Advanced Computer Security, part I: Basic cryptography

T. Karvi

October 2010

Number theory

- Knowledge about modulo arithmetics and finite fields is essential when studying both traditional symmetric ciphers and public key cryptography.
- In this chapter we introduce the most important concepts in these areas. Courses in the department of mathematics (Algebra I and II) offer more detailed material. This introduction should be enough for understanding RSA and key exchange protocols.
- But when studying more advanced elliptic curve cryptography this introduction or even the above mentioned mathematical courses are not enough, but it is necessary to read more about finite fields and their algorithmic methods.
- It is necessary to use even algebraic geometry. However, this course does not touch these advanced methods.

Modulo arithmetics

Modulo or mod operation is important when dealing with public key cryptography. It behaves well with respect to addition and multiplication:

$$\begin{aligned} ((a \bmod n) + (b \bmod n)) \bmod n &= (a+b) \bmod n, \\ ((a \bmod n)(b \bmod n)) \bmod n &= (ab) \bmod n. \end{aligned}$$

Let us denote by $\mathbf{Z}_n$ the set of integers $\{0, 1, 2, \cdots, n-1\}$. In other words, $\mathbf{Z}_n$ is the set of residues modulo n.

We can define addition $\oplus$ and multiplication $\otimes$ in the set $\mathbf{Z}_n$ as follows:

Definition

Let $a, b \in \mathbf{Z}_n$. Define

$$a \oplus b = (a + b) \mod n, \qquad a \otimes b = (ab) \mod n.$$

Operation mod is the normal modulo operation. Instead of $\oplus$ and $\otimes$ ordinary notations for addition and multiplication are used, if it is clear that we mean modulo operations.

Inverses

- In the set $\mathbf{Z}_n$ every element has an inverse with respect to addition.
- In other words, if $a \in \mathbf{Z}_n$, then there is $b \in \mathbf{Z}_n$ such that

$$a \oplus b = 0$$

.

- For example, if $n = 5$ and $a = 3$, then the inverse of a with respect to addition is 2, because $(3 + 2) \mod 5 = 0$.

Multiplicative inverse

If $a \in \mathbf{Z}_n$, then the multiplicative inverse of a is an element $b \in \mathbf{Z}_n$ such that $ab \mod n = 1$. The existence of multiplicative inverses is a more difficult question than additive inverses. The basic result is the following:

Theorem

An element $a \in \mathbf{Z}_n$ has a multiplicative inverse if and only if $gcd(a, n) = 1$. If $gcd(a, n) = 1$, then the multiplicative inverse is unique. (gcd means the greatest common divisor.) $\square$

For example, if $n = 12$, then 1, 5, 7 and 11 have multiplicative inverses in $\mathbf{Z}_{12}$, because the gcd of those numbers with respect to n is one. As a matter of fact, the inverse of 5 is 5 (similarly with 7 and 11).

Theorem

In $\mathbf{Z}_n$, every nonzero element has a multiplicative inverse if and only if n is a prime. □

Multiplicative inverses are found with the help of Extended Euclidean algorithm. (See exercises).

Group $U(\mathbf{Z}_n)$

- Denote by the symbol $U(\mathbf{Z}_n)$ the set of elements in $\mathbf{Z}_n$ that have multiplicative inverses modulo n.
- When we consider the set $U(\mathbf{Z}_n)$, we consider only multiplication, never addition.

Group $U(\mathbf{Z}_n)$

- Denote by the symbol $U(\mathbf{Z}_n)$ the set of elements in $\mathbf{Z}_n$ that have multiplicative inverses modulo n.
- When we consider the set $U(\mathbf{Z}_n)$, we consider only multiplication, never addition.
- We know that $U(\mathbf{Z}_p) = \{1, 2, \cdots, p-1\}$, if p is a prime. In fact, $U(\mathbf{Z}_p)$ is cyclic i.e. there is an element $a \in U(\mathbf{Z}_p)$ that generates the set $U(\mathbf{Z}_p)$ (that is to say: when k runs through the numbers $0, 1, \cdots, p-1$, then a^k goes through all the elements in $U(\mathbf{Z}_p)$.
- This kind of a is called a primitive root modulo p.

Example

2 is a primitive root mod 5, because $2^1 = 2$, $2^2 = 4$ and $2^3 = 3$. On the other hand, 2 is not a primitive root mod 7, because $2^3 \mod 7 = 1$, but 3 is a primitve root.

- It is known that there are always primitive roots modulo a prime.
- Guessing is a rather effective algorithmic way to find primitive roots.
- There are methods that test the correctness of a guess quicker than trying all the exponents. We skip the description of these methods.
- (Note: Emil Artin has formulated the following famous hypothesis: If $a > 1$ is not of the form b^2 for some b, then there are infinitely many primes that have a as a primitive root. Even if some progress has been taken, the proof of the hypothesis is still widely open.)

