communication in the presence of adversaries.

Goals:

- **privacy**: adversary learns nothing about message sent
- **authentication**: recipient of message can convince herself that the message as received originated with alleged sender.
- **signatures**: recipient of a message can convince a third party that the message as received originated with the alleged signer.
- minimality: nothing is communicated except that which is specifically desired to be communicated.
- simultaneous exchange: something of value not released until something else of value received.
- multi-party coordination: parties can coordinate activities towards common goal even in presence of adversaries.

The cast of characters

Alice: first participant in all protocols

Bob: second participant in all protocols

Eve: eavesdropper

Mallory: malicious active attacker

Trent: trusted arbitrator

Cryptanalysis

One possibility: security through obscurity

Restricted algorithms: depend on keeping inner workings of algorithm secret.

Difficult for communications between parties with no prior contact, as in Internet commerce applications.

Wildly optimistic to assume details of security mechanisms won't fall into the wrong hands.

No quality control or standardization

Kerckhoff's Doctrine

Associated with algorithm is key. All security is in key.

Kerckhoffs: If the strength of your cryptosystem relies on the fact that the attacker does not know the algorithm's hidden workings, you're sunk!!!

Key question: can we do it?

As Edgar Allan Poe wrote in "The Gold-Bug":

It may well be doubted whether human ingenuity can construct an enigma of the kind which human ingenuity may not, by proper application, resolve.

Key-Based Security
All security in key; alg can be published

$E_{K1}(M) = C$, $D_{K2}(C) = M$

keyspace -- range of possible key values

Symmetric algorithms: encryption key can be calculated from decryption key and vice versa (usually same)

- stream ciphers -- operate on plaintext a bit (byte) at a time
- block ciphers -- operate on group (block) of bits.

Examples: DES, RC4, IDEA, Blowfish....

Notable feature: fast

Protocol for communicating using symmetric cryptography

Alice and Bob agree on a cryptosystem

Alice and Bob agree on a key

Alice encrypts plaintext using encryption algorithm and key => ciphertext

Alice sends ciphertext to Bob

Bob decrypts ciphertext with same algorithm and key and reads it.

Problems

keys must be distributed in secret

if key compromised, all is lost

keys needed grows like Omega (n^2)

What does it mean to say an algorithm is unbreakable?

Shannon's Theory

"Communication Theory of Secrecy Systems" by Claude Shannon, 1949.

Many important ideas

Two approaches to discussing security of a cryptosystem

Computational security

cannot be broken with "available resources" current or future.

- best known method of breaking system takes unreasonably large amount of time
- can reduce the security of the cryptosystem to some well-studied problem that is thought to be difficult.

Unconditional security

cannot be broken, even with infinite computational resources.

Unconditional Security

Framework: probability theory

Probability distribution over plaintexts (known to Eve)

Probability distribution over keys.

Perfect Secrecy

Cryptosystem has perfect secrecy if
Prob (plaintext | ciphertext) = Prob(plaintext)

I.e., a posteriori probability of plaintext, given that the ciphertext is observed is identical to the a priori probability of plaintext.

Realization of Perfect Secrecy: The One-time Pad (1917)

$P = C = K$ = n bit strings
$E_K(p)$ = bitwise xor of K and p = c.
$D_K(c)$ = bitwise xor of K and c = p

Problems:

- amount of key that must be communicated securely equals amount of plaintext
- severe key management problems since can use each key for only one encryption
  (vulnerable to known plaintext attack)

Computational Security: Types of cryptanalytic attack

ciphertext only -- analyze ciphertexts to gain info aobut plaintext
known-plaintext attack -- intercept ciphertexts for which plaintexts are known
chosen-plaintext attack -- choose plaintexts to be encrypted
adaptive-chosen-plaintext attack -- modify choices based on results
chosen ciphertext attack -- get Bob to decrypt ciphertexts of choosing
rubber-hose cryptanalysis -- threaten, blackmail, torture until key is released

Public Key Cryptography

We stand today on the brink of a revolution in cryptography.

Diffie and Hellman, 1976

Public-Key Cryptography

Private key crypto => Alice and Bob must secretly choose K, exposure of E_K renders system insecure => requires prior communication of the key K using secure channel before any ciphertext is transmitted.

