Cryptology – something old and something new

THOMAS JOHANSSON

CRYPTO AND SECURITY

DEPT. OF ELECTRICAL AND INFORMATION TECHNOLOGY, LUND UNIVERSITY, SWEDEN

Cryptology, what is that?

- the study of *cryptographic primitives*
- a cryptographic primitive - low-level cryptographic algorithms (black-box description), building block, ...
- a cryptographic primitive contains a mathematically oriented description
- cryptographic primitives are used to build *secure systems*

LUND UNIVERSITY

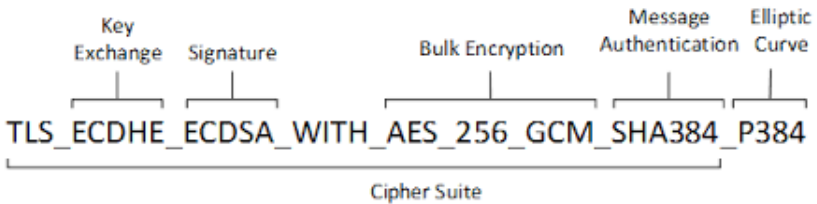
Cryptographic primitives

- Cryptographic hash function, (compute a hash value for a message (SHA-256))
- Symmetric-key encryption (encrypt/decrypt with a shared key (AES))
- Public-key encryption (encrypt with a public key and decrypt with a different secret key (RSA))
- Digital signatures (confirm the authorship and authenticity of a message by its signature (DSS))
- Mix network (pool communications from many users to anonymize what came from whom)
- Private information retrieval (get database information without server knowing which item was requested)
- Commitment scheme (commit to a chosen value hidden to others, with the ability to reveal it later)

LUND UNIVERSITY

Example of an application

- HTTPS protocol through TLS
- secure communication over a computer network
- you need a Public Key Infrastructure, key agreement, signatures, encryption and data integrity



LUND UNIVERSITY

Cryptographers, who are they?

- mathematicians, theoretical computer scientists, electrical engineers
- International Association for Cryptologic Research, IACR, iacr.org

discrete algebraic structures, probability theory and statistics

LUND UNIVERSITY

Historic view

- secrecy

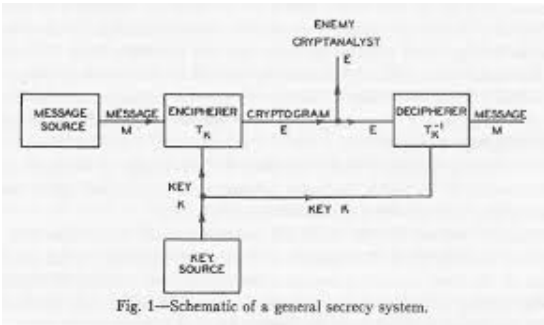


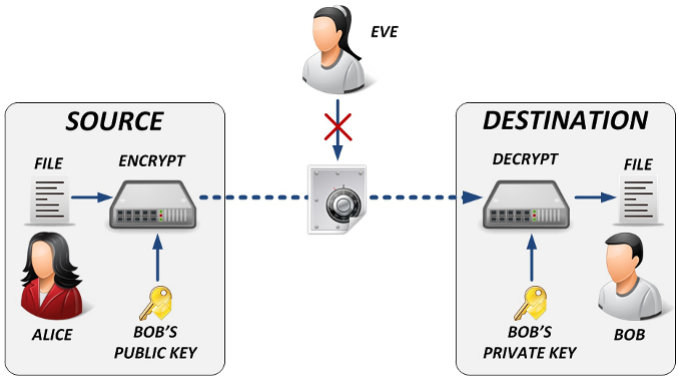
Fig. 1—Schematic of a general secrecy system.

(Shannon, 1949)

LUND UNIVERSITY

The big game changer I

- the birth of Public Key Cryptography (Diffie, Hellman 1976 and R. Merkle)



Public-key encryption scheme

- A set of encryption transformations $\{E_e : e \in \mathcal{K}\}$ and a set of decryption transformations $\{D_d : d \in \mathcal{K}\}$. For each $e \in \mathcal{K}$ there is a corresponding $d \in \mathcal{K}$ such that $D_d(E_e(M)) = M$, $\forall M$.
- Furthermore, after choosing such a pair (e, d), the *public key* e (or the public parameter) is made public, while the associated *secret key* d is kept secret.
- For the scheme to be secure, it must be computationally infeasible to compute d as well as computing $E_e^{-1}(C)$, knowing the public value e.

