A one-way function from thermodynamics and applications to cryptography

Norbert Hungerbühler[1] and Michael Struwe[2]

Abstract: In the past two decades cryptological methods have become increasingly indispensable tools to ensure security of electronic data and their transmission. Simultaneously with the development of the mathematical methods which ensure safe electronic communication, more and more sophisticated techniques emerged which allow to attack codes on increasingly powerful computers. Modern cryptosystems are based on number theory, commutative group theory, or algebraic geometry. The Diffie-Hellman key exchange protocol and the RSA public key method are considered particularly secure and are widespread in electronic banking. However, in the near future even such encryption methods might become vulnerable to factorization algorithms on quantum computers. To counteract this threat quantum cryptographic methods may need to be derived. Here, we propose a simpler public key cryptosystem for the quantum age whose security is based upon the second principle of thermodynamics. We use the fact that the heat equation is ill-posed in backward time direction.

Zusammenfassung: Die Sicherheit von elektronischen Daten und deren Übermittlung beruht heutzutage auf ausgeklügelten kryptologischen Methoden, die z. B. im e-Banking Verwendung finden. Im gleichen Maße, wie immer sicherere mathematische Methoden entwickelt wurden, um vertrauliche elektronische Kommunikation zu schützen, sind auch immer raffiniertere Techniken entstanden, um diese Codes mit Hilfe immer leistungsfähigerer Computer zu brechen. Moderne Verschlüsselungsmethoden basieren unter anderem auf zahlentheoretischen Grundlagen, auf der Theorie kommutativer Gruppen und auf algebraischer Geometrie. Die als besonders sicher geltenden Protokolle von Diffie und Hellman und das RSA-Public Key Verfahren scheinen in nicht allzu ferner Zukunft durch Faktorisierungsmethoden auf Quantencomputern bedroht. Als Abhilfe werden Chiffrierungsmethoden erprobt, die ihrerseits Quanteneffekte ausnützen. Im Unterschied dazu beruht die Sicherheit des hier vorgestellten Public Key Verfahrens auf dem zweiten Hauptsatz der Thermodynamik. Dabei wird benutzt, dass die Wärmeleitungsgleichung nur in einer Zeitrichtung lösbar ist.

1 Introduction

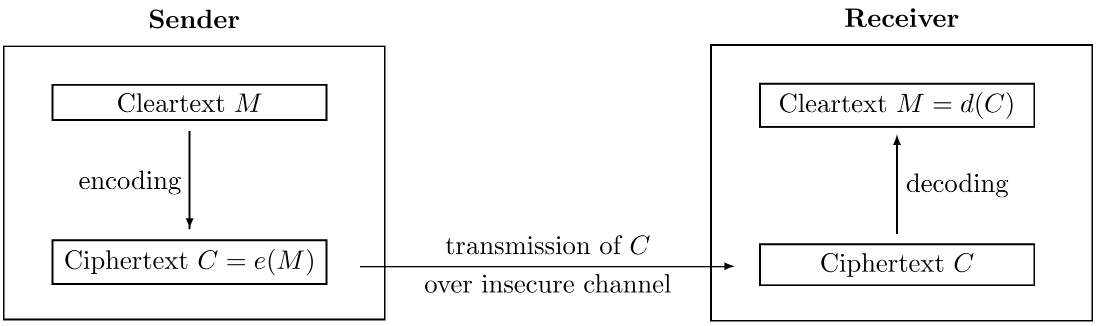
1.1 D elw ri klvwrub

Suppose someone intends to send a message to someone else and wants to make sure an intermediary cannot intercept and read the message, intercept and modify the message, or build a realistic-looking substitute message. The receiver, on the other hand, wants to be sure that the message is really from this sender and not from (or modified by) a third party. This is the general problem cryptography has to solve. Another way to describe the scenario is the following: How can two people talk in public, such that everybody can listen to their conversation, and such that in the end the two people have exchanged a certain amount of information, but no other listener has gained this information.

[1]Département de Mathématiques, Université de Fribourg, Pérolles, CH-1700 Fribourg.
norbert.hungerbuehler@unifr.ch
NH is supported in part by the Swiss National Science Foundation, grant 2100-063464.00/1.
[2]Departement Mathematik, ETH Zürich, CH-8092 Zürich. michael.struwe@math.ethz.ch

The answer is surprisingly simple: The two people who want to exchange information have to use a language, which nobody else understands. In order to have tap-proof radio communication, the little known native Indian language of the Navajo was actually used by the US in World War II, especially in the battle of Iwo Jima. The method works as long as no third party understands the language or is in possession of a dictionary.

The classical cryptographic approach is that the sender and the receiver of a message have, in advance, to agree on a cipher: A cipher consists of two functions, an injective function e (called encoding) and its inverse d (called decoding) such that every cleartext message M can be encrypted into a ciphertext $C = e(M)$ to hide its substance, and by the receiver decrypted using $d(e(M)) = M$. So, a cipher can be interpreted as a bilingual dictionary. The following figure shows the situation schematically:



A very simple example is Caesar's cipher (used by Julius Caesar during the Gallic wars to communicate with his generals), where encryption consists of replacing every letter in M by the letter which is located (cyclically) three places ahead in the used alphabet:

$$\begin{array}{ccc} \texttt{A} & \underset{d}{\overset{e}{\rightleftarrows}} & \texttt{D} \\ \texttt{B} & \underset{d}{\overset{e}{\rightleftarrows}} & \texttt{E} \\ \vdots & \vdots & \vdots \\ \texttt{Z} & \underset{d}{\overset{e}{\rightleftarrows}} & \texttt{C} \end{array}$$

So, e.g., $e(\texttt{CAESAR}) = \texttt{FDHVDU}$. The title of Section 1.1 is encoded this way. Caesar's cipher is the forefather of all shift cipher systems.

