Encryption Details

COMP620

"Encryption … is a powerful defensive weapon for free people. It offers a technical guarantee of privacy, regardless of who is running the government… It's hard to think of a more powerful, less dangerous tool for liberty."

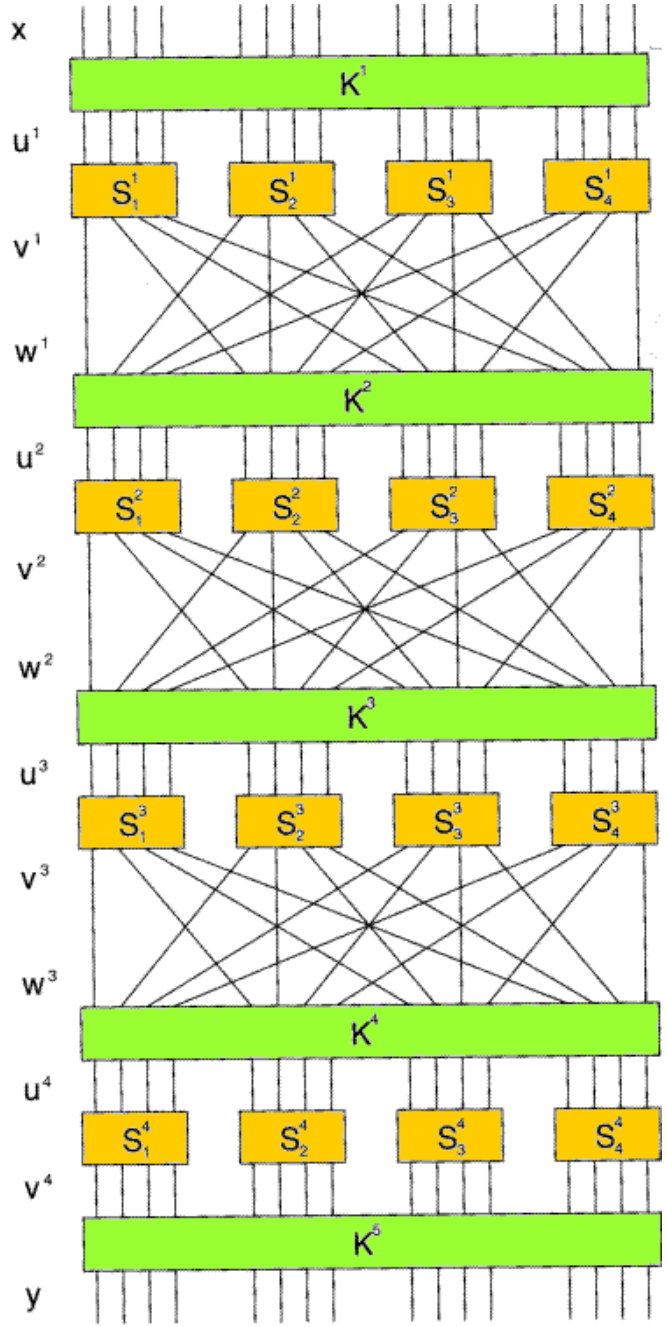
Esther Dyson

Goals for Today

- Understand how some of the most common encryption algorithms operate
- Learn about some new potential encryption systems

Substitution Permutation Ciphers

- A Substitution Permutation encryption algorithm typically involves three phases, which are often repeated
- **Substitution** – the substitution of a bit pattern with another
- **Permutation** – the rearrangement of the bits
- **Exclusive OR** with a key



