Complexity and Cryptography

Thomas Zeugmann

Hokkaido University
Laboratory for Algorithmics

https://www-alg.ist.hokudai.ac.jp/~thomas/COCR/

Lecture 13: Public Key Cryptography

Motivation I

Question

Why do we need public key cryptography?

Motivation I

Question

Why do we need public key cryptography?

So far, we mainly considered two-way cryptosystems. In such cryptosystems the security of the communication has been mainly established by a *private key*.

This classical key management requires the *exchange of the secret key* which may be well imaginable if the number of participants is small.

Motivation I

Question

Why do we need public key cryptography?

So far, we mainly considered two-way cryptosystems. In such cryptosystems the security of the communication has been mainly established by a *private key*.

This classical key management requires the *exchange of the secret key* which may be well imaginable if the number of participants is small.

Imagine a bank with tens of thousands customers all over the word who like to access their accounts via their computers at home. It seems absolutely hopeless to exchange frequently secret keys with all customers.

Motivation II

Most customers have a much larger range of applications than simply accessing their bank accounts over the internet, e.g.; shopping over the net using a credit card, frequently exchange of email with varying addressees, or using different computers over a net.

Motivation II

Most customers have a much larger range of applications than simply accessing their bank accounts over the internet, e.g.; shopping over the net using a credit card, frequently exchange of email with varying addressees, or using different computers over a net.

Another important problem is *authentication*. The main problem addressed here is to ensure that a message received indeed originates from the source it pretends to have been sent out.

In classical communication via letters, this problem has been solved by using hand written signatures (in the western hemisphere) or a “hanko,” i.e., a personal seal (e.g., in China, Japan). Thus, we need something equivalent, i.e., an *electronic signature*.

Motivation III

All those real and potential applications stimulated a huge amount of research during the last three decades.

The key observation to be made is that a key in classical two-way cryptosystems has actually two separate tasks, i.e., enciphering and deciphering.

Motivation III

All those real and potential applications stimulated a huge amount of research during the last three decades.

The key observation to be made is that a key in classical two-way cryptosystems has actually two separate tasks, i.e., enciphering and deciphering.

Diffie and Hellman (1976) proposed a new approach, i.e., *public key cryptography*.

They proposed to replace the secrete key by two keys. One key is used for encryption, and one key for decryption. The revolutionary idea, however, was to make the key for encryption *publicly available*. The key for decryption is kept *secret* by the receiver.

The General Scheme of Public Key Cryptography I

The general scenario can be described as follows: Assume ℓ communicating parties $P_1, \ldots, P_\ell$. Each party P_i chooses and publishes its *public key* k_i, and keeps its *secret key* $\widetilde{k}_i$ private.

Suppose, party P_i wishes to send a message to party P_j. Then P_i looks for P_j's public key in the list of all public keys. Next, P_i enciphers its message using k_j and sends it out. The receiver P_j exploits her private key $\widetilde{k}_j$ and deciphers the message received from P_i.

The General Scheme of Public Key Cryptography I

The general scenario can be described as follows: Assume ℓ communicating parties $P_1, \ldots, P_\ell$. Each party P_i chooses and publishes its *public key* k_i, and keeps its *secret key* $\widetilde{k}_i$ private.

Suppose, party P_i wishes to send a message to party P_j. Then P_i looks for P_j's public key in the list of all public keys. Next, P_i enciphers its message using k_j and sends it out. The receiver P_j exploits her private key $\widetilde{k}_j$ and deciphers the message received from P_i.

The only problem is how to realize this nice idea.

The General Scheme of Public Key Cryptography II

There must be some connection between the public and the private key, since otherwise it is very hard to imagine how the deciphering can be performed. This leads to the following

The General Scheme of Public Key Cryptography II

There must be some connection between the public and the private key, since otherwise it is very hard to imagine how the deciphering can be performed. This leads to the following

Requirements:

- Given the public key, it must be extremely hard to compute the private one.
- Computing the cipher must be easy, while deciphering has to remain extremely hard, too, without knowing the private key.

The General Scheme of Public Key Cryptography II

There must be some connection between the public and the private key, since otherwise it is very hard to imagine how the deciphering can be performed. This leads to the following

Requirements:

- Given the public key, it must be extremely hard to compute the private one.
- Computing the cipher must be easy, while deciphering has to remain extremely hard, too, without knowing the private key.

These requirements directly lead to the idea of *one-way functions*.

One-Way Functions

Definition 1 (One-Way Functions)

Let X, Y be non-empty sets. A *one-way function* f is an injective function $f \colon X \to Y$ such that $f(x)$ can be computed in time polynomial in the length $|x|$ of x for all $x \in X$ but there is no algorithm computing $f^{-1}(y)$ efficiently for any interesting fraction of arguments $y \in \text{range}(f)$.

One-Way Functions

Definition 1 (One-Way Functions)

Let X, Y be non-empty sets. A *one-way function* f is an injective function $f \colon X \to Y$ such that $f(x)$ can be computed in time polynomial in the length $|x|$ of x for all $x \in X$ but there is no algorithm computing $f^{-1}(y)$ efficiently for any interesting fraction of arguments $y \in \text{range}(f)$.

Unfortunately, no one has yet proved the existence of one-way functions. Complexity theory is still not ready to handle this extremely difficult problem. Moreover, classical complexity theory mainly deals with *worst-case* complexity what is by no means ideal from the viewpoint of cryptology. The reasons for this are as follows:

Points of Concern I

(1) A problem having high worst-case complexity may be anyway easily solvable for most of its instances. And no *non-linear lower bound* for a particular problem is known.

(2) For all practical purposes it is sufficient to possess an efficient *probabilistic* algorithm computing f^{-1}. That means, even if we would know that no deterministic algorithm computes f^{-1} for almost all inputs in polynomial time, we cannot conclude the relevant cryptosystem to be secure.

(3) Even worse, having a proof that no probabilistic algorithm computes f^{-1} for almost all inputs in polynomial time is not sufficient to derive reasonable conclusions concerning the security of the relevant cryptosystem. Still, it may be possible to invert f for almost all inputs of *practical* length; e.g., given a proved tight lower bound of $n^{\log\log n}$, then for all inputs y of length $|y| \leqslant 2^{2^{10}}$ the inversion of f can be performed in time less than or equal to $|y|^{10}$.

