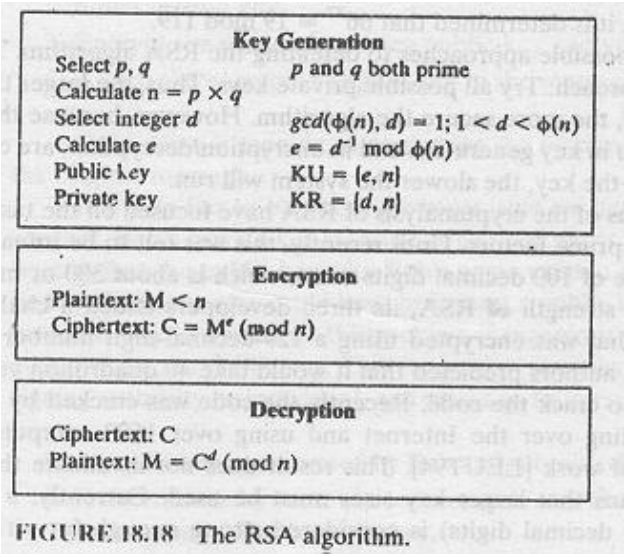
RSA Public Key Encryption Algorithm



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	$e = d^{-1} \bmod \phi(n)$
Public key	$KU = \{e, n\}$
Private key	$KR = \{d, n\}$

Encryption

Plaintext: $M < n$

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

Decryption

Ciphertext: C

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

FIGURE 18.18 The RSA algorithm.

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

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.

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**

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.

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**.

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 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 mod (m)* means that when *a* is divided by *m* the remainder is *b*.
- Examples
- 11 = 1 mod (5)
- 20 = 2 mod (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) mod 2400 = 2401 – 2400 = 1 mod 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 φ(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 φ(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, φ(7) = 6. Clearly this result will extend to all prime numbers. Namely, if p is prime, φ(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, $a^{\varphi(n)} * a = a^{\varphi(n)+1} \bmod n = a$.

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

- 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
  - $e * d = 1 \bmod (p - 1)(q - 1)$
- 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 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 $1 \bmod ((p-1)(q-1))$. 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, ϕ(n)) = 1

RSA Key Generation (cont'd)

- Solving the congruence to find the decryption key d:
  - $e.d \equiv 1 \bmod \phi(n)$ $0 <= d <= n$
- Publishing the public encryption key: Kpub={e,n}
- Securing the private decryption key: Kpvt={d,p,q}

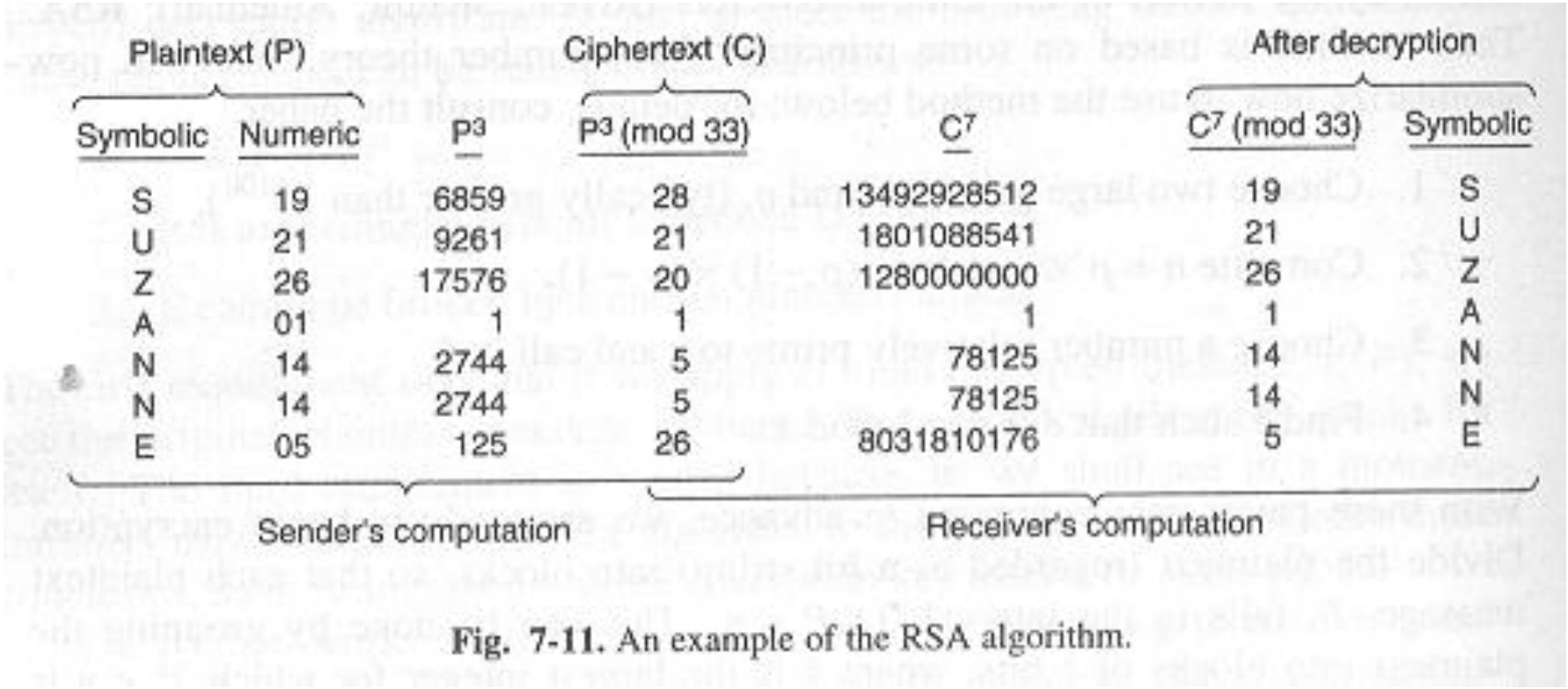
Encryption with RSA

- 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 Example

- *p = 3*
- *q = 11*
- *n = p X q = 33* -- This is the *modulus*
- *z = (p-1) X (q -1) = 20* -- This is the totient function ϕ(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 mod 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 (mod 33)	C^7	C^7 (mod 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	26	8031810176	5	E

Sender's computation

Receiver's computation

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

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.