Since the Spartan army used the stick cipher method[3], the development of cryptography was

[3] The Spartans enciphered and concealed a message by using a scytale, a special stick and belt. The encipherer would wrap the belt around the stick and write a message lengthwise on it. The belt was then unwound from the stick and sent to another person. Using a stick of similar size, the decipherer would wrap the belt around the stick to watch the secret message appear. If a stick of the wrong size was used, the message would be scrambled. This is an example of a permutation cipher.

dominated by its use in military organizations and secret services (and still much of the terminology used in cryptography has military origin). In World War II, a group of mathematicians, working in Bletchley Park and lead by Alan M. Turing, succeeded to break the German *Enigma* cryptosystem. This was a crucial advantage for the Allied forces in the further process of war. (For the full story see, e.g., [25], [26] or [16].) Reminiscent of these days, import, export and domestic use of cryptographic techniques was tightly controlled in many countries. For example, export of so called strong cryptographic technology was prohibited in the United States until 2000, and in France until early 1999 it was not allowed to encode personal documents. Today, most countries have signed the *Wassenaar Arrangement*, which regulates these questions in a more liberal way. With the dawn of electronic data transmission, cryptography is no longer only a concern of the army: Whenever sensitive or personal data are electronically transferred over the world wide web, cryptosystems are used. Cryptographic systems are the mathematical guard dogs that keep credit card numbers safe, medical information secure, and your on-line banking transactions confidential.

That we should not blindly trust cryptographic methods was drastically shown in October 2000: Then a team lead by Fredrik Almgren succeeded to break a 512-bit RSA code (see Section 1.3) which was considered secure until a few years ago and in use even these days (today 1024 bits are recommended). To break this code was the hardest part in Simon Singh's "Cipher Challenge", a competition announced in his book [25]. The challenge consisted in decrypting a set of ten messages that were encyphered by ten different methods whose complexity increased from stage to stage. The competition began in September 1999 and Singh offered £1000 to the cryptanalyst (or the team) leading after one year. He also offered £10'000 to the first person or team to crack all ten codes. On October 1, 2000, Jim Gillogly, John Palagyi and the EFF team (Electronic Frontier Foundation) received the £1000 for having succeeded in the first nine problems. Only one week later, Almgren's team solved the tenth problem, encrypted with RSA and DES. In particular, they were able to adapt the so-called General Number Field Sieve (GNFS) algorithm in such an efficient way that not even a Cray supercomputer or a vector computer was necessary to break the code[4]. The full solution of the Cipher Challenge can be found in [2].

1.2 Cryptosystems with keys

Traditionally, in cryptography the sender is called Alice, the receiver Bob and the opponent Oscar (or Eve from eavesdropper). The general assumption is that Eve knows the cryptosystem being used. This is usually referred to as *Kerckhoff's principle*. Of course, if Eve does not know the cryptosystem, that will make her task, namely to break the cipher, more difficult. However, one does not want to base the security of a code on this premise.

Obviously, a shift cipher is very easy to break, even if the shift itself is not known (just try all shifts of the used alphabet). More sophisticated cryptographic methods are substitution cipher, affine cipher, Vigenère cipher, Hill cipher, permutation cipher etc. For a description of the mentioned cryptosystems, and many more, see [25], [7], [19], [14], [15], [17], [18], [20], [23], [27], [28], and [29]. In the mentioned references, also the corresponding cryptanalysis (i.e. the art to break these codes) is exhaustively described.

The idea to make it harder to break a code is to use a *key* κ such that encryption e_κ and decryption d_κ depend on κ [5]. To break the cryptosystem, Eve would have to find out the value of the key, either by guessing or exhaustive search (which is hopeless if the key space is large enough), or by analyzing the structure of the encrypted data and try to find a clue on the cleartext or the

[4] Actually, a 512-bit number was factorized already in spring 2000 by a group at the CWI, the *Centrum voor Wiskunde en Informatica* in Amsterdam, on a 16-processor Cray C90.

[5] In asymmetric systems, the keys for encryption and decryption are different.

key. Suppose, for example, that we want to encrypt a text M of length n, written with the 26 letters of the English alphabet, each represented by a number between 0 and 25. We may encrypt $M = (m_1, \ldots, m_n)$ in the following way: We choose a key $\kappa = (\kappa_1, \ldots, \kappa_n)$ and define $C = e_\kappa(M) = (c_1, \ldots, c_n)$ by

$$c_i = m_i + \kappa_i \quad \text{in } \mathbb{Z}_{26}.$$

Decryption is given by

$$m_i = c_i - \kappa_i \quad \text{in } \mathbb{Z}_{26}.$$

Since the key κ is used only once, for the particular message M, this system is called one-time pad. G. S. Vernam proposed in 1926 to use for κ a random sequence of the elements of $\{0, \ldots, 25\}$, each with the probability $\frac{1}{26}$ and chosen independently. The resulting cipher is an example of an unconditionally secure cryptosystem, since the data stream of C does not possess any structure reminiscent of M and can therefore not be broken, even with unlimited computational resources. In spite of its security, the system has two major drawbacks: There is no mathematical way of generating independent random numbers, and (more serious from a practical point of view) Alice and Bob have first to meet to agree on κ. This may be workable on a limited number of occasions, but becomes virtually impracticable for large-scale application of cryptography like in e-banking or for use in the army. This difficulty of sharing keys is referred to as the *key distribution problem.*

1.3 Public key cryptosystems

Public key cryptography tries to overcome the previously mentioned difficulty and to find a way that Alice and Bob can agree on a key over an insecure channel or even in public without any prior secrets. To illustrate the general idea, consider the following situation: Alice wants to send a secret message to Bob. Before sending the message, enclosed in metal box, she receives from Bob an open lock (representing Bob's public key) which she uses to seal the box. Only Bob knows the combination (representing Bob's secret key) to unlock it, i.e. even Alice cannot reopen the box with her message which she then sends to Bob. In particular, Alice's message can consist of a key κ to use for a future transmission.