Substitution Permutation stages

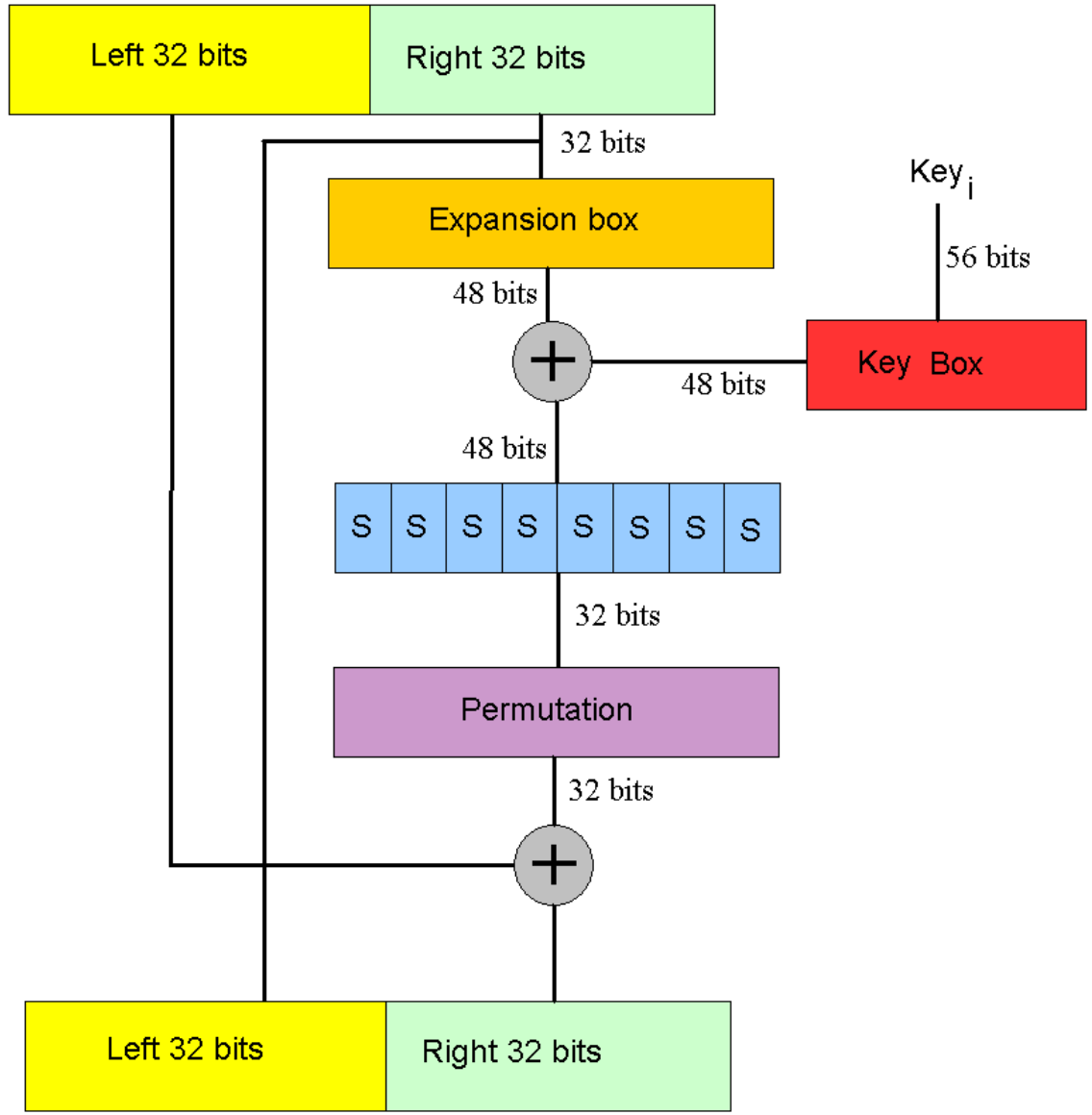
- The K box XORs the input with the key for that round
- The S box performs a substitution

diagram from "Cryptography Theory and Practice", 3rd ed. by Douglas Stinson

DES Algorithm

- DES is a substitution permutation cipher
- There are 16 stages
- The data is split into the left and right half. Each 32 bit half is handled differently
- The 56 bit key is divided into two 28 bit halves which are used to create unique 48 bit keys for each stage

DES Stage


Left 32 bits
Right 32 bits
32 bits
Expansion box
Key i
56 bits
48 bits
Key Box
48 bits
48 bits
S S S S S S S S
32 bits
Permutation
32 bits
Left 32 bits
Right 32 bits

DES Effectiveness

- Analysis has found few weaknesses in DES, specifically with the S boxes
- Differential cryptanalysis can find the key with 2^{47} chosen plaintext/ciphertext pairs
- The DES key is too short
- Brute force attacks can defeat DES

Triple DES

- To improve security while still using DES, triple DES is usually used.