Consider an arbitrary n and the existence of primitive roots in the set $U(\mathbf{Z}_n)$. The basic result is the following.

Theorem

An arbitrary natural number n has primitive roots, if and only if n is of the form 2, 4, p^a or $2p^a$, where p is a prime. □

Finite fields

- A field is a set with two operations, addition and multiplication. Sometimes these operations have nothing to do with ordinary addition and multiplication with numbers.
- There are neutral elements with respect to both operations. So there is an element 0 such that $x + 0 = 0 + x = x$ (addition),
- and there is an element 1 such that $1x = x1 = x$ (multiplication).
- Elements have inverses with respect to these operations.
- In addition, the operations are commutative, associative and distributive.
- Examples of infinite fields are $\mathbf{R}$ ja $\mathbf{C}$. The basic examples of finite fields are the fields $\mathbf{Z}_p$.

Finite fields II

- Let $GF(m)$ be a finite field with m elements. Only certain numbers m are possible.
- In fact, m must be of the form p^n, where p is a prime, the characteristics of the field.
- We show in the following how to construct the field $GF(p^n)$. The construction is based on polynomials. The same method is used in the Rijndael cipher, too.

Finite fields III

- Let f be an irreducible polynomial

$$a_n X^n + a_{n-1} X^{n-1} + \cdots + a_1 X + a_0,$$

  where the coefficients $a_i \in \mathbf{Z}_p$, p a prime.
- Irreducibility means that there are no polynomials g and h such that $\deg(g) \geq 1$, $\deg(h) \geq 1$ and $f = gh$. (Normal multiplication of polynomials, but the coefficients are added and multiplied modulo p.)
- Denote $f \in \mathbf{Z}_p[X]$, when f is a polynomial with coefficients in $\mathbf{Z}_p$.

Finite fields IV

- If also $g \in \mathbf{Z}_p[X]$ and g is divided by f, then we get the result of the division and the residue $h \in \mathbf{Z}_p[X]$.
- Then $\deg(h) < \deg(f)$.
- There can be only a finite amount of different residues, when divided by f, because there are only a certain finite number of coefficients and the degree of the residue is less that the degree of f.
- As a matter of fact, there are exactly p^n residues.
- When f has been fixed and the degree of f is n, we denote the set of residues by $GF_f(p^n)$. It turns out that $GF_f(p^n)$ is a field, when addition $\oplus$ and multiplication $\otimes$ are defined as follows.

Finite fields V

- Let $g, h \in GF_f(p^n)$. Set

$$g \oplus h = (g + h) \mod f, \qquad g \otimes h = (gh) \mod f.$$

- It is necessary to show that the multiplication has inverses.
- If $g \in GF_f(p^n)$, then by applying the Extended Euclidean algorithm we can find polynomials r and s such that

$$rg + sf = 1.$$

  Now r is the inverse of g with respect to the multiplication.
- The field $GF_f(p^n)$ does not depend on the choice of f. If g is another irreducible polynomial of the same degree, then the field $GF_g(p^n)$ is isomorfic with the field $GF_f(p^n)$.

Finite fields VI

- Let us construct, for example, the field $GF(2^2)$.
- We need first an irreducible polynomial $f \in \mathbf{Z}_2[X]$ of degree two.
- The polynomial $f(X) = X^2 + X + 1$ is such.
- If it were reducible, it would be the product of two polynomials of degree one. Then it would have at least one root. Our polynomial f has, however, no roots in $\mathbf{Z}_2$ and so it must be irreducible.
- The elements of the field $GF(4)$ are the residue polynomials modulo f, i.e. the polynomials 0, 1, X ja $X + 1$. The addition operation is seen in the following table:

Finite fields VII

+	0	1	X	X+1
0	0	1	X	X+1
1	1	0	X+1	X
X	X	X+1	0	1
X+1	X+1	X	1	0

Finite fields VII

+	0	1	X	X+1
0	0	1	X	X+1
1	1	0	X+1	X
X	X	X+1	0	1
X+1	X+1	X	1	0

The multiplication operation is seen in the following table:

*	1	X	X+1
1	1	X	X+1
X	X	X+1	1
X+1	X+1	1	X

Note how the addition and multiplication is based on the corresponding operations in $\mathbf{Z}_2$.

Famous Problems in Cryptography

Modern cryptography is based on some mathematical problems which are difficult to solve.

i) Factorization: *Given an integer n, find a prime p that factors n (i.e. $p|n$).* No polynomial algorithm is known for this problem. On the other hand, it is not known to NP-complete and no lower bound has been proved.

ii) Discrete logarithm: *Given a prime p, a primitive root a of p, and a number $a^s \mod p$, find s.* No polynomial algorithm is known for this problem. No lower bound has been proved. Essentially the same problem remains hard, if numbers are replaced with other elements, for example elliptic curve points.