Points of Concern I

(1) A problem having high worst-case complexity may be anyway easily solvable for most of its instances. And no *non-linear lower bound* for a particular problem is known.

(2) For all practical purposes it is sufficient to possess an efficient *probabilistic* algorithm computing f^{-1}. That means, even if we would know that no deterministic algorithm computes f^{-1} for almost all inputs in polynomial time, we cannot conclude the relevant cryptosystem to be secure.

(3) Even worse, having a proof that no probabilistic algorithm computes f^{-1} for almost all inputs in polynomial time is not sufficient to derive reasonable conclusions concerning the security of the relevant cryptosystem. Still, it may be possible to invert f for almost all inputs of *practical* length; e.g., given a proved tight lower bound of $n^{\log\log n}$, then for all inputs y of length $|y| \leqslant 2^{2^{10}}$ the inversion of f can be performed in time less than or equal to $|y|^{10}$.

Points of Concern I

(1) A problem having high worst-case complexity may be anyway easily solvable for most of its instances. And no *non-linear lower bound* for a particular problem is known.

(2) For all practical purposes it is sufficient to possess an efficient *probabilistic* algorithm computing f^{-1}. That means, even if we would know that no deterministic algorithm computes f^{-1} for almost all inputs in polynomial time, we cannot conclude the relevant cryptosystem to be secure.

(3) Even worse, having a proof that no probabilistic algorithm computes f^{-1} for almost all inputs in polynomial time is not sufficient to derive reasonable conclusions concerning the security of the relevant cryptosystem. Still, it may be possible to invert f for almost all inputs of *practical* length; e.g., given a proved tight lower bound of $n^{\log\log n}$, then for all inputs y of length $|y| \leqslant 2^{2^{10}}$ the inversion of f can be performed in time less than or equal to $|y|^{10}$.

Points of Concern II

(4) Since all practically appearing inputs are below some length, even *non-uniform* families of different algorithms inverting f may be interesting for a cryptanalyst.

There are, however some functions f which are widely considered to be good candidates for one-way functions, e.g.; modular exponentiation (the inverse is computing the discrete logarithm), computing the product of prime numbers (the inverse is factoring a given number into its prime factors), and computing $M = \sum_{i=0}^{m} x_i a_i$, where $(a_0, \ldots, a_m) \in \mathbb{N}^{m+1}$ and $(x_0, \ldots, x_m) \in \{0, 1\}^{m+1}$ (the inverse is the general subset-sum problem).

Points of Concern II

(4) Since all practically appearing inputs are below some length, even *non-uniform* families of different algorithms inverting f may be interesting for a cryptanalyst.

There are, however some functions f which are widely considered to be good candidates for one-way functions, e.g.; modular exponentiation (the inverse is computing the discrete logarithm), computing the product of prime numbers (the inverse is factoring a given number into its prime factors), and computing $M = \sum_{i=0}^{m} x_i a_i$, where $(a_0, \ldots, a_m) \in \mathbb{N}^{m+1}$ and $(x_0, \ldots, x_m) \in \{0, 1\}^{m+1}$ (the inverse is the general subset-sum problem).

The General Scheme of Public Key Cryptography III

Next we describe how to apply one-way functions to the solution of public key cryptography.

Clearly, we cannot directly apply one-way functions f for enciphering messages. Additionally, we have to incorporate an idea how the receiver can circumvent the difficulty of inverting f. As outlined above, solely the receiver possesses the additional information provided by her secret key. This additional information should enable her to decrypt the ciphertext.

The General Scheme of Public Key Cryptography III

Next we describe how to apply one-way functions to the solution of public key cryptography.
Clearly, we cannot directly apply one-way functions f for enciphering messages. Additionally, we have to incorporate an idea how the receiver can circumvent the difficulty of inverting f. As outlined above, solely the receiver possesses the additional information provided by her secret key. This additional information should enable her to decrypt the ciphertext.

The following definition formalizes this idea: Furthermore, we use K_p and K_s to denote the set of public keys and private (secret) keys, respectively.

The General Scheme of Public Key Cryptography IV

Definition 2 (Trap-Door Function)

A *trap-door function* $f\colon Pt \times K_p \to Ct$ is a function satisfying the following requirements:

(i) $h_{k_p} = f(\,\cdot\,, k_p)$ is a one-way function for every $k_p \in K_p$;

(ii) there exists a polynomial q such that the time to compute $h_{k_p}(x)$ is uniformly bounded by $q(|x|)$ for all $k_p \in K_p$;

(iii) there exist a one-way function $d\colon K_s \to K_p$ and a polynomial time computable function $g\colon K_s \times Ct \to Pt$ such that $y = f(x, k_p)$ implies $x = g(d^{-1}(k_p), y)$ for all $x \in Pt$, $k_p \in K_p$, and $y \in Ct$.

The General Scheme of Public Key Cryptography V

The information $d^{-1}(k_p)$ constitutes the *trap-door* enabling the receiver to decipher the message obtained. Thus, the general scenario for public key cryptography outlined above can be realized as follows:

The General Scheme of Public Key Cryptography V

The information $d^{-1}(k_p)$ constitutes the *trap-door* enabling the receiver to decipher the message obtained. Thus, the general scenario for public key cryptography outlined above can be realized as follows:

Each party $P_1, \ldots, P_\ell$ is equipped with algorithms for computing f, g, and d. Furthermore, we assume $|K_s| \geqslant \ell$. Now, party P_i chooses its private key $\widetilde{k}_i \in K_s$ such that $\widetilde{k}_i \neq \widetilde{k}_j$ for $i \neq j$, where $i, j \in \{1, \ldots, \ell\}$. How to realize this requirement is discussed later. Then, she computes $k_i = d(\widetilde{k}_i)$ and *publishes* it.

The General Scheme of Public Key Cryptography VI

The message exchange is performed using the following protocol: Suppose P_i wishes to send a message x to P_j.

(1) P_i computes $y = f(x, k_j)$ using P_j's public key k_j, and sends y over a public channel to P_j.