  $ciphertext = E_{Key3}(D_{Key2}(E_{Key1}(plaintext)))$

- The three keys can be:
  - All different – best security using 168 bit keys
  - Key1 = Key3, Key2 different – Good with 112 bit keys
  - All identical – same as single DES

Advanced Encryption Standard

- AES is also known as the *Rijndael* algorithm
- Selected in 2000 as the new standard after an open international competition
- Created by Belgian researchers Rijmen and Daemen
- Available world-wide royalty free
- Approved by the National Security Agency (NSA) for top secret information

AES State

- AES encrypts blocks of 128 bits
- Keys can be either 128 bits, 194 bits or 256 bits
- AES operates on a 4×4 array of bytes, termed the *state*
- The 16 bytes, b_0, b_1, … b_{15} are arranged as

$$\begin{bmatrix} b_0 & b_4 & b_8 & b_{12} \\ b_1 & b_5 & b_9 & b_{13} \\ b_2 & b_6 & b_{10} & b_{14} \\ b_3 & b_7 & b_{11} & b_{15} \end{bmatrix}$$

AES Algorithm

Initial Stage

- AddRoundKey

Each Stage

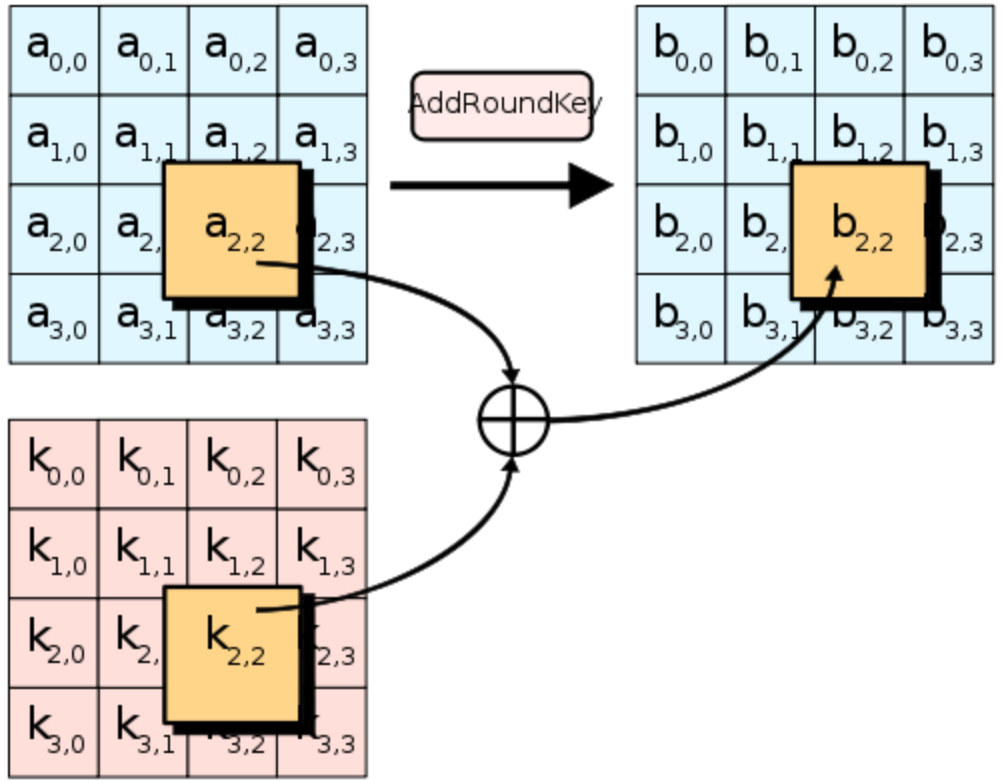
- SubBytes—a substitution step
- ShiftRows—rows are shifted cyclically
- MixColumns—each column of the state is multiplied with a fixed polynomial
- AddRoundKey—each byte is XOR with the stage key

Final Stage(no MixColumns)