Public-key system:

Idea: to find a cryptosystem where computationally infeasible to determine D given E => E could be made public by publishing it in a directory.

=> Alice can send encrypted message using public E

Bob is only person who can decrypt it, using secret decryption rule.

Public-Key Cryptography

Idea due to Diffie & Hellman 1976 (indep Merkle)
First realization 1977: Rivest, Shamir, Adleman.
Since then, many others.
Security rests on different computational problems.

- RSA -- factoring large integers (??)
- McEliecee -- decoding linear code (NP-complete)
- El Gamal -- discrete logarithm problem (??)
- Chor-Rivest -- knapsack (NP-complete)
- ....

Public-key cryptosystem can never provide unconditional security.

Idea: uses trapdoor one-way function.

E_K should be easy to compute
D_K (inverting E_K) should be hard
=> E_K should be a **one-way** function
Example of possible one-way function:
n=pq (p,q two large prime numbers)
$f(x) = x^b \bmod n$
Don't want E_K to be one-way from Bob's point of view => Bob must possess trapdoor, secret information that permits easy inversion of E_K

One-Way Functions

A function f is one-way if, given x it is easy to compute f(x), but given f(x) it is hard to compute x (superpolynomial, or exponential time).

Trapdoor one-way functions: easy to compute f(x) given x, hard to compute x given f(x). But there is some secret information y, s.t. given f(x) and y, easy to compute x.

Number Theoretic Preliminaries

… both Gauss and lesser mathematicians may be justified in rejoicing that there is one science [number theory] at any rate… whose very remoteness from ordinary human activities should keep it gentle and clean.

From the autobiography
A Mathematician's Apology
of G.H. Hardy,
number theorist and pacifist,
1940

Number-theoretic Preliminaries

Modular arithmetic
- Digression about cryptanalytic attacks

Prime numbers
Greatest common divisor
Inverses modulo a number
Fermat's Little Theorem
Euler Totient Function
Chinese Remainder Theorem
- Factoring
- Prime Number Generation

Modular Arithmetic

"clock arithmetic"
If Mildred says she'll be home by 10:00 and she's 13 hours late, what time does she get home and for how many years does her father ground her? => arithmetic modulo 12.
(10 + 13) mod 12 = 23 mod 12 = 11 mod 12
23 = 11 mod 12
a = b (mod n) if a = b + k n for some integer k
<=> a is congruent to b, modulo n
<=> b is the residue of a, modulo n
<=> a non-negative, 0 <= b<= n => b is remainder of a when divided by n.

Digression to use modular arithmetic; cryptanalysis

Caesar cipher:
plaintext a b c d e ….
+ K =
ciphertext d e f g h

K is offset between 0 and 26 => $E_K(P) = P+K \bmod 26$

Easy attack: try all possible keys.

Moral: cryptosystem insecure if # keys too small.

Possible solution: use more keys => Addition Cipher

Imagine alien alphabet with 10^{12} keys.

Break message up into blocks of numbers between 0 and $10^{12} - 1$

Known-plaintext attack

Alice sends Bob message using block size of 20 digits.

Eve intercepts, and she knows message starts with "Dear Bob" => knows both plaintext and ciphertext of first block.

cyph = plain + key mod 10^{20}

=> key = cyph - plain mod 10^{20}

Also works if Eve knows of some number of ways Alice's message likely to begin. Tries each one => set of possible keys => tries each key on entire message.

Using prior knowledge about message -- in English

Ciphertext-only attack

To get useful information about plaintext.

Each month Alice sends Bob amount to spend, encrypted with 20-digit addition cipher, same key.

Eve intercepts Jan and Feb ciphertexts.

Jan ciph = Jan plain + key mod 10^{20}

Feb ciph = Feb plain + key mod 10^{20}

Jan ciph - Feb ciph = Jan plain - Feb plain mod 10^{20}

Modular Arithmetic (cont.)

Like normal arithmetic: commutative, associative and distributive

Can reduce intermediate results modulo n.