A mathematical version of the described procedure was first proposed 1976 by Diffie and Hellman in [11] (see also [10]): The system has two public parameters, a prime number p and a primitive element α which generates the finite field $GF(p)$ in the sense that for every number $n \in \{1, \ldots, p-1\}$ there exists a unique $k \in \{1, \ldots, p-1\}$ such that $n \equiv \alpha^k \mod p$. If Alice and Bob now want to agree on a key over an insecure channel or in public, they proceed as follows:

1. First Alice chooses a private (random) value $x_A \in \{2, \ldots, p-2\}$ [6] and Bob chooses a value $x_B \in \{2, \ldots, p-2\}$. Both, Alice and Bob, keep their values secret.
2. Then Alice and Bob publish the values
$$y_A = \alpha^{x_A} \text{ modulo } p \text{ and } y_B = \alpha^{x_B} \text{ modulo } p.$$
3. Finally Alice and Bob compute $y_B^{x_A}$ modulo p and $y_A^{x_B}$ modulo p respectively. Since
$$\kappa := y_B^{x_A} \equiv \alpha^{x_B x_A} \equiv \alpha^{x_A x_B} \equiv y_A^{x_B} \mod p,$$
Alice and Bob have now the shared secret key κ.

[6] $p-1$ is to be excluded because of Fermat's little theorem.

This protocol depends on the practical infeasibility to compute the discrete logarithm in $GF(p)$. In other words, if an eavesdropper knows p, α, y_A and y_B, it is difficult or virtually impossible for him to compute x_A or x_B (and hence κ) if p is a large prime. Today, a length of 1024 bits for the prime p is considered to guarantee a good level of security.

Notice however, that the public values y_A and y_B have to be published in a trusted directory or in a certified document that associates a person with a specific public key. Differently stated: if the values y_A and y_B are exchanged over an insecure channel, then the procedure is vulnerable to a so called man-in-the-middle attack, where an intruder intercepts y_A and y_B and replaces them by his own values $y_{A'}$ and $y_{B'}$.

Using a similar idea as Diffie and Hellman, Rivest, Shamir and Adleman developed 1977 the RSA cryptosystem (see [1]), which offers both, encryption and digital signature (authentication). RSA uses computation in $\mathbb{Z}_n$, where n is the product of two distinct large primes p and q and is based upon the following lemma:

Lemma 1 *Let $n = pq$ for two primes $p \neq q$. Then for all $x \in \mathbb{Z}_n$ and all positive integers k, we have*

$$x^{1+k\varphi(n)} \equiv x \mod n.$$

Here φ denotes the Euler function, i.e. the cardinality of $\{i : 0 < i < n, \gcd(n,i) = 1\}$, hence $\varphi(n) = (p-1)(q-1)$.

The RSA system for Alice and Bob consists of the following steps:

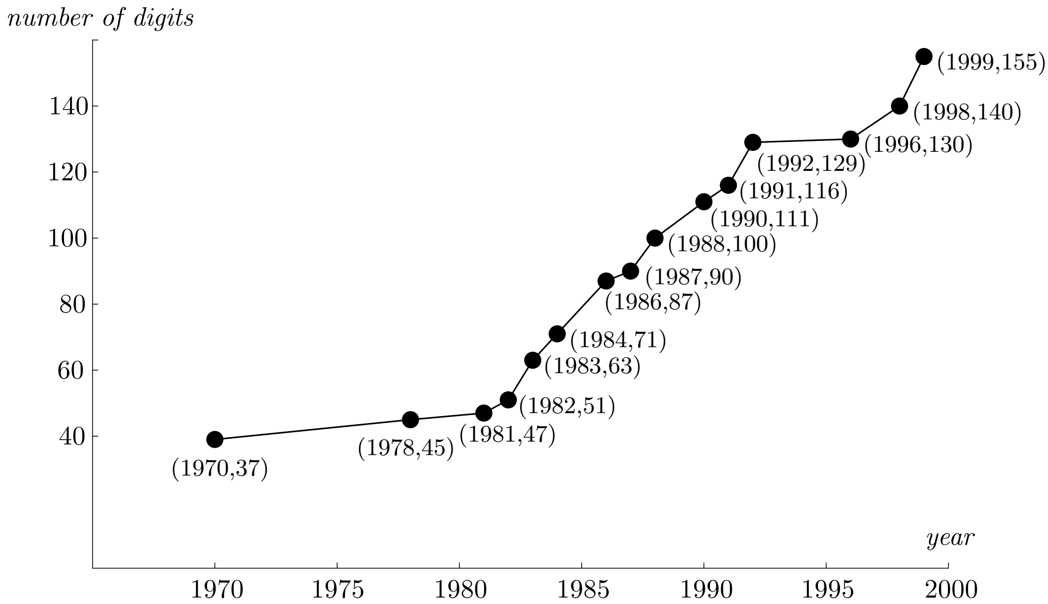
1. Bob chooses two distinct large primes p and q. He defines $n = pq$, chooses a random integer a and computes the integer b such that $ab \equiv 1 \mod \varphi(n)$, using the extended Euclid algorithm.
2. Bob publishes the values n and a in a directory. The values of p, q and b are kept secret.
3. When Alice wants to send a message $x \in \mathbb{Z}_n$ to Bob[7], she reads the values n and a and calculates $y = x^a \mod n$ and sends y to Bob.
4. To decrypt the message y, Bob uses the secret value b and calculates (using the previous lemma) $y^b \equiv x^{ab} \equiv x^{1+k\varphi(n)} \equiv x \mod n$.

