CS 465 Computer Security

RSA

Daniel Zappala, adapted from Kent Seamons
Fall 2018

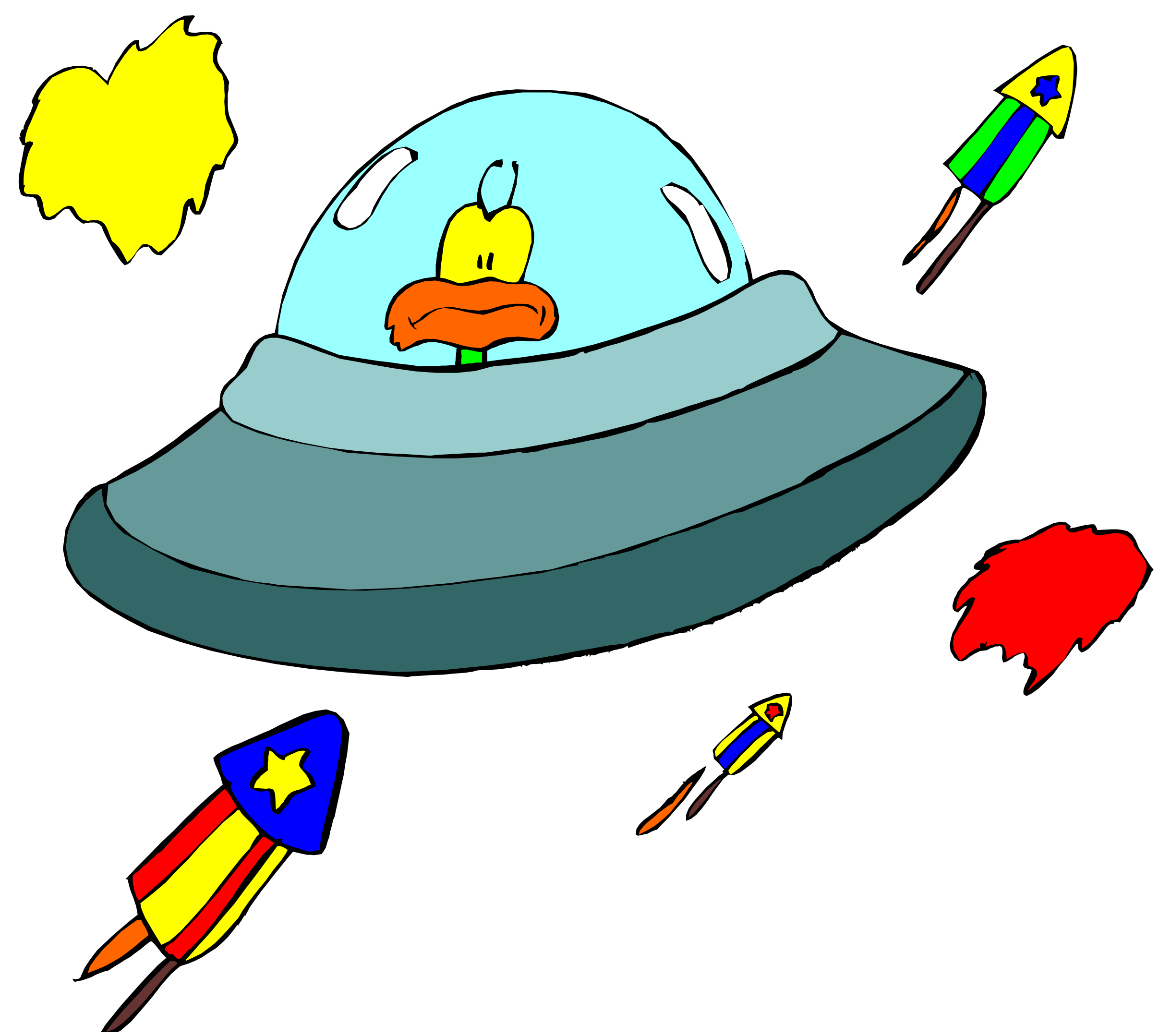
Recap

- Number theory
    - What is a prime number?
    - What is prime factorization?
    - What is a GCD?
    - What does relatively prime mean? What does co-prime mean?
    - What does congruence mean?
    - What is the additive inverse of 13 % 17 ?
    - What is the multiplicative inverse of 7 % 8 ?