It is a little worrisome that practically all the cryptographic protocols depend on these two mathematical problems.

Block ciphers: Rijndael

- In the block ciphers, plain texts are partitioned into blocks, whose lengths are typically 64 or 128 bits.
- Every block is encrypted in the same way. Blocks are sent to a receiver, usually chaining them in one way or another. Chaining prevents an opponent to change the order of the blocks or to duplicate them.
- As an example of a modern block cipher we examine one system, Rijndael, more closely. It was a surprise winner in the competition for the new encryption standard (Advanced Encryption Standard, AES) (arranged by USA).
- This competition, arranged by NIST, started in January 1997 and Rijndael was declared a winner in April 2000. The designers were the Belgians Joan Daemen and Vincent Rijmen.

Block ciphers: Rijndael II

- There were 15 proposals in the first round. These proposals came from 11 different countries.
- In 1999 five finalists were chosen. These were Rijndael (BE), Serpent (UK-IL-DK), Twofish (USA), RC6 (USA), Mars (USA).
- In the evaluation of the finalists, the efficiency of software and hardware implementations was emphasized. Finally, the winner was Rijndael.

Block ciphers: Rijndael III

- In this course we describe the structure of Rijndael in a concise style. Those who want a wider description can read the book J. Daemen and V. Rijmen, *The Design of Rijndael*, Springer 2002. In addition, W. Stallings, *Cryptography and Network Security*, Third Edition, Prentice Hall 2003, contains quite a good and broad description of the method.
- Rijndael is a block cipher. The length of a block may vary and the same is true for keys. The length of a block or key can be a multiple of 32 with minimum 128 and maximum 256 bits.
- Inputs and outputs to Rijndael are one-dimensional arrays consisting of 8 bit bytes. Several rounds are used in order to encrypt a plaintext.

Rijndael I

- The rounds operate on intermediate results that are called states.
- A state can be represented as a matrix of bytes. There are four rows in a matrix. The number of columns in a state is N_b that is the same as the length of a block divided by 32.
- A key is represented with the help of a matrix of four rows. The number of columns is denoted by N_k which is the same as the length of the key divided by 32.

For example, the following matrices represent a state and a key:

Example

p_0	p_4	p_8	p_{12}
p_1	p_5	p_9	p_{13}
p_2	p_6	p_{10}	p_{14}
p_3	p_7	p_{11}	p_{15}

k_0	k_4	k_8	k_{12}	k_{16}	k_{20}
k_1	k_5	k_9	k_{13}	k_{17}	k_{21}
k_2	k_6	k_{10}	k_{14}	k_{18}	k_{22}
k_3	k_7	k_{11}	k_{15}	k_{19}	k_{23}

The first matrix represent a plaintext block. It has $N_b = 4$, so that the length of the block is $4 \times 32 = 128$. In the case of the key matrix, $N_k = 6$ and the length of the block is thus $6 \times 32 = 192$.

Rijndael II

Rijndael consists of the following phases:

```
Rijndael(State, CipherKey)

begin

    KeyExpansion(CipherKey, ExpandedKey);
    AddRoundKey(State, ExpandedKey[0]);
    for i := 1 until Nr-1 loop
        Round(State, ExpandedKey[i]);
    end for;
    FinalRound(State, ExpandedKey[Nr]);
end.
```

The encryption takes place in the routine `Round`. It consists of four phases:

```
Round(State, ExpandedKey[i])

begin

    SubBytes(State);
    ShiftRows(State);
    MixColumns(State);
    AddRoundKey(State, ExpandedKey[i]);
end;
```

The routine `FinalRound` is nearly the same as `Round`:

```
FinalRound(State, ExpandedKey[Nr])

begin

    SubBytes(State);
    ShiftRows(State);
    AddRoundKey(State, ExpandedKey[Nr]);
end;
```

SubBytes

- All transformations applied in Rijndael are linear transformations (check linear algebra: matrices = linear mappings).
- The only exception is the procedure SubBytes which is non-linear. It mixes a block using the following principle:
- Rijndael uses 16×16 array, so called S-box, whose values are hexadecimal numbers.
- Every byte in a state is transformed into another byte as follows. The first four bits in a byte are intepreted as a hexadecimal number $0 \cdots F$, and similarly the four righmost bits. These numbers are used as indexes when picking a new 8 bit value from the S-box. The old byte is replaced by this new byte, picked from the S-box.

SubBytes II

- The S-box is designed such that the transformation is non-linear and that it mixes bytes well.
- Of course, the transformation must invertible. Otherwise the decryption will not succeed.
- The main motivation for the S-box is to make the differential and linear cryptoanalysis more difficult.
- If all operations in the encryption were linear, then the above analysis methods would work better. The S-box makes a non-linear transformation, what prevents the straightforward application of these analysis methods.

