

RSA Public Key Encryption Algorithm

Key Generation

Select p, q	p and q both prime
Calculate $n = p \times q$	
Select integer d	$\gcd(\phi(n), d) = 1; 1 < d < \phi(n)$
Calculate e	$e = d^{-1} \pmod{\phi(n)}$
Public Key	KU = $\{e, n\}$
Private Key	KR = $\{d, n\}$

Encryption

Plaintext: $M < n$
Ciphertext: $C = M^e \pmod{n}$

Decryption

Ciphertext: C
Plaintext: $M = C^d \pmod{n}$

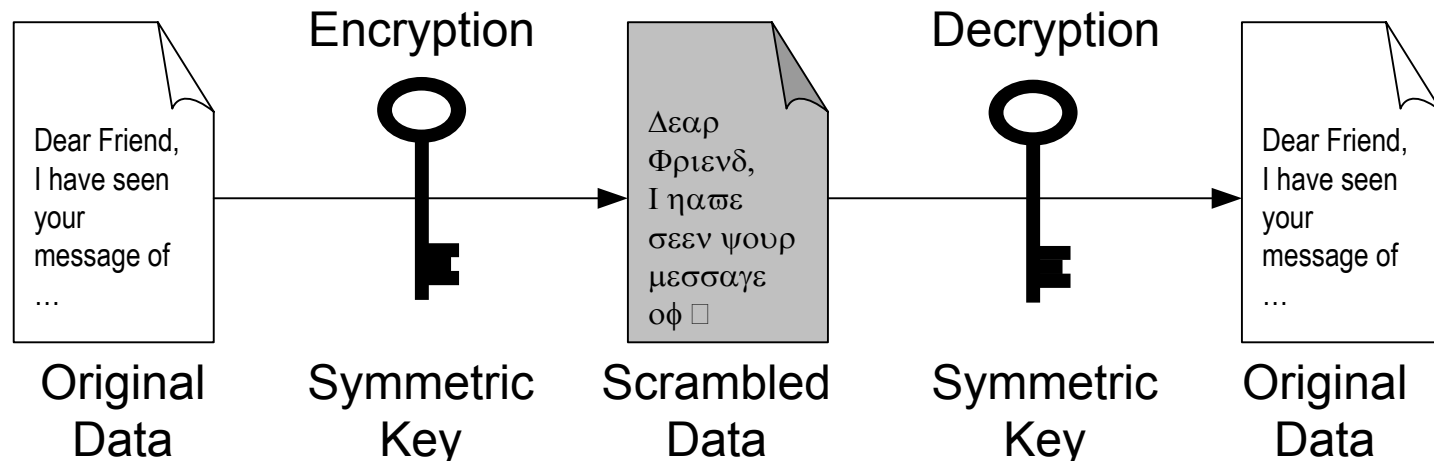
The best known public key cryptosystem is RSA
- named after its authors, Rivest, Shamir and
Adelman

Lecture Plan

- Review of Encryption
- Symmetric and Asymmetric Encryption
- Public Key Cryptography
- Math Behind RSA
- RSA Basic Algorithm
- RSA Algorithm Example
- Uses of RSA
- Security of RSA Algorithm
- RSA FAQs

