Security - Cryptography

Note Title 2/10/2011

Announcements

- Lab will be posted after class.
  Required reading for Tuesday.

Cryptography is old:

Caesar Cyphers:

plain →

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Q W E R T Y U I O P A S D F G H J K L Z X C V B N M

cipher →

plain: EXAMPLE → TUQDHST

cipher: KQFRGD → RANDOM

How would you attack the Caesar cipher?

- Frequency Analysis
- Brute Force
- Physical means
- Pattern matching

Today

1. Symmetric encryption
   DES, 3DES, AES

2. Assymmetric encryption ← 1976
   public key cryptography
   RSA
   elliptic curve cryptography

DES

- Adopted in 1977 by what is now NIST (National Institute of Standards & Tech.)
- Encrypts 64 bits of plaintext using a key of 56 bits

Essential element:

XOR with a secret key

In July 1998, DES was officially broken by a machine the EFF built for under $250,000. (over 6-8 months)

They subsequently published details of their approach.

→ 56 bits

2^{56} keys

Note: Not as simple as you'd think.

What do we routinely do to everything before sending it somewhere?

files get compressed.

- Essentially, the only way to attack DES is, brute force.

3DES:

- Last ditch effort to save DES.
- Repeat DES 3 times with different keys - total of 168 bits.

Actually fairly secure.

Main drawback:

- Slow

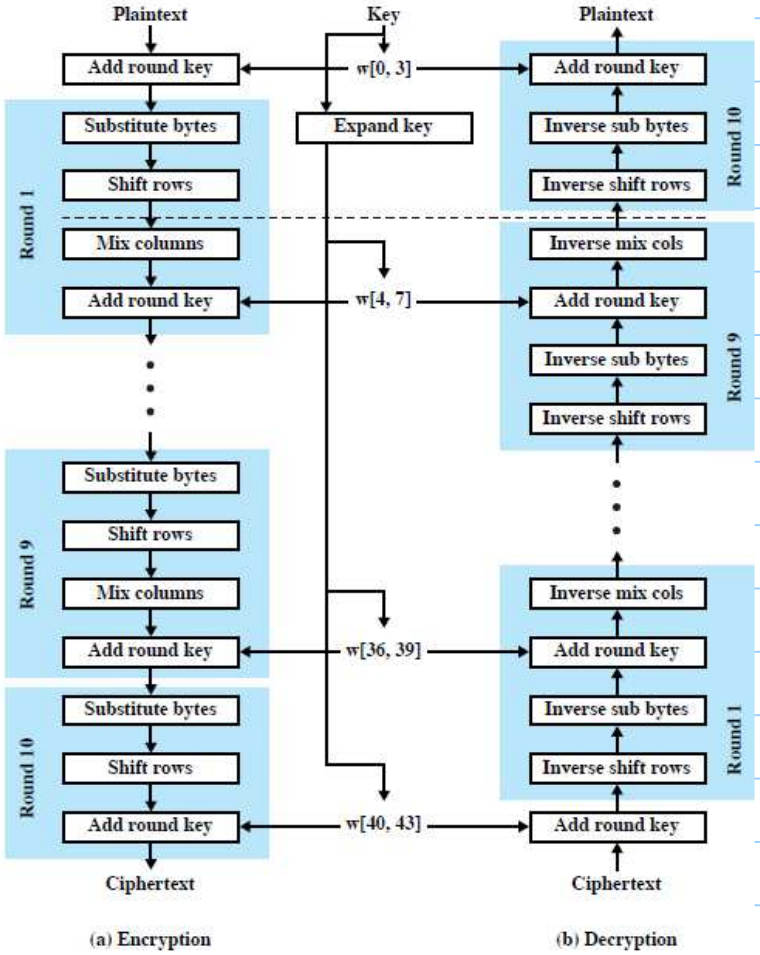
AES: Advanced Encryption Standard

- block length is 128 bits, & keys are (128, 192, or 256 bits.)

