# Cryptology – Methods, Applications and Challenges

Claus Diem

**Abstract.** Information processing by electronic devices leads to a multitude of security-relevant challenges. With the help of cryptography, many of these challenges can be solved and new applications can be made possible.

What methods are hereby used? On which mathematical foundations do they rest? How did the prevailing ideas and methods come about? What are the current developments, what challenges exist and which future challenges can be predicted?

**2010 Mathematics Subject Classification.** Primary 01-A65; Secondary 01-A67, 94-03, 11-03

**Keywords.** Cryptology, cryptography

## 1. Encrypted messages and more

Secret agents, online stores and pupils exchanging "secret messages" consisting of nonsensical symbols all use it: cryptography. The word "cryptography" is derived from the Greek words "κρυπτος", 'hidden', and "γραφειν", 'to write'; it is therefore about secret writing.

The sender *encrypts* the message and the receiver *decrypts* it with an agreed-upon secret. The endeavor to read encrypted messages without knowledge of the secret is called *cryptanalysis*. It is common to summarize both aspects under *cryptology*.

Due to the technical development in the field of electronics, the notions of cryptography and cryptology are nowadays used more broadly; the goals of cryptography now cover all aspects of security in processing, transmission and use of information in the presence of an adversary.

In this way, cryptographic methods have entered many different areas. One can use them to ensure *confidentiality* in any kind of electronic communication. They are used for *authentication* when unlocking a car or releasing an immobilizer, withdrawing money with a bank card or identifying oneself at a border with a passport, for example. Documents are nowadays often signed digitally with cryptographic methods, for example by a notary; like this the *non-repudiation* of agreements can be guaranteed. With digital signatures one can also guarantee the *integrity* of electronic data, that is, that the data has not been tampered with; this is for example used in passports.

What is the current state of cryptography in a world of electronic devices in which data acquisition and processing are continuously increasing? What are its foundations, what are its applications? How did the prevailing ideas and methods develop historically and what current developments and challenges are there?

## 2. Classic ideas

Until the end of the First World War, cryptology developed slowly and the schemes used were, mathematically speaking, elementary from today's point of view. Nonetheless, many notions and ideas of the past are still fundamental.

**2.1. Approaches.** To start with, some definitions: In using a cryptographic scheme, texts are encrypted, sent, received and then decrypted. The primary text is called *plain text* and the encrypted text is called *cipher text*; instead of *en-/decrypting* one can also speak of *en-/deciphering*.

Classically, there are two fundamentally different approaches to encrypting: First, one can apply cryptographic methods at the atomic level, that is, at the level of letters; second, one can start at the level of words.

The second approach is easier to describe because the method allows for less variation: With the help of a special *code book*, words of the standard language are replaced by *code words*. These code words can be other words of the standard language or arbitrary combinations of letters and numeric symbols.

For the first approach an enormous amount of schemes have been developed over the centuries. Two obvious ideas have thereby occurred over and over: *substitution* and *transposition*.

With the method of substitution, individual symbols (letters, the space character, numeric symbols, punctuation marks) are replaced by other symbols (or combinations of symbols). These symbols can be made up or they can be normal letters and numeric symbols. Even if may be appealing to use "mysterious" symbols, this does not make any difference from the point of view of security.

With the method of transposition, the order of the symbols in the text is changed according to a certain rule. A good example is: Write the text line by line in a table from the upper left corner to the lower right corner, then read off the columns of the text in a prearranged order.

Of course, one can combine these methods with each other. One can, for example, first use a code book and then a transposition and finally a substitution.

Classic cryptographic methods rely on common secrets between the sender and the receiver. Generally, the methods offer the possibility to encrypt texts with variable secrets. With the code book-method, it is the code book itself, with the substitution method it is the substitution table and with the transposition method as described above it is the number and order of the columns.

This secret is called the *key*. One can thus – at least in the examples discussed above – distinguish between the cryptographic scheme itself and the key. We shall see that this distinction is of particular relevance.

**2.2. The race of cryptology.** As nobody wants to use an insecure cryptographic scheme, potential schemes are tested for possible attacks in advance and schemes in use are continuously reviewed.

This establishes a race between designing and attacking schemes. Here we present the progression of this race exemplarily with the substitution method.

The following presentation is idealized rather than historical, even though corresponding thoughts and developments did indeed occur in the course of centuries.

With the substitution method as described, one immediately notices that symbols like `E` or the space character occur in natural languages more often than others. In general, this also holds for a particular plain text. It is therefore reasonable to assume that, for example, the most frequent symbol in a cipher text corresponds to `E` or to the space character. A text of a few lines in a known language can, generally speaking, be recovered by considering the frequencies of the symbols. This encryption scheme can thus be considered broken. One might then ask: Is there a variant of the scheme for which the described attack is not feasible anymore?

Yes, there is one: One can ensure that in the cipher text all symbols appear with about the same frequency. This method is based on the idea that one symbol might be encrypted to various distinct symbols. Suppose that we use 1000 distinct symbols for writing cipher texts. If now the letter `E` appears with a probability of 12,7 % in a given language, 127 distinct symbols are assigned to it. When encrypting, for each occurrence of `E` one of the 127 symbols is chosen at random and used in its place. The other symbols of the plain text are encrypted in the same manner. In this way, the frequency analysis as described above fails. But how can one up with 1000 distinct symbols? Well, this is easier than one might first think: One starts with the ten numerical symbols from `0` to `9` and regards each string of three of these symbols as a symbol in itself. In concrete terms, the letter `E` is therefore represented by 127 chosen combinations of strings of three numerical symbols.

After the encryption scheme has been modified, it is natural to ask if the attack method can be modified as well. Yes, it can: One no longer considers the frequencies of symbols in a language and in the cipher text (in the concrete example, in the cipher text, a string of three numerical symbols is considered as one symbol) but of so-called *bigrams*, which are combinations of two symbols or of *trigrams*, that is, combinations of three symbols.