Review of Encryption

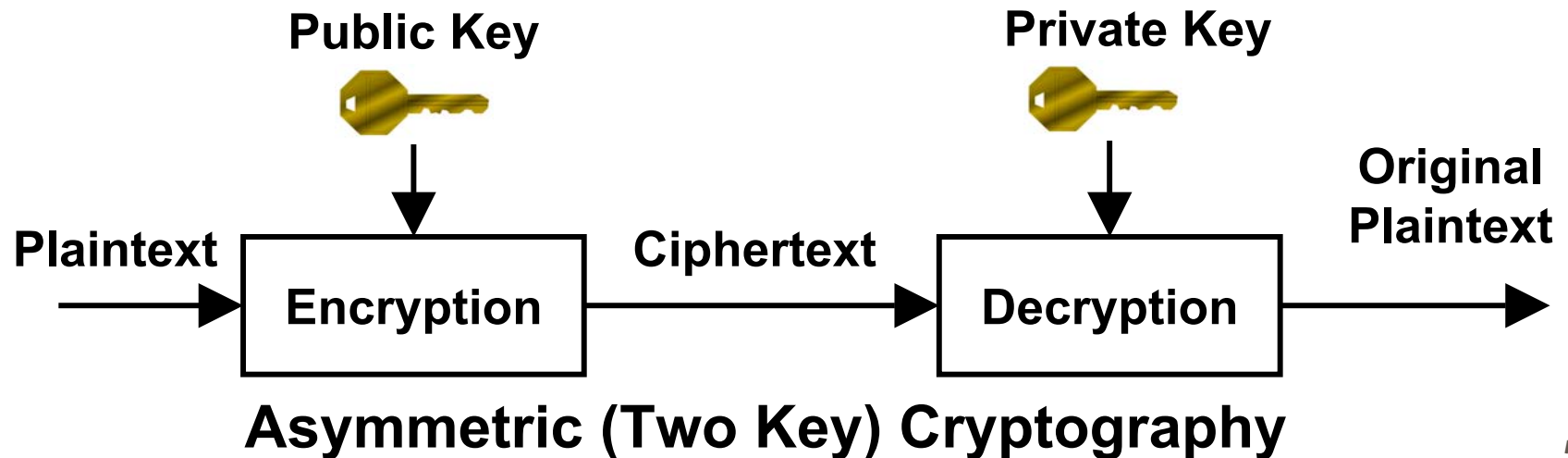
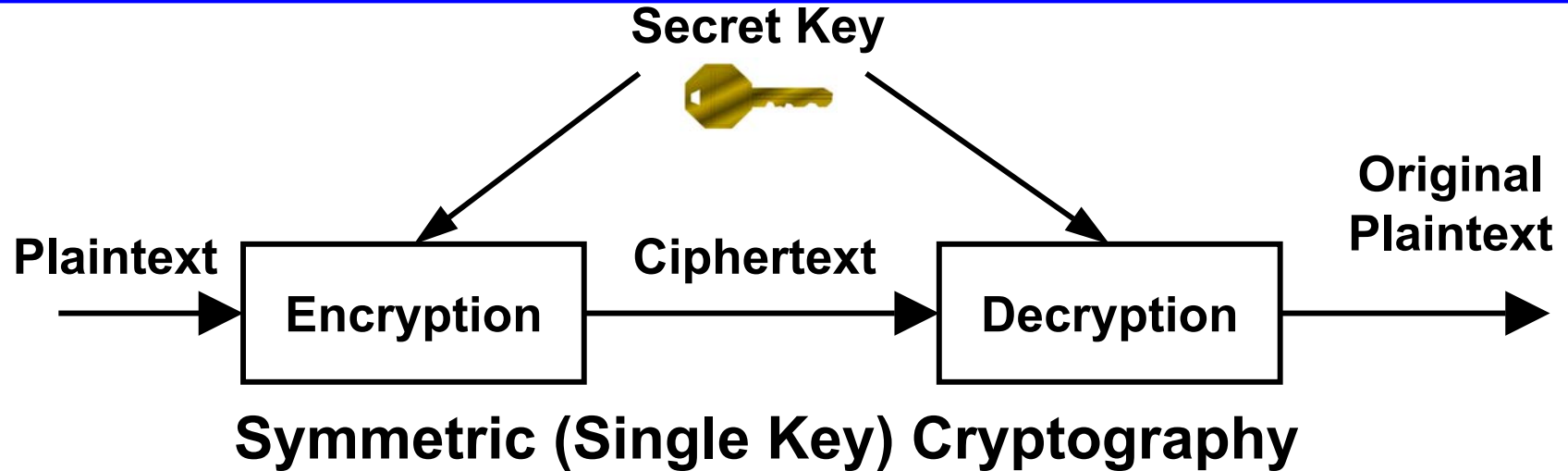
- A message in its original form (plaintext) is encrypted into an unintelligible form (ciphertext) by a set of procedures known as an encryption algorithm (cipher) and a variable, called a key; and the ciphertext is transformed (decrypted) back into plaintext using the encryption algorithm and a key.



Review of Encryption

- Encryption $C = E_K(P)$
- Decryption $P = E_K^{-1}(C)$
- E_K is chosen from a family of transformations known as a cryptographic system.
- The parameter that selects the individual transformation is called the key K , selected from a key space K . For a K -bit key the key space size is 2^K

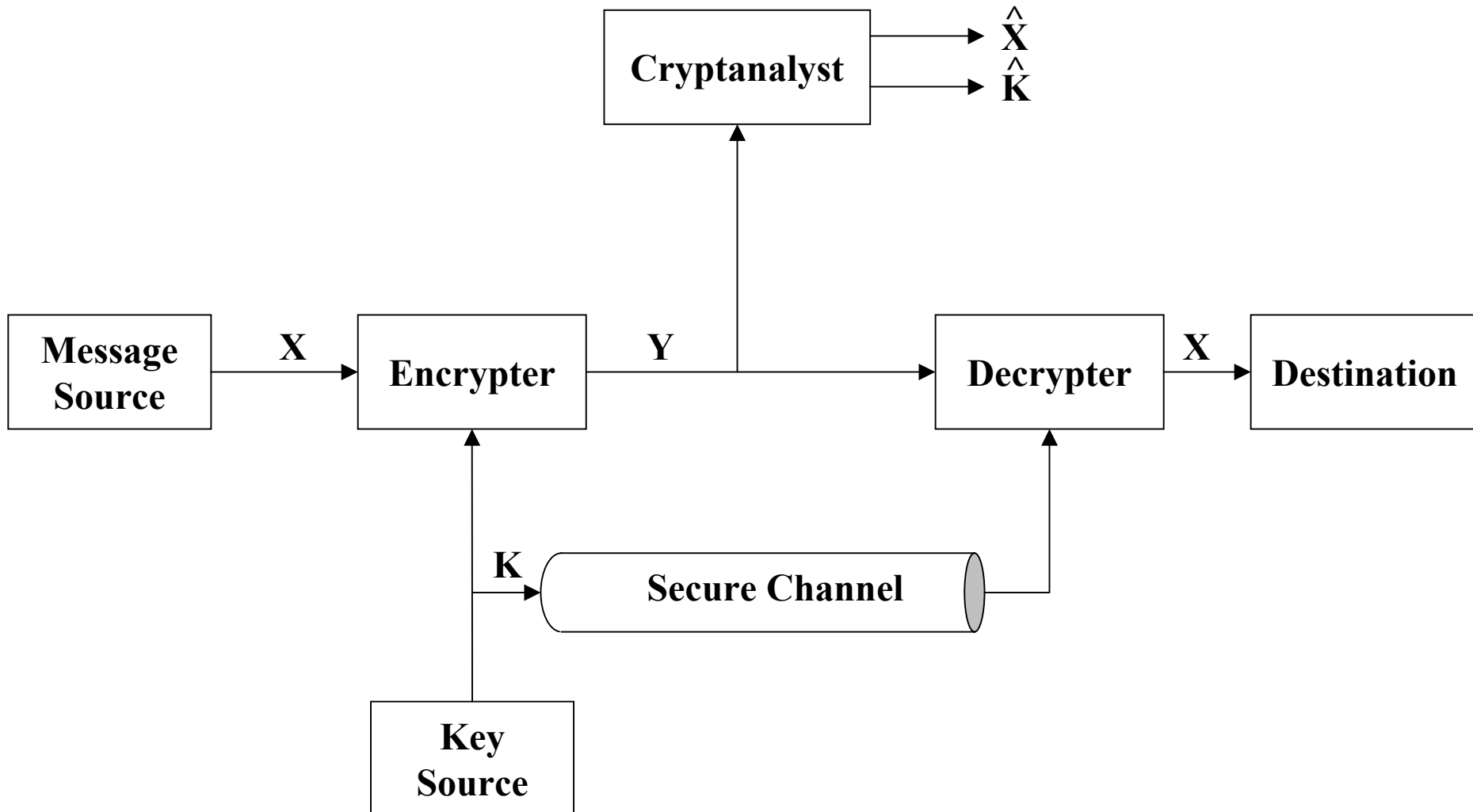
Comparison of Symmetric and Asymmetric Encryption