Recap: Diffie-Hellman

- You’re trapped in your spaceship
- You have enough energy to send a single message to your HQ
- You have:
  - HQ’s public DH values
    - g=5, p = 875498279345…

      g^a = 32477230478…
  - Your AES implementation from Labs #1 & 2
  - An arbitrary precision calculator
- How can you construct your message so that it will be safe from eavesdroppers?

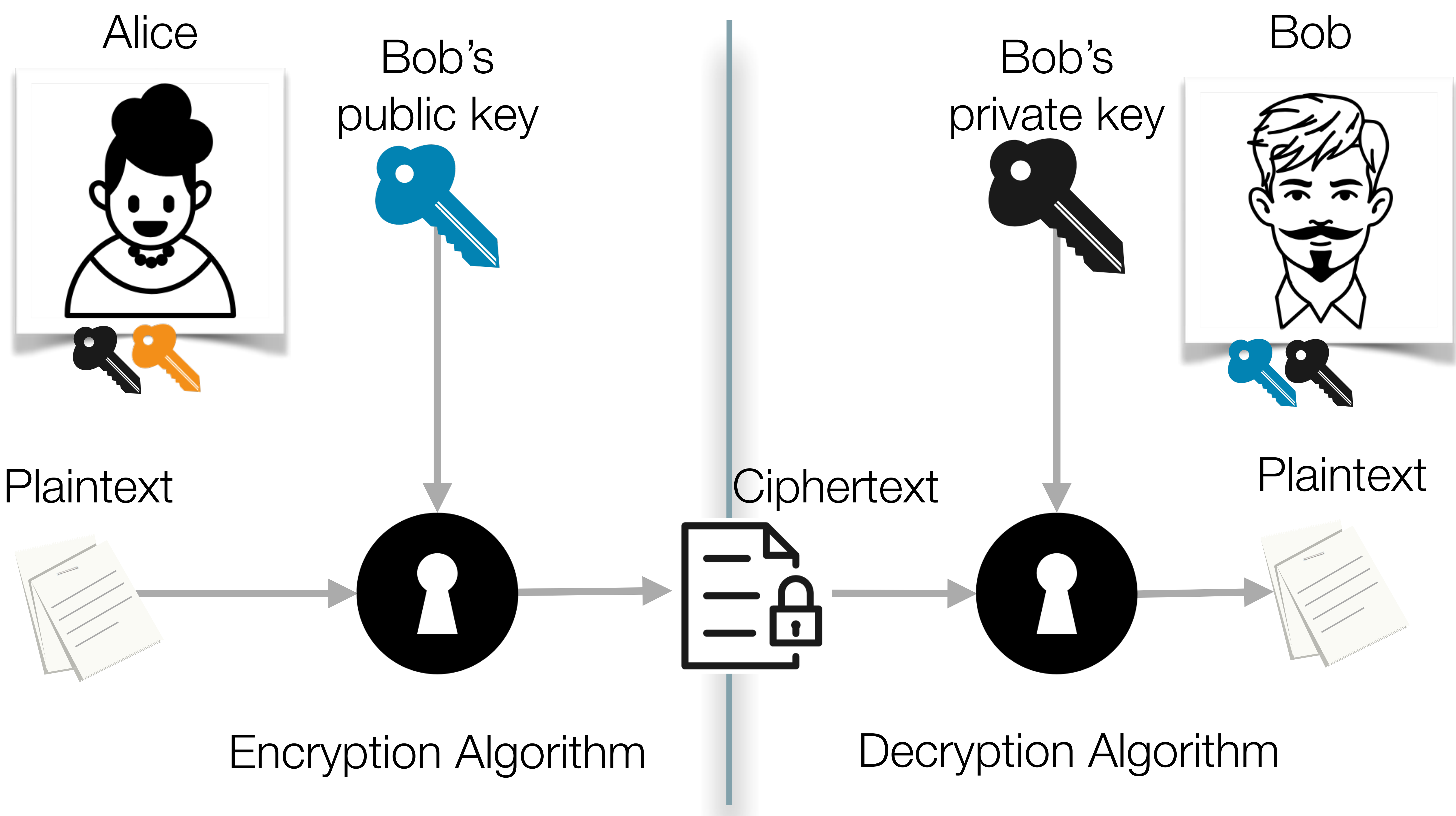


Public Key Encryption

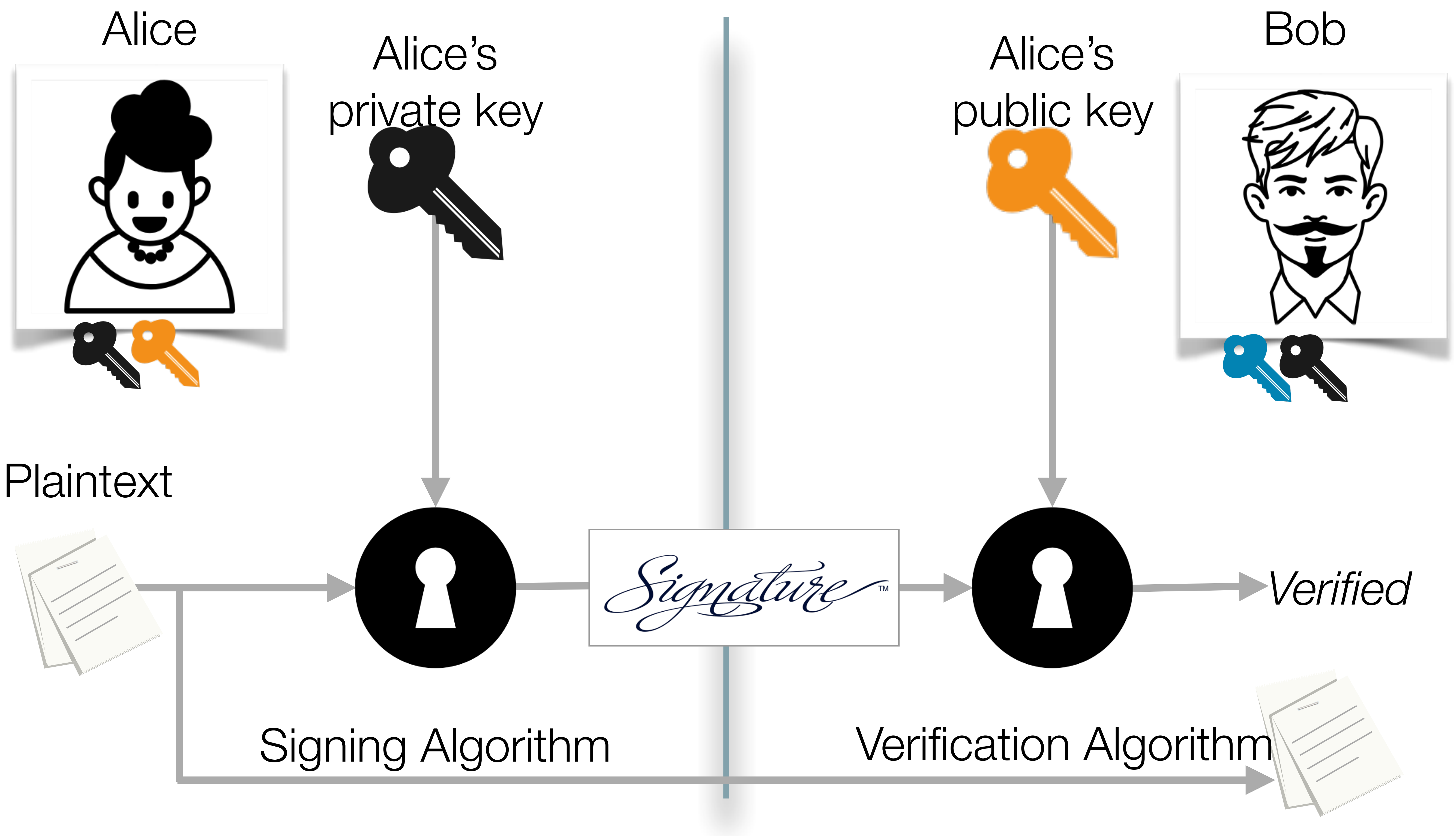
Public Key Terminology

- Public Key
- Private Key
- Digital Signature
- You encrypt with a public key, and you decrypt with a private key
- You sign with a private key, and you verify with a public key

Public Key Encryption Model



Public Key Digital Signature Model



History of RSA

- Invented in 1977 by Ron Rivest, Adi Shamir, Leonard Adleman
- Patented until 2000
- It’s withstood years of extensive cryptanalysis
  - Suggests a level of confidence in the algorithm
  - Example of successful attacks against implementations
    - Side channel attacks
    - Poor random number generators