Essentially, 4 operations performed many times:

1) substitute bytes
2) Permute
3) Mix Columns
4) Add round key (an XOR with part of the key which changes each round)

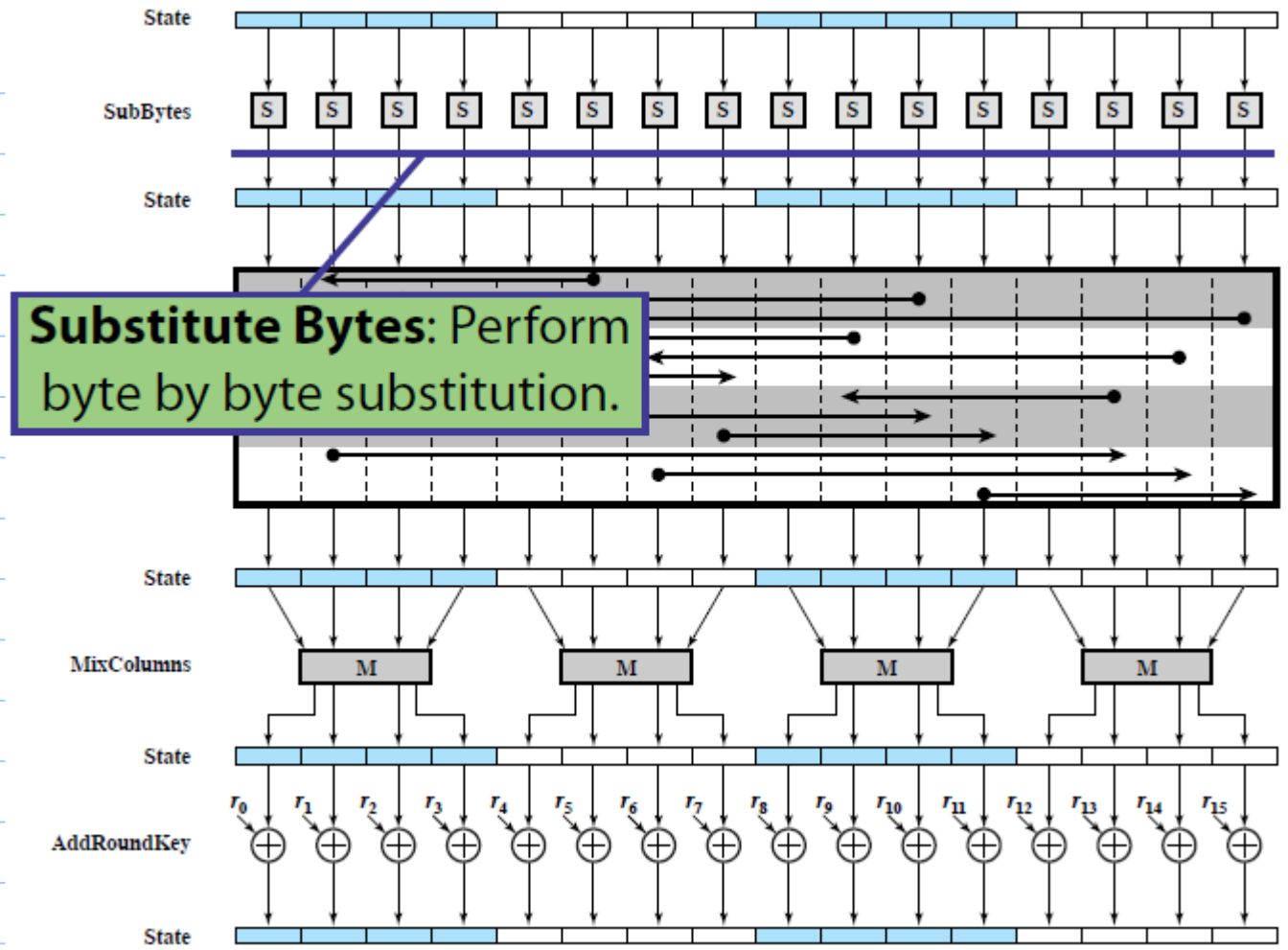
The algorithm:

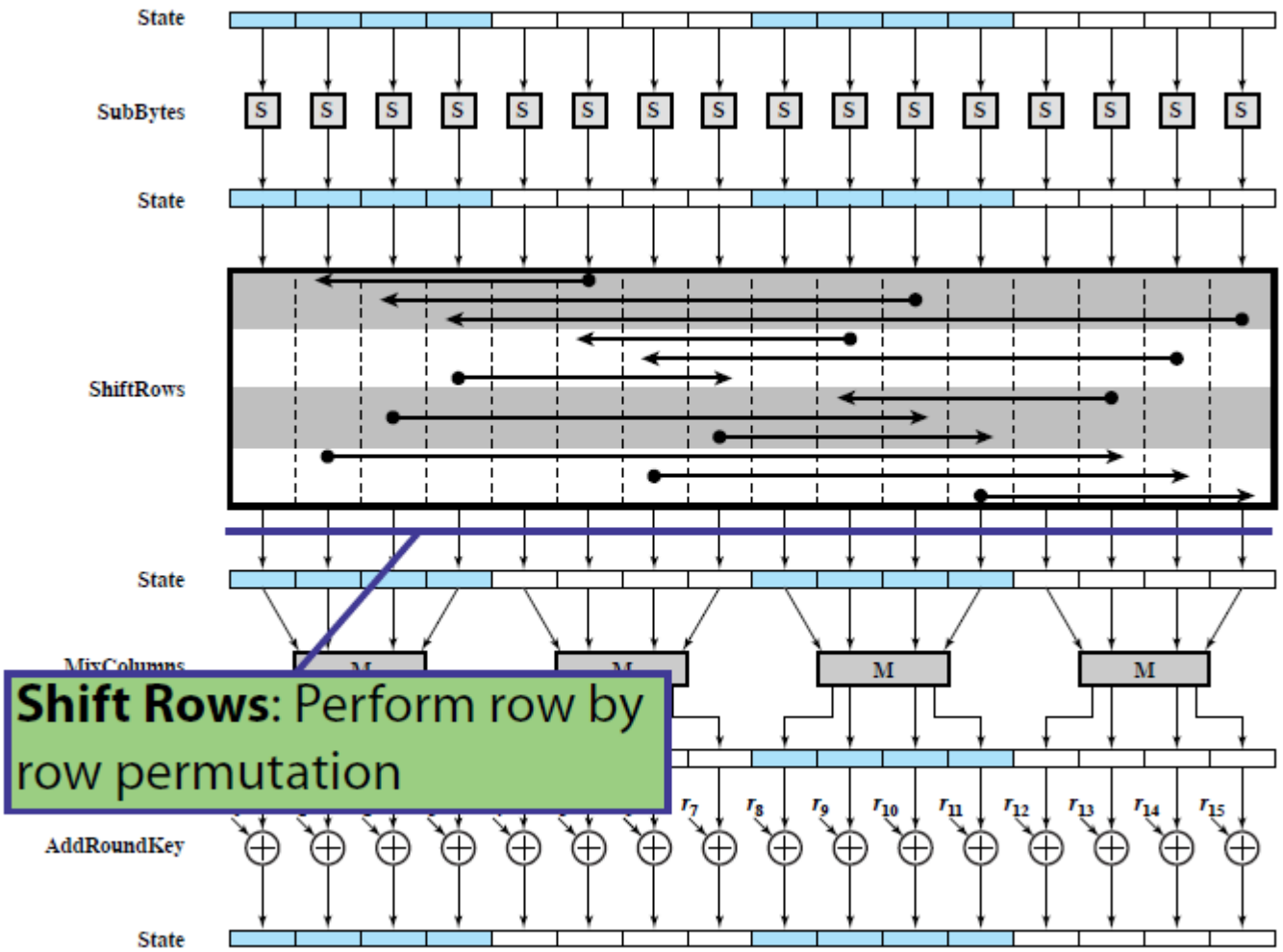


Practical Issues

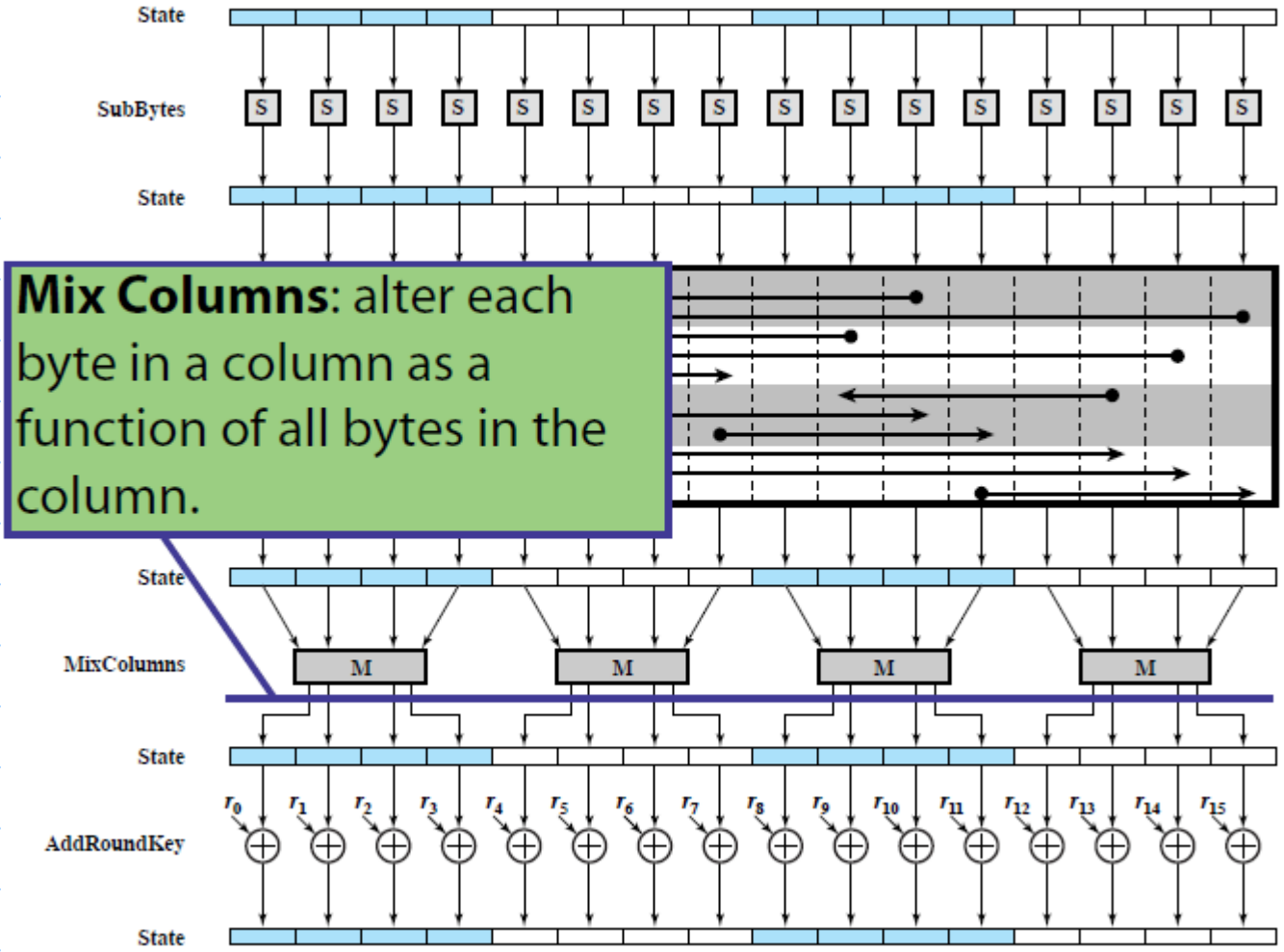
- less secure for lengthy messages, particularly if part of plaintext is known which repeats

