Some slides for Algebra 2 EVU

Diego Ruano

Department of Mathematical Sciences
Aalborg University
Denmark

13-12-2013

Public-key crytography (from Wikipedia)

- The key used to encrypt a message is not the same as the key used to decrypt it.
- Each user has a pair of cryptographic keys-a public key and a private key. The private key is kept secret, while the public key may be widely distributed.
- Messages are encrypted with the recipient's public key and can only be decrypted with the corresponding private key.
- The keys are related mathematically, but the private key cannot feasibly (ie, in actual or projected practice) be derived from the public key.
- The discovery of algorithms that could produce public/private key pairs revolutionized the practice of cryptography beginning in the middle 1970s.

[Wikipedia]

You considered RSA in Algebra 1. We will consider three cryptosystems in Algebra 2:

1. Knapsack (Merkle-Hellman)
2. ElGamal,
3. McEliece

Knapsack is very fast and elegant, but was broken in 1982. However, there have been several improvements that have also been broken (in the 80's and 90's)

ElGamal is nowadays used in practice. It has been improved using elliptic curves: it has smaller key sizes and faster operations. New standards are coming.

McEliece is sequre against quantum computer attacks.

Knapsack crytosystem (Merkle-Hellman)

A knapsack problem:

- Consider a knapsack (or rucksack) with volume N
- Consider n objects with volume $e_1, \ldots, e_n$
- Maybe we cannot put everything in the knapsack, but we want to fill it. That is, we want to find $I \subset \{1, \ldots, n\}$ such that

$$\sum_{i \in I} e_i = N$$

Our knapsack problem

Given $e_1, \ldots, e_n \in \mathbb{N}$ and $N \in \mathbb{N}$, find a binary number k with n bits $k = (\lambda_1, \ldots, \lambda_n)$ ($\lambda_i = 0$ means object e_i is not in the knapsack) such that:

$$\sum_{i=1}^{n} \lambda_i e_i = N$$

Our knapsack problem is NP-Complete, but there is an easy case:

$$e_i > \sum_{j=1}^{i-1} e_j, \quad \forall\, i$$

Example: $(e_1, \ldots, e_5) = (2, 3, 7, 15, 31)$ and $N = 24$.
$24 - 15 = 9 \quad \rightarrow e_4$
$9 - 7 = 2 \quad \rightarrow e_3$
$2 - 2 = 0 \quad \rightarrow e_1$
Hence $N = 2 + 7 + 15$ and $k = (1, 0, 1, 1, 0) = 24$.

Message: is binary (0's and 1's). We cut it in blocks of length n. Consider that we send M, a block of length n.

1. Bob chooses an easy knapsack $(e_1, \ldots, e_n)$ and $N \in \mathbb{N}$ such that $N > \sum_{i=1}^{n} e_i$ (why?$\rightarrow$ unique encryption). He also chooses $w \in \mathbb{N}$ such that $0 < w < N$ and $\gcd(w, N) = 1$ (why?)
2. Bob computes $[w^{-1}]_N$ ($[w][w^{-1}] = [1]$) and $(a_1, \ldots a_n)$, with $0 < a_i < N$ where

$$a_i \equiv we_i \pmod{N}$$

   Secret Key: $(e_1, \ldots, e_n), N, w, w^{-1}$
   Public Key: $(a_1, \ldots, a_n)$
3. Alice wants to send $M = (M_1 \ldots, M_n) \in (\mathbb{Z}/2\mathbb{Z})^n$. She computes

$$C = \sum_{i=1}^{n} M_i a_i$$

   and sends it to Bob

4. Bob gets C. He computes $[Cw^{-1}]_N$ because

$$w^{-1}C \equiv \sum_{i=1}^{n} w^{-1}a_i M_i \equiv \sum_{i=1}^{n} e_i M_i (\mathrm{mod}\ N)$$

   We have $[Cw^{-1}]_N = [\sum M_i e_i]_N$. Note that $\sum M_i e_i \leq \sum e_i < N$, then $0 < \sum M_i e_i < N$ and encryption is unique.