Secret Key Cryptography Problems

- Traditional (secret key) cryptography uses a single key shared by both sender and receiver. This has some drawbacks:
 - If this key is disclosed communications are compromised - anyone who learns the method of encryption and gets the key, or a number or sequence of numbers or the sequences' equivalent of numbers that are used as a random input into the encrypted system, can break the key.
 - Keys must be exchanged before transmission with any recipient or potential recipient of your message. So, to exchange keys you need a secure method of transmission, but essentially what you've done is create a need for another secure method of transmission. This means that you must either use a secure channel or meet in person in order to share this key. This can be a large problem, and is certainly less than convenient.
 - Also does not protect sender from receiver forging a message and claiming is sent by sender, parties are equal.

Secret Key Cryptography Problems



Public-Key Cryptography

- **Public-key (or two-key) 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**

Alice, Bob and Trudy

- In a Public Key system when Alice sends email to Bob, she finds his public key (possibly in a directory of some sort) and encrypts her message using that key. Unlike secret-key cryptography, though, the key used to encrypt will not decrypt the ciphertext. Knowledge of Bob's public key will not help an eavesdropper. To decrypt, Bob uses his private key. If Bob wants to respond to Alice, he will encrypt his message using her public key.
- Trudy (from Intruder) tries to disrupt the communication between Alice and Bob

Public-Key Cryptography

Requirements

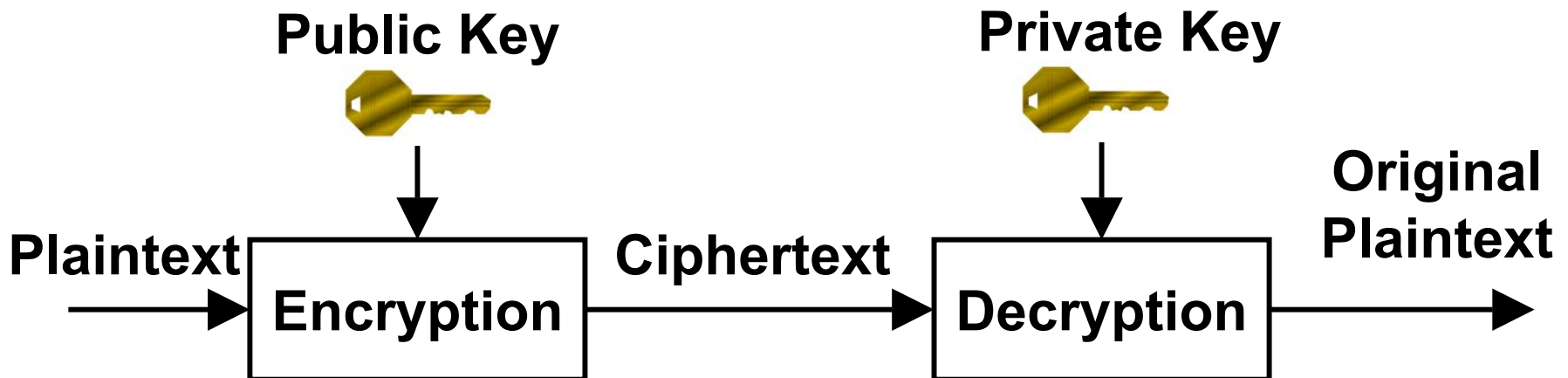
- The public-key is easily computed from the private key and other information about the cipher
- However, knowing the public-key and public description of the cipher, it is still computationally infeasible to compute the private key
- Thus the public-key may be distributed to anyone wishing to communicate securely with its owner (although secure distribution of the public-key is a non-trivial problem - the key distribution problem)

Public Key Encryption Systems

- Because a different key is used on each side of the process, public key systems are also known as 'asymmetric systems'.
- The distribution of keys for public key systems is generally much easier because it is not normally necessary to keep the public key secret.
- The private key, on the other hand, must remain secret or else security is compromised.