$(a*b) \bmod n = ((a \bmod n)(b \bmod n) \bmod n)$

Speeding up exponentiation in modular arithmetic

$a^8 \bmod n = (a*a*a*a*a*a*a*a) \bmod n$

$= ((a^2 \bmod n)^2 \bmod n)^2 \bmod n$

$a^{25} \bmod n = (a*a^8 * a^{16}) \bmod n$

$= (a* ((a^2)^2)^2 * (((a^2)^2)^2)^2) \bmod n$

$= (a* (((a * a^2)^2)^2)^2) \bmod n$

Can be done in O(log x) operations [$a^x \bmod n$]

Prime Numbers and GCD

A prime number is an integer greater than 1 whose only factors are 1 and itself. No other number evenly divides it.

Examples: 2, 3, 5, 7, 11, 13,...,73, 2521, $2^{756839} - 1$

Two numbers, a and n, are **relatively prime** when they share no factors in common other than 1, i.e., the **greatest common divisor (gcd)** of a and n is equal to 1.

gcd(a,n) = 1

Examples: 4, 9

15, 28

15, 27

5, 12

Inverses Modulo a Number

4 and 1/4 are inverses because 4* 1/4 = 1

In modulo world, want to find x such that

$1 = a*x \pmod n$

Also written $a^{-1} = x \pmod n$

Has unique solution if a and n are relatively prime.

If a and n aren't relatively prime, has no solution.

$4x = 1 \pmod 7$ <=> Finding x, k such that $4x = 7k + 1$

Inverse of 5 modulo 14 = 3

2 has no inverse modulo 14.

Extended Euclidean Algorithm

can be used to calculate the gcd of two numbers a and b

to calculate the the multiplicative inverse modulo n of a number a.

Fermat's Little Theorem

If m is a prime and a is not a multiple of m, then

$a^{m-1} = 1 \bmod m$

Euler Totient Function (Euler phi function $\phi(n)$)

$\phi(n)$ = number of positive integers less than n that are relatively prime to n (n > 1).

If n is prime, $\phi(n)$ =

If n = pq, where p and q are prime,

$\phi(n)$ =

Multiplicative Inverse

Euler's generalization of Fermat's Little Theorem.

If gcd(a,n) = 1, then

$a^{\phi(n)} \bmod n = 1$

=> $a^{-1} \bmod n$ =

Example: $5^{-1} \bmod 7$

Special case of Chinese Remainder Theorem

p and q prime

There is a unique x < pq such that

$x = a \bmod p$

$x = b \bmod q$

Can find it by first finding u such that $uq=1 \bmod p$

Then $x = (((a-b)u) \bmod p)q + b$

Easy corollary: $x = a \bmod p$, $x = a \bmod q$ => $x = a \bmod n$.

Summary of Number-theoretic Preliminaries

Modular Arithmetic:

- Speed up calculations by reducing modulo n
- Exponentiation is fast.

Two numbers relatively prime when share no common factors.

$a^{-1} \pmod n$ is the number x such that $ax \pmod n = 1$.

- has unique solution if a and n are relatively prime.
- has no solution otherwise.

Extended Euclidean Algorithm

- to calculate gcd(a,b)
- to calculate $a^{-1} \pmod n$

Summary, cont.

Fermat's Little Theorem: If m is a prime and a is not a multiple of m, then

$a^{m-1} = 1 \bmod m$

Euler phi function $\phi(n)$ = number of positive integers less than n that are relatively prime to n ($n > 1$).

If n = pq, where p and q are prime, $\phi(n) = (p-1)(q-1)$

Euler's generalization of FLT: If gcd(a,n) = 1, then

$a^{\phi(n)} \bmod n = 1$

Summary, cont.:

Special Case of Chinese Remainder Theorem:

p, q prime, n= pq

There is a unique $x < pq$ such that

$x = a \bmod p$

$x = b \bmod q$

Easy corollary: $x = a \bmod p$, $x = a \bmod q$

$\Rightarrow x = a \bmod n$.

The RSA Cryptosystem

RSA

Bob selects at random 2 large prime numbers p and q, say 150 decimals each.

Compute n = pq

Bob chooses integer e relatively prime to $\phi(n) = (p-1)(q-1)$.

Compute d as multiplicative inverse of e, modulo $\phi(n)$.

Publish P=(e,n) as public key.

Keep secret S=(d,n) as private (secret) key.