(2) P_j receives y and uses her private key $\widetilde{k}_j$ to compute $x = g(\widetilde{k}_j, y)$.

The General Scheme of Public Key Cryptography VI

The message exchange is performed using the following protocol: Suppose P_i wishes to send a message x to P_j.

(1) P_i computes $y = f(x, k_j)$ using P_j's public key k_j, and sends y over a public channel to P_j.

(2) P_j receives y and uses her private key $\widetilde{k}_j$ to compute $x = g(\widetilde{k}_j, y)$.

We proceed by providing an example for a concrete public key cryptosystem.

Merkle and Hellman's Public Key Cryptosystem I

Within the Merkle and Hellman's public key cryptosystem plaintext is encoded into bit-vectors of length n, i.e., $b = (b_0, \ldots, b_{n-1})$, $b_i \in \{0, 1\}$.

For example, we may encode the 26 letters of the Latin alphabet by using 00000 for A, 00001 for B, $\ldots$, and 11001 for Z. Thus, each letter comprises 5 bits. For error detecting, it may be recommendable to add some check bits using, for example, a Hamming code. For keeping our examples small, we neglect the issue of error detecting here and use the bit strings given above.

Merkle and Hellman's Public Key Cryptosystem II

The public key is a knapsack vector $a = (a_0, \ldots, a_{n-1})$, $a_i \in \mathbb{N}$. How to choose a is described below. The enciphering c of a plaintext b is computed by

$$c = ab^\top = \sum_{j=0}^{n-1} a_j b_j \tag{1}$$

(* $b^\top$ denotes the transpose of b *).

Merkle and Hellman's Public Key Cryptosystem II

The public key is a knapsack vector $a = (a_0, \ldots, a_{n-1})$, $a_i \in \mathbb{N}$. How to choose a is described below. The enciphering c of a plaintext b is computed by

$$c = ab^\top = \sum_{j=0}^{n-1} a_j b_j \qquad (1)$$

(* $b^\top$ denotes the transpose of b *).

The result is a number c between 0 and $\sum_{j=0}^{n-1} a_j$. This number c is represented as a bit string of length $\ell = \lceil \log(1 + \sum_{j=0}^{n-1} a_j) \rceil$ including leading zeros.

Merkle and Hellman's Public Key Cryptosystem III

We describe the trap-door information and how to choose a. The trap-door is a pair $(w, m) \in \mathbb{N} \times \mathbb{N}$ which should be large and has to satisfy

$$w \quad < \quad m \quad \text{and} \quad \gcd(w, m) = 1\,, \tag{2}$$

Next, we choose n values $\hat{a} = (\hat{a}_0, \ldots, \hat{a}_{n-1})$ such that

$$m \quad > \quad \sum_{i=0}^{n-1} \hat{a}_i \quad \text{and} \quad \hat{a}_j > \sum_{i=0}^{j-1} \hat{a}_i \quad \text{for all } j = 1, \ldots, n-1\,. \tag{3}$$

So, party P chooses w, m, and $\hat{a}$ such that Conditions (2) and (3) are satisfied. Then P computes $a = (a_0, \ldots, a_{n-1})$, where $a_i = (\hat{a}_i w) \bmod m$ for all $i = 0, \ldots, n-1$. The vector a is P's public key and the pair (w, m) and the vector $\hat{a}$ is P's private key.

The public key is published, and the private key (w, m) is kept secret.

Merkle and Hellman's Public Key Cryptosystem IV

Let c be a ciphertext received. The deciphering is done using the following procedure *dec*:

(1) Compute $\hat{c} = (w^{-1}c) \bmod m$,

(2) Compute $b = (b_0, \ldots, b_{n-1})$ as follows:

If $\hat{c} \geqslant \hat{a}_{n-1}$ then set $b_{n-1} = 1$ else set $b_{n-1} = 0$

For $j = n-2, n-3, \ldots, 0$, if

$$\hat{c} - \sum_{i=j+1}^{n-1} \hat{a}_i b_i \geqslant \hat{a}_j \text{ then set } b_j = 1 \text{ else set } b_j = 0.$$

Merkle and Hellman's Public Key Cryptosystem V

Example 3

Party P chooses $n = 5$, $(w, m) = (2550, 8443)$, and $\hat{a} = (171, 196, 457, 1191, 2410)$. So Conditions (2) and (3) are satisfied. The modular inverse of 2550 in $\mathbb{Z}_{8443}$ is 3950. So P's public key is $a = (5457, 1663, 216, 6013, 7439)$. Let $x = L$; then $b = (0, 1, 0, 1, 1)$; thus $c = 1663 + 6013 + 7439 = 15115$, and the message sent is c in binary, i.e., 11101100001011. Suppose P has received c. So, P computes successively

Merkle and Hellman's Public Key Cryptosystem V

Example 3

Party P chooses $n = 5$, $(w, m) = (2550, 8443)$, and $\hat{a} = (171, 196, 457, 1191, 2410)$. So Conditions (2) and (3) are satisfied. The modular inverse of 2550 in $\mathbb{Z}_{8443}$ is 3950. So P's public key is $a = (5457, 1663, 216, 6013, 7439)$. Let $x = L$; then $b = (0, 1, 0, 1, 1)$; thus $c = 1663 + 6013 + 7439 = 15115$, and the message sent is c in binary, i.e., 11101100001011.

Suppose P has received c. So, P computes successively

(1) $\hat{c} = (w^{-1}c) \equiv 3950 \cdot 15115 \equiv 3797 \bmod 8443$.

Merkle and Hellman's Public Key Cryptosystem V

Example 3

Party P chooses $n = 5$, $(w, m) = (2550, 8443)$, and $\hat{a} = (171, 196, 457, 1191, 2410)$. So Conditions (2) and (3) are satisfied. The modular inverse of 2550 in $\mathbb{Z}_{8443}$ is 3950. So P's public key is $a = (5457, 1663, 216, 6013, 7439)$. Let $x = L$; then $b = (0, 1, 0, 1, 1)$; thus $c = 1663 + 6013 + 7439 = 15115$, and the message sent is c in binary, i.e., 11101100001011.

Suppose P has received c. So, P computes successively

(1) $\hat{c} = (w^{-1}c) \equiv 3950 \cdot 15115 \equiv 3797 \bmod 8443$.