Public Key Encryption

- Key Pairs (Public and Private).
- Publish one key, keep the other secret.
- Anyone who wants to send you a message encrypts it using your public key.
- To read a message you decrypt it with the private key.



Public Key Encryption

- A good public key algorithm:
 - Infeasible to derive one key from the other
 - Keys are interchangeable
- Simplifies (but does not solve) key distribution problem
- Public key is slower than secret key algorithms
 - RSA is about 1000-5000 times slower than DES
 - Public key encryption is sometimes used to encrypt a secret key algorithm's session key

Comparison of SK and PK

DISTINCT FEATURES

SECRET KEY

PUBLIC KEY

NUMBER OF KEYS

Single key.

Pair of keys.

TYPES OF KEYS

Key is secret.

One key is private, and one key is public.

SIZE OF KEY

50-250 bits

500-2500 bits

RELATIVE SPEEDS

Faster.

Slower.

Public Key Encryption Has Foundations in Mathematics

- Public key crypto-systems were developed from some very subtle insights about the mathematics of large numbers and how they relate to the power of computers.
- Public Key Encryption works because of what is known in math as a trapdoor problem.
- A trapdoor is a mathematical formula that is easy to work forward but very hard to work backward.

Trapdoors are also called One-Way Functions

- The challenge of public-key cryptography is developing a system in which it is impossible (or at least intractable) to deduce the private key from the public key.
- This can be accomplished by utilizing a one-way function. With a one-way function, given some input values, it is relatively simple to compute a result. But if you start with the result, it is extremely difficult to compute the original input values.
- In mathematical terms, given x , computing $f(x)$ is easy, but given $f(x)$, it is extremely difficult to determine x .

Multiplication is a Mathematical Trapdoor

- It turns out that multiplication can be a one-way function.
- In general it is easy (especially on computers) to multiply two big prime numbers.
- But for most very large numbers, it is extremely time-consuming to factor them.

Multiplication/Factorization

Trapdoor Function

- Public key algorithms depend on a person publishing a large public key and others being unable to factor this public key into its component parts.
- Because the creator of the key knows the factors of his or her large number, he or she can use those factors to decode messages created by others using his or her public key.
- Those who only know the public key will be unable to discover the private key, because of the difficulty of factoring the large number.

Inventors of Public Key Cryptography

- Public Key Cryptosystem idea was invented independently by Whitfield Diffie and Martin Hellman, who presented their concept in 1976, and Ralph Merkle.
- The first public-key algorithm was the Diffie-Hellman key exchange algorithm.
- The first general algorithm which could be used for encryption and decryption was based upon the knapsack problem.
- The first system which could do encryption/decryption as well as signatures was RSA.

Math Behind RSA

RSA is a **public-key cryptosystem** that MIT professors Ronald L. Rivest, Adi Shamir and Leonard M. Adleman invented in 1977. The system is based on several mathematical principles in number theory.

Prime Numbers ...

- A **prime number**, or prime, is a number that is evenly divisible by only 1 and itself.
- For instance 10 is not prime because it is evenly divisible by 1, 2, 5 and 10. But 11 is prime, since only 1 and 11 evenly divide it.
- The numbers that evenly divide another number are called **factors**. The process of finding the factors of a number is called **factoring**.

Factoring a Number ...

- For example, factoring 15 is simple, it is $3 * 5$. But what about 6,320,491,217?
- Now how about a 155-digit number? Or 200 digits or more? In short, factoring numbers takes a certain number of steps, and the number of steps increases subexponentially as the size of the number increases. That means even on supercomputers, if a number is sufficiently large, the time to execute all the steps to factor it would be so great that it could take years to compute.

Modular Math

- Modular math means that the only numbers under consideration are the non-negative integers less than the modulus. So for mod n , only the integers from 0 to $(n - 1)$ are valid operands and results of operations will always be numbers from 0 to $(n - 1)$. Think of military time where the modulus is 2400. For instance, 2200 plus 400 (10:00 PM plus 4 hours) is not 2600. Once you reach 2400, you start over at 0. Hence, $2200 + 400 \text{ mod } 2400$ is $2600 - 2400 = 0200$, or 2:00 in the morning. Likewise, if we start at 0, or midnight, 6 times 500 (say six 5-hour shifts) is not 3000, but 0600, or 6:00 AM the following day.

Modular Arithmetic

- $a = b \pmod{m}$ means that when a is divided by m the remainder is b .
- Examples
 - $11 = 1 \pmod{5}$
 - $20 = 2 \pmod{6}$

Modular Math and Prime Numbers

- Prime numbers possess various useful properties when used in modular math.
- The RSA algorithm takes advantage of these properties.

