

RSA Cryptography: Motivation

Alice
Bob



RSA Cryptography: Motivation

Alice
Bob

We need a design such that Eve can also get a supply of Bob's locks, but cannot deduce the key.



RSA Cryptography: Motivation

We want a function $E(m, K_e)$ which converts a message m into some ciphertext c which appears meaningless to any observer— $E(m, K_e)$ must be a *one-way* function.

Alice (or anybody else) can use this function to encrypt her message before sending it to Bob.

Of course, Bob must be able to apply some secret function $D(c, K_d)$ to recover m —For any m , it must hold that $m = D(E(m, K_e), K_d)$.

What we need is called a *trapdoor* function.

RSA Cryptography: Motivation

Recall that

$$f(x) \equiv g^x \pmod{p}$$

is an effective one-way function, with a fixed g and prime modulus p .

Unfortunately, this is believed to be very difficult to invert for anybody.

(There are no known trapdoors)

RSA Cryptography: Motivation

What about

$$f(x) = x^e \pmod{N}$$

with a fixed e and composite modulus N ?

This function is one-way, for carefully chosen e, N , and yet it has a trapdoor!


RSA Cryptography: Mathematical Principles

To understand why, we will need the **Euler totient** (or ϕ) function.

For any n , $\phi(n)$ is defined as the number of positive integers x in the range $1 \leq x \leq n$ such that $x \perp n$.

(Note: $x \perp n$ is shorthand for "x and n are relatively prime")

$\phi(15) = 8$ since $\{1, 2, 4, 7, 8, 11, 13, 14\}$ are $\perp 15$



Leonhard Euler 1707-1783

RSA Cryptography: Mathematical Principles

Theorem #1: If p is prime, then $\phi(p) = p - 1$.

Theorem #2: If $x = p^n$ for some prime p , then $\phi(x) = p^n - p^{n-1}$.

Theorem #3: If $x = m \cdot n$, with $m \perp n$, then $\phi(x) = \phi(m) \cdot \phi(n)$.

RSA Cryptography: Mathematical Principles

Given the factorization of a number N :

$$N = p_1^{e_1} \cdot p_2^{e_2} \cdot \dots \cdot p_n^{e_n}$$

$$\phi(N) = \phi(p_1^{e_1}) \cdot \phi(p_2^{e_2}) \cdot \dots \cdot \phi(p_n^{e_n})$$

$$\phi(N) = (p_1^{e_1} - p_1^{e_1-1}) \cdot (p_2^{e_2} - p_2^{e_2-1}) \cdot \dots \cdot (p_n^{e_n} - p_n^{e_n-1})$$

So, computing $\phi(N)$ is easy if the factorization of N is known.

Euler proved this in the 1730's.

RSA Cryptography: Mathematical Principles

$\phi(N)$ can be a type of "trapdoor" knowledge.

Suppose Alice picks two primes p, q and then computes $N = p \cdot q$. She knows

$$\phi(N) = \phi(p) \cdot \phi(q) = (p - 1) \cdot (q - 1)$$

But if she publishes N , nobody else can easily compute $\phi(N)$ without factoring N .

As it happens, $\phi(N)$ enables Alice to invert the otherwise one-way function $f(x) = x^e \pmod{N}$

RSA Cryptography: Mathematical Principles

Euler's Theorem: For any x, N , with $x \perp N$,

$$x^{\phi(N)} \equiv 1 \pmod{N}$$

(Corollary) Fermat's Little Theorem:

For any prime p , $x^{p-1} \equiv 1 \pmod{p}$

Pierre Fermat stated, but did not prove, in 1640.

RSA Cryptography: The Basic System

1. Alice randomly picks two large (~512 bit) primes p, q .
2. Alice computes $N = p \cdot q$.
3. Alice computes $\phi(N) = (p - 1) \cdot (q - 1)$
4. Alice picks an encryption exponent e , with $e \perp \phi(N)$
5. Alice computes her decryption exponent d , which satisfies

$$e \cdot d \equiv 1 \pmod{\phi(N)}$$

Such a d must exist, since $\gcd(e, \phi(N)) = 1$, there are integers d, k , such that $d \cdot e + k \cdot \phi(N) = 1$.