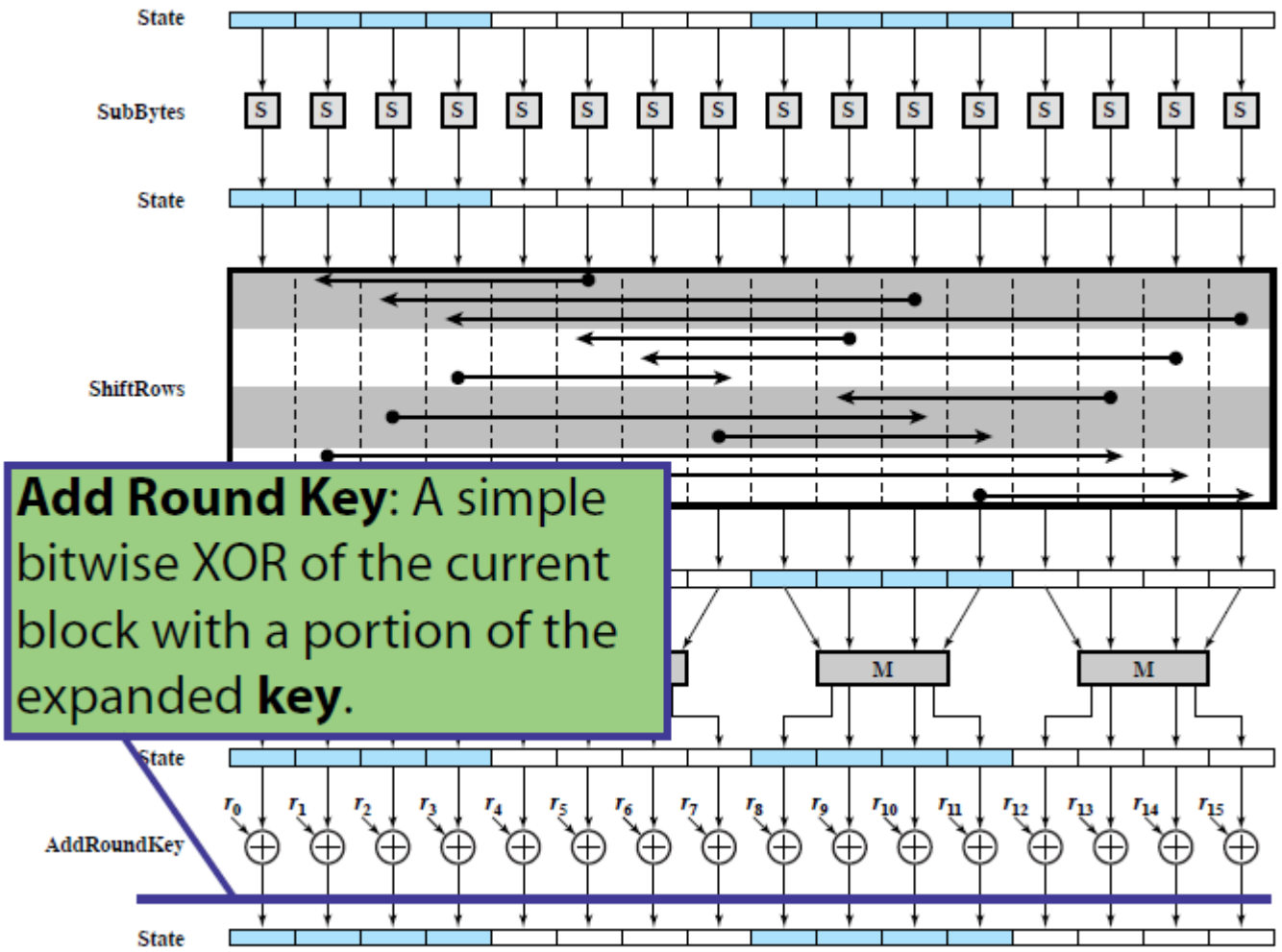
- Hardware is getting faster!


State
SubBytes
S
State
Substitute Bytes: Perform byte by byte substitution.
State
MixColumns
M
State
AddRoundKey
r_0 r_1 r_2 r_3 r_4 r_5 r_6 r_7 r_8 r_9 r_{10} r_{11} r_{12} r_{13} r_{14} r_{15}
State



Shift Rows: Perform row by row permutation


State
SubBytes
State
Mix Columns: alter each byte in a column as a function of all bytes in the column.
State
MixColumns
M
M
M
M
State
AddRoundKey
State


State
SubBytes
State
ShiftRows
M
M
State
AddRoundKey
State
Add Round Key: A simple bitwise XOR of the current block with a portion of the expanded key.