Modular Inverse

- Another aspect of modular math is the concept of a modular inverse.
- Two numbers are the modular inverses of each other if their product equals 1.
- For instance, $7 * 343 = 2401$, but if our modulus is 2400, the result is:
- $(7 * 343) \bmod 2400 = 2401 - 2400 = 1 \bmod 2400$

Relatively Prime

- Two numbers are **relatively prime** if they share only one factor, namely 1.
- For example, 10 and 21 are relatively prime. Neither is prime, but the numbers that evenly divide 10 are 1, 2, 5 and 10, whereas the numbers that evenly divide 21 are 1, 3, 7 and 21.
- The only number in both lists is 1, so the numbers are relatively prime.

Euler's phi-function

- In the eighteenth century, the mathematician Leonhard Euler (pronounced "Oiler") described $\varphi(n)$ as the number of numbers less than n that are relatively prime to n .
- The character φ is the Greek letter "phi" (in math circles it rhymes with "tea" in the academic organization Phi Beta Kappa it rhymes with "tie"). This is known as Euler's phi-function.

Euler's phi-function

- So $\varphi(6)$, for instance, is 2, since of all the numbers less than 6 (1, 2, 3, 4 and 5), only two of them (1 and 5) are relatively prime with 6. The numbers 2 and 4 share with 6 a common factor other than 1, namely 2. And 3 and 6 share 3 as a common factor.
- What about $\varphi(7)$? Because 7 is prime, its only factors are 1 and 7. Hence, any number less than 7 can share with 7 only 1 as a common factor. Without even examining those numbers less than 7, we know they are all relatively prime with 7. Since there are 6 numbers less than 7, $\varphi(7) = 6$. Clearly this result will extend to all prime numbers. Namely, if p is prime, $\varphi(p) = (p - 1)$.

Exponentiation

- **Exponentiation** is taking numbers to powers, such as 2^3 , which is $2 * 2 * 2 = 8$. In this example, 2 is known as the **base** and 3 is the **exponent**. There are some useful algebraic identities in exponentiation.
- $(b^x) * (b^y) = b^{x+y}$
- $(b^x)^y = b^{xy}$

Exponential Period modulo n

- Euler noticed that $\varphi(n)$ was the “exponential period” modulo n for numbers relatively prime with n .
- What that means is that for any number $a < n$, if a is relatively prime with n , $a^{\varphi(n)} \bmod n = 1$.
- So if you multiply a by itself $\varphi(n)$ times, modulo n , the result is 1. Then if you multiply by a one more time, you are finding the product of $1 * a$ which is a , so you are starting over again.
- Hence, $a^{\varphi(n)} * a = a^{\varphi(n)+1} \bmod n = a$.

Exponential Period modulo n

- For example, if n is 5 (a prime number), then $\varphi(5) = 4$.
Let a be 3 and compute
- $a^{\varphi(n)} \bmod n = 3^4 = 3 * 3 * 3 * 3 \bmod 5$
- $= 81 \bmod 5$
- $= 1 \bmod 5$

Using it to build our PK Cryptosystem

- We can take advantage of this fact in the following way. Take a number m , and raise it to some power e modulo p ,
 - $c = m^e \bmod p$
- Now take the result of that exponentiation, c , and raise it to some other power d ,
 - $c^d \bmod p$
- That is equivalent to
 - $(m^e)^d \bmod p$
- which is equivalent to
 - $m^{ed} \bmod p$
- How is that useful?

Using it to build our PK Cryptosystem

- Suppose someone gave you c , e and p and said, “I computed $c = m^e \bmod p$. Find d such that $c^d \bmod p = 1$.” You would simply find d such that $e * d = \varphi(p)$. Because then
 - $c^d \bmod p = (m^e)^d = m^{ed} = m^{\varphi(p)} = 1 \bmod p$
- But now suppose someone gave you c , e and p and said, “I computed $c = m^e \bmod p$. I want you to find d such that $c^d \bmod p = m$.” You would need to find d such that $e * d = \varphi(p) + 1$. Because then
 - $c^d \bmod p = (m^e)^d = m^{ed} = m^{\varphi(p)+1} = m \bmod p$

Using it to build our PK Cryptosystem

- For example, let $c = 4$, $e = 3$ and $p = 11$. To find m , determine d such that $3d = \phi(11) + 1$. Since 11 is prime, $\phi(11) = 10$. So find d where $3d = 11$. But wait, because we are dealing with integers only, there is no d that will satisfy that equation $3d = 11$. Note that $3 * 3 = 9$ and $3 * 4 = 12$.
- We can make it work with modular math. $\phi(p) + 1$ is $1 \pmod{\phi(p)}$. Remember, when we reach the modulus, we start over at 0. Hence,
 - $(\phi(p) + 1) \pmod{\phi(p)} = (\phi(p) + 1) - \phi(p) = 1 \pmod{\phi(p)}$
- So what you want to find is d such that $e * d = 1 \pmod{\phi(p)}$. Remember, this is known as the modular inverse.

Using it to build our PK Cryptosystem

- Could this be our public-key cryptosystem? Find a prime, p , pick a public exponent, e , and make those two values public.
- Using the extended Euclidian algorithm, determine d , the inverse of the public exponent modulo $\varphi(p) = (p - 1)$.
- Keep d private. When people want to send you a message m , they can encrypt and produce ciphertext c by computing $c = m^e \bmod p$. To recover the plaintext message, you compute $m = c^d \bmod p$.