- SubBytes
- ShiftRows
- AddRoundKey

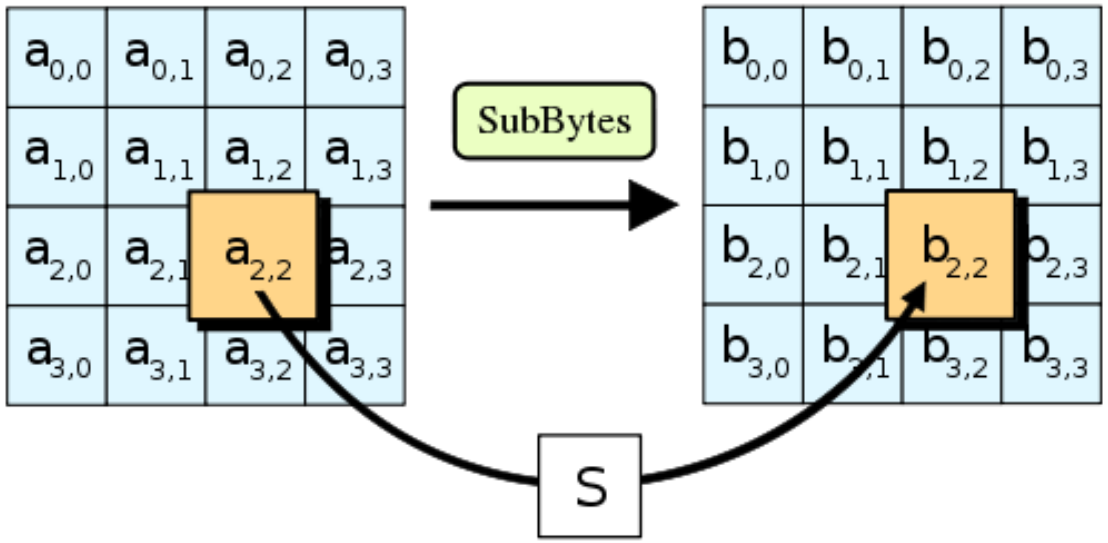
AddRoundKey Step

- Each byte of the data is XOR with the key for that stage



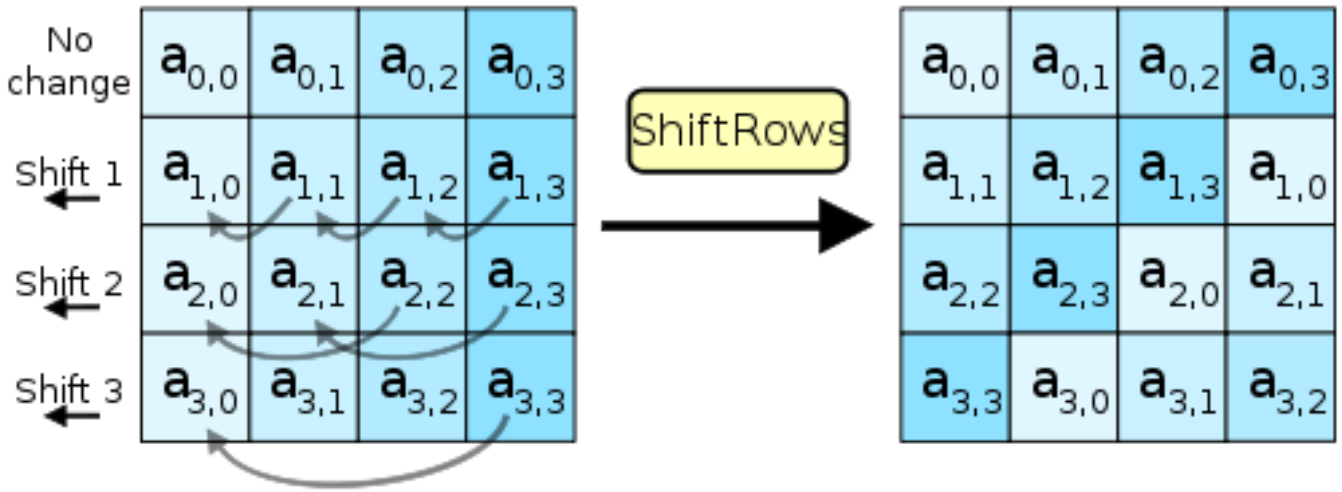
SubBytes Step

- Each byte in the array is updated using an 8 bit substitution box, the Rijndael S-box.