An Application: Message Authentication

Suppose we have a message where we need to know identity of sender.

We could use encryption, but it is slow.

Also - message could still be compromised.

Authentication Tags: Message Authentication Code (MAC)

Use DES:

Take message M plus secret key K_{AB}, which A & B share.

Compute $MAC_M = F(K_{AB}, M)$ + M

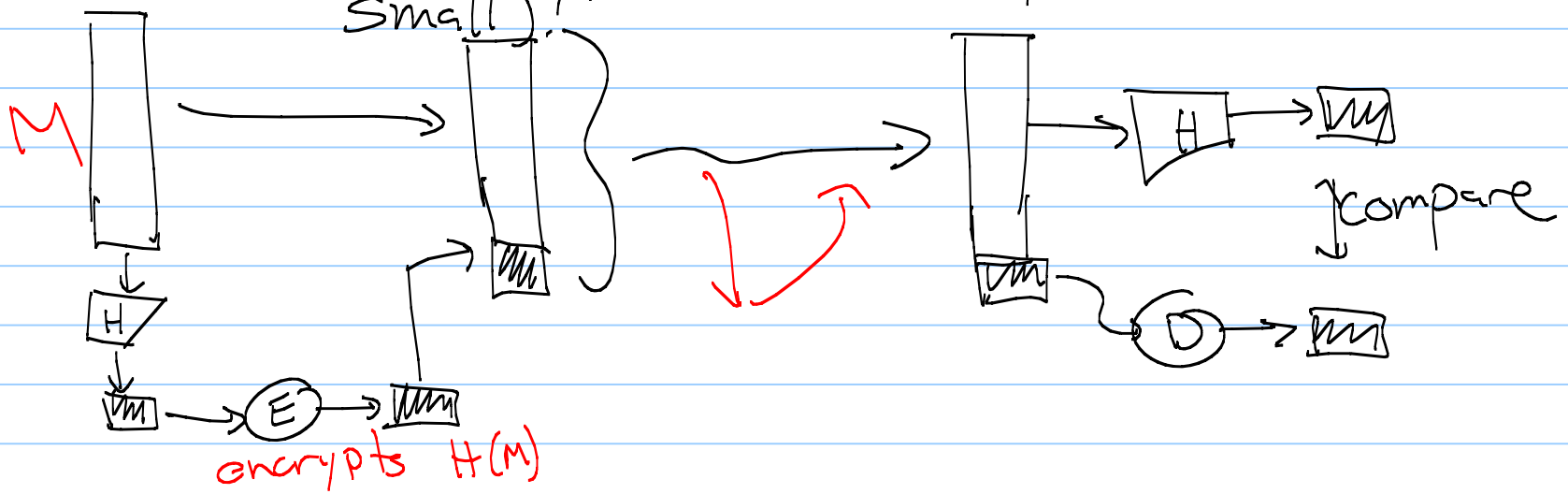
($F(K_{AB}, M)$ gets attached to message)

(Similar to encryption, so still vulnerable.)

One Way Hashing

A & B pre-arrange a hash function.

Hash M down to a small value, then encrypt hash output (which is small).



Can avoid encryption using keyed hash MAC, which incorporates a secret key into the hash function.

Essentially just hashes M + key & attaches it.

An attacker would not know key, & so couldn't impersonate sender.

(Even if hash function is public, can't impersonate the sender.)

Public Key

First revolution in cryptography in thousands of years!

"New directions in Cryptography"
by Diffie & Hellman, 1976

At its base, it is taking logarithms!

choose X, secret key.

Set $Y = \alpha^X \bmod q$, for $1 \le X \le q-1$

(want q prime, or close to it)

So $X = \log_\alpha Y \bmod q$

X is secret key

Y is public key

Publish: Y, α, q

$4 \bmod 3 = 1$

Now: Alice selects X_a & keeps it secret, but publicizes:

$$Y_a = \alpha^{X_a} \bmod q$$

(so Y_a, α, & q are known)

- Computing Y_a given X_a is easy
- Computing X_a given Y_a is hard (discrete log problem)

Bob wants to talk to Alice, but needs a secret key.