So $e \cdot d \equiv 1 \pmod{\phi(N)}$.

RSA Cryptography: The Basic System

1. Alice randomly picks two large (~512 bit) primes p, q .
2. Alice computes $N = p \cdot q$.
3. Alice computes $\phi(N) = (p - 1) \cdot (q - 1)$
4. Alice picks an encryption exponent e , with $e \perp \phi(N)$
5. Alice computes her decryption exponent d , which satisfies

$$e \cdot d \equiv 1 \pmod{\phi(N)}$$

Such a d must exist, since $\gcd(e, \phi(N)) = 1$, there are integers d, k , such that $d \cdot e + k \cdot \phi(N) = 1$.

So $e \cdot d \equiv 1 \pmod{\phi(N)}$.

Alice's **Public Key** is (N, e) . Her **Private Key** is (N, d) .

To encrypt message M for Alice, compute $E(M, N, e) \equiv M^e \pmod{N}$

To decrypt ciphertext C , Alice computes $D(C, N, d) \equiv C^d \pmod{N}$

RSA Cryptography: The Basic System

To be usable, the system must satisfy
 $M = D(E(M, N, e), N, d)$

We can verify that:

$$(M^e)^d \equiv M^{ed} \pmod{N}$$

RSA Cryptography: The Basic System

To be usable, the system must satisfy
 $M = D(E(M, N, e), N, d)$

We can verify that:

$$\begin{aligned} (M^e)^d &\equiv M^{ed} \pmod{N} \\ &\equiv M^{k \cdot \phi(N) + 1} \pmod{N} \quad (\text{for some } k) \\ &\equiv M^{k \cdot \phi(N)} \cdot M \pmod{N} \\ &\equiv (M^{\phi(N)})^k \cdot M \pmod{N} \\ &\equiv 1^k \cdot M \pmod{N} \quad (\text{by Euler's Theorem}) \\ &\equiv M \pmod{N} \end{aligned}$$

RSA Cryptography: An Example

Suppose Bob picks $p = 97, q = 113, N = p \cdot q = 10961$.

- $\phi(N) = (p - 1) \cdot (q - 1) = 96 \cdot 112 = 10752$
- Set $e = 11$
- Set $d = 1955$, since $e \cdot d \equiv 1 \pmod{\phi(N)}$

(It's easy to find d with the Extended Euclidean Algorithm)

Alice wants to send the message **5678**. She encrypts:

$$\begin{aligned} C &\equiv M^e \pmod{N} \\ C &\equiv (5768)^{11} \pmod{10961} \\ C &= 9654 \end{aligned}$$

Bob receives the message 9654 and decrypts:

$$\begin{aligned} M &\equiv C^d \pmod{N} \\ M &\equiv (9654)^{1955} \pmod{N} \\ M &= 5678 \end{aligned}$$

RSA Cryptography: Square and Multiply

Is computing $M^e \pmod{N}$ realistic when M, N are large? Yes, using the Square-and-Multiply Algorithm.

Suppose we want to compute $3^{41} \pmod{187}$

First, compute squares:

$$\begin{aligned} 3^1 &\equiv 3 \pmod{187} & 3^2 &\equiv 9 \pmod{187} & 3^4 &\equiv 81 \pmod{187} \\ 3^8 &\equiv 16 \pmod{187} & 3^{16} &\equiv 69 \pmod{187} & 3^{32} &\equiv 86 \pmod{187} \end{aligned}$$

Now, write 41 in binary: $101001 = 32 + 8 + 1$.

Finally, multiply the appropriate powers of two:

$$3^{41} \equiv 3^1 \cdot 3^8 \cdot 3^{32} \equiv 3 \cdot 16 \cdot 86 \equiv 14 \pmod{187}$$

This algorithm requires at most $2 \cdot \log_2(e)$ multiplications

RSA Cryptography: General Security

Suppose Eve sees $C \equiv M^e \pmod{N}$, with e, N , public.