SubBytes

- In this context it is interesting that the inventors of Rijndael refer to the article by Kaisa Nyberg "Differentially uniform mappings for cryptography", *Advances in Cryptology, Proc. Eurocrypt'93, LNCS 765*, T. Helleseth, ed. Springer-Verlag, 1994, pp. 55-64.
- The article examines principles according to which it is possible to generate a good S-box.
- The designers of Rijndael have taken one suggestion of the article. In this course we do not start to examine the theory of S-boxes which demands for example knowledge about finite fields.

ShiftRows

This is a transposition of bytes which shifts rows cyclically (compare the similar operation in machine languages). The following matrices show how this shift works. The first matrix shows the initial situation and the second the result.

a	b	c	d
e	f	g	h
i	j	k	l
m	n	o	p

a	b	c	d
f	g	h	e
k	l	i	j
p	m	n	o

MixColumns

This step can be formulated using the multiplication of matrices. It must be noted, however, that both addition and multiplication take place in the field $GF(2^8)$. The matrix operation is applied to a state column by column. For one column the transformation is as follows:

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \times \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

MixColumns

- The values of a column, a_i, are bytes of 8 bits. These bytes are intepreted as elements of the field $GF(2^8)$ i.e. polynomials.
- If for example $a_0 = 01001101$, then it represents the polynomial $X^6 + X^3 + X^2 + 1$.
- In the same way the numbers in the coefficient matrix are interpreted as bytes and furthermore as polynomials as presented above.
- The matrix multiplication is normal, but the elements are considered to be in the field $GF(2^8)$.
- Thus for example

$$b_0 = (2 \otimes a_0) \oplus (3 \otimes a_1) \oplus a_2 \oplus a_3,$$

where $\oplus$ means the addition of polynomials and $\otimes$ means the multiplication of the polynomials in the field $GF(2^8)$.

AddRoundKey

- In this step a simple one time pad encryption is performed to the mixed plaintext.
- The secret key used in this operation is obtained of the secret master key using transformations defined for keys.
- The key is added with the state using the XOR-operation bit by bit (addtion modulo two).

Usage of the key

The master key is used to generate round keys for every round. These round keys are used in the step AddRoundKey. The generation of round keys is no more difficult than the encryption itself, but we skip it in this course.

The security of Rijndael

- The competition was open and the candidates were evaluated openly and internationally. Because no clear vulnerabilities were detected, it seemed quite safe. There are, however, some problems which were detected afterwards.
- *Algebraic approaches* can be applied to Rijndael and they nearly broke the cipher. The idea is to formulate a system of algebraic equations according to the functioning of a cipher.
- The algebraic analysis of the 128 bit Rijndael has led to a system of equations with 16 000 unknown and 8000 second order equations (Courtois and Pieprzyk. Cryptanalysis of block ciphers with overdefined systems of equations. IACR eprint server http://www.iacr.org).

- Thus the system is *Diophantine*, i.e. there are more unknowns than equations. There are no mechanical solution methods for such systems as Yuri Matiyasevich (Finnish transliteration Juri Matijasevits) showed already 1970 (when he solved this so called Hilbert's 10th problem).
- However, it may be the case that some special systems can be solved mechanically. As a matter of fact, different kind of equation systems can be deduced from AES.
- Thus the security of the new standard AES depends on these equations which maybe can be solved some day.

Side channel attacks

- More serious threat is a side channel attack that was discovered in 2005 (Daniel J. Bernstein: Cache-timing attacks on AES).
- Bernstein demonstrates complete AES key recovery from known-plaintext timings of a network server on another computer. This attack should be blamed on the AES design, not on the particular AES library used by the server; it is extremely difficult to write constant-time high-speed AES software for common general-purpose computers.

Public Key Cryptography

- The basic idea of public key encryption or asymmetric encryption is that encryption can be done using a public key. The receiver decrypts the message using his secret private key. One essential condition is that it is not possible to detect the secret key even if the encryption key is public.
- The advantage of public key encryption is that everybody can send an encrypted message to a receiver without first agreeing of keys with the receiver.
- The receiver is the only one who can decrypt the message with his secret key.
- The idea of public key encryption was published first by Diffie and Hellman in 1976. In some sources Merkle is mentioned, too.
- The method they suggested was theoretical and unpractical.

Public key cryptography II

- The first practical and public method was RSA which was developed by Rivest, Shamir and Adleman in 1977.
- RSA is still the most popular public key method.
- In 1997 CEG (British cryptographical organisation) published documents that James Ellis had already in 1970 invented public key encryption.
- Similarly, in 1973 Clifford Clocks had described one version of RSA, where the encryption key was the same as the modulus n.
- After RSA, there have been many other suggestions. The most important are:

- Merkle's and Hellman's knapsack. The knapsack problem is NP-complete but anyway it has turned out to be vulnerable. There have been many versions, but only Chor's and Rivest's version has resisted breaking attempts.
- McEliece's method is based on algebraic coding theory.
- Elliptic curve method. An elliptic curve is a second degree polynomial curve defined in the complex plane. Instead of complex numbers, it is possible to use finite fields. In this case the point set is finite, too. This set can be used in encryption. The advantage is a shorter key.