Now: Bob wants to send a message.

(He has Y_a, α, and q, as well as X_b and Y_b.)

He uses a secret key:

$$K_{ab} = (Y_a)^{X_b} = (\alpha^{X_a})^{X_b} \equiv \alpha^{X_a X_b} \mod q$$

X_b: Bob's private

Y_a: Alice's public

Alice has: X_a, Y_a, Y_b, α, and q.

She can compute:

$$K_{ba} = (Y_b)^{X_a} = (\alpha^{X_b})^{X_a} = \alpha^{X_b \cdot X_a} \bmod q$$

Y_b: Bob's public key

X_a: her private key

K's are equal.

Eve (the eavesdropper) has only Y_a, Y_b, α, and q.

Hard to get K_{ab} from this!

$$Y_a Y_b \bmod q$$

$$= (\alpha^{X_a})(\alpha^{X_b}) \bmod q = \alpha^{(X_a + X_b)} \bmod q$$

$$(Y^a)^{Y_b} = (\alpha^{X_a})^{(\alpha^{X_b})} \bmod q \neq K_{ab}$$

$$= \alpha^{\alpha^{X_b} \cdot X_a} \bmod q$$

2 mod 11

So getting a secret key set up is now pretty easy!

Note: This all depends on the fact that it isn't "easy" to compute X given Y, where

$$Y = \alpha^X \bmod q$$

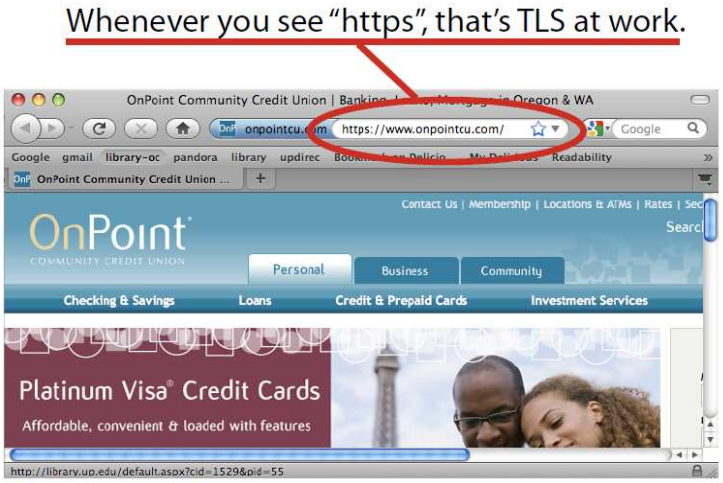
(so $X = \log Y \bmod q$)

There are ways to attack this besides brute force.

RSA

- 1977 by Rivest, Shamir, & Adleman
- Most widely used public key cryptosystem.

- Used in transport Security layer (TSL)



Uses more number theory:

- Choose 2 prime numbers p & q. (secret)
- Set $n = pq$. ← public
- Compute $\phi(n) = \phi(pq)$

Here, $\phi(n)$ = # of numbers $< n$ which are relatively prime to n. ← no common divisors

What is $\phi(p)$? $= p - 1$

Neat fact:

$$\phi(n) = \phi(pq) = \phi(p)\phi(q) = (p-1)(q-1)$$

- Find e with $1 < e < \phi(n)$ where e & $\phi(n)$ are relatively prime.
  - e is public key
- Compute d such that $d \cdot e = 1 \bmod \phi(n)$ (inverse mod $\phi(n)$)

$e = 2 \bmod 11$

what is d? $d = 6$, since $6 \cdot 2 = 12 = 1 \bmod 11$

Bob has e and n
(plus m, his message).

Sends $c = m^e \pmod{n}$

↑ ciphertext

Sends c to Alice

Alice knows d, the private key.

So she computes

$$c^d = m^{ed} \bmod n$$

$$= m^1 \bmod n$$

Key fact: exponents mod n can be taken mod $\phi(n)$

Without d, this isn't easy!

Bad guy: c, n, e

difficulty of factoring

Elliptic Curve Cryptography

- Same type of operations as RSA, but over different groups.

- Main advantage: smaller keys are more secure.

- Disadvantage: cost