(2) Since $3797 > 2410$ we set $b_4 = 1$.

Merkle and Hellman's Public Key Cryptosystem V

Example 3

Party P chooses $n = 5$, $(w, m) = (2550, 8443)$, and $\hat{a} = (171, 196, 457, 1191, 2410)$. So Conditions (2) and (3) are satisfied. The modular inverse of 2550 in $\mathbb{Z}_{8443}$ is 3950. So P's public key is $a = (5457, 1663, 216, 6013, 7439)$. Let $x = L$; then $b = (0, 1, 0, 1, 1)$; thus $c = 1663 + 6013 + 7439 = 15115$, and the message sent is c in binary, i.e., 11101100001011.

Suppose P has received c. So, P computes successively

(1) $\hat{c} = (w^{-1}c) \equiv 3950 \cdot 15115 \equiv 3797 \bmod 8443$.

(2) Since $3797 > 2410$ we set $b_4 = 1$.

(3) Next, we compute $3797 - 2410 = 1387 > 1191$. So, $b_3 = 1$.
Now, $3797 - (2410 + 1191) = 196 < 457$, so $b_2 = 0$.
Since $3797 - (2410 + 1191) = 196$ we set $b_1 = 1$.
Finally, $3797 - (2410 + 1191 - 196) = 0$, and thus $b_0 = 0$.

Merkle and Hellman's Public Key Cryptosystem VI

We continue with some remarks concerning the complexity of the problems involved. The difficult problem used in the design of the trap-door function f is the knapsack or subset-sum problem which is known to be $\mathcal{NP}$-complete.

However, several subclasses of this problem are known for which it is easy to solve the decision problem. For example, if $a_i < a_{i+1}$ for all $i = 0, \ldots, n-2$, then the knapsack problem can be solved by using at most n subtractions as outlined in our deciphering procedure. Another subclass consists of all vectors $(a_0, \ldots, a_{n-1}) \in \mathbb{N}^n$ for which all a_i are powers of 2. In the latter case, one has simply to compute the binary representation of M. A more sophisticated subclass has been described in Lagarias and Odlyzko (1983).

Merkle and Hellman's Public Key Cryptosystem VI

We continue with some remarks concerning the complexity of the problems involved. The difficult problem used in the design of the trap-door function f is the knapsack or subset-sum problem which is known to be $\mathcal{NP}$-complete.

However, several subclasses of this problem are known for which it is easy to solve the decision problem. For example, if $a_i < a_{i+1}$ for all $i = 0, \ldots, n-2$, then the knapsack problem can be solved by using at most n subtractions as outlined in our deciphering procedure. Another subclass consists of all vectors $(a_0, \ldots, a_{n-1}) \in \mathbb{N}^n$ for which all a_i are powers of 2. In the latter case, one has simply to compute the binary representation of M. A more sophisticated subclass has been described in Lagarias and Odlyzko (1983).

Merkle and Hellman's Public Key Cryptosystem VII

Thus, special care has to be taken. In particular, Merkle and Hellman (1978) hoped that the modular transformation of the trap-door knapsack $\hat{a}$ will result in almost all cases in a hard one. But A. Shamir (1982) proposed a polynomial time method for breaking the Merkle and Hellman (1978) public key cryptosystem.

Merkle and Hellman's Public Key Cryptosystem VII

Thus, special care has to be taken. In particular, Merkle and Hellman (1978) hoped that the modular transformation of the trap-door knapsack $\hat{a}$ will result in almost all cases in a hard one. But A. Shamir (1982) proposed a polynomial time method for breaking the Merkle and Hellman (1978) public key cryptosystem.

Since then, more sophisticated methods have been proposed to design public key cryptosystem that are based on the difficulty of the subset-sum problem. Since all proposed systems are not very satisfying we continue by looking at the most widely used public key cryptosystems that are based on the difficulty of number theoretic problems.

The RSA Public Key Cryptosystem I

This system is based on the difficulty of factoring and/or taking discrete roots modulo a composite modulus. It has been invented by by R. Rivest, A. Shamir and L. Adleman in 1977.

Let A (= Alice) be anybody wishing to participate at the RSA cryptosystem. Then, Alice has to do the following:

The RSA Public Key Cryptosystem I

This system is based on the difficulty of factoring and/or taking discrete roots modulo a composite modulus. It has been invented by by R. Rivest, A. Shamir and L. Adleman in 1977.

Let A (= Alice) be anybody wishing to participate at the RSA cryptosystem. Then, Alice has to do the following:

(1) Choose randomly two huge primes p_A and q_A (at least 1000 bits each, and the bigger prime should preferably have some more digits than the smaller one).

The RSA Public Key Cryptosystem I

This system is based on the difficulty of factoring and/or taking discrete roots modulo a composite modulus. It has been invented by by R. Rivest, A. Shamir and L. Adleman in 1977.

Let A (= Alice) be anybody wishing to participate at the RSA cryptosystem. Then, Alice has to do the following:

(1) Choose randomly two huge primes p_A and q_A (at least 1000 bits each, and the bigger prime should preferably have some more digits than the smaller one).

(2) Compute $n_A = p_A q_A$ and calculate $\varphi(n_A) = \varphi(p_A)\varphi(q_A) = (p_A - 1)(q_A - 1) = n_A - p_A - q_A + 1$.

The RSA Public Key Cryptosystem I

This system is based on the difficulty of factoring and/or taking discrete roots modulo a composite modulus. It has been invented by by R. Rivest, A. Shamir and L. Adleman in 1977.

Let A (= Alice) be anybody wishing to participate at the RSA cryptosystem. Then, Alice has to do the following:

(1) Choose randomly two huge primes p_A and q_A (at least 1000 bits each, and the bigger prime should preferably have some more digits than the smaller one).

(2) Compute $n_A = p_A q_A$ and calculate $\varphi(n_A) = \varphi(p_A)\varphi(q_A) = (p_A - 1)(q_A - 1) = n_A - p_A - q_A + 1$.

(3) Choose randomly any number $e_A \in \{1, \ldots, \varphi(n_A)\}$ such that $\gcd(e_A, \varphi(n_A)) = 1$.