One Change ...

- There is, of course, one reason this could not be a useful system. Our private key is the inverse of e modulo $(p - 1)$. Since p is public, anyone can compute $(p - 1)$ and therefore determine d .
- The RSA algorithm solves that problem by using an important property of Euler's phi-function. It is "multiplicative." If p and q are relatively prime, then $\varphi(pq) = \varphi(p)\varphi(q)$. Hence, for primes p and q and $n = pq$,
- $\varphi(n) = (p - 1)(q - 1)$.

Coming to RSA ...

- Previously we chose a prime number p to be the modulus. Now, instead, we find two large primes, p and q , and use their product
 - $n = pq$
- as the modulus. We still choose a public exponent, e , and using the extended Euclidian algorithm find d , the inverse of e modulo $\varphi(n)$. This time, however, we are finding the d that satisfies
 - $e * d = 1 \text{ mod } (p - 1)(q - 1)$
- The pair (n, e) is the public key and d is the private key. The primes p and q must be kept secret or destroyed.

Coming to RSA ...

- To compute ciphertext c from a plaintext message m , find
 - $c = m^e \bmod n$
- To recover the original message, compute
 - $m = c^d \bmod n$
- Only the entity that knows d can decrypt.
- Because of the relationship between d and e , the algorithm correctly recovers the original message m , since
 - $c^d \bmod n = (m^e)^d = m^{ed} = m^1 = m \bmod n$

Coming to RSA ...

- Anyone else who wants to compute d , must first know $\varphi(n)$, but to know $\varphi(n)$ one must know p and q . In other words, they must factor n .
Remember the one-way function? We knew that multiplying big prime numbers can be a one-way function, we simply needed to figure out a way to use that fact.
- Here it is, build the private key using two primes and the public key using their product.

Coming to RSA ...

- There is one more condition, the public exponent e must be relatively prime with $(p - 1)(q - 1)$. That is because if e is not relatively prime with $(p - 1)(q - 1)$, there will be no modular inverse.
- Incidentally, in practice you would generally pick e , the public exponent first, then find the primes p and q such that e is relatively prime with $(p - 1)(q - 1)$. There is no mathematical requirement to do so, it simply makes key pair generation a little easier.
- In fact, the two most popular e 's in use today are $F0 = 3$ and $F4 = 65,537$. The F in F0 and F4 stands for Pierre de Fermat, the 17th century mathematician who first described the special properties of these and other interesting numbers.

Application of Public-Key Ciphers

- Three important uses of public-key algorithms:
 - **Public-Key Distribution Schemes (PKDS)** - where the scheme is used to securely exchange a single piece of information (whose value depends on the two parties, but cannot be set). This value is normally used as a session key for a private-key scheme
 - **Signature Schemes** - used to create a digital signature only, where the private-key signs (create) signatures, and the public-key verifies signatures
 - **Public Key Schemes (PKS)** - used for encryption, where the public-key encrypts messages, and the private-key decrypts messages.
- Any public-key scheme can be used as a PKDS, just by selecting a message which is the required session key
- Many public-key schemes are also signature schemes (provided encryption and decryption can be done in either order)

RSA

- RSA - named after Rivest, Shamir and Adleman, the inventors - was the first public-key scheme which was capable of signatures as well as encryption.
- It is the easiest to understand as well as the most popular to implement
- RSA obtains its security from the difficulty of factoring large numbers.

RSA Public-Key Cryptosystem

- RSA was proposed by Rivest, Shamir & Adleman in 1977: R L Rivest, A Shamir, L Adleman, “On Digital Signatures and Public Key Cryptosystems”, Communications of the ACM, vol 21 no 2, pp120-126, Feb 1978
- The algorithm is patented in North America (although algorithms cannot be patented elsewhere in the world) this is a source of legal difficulties in using the scheme

RSA Algorithm

- First choose two large prime numbers, p and q , and find their product, n . n is also called modulus in RSA jargon.
- Compute $z = (p-1)(q-1)$
- Next choose a number e , relatively prime to $z = (p-1)(q-1)$ - this is the encryption key.
- Finally compute d such that the product of e and d is congruent to $1 \pmod{((p-1)(q-1))}$. This is the decryption key.

RSA Algorithm

- Obviously, d can only be recovered if you reveal p and q , or if p and q are recovered from n , the modulus. Since we are assuming the factorization of n to be too hard to attempt, d cannot be recovered from e . Or so it is currently speculated. It has not, so far, been proven.
- Now e and n together form the public key, while d and n together form the private key.

RSA Key Generation

- To use the scheme, first generate keys:
 - Key-Generation by each user consists of:
 - selecting two large primes at random (~ 100 digit), p, q
 - calculating the system modulus $n=p.q$ and p, q are primes
 - selecting at random the encryption key e ,
 - $e < n, \gcd(e, \phi(n)) = 1$

RSA Key Generation (cont'd)

- Solving the congruence to find the decryption key d :
 - $e \cdot d \equiv 1 \pmod{\phi(n)}$ $0 < d < n$
- Publishing the public encryption key:
 $K_{\text{pub}} = \{e, n\}$
- Securing the private decryption key:
 $K_{\text{pvt}} = \{d, p, q\}$

Encryption with RSA

- To encrypt a plaintext message block m , compute
 - $C = M^e \bmod n$
- To decrypt the block, compute
 - $M = C^d \bmod n$
- Each plaintext block must be smaller than the value of n .