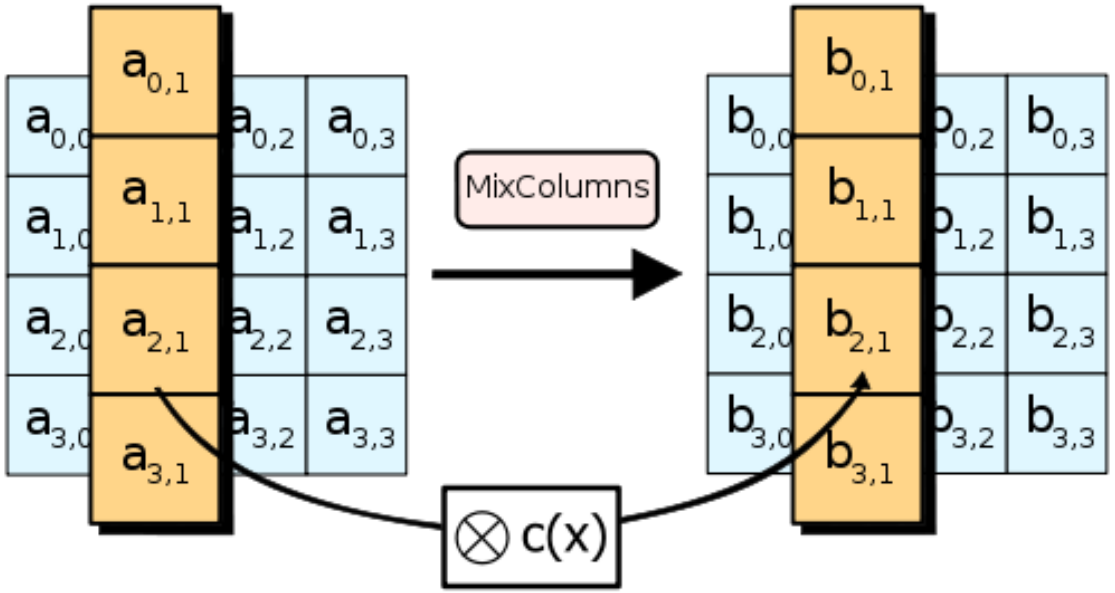
ShiftRows Step

- This cyclically shifts the bytes in each row by a certain offset.



MixColumns Step

- Each column is multiplied by a fixed polynomial. All four input bytes determine the four output bytes.



AES Implementation

- It is possible to speed up execution of this cipher by combining SubBytes and ShiftRows with MixColumns, and transforming them into a sequence of table lookups
- This requires 4KB of lookup tables
- A stage can now be done with 16 table lookups and 12 XOR, followed by four XOR in the AddRoundKey step
- Intel and AMD have added instructions to their processors to perform an AES stage

AES Security

- There have been no published methods that practically defeat AES better than brute force
- An attack was found that allows an attacker to defeat the encryption 4 times faster than brute force
- The Snowden documents claim the NSA is looking into techniques to defeat AES

What is one fourth of 2^{256}?