Again the scheme is broken, which raises the question if this encryption scheme can again be improved or if one should maybe use a completely different method.

A natural answer to that question is that the concept of a substitution is so fundamental that it should be part of the method in any case. One might for example combine a substitution with a transposition.

So far, the attacks considered have not taken into account information about the sent messages themselves, but such information might also be used. For example, letters usually begin and end with a common salutation phrase and military messages are often highly standardized. If this leads nowhere, an attacker might also try to have the user of a cryptographic scheme encrypt a message foisted on him, such that the attacker has for a given plain text the corresponding cipher text. It is conceivable that the attacker can extract a sought-after information from a secret message with the help of such plain text – cipher text pairs even without finding the key.

## 3. New ideas

According to the technical development the history of cryptology can be divided into three periods:

1. The *paper-and-pencil era* until about the end of World War I.
2. The *era of electric-mechanic cipher machines* from about the end of World War I until about 1970.
3. The *electronic era* from about 1970.

As the name indicates, the first period was characterized by the fact that at most simple mechanical devices came to use for secret writing. For breaking schemes, beside ad hoc approaches mainly statistical methods as described in Section 2.2 were applied.

In the second period, for encryption next to schemes for writing by hand, electric-mechanic machines like the German Enigma were used. The increase of sophistication in the electronic-mechanic encryption machines was countered by cryptoanalytic methods which exceeded purely statistical techniques. Cryptanalysis was a driving force in the development and construction of the first electronic computing machines.

The electronic era started with the advent of data processing. As the methods developed in the beginning of the era are still being used or the current methods are direct successors of these methods, it can be seen as the present age of cryptology. This period is characterized not only by the used technology but also by its strive for scientific methods in cryptography, a strong connection to mathematics and a high innovation speed.

In this section we mainly want to retrace the development in this near past, focussing on conceptual ideas.

**3.1. Four seminal texts.** We reduce the first two periods to four texts in which methods still of relevance today are developed. This condensed presentation does surely not do justice to the history of cryptology before the electronic era. A reader whose interest is aroused is invited to read the definitive book on the history of cryptology, David Kahn's *The Codebreakers*, first published in 1967 with a new edition from 1996 ([18]). For presentations of newer results on the history of cryptology we recommend the books *Codeknacker und Codemacher* by Klaus Schmeh ([30]) and *CryptoSchool* by Joachim von zur Gathen ([10]).

**3.1.1. رسالة أبي يوسف يعقوب بن إسحاق الكنديّ في استخراج المعمّى إلى أبي العبّاس (Abū Yūsuf Yaʿqūb ibn Isḥāq al-Kindī: The missive on cryptanalysis to Abū l-ʿAbbās).** By current knowledge, the first systematic presentation of cryptology originates from the Islamic middle ages. The author is the Aristotelian philosopher al-Kindī, who resided in Baghdad in the 9$^{\text{th}}$ century C.E.

This text was long considered lost, as were two other Arabic texts on cryptology from the 13$^{\text{th}}$ and 14$^{\text{th}}$ century. In the 1980s, for each of the three texts, one manuscript was found and subsequently edited and published ([24]).

In the manuscript presumably containing al-Kindī's treatise, different schemes based on substitution and transposition of letters are discussed; the schemes are even categorized in a tree diagram. The cryptanalysis is developed on the basis of frequency analysis and even bi- and digrams are considered.

By the way: Words of European languages like "cipher" (or "cypher"), "chiffre", "cifre", "cifra" or "Ziffer" come from the Arabic word for zero, "صفر" ("ṣifr"). Over the course of time, these and similar words have meant zero, numerical symbol, scheme for en-/deciphering and cipher text.

**3.1.2. Leon Battista Alberti: De Componendis Cyfris (On the writing in ciphers).** The first passed down European text on cryptology was written in about 1466. Its author Leon Battista Alberti is regarded as the embodiment of a "universal Renaissance man"; he worked as an architect and wrote remarkably many works about the most diverse subjects and also literary works.

Alberti advocates changing the substitution table during encryption. For this, he invented – how it seems – the *cipher disks*, which were popular devices for cryptography until the 19$^{\text{th}}$ century. These devices consist of two disks which are clinched in the middle, whereby the lower disk is larger than the upper one. On the boundaries of each of these disks an alphabet is written. For every position of the disks with respect to each other one therefore obtains a particular substitution.

For the scheme envisioned by Alberti, the order of one of the alphabets is permuted and the position is changed after some words. In addition, Alberti proposed to use a code book for the most important words.

This concrete scheme does not seem to have been much in use, however. Rather, later a simpler but much less secure scheme became popular among laypersons of cryptology under the name *Vigenère scheme*. For this scheme, both alphabets are in identical (common) order and the position is changed after ever letter. The common secret is now a keyword (codeword). If this is for example `DISK`, then the `A` is first turned to the `D`, then to the `I`, then to the `S` and finally to the `K`, after which one starts anew. Professional cryptographers knew however that this scheme was rather weak and used more elaborate schemes involving code books. (For more information see Chapter 4 of [18].)

*The oldest known preserved cipher disk, a French disk from the time of Louis XIV Source: Nicholas Gessler Collection*

**3.1.3. Auguste Kerckhoffs: La Cryptographie Militaire.** This work ([19]) from 1883 is arguably the one from the paper-and-pencil period which is most ref-

fered to today. Kerckhoffs, a Dutch linguist and cryptographer residing in France, enunciates six principles for military cryptography, which were later called *Kerckhoffs' principles*. The first three are the most important ones. In slightly pointed and revised form, these are:

1. The scheme must be de facto, if not mathematically, unbreakable.
2. The usage of the encryption device must not require a secret, and it must be possible that such a device falls into the wrong hands without disadvantage.
3. The key must be transferable without written notes and kept in mind and exchanged at the will of the correspondents.

