Lecture 9

Public Key Cryptography

1

Key pre-distribution: Diffie-Hellman

"New Directions in Cryptography" 1976

System - wide parameters :

p - large prime,

a - generator in Z^*_p

Alice's secret: v, public: $y_a = a^v \bmod p$

Bob's secret: w, public: $y_b = a^w \bmod p$

Alice has: $y_b = a^w \bmod p$

Bob has: $y_a = a^v \bmod p$

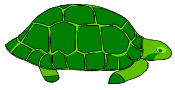
$K_{ab} = (y_b)^v \bmod p$

=

$K_{ba} = (y_a)^w \bmod p$

2

Public Key pre-distribution: Diffie-Hellman



Alice computes K_{ab}

Bob computes $K_{ab}=K_{ba}$

Secure communication with K_{ab}

Eve knows: p, a, y_a and y_b

3

Public Key pre-distribution: Diffie-Hellman

Diffie - Hellman Problem:

p - *large prime,* a - *generator in* Z^*_p

Given:

$y_a = a^v \bmod p$ *and* $y_b = a^w \bmod p$

FIND: $a^{vw} \bmod p$

Discrete Log Problem:

Given:

$y_a = a^v \bmod p$

FIND: v

4

Public Key pre-distribution: Diffie-Hellman

Decision DH Problem:

p - large prime, a - generator

Given :

$y_a = a^v \bmod p,\ \ y_b = a^w \bmod p$

Distinguish :

$K_{ab} = a^{vw} \bmod p$

from a random number!

- DH Assumption: DH problem is HARD (not P)
- DL Assumption: DL problem is HARD (not P)
- DDH Assumption: solving DDH problem is HARD (not P)

Interactive (Public) Key Exchange: Diffie-Hellman

Choose random v

Compute $K_{ab} = (y_b)^v \bmod p$

$y_a = a^v \bmod p$

$y_b = a^w \bmod p$

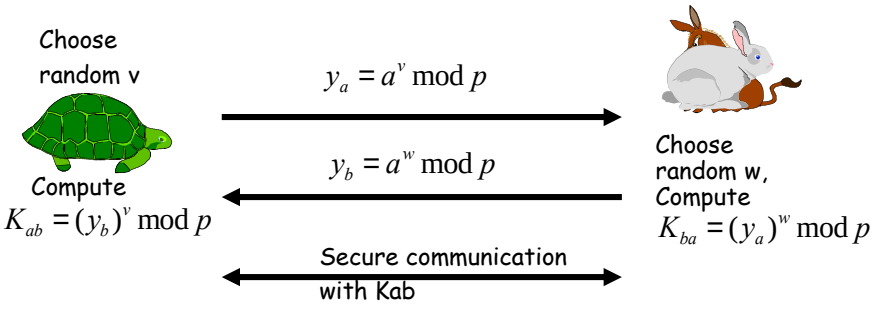
Secure communication with Kab

Choose random w, Compute $K_{ba} = (y_a)^w \bmod p$

Eve is passive…

The rabbit-in-the-middle-attack

(assume Eve is an active adversary!)



RSA (1976-8)

$Let\ n = pq \quad where \quad p,q\ -\ large\ primes$

$e,d \in_R Z_n \quad and \quad ed \equiv 1 \bmod \Phi(n)$

$where: \quad \Phi(n) = (p-1)(q-1) = pq - p - q + 1$

$Secrets: p,q,d$

$Publics: n,e$

$Encryption: message = m < n$

$E(x) = y = m^e \bmod n$

$Decryption: ciphertext = y$

$D(y) = x' = y^d \bmod n$

Why does it all work?

$x \hat{I} \; Z_n^*$

$x^{ed} = x^{1 \bmod F(n)} \bmod n =$

$x^{c*F(n)+1} \bmod n = x$

But, recall that:

$g^{F(n)} = 1 \bmod n$ (Lagrange)

9

How does it all work?

Example: p=5 q=7 n=35 (p-1)(q-1)=24=3*2^3

pick e=11, d=11

x=2, E(x)=2048 mod 35 =18=y

y=18, D(y)=6.426841007923e+13 mod 35 = 2

Example: p=17 q=13 n=221 (p-1)(q-1)=192=3^4*2

pick e=5, d=77 Can we pick 16? 9? 27? 185?

x=5, E(x)=3125 mod 221 = 31

D(y)=31^{77}=

6.836761427754420001963955995 58e+114 mod 221 = 5

10

Why is it secure?

Conjecture: breaking RSA is *polynomially equivalent* to factoring n. Recall that n is very, very large!

Why: n has unique factors p, q

Given p and q, computing (p-1)(q-1) is easy:

$$ed \equiv 1 \bmod \phi(n)$$

Use extended Euclidian!

11

Exponentiation Costs

- Integer multiplication -- $O(b^2)$ where b is bitsize of base m
- Modular reduction -- $O(b^2)$
- Thus, modular multiplication -- $O(b^2)$
- Modular exponentiation -- m^e mod n
- Naïve method: e-1 modular products -- $O(b^2 * e)$
- BUT what if e is large, (almost) as large as n?

- Let L= |e| (e.g., L=1024 for 1024-bit RSA exponent)
- We can assume b and L are close
- Square-and-multiply method works in $O(b^3)$ time… $O(b^2 * 2L)$

12

Square-and-Multiply

goal: compute $m^e \bmod n$

```
- - - - - - - - - - - - -
l = sizeof(n);
temp = 1;
for (i = l - 1; i >= 0; i - - )
{   temp* = temp;
    temp % = n;
    if (e[i])
    {    temp* = m;
         temp% = n;
    }
}
```

From left to right in e

- Example 1: e=100
- Example 2: e=10000000
- Example 3: e=11111111