RSA Algorithm

Key Generation

Select p, q	p and q both prime
Calculate $n = p \times q$	
Select integer d	$\gcd(\phi(n), d) = 1; 1 < d < \phi(n)$
Calculate e	$e = d^{-1} \text{ mod } \phi(n)$
Public Key	$KU = \{e, n\}$
Private Key	$KR = \{d, n\}$

Encryption

Plaintext: $M < n$
Ciphertext: $C = M^e \pmod{n}$

Decryption

Ciphertext: C
Plaintext: $M = C^d \pmod{n}$

RSA Example

- $p = 3$
- $q = 11$
- $n = p \times q = 33$ -- This is the *modulus*
- $z = (p-1) \times (q-1) = 20$ -- This is the totient function $\phi(n)$.
There are 20 relative primes to 33. What are they? 1, 2, 4, 5, 7, 8, 10, 13, 14, 16, 17, 19, 20, 23, 25, 26, 28, 29, 31, 32
- $d = 7$ -- *7 and 20 have no common factors but 1*
- $7e = 1 \pmod{20}$
- $e = 3$
- $C = P^e \pmod{n}$
- $P = C^d \pmod{n}$

RSA Example

Plaintext (P)		Ciphertext (C)			After decryption	
Symbolic	Numeric	P^3	$P^3 \pmod{33}$	C^7	$C^7 \pmod{33}$	Symbolic
S	19	6859	28	13492928512	19	S
U	21	9261	21	1801088541	21	U
Z	26	17576	20	1280000000	26	Z
A	01	1	1	1	1	A
N	14	2744	5	78125	14	N
N	14	2744	5	78125	14	N
E	05	125	28	8031810176	5	E
Sender's computation				Receiver's computation		

Fig. 7-11. An example of the RSA algorithm.

Digital Signatures Using RSA

- Generally $D_{K_{Pvt}}(E_{K_{Pub}}(P))=P$
- RSA also has the property $D_{K_{Pub}}(E_{K_{Pvt}}(P))=P$
- Since the text can also be encrypted with K_{Pvt} and decrypted with K_{Pub} , it is possible to use RSA for signatures, where a block of data is encrypted with the private key, and can be decrypted with the public key to show that the sender truly did sign/send that data him/herself.

Digital Signatures

- However, simply using the encryption of a plaintext document using the private key is not only inefficient (producing a much-too-large signature) but also insecure.
- Bruce Schneier describes a possible attack in this situation in *Applied Cryptography*.
- It is important to use a one-way hash function before signing a document.

RSA Summary

- RSA (Rivest-Shamir-Adelman) is the most commonly used public key algorithm.
- Can be used both for encryption and for digitally signing.
- It is generally considered to be secure when sufficiently long keys are used (512 bits is insecure, 768 bits is moderately secure, and 1024 bits is good, for now).
- The security of RSA relies on the difficulty of factoring large integers. Dramatic advances in factoring large integers would make RSA vulnerable.
- RSA is currently the most important public key algorithm. It is patented in the United States (expired year 2000), and free elsewhere.

RSA Weaknesses

- At present, 512 bit keys are considered weak, 1024 bit keys are probably secure enough for most purposes, and 2048 bit keys are likely to remain secure for decades.
- One should know that RSA is very vulnerable to **chosen plaintext attacks**. There is also a new **timing attack** that can be used to break many implementations of RSA. The RSA algorithm is believed to be safe when used properly, but one must be very careful when using it to avoid these attacks.

The Previous History of Factoring

- The security of the RSA cryptosystem therefore depends on the fact, that it is **practically impossible to factor the large known modulus n** . So nobody can infer the two primes p and q from his or her knowledge of the publicly known modulus to gain the secret key.
- **70-digit numbers** will be factored today (1998) on a workstation within **10 hours**.
- **100-digit numbers** will be factored on a workstation within **1 year**.

The Previous History of Factoring

- **129**-digit numbers :

- In August 1977 Martin Gardner asked the readers of Scientific American to factor

114 381 625 757 888 867 669 235 779 967 146 612 010 218 296
721 242 362 562 561 842 935 706 935 245 733 897 830 597 123
563 958 705 058 989 075 147 599 290 026 879 543 541 .

- **Some 16 years later**, in April 1994 the factors were presented by Paul Leyland (University of Oxford), Michael Graff (University of Iowa) and Derek Atkins (MIT). They had been supported by over 600 volunteers running a computerprogram written by K. Lenstra (Bell Labs, Morristown, New Jersey) on their workstations at night sharing the work of factoring over the internet.

The Previous History of Factoring

- **140**-digit numbers are the smallest numbers not having been factored in **1996**.
 - They will be factored within about **5 years** using large-scale networking.
- **160**-digit numbers:
 - In 1996 experts expect factoring to be possible within about **5 years** using a new method of factoring known as number field sieve.
- **200**-digit numbers:
 - The time for factoring is estimated at **52 000 000 years** in 1998

How Large Should the Primes be?