Public key encryption cannot quarantee the confidentiality in every case. If an enemy has the cipher text, he can encrypt every possible clear text with the public key and compare the result with the cipher text. If the result is the same, the clear text has been found. Thus there must be a huge amount of possible clear texts.

RSA

The keys in RSA are as follows:

- public key is the pair (e, n);
- secret key is the pair (d, n);
- a plain text is divided into blocks and the length of one block, as a binary number, must be less that n; thus a block consists at most of $\log_2(n)$ bits.

Encryption is done using the following formula:

$$C = M^e \mod n.$$

Decryption is done by the formula

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

RSA

Before the system is working,

- one has to find suitable numbers e, d and n,
- computations M^e and C^d must be done efficiently with all $M < n$,
- d cannot be deducible easily from e and n.

RSA

Numbers e, d and n are chosen as follows:

1. Generate two large primes p and q.
2. Compute $n = pq$ and $\Phi(n) = (p-1)(q-1)$.
3. Choose a random number e such that $1 < e < \Phi(n)$ and $\gcd(e, \Phi(n)) = 1$.
4. Compute $d = e^{-1} \mod \Phi(n)$.
5. Publish e and n.

RSA

- Because of these selections, we have $(M^e \mod n)^d \mod n = M$.
- In order to show this we need some basic theorems in number theory, as for example Fermat's little theorem.
- These basic results have been presented in many books on computer security.
- Instead, the special cases where $M = p$ or $M = q$ have been passed in most books, but the system works with these values, too. The proof uses the Chinese remainder theorem.

Example

- $p = 101$, $q = 113$, $n = 11413$, $\Phi(n) = 100 \cdot 112 = 11200$.
- Choose first e. Because $11200 = 2^6 5^2 7^1$, then e cannot be divisible by 2, 5 or 7. Let $e = 3533$.
- Then $e^{-1} = 6597$ modulo 11200.
- The public key is $(3533, 11413)$.
- Let $M = 9726$.The cipher text is got by calculating $9726^{3533} \mod 11413 = 5761$.
- Decryption results in the original plain text: $5761^{6597} \mod 11413 = 9726$.

Implementing RSA

Implementing RSA is rather complicated, because many things must be taken into account:

- Primes p and q must be secret, not even parts of these numbers cannot be revealed.
- Low exponents must be avoided.
- Short plaintexts must be preprocessed before encryption.
- Side channel attacks must be taken account, especially in card applications.
- Modulo operations must be done efficiently.

We check every one of these items a little bit more carefully.

RSA: p ja q

Theorem

Let $n = pq$ be m bits. If the first or last $m/4$ bits of p are known, then n can be factored efficiently.

See D.Coppersmith, "Small solutions to polynomial equations, and low exponent RSA vulnerabilities," *J. Cryptology 10 (1997), 233-260.*

RSA: p ja q

Theorem

Let $n = pq$ be m bits. If the first or last $m/4$ bits of p are known, then n can be factored efficiently.

See D.Coppersmith, "Small solutions to polynomial equations, and low exponent RSA vulnerabilities," *J. Cryptology 10 (1997), 233-260.*

Theorem

Assume that (n, e) is public key and that n is m bits. Let d be the decryption key. If one knows the last $m/4$ bits of d, then d can be calculated in linear time with respect to $e \log_2 e$.

See D. Boneh, G.Durfee, and Y. Frankel, "An attack on RSA given a fraction of the private key bits," *Advances in Cryptology - ASIACRYPT '98*, LNCS 1514, Springer-Verlag, 1998, pp.25-34.

Small exponents

- $e = 3$ is a weak value.
- d must be large enough so that the brute force attack does not work.

Theorem

Let p and q be primes and $q < p < 2q$. Let $n = pq$ and let d and e be such that $1 \leq d, e < \phi(n)$, $de \equiv 1$ mod $(p-1)(q-1)$. If now $d < \frac{1}{3}n^{1/4}$, then d can be calculated efficiently in polynomial time with respect to log n.

See Trappe, Washington, Introduction to Cryptography with Coding Theory, Pearson International 2006, pp. 170-171.

Short plaintexts

- Consider the situation where 56 bit DES key is written as a number $m \approx 10^{17}$.
- This number is encrypted with RSA, $c \equiv m^e \pmod{n}$.
- Even if m is small, c is large, about 200 digits.
- An enemy can break the encryption as follows: He makes two lists
  1. $cx^{-e} \pmod{n}$ for all x, $1 \leq x \leq 10^9$.
  2. $y^e \pmod{n}$ for all y, $1 \leq y \leq 10^9$.
- Now he searches for correspondences in both lists. If this kind of correspondence is found, then $cx^{-e} \equiv y^e$ for some x ja y.
- Then $c \equiv (xy)^e \pmod{n}$, so $m \equiv xy \pmod{n}$.