The RSA Public Key Cryptosystem I

This system is based on the difficulty of factoring and/or taking discrete roots modulo a composite modulus. It has been invented by by R. Rivest, A. Shamir and L. Adleman in 1977.

Let A (= Alice) be anybody wishing to participate at the RSA cryptosystem. Then, Alice has to do the following:

(1) Choose randomly two huge primes p_A and q_A (at least 1000 bits each, and the bigger prime should preferably have some more digits than the smaller one).

(2) Compute $n_A = p_A q_A$ and calculate $\varphi(n_A) = \varphi(p_A)\varphi(q_A) = (p_A - 1)(q_A - 1) = n_A - p_A - q_A + 1$.

(3) Choose randomly any number $e_A \in \{1, \ldots, \varphi(n_A)\}$ such that $\gcd(e_A, \varphi(n_A)) = 1$.

(4) Compute $d_A = e_A^{-1} \bmod \varphi(n_A)$. Publish $K_A = (n_A, e_A)$ and keep p_A, q_A, and d_A secret.

The RSA Public Key Cryptosystem II

Now, assume any other participant B (=Bob) wishing to communicate secretly with Alice.

The RSA Public Key Cryptosystem II

Now, assume any other participant B (=Bob) wishing to communicate secretly with Alice.

Then, Bob codes his plaintext into a binary number w.

The RSA Public Key Cryptosystem II

Now, assume any other participant B (=Bob) wishing to communicate secretly with Alice.

Then, Bob codes his plaintext into a binary number w.

Using Alice's public key $K_A = (n_A, e_A)$ Bob calculates the cipher $c = w^{e_A} \bmod n_A$ and sends c to Alice.

The RSA Public Key Cryptosystem II

Now, assume any other participant B (=Bob) wishing to communicate secretly with Alice.

Then, Bob codes his plaintext into a binary number w.

Using Alice's public key $K_A = (n_A, e_A)$ Bob calculates the cipher $c = w^{e_A} \bmod n_A$ and sends c to Alice.

Alice deciphers c by computing $c^{d_A} \bmod n_A$.

The RSA Public Key Cryptosystem II

Now, assume any other participant B (=Bob) wishing to communicate secretly with Alice.

Then, Bob codes his plaintext into a binary number w.

Using Alice's public key $K_A = (n_A, e_A)$ Bob calculates the cipher $c = w^{e_A} \bmod n_A$ and sends c to Alice.

Alice deciphers c by computing $c^{d_A} \bmod n_A$.

The correctness of this protocol ist established by the following theorem:

The RSA Public Key Cryptosystem II

Now, assume any other participant B (=Bob) wishing to communicate secretly with Alice.

Then, Bob codes his plaintext into a binary number w.

Using Alice's public key $K_A = (n_A, e_A)$ Bob calculates the cipher $c = w^{e_A} \bmod n_A$ and sends c to Alice.

Alice deciphers c by computing $c^{d_A} \bmod n_A$.

The correctness of this protocol ist established by the following theorem:

Theorem 1

$w = c^{d_A} \bmod n_A$.

Proof

Proof. By assumption, $c = w^{e_A} \bmod n_A$. First, assume $\gcd(w, n_A) = 1$; then applying the Theorem of Euler

$$c^{d_A} \equiv (w^{e_A})^{d_A} \equiv w^{e_A d_A} \equiv w^{(e_A d_A) \bmod \varphi(n_A)} \equiv w \bmod n_A \,,$$

since $e_A d_A \equiv 1 \bmod \varphi(n_A)$.

Proof

Proof. By assumption, $c = w^{e_A} \bmod n_A$. First, assume $\gcd(w, n_A) = 1$; then applying the Theorem of Euler

$$c^{d_A} \equiv (w^{e_A})^{d_A} \equiv w^{e_A d_A} \equiv w^{(e_A d_A) \bmod \varphi(n_A)} \equiv w \bmod n_A ,$$

since $e_A d_A \equiv 1 \bmod \varphi(n_A)$.

What can be said if $\gcd(w, n_A) \neq 1$?

Proof

Proof. By assumption, $c = w^{e_A} \bmod n_A$. First, assume $\gcd(w, n_A) = 1$; then applying the Theorem of Euler

$$c^{d_A} \equiv (w^{e_A})^{d_A} \equiv w^{e_A d_A} \equiv w^{(e_A d_A) \bmod \varphi(n_A)} \equiv w \bmod n_A\,,$$

since $e_A d_A \equiv 1 \bmod \varphi(n_A)$.

What can be said if $\gcd(w, n_A) \neq 1$?

Suppose that exactly one of the two primes p_A and q_A does divide w, say p_A. The Little Theorem of Fermat is telling us

$$w^{q_A - 1} \equiv 1 \bmod q_A\,.$$

Since $\varphi(n_A) = (p_A - 1)(q_A - 1)$, we can conclude

$$w^{\varphi(n_A)} \equiv (w^{q_A-1})^{p_A-1} \equiv 1^{p_A-1} \equiv 1 \bmod q_A\,.$$

Moreover, $e_A d_A \equiv 1 \bmod \varphi(n_A)$, and thus $e_A d_A = j\varphi(n_A) + 1$ for some positive integer j.

Proof – continued

Consequently,

$$w^{e_A d_A} \equiv w \bmod q_A \ .$$

But the last congruence is also true modulo p_A, since, by assumption, p_A divides w, and thus $w^{e_A d_A} - w \equiv 0 \bmod p_A$. Hence, we can conclude $w^{e_A d_A} \equiv w \bmod n_A$.

Proof – continued

Consequently,

$$w^{e_A d_A} \equiv w \bmod q_A \ .$$

But the last congruence is also true modulo p_A, since, by assumption, p_A divides w, and thus $w^{e_A d_A} - w \equiv 0 \bmod p_A$. Hence, we can conclude $w^{e_A d_A} \equiv w \bmod n_A$.

Finally, if both p_A and q_A divide w, then we trivially have $w^{e_A d_A} - w \equiv 0 \bmod p_A$ as well as $w^{e_A d_A} - w \equiv 0 \bmod q_A$, and thus $w^{e_A d_A} \equiv w \bmod n_A$. ∎

The RSA Public Key Cryptosystem III

Question

What can be said concerning the security of the RSA cryptosystem?