With respect to the first demand Kerckhoffs states: It is generally assumed that it is sufficient during war if a cipher system offers security for three or four hours. However, there is very well information which is important for more than a couple of hours. "Without enumerating all thinkable possibilities", Kerckhoffs mentions communication from a sieged city to the outside. That a good cryptographic scheme should offer security independently of all eventualities is an idea which has remained ever since.

The argumentation concerning the second demand is: Cryptography always demands secrets. Armies are now so large that one has to assume that the enemy knows all secrets which are known to a large number of soldiers. It is therefore imperative that only very few people must know of the secret. This means in particular that no extensive, hard to keep secret code books should be used. This demand is reinforced by the third principle.

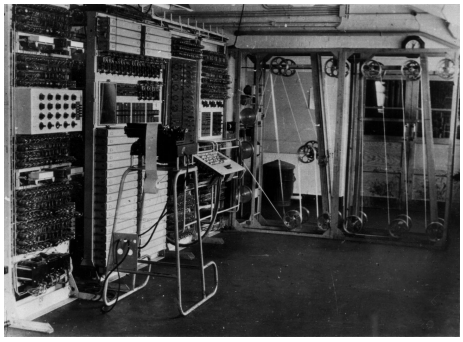
These principles and their justification were seminal for the development of cryptography, particularly concerning the establishment of scientific methods. The first two principles are still considered to be fundamental for the design of cryptographic devices. Concerning the third principle, one still demands that the key can be exchanged easily but not that it can be kept in mind (cf. Section 3.2.7).

**3.1.4. Claude Shannon: Communication Theory of Secrecy Systems.** Published in 1948 by the US-American Claude Shannon, this work ([31]) was of similar importance for the development of cryptology as the one by Kerckhoffs. Developing an abstract theory of en- and decryption, Shannon describes cryptology in mathematical terms like no one before him. For example, he describes encryption in a cipher system as a function in two variables, one for the key and one for the message (plain text). The claims are then formulated as mathematical theorems and proven accordingly.

Shannon asks how much information a cipher text gives about a plain text, given that the attacker already has some information. He uses stochastic and statistical methods as well as the so-called *information theory* developed by himself and introduces the notion of *perfect security* which captures that an attacker gains no information. Thereafter he proves that perfect security requires that the key written as a string of symbols is at least as long as the message to be transmitted securely. This illustrates that perfect security as defined by Shannon is incompatible with Kerckhoffs' principles.

Shannon also introduces two relevant principles for the construction of cipher schemes: *diffusion* and *confusion*. In both cases, it is the goal to impede the application of statistical methods for cryptanalysis. Diffusion means that small changes of the plain text affect large parts of the cipher text. Confusion as defined by Shannon means that the relationship between cipher text and key is difficult. This requirement was later extended by the demand that the relationship between cipher text and plain text shall also be complex.

**3.2. The electronic era.** During Word War II, a group of British scientists constructed one of the first vacuum-tube computers, called *Colossus*, to decipher the most high-level German military communication; their struggle was successful and one of the biggest successes of cryptanalysis ever.



*The Colossus computer*
*Source: British National Archive*

For encryption and other aspects of cryptography computers were not relevant before the advent of electronic data processing at the end of the 1960s. The new technology soon went along with further changes: Handling and research with cryptography became more open, the notion of cryptography got a broader scope and in a completely new way mathematical concepts were employed. This development continues to the present day.

Computers store data as strings of bits; these bits are now the atoms that the letters were in the paper-and pencil period. It is natural to apply encryption at this level – what all schemes discussed in the following do. We recall that one can express natural numbers (which by definition shall also include the number 0) in the binary system, that is, by strings of bits. For example, the number 10 is represented by the string `1010`. The number of bits necessary to represent a number in this way is called the *bit-length* of the number; for a number $a \neq 0$ it is about $\log_2(a)$. Moreover, by padding such strings with strings of 0's in front, one can identify the bit-strings of a length exactly $\ell$ with the natural numbers smaller than $2^\ell$.

**3.2.1. From the Data Encryption Standard to the Advanced Encryption Standard.** In the year 1973 the predecessor of the US-American National Institute of Standards and Technology (NIST) made a public offer for a to-be standardized cipher algorithm (an *algorithm* being a method of computation). After no submitted algorithm was considered adequate, in 1977 a variant of an algorithm submitted by IBM was chosen as the *Data Encryption Standard (DES)*. Like all standards issued by NIST, also this standard has applied officially only to the US-government and its contractors. Nonetheless, it developed fast into a de facto

industry standard – as was to be expected.

In the algorithm, blocks consisting of 64 bits each are encrypted. The algorithm consists of 16 rounds in which the same method is applied over and over again. At the beginning of each round, a part of the key and a part of the current intermediate result are bitwise merged via the *exclusive or (XOR)* operation. Thereafter the intermediate result is partitioned into small blocks of 6 bits each. With the help of fixed but "randomly looking" tables, each of the 6 bit long strings is changed to another sting. Finally, the blocks are transposed among each other.

One can say that the algorithm makes repeated uses of Shannon's ideas: Via the tables, one obtains confusion and via the transposition one obtains diffusion.

Up until now, no practically relevant attack on DES has been found that is faster than pure trial-and-error (which is also called the *brute force method*). A problem is however the key length of just 56 bits, which was right away criticized by the cryptographers Whitfield Diffie and Martin Hellman (who will play a crucial role in Section 3.2.4) as being too short. With the rapid development of computers the key length really got intolerably short. Because of this, in 1999 NIST initiated an open competition for a new encryption standard which should be known under the name of *Advanced Encryption Standard (AES)*. This time, there was a lively participation and a contribution from two Belgian cryptographers was chosen for the Advanced Encryption Standard.