LUND UNIVERSITY

The RSA public-key encryption scheme

Let $n = pq$, where p and q are two large primes. Let $\mathcal{M} = \mathcal{C} = \mathbb{Z}_n$. Pick a number e relatively prime to $\phi(n)$ and calculate a number d such that $ed = 1 \bmod \phi(n)$. The public key is the two numbers (n, e) and the (public) encryption transformation $E(M)$ is

$$E(M) = M^e \bmod n.$$

The secret key is the number d (as well as p, q and $\phi(n)$) and the secret decryption transformation $D(C)$ is

$$D(C) = C^d \bmod n.$$

LUND UNIVERSITY

Verify that decrypting a ciphertext returns the encrypted plaintext.

$$D(C) = C^d = (M^e)^d = M^{ed} \bmod n.$$

Now we note that $ed = 1 \bmod \phi(n)$, which means that we can write

$$ed = 1 + t \cdot \phi(n),$$

for some integer t. S we can continue

$$D(C) = M^{ed} = M^{(1+t\cdot\phi(n))} = M \cdot M^{t\cdot\phi(n)} \bmod n.$$

From Euler's formula we know that $x^{\phi(n)} = 1$ for any $x \in \mathbb{Z}_n^*$. So assuming that M is invertible we have

$$D(C) = M \cdot M^{t\cdot\phi(n)} = M \cdot 1 \bmod n.$$

Security of the RSA cryptosystem

- If we can factor n then we can compute $\phi(n)$ and d.
- RSA relies on the *factoring problem*.
- This does not mean that breaking RSA is equivalent to solving a factorization problem. It is not known whether RSA can be broken without factoring n.

LUND UNIVERSITY

The computational complexity of factoring

- We often use the function

$$L_n(\alpha, \beta) = \exp((\beta + o(1))(\log n)^{\alpha}(\log \log n)^{1-\alpha}).$$

- An algorithm with complexity $O(L_n(\alpha, \beta))$ for $0 < \alpha < 1$ is said to have *sub-exponential* behaviour.
- Number Field Sieve: This is currently the most successful method for numbers with more than 100 decimal digits. It can factor numbers of the size of 2^{512} and has complexity $L_n(1/3, 1.923)$.
- For long-term security one would need to take a size of over 2048 bits.
- RSA-768 was factored in 2009.

LUND UNIVERSITY

How fast can we compute a ciphertext?

- Complexity of computing $x^e \bmod n$?
- Complexity of computing $xy \bmod n$?
- Basic school book n^2; Karatsuba $n^{1.58}$; ...

$$n = 279978339112213278708294676387226016210704467869554285375600099293261284001076093456710529553608560618223519109513657886371059544820065767750985805576135790987349501441788631789462951872378692218239 83.$$

It still requires many operations to compute $M^e \bmod n$ ...

LUND UNIVERSITY

The El Gamal cryptosystem (1985)

- Domain parameters: p large prime; g an element in $\mathbb{Z}_p^*$.
- Public key: $h = g^x \bmod p$
- Encryption: Pick a random $k \in \mathbb{Z}_p$. For message $m \in \mathbb{Z}_p$ generate the ciphertext

$$(c_1, c_2) = (g^k, m \cdot h^k).$$

- Decryption:

$$\frac{c_2}{c_1^x} = \frac{m \cdot h^k}{g^{kx}} = m.$$

LUND UNIVERSITY

The Discrete Log Problem

- Domain parameters: p large prime; g generator for a large subgroup in $\mathbb{Z}_p^*$.
- Given
$$h = g^x \bmod p,$$
compute x.
- Index calculus algorithms: can use methods similar to the NFS
- Complexity: subexponential $L_p(1/3, c)$ requires p to be 1024-bit or 2048-bit.