The RSA Public Key Cryptosystem III

Question

What can be said concerning the security of the RSA cryptosystem?

By its construction, breaking the RSA cipher is as most as hard as finding discrete roots modulo the composite number n_A. Computing discrete roots must be judged as feasible if the prime factorization of n_A is known. Therefore, the cryptanalysis of the RSA cryptosystem is as most as hard as factoring. However, there is no known efficient algorithm for factoring a large composite number except for quantum computers which are currently not available.

The RSA Public Key Cryptosystem IV

However, some care must be taken be choosing the primes p and q. Obviously, the chosen primes should be nowhere listed. Thus, testing primality is such an important problem.

The RSA Public Key Cryptosystem IV

However, some care must be taken be choosing the primes p and q. Obviously, the chosen primes should be nowhere listed. Thus, testing primality is such an important problem.

Moreover, one has to avoid primes of special form, e.g., $p = 2^e \pm 1$.

The RSA Public Key Cryptosystem IV

However, some care must be taken be choosing the primes p and q. Obviously, the chosen primes should be nowhere listed. Thus, testing primality is such an important problem.

Moreover, one has to avoid primes of special form, e.g., $p = 2^e \pm 1$.

As mentioned above, the difference between p and q should be large. For seeing this, we prove the following theorem:

The RSA Public Key Cryptosystem IV

However, some care must be taken be choosing the primes p and q. Obviously, the chosen primes should be nowhere listed. Thus, testing primality is such an important problem.

Moreover, one has to avoid primes of special form, e.g., $p = 2^e \pm 1$.

As mentioned above, the difference between p and q should be large. For seeing this, we prove the following theorem:

Theorem 2

Let p and q be primes such that $p > q$ and $p - q = O((\log p)^c)$ for a "moderate" constant $c \in \mathbb{N}$. Then, there exists an efficient algorithm for factoring $n = pq$.

The RSA Public Key Cryptosystem V

Proof. Consider

$$\frac{(p+q)^2}{4} - n = \frac{p^2 + 2pq + q^2 - 4pq}{4} = \frac{(p-q)^2}{4} \tag{4}$$

Thus, the left side is a perfect square. Then the following method may be applied for factoring n:

The RSA Public Key Cryptosystem V

Proof. Consider

$$\frac{(p+q)^2}{4} - n = \frac{p^2 + 2pq + q^2 - 4pq}{4} = \frac{(p-q)^2}{4} \tag{4}$$

Thus, the left side is a perfect square. Then the following method may be applied for factoring n:

(1) Compute $\sqrt{n}$ within a precision of $\lfloor(\log n + 2)/2\rfloor$ bits.

The RSA Public Key Cryptosystem V

Proof. Consider

$$\frac{(p+q)^2}{4} - n = \frac{p^2 + 2pq + q^2 - 4pq}{4} = \frac{(p-q)^2}{4} \tag{4}$$

Thus, the left side is a perfect square. Then the following method may be applied for factoring n:

(1) Compute $\sqrt{n}$ within a precision of $\lfloor (\log n + 2)/2 \rfloor$ bits.

(2) Check for $x = \lfloor \sqrt{n} \rfloor + 1,\ \lfloor \sqrt{n} \rfloor + 2,\ \ldots$ whether or not $x^2 - n$ is a perfect square.
If it is, output $p = x + \sqrt{x^2 - n}$ and $q = x - \sqrt{x^2 - n}$.

The RSA Public Key Cryptosystem VI

The correctness of the above algorithm can be shown as follows: First, assume that $\sqrt{n}$ has been computed within a precision of $\lfloor(\log n + 2)/2\rfloor$ bits. Then we have $\lfloor\sqrt{n}\rfloor$. (Exercise). Next, observe that

$$\frac{p+q}{2} > \sqrt{pq} = \sqrt{n} \tag{5}$$

by applying the well-known inequality between arithmetic and geometric mean. Consequently, $\lfloor\sqrt{n}\rfloor < (p+q)/2$ (* note that $(p+q)/2$ is always an integer *). Thus, the algorithm must terminate by finding a perfect square and the first x found must fulfill $x^2 \leqslant \frac{(p+q)^2}{4}$ by (4) and (5). Thus, $x \leqslant \frac{p+q}{2}$.

The RSA Public Key Cryptosystem VII

Case 1. $x = \dfrac{p+q}{2}$.

Let $z^2 = x^2 - n$. By (4) we directly obtain $z = \frac{p-q}{2}$. Hence, $x + z = p$ and $x - z = q$.

The RSA Public Key Cryptosystem VII

Case 1. $x \;=\; \dfrac{p+q}{2}$.

Let $z^2 = x^2 - n$. By (4) we directly obtain $z = \frac{p-q}{2}$. Hence, $x + z = p$ and $x - z = q$.

Case 2. $x \;<\; \dfrac{p+q}{2}$.

Since $z^2 = x^2 - n$ we obtain $n = x^2 - z^2 = (x+z)(x-z)$. Thus, $p = x + z$ and $q = x - z$, and therefore, $(p+q)/2 = x$, a contradiction.

This proves the correctness.

The RSA Public Key Cryptosystem VII

Case 1. $x \;=\; \dfrac{p+q}{2}$.

Let $z^2 = x^2 - n$. By (4) we directly obtain $z = \frac{p-q}{2}$. Hence, $x + z = p$ and $x - z = q$.

Case 2. $x \;<\; \dfrac{p+q}{2}$.

Since $z^2 = x^2 - n$ we obtain $n = x^2 - z^2 = (x+z)(x-z)$. Thus, $p = x + z$ and $q = x - z$, and therefore, $(p+q)/2 = x$, a contradiction.

This proves the correctness.

Finally, we show that the algorithm above is efficient.

The RSA Public Key Cryptosystem VIII

The computation of $\sqrt{n}$ up to the desired precision can be easily performed using the Newton method, i.e.,

$$x_{\ell+1} := x_\ell - \frac{x_\ell^2 - n}{2x_\ell} \qquad \text{for } \ell = 0, \ldots, \lfloor (\log n + 2)/2 \rfloor$$

with $x_0 = n$.