For large p and q it is difficult to calculate the private key b from n and a, since it is difficult to find the factors of a large number n. It is currently possible to factor integers of about 150 digits. Most RSA implementations use numbers of 1024 bits corresponding to about 308 digits (see [22]) which seems to guarantee security at least for the next few years (see the figure below). The best general factoring algorithm today is the number field sieve (NFS) which, for a given number n, runs in time approximately $O(e^{1.9(\log n)^{1/3}(\log \log n)^{1/2}})$ (see [21] or [8]). Nevertheless, Peter Shor proposed in [24] polynomial time factorization and discrete logarithm algorithms for so called quantum computers (see Section 1.5). To the present day, no real quantum computer exists, but in the future such a machine may break cryptosystems which are based on factorization or discrete logarithms.

Actually, for no presently used cryptosystem it is *proven*, that it is difficult (i.e. not possible in polynomial time) to break it.

The following graphics summarizes the history of the records for the factorization of integers between 1970 and 1999:

[7] We can associate to a text for example its ASCII code and interpret it as a number. A longer text is segmented into blocks of adequate length.



It is possible to use the same idea as above to implement a digital signature, used, e.g., to sign legal documents electronically or in electronic voting systems: Suppose Alice wants to send a message x to Bob in such a way that Bob is assured the message is both authentic, has not been tempered with, and from Alice. Alice creates a digital signature s by exponentiation: $s = x^b \mod n$, where b is Alice's private key. She sends x and s (encrypted) to Bob. To verify the signature, Bob checks if the equality $x \equiv s^a \mod n$ holds, where n and a are Alice's public key. Therefore, if Bob can verify the last equality, then the only person who could have send x and s is Alice. In order to have shorter signatures, so called hash functions are used to compute the hash value or message digest $h(x)$. This value is then used by Alice to generate the signature $s = h(x)^b \mod n$ (see e.g. [20]).

Notice, that in the RSA system, no previous key agreement is necessary. However, cryptosystems as described in Section 1.2 have the advantage to be faster than RSA by a factor between 100 and 10'000 (depending on the implementation). Therefore a commonly used technique is to first agree on a session key (e.g. via the described Diffie-Hellman protocol) and then to use one of the fast algorithms, like DES (Data Encryption Standard), IDEA (International Data Encryption Algorithm) or the new AES Rijndael algorithm (Advanced Encryption Standard).

1.4 One-way functions

Public key cryptosystems are based on so called one-way functions. A one-way function is a map $c : A \to B$ between two finite sets A and B such that $c(x)$ is easy to compute, but for virtually all $y \in c(A)$, it is computationally infeasible to find an x such that $y = c(x)$. The prototype example is the discrete exponentiation used in the Diffie-Hellman key exchange protocol with a hard to compute discrete logarithm as inverse. The RSA system is based on a trapdoor one-way function, i.e. a one-way function for which the inverse *is* easy to compute, given a certain piece of information (the trapdoor), but difficult otherwise. In the RSA case, the trapdoor is the factorization of n.

How can such cryptographic methods be challenged by quantum computers?

1.5 Quantum computer

A quantum computer is a device that processes information stored on quantum variables such as spins, photons, or atoms. The discrete character of quantum variables makes them natural candidates for storing digital information: a spin can point "up" or "down" representing one bit of information. Unlike a bit of classical information, which can only register 0 or 1, a quantum bit ("qubit") can in some sense register 0 and 1 at once: a spin can be in a superposition of two states a (spin up) and b (spin down), where a and b are complex numbers that can vary continuously. That is, quantum information possesses both digital and analog qualities.

The original quantum computer proposed by Benioff [4] stored digital information on quantum bits, but performed computations that were essentially classical: bits were not put in superpositions. Benioff offered a classical Turing machine which used quantum mechanics in its workings, thus showing that theoretically a quantum computer was at least as powerful as a classical computer.

In 1982 Richard Feynman [13] showed that a classical Turing Machine (and hence any classical computer) could not simulate a quantum mechanical system without suffering exponential slowdown. He asked whether quantum computers might be more efficient than classical computers for providing analog simulations of other quantum systems.

David Deutsch and Richard Jozsa presented in [9] an algorithm that could be run in poly-log time on a quantum computer, but required linear time on a deterministic Turing machine. This may have been the first example of a quantum computer being shown to be exponentially faster than a deterministic Turing machine. However, the problem could also be solved in poly-log time on a probabilistic Turing machine.

Shor's algorithm for factoring exploits both digital and analog properties of quantum computers: interference between quantum bits in superpositions of different logical states plays a key role. A quantum computer is not speedier (in terms of clock speed) than its classical rival. The difference, e.g. in the factorization problem, is rather in how it goes about factoring numbers. Whereas the classical computer factors sequentially, sampling different combinations of numbers one after another in lockstep until finished, the quantum computer does it all at once in a process known as quantum parallelism.

One day, quantum computers may factor numbers of the size currently used for RSA in seconds. And we cannot just use larger numbers (e.g. generated on quantum computers) to regain security. Therefore, quantum computing may make RSA obsolete in the future.