Textbook RSA

- m = message
- c = ciphertext
- e = public exponent
- d = private exponent
- n = modulus

- RSA Encryption: $c = m^e \% n$

- RSA Decryption: $m = c^d \% n$

Why Public Key Encryption Works

The Math Behind RSA

- RSA encrypt/decrypt operations are simple
- The math to get to the point where these operations work is not so simple (at first)
    - Fermat’s little theorem
    - Euler’s generalization of Fermat’s little theorem

Fermat's Little Theorem

- If
  - p is prime
  - a is not divisible by p
- Then Fermat's theorem states
  - $a^{p-1} \equiv 1 \pmod{p}$
- This serves as the basis for
  - Fermat's primality test
  - Euler's generalization

Pierre de Fermat
(1601-1655)

Euler's Generalization of Fermat's Little Theorem

- Euler said
  - $a^{\varphi(n)} \equiv 1 \pmod{n}$

n doesn't need to be prime
a must still be co-prime to n

- $\varphi(n)$
  - Euler's totient function
  - The number of values less than n which are relatively prime to n
  - Multiplicative group of integers (Z_n^*)
- RSA is interested in values of n that are the product of two large prime numbers p and q

Leonhard Euler
(1707-1783)

Computing $\varphi(n)$ in RSA

- When p * q = n, and p and q are prime, what is $\varphi(n)$?
    - $\varphi(n)$ = the number of integers between 0 and n that are co-prime to n
- Proof (When p * q = n)
    - Observations
      1) there are p-1 multiples of q between 1 and n
      2) there are q-1 multiples of p between 1 and n
      <u>These multiples are not co-prime to n</u>

      Definition:
      $\varphi(n)$ = # of values between 0 and n <u>minus</u>
      # of values between 0 and n not relatively prime to n

$$\begin{aligned}\varphi(n) &= [\,n - 1] - [(p-1) + (q-1)] \\ &= [pq - 1] - (p-1) - (q-1)) \\ &= pq - p - q + 1 \\ &= (p-1)(q-1)\end{aligned}$$

(p-1)(q-1)

Why not?

RSA

- Euler said: $a^{\varphi(n)} \equiv 1 \pmod{n}$
  - $m^{(p-1)(q-1)} \equiv 1 \pmod{n}$
- Notice: $m^{(p-1)(q-1)} * m \equiv m^{(p-1)(q-1)+1} \equiv m \pmod{n}$
  - $m^{\varphi(n)+1} \equiv m \pmod{n}$
- Let $e*d = k*\varphi(n) + 1$
  - Then $e*d \equiv 1 \pmod{\varphi(n)}$
  - Therefore $m^{ed} \equiv m^{k*\varphi(n)+1} \equiv m^{\varphi(n)} * m^{\varphi(n)} * \ldots * m \equiv m \pmod{n}$
- RSA encryption: $m^e = c \pmod{n}$
- RSA decryption: $c^d = m \pmod{n}$

Why is RSA secure?

If you could factor n into p and q, then you know $\varphi(n)=(p-1)(q-1)$, and now you can easily calculate d (e is public).

This is called the “trap door” in RSA. Knowing the prime factors is what makes it easy to decrypt.

It’s hard to factor large primes and hard to find d without knowing the factorization.

How To Use Public Key Encryption

Steps for RSA Encryption

- Select p, q (large prime numbers)
- $n=p*q$
- $\varphi(n) = (p-1)(q-1)$
- Select integer e where e is relatively prime to $\varphi(n)$
    - Common values for e are 3 and 65537. Why?
- Calculate d, where $d*e = 1 \pmod{\varphi(n)}$
- Public key is KU = {e, n}
- Private key is KR = {d, n}

RSA Usage

- Given $m^e = c \pmod{n}$ and $c^d = m \pmod{n}$
  - What restrictions should be placed on m?
- For bulk encryption (files, emails, web pages, etc)
  - Never, never, never encrypt data directly using RSA — inefficient and insecure
  - Always use symmetric encryption for data, and use RSA to encrypt the symmetric key, using a secure padding scheme
- Digital signatures
  - Do not sign the entire document — too slow
  - Sign (encrypt) a hash of the document using the private key

How To Calculate RSA Values

How do we get p, q, e, and d?