LUND UNIVERSITY

The El Gamal cryptosystem again

- El Gamal works for any finite group!
- The discrete log problem in a group G:
- Given a finite group $h = g^x$ compute x.
- The computational complexity for solving the DLP depends on the group!
- $< \mathbb{Z}_p, + >$ easy; $< \mathbb{Z}_p^*, \cdot >$ subexponential; elliptic curve groups - exponential (?)

LUND UNIVERSITY

Elliptic curve crypto

- Suggested in the 80s, wide-spread from 2004-.
- Example:

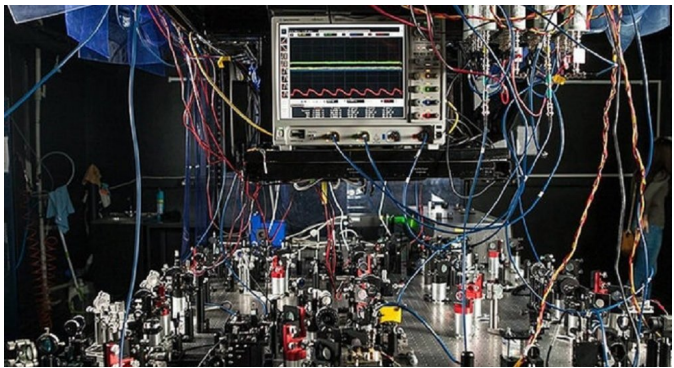
$$E : Y^2 = X^3 + X + 3$$

  over the field $\mathbb{Z}_p = \mathbb{Z}_7$.
  Points on the curve: $\mathrm{O}, (4,1), (6,6), (5,0), (6,1), (4,6)$
  Group law for adding points, e.g., $(4,1) + (6,6) = (5,0)$
- The computational complexity for the best algorithms solving the DLP on some elliptic curve groups is exponential.
- Modulus p can be much smaller! (top breaking record is 112-bit ECDLP)

LUND UNIVERSITY

The big game changer II ?

- the birth of quantum computing



LUND UNIVERSITY

The big game changer II ?

Shor's algorithm (1994)

If a large enough (number of qubits) quantum computer is available, then

- The factorization problem can be solved in polynomial time.
- The discrete log problem can be solved in polynomial time.
- Most public-key crypto of today is not secure...
- ...the world is collapsing...

LUND UNIVERSITY

The Quantum era

- The Grover algorithm - affects symmetric crypto (but not so serious)
- Some things we encrypt today might need to be kept secret for the upcoming 30-50 years.
- we need to find a new solution now...
- the area of crypto in the presence of a quantum computer is called *post-quantum cryptography*

LUND UNIVERSITY

Post-Quantum Crypto algorithms and future standards

- Most of todays security solutions are based on the difficulty of either *factoring* or solving the *discrete log* problem (RSA, DH, ECDH, . . .).
- If a (large enough) quantum computer can be built in 5 (or 10, or 50) years, it can solve these problems fast (in polynomial time).
- A huge threat that forces everyone to reconsider how to build future security algorithms.
- NIST now has an ongoing post-quantum standardization project.

LUND UNIVERSITY

Post-Quantum Crypto are usually based on ...

- Lattice-based crypto (Learning with errors)
- Code-based crypto
- Multivariate crypto (solving systems of multi-variate quadratic equations)
- hash-based crypto

Our research investigates how difficult instances from these different problems are.

LUND UNIVERSITY

Explaining Learning with Errors (LWE)

Basic Linear Algebra in $\mathbb{R}$

$$\begin{pmatrix} 3 & 5 & 2 & -4 & -1 \\ 4 & -1 & 3 & -4 & 3 \\ -2 & 2 & 2 & 3 & -3 \\ 1 & 0 & -4 & -4 & 1 \\ 0 & -5 & 2 & -2 & 1 \end{pmatrix} \begin{pmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \end{pmatrix} = \begin{pmatrix} -6 \\ 20 \\ -9 \\ -5 \\ 20 \end{pmatrix}$$