1.6 Quantum cryptography

While classical cryptography employs various mathematical techniques to restrict eavesdroppers from learning the contents of encrypted messages, in quantum cryptography the information is protected by the laws of physics: Quantum cryptography provides means for two parties to exchange a key over a channel with any desired security of communication.

There are at least three main types of quantum cryptosystems for the key exchange, these are:

(A) Cryptosystems with encoding based on two non-commuting observables, proposed by S. Wiesner [30], and by C. H. Bennett and G. Brassard [5].

(B) Cryptosystems with encoding built upon quantum entanglement and the Bell Theorem, proposed by A. K. Ekert [12].

(C) Cryptosystems with encoding based on two non-orthogonal state vectors, proposed by C. H. Bennett [6].

Quantum cryptosystem (A) can be explained with the following simple example. The system includes a transmitter and a receiver. A sender may use the transmitter to send photons in one of four polarizations: 0, 45, 90, or 135 degrees. A recipient at the other end uses the receiver to measure the polarization. According to the laws of quantum mechanics, the receiver can distinguish between rectilinear polarizations (0 and 90), or it can quickly be reconfigured to discriminate between diagonal polarizations (45 and 135); it cannot, however, distinguish both types. The key distribution requires the following steps. The sender sends photons with one of the four polarizations which are chosen at random. For each incoming photon, the receiver chooses at random the type of measurement: either the rectilinear type or the diagonal type. The receiver records the results of the measurements but keeps them secret. Subsequently the receiver publicly announces the type of measurement (but not the results) and the sender tells the receiver which measurements were of the correct type. The two parties (the sender and the receiver) keep all cases in which the receiver measurements were of the correct type. These cases are then translated into bits (1's and 0's) and thereby become the key. An eavesdropper is bound to introduce errors to this transmission because he does not know in advance the type of polarization of each photon and quantum mechanics does not allow him to acquire sharp values of two non-commuting observables (here rectilinear and diagonal polarizations). The two legitimate users of the quantum channel test for eavesdropping by revealing a random subset of the key bits and checking (in public) the error rate. Whenever they are not happy with the security of the channel they can try to set up the key distribution again.

Recently quantum cryptology has been tested to work successfully for a transmission over 10 km of fiber optic cable.

2 A cryptographic system based on the second principle of thermodynamics

In this section we propose the heat flow as a cryptographic system that may resist attacks even by quantum computers.

2.1 A one-way function

Consider the heat equation

$$u_t - \Delta u = 0 \qquad \text{on } \mathbb{R}_+ \times \mathbb{R}^n \tag{1}$$

with initial data

$$u(0, x) = a(x) \qquad \text{for } x \in \mathbb{R}^n \tag{2}$$

at time $t = 0$. Mathematically speaking, the evolution problem for (1) in the forward time direction is a well-posed initial-value problem, whose solution can be modelled with extreme accuracy also numerically. In contrast, the evolution problem in backward time is ill-posed, and numerical algorithms for solving equation (1), (2) for $t < 0$ are inherently unstable. Physically, this observed impossibilty of reversing the arrow of time is a consequence of the second principle of thermodynamics. Therefore the forward heat flow seems a promising candidate for a one-way function.

Denoting as Γ the fundamental solution to the heat equation, given by

$$\Gamma(t,x) = \frac{1}{(4\pi t)^{n/2}} \exp\Big(-\frac{|x|^2}{4t}\Big)$$

for $t > 0$, the unique solution of (1), (2) is given by

$$u(x,t) = \int_{\mathrm{I\!R}^n} \Gamma(t, y-x) a(y) dy.$$

In particular, at time $t = 1$, letting

$$\gamma(x) = \Gamma(1,x)$$

we have

$$\alpha(x) := u(1,x) = \int_{\mathrm{I\!R}^n} \gamma(x-y) a(y) dy$$

or

$$\alpha = \gamma * a$$

for short, where

$$(f * g)(x) = \int_{\mathrm{I\!R}^n} f(x-y) g(y) dy$$

denotes the convolution of any two suitable functions f and g on $\mathrm{I\!R}^n$. Note that $f * g = g * f$. Moreover, for $f \in L^p(\mathrm{I\!R}^n)$, $g \in L^q(\mathrm{I\!R}^n)$ with $\frac{1}{p} + \frac{1}{q} \geqslant 1$ we have $f * g \in L^r(\mathrm{I\!R}^n)$ if $\frac{1}{p} + \frac{1}{q} = 1 + \frac{1}{r}$. In this case, the inequality

$$\|f * g\|_{L^r} \leqslant \|f\|_{L^p} \|g\|_{L^q}$$

holds. Since $\gamma \in L^1(\mathrm{I\!R}^n)$ with $\|\gamma\|_{L^1} = 1$, for $a \in L^p(\mathrm{I\!R}^n)$, letting $\alpha = \gamma * a$ as above, we have $\alpha \in L^p(\mathrm{I\!R}^n)$ with

$$\|\alpha\|_{L^p} \leqslant \|a\|_{L^p}, \qquad 1 \leqslant p \leqslant \infty.$$