Short plaintexts II

- Is this attack realistic? Assume that m is the product of two numbers x y, and both numbers are less than 10^9.
- In this case these numbers can be found in the lists of the attacker. Not all m are of this form, but many are, and then it is not necessary for the attacker to go through all of 10^{17} possibilities. It is necessary to go through only 2×10^9 calculations and comparisons.
- Preventing this attack: Before encryption, add random bits to the end and start of m forming thus a longer plaintext.

OAEP

- There is a more developed method, Optimal Asymmetric Encryption Padding, OAEP.
- Bellare and Rogaway 1994.
- Assume that A wants to send message m to B, whose RSA key is (n, e), where n is k bits.
- Choose beforehand two positive integers, k_0 and k_1, $k_0 + k_1 < k$.
- A's message can be at most $k - k_0 - k_1$ bits.
- Let G be a function, whose input is a string of k_0 bits and whose output is a string of $k - k_0$ bits.
- Let H be a function, whose input is a string of $k - k_0$ bits and output is a string of k_0 bits. G and H are usually hash functions.

OAEP

The processing and encryption of a plaintext is done as follows:

- $m \mapsto m0^{k_1}$.
- Choose a random string r of k_0 bits.
- $x_1 = m0^{k_1} \oplus G(r)$, $x_2 = r \oplus H(x_1)$.
- If the catenation $x_1||x_2$ as a number is larger than n, A chooses a new r and makes the previous calculations again.
- If $x_1||x_2 < n$, A encrypts: $E(m) = (x_1||x_2)^e \ (\mod n)$.

OAEP

Decryption is done as follows:

- B decrypts the ciphertext and writes the result in the form

$$c^d \pmod{n} = y_1||y_2,$$

  where y_1 is of $k - k_0$ bits and y_2 k_0 bits.
- The B calculates

$$m0^{k_1} = y_1 \oplus G(H(y_1) \oplus y_2).$$

- B takes away k_1 zeros at the end and gets the original plaintext.

Side channel attacks

- Sometimes it is possible to deduce the secret key by observing the time or power consumption used for calculations.
- These kind of attacks are called side channel attacks.
- It is difficult to protect against side channel attacks, because various means must be applied: at the machine level (adding noise, special logic, damping of power source), at the algorithmic level (randomizing) and at the protocol level (changing the keys often enough).
- Side channel attacks must be taken into account especially in card applications.

Finding large primes

- The best method seems to be to generate first large random numbers and to test, if they are primes.
- Testing primes is fast when using randomized algorithms such as Soloway and Strassen or Miller and Rabin tests.
- According to the famous prime number theorem there are about $N/\ln N$ primes between 1 and N. If one is searching a prime of 512 bits, on average it is necessary to generate about 177 numbers before finding a prime.

Generating the encryption key e

- Number e is also generated randomly and after this it is tested, if $\gcd(e, \Phi(n)) = 1$.
- Both the gcd test and the calculation of d can be done at the same time using so called Extended Euclidean algorithm.

Power calculations

- Powers $x^b \mod n$ are calculated as follows:
- First represent b in the binary form $b = \sum_{i=0}^{k} b_i 2^i$, where $b_i = 0$ or 1.
- Use this as the basis of the algorithm:

```
1. z := 1;
2. for i = k downto 0 do
3.     z := (z z) mod n;
4.     if bi = 1 then z := z x mod n end if;
5. end for.
```

Security of RSA

- The security of RSA depends on the fact how fast large numbers can be factorized.
- This problem is equivalent with the square root problem in modular arithmetics.
- Already in 1996 a 130 digit number (431 bits) was factored. The computations were distributed, using hundreds or thousands of computers. The cpu time used was 500 MIPS years.
- At the end of 2003 a number of 174 digits (576 bits) was factored and the factoring team received 10 000 dollars.
- In May 2005 a 200 digit number was factored, but that number did not belong to the prize list.

- Prize was promised to be given to those who factor one 193 digit number (20 000 dollar) and 212 digit number (704 bits, 30 000 dollar).
- At the end of the prize list is one 617 digit number (2048 bits) and the factoring of that number was announced to produce 200 000 dollar. Prizes were given by RSA laboratories to encourage the research in number theory and factoring and to help appliers to deduce suitable key lengths.
- However, prices are no more payed (at least from 2007 onwards).

Requirements to the parameters

At this moment the requirements for the keys are:

- n must be between 1024 and 2048 bits,
- p and q must be near each other, between 10^{75} and 100^{100},
- $p-1$ and $q-1$:n must contain a large prime factor,
- $\mathrm{syt}(p-1, q-1)$ must be small.