Domain of plaintexts is $\mathbf{Z}_n$

Encryption function $E(M) = C = M^e \pmod n$

Decryption function $D(C) = C^d \pmod n$

Example

p=47, q = 71 => n= pq = 3337

encryption key e must have no factors in common with (p-1)(q-1)=46 * 70 = 3220.

Choose e at random to be 79.

$\Rightarrow d = 79^{-1} \bmod 3220 = 1019$. [(1019 x 79) mod 3220 = 1]

Publish e, n; keep d secret.

To encrypt M = 688, then $E(M) = 688^{79} \bmod 3337 = 1570$

To decrypt C= 1570, then $D(C) = 1570^{1019} \bmod 3337 = 688$

Issues

Why does it work?

How do we implement it?

What kind of security guarantees does it provide?

Why does it work? Encryption & decryption inverses

$N=pq$, $de = 1 \bmod \phi(n)$
$E(M) = M^e \bmod n = C$
$D(C) = C^d \bmod n$.

Assume M and n relatively prime

$de = 1 \bmod \phi(n) \Rightarrow de = t\,\phi(n) + 1$ for t integer

$\Rightarrow (M^e)^d \bmod n = M^{ed} \bmod n$
$= M^{t\,\phi(n)+1} \bmod n$
$= (M^{\phi(n)})^t M \bmod n$
$= 1^t M \bmod n$ by F.L.T.
$= M \bmod n$

Implementing RSA

Bob generates two large primes, p & q
- Probabilistic primality testing $O((\log n)^3)$

Bob computes $n = pq$ and $\phi(n) = (p-1)(q-1)$

Bob chooses random e $(1 < e < \phi(n))$ such that $\gcd(e, \phi(n)) = 1$
- Euclidean algorithm

Bob computes $d = e^{-1} \bmod \phi(n)$
- Extended Euclidean Algorithm $O((\log n)^2)$

Bob publishes n and e in a directory as his public key.

Probabilistic Primality Testing

FLT: $a^{m-1} = 1 \bmod m$, for m prime (*)

For most composite numbers, equation false for more than half the a's.

Gives way to test a number m to see if it's prime.
- Choose a random $1 <= a <= m-1$.
- Raise it to power m-1 to see if equation (*) is true.
- If not, m isn't prime.
- If is, repeat with bunch more random a's.

Probabilistic Primality Testing

To find a random 100 digit prime number, pick random 100 digit number, and perform test.

Keep going until you choose a number that turns out to be prime.

Luckily, primes are plentiful.

(Prime Number Theorem: # primes <= N about N/ ln N)

=> About 1/230 100 digit numbers are prime.

A few obvious questions….

If everyone needs a different prime number, won't we run out?
- About 10^{151} prime numbers < 512 bits

What is two people accidentally pick the same prime number?

If someone creates a database of all primes, won't he be able to use the database to break public-key algorithms?

Implementing RSA, cont.

Break input into numerical blocks smaller than n.

Encryption and decryption
- Modular exponentiation

Security of RSA

Rests on difficulty of factoring large numbers.

If factoring large integers is easy, breaking RSA is easy

Converse unproven: it is **conjectured** that if factoring large numbers is hard, breaking RSA is hard.

Sure as hell better make sure n is a very very big number.

Factoring

Factoring a number means finding its prime factors.

$10 = 2 * 5$

$60 = 2 * 2 * 3 * 5$

$833816926455584605284210207 1 =$
$179424673 * 2038074743 * 22801763489$

To factor a number n, best known algorithm (number field sieve) has exponential running time

$$e^{c (\ln n)^{1/3} (\ln \ln n)^{2/3}}$$

A bit of factoring history...

1971 The big news was the factoring of a 41 digit number.

1991 RSA Data Security Inc set up RSA Factoring Challenge: list of hard numbers, each product of 2 primes, ranging from 100 digits to 500 digits.

1994 "RSA129", one of challenge numbers, 129 digits (428) bits), factored over 8 months, using 1600 computers on Internet around the world (~5000 MIPS-years)

"We conclude that commonly used 512-bit RSA moduli are vulnerable to any organization prepared to spend a few million dollars and to wait a few months."

With this method, a 250-digit number would take 100,000,000 times as long.

General Comments About Public-Key Cryptosystems

Slow.