- The two primes, p and q , which compose the modulus, should be of roughly equal length; this will make the modulus harder to factor than if one of the primes was very small. Thus if one chooses to use a 768-bit modulus, the primes should each have length approximately 384 bits. If the two primes are extremely close (identical except for, say, 100 - 200 bits), there is a potential security risk, but the probability that two randomly chosen primes are so close is negligible.

Could users of RSA run out of distinct primes?

- As Euclid proved over two thousand years ago, there are infinitely many prime numbers. Because RSA is generally implemented with a fixed key length, however, the number of primes available to a user of the algorithm is effectively finite. Although finite, this number is nonetheless very large. The Prime Number Theorem states that the number of primes less than or equal to n is asymptotic to $n/\ln n$. Hence, the number of prime numbers of length 512 bits or less is roughly 10^{150} . This is greater than the number of atoms in the known universe.

How is RSA used for privacy in practice?

- In practice, RSA is often used together with a secret-key cryptosystem, such as DES, to encrypt a message by means of an RSA digital envelope.
- Suppose Alice wishes to send an encrypted message to Bob. She first encrypts the message with DES, using a randomly chosen DES key. Then she looks up Bob's public key and uses it to encrypt the DES key. The DES-encrypted message and the RSA-encrypted DES key together form the RSA digital envelope and are sent to Bob. Upon receiving the digital envelope, Bob decrypts the DES key with his private key, then uses the DES key to decrypt the message itself. This combines the high speed of DES with the key-management convenience of RSA.

Is RSA an official standard today?

- RSA is part of many official standards worldwide. The ISO (International Standards Organization) 9796 standard lists RSA as a compatible cryptographic algorithm, as does the ITU-T X.509 security standard. RSA is part of the Society for Worldwide Interbank Financial Telecommunications (SWIFT) standard, the French financial industry's ETEBAC 5 standard, the ANSI X9.31 rDSA standard and the X9.44 draft standard for the U.S. banking industry. The Australian key management standard, AS2805.6.5.3, also specifies RSA.
- RSA is found in Internet standards and proposed protocols including S/MIME IPsec, and TLS, the Internet standards-track successor to SSL, as well as the PKCS standard for the software industry. The OSI Implementers' Workshop (OIW) has issued implementers' agreements referring to PKCS, which includes RSA.
- A number of other standards are currently being developed and will be announced over the next few years; many are expected to include RSA as either an endorsed or a recommended system for privacy and/or authentication. A comprehensive survey of cryptography standards can be found in publications by Kaliski [[Kal93b](#)] and Ford [[For94](#)].

Is RSA Currently in Use?

- RSA is currently used in a wide variety of products, platforms, and industries around the world. It is found in many commercial software products and is planned to be in many more. RSA is built into current operating systems by Microsoft, Apple, Sun, and Novell. In hardware, RSA can be found in secure telephones, on Ethernet network cards, and on smart cards. In addition, RSA is incorporated into all of the major protocols for secure Internet communications, including S/MIME, SSL and S/WAN. It is also used internally in many institutions, including branches of the U.S. government, major corporations, national laboratories, and universities.
- RSA technology is licensed by more than 350 companies. The estimated installed base of RSA encryption engines is around 300 million, making it by far the most widely used public-key cryptosystem in the world. This figure is expected to grow rapidly as the Internet and the World Wide Web expand.

How Fast is RSA?

- An "RSA operation," whether encrypting, decrypting, signing, or verifying is essentially a modular exponentiation. This computation is performed by a series of modular multiplications.
- In practical applications, it is common to choose a small public exponent for the public key. In fact, entire groups of users can use the same public exponent, each with a different modulus. (There are some restrictions on the prime factors of the modulus when the public exponent is fixed.) This makes encryption faster than decryption and verification faster than signing.

How Fast is RSA?

- With the typical modular exponentiation algorithms used to implement RSA, public key operations take $O(k^2)$ steps, private-key operations take $O(k^3)$ steps, and key generation takes $O(k^4)$ steps, where k is the number of bits in the modulus. "Fast multiplication" techniques, such as FFT-based methods, require asymptotically fewer steps. In practice, however, they are not as common due to their greater software complexity and the fact that they may actually be slower for typical key sizes.

How Fast is RSA?

- The speed and efficiency of the many commercially available software and hardware implementations of RSA are increasing rapidly. On a 90 MHz Pentium, has a throughput for private-key operations of 21.6 kbits per second with a 512-bit modulus and 7.4 kbits per second with a 1024-bit modulus. The fastest RSA hardware has a throughput greater than 300 kbits per second with a 512-bit modulus, implying that it performs over 500 RSA private-key operations per second (There is room in that hardware to execute two RSA 512-bit RSA operations in parallel, hence the 600 kbits/s speed reported in [[SV93](#)]. For 970-bit keys, the throughput is 185 kbits/s.). It is expected that RSA speeds will reach 1 mbits/second in late 1999.

How Fast is RSA?

- By comparison, DES and other block ciphers are much faster than RSA. In software, DES is generally at least 100 times as fast as RSA.
- In hardware, DES is between 1,000 and 10,000 times as fast, depending on the implementation.
- Implementations of RSA will probably narrow the gap a bit in coming years, due to high demand, but DES will get faster as well.