In the following we fix $p = 1$ for convenience, because in this case $*$ generates an associative and symmetric operator on $L^1(\mathrm{I\!R}^n)$. As is well known, for any Lebesgue-integrable function a the function $\alpha = \gamma * a$ is well-defined and real analytic. Moreover, for $a \neq b \in L^1(\mathrm{I\!R}^n)$ with corresponding $\alpha = \gamma * a$ and $\beta = \gamma * b$, it follows that $\alpha - \beta = \gamma * (a - b) \neq 0$. Otherwise, by the Cauchy-Kowalewski theorem, $\Gamma(t,\cdot) * a = \Gamma(t,\cdot) * b$ for all $t > 0$. Letting $t \searrow 0$, we then obtain

$$a = \lim_{t \searrow 0} \Gamma(t,\cdot) * a = b$$

contrary to the assumption. Thus, the map

$$\begin{array}{rcl} \mathscr{A} : L^1(\mathrm{I\!R}^n) & \to & L^1(\mathrm{I\!R}^n) \\ a & \mapsto & \alpha = \gamma * a \end{array}$$

is well-defined, continuous and injective, but not onto. This is, basically, our one-way map. Inverting $\mathscr{A}$ on its range $\mathrm{im}(\mathscr{A})$ is "difficult" as a consequence of the second principle of thermodynamics. In Section 3 we illustrate this fact in the case of periodic functions defined on a lattice, corresponding to discretizing (1), (2) on a spatially periodic domain.

2.2 Applications

In cryptography the above function $\mathscr{A}$ might be applied in various ways. We illustrate here two possible applications: Public key cryptography (where we indicate an analogue to the Diffie-Hellman key agreement protocol in Section 2.2.1) and password verification (in Section 2.2.2.A and B).

2.2.1 Key exchange over an insecure channel

Alice and Bob choose secret passwords a and b respectively. Their corresponding public keys will be $\alpha = \gamma * a$ and $\beta = \gamma * b$. If Alice wants to send a message M to Bob, she encrypts M using as key

$$\kappa = \beta * a$$

with a standard symmetric key method and sends the ciphertext $C = e_\kappa(M)$ to Bob. She tells Bob openly that she is the sender of this message. Bob, upon receiving the message supposedly from Alice, looks up Alice' public-key α in a trusted directory and generates

$$\kappa' = \alpha * b$$

using his private key b. By symmetry and associativity of the convolution product we have

$$\kappa' = \alpha * b = (\gamma * a) * b = \gamma * (a * b) = \gamma * (b * a) = (\gamma * b) * a = \beta * a = \kappa$$

and therefore Bob can easily decrypt $M = d_\kappa(C)$. Moreover, at the same time the message is authenticated. Indeed, Eve, imposing as Alice, in order to generate

$$\kappa = \gamma * (a * b) = \alpha * b = \beta * a$$

from knowledge of the public data α, β and γ would have to be in possession of either a, b or $a * b$.

2.2.2 Password verification

(A) Simple password verification. In the simplest case, a new user, say Alice, of a computer system chooses a secret password p and enters it into the system. The system stores only the encrypted password $\pi = \gamma * p$ in a (world readable) file. Each time a password p', supposedly from Alice, has to be verified, the value $\pi' = \gamma * p'$ is compared to π. If the values agree, the authentication is granted.

(B) One-time passwords. If a user, again Alice, has to be authenticated over an insecure channel, an adversary may put himself between Alice and the computer system and intercept the password Alice sends. Even if the password was encrypted, the adversary may send it later on himself, claiming he is Alice. The following protocol avoids this man-in-the-middle attack: Alice chooses, as above, a secret password p. In contrast to Section (A),

$$\pi_N = \gamma *^N p = \underbrace{\gamma * \gamma * \ldots * \gamma}_{N \text{ times}} * p = \Gamma(N, \cdot) * p$$

is now stored by the computer system in a (world readable) file. Here, $N > 1$ is a fixed natural number. If Alice wants to login remotely for the n-th time, the computer system tells her to use $\pi_{N-n} = \gamma *^{N-n} p$ as one-time password. Alice computes the value of π_{N-n} and sends it to the computer who compares $\gamma *^n \pi_{N-n}$ with π_N. Again, if the values agree, the authentication is successful. Since every password π_{N-n} is valid for only one login, it is of no use for a potential adversary, since it is “impossible” to deduce π_i from π_j for $i < j$. Of course, after the last number $n = N$ is consumed, during the last session, Alice has to agree with the computer system on a new password $\pi'_N = \gamma *^N p'$.

3 A toy model

To have a simple model on which we can test our theoretical predictions, let us choose $n = 1$ and, for further convenience, let us restrict our attention to initial data that are periodic on the real line with period 2π. Moreover, for our purposes it is advantageous to let (1) act on the set of M-band limited functions, i.e. functions f with the property that $\hat{f}(\lambda) = 0$ whenever $|\lambda| \geqslant M$: This space V_M of Fourier-polynomials of order $M-1$ then is a finite dimensional real vector space of dimension $2M-1$, and $\mathscr{A}$ acts bijectively on this space. Moreover, M-band limited functions are real analytic and they are subject to Shannon's sampling theorem. The trapezoidal rule then integrates such functions exactly (i.e. without discretization error) provided the step size is chosen sufficiently small. For the reader's convenience, we include the following classical result on the trapezoidal rule:

Lemma 2 *For $f \in V_M$ and a natural number $m \geqslant M$ we have*

$$\int_0^{2\pi} f(x)dx = \tfrac{2\pi}{m} \sum_{l=1}^{m} f(\tfrac{2\pi l}{m}).$$

By trigonometric interpolation, for $N = 2M-1$, the mapping

$$\varphi : V_M \to \mathrm{I\!R}^N, \quad a \mapsto \varphi(a) = (a_k)_{k=1,\dots,N}, \quad a_k = a(\tfrac{2\pi k}{N})$$

is an isomorphism of vector spaces. Two immediate consequences of Lemma 2 are:

Corollary 3 *If $a \in V_M$, $\varphi(a) = (a_k)_{k=1,\dots,N}$, $N = 2M-1$, the Fourier-transform is given by*

$$\hat{a}_k = \frac{1}{2\pi} \int_0^{2\pi} e^{-ikx} a(x)dx = \tfrac{1}{N} \sum_{l=1}^{N} e^{-\frac{2\pi i k l}{N}} a_l$$

for $k = -M+1, \dots, M-1$ (and of course $\hat{a}_k = 0$ for $|k| \geqslant M$).