Quantum computer

- One threat in the future is quantum computer, because it factors numbers very fast.
- It is not clear, if a realistic quantum computer can be built.
- In 2002 a quantum computer was built which had 7 qubits and which was capable to factor number 15.
- Canadian D-Wave company announced to make a quantum computer for commercial purposes already in 2008. Nothing came of it.
- Besides quantum computers and other previous threats there is still mathematical research which can find quick ways to do factoring. No lower bounds for factoring are known.

Elliptic curve encryption

- Nowadays one hears more and more suggestions that instead of RSA also elliptic curve cryptography should be applied, both in real systems and in teaching.
- It is a little unsure how long elliptic curve encryption is safer that RSA.
- RSA is based on elementary number theory and it is easy to understand. Elliptic curve cryptography demands deeper knowledge on algebra.
- Especially the programs attempting to solve the discreet logarithm problem on elliptic curves apply deep mathematics, for example algebraic geometry. Maybe these methods have not been studied as extensively as the methods of RSA.
- The situation is changing, because elliptic curve methods have been started to use in applications more and more. This has some implications to the teaching of computer security.

Digital signatures with the help of RSA I

- In electronic commerce and when using certificates it is necessary to be able to show that a certain person or organisation is really the sender of the message. The aim of a digital signature is that it shows without doubt the sender, the date of the sending and, in addition, a third party must be able to verify the signature.
- Verification or authentication means, in this context, that a signed message can not been altered with influencing on the signature.
- Direct digital signature is based on public key encryption. One precondition is that the encryption satisfies

$$E_{K_p}(D_{K_s}(M)) = M.$$

  For example, RSA satisfies the formula.
- If the condition is valid, a message is signed by "encrypting" it with the sender's secret key, D_{K_s}.

Digital signatures with the help of RSA II

- The receiver decrypts the message using the sender's public key and checks that the result is the plaintext of the message.
- If the sender wants that nobody except the receiver is able to read the message, the message can be encrypted with the public key of the receiver.

- Assume that an enemy chooses first number y_1 and then makes a message $m_1 = y_1^{e_A}$.
- Now A cannot deny that he has not written the message m_1. On the other hand, it is very probable that m_1 is not a meaningful message. It is likely a random bit string.
- Often, with signatures, hash functions are used. It is the hash which is signed, not the original message.

Other methods of digital signatures

- There are many other methods to do digital signatures.
- ElGamal signature is based on discreet logarithms. It is not very practical, because the signature is quite long.
- Digital Signature Standard is a modification of ElGamal and it is small enough. It was developed partially because RSA was still patented.
- There are many others. See for example the book Menezes, van Oorschot, Vanstone: *Handbook of Applied Cryptography.*

Key generation

- In symmetric encryption, both sides need the same key. In computer networks, the key delivery can be done using a key server and public key encryption.
- If there are no key servers, it is possible to use either key agreement protocols or key generation protocols.
- A key agreement protocol is such that the other participant generates a key and sends it to the other. It is also possible to get a key from a trusted third party in a safe way.
- A key generation protocol is such that the participants do not change keys, but they change only partial information that can be used to generate the keys without outsiders interference..
- In some cases the distinction between the change and generation is unclear. In the following chapter we will present several key agreement protocols. In the following example we consider how to generate keys.

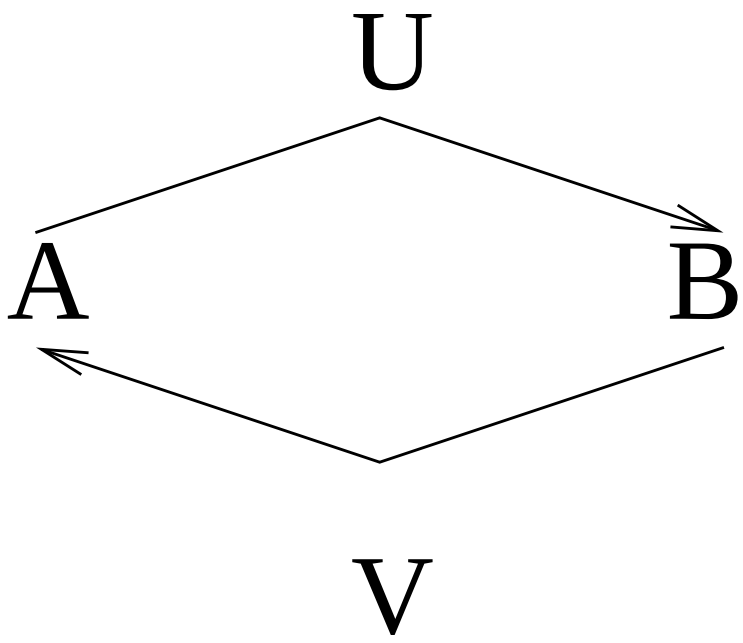
Diffie-Hellman key generation I

- The first and best known key generation protocol is Diffie-Hellman protocol, which is based on the discreet logarithm problem.
- Suppose that p is a prime and α a generator (primitive root) in $\mathbf{Z}_p^*$.
- The values p and α are public and they are used to calculate a common key K, $0 \leq K \leq p-1$, to A and B using the following method.