The algorithm is similar to its predecessor: Again blocks are handled and the algorithm works in rounds, in each of which the key is fed in and confusion and diffusion is generated. In comparison with DES, this algorithm relies on mathematically clear and elegant constructions. With these constructions it can be proven that particular potential attacks are impossible.

As to be expected, no practically relevant attack has yet been found against AES – provided that an attacker does not have direct information about the computations in the algorithm. With a required key length of at least 128 bits the algorithm then seems to offer optimal security for many decades to come.

The assumption that an attacker does not have direct information about the computations seems to be innocent and clearly satisfied in practice. There are, however, surprising ways in which an attacker can obtain such information. Particularly, there are several practically relevant attacks on "straight-forward" implementations on AES which rely on an analysis of the running times (see [5]).

**3.2.2. Password encryption.** The development of multi-user computers led to a problem: If each user has a password, how can all these passwords be secured against espionage? In particular, how can it be ensured that a system administrator, who has access to the complete system, cannot read the passwords?

Let us assume that we have a function $f$ which assigns to every password an "encrypted password" and which has the following properties: First, $f$ can be computed in a fast manner and second, it is practically impossible to find for an encrypted password $C$ a password $P$ which is mapped to $C$ via $f$, that is, with $f(P) = C$. Then one obtains the following authentication scheme:

Instead of storing the password $P$ of a particular user, let us call her Ursa, one

stores $C = f(P)$. If now a user claiming to be Ursa inputs a password $P'$, one checks if $f(P) = f(P')$ holds. If this is the case, the user is authenticated as Ursa.

This idea was developed by Roger Needham at the University of Cambridge and called *one-way cipher* by his colleague Maurice Wilkies ([34]).

Building upon an encryption scheme like DES or AES, one can realize this idea as follows: One applies the encryption scheme with $P$ as key and a constant plain text like `0...0`; the result is then $f(P)$. If the enciphering scheme is secure, one obtains a function with the desired properties.

Besides the application itself, the idea is interesting for two reasons: First, it illustrates that there is a deep relationship between the classic goal of cryptography, confidentiality, and other goals like authentication. Second, the idea of a function $f$ as described leads to a connection with *complexity theory*.

**3.2.3. Complexity theory.** With the development of computers a new discipline emerged: computer science. Inside of computer science in turn developed *theoretical computer science* which comprises the theoretical study of algorithms in well-defined formal settings. From a scientific point of view, theoretical computer science is part of mathematics.

An important part of theoretical computer science is *complexity theory*. Here questions of the following kind are studied: Let a particular computational problem, for example the addition or the multiplication of natural numbers or the problem of factoring natural numbers, be given. How fast can then computations for larger and larger numbers be performed with an algorithm for an idealized elementary computing device? Thus, not computations for concrete inputs (also called *instances*) and also not inputs of a concrete order of magnitude are considered but all thinkable computations for all (infinitely many) inputs. One hereby imagines that the idealized computing device operates bit-wise, and one measures its running time accordingly. We exemplify this with the mentioned examples, starting with the addition of two natural numbers.

As remarked at the beginning of Section 3.2, the two natural numbers, say $a$ and $b$, shall be given in binary representation. The running time shall be expressed with respect to the input length, which we denote by $\ell$. ($\ell$ is about equal to the sum of the bit-lengths of the the numbers, that is, to $\log_2(a) + \log_2(b)$ if $a, b \neq 0$.) With school-book addition one obtains a running time of at most $C \cdot \ell$ for some constant $C > 0$.

Analogously we consider the problem of multiplying two natural numbers $a$ and $b$. With school-book multiplication one needs no more than $\ell$ additions of natural numbers of input length at most $2\ell$; one obtains thus a running time of at most $C' \cdot \ell^2$ for a constant $C' > 0$.

One expresses this as follows: The upper bound on the running time of the addition algorithm is *linear* in $\ell$ and the upper bound on the running time of the multiplication algorithm is *quadratic* in $\ell$. There are also other methods for multiplication as the usual school-book method. For example, there is a method with which one can achieve a running time of at most $C'' \cdot \ell^{1,5}$ for a constant $C'' > 0$. This upper bound is from a certain size of $\ell$ onward better (that is,

smaller) than the classic, quadratic one – independently of the constants $C'$ and $C''$. In other words, the bound obtained via the alternative method is better than the classic one for all but finitely many inputs.

One says then that the bound obtained with the alternative method is *asymptotically* better. In complexity theory the focus is on such asymptotic statements, and from the point of view of complexity theory the bound obtained with the alternative method is considered to be better. Nevertheless, this does not say anything about which method is faster for CONCRETE numbers $a$ and $b$. Such statements are usually not addressed in complexity theory.

A running time which can be upper-bounded by $C \cdot \ell^k$ for some $C, k > 0$ is called *polynomial*. In complexity theory algorithms with polynomial running time are considered to be "fast" in a qualitative way and simply called "fast" or "efficient". To highlight the complexity theoretic approach, we use the term "qualitatively fast" and also modify other terms of complexity theory accordingly.

We see that the algorithms for the addition or for the multiplication two natural numbers have polynomial running time and are therefore qualitatively fast in the sense just defined.

Let us now consider the problem of integer factorization, where the integer is again variable. No method is known with which one can solve this problem qualitatively fast, that is, in polynomial time. This holds also if one allows that algorithms "throw dice" during a computation, an operation we allow from now on when we speak of an "algorithm". More specifically, no qualitatively fast algorithm is known that computes for a non-negligible portion of products $pq$ of two prime numbers $p, q$ of the same bit-length the factorization, whereby the notion "non-negligible" can also be defined in a precise manner. Even more, no qualitatively fast algorithm is known that computes for a non-negligible portion of products of two natural numbers of the same bit-length a factorization into two natural numbers of the same bit-length.