Corollary 4 *If $a, b \in V_M$, $a_k = a(\frac{2\pi k}{N})$, $b_k = b(\frac{2\pi k}{N})$, $N = 2M-1$, the convolution $c(\cdot) = (a * b)(\cdot) = \int_0^{2\pi} a(\cdot - y)b(y)dy$ belongs to V_b, and $\varphi(c) = (c_k)_{k=1,\dots,N}$ is given by*

$$c_k = \tfrac{2\pi}{N} \sum_{l=1}^{N} a_{k-l} b_l.$$

For 2π-periodic functions, the heat equation (1) transforms into the ordinary differential equation for $\hat{a}_k = \hat{a}_k(t)$

$$\frac{d}{dt} \hat{a}_k + |k|^2 \hat{a}_k = 0, \quad k \in \mathbb{Z}.$$

Hence, we conclude that $\hat{a}_k$ is given by

$$\hat{a}_k(t) = e^{-k^2 t} \hat{a}_k(0), \quad k \in \mathbb{Z}.$$

In particular, at time $t = 1$, we obtain

$$\hat{a}_k(1) = \hat{\alpha}_k = e^{-k^2} \hat{a}_k(0).$$

Thus, for initial values $a \in V_M$, we conclude that the solution α at time $t = 1$ also belongs to V_M and that, for $x_m = \frac{2\pi m}{N}$,

$$\begin{aligned}
\alpha_m &= \alpha(x_m) \\
&= \sum_{k=-M+1}^{M-1} \hat{a}_k(1) e^{ikx_m} \\
&= \sum_{k=-M+1}^{M-1} \hat{a}_k(0) e^{ikx_m - k^2} \\
&\overset{\text{Cor. 3}}{=} \tfrac{1}{N} \sum_{k=-M+1}^{M-1} \sum_{l=1}^{N} e^{\frac{2\pi i k(m-l)}{N} - k^2} a_l.
\end{aligned}$$

In other words, the linear map $A : (a_k)_{k=1,\ldots,N} \mapsto (\alpha_k)_{k=1,\ldots,N}$, is represented by the symmetric matrix $(A_{lm})_{1 \leqslant l,m \leqslant N}$ with

$$A_{ml} = \gamma(m-l) = \tfrac{1}{N} \sum_{k=-M+1}^{M-1} e^{\frac{2\pi i k(m-l)}{N} - k^2} = \tfrac{1}{N}\Big(1 + 2\sum_{k=1}^{M-1} e^{-k^2} \cos(\tfrac{2\pi k(m-l)}{N})\Big). \tag{3}$$

Conversely, the inverse map A^{-1} is given by

$$\begin{aligned}
a_m &= a(x_m) \\
&= \sum_{k=-M+1}^{M-1} \hat{a}_k(0) e^{ikx_m} \\
&= \sum_{k=-M+1}^{M-1} \hat{a}_k(1) e^{ikx_m + k^2} \\
&\overset{\text{Cor. 3}}{=} \tfrac{1}{N} \sum_{k=-M+1}^{M-1} \sum_{l=1}^{N} e^{\frac{2\pi i k(m-l)}{N} + k^2} a_l
\end{aligned}$$

with exponentially diverging coefficients

$$A^{ml} = \tfrac{1}{N} \sum_{k=-M+1}^{M-1} e^{\frac{2\pi i k(m-l)}{N} + k^2} = \tfrac{1}{N}\Big(1 + 2\sum_{k=1}^{M-1} e^{k^2} \cos(\tfrac{2\pi k(m-l)}{N})\Big). \tag{4}$$

We may summarize the preceding calculation as follows:

Proposition 5 *The solution $\alpha = \mathscr{A}(a)$ at time $t = 1$ of the heat flow problem (1) with initial data $a \in V_M$ is given by*

$$\alpha = \varphi^{-1}(A\varphi(a)) \in V_M,$$

where A is given by (3).

By the isomorphism φ, we may identify V_M with $\mathbb{R}^N$, $N = 2M - 1$. This also suggests to use the linear map

$$\begin{aligned}
A : \mathbb{R}^N &\to \mathbb{R}^N \\
(a_k)_{k=1,\ldots,N} &\mapsto (\alpha_k)_{k=1,\ldots,N}, \quad \alpha_m = \sum_{l=1}^{N} A_{ml} a_l
\end{aligned}$$

for a concrete implementation of the proposed cryptosystem and to take $K_N = \{0,1\}^N \subset \mathbb{R}^N$ as our secret key space for sufficiently large $N = 2M - 1$.

If we attempt to implement this scheme numerically, however, we are faced with two difficulties. First, it is not clear that for any $k \in K_N$ the image Ak can be uniquely rounded to machine-size precision. Moreover, once the first problem is overcome we can certainly use the explicit formula (4) to compute k from Ak, using double precision, if necessary.

