Public Key Cryptography

CSG 252 Fall 2006

Riccardo Pucella

All cryptosystems until now have used a shared key between senders and receivers.

Disadvantage: need to somehow share a key before communicating.

In 1976, Diffie and Hellman proposed a scheme where this was not necessary. The idea: provide each agent with two keys, one to encrypt, one to decrypt.

- publish the key to encrypt
- keep secret the key to decrypt

Anybody can send Alice an encrypted message, only Alice can read it.

Need that the decryption key not be derivable from the encryption key!

Diffie and Hellman formalized the above as follows.

They said to get a public key cryptosystem, you need a *one-way trapdoor function*:

- one-way: a function whose inverse is hard to compute
- trapdoor: if you have a specific hint, however, you can invert the function easily.

(To encrypt, apply the one-way function; to decrypt, use the hint to invert function.)

Problem became: can you find one-way trapdoor functions?

2 most likely candidates:

1. Factorization (leads to RSA cryptosystem)
2. Discrete Log problem (leads to ElGamal cryptosystem)

But no one has ever proved that these are one-way trapdoor functions.

In fact, no one knows if a one-way function (let alone trapdoor) actually exists.

All candidates tend to involve number theory or algebra.

Number Theory Background

Recall $ax \equiv 1 \mod n$ has a solution for x if and only if $\gcd(a, n) = 1$ (a and n are relatively prime).

$\phi(n)$ is the number of elements in $\mathbb{Z}_n$ relatively prime to n.

Define $\mathbb{Z}_n^* = \{a \in \mathbb{Z}_n \mid \gcd(a, n) = 1\}$.

(Note if p is prime, $\mathbb{Z}_p^* = \{1, \ldots, p-1\}$, which is isomorphic to $\mathbb{Z}_{p-1}$.)

$\mathbb{Z}_n^*$ has some interesting properties with respect to multiplication; Define $a_1 \cdot a_2$ to be $a_1 a_2 \mod n$

- $(a_1 \cdot a_2) \cdot a_3 = a_1 \cdot (a_2 \cdot a_3)$
- $a_1 \cdot a_2 = a_2 \cdot a_1$
- $a \cdot 1 = a$
- For all $a \in \mathbb{Z}_n^*$, $\exists a^{-1} \in \mathbb{Z}_n^*$ such that $a \cdot a^{-1} = 1$

$\mathbb{Z}_n^*$ forms an *Abelian group* under multiplication.

Theorem: If $b \in \mathbb{Z}_n^*$, then $b^{\phi(n)} \equiv 1 \mod n$.

Theorem: If p is prime and $b \in \mathbb{Z}_p$, then $b^p \equiv b \mod p$.

RSA Cryptosystem

Let $n = pq$, where p and q are primes.

Let $\mathcal{P} = \mathcal{C} = \mathbb{Z}_n$.

Let $\mathcal{K} = \{(n, p, q, a, b) \mid ab \equiv 1 \mod \phi(n)\}$.

To encrypt with $k = (n, p, q, a, b)$, use $e_k(x) = x^b \mod n$.

To decrypt with $k = (n, p, q, a, b)$, use $d_k(y) = y^a \mod n$.

To use: choose p and q large primes, compute $n = pq$. Can check that $\phi(n) = (p-1)(q-1)$. Choose b such that $\gcd(b, \phi(n)) = 1$. (Just choose randomly until you find one.) Let $a \equiv b^{-1} \mod \phi(n)$. Publish n and b, keep p, q, a private.

Check: encryption and decryption are inverses. Let $x \in \mathbb{Z}_n^*$. (A different argument is needed if

$x \in \mathbb{Z}_n - \mathbb{Z}_n^*$; see Stinson.) We know $ab \equiv 1 \mod \phi(n)$. Thus, $ab = t\phi(n) + 1$ for some $t \geq 1$.

$$\begin{aligned}
d_k(e_k(x)) &= (x^b)^a \mod n \\
&\equiv x^{ab} \mod n \\
&\equiv x^{t\phi(n)+1} \mod n \\
&\equiv (x^\phi(n))^t x \mod n \\
&\equiv 1^t x \mod n \\
&\equiv x \mod n
\end{aligned}$$

Security of the cryptosystem based on the belief that e_k is a one way function.

It is a trapdoor function. What's the hint? Have factorization $n = pq$. With a, n, p, q, can recover b, because $b \equiv a^-1 \mod \phi(n)$.

For this to be secure, need to have n be hard to factor into p and q; thus, need p and q to be large enough (in practice, larger than 512 bits).

Mmm: how do you find primes of size greater than 512 bits? Best approach: generate randomly and test primality. Change of finding a prime: $1/\ln 2^{512} \approx 1/355$. Lots of algorithms for testing primality. Cf Stinson §5.4.