Can she recover M ?

- It is strongly believed, but not proven, that Eve cannot efficiently revert C without knowing $\phi(N)$.
- It has been proven that finding $\phi(N)$ is equivalent to factoring N .
- It is strongly believed, but not proven, that factoring large numbers is difficult.

RSA Cryptography: The Factoring Problem

Suppose $N = 279978339112213278708294676387226016210704467869$
 $5542853756000992932612840010760934567105295536085606182$
 $2351910951365788637105954482006576775098580557613579098$
 $734950144178863178946295187237869221823983$

What are p and q ?

RSA Cryptography: The Factoring Problem

Suppose N= 279978339112213278708294676387226016210704467869
 554285375600992932612840010760934567105295536085606182
 2351910951365788637105954482006576775098580557613579098
 734950144178863178946295187237869221823983

What are p and q?

This was the RSA-200 challenge, broken 5/9/05 by Jens Franke's team at the University of Bonn, Germany. Their prize was \$20,000 US.

p = 3532461934402770121272604978198464368671197400197625023
 649303468776121253679 423200058547956528088349

q = 792586995447833303334708584148005968773797585736421996
 0734330341455767872818 152135381409304740185467

RSA Cryptography: The Factoring Problem

How did they do it? Using a sophisticated algorithm called the **General Number Field Sieve**.

It's complexity: $O\left(\exp\left(\left(\frac{64}{9}\log n\right)^{\frac{1}{3}}(\log\log n)^{\frac{2}{3}}\right)\right)$

So a 200-digit (663 bit) N does not require 2^{663} work to factor. As a result, typically a 1,024-bit N value is used, which requires $\sim 2^{80}$ work to factor.

The U.S. Government uses N values as high as 15,360 bits for TOP SECRET communications.

RSA Cryptography: Key Length and Complexity

Traditionally, "bits of security" are used to indicate the strength of a cryptographic system. For a k-bit symmetric key, exhaustive search will take 2^k work. What is secure?

- < 40 bits Totally insecure, can be cracked on a PC
- 56 bits Key length of DES, 1976 standard (now obsolete)
- 64 bits Largest publicly cracked keys, by Distributed.net
- 70-80 bits Allegedly searchable by the NSA
- 88 bits Would require testing 1 Bil. keys/sec for the age of the universe
- 128 bits Standard for AES, as of 2001
- 256 bits Required for US Government TOP SECRET material

Typical RSA: 1,024 bit N provides ~ 80 bits of security.

RSA Cryptography: Performance Dilemma

Increasing the security margin requires greatly increasing the key size due to the existence of sub-exponential factoring algorithms. Additionally, factoring algorithms are still improving.

Yet the time to encrypt and decrypt increases cubically with the key size.

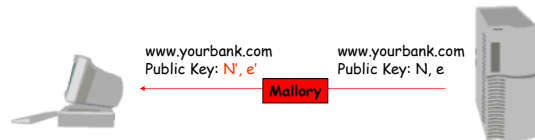
As a result, RSA performance is poor, and degrades quickly as security requirements increase. Full messages are virtually never encrypted with RSA, only small cryptographic keys. Symmetric ciphers are used to encrypt actual data.

RSA Cryptography: Implementation

- As described, "textbook" RSA is vulnerable to many subtle attacks which do not require factoring.
- Implementing RSA properly is very difficult. Always use a proven library.

RSA Cryptography: PKI

The final question: how to securely get public keys onto your computer?



RSA Cryptography: PKI

Idea #1: Use a "secure" channel



Do you want to type

279978339112213278708294676
387226016210704467869554285
375600099293261284001076093
4567105295536085606182235191
095136578863710595448200657
677509858055761357909873495
0144178863178946295187237869
221823983

just to send an email?

RSA Cryptography: PKI

Idea #2: Certificates



Everybody agrees to trust a common Certificate Authority (CA)

The CA takes a file with your name & public key, then digitally signs it after verifying your identity

This is called a certificate

Anybody who trusts the CA can then trust the identity behind your public key

RSA Cryptography: PKI

Who picks the root Certificate Authority?