Vulnerable to exhaustive search, and chosen-ciphertext attacks.

Hybrid Cryptosystems

In practice, public-key crypto used to secure and distribute **session keys**, which are then used with private-key crypto to secure message traffic.

Bob sends Alice his public key.

Alice generates random session key K, encrypts it using Bob's public key, and sends it to Bob.

Bob decrypts Alice's message using his private key to recover session key.

Both encrypt their communications using same session key.

Public-key crypto solves important key-management problem.

Exercise: Show RSA encryption and decryption inverses.

$N=pq$, $de = 1 \bmod \phi(n)$

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

$D(C) = C^d \bmod n$.

Show $(M^e)^d \bmod n = 1$, also in case where M, n not relatively prime.

Digital Signatures

Digital Signatures

Hand-written signatures used as proof of authorship or agreement with contents of a document.

- authentic
- unforgeable
- not reusable
- unalterable
- cannot be repudiated.

Not so obvious how to do on a computer.

- Trivial to copy, cut and paste, easy to modify,….

Digital Signatures

Two components:

- secret signing algorithm S_k (M) = Signature
- public verification algorithm V_k (M, Signature)
  V_k (M, Signature)= true if S_k(M) = Signature,
  false otherwise.

Signing Documents with Public Key Cryptography

How Alice sends a **signed** message to Bob using RSA

- Alice signs with her private key. S (M) = D_A(M)
- Bob verifies with Alice's public key. V (C) = E_A(C)

A B

$C = D_A(M)$ C,M

----------> $M = E_A(C)$

Authentic, unforgeable, not reusable, unalterable, can't be repudiated.

Just to be completely clear… Using RSA…

How Alice sends a **secret** message to Bob

A B

$C = E_B(M)$ C

----------> $M = D_B(C)$

How Alice sends a **signed** message to Bob

A B

$C = D_A(M)$ C

----------> $M = E_A(C)$

Digital Signatures

=> can use RSA public-key cryptosystem to provide digital signatures.

There are many other digital signature schemes:

- Discrete Log Signature Schemes.
- DSA
- …..

Problem

Copy of signed digital message identical to original.

- Bob can cheat by reusing document and signature together.
- Example: Alice sends Bob a signed digital check for $1000.

Solution: timestamp.

Digital Signatures + Encryption

proof of authorship + privacy

Alice signs with her private key then encrypts with Bob's public key

A B

$C = E_B(S_A(M))$ C

---------->

Bob decrypts with his private key, then verifies with Alice's public key.

$S_A(M) = D_B(C)$

$M = V_A(C)$

Issues

Bad idea to encrypt then sign.

Timestamps should be used to prevent reuse of messages.

Digital Signatures useful for

Authentication: protocol by which the receiver of a message is convinced of the identity of the sender and the integrity of the message.

Chosen Ciphertext Attack Against RSA: Scenario 1

Eve collects c. She needs m, for which $m = c^d$

she chooses random $r < n$, she gets Bob's public key and then computes

$x = r^e \bmod n \Rightarrow x^d = r^{ed} \bmod n$

$\Rightarrow x^d = r \bmod n$

$y = xc \bmod n$

$t = r^{-1} \bmod n$

Eve gets Bob to decrypt y with his private key. Bob sends Eve

$u = y^d \bmod n$.

Now Eve computes

$tu \bmod n = r^{-1} y^d \bmod n = r^{-1} x^d c^d \bmod n$

$= c^d \bmod n = m$.

Chosen Ciphertext Attack Against RSA: Scenario 2

Trent is a computer notary public. When Alice wants a document notarized, she sends it to Trent who signs it with an RSA digital signature.

Mallory wants Trent to sign a message he otherwise wouldn't, call it m'

Mallory chooses arbitrary x and computes $y = x^e \bmod n$ (where e is Trent's public key).

Then he computes $m = ym' \bmod n$ and sends m to Trent to sign.

Trent returns $m^d \bmod n = (ym')^d \bmod n = xm'^d \bmod n$.

Mallory calculates $(m^d \bmod n) x^{-1} \bmod n = m'^d \bmod n$, which is the signature of m'.

What's going on?

$(xm)^d \bmod n = x^d \, m^d \bmod n$