The solution to both these problems in the toy model that we have in mind appears to be quite simple. In fact, the eigenvalues of A are $\{1, e^{-1}, e^{-1}, e^{-4}, e^{-4}, \ldots, e^{-(M+1)^2}, e^{-(M+1)^2}\}$. Therefore, the condition number of A is

$$\mathrm{cond}(A) = \frac{\max\{|\lambda| \,:\, \lambda \in \sigma(A)\}}{\min\{|\lambda| \,:\, \lambda \in \sigma(A)\}} = e^{(M-1)^2}.$$

This indicates that if we substitute the matrix A, for instance, by the matrix $A' = \mathrm{round}_\delta\, A$, where $\mathrm{round}_\delta(x)$ is the closest machine size real number to x of a chosen precision $\delta > 0$, here acting on the coefficients of A, then it will be numerically infeasible to compute the inverse of the matrix A'. Benchmark tests with Mathematica indicate that the computational effort to solve the linear system $A'a = \alpha$ for $\delta \sim e^{-(M-1)^2}$ grows exponentially with M.

Observe that the rounding operation preserves the "convolution structure" $A'_{ml} = \gamma'(m-l)$. Moreover, for any $k \in K_N$ the "encrypted" key $A'k$ will be of machine size. The methods described above for password verification and key exchange over an insecure channel can therefore be carried over without any change to the discrete model.

4 Implementation on a quantum computer

Since Shor's proposal, a considerable amount of effort has been expended to find other problems that quantum computers might solve more easily than classical computers. As proposed by Feynman, quantum computers might be used to simulate certain continuous problems, like the heat equation, and therefore to solve these problems without discretization error. Such a technique might be used to implement the thermodynamic one-way function introduced in Section 2 directly, without prior discretization. Then, quantum teleportation might be the way to transmit the public key $\alpha = \gamma * a$. This method would thus offer an alternative to the current quantum cryptography.

Acknowledgment: We thank Michael Rabin, whose Nachdiplom lecture in the summer term 2000 at ETH sparked our interest in the subject. And we thank Carlo Matteotti, Ueli Maurer, Daniel Neuenschwander and François Weissbaum for their expertise, inspiring discussions, and for sharing ideas.

References

[1] L. M. Adleman, R. L. Rivest, A. Shamir: A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM **21** (1978), 120–126.

[2] Fredrik Almgren, Gunnar Andersson, Torbjörn Granlund, Lars Ivansson, Staffan Ulfberg: http://answers.codebook.org

[3] Kendall E. Atkinson: *An introduction to numerical analysis.* Wiley and Sons, New York, 1989

[4] Paul A. Benioff. The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines. J. Stat. Phys. **22** (1980), 563–591.

[5] Charles H. Bennett, G. Brassard: Quantum Cryptography: Public Key Distribution and Coin Tossing. Proceedings of IEEE International Conference on Computers Systems and Signal Processing, Bangalore India, December 1984, 175–179.

[6] Charles H. Bennett: Quantum cryptography using any two nonorthogonal states. Phys. Rev. Lett. **68** (1992), 3121–3124.

[7] Albrecht Beutelspacher, Jörg Schwenk, Klaus-Dieter Wolfenstetter: *Moderne Verfahren der Kryptographie. Von RSA zu Zero-Knowledge.* Vieweg, 2001

[8] David M. Bressoud: *Factorization and primality testing.* Springer, 1989

[9] David Deutsch, Richard Jozsa: Rapid solution of problems by quantum computation. Proc. R. Soc. Lond. A **439** (1992), 553–558.

[10] Whitfield Diffie: The first ten years of public-key cryptography. Proceedings of the IEEE **76** (1988), 560–577.

[11] Whitfield Diffie and Martin E. Hellman: New directions in cryptography. IEEE Transactions on Information Theory **22** (1976), 644–654.

[12] Arthur K. Ekert: Quantum cryptography based on Bell's theorem. Phys. Rev. Lett. **67** (1991), 661–663.

[13] Richard P. Feynman. Simulating physics with computers. Int. J. Theor. Phys. **21** (1982), 467–488

[14] William F. Friedman: *Elementary military cryptography.* Aegean Park Press, 1976

[15] William F. Friedman: *Advanced military cryptography.* Aegean Park Press, 1976

[16] Andrew Hodges, Douglas Hofstatder: *Alan Turing: The Enigma.* Walter & Company, 2000

[17] Neal Koblitz: *A course in number theory and cryptography.* Springer, 1987

[18] Neal Koblitz: *Algebraic aspects of cryptography.* Springer, 1998

[19] James L. Massey: *Cryptography: Fundamentals and applications.* Copies of transparencies of a course given in Engelberg. Advanced technology seminars, 1995

[20] Alfred J. Menezes, Paul C. van Oorschot, Scott A. Vanstone: *Handbook of applied cryptography.* CRC Press, Boca Raton etc., 1997

[21] Hans Riesel: *Prime numbers and computer methods for factorization.* Birkhäuser, Boston, 1985

[22] RSA Laboratories: *Frequently asked questions about today's cryptography.* http://www.rsa.com

[23] Bruce Schneider: *Applied cryptography: Protocols, Algorithms, and source code in C.* John Wiley & Sons, 1994

[24] Peter W. Shor: Algorithms for quantum computation: Discrete logarithms and factoring. Proc. 35 Annual Symposium of Foundations of Computer Science, IEEE Computer Society Press (1994), 124–135.

[25] Simon Singh: *The Code Book: The Science of Secrecy from Ancient Egypt to Quantum Cryptography.* Anchor Books, 2000

[26] Michael Smith: *Station X - the codebreakers of Bletchley Park.* Channel 4 books, 2000

[27] Douglas R. Stinson: *Cryptography: Theory and practice.* CRC Press, Boca Raton etc., 1995

[28] Henk C. A. van Tilborg: *An introduction to cryptology.* Kluwer Academic Publishers, 1988

[29] D. Welsh: *Codes and cryptography.* Oxford University Press, 1988

[30] S. Wiesner: Conjugate coding. SIGACT News 15, 78 (1983); original manuscript written circa 1969.