The RSA Public Key Cryptosystem VIII

The computation of $\sqrt{n}$ up to the desired precision can be easily performed using the Newton method, i.e.,

$$x_{\ell+1} := x_\ell - \frac{x_\ell^2 - n}{2x_\ell} \quad \text{for } \ell = 0, \ldots, \lfloor (\log n + 2)/2 \rfloor$$

with $x_0 = n$. Taking into account that $\sqrt{n} > q = p - (p - q) \approx p - (\log p)^c$ we see by (4) that

$$\frac{p+q}{2} = p - \frac{p-q}{2} \approx p - \frac{(\log p)^c}{2} > \sqrt{n}. \tag{6}$$

Hence, $(p+q)/2$ is only "a bit" greater than $\sqrt{n}$. Consequently, the algorithm has to try at most $(\log p)^c/2$ many candidates until its search terminates. But this is a polynomial in the length of the input, and hence we are done. ∎

The Diffie–Hellman Public Key Cryptosystem I

Finally, we take a look at the Diffie–Hellman Public Key Cryptosystem. It is based on discrete logarithms.

Problem: Discrete Logarithms

Input: Prime number $p \in \mathbb{N}$, $b \in \mathbb{Z}_p^*$, and a generator g for $\mathbb{Z}_p^*$.

Problem: Compute the index x such that $x = \mathrm{dlog}_g\, b$.

The Diffie–Hellman Public Key Cryptosystem I

Finally, we take a look at the Diffie–Hellman Public Key Cryptosystem. It is based on discrete logarithms.

Problem: Discrete Logarithms

Input: Prime number $p \in \mathbb{N}$, $b \in \mathbb{Z}_p^*$, and a generator g for $\mathbb{Z}_p^*$.

Problem: Compute the index x such that $x = \operatorname{dlog}_g b$.

So far, there is no known algorithm efficiently computing discrete logarithms for any standard model of sequential or parallel computation. On the other hand, a quantum computer would be able to compute discrete logarithms in polynomial time.

The Diffie–Hellman Public Key Cryptosystem II

The Diffie–Hellman Public Key Cryptosystem requires a system designer. The system designer chooses a huge prime q (preferably more than 4000 bits) and a generator g for $\mathbb{Z}_q^*$. The prime q and the generator g are global information, and thus known to everybody participating in this public key cryptosystem.

The Diffie–Hellman Public Key Cryptosystem II

The Diffie–Hellman Public Key Cryptosystem requires a system designer. The system designer chooses a huge prime q (preferably more than 4000 bits) and a generator g for $\mathbb{Z}_q^*$. The prime q and the generator g are global information, and thus known to everybody participating in this public key cryptosystem.

Next, we describe how the public and the secret key, respectively, are chosen by any participant. Subsequently, we provide the protocol used.

The Diffie–Hellman Public Key Cryptosystem III

Let A be any participant.

A chooses randomly a number $x_A \in \{2, \ldots, q-1\}$ and computes $y_A = g^{x_A} \bmod q$. That is, $x_A = \mathrm{dlog}_g y_A$.

Participant A publishes y_A as her public key and keeps x_A secret.

The Diffie–Hellman Public Key Cryptosystem III

Let A be any participant.
A chooses randomly a number $x_A \in \{2, \dots, q-1\}$ and computes $y_A = g^{x_A} \bmod q$. That is, $x_A = \mathrm{dlog}_g y_A$.
Participant A publishes y_A as her public key and keeps x_A secret.
Now, let A and B be any two participants who wish to communicate. Assume, B likes to send a message to A. Then the following key is used:
B computes the key $K_{AB} = y_A^{x_B} \bmod q$.
Assume, A likes to send a message to B. Then, A computes $K_{BA} = y_B^{x_A} \bmod q$.

The Diffie–Hellman Public Key Cryptosystem IV

The following theorem shows the usefulness of these keys:

Theorem 3

$K_{AB} = K_{BA}$

The Diffie–Hellman Public Key Cryptosystem IV

The following theorem shows the usefulness of these keys:

Theorem 3

$K_{AB} = K_{BA}$

Proof.

$$
\begin{aligned}
K_{AB} &= y_A^{x_B} \equiv (g^{x_A})^{x_B} \equiv g^{x_A x_B} \\
&\equiv (g^{x_B})^{x_A} \equiv y_B^{x_A} \equiv K_{BA} \bmod q\,.
\end{aligned}
$$

$\blacksquare$

Remarks

Note, however, that A and B compute K_{BA} and K_{AB}, respectively, using *different* information. That is, A computes K_{BA} using her own secret key x_A and the public key y_B published by B while Bob calculates K_{BA} from his secret key x_B and Alice's public key y_A.

Remarks

Note, however, that A and B compute K_{BA} and K_{AB}, respectively, using *different* information. That is, A computes K_{BA} using her own secret key x_A and the public key y_B published by B while Bob calculates K_{BA} from his secret key x_B and Alice's public key y_A.

On the other hand, any cryptanalyst does neither possess x_A nor x_B, hence she must compute K_{BA} solely from y_A and y_B. Since $K_{AB} = y_A^{\mathrm{dlog}_g y_B} \bmod q$ this is at most as hard as computing discrete logarithms. Moreover, so far no easier method is known for computing K_{BA} from y_A and y_B. Therefore, it is widely believed that computing K_{BA} from y_A and y_B is hard.

The Diffie–Hellman Public Key Cryptosystem V

Next, we describe how the parties A and B can communicate. Taking Theorem 3 into account, two possibilities are imaginable. First, the system designer additionally provides any sufficiently advanced classical two-way cryptosystem, for example the AES. Then, any pair of users wishing to secretly communicate may use the key K_{BA}. Thus, *no key exchange* is required in *advance*.

The Diffie–Hellman Public Key Cryptosystem V

Next, we describe how the parties A and B can communicate. Taking Theorem 3 into account, two possibilities are imaginable. First, the system designer additionally provides any sufficiently advanced classical two-way cryptosystem, for example the AES. Then, any pair of users wishing to secretly communicate may use the key K_{BA}. Thus, *no key exchange* is required in *advance*.