Attacks against RSA:

1. Factoring — huge literature. Cf Stinson §5.6.
2. Compute $\phi(n)$ directly.

   No easier than factoring: if you have n and $\phi(n)$, it is almost trivial to get a factorization. You have the equations:

   $$\begin{aligned}
   n &= pq \\
   \phi(n) &= (p-1)(q-1)
   \end{aligned}$$

   Putting $q = n/p$, we get $\phi(n) = (p-1)(n/p-1)$, that is, $p^2 - (n - \phi(n) + 1)p + n = 0$. Solve for p, and get $q = n/p$.
3. What about discovering a directly?

   Can similarly prove that given a and n, can factor n. This is done via a randomized algorithm that uses a and n to factor n.

ElGamal Cryptosystem

Let G be a multiplicative group, e.g., $\mathbb{Z}_n^*$.

Order of an element $\alpha \in G$ is the smallest n such that $\alpha^n = 1$ in G.

Given $\alpha \in G$ of order n, define $\langle\alpha\rangle = \{\alpha^0, \alpha^1, \ldots, \alpha^{n-1}\}$. $\langle\alpha\rangle$ is a subgroup of G.

Definition: given G a multiplicative group, $\alpha \in G$ an element of order n, and $\beta \in \langle\alpha\rangle$, the *discrete log of* β is the unique integer a $(0 \leq a \leq n-1)$ such that

$$\alpha^a = b \text{ in } G.$$

Why is this interesting? Computing discrete logs is believed to be hard when G is well chosen.

The ElGamal cryptosystem is based on discrete logs in $\mathbb{Z}_p^*$ for a prime p.

Theorem: $\mathbb{Z}_p^* = \langle\alpha\rangle$ for some $\alpha \in \mathbb{Z}_p^*$. Such an α is called a primitive element, it has order $p-1$, since $\left|\mathbb{Z}_p^*\right| = p-1$.

The ElGamal cryptosystem is defined as follows.

Let p be a prime such that the discrete log is believed hard in Z_p^* ($p > 300$ digits, $p-1$ has at least one "large" prime factor).

Let $\alpha \in \mathbb{Z}_p^*$ be a primitive element.

$\mathcal{P} = \mathbb{Z}_p^*$

$\mathcal{C} = \mathbb{Z}_p^* \times \mathbb{Z}_p^*$.

$\mathcal{K} = \{(p, \alpha, a, \beta) \mid \beta \equiv \alpha^a \mod p\}$. Public part of key is p, α, β, private part is a.

To encrypt using $K = (p, \alpha, a, \beta)$, choose a random (secret) $k \in \mathbb{Z}_{p-1}$, and

$$e_K(x, k) = (\alpha^k \mod p, x\beta^k \mod p).$$

Idea: plaintext x is marked by β^k. Send along α^k to tell receiver what k to use.

To decrypt using $K = (p, \alpha, a, \beta)$:

$$d_K(y_1, y_2) = y_2(y_1^a)^{-1} \mod p.$$

Check:

$$\begin{aligned} d_K(\alpha^k \mod p, x\beta^k \mod p) &= x\beta^k(\alpha^{ka})^{-1} \mod p \\ &\equiv x\beta^k((\alpha^a)^k)^{-1} \mod p \\ &\equiv x\beta^k(\beta^k)^{-1} \mod p \\ &\equiv x \mod p. \end{aligned}$$

Given p, α, β, the attacker "needs" to compute a such that $\alpha^a \equiv \beta \mod p$; that is, he needs to find the discrete log. Stinson §6.2 and §6.3 presents algorithms to attack the discrete log problem.

Elliptic Curves

ElGamal cryptosystem can be implemented in any group where the discrete log problem is believed to be difficult.

In the above, we used $\mathbb{Z}_p^*$.

Other groups are popular. One such is elliptic curves modulo a prime $p > 3$. Let $a, b \in \mathbb{Z}_p$ be such that $4a^3 + 27b^2 \neq 0$. A (nonsingular) elliptic curve modulo p is the set E of all $x, y \in \mathbb{Z}_p$ such that $y^2 \equiv x^3 + ax + bmodp$, together with a special point $\mathcal{O}$ called the point at infinity.

We can make E into a group by definition an operation $+$ on points. See Stinson for details.

Lots of research on elliptice curves. They occur in many places. (Fermat's Last Theorem can be recast as a problem over elliptic curves, and in fact was finally proved in that way.)

Summary

Public key cryptography solves the key distribution problem.

Trade off: computational cost. It is more expensive to perform number-theoretic manipulations as required by RSA or ElGamal than to perform bit-twiddling operations like DES or AES.

Obvious solution: use both!

1. Use public key cryptography to exchange a secret (temporary) shared key.
2. Use the shared key and an efficient cryptosystem to exchange data.