Following the ideas and notions of complexity theory, the situation just discussed can also be expressed as follows: Let $f$ be the function which assigns to a tuple $(m, n)$ of natural number $m, n$ of the same bit-length its product $mn$, that is, $f(m, n) = mn$. This function is computable qualitatively fast. However, no algorithm is known with which one can compute for a non-negligible portion of values $y$ of the function so-called *preimages* (which are here tuples $(m, n)$ of natural numbers of the same bit-length with $mn = y$) in a qualitatively fast way.

If there really is no such algorithm, the function considered is a so-called *one-way function*. With the notion of one-way functions the practical problem of "one-way encryption" as described in the preceding section is linked with complexity theory.

The now prevailing complexity theoretic point of view on cryptology is completely different from Shannon's. Whereas Shannon addressed the question IN HOW FAR a cipher text determines the plain text or in how far one could compute information on plain texts from cipher texts if one had an arbitrarily large amount of computing power, the complexity theoretic point of view is: Can one in a certain sense compute plain texts from cipher texts in a FAST MANNER or should this task

considered to be infeasible?

Concerning the notion of "fast" one has however to act with caution: As already remarked, in complexity theory, concrete computations (like a concrete multiplication or factorization) are not considered. Rather, qualitative statements are made on the speed of solution methods for computational problems for arbitrary (and thus arbitrarily large) instances, that is, inputs. The statement that a particular function is a one-way function is therefore not a statement on the computational difficulty for concrete instances. There is thus a GAP between the complexity theoretic consideration and a possible practical application in cryptography like in password encryption. Closing this gap is not an easy task.

Also from a theoretical point of view there are problems concerning the notion of one-way function: Not a single function is proven to be indeed a one-way function – even if there are some good candidates for such functions, like the one just described.

The situation is even more tricky: If one can prove for a single function that it is one-way, one has solved the most famous open problem in theoretical computer science and one of the most prominent problems in mathematics, the *P versus NP problem*. This problem is considered to be one of the most difficult open questions of mathematics. It is one of the so-called *millennium problems* by the *Clay Mathematics Institute*; a solution is awarded with one million US dollars. A simple – however non-classic – formulation of this problem is as follows: Is there a qualitatively fast non-randomized algorithm which computes a solution to the following question, called the *subset sum problem*: Given arbitrarily many natural numbers $a_1, \ldots, a_k$ and a further natural number $S$, is the sum over SOME of these numbers equal to $S$? For example, for the four numbers $3; 7; 13; 21$ and $S = 31$ the answer is "yes" because $3 + 7 + 21 = 31$, whereas the answer is "no" if one changes $S$ to 30 or to 32. Now the following is known: If there exists any one-way function, the $P$ versus $NP$ problem has a negative answer, that is, there is no such algorithm.

**3.2.4. Key exchange and cryptography with public keys.** According to Kerckhoffs' principles a cryptographic scheme should be distinguished from the key. The scheme should be generally known, whereas evidently the individual keys have to be kept secret. Two parties who wish to communicate with each other can publicly agree on a common scheme. But it seems to be clear, even self-evident, that the two parties cannot agree in public on a common key.

That this is nonetheless possible was shown by Whitfield Diffie and Martin Hellman in 1976 in a work with the seminal title "New directions in cryptography" ([7]). They presented a scheme which is now called the *Diffie-Hellman method* or *scheme*.

With the mentioned scheme, two people, which in cryptography are always called *Alice* and *Bob*, can agree on a common secret in public.

We briefly present the scheme. For this, we first recall the so-called *modulo computing*:

We choose a natural number $m \geq 2$, the so-called *modulus*. The most fun-

damental operation is now to take the remainder of an integer $a$ with respect to division by $m$. The resulting integer, which is always between 0 and $m-1$ (inclusively) is called the *remainder of $a$ modulo $m$* and is denoted by $a \bmod m$.

Upon computation *modulo $m$* one computes in the set of integers $\{0, 1, \ldots, m-1\}$ and after each operation the remainder modulo $m$ is taken: If $a$ and $b$ are two natural numbers smaller than $m$, the result of *modulo addition* is $(a+b) \bmod m$ and the result of *modulo multiplication* is $(a \cdot b) \bmod m$. Moreover, for the given integer $a$ and a natural number $e$ the result of *modulo exponentiation* is $a^e \bmod m$.

For a prime number $p$ and two natural numbers $a, b$ between 1 and $p-1$ (1 and $p-1$ included) one has also $(a \cdot b) \bmod p \neq 0$. This can be seen as an analogue to the fact that the product of two non-zero integers is also non-zero. One calls the domain $\{0, \ldots, p-1\}$ with the given operations of computation the *finite prime field* with respect to the prime number $p$ and denotes it by $\mathbb{F}_p$. These domains of computation have in various aspects analogous properties to the domain of rational numbers, $\mathbb{Q}$; they can be seen as finite analogs of the infinitely large domain $\mathbb{Q}$.

An important aspect of modular computation with respect to the scheme by Diffie and Hellman is: If $p, a$ and $e$ are given, one can compute $a^e \bmod p$ qualitatively fast, that is, in polynomial time.

The scheme is now as follows: First, Alice and Bob agree on a (large) prime number $p$ and a positive integer $a < p$. These two numbers can be known to everybody and might be shared by a large group of people. For this reason, we call $p$ and $a$ *public parameters*. Now, Alice chooses a natural number $x < p-1$ and Bob chooses a natural number $y < p-1$ at random. Both keep their numbers secret. Alice then computes $X := a^x \bmod p$ and sends this to Bob and he computes $Y := a^y \bmod p$ and sends this to Alice. Now, $X^y \bmod p = a^{xy} \bmod p = Y^x \bmod p$. This integer can be computed by both and shall be the common secret.