Note that public key cryptosystems are relatively slow compared to classical cryptosystems (at least to our present stage of technology and theoretical knowledge). Thus, it is sometimes more realistic to use them in the limited role in conjunction with a classical cryptosystem in which the actual messages are transmitted as described above.

The Diffie–Hellman Public Key Cryptosystem VI

Second, the communication is performed by using directly the Diffie–Hellman system. The underlying idea is best explained using a bag having two locks.

The Diffie–Hellman Public Key Cryptosystem VI

Second, the communication is performed by using directly the Diffie–Hellman system. The underlying idea is best explained using a bag having two locks.

Each partner possesses exclusively one key for one of the two locks. Initially, both locks are unlocked. Now, A puts the message w into the bag and locks *her lock* with *her key*. The bag is taken by a messenger who delivers the bag to B. Obviously, B is *not* able to *unlock* the bag right now, since he possesses only the key for the other lock. Therefore, B locks *his* lock using *his* key, and returns the bag to the messenger who is returning it to A. Now, A may *unlock* her lock, but the bag remains anyway *locked*. Finally, the messenger delivers the bag again to B. Now, B may *unlock* the bag using *his* key, and thus, he finally has access to the secret message w.

The Diffie–Hellman Public Key Cryptosystem VII

The protocol described above has the following advantage: The public messenger always delivered a locked bag. On the other hand, A and B could exchange a secret message without *exchanging* any *key*.

The Diffie–Hellman Public Key Cryptosystem VII

The protocol described above has the following advantage: The public messenger always delivered a locked bag. On the other hand, A and B could exchange a secret message without *exchanging* any *key*.

The impact of this method can be hardly overestimated. Looking back into the history of cryptography, we see that the cryptography community unanimously agreed, for thousands of years, that the only way for two parties to establish secure communications was to first exchange a secret key. This was so much common wisdom that nobody questioned it. If the recipient did not have a secret key giving her the information needed to decrypt the message efficiently, how could she be in a better position than an eavesdropper?

The Diffie–Hellman Public Key Cryptosystem VIII

Now, we outline the formal realization of the idea described above to use a bag with two locks. For that purpose, a small modification of the choices for x_A and x_B, respectively, has to be made. Again, A and B randomly choose a number x_A and x_B between 2 and $q - 1$.

The Diffie–Hellman Public Key Cryptosystem VIII

Now, we outline the formal realization of the idea described above to use a bag with two locks. For that purpose, a small modification of the choices for x_A and x_B, respectively, has to be made. Again, A and B randomly choose a number x_A and x_B between 2 and $q - 1$. *Additionally*, they must ensure that $\gcd(x_A, q - 1) = 1$ and $\gcd(x_B, q - 1) = 1$, respectively. This can be easily done by using the Euclidean algorithm, i.e., if the randomly chosen number does not fulfill this requirement a new number is randomly chosen until one is found that is relatively prime to $q - 1$. Moreover, as shown previously the Euclidean algorithm can be also used for computing $x_A^{-1} \bmod (q - 1)$ and $x_B^{-1} \bmod (q - 1)$, respectively.

The Diffie–Hellman Public Key Cryptosystem IX

Suppose, A wishes to send B a message, and let w be A's plaintext.

(i) A sends $w^{x_A} \bmod q$ to B,

(ii) B returns $w^{x_A x_B} \bmod q$ to A,

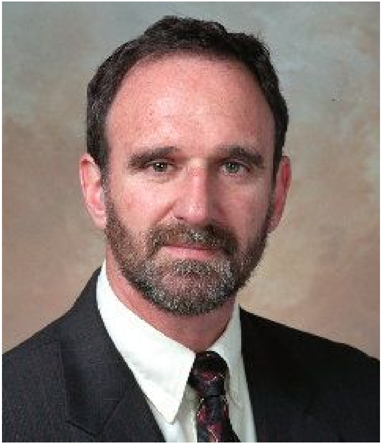
(iii) A computes $\hat{w} \equiv (w^{x_A x_B})^{x_A^{-1}} \bmod q$ and sends the result $\hat{w}$ to B,

(iv) B deciphers $\hat{w}$ by calculating $\hat{w}^{x_B^{-1}} \bmod q$.

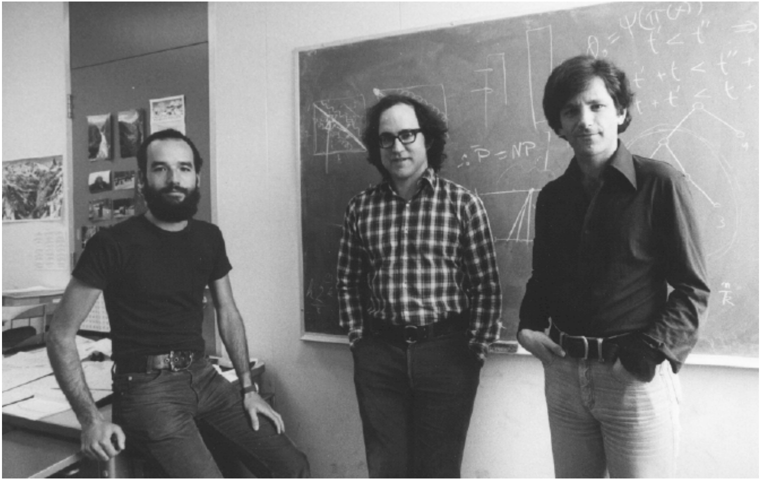
The correctness of the above algorithm is an immediate consequence of the Theorem of Euler.

Thank you!

Ralph C. Merkle

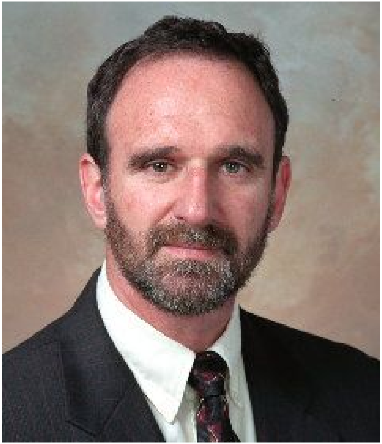


Martin Hellman



Adi Shamir, Ron Rivest, Leonard Adleman

Whitfield **Diffie**



Martin **Hellman**