In all fairness:

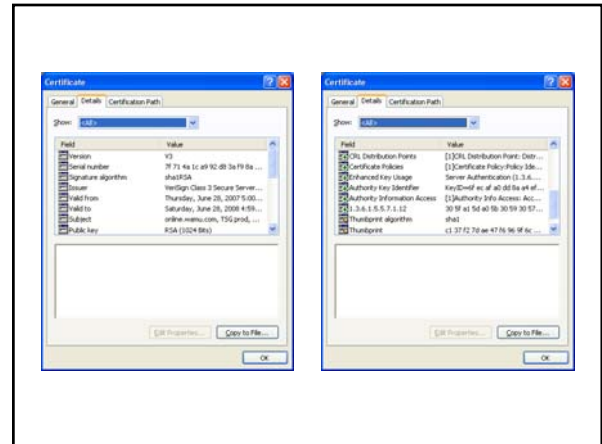
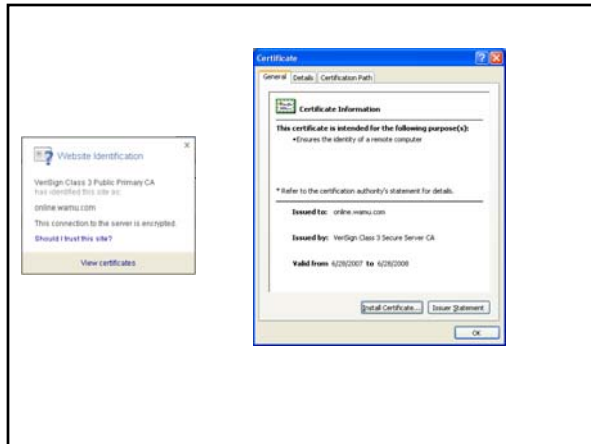


RSA Cryptography: PKI



In your browser, the padlock icon indicates an SSL session.

By clicking on it, you can examine the certificate you've agreed to trust.

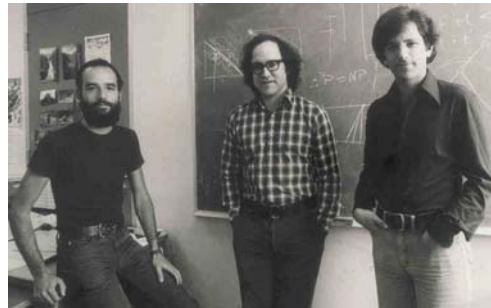


RSA Cryptography: PKI



- VeriSign, Inc founded in 1995
- Based in Mountain View
- Currently the largest root CA
- 2007 revenue: \$1.5 Billion
- Price for a 1-year certificate: \$399
- Price to allow 128-bit encryption: \$995

RSA Cryptography: History



Adi Shamir, Ronald Rivest, and Len Adleman publish RSA at MIT in 1977.

RSA Cryptography: History



2002: Rivest, Shamir, and Adleman receive ACM Turing Award.

RSA Cryptography: Alternate History

1997: Newly declassified documents reveal that English mathematician Clifford Cocks, while working at the British GCHQ in 1973, invented 'RSA' cryptography, four years earlier than the official invention. Cocks rarely receives mention for inventing RSA, although he is now chief mathematician at GCHQ.



RSA Cryptography: Social Implications



By 1991, Phil Zimmermann had decided that the availability of strong encryption software was necessary in the modern age to protect individual freedom from the government.

Zimmermann posted version 1.0 of **Pretty Good Privacy (PGP)** for free, enabling anybody to protect their email with strong RSA encryption and signatures.

Criminal charges against Zimmermann were filed in 1993 for patent violations and export of munitions. The case against Zimmermann was extremely unpopular, and charges were dropped in 1996.

Today, OpenPGP is maintained by GNU, and provides RSA encryption free to anybody.

If you want to learn more:

- CS 255 Introduction to Cryptography
- CS 155 Computer & Network Security
- Math 110 Applied Number Theory

Thanks to Joe Bonneau, former CS103A TA, for many of these slides.