LUND UNIVERSITY

A Slight Modification - the LWE problem

- Arithmetic in $\mathbb{Z}_{13} = \{-6, -5, \ldots, 5, 6\}$
- e_i small (for example $e_i \in \{0, 1\}$)

$$\begin{pmatrix} 3 & 5 & 2 & -4 & -1 \\ 4 & -1 & 3 & -4 & 3 \\ -2 & 2 & 2 & 3 & -3 \\ 1 & 0 & -4 & -4 & 1 \\ 0 & -5 & 2 & -2 & 1 \\ -3 & 1 & 2 & -1 & -4 \\ 2 & -1 & 3 & -1 & 3 \end{pmatrix} \begin{pmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \end{pmatrix} + \begin{pmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \\ e_6 \\ e_7 \end{pmatrix} = \begin{pmatrix} -5 \\ -6 \\ 4 \\ -4 \\ -6 \\ 4 \\ 3 \end{pmatrix}$$

LUND UNIVERSITY

A Slight Modification - the LWE problem

- Arithmetic in $\mathbb{Z}_{13} = \{-6, -5, \ldots, 5, 6\}$
- e_i small (for example $e_i \in \{0, 1\}$)

$$\begin{pmatrix} 3 & 5 & 2 & -4 & -1 \\ 4 & -1 & 3 & -4 & 3 \\ -2 & 2 & 2 & 3 & -3 \\ 1 & 0 & -4 & -4 & 1 \\ 0 & -5 & 2 & -2 & 1 \\ -3 & 1 & 2 & -1 & -4 \\ 2 & -1 & 3 & -1 & 3 \end{pmatrix} \begin{pmatrix} 2 \\ -3 \\ 2 \\ 0 \\ 1 \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} -5 \\ -6 \\ 4 \\ -4 \\ -6 \\ 4 \\ 3 \end{pmatrix}$$

LUND UNIVERSITY

Learning with Errors

There is a secret vector **s** in $\mathbb{Z}_q^n$. We can query an oracle:

The LWE Oracle with Parameters $(n, q, \mathcal{X})$:

1. Uniformly picks **a** from $\mathbb{Z}_q^n$.
2. Picks a small 'error' e.
3. Outputs the pair $(\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle + e)$ as a sample.

Error Distribution

Discrete Gaussian over $\mathbb{Z}_q$ with mean 0 and standard deviation σ.

$D_{\sigma,0}$ returns integers $x \in \mathbb{Z}_q$ with prob. proportional to $\exp(-x^2/(2\sigma^2))$.

Lund University

LWE Problem Parameters

The LWE Parameters

1. Dimension n
2. Alphabet size q
3. The standard deviation σ.

How difficult is it to solve an LWE instance?

LUND UNIVERSITY

Asymptotic Complexity of Solving LWE

There are several reasons to believe the LWE problem is hard (Regev 2005).

- The best known algorithms for LWE run in exponential time (quantum algorithms do not help much).
- LWE (for certain parameters) is known to be hard based on certain assumptions regarding the worst-case hardness of standard lattice problems such as GAPSVP (a decision version of the shortest vector problem)

LUND UNIVERSITY

Complexity of Solving LWE

- $q = n^{c_q}$
- $\sigma = n^{c_s}$
- Complexity of Solving LWE is usually $2^{(c+o(1))n}$, where constant c depends on c_q and c_s.

Example (Regev instances)

$c_q = 2$ *and* $c_s = 1.5$.

LUND UNIVERSITY

Results

Time complexities are on the form $2^{(c+o(1))n}$

Algorithm	Complexity Exponent (c)
Quantum Computer	0.8856
Classical Computer	0.8951
Previous best	0.9299

LUND UNIVERSITY

Back to crypto - Ring-LWE

$\mathbb{Z}_q/(f(x))$, where $f(x) = x^n + 1$, where $n = 2^m$, is a common choice.
There is a secret polynomial $s(x)$ in $\mathbb{Z}_q/(x^n + 1)$.

The Ring-LWE Oracle with Parameters $(n, q, \mathcal{X})$:

1. Uniformly picks $a(x)$ from $\mathbb{Z}_q/(x^n + 1)$.
2. Picks a small 'error' polynomial $e(x)$ from $D^n_{\sigma,0}$.
3. Outputs $(a(x), b(x)) = (a(x), a(x)s(x) + e(x))$ as a sample.