A. 2^{64}

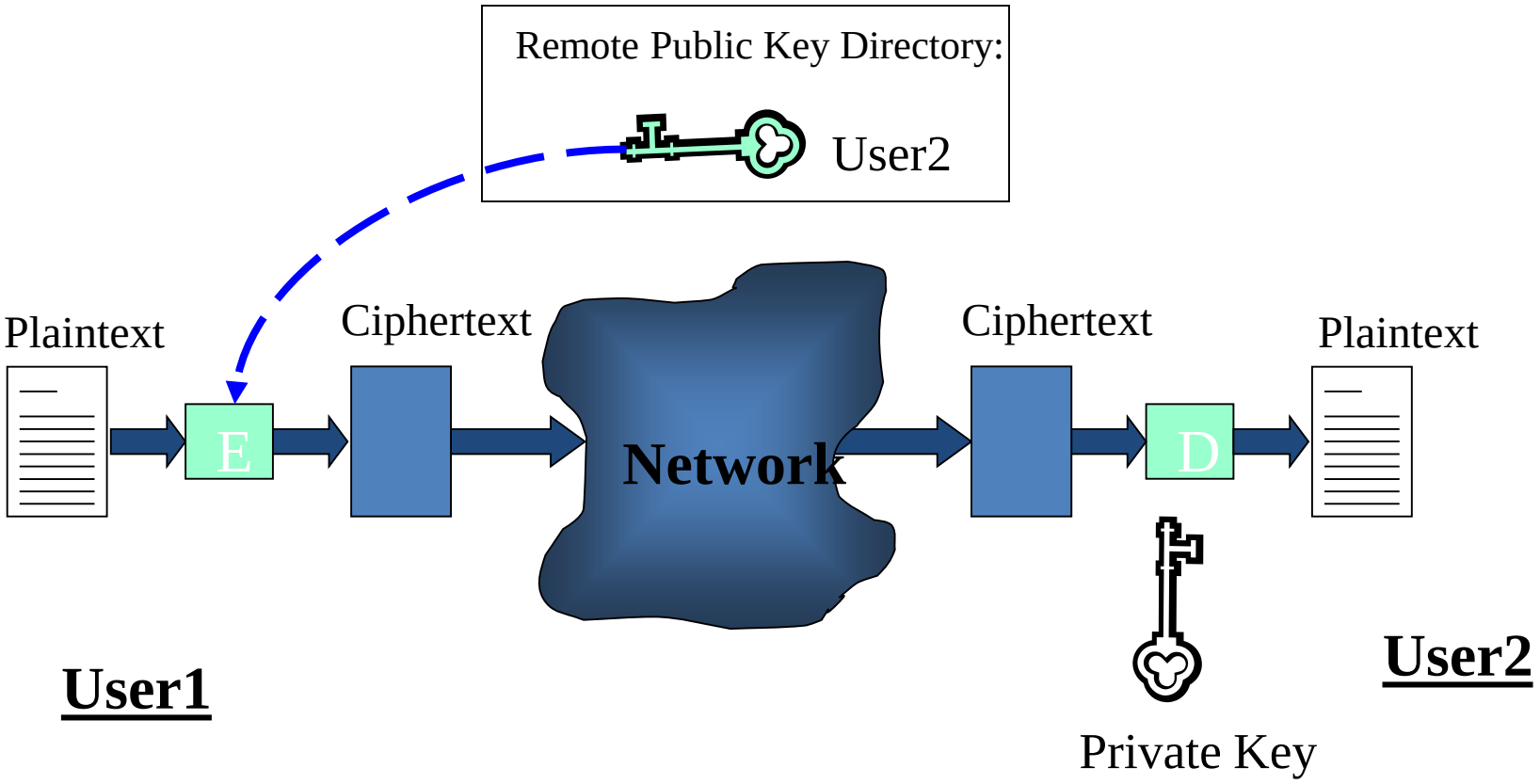
B. 0.5^{256}

C. 2^{252}

D. 2^{254}

Asymmetric Key Cryptography

- Public key different from private key
- RSA encryption is an example of Asymmetric Key Cryptography



RSA Algorithm

- RSA is an asymmetric encryption algorithm developed in 1977 by Rivest *(author of COMP785 text)*, Shamir and Adleman
- Secretly developed by Clifford Cocks in 1973
- The most commonly used public key algorithm
- Now in the public domain

RSA Encryption

- The sender encrypts message m as

$$c = m^e \bmod n$$

- ***c*** is sent to the receiver
- The receiver computes

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

RSA Algorithm

- Select secret primes ***p*** and ***q*** and form ***n*** = ***p*******q***
- The totient $\Phi(\boldsymbol{n})$ = (***p***-1)*(***q***-1)
- Choose ***e*** with *gcd(**e**,* $\Phi(\boldsymbol{n})$ *) = 1*
- Compute ***d*** such that ***d*******e*** = *1 mod* $\Phi(\boldsymbol{n})$
- The public key is ***n*** and ***e***
- The secret key is ***d***

RSA Example

- Let p = 101 and q = 113
- $n = 11413 = p * q$
- $\Phi(n) = (p-1)*(q-1) = 11200$
- Choose e = 3533 (gcd of e and $\Phi(\mathbf{n})$ is 1)
- $d = e^{-1} \bmod 11200 = 6597$
- Assume m = 9726
- Encryption: $c = 9726^{3533} \bmod 11413 = 5761$
- Decryption: $m = 5761^{6597} \bmod 11413 = 9726$

Sizes

- Typically large values of p and q are selected to make it difficult to factor.
- Therefore n is very large, $\log_2 n$ bits in length
- The pieces of plain text to encrypt, m, must be $0 < m < n$
- The cipher text can be up to $\log_2 n$ bits long

Performance

Which is the correct order from fastest to slowest

A. RSA, DES, AES
B. RSA, AES, DES
C. AES, DES, RSA
D. DES, AES, RSA
E. all about the same

Another RSA Example

- Let p = 885320963 and q = 238855417
- n = 211463707796206571
- Choose e = 9007
- $d = e^{-1} \bmod 11200$ = 116402471153538991
- Assume **m** = 30120
- c = 113535859035722866
- Note that c is much larger than m
- It requires more space to store c than m

RSA Key Concern

- For $n = p*q$, if we know the first or last $(\log_2 \mathbf{n})/4$ digits of m, we can efficiently factor n
- If p has 100 digits, then if we know the first or last 50 bits of p or q, we can factor n
- Imagine we select **p** and **q** by picking a random 50 bit number and multiplying it by 2^{50} to get 100 bits. We then add 1 and test for primality until we get a prime
- We can guess the lowest 50 bits of the number