5. Bob uses the easy knapsack to find $(M_1, \ldots, M_n)$ from $\sum M_i e_i$.

6. Eve?, she gets $C = \sum_{i=1}^{n} M_i a_i$, but it is not an easy knapsack.

Example knapsack

- $M = (1, 1, 0, 0, 1)$
- $N = 61$, $w = 17$, $\gcd(17, 61) = 1$
- $w^{-1} \equiv 18 (\mathrm{mod}\ 61)$
- $a_1 = 17 \cdot 2 \equiv 34 (\mathrm{mod}\ 61)$
  $a_2 = 17 \cdot 3 \equiv 51 (\mathrm{mod}\ 61)$
  $a_3 = 17 \cdot 7 \equiv 58 (\mathrm{mod}\ 61)$
  $a_4 = 17 \cdot 15 \equiv 11 (\mathrm{mod}\ 61)$
  $a_5 = 17 \cdot 31 \equiv 39 (\mathrm{mod}\ 61)$
- Public Key=$(34, 51, 58, 11, 39)$, so to encrypt (1,1,0,0,1) we have $34 + 51 + 39 = 124$. Alice sends 124.
- Bob receives 124 and computes $124 \cdot 18 \equiv 36 (\mathrm{mod}\ 61)$. Then he has an easy knapsack for 36:
  $36 - 31 = 5 \quad \rightarrow e_5$
  $5 - 3 = 2 \quad \rightarrow e_2$
  $2 - 2 = 0 \quad \rightarrow e_1$, and recovers $M = (1, 1, 0, 0, 1)$
- Eve could do: $124 = a_1 + a_2 + a_5 = 34 + 51 + 39$ but this is a difficult knapsack (for large numbers!)

ElGamal

Based on discrete logarithm problem:

Given a prime p and $y, g \in \mathbb{N}$, find x such that

$$y \equiv g^x \pmod{p}$$

1. Alice and Bob choose p, a big prime, and $g \in \mathbb{N}$ s.t. $0 < g < p$ and g has order $p-1$ in $(\mathbb{Z}/p\mathbb{Z})^*$ (a generator of $(\mathbb{Z}/p\mathbb{Z})^*$)
2. Alice chooses a, with $0 < a < p$ and computes $[g^a]_p$.
   Secret Key=a
   Public Key=$[g^a]_p$
3. Bob chooses b with $0 < b < p$ and computes $[g^b]_p$.
   Secret Key=b
   Public Key=$[g^b]_p$

4. Alice wants to send a message m, $0 < m < p$ to Bob. She sends:
$$\left([g^a]_p, [m(g^b)^a]_p\right)$$
5. Bob gets $([x_1]_p, [x_2]_p)$ and computes
$$[x_2]_p([x_1^b]_p)^{-1} = [mg^{ab}]_p([g^{ab}]_p)^{-1} = [m]_p$$
and since $m < p$ he can recover m.

To encrypt the message one uses the public key of the receiver and the secret key of the sender.

6. Eve?: she had to compute b from $[g^b]_p$

Example 8.19

Shor's algorithm

http://www.newscientist.com/article/mg22029445.100-my-quantum-algorithm-wont-break-the-internet-yet.html

http://www.newscientist.com/blog/technology/2007/09/how-quantum-computer-factorises-numbers.html

[Wikipedia]

[Quantum Computer]

McEliece or Code-based cryptography

1. Fix integers k, n, t. Consider the finite field with q elements.
2. Bob fixes G, a $k \times n$ generator matrix of a $[n, k]$-linear code that can correct t errors, and for a which an efficient decoding algorithm Dec is known. One can fix $t = \lfloor (d-1)/2 \rfloor$.
3. Bob fixes S, a random non-singular $k \times k$ matrix.
4. Bob fixes P, a random $n \times n$ permutation matrix (has exactly one entry 1 in each row and each column and 0's elsewhere)
5. Bob computes the $k \times n$ matrix: $\widehat{G} = SGP$.

Bob's public key is $(\widehat{G}, t)$
Bob's private key is (S, G, P).

McEliece or Code-based cryptography

- Alice wants to send a message $m \in \mathbb{F}_q^k$.
- Alice obtains the public key of Bob and chooses a random error vector e with $wt(e) \leq t$.
- Alice sends $c = m\widehat{G} + e$ to Bob.

Bob recovers the message using his private key:

- Bob computes $\widehat{c} = cP^{-1}$.
- Bob decodes $\widehat{c}$ and obtains $\widehat{m} = Dec(\widehat{c})$
- Bob computes $m = \widehat{m}S^{-1}$.

Why does it work?

$$\widehat{c} = cP^{-1} = (m\widehat{G} + z)P^{-1} = (mSGP + z)P^{-1} = (mS)G + zP^{-1}$$

Note that $(mS)G$ is a codeword and that $wt(zP^{-1}) = t$.

McEliece or Code-based cryptography

Historical drawback:

- The public key is very large.
- (Less significant:) There is a message expansion by a factor of n/k (ElGammal factor's message expansion is 2).

However:

> The algorithm has never gained much acceptance in the cryptographic community, but is a candidate for "post-quantum cryptography", as it is immune to attacks using Shor's algorithm and –more generally– measuring coset states using Fourier sampling

From [Wikipedia]

Generalized ElGamal

ElGamal encryption scheme is tipcally described using the multplicative grup $\mathbb{Z}_P^*$, but it can be easily generalize to work in any finite cyclic group G.

Some links:

- [Wikipedia: Elliptic crytopgraphy]
- [Wikipedia: Elliptic curve]
- [Easy example]
- [Safe curves]