We now take the view of an eavesdropper who wants to compute the presumed secret. He receives $p, a, X, Y$ and wants to compute $X^y \bmod p = Y^x \bmod p$. The problem to perform such a computation is now called the *Diffie-Hellman problem*.

A possible approach is to compute from $p, a, X$ a natural number $x$ with $a^x \bmod p = X$ because then the eavesdropper can easily also compute $Y^x \bmod p$. This computational problem is the so-called *discrete logarithm problem*.

A year after Diffie and Hellman's work, Ron Rivest, Adi Shamir and Leonard Adleman published an encryption scheme in which every user has a pair of keys consisting of a *private* and a *public* key ([28]). The idea of this so-called *RSA-scheme* is: Alice can distribute her public key and every person can use this key to send Alice an encrypted message, but only Alice can read the message with her private key. As one now uses two keys with distinctively different roles, one speaks here of an *asymmetric encryption scheme*, whereas the classical encryption schemes with a single, common key are called *symmetric encryption schemes*.

Also the RSA scheme is based on the difficulty of a computational problem, in this case of the problem of factoring the product of two (large) prime numbers; this problem was already discussed above. According to today's knowledge, the discrete logarithm problem (with random choice of $p$ and $a$) and the factorization

**The Diffie-Hellman protocol**

Alice and Bob (in public) agree on a suitable prime number $p$ and a suitable natural number $a < p$.

| **Alice** | **Bob** |
|---|---|
| Chooses $x < \mathrm{ord}(a)$. | Chooses $y < \mathrm{ord}(a)$. |
| Computes $X := a^x \bmod p$. | Computes $Y := a^y \bmod p$. |
| Sends $X$ zu Bob. | Sends $Y$ to Alice. |
| | $\xrightarrow{X}$ |
| | $\xleftarrow{Y}$ |
| Computes $Y^x \bmod p$. | Computes $X^y \bmod p$. |

Alice and Bob computed the same number, as $Y^x \bmod p = a^{xy} \bmod p = X^y \bmod p$.

problem can be considered to be equally difficult at equal input length; more on the difficulty of the discrete logarithm problem in sections 3.2.6 and 4.4.2.

In addition to encrypting texts, the RSA scheme can also be used for signing: A text can be signed by encrypting it with the private key. An alleged signature can then be checked by decryption with the public key and comparison with the original text. Cryptographic schemes relying on the use of public keys form what is now called the area of *public key cryptography*.

For their contributions to cryptography, both Rivest, Shamir and Adleman as well as Diffie and Hellman received the Turing Award, the most prestigious prize in computer science, named after one of the pioneers of computer science, Alan Turing: Rivest, Shamir and Adleman received the prize in 2002 and Diffie and Hellman in 2015.

**3.2.5. Protocols, active attackers and reductive security results.** Let us assume that Alice wants to communicate confidentially with Bob over the internet. She uses the Diffie-Hellman scheme with concrete parameters $p$ and $a$ to establish a common secret with Bob and then the cipher AES with the common secret as the key.

As just shown, an eavesdropper can break the Diffie-Hellman scheme (for concrete parameters $p$ and $a$) if and only if he can solve the Diffie-Hellman problem for the same parameters. One could be tempted to conclude from this that Alice has chosen an adequate scheme for her goal if the Diffie-Hellman problem is unsolvable for the concrete parameters and no attack on AES is known.

This is however not the case. The reason is simple: From the moment of the

key exchange on, an attacker could intercept her communication with Bob and masquerades as Bob with respect to Alice. Alice would then send confidential information to the attacker.

Maybe Alice would notice this after some time. An attacker can however proceed even more skillfully: He masquerades as Bob towards Alice and as Alice towards Bob and establishes a common key with each of them. He can then decrypt the messages from Alice to Bob and from Bob to Alice, read them, reencrypt them and send them off again. One speaks here of a so-called *man-in-the-middle attack*.

The demonstrated dysfunctionality of the scheme was due to the complete lack of authentication. The shown problems can however also be interpreted more abstractly: A cryptographic scheme can be INSECURE with respect to an ACTIVE ATTACKER even if it is SECURE AGAINST AN EAVESDROPPER, that is, a PASSIVE ATTACKER.

Whereas the obvious insecurity of the Diffie-Hellman scheme against active attacks highlights the importance to consider such attacks, such attacks were already relevant for classic cryptographic schemes. One idea for an active attack was already described in Section 2.2: One tricks a user of a cryptographic scheme to encrypt a given message and tries to obtain from the resulting pair of plain and cipher texts information about other received encrypted messages

Cryptographic schemes, in particular interactive cryptographic schemes, allow for a confusing magnitude of manipulation possibilities by active attackers. Some of these are evident, others in turn are not. To be convinced of the security of a scheme (for concrete parameters), one would like to have a strong argumentation that the scheme stays secure no matter which strategy an attacker chooses.

For just about every scheme in use today, it is thinkable that the security can be compromised even by passive attackers if for a particular basic algorithmic problem a new, surprisingly efficient algorithm is found. For example, the security of the Diffie-Hellman scheme with concrete parameters relies on the difficulty of the corresponding Diffie-Hellman problem with the same parameters. As such basic attacks cannot be avoided anyway, one tries to base the argumentation explicitly on the difficulty of some underlying problem. For a given task, like encryption, the goal is then to find an efficient scheme for which a large class of attacks can be ruled out if only a basic algorithmic problem is sufficiently difficult.

Such an approach requires adequate mathematically rigorous but also manageable definitions. Just the task to find such definitions is not an easy one.