$D^n_{\sigma,0}$ returns n integers from $D_{\sigma,0}$.

LUND UNIVERSITY

Distributing keys using Ring-LWE

Think of vectors as polynomials in $\mathbb{Z}_q/(x^n+1)$.
Public key generation by Bob: Chooses random a(x), small $s(x)$, $e(x)$ and form

$$b(x) = a(x)s(x) + e(x).$$

Publish $(b(x), a(x))$.
Alice sends a key to Bob: Chooses small $s'(x)$, $e'(x)$, $e''(x)$ and compute

$$c(x) = a(x)s'(x) + e'(x)$$

and

$$c'(x) = b(x)s'(x) + e''(x) + q/2 \cdot k(x).$$

Here $k(x)$ is a binary polynomial (key). Alice sends $(c(x), c'(x))$ to Bob.

LUND UNIVERSITY

Distributing keys using LWE

Bob recovering the key: Compute

$$m(x) = c'(x) - c(x)s(x).$$

If m_i closest to $q/2$ then $k_i = 1$ else if m_i closest to 0 then $k_i = 0$.
Explanation:

$$m(x) = c'(x) - c(x)s(x) = b(x)s'(x) + e''(x) + q/2 \cdot k(x) - (a(x)s'(x) + e'(x))s(x)$$

$$= (a(x)s(x) + e(x))s'(x) + e''(x) + q/2 \cdot k(x) - (a(x)s'(x) + e'(x))s(x)$$

$$= q/2 \cdot k(x) + \underbrace{e(x)s'(x) + e''(x) - e'(x)s(x)}_{\textit{small}}.$$

LUND UNIVERSITY

Example

$\mathbb{Z}_{19}/(x^8+1)$, $\mathbb{Z}_{19} = \{-9, -8, \ldots, 8, 9\}$, small$=\in \{-1, 0, 1\}$
Public key generation by Bob: Chooses random

$$a(x) = 2x^7 - 9x^6 - 4x^5 - 4x^4 - 3x^3 + 5x^2 - 4x + 5,$$

small $s(x) = x^7 + x^3 - x^2 - 1$, $e(x) = -x^6 + x^5 - x^2 + x$ and form
$b(x) = a(x)s(x) + e(x)$

$$b(x) = 3x^7 + 7x^6 + 3x^5 - 1x^4 - 3x^3 + 9x^2 - 8x - 6.$$

Publish $(b(x), a(x))$.

LUND UNIVERSITY

Example

Alice sends a key to Bob: Let $k = (1, 0, 1, 0, 1, 0, 1, 0)$, or $k(x) = x^7 + x^5 + x^3 + x$.
Chooses small $s'(x) = x^5 + x^4 - x^2$, $e'(x) = -x^7 + x^5 + x^3 + x$,
$e''(x) = x^6 + x^3 - x$ and compute $c(x) = a(x)s'(x) + e'(x)$

$$c(x) = 5x^7 + 5x^6 + 5x^5 - 2x^4 - 7x^3 + 8x^2 - 8x - 2$$

and $c'(x) = b(x)s'(x) + e''(x) + 9 \cdot k(x)$

$$c'(x) = -7x^7 + 3x^6 - 2x^5 + x^4 + 8x^3 - 4x^2 + 9x - 8$$

Alice sends $(c(x), c'(x))$ to Bob.

LUND UNIVERSITY

Example

Bob recovering the key: Compute

$$m(x) = c'(x) - c(x)s(x).$$

$$m(x) = 7x^7 - 1x^6 - 7x^5 + x^4 - 7x^3 + 9x + 1.$$

If m_i closest to $q/2$ then $k_i = 1$ else if m_i closest to 0 then $k_i = 0$.

$$k = (1, 0, 1, 0, 1, 0, 1, 0)$$

LUND UNIVERSITY

Current PQ-crypto research topics

basis for future security standards

- NIST standardization project contains 84 different proposals
- Difficulty of underlying problems
- Security of specific proposals (different types of primitives)
- Hardware implementation aspects
- Software implementation for constrained devices
- Side-channel attacks and its protection

LUND UNIVERSITY

Thank you for your attention!
Any questions?