1. A chooses a random a_A, $0 \leq a_A \leq p-2$;
2. A calculates $\alpha^{a_A} \mod p$ and sends it to B;
3. B chooses a random a_B, $0 \leq a_B \leq p-2$;
4. B calculates $\alpha^{a_B} \mod p$ and sends it to A;
5. A computes the key $K = (\alpha^{a_B})^{a_A} \mod p$,
   B computes the key $K = (\alpha^{a_A})^{a_B} \mod p$.

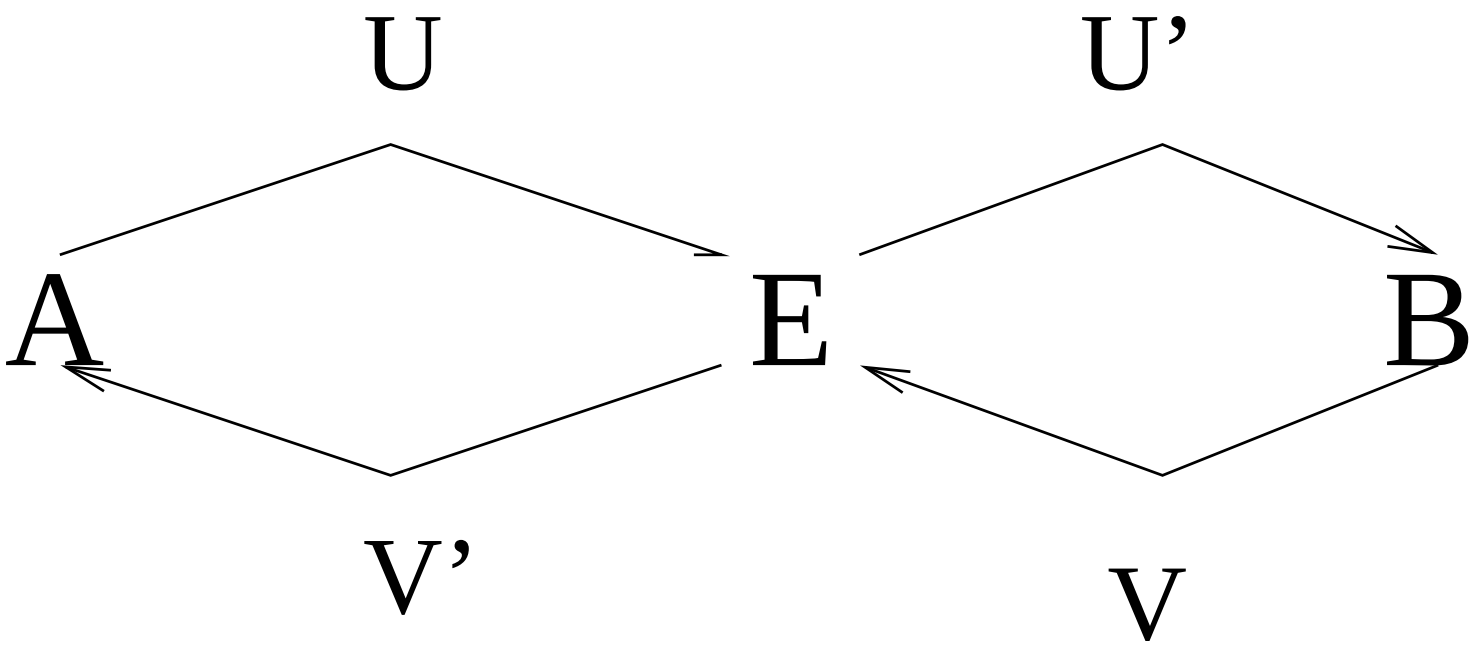
Diffie-Hellman key generation II

The following diagram shows the working of the protocol. In the diagram $U = \alpha^{a_A}$ and $V = \alpha^{a_B}$.



Unfortunately the protocol is vulnerable. An active enemy can use the man-in-the-middle attack as it is shown in the picture:

Diffie-Hellman key generation III



In the diagram $U' = \alpha^{a'_A}$ and $V' = \alpha^{a'_B}$.
Here A has agreed the key with the enemy even if he thinks to agree with B. The same is true with B.

- Clearly it is necessary to know with whom you are communicating.

Diffie-Hellman key generation IV

- Before changing information A and B can acertain each other's indentity using a separate protocol, but this does not help, if E is quet during this time and starts to act only when the key agreement protocol starts.
- Thus it is necessary to take care of authentication and key generation at the same time.
- There are this kind of key agreement protocols and we go through them in the next part.