The problems already start with the notion of "scheme". The usual mathematical definition is based on interacting algorithms whose inner computations are invisible for the attacker. One can also say that one abstracts from the inner computations and only considers the input-output-relationships. Such an abstracted scheme is called a *protocol* in cryptology. A protocol can be used for arbitrarily large inputs and comprises all necessary steps. For example, a protocol for the informally described Diffie-Hellman scheme begins with a setup phase. In this phase, after the input of some parameter size (like "1000 bits" or so), a suitable pair $(p, a)$ is chosen. Such a protocol is to be distinguished from an *implementation* which

consists of computer programs realizing the protocol. An *attacker* is then always an algorithm which interacts with the protocol.

An even greater challenge is finding adequate formal definitions for "secure" for different aspects of cryptography like establishing a common secret, encryption or decryption.

The established definitions mirror the complexity theoretic point of view described in Section 3.2.3. This means that not absolute statements on the security for concrete input lengths but QUALITATIVE statements for arbitrarily large input lengths are made. Following our general terminology introduced in Section 3.2.3, we emphasize this by using the term "qualitatively secure".

There are several, related formal definitions of *qualitatively secure* based on different attack scenarios. Most commonly, one nowadays bases the definitions on the idea of *games*. One hereby imagines that an "intelligent" attacker (also called *adversary*) plays a game against a simplistic *challenger*. One should hereby keep in mind that in fact the "attacker" is merely an algorithm.

To give an idea of this approach, we now give a slightly informal description of the strongest currently considered notion of qualitative security for symmetric encryption schemes, *Indistinguishability under adaptive chosen cipher text attack (IND-CCA2-security)*. It is based on the following game:

(1) After the input of the key length, a secret key is chosen (in a randomized way) by a so-called setup algorithm and given to the challenger.

(2) The attacker chooses some texts and sends them to the challenger with the request to either de- or encrypt them. The challenger sends the results back to the attacker.

(3) The attacker chooses two different texts $M_1$ and $M_2$. He sends them both to the challenger. The challenger chooses one of the texts with equal probability, encrypts it to a text $C$ and sends $C$ back to the attacker.

(4) The attacker again chooses some texts to be encrypted and some texts different from $C$ to be decrypted. He sends them to the challenger who performs the desired operations and sends the results back to the attacker.

(5) The attacker opts for $M_1$ or $M_2$.

During the whole game, the attacker can adapt his strategy according to previously obtained information. The attacker *wins* if he opts for the same text which the challenger has chosen in Step 3. Nota bene: If the attacker merely guesses, he wins with a probability of $\frac{1}{2}$.

Let us call an attacker *qualitatively successful* if it is qualitatively fast and it wins with a probability which is non-negligibly larger than $\frac{1}{2}$. Then a scheme is called *IND-CCA2-secure* if there is no qualitatively successful attacker for the game just described.

Coming back to the game, let us note that in Step 3 the attacker may even choose $M_1$ or $M_2$ (or both) to be identical to a plain text chosen in Step 2. Similarly, in Step 4 the attacker may send the texts $M_1$ or $M_2$ to the challenger for encryption. The only request which is not allowed is to ask for the decryption of $C$. This

implies that in order that an encryption scheme can be IND-CCA2-secure, it must be randomized.

To base definitions of security on games was a landmark idea. For this idea and related contributions to the mathematical foundations of cryptography, Shafi Goldwasser and Silvio Micali received the Turing Award in 2012.

On the basis of an attack scenario as the one given one can then try to establish a *reductive security result* for a given scheme. For this, one additionally fixes an underlying computational problem. A reductive security result, also called a *security reduction* or simply a *reduction*, is then a mathematical statement of the form: Every qualitatively successful attack of the considered kind on the protocol leads to a qualitatively fast algorithm for the computational problem. If such a result has been proven, one obtains: If there is no qualitatively fast algorithm for the computational problem, the protocol is qualitatively secure with respect to the considered kind of attacks.

Ideally, the security of suitable protocols (with respect to a wide range of attacks) for a multiplicity of cryptographic applications of modern times, like encryption, signature or authentication, would be reduced via reductive security results to a small number of algorithmic problems like to the factorization problem, the discrete logarithm problem or the Diffie-Hellman problem. These basic problem would then be studied exhaustively by the scientific community. This rigorous approach is particularly advocated in a two volume work by Oded Goldreich from 2001 and 2004 called *Foundations of Cryptography* ([14]), which can be seen as a first consolidation of the subject.

One should however note that, just as in Section 3.2.3, there is always a GAP between complexity theoretic considerations and praxis. Concerning the practical use of a protocol, there are indeed several potential problems, even if a reductive security result for a broad attack scenario and a seemingly strong underlying problem has been proven:

- For a practically useful result it must be determined for which input lengths (or parameters) a protocol shall be used. If one takes the approach via reductive security results seriously, one must proceed as follows: One chooses a reductive security result with respect to a seemingly strong attack scenario and underlying computational problem. The result must not only qualitatively but explicitly and quantitatively relate the computational complexity of attacks and of the computational problem. One reasons for which input length this algorithmic problem is practically unsolvable. On the basis of these two statements, one computes how large the key length has to be in order that no practically relevant attack is possible if indeed the underlying problem is as difficult as assumed.

  This is often not done, in particular because the key length would then be unmanageably large and/or the scheme too slow. Rather, often shorter key lengths are chosen or other scheme are considered which are inspired by the rigorously analyzed one, but nonetheless different.
- Self-evidently the implementation has to correspond to the description, it may therefore not contain any mistakes. To rule this out is difficult.

- Even on implementations which correspond to the specifications there are often nonetheless attack possibilities. Notwithstanding that the attack scenarios considered are very broad, it is always assumed that an attacker does not know anything about the internal computations. However, concrete products often "radiate" in the literal and the figurative sense and this "radiation" might be be used for subtle attacks. An example are the attacks via running time already mentioned in Section 3.2.1.