- What is p?
  - How do we get it?
- What is q?
  - How do we get it?
- What is e?
  - How do we get it?
  - What is the relationship of e and (p-1)(q-1)?
- What is d?
  - How do we get it?

Calculating d

- Goal: find d such that ed = 1 (mod $\varphi(n)$)
- Use the extended Euclidean algorithm
    - Calculates x and y such that $ax+by=gcd(a,b)$
    - Let $a=e$, $b=\varphi(n)$. $gcd(e,\varphi(n))=1$ because they are co-prime
    - Then you have: $ex+\varphi(n)y=1$
    - Take this modulo $\varphi(n)$ and you get: $ex\equiv1 \pmod{\varphi(n)}$
    - $x=d$ (if x is negative, simply add $\varphi(n)$)

Extended Euclidean Algorithm

- Let p = 5, q = 11, n = 55, e=17, and φ(n)=40
  - 17d+40k=1
  - 40=2×17+6 GCD with remainder
  - 17=2×6+5 GCD with remainder
  - 6=1×5+1 (stop at remainder 1)
- Rewrite
  - 6−1×5=1
- Substitute
  - 6−1×5=1
  - 6−1×(17−2×6)=1
  - (40−2×17)−1×(17−2×(40−2×17))=1
- Simplify
  - (−7)×17+3×40=1
  - d=−7 -> add 40 (the modulus) and get d = 33
- Public key = {17,55}
- Private key = {33,55}

Practice

```
p=5, q=11, e=3
```

Practice

```
p=5, q=11, e=3

n = p*q = 55
φ(n) = (p-1)(q-1) = 4*10 = 40


Calculate d

3*d + 40*k = 1
40 = 13*3 + 1                (no substitution steps needed)
(-13)*3 + 40 = 1
d = -13 + 40 = 27


Public Key = {3, 55}
Private Key = {27, 55}
```

An Exception!

GCD(e, φ(n)) must be 1

- Be sure to check, otherwise you need a new e
- Easy algorithm:
  - GCD(x, y) = GCD(y, x %y)   if x > y   (recursive computation)
- Example
  - GCD(40, 3) = GCD(3, 1) = 1
  - GCD(120,3) = 3!

Practice

```
p=11, q=13, 2 < e <= 8
```

Practice

```
p=11, q=13, 2 < e <= 8

n = p*q = 143
φ(n) = (p-1)(q-1) = 10*12 = 120



Calculate d

GCD (φ(n),e) = GCD(120,3)= 3, GCD(120,5) = 5, GCD(120,7) = GCD(17,1) = 1
7*d + 120*k = 1
120 = 17*7 + 1                (no substitution steps needed)
(-17)*7 + 120*1 = 1


d = -17 + 120 = 103

Public Key = {7, 143}
Private Key = {103, 143}
```

More Practice

Practice

```
p=5, q=13, e=5
```

Practice

```
p=5, q=13, e=5

n = p*q = 65
φ(n) = (p-1)(q-1) = 4*12 = 48

Calculate d

GCD (φ(n),e) = GCD (48,5) = GCD(5,3) = GCD(3,2) = 1
5*d + 48*k = 1              (notice how these match the substitution steps)
48 = 9*5 + 3
5 = 1*3 + 2
3 = 1*2 + 1
(substitute)
3 - 1*2 = 1
3 - 1*(5 - 1*3) = 1
48 - 9*5 - 1*(5 - 1*(48 - 9*5)) = 1
48 - 9*5 - 1*5 + 1*48 - 9*5 = 1
2*48 - 19*5 = 1
d = -19 + 48 = 29

Public Key = {5, 65}
Private Key = {29, 65}
```

Practice

```
p=17, q=11, e=7
```

Practice

```
p=17, q=11, e=7

n = p*q = 187
φ(n) = (p-1)(q-1) = 16*10 = 160

Calculate d

GCD (φ(n), e) = GCD (160, 7) = GCD (7,6) = 1
7*d + 160*k = 1
160 = 22*7 + 6
7 = 1*6 + 1
(substitute)
7 - 1*6 = 1
7 - 1*(160 - 22*7) = 1
7 - 160 + 22*7 = 1
23*7 - 1*160 = 1
d = 23

Public Key = {7, 187}
Private Key = {23, 187}
```