**3.2.6. The influence of number theory.** From a mathematical point of view, the discrete logarithm problem and the factorization problem fall not only in the realm of complexity theory but also in that of number theory. Thus the work by Diffie and Hellman created a connection between cryptology and this well-established field of pure mathematics. This connection is remarkable as mathematicians assumed just a few decades ago that especially number theory is immune against applications, in particular for military purposes. To this effect, the famous number theorist Godfrey Harold Hardy writes in his *A Mathematician's Apology* ([17]) from the war year 1940 that "real" (that is, deep) mathematics is "harmless and innocent" and concretely: "No one has yet discovered any warlike purpose to be served by the theory of numbers or relativity, and it seems very unlikely that anyone will do so for many years."

Cryptology is now unimaginable without number theory and related areas of mathematics. The importance of number theoretic methods gets particularly clear with the discrete logarithm problem, that is, the following algorithmic problem:

Given a prime number $p$, a positive integer $a < p$ and a further positive integer $b < p$ for which there is an $x$ with $a^x \bmod p = b$, compute such an $x$.

The obvious first try is to solve the problem by brute force, that is, to test for given $p, a, b$ consecutively for $x = 1, 2, 3, \ldots$ whether the equation $a^x \bmod p = b$ is satisfied.

To secure a system against this basic attack, the public parameters $p, a$ must be chosen appropriately. But how can the running time be estimated for given $p, a$? Already for this basic question, elementary number theory is relevant:

As one might expect, the number of possible values $b = a^x \bmod p$ is crucial. This number is called the *order of $a$ modulo $p$* and denoted $\mathrm{ord}(a)$. The values $a^x \bmod p$ lie between 1 and $p-1$ (inclusively), therefore the order is at most $p-1$. Furthermore, it holds, as can be shown: $a^{\mathrm{ord}(a)} \bmod p = 1$. If one multiplies this consecutively by $a$, one obtains $a^{\mathrm{ord}(a)+1} \bmod p = a$, $a^{\mathrm{ord}(a)+2} \bmod p = a^2 \bmod p$ and in full generality for every natural number $e$: $a^{\mathrm{ord}(a)+e} \bmod p = a^e \bmod p$. It follows that every possible value of $a^x \bmod p$ is taken for exactly one $x$ between 0 and $\mathrm{ord}(a) - 1$ (inclusively). If one lets $x$ run from 0 to $\mathrm{ord}(a) - 1$, there is thus exactly one $x$ with $a^x \bmod p = b$. For fixed $p$ and $a$ and completely random $b$ one needs in average $\frac{\mathrm{ord}(a)}{2}$ tries until one has found the solution.

Already Carl Friedrich Gauß proved that in his *Disquisitiones Arithmeticae*, published in 1801 ([11]), that for every prime number $p$ there is an $a$ of order $p-1$. Let us consider such a pair, that is, let $p$ be a prime and let $a < p$ be a positive integer of order $p - 1$.

For the brute force algorithm considered so far, one needs about a time which is given by $p$. By comparison: To compute for given $p, a$ and an $x < p$ the value $a^x \bmod p$, one needs a running time which is about given by $\log_2(p)^3$. If for example $p$ has 100 bits (about 30 positions in the decimal system), $\log_2(p)^3$ is about 300 whereas $p$ is about $2^{100}$, that is, about $10^{30}$. The difference is enormous.

At the time of publication of the article by Diffie and Hellman it already was known that one cannot only obtain a running time of about $p$ but of about $\sqrt{p}$. The idea of this can be described as follows: One computes numbers $a^c \bmod p$ and $a^d b \bmod p$ for arbitrary natural numbers $c$ and $d$ smaller than $p$ and saves the results. Then one searches for a so-called *collision* $a^c \bmod p = a^d b \bmod p$. Such a collision leads to $a^{c-d} \bmod p = b$ if $c \geq d$ and to $a^{c-d+(p-1)} \bmod p = b$ if $c < d$. Maybe surprisingly one needs on average only about $\sqrt{p}$ results $a^c \bmod p$ and $a^d b \bmod p$ before one can find a collision as desired.

With the help of a classic number theoretic method, known under the name *Chinese Remainder Theorem*, this can be further improved. In total, one can obtain a running time of about $\sqrt{\ell}$, where $\ell$ is the largest prime divisor of $p - 1$.

Now, for a prime $p \geq 5$ the integer $p - 1$ is never prime as it is even and not 2. It is therefore of interest to consider primes $p$ for which $\frac{p-1}{2}$ is also a prime. Such primes are called *Sophie-Germain primes*. Interestingly, it is not proven that there are infinitely many such primes, this is however conjectured.

If we consider Sophie-Germain primes, the running time of the best methods considered so far is again about $\sqrt{p}$.

There is however yet another method to solve the discrete logarithm problem, which can be called the *relation method*. This method is considerably more efficient than the collision method if the order of $a$ is about that of $p$. It was already described by the mathematician Maurice Kraitchik in his *Théorie des Nombres* from 1922 ([22]) but fell into oblivion and was rediscovered after the publication of the work by Diffie and Hellman. We will not present the method here and solely remark that the relation method uses that one can factorize many natural numbers into products of substantially smaller (prime) numbers.

A natural question is now if there is a variant of the described discrete logarithm problem for which the mentioned algorithms do not work. For this one wants to substitute the domain of computation $\{1, \ldots, p-1\}$ with modulo multiplication by another suitable domain with a completely different computing operation. It turns out that one cannot avoid the collision method under any circumstances. The reason is that the collision method relies directly on the computing operation itself. But is there a domain in which no better method is known, in which in particular the relation method does not work?

To put this idea into practice, Neal Koblitz and Victor Miller in 1981 independently proposed what is now called *elliptic curve cryptography*.

Elliptic curves are not ellipses, even if the name suggests this; the name relies on a "historical coincidence". However, one can explain what elliptic curves are by starting with the circle, which is a particular ellipse: The equation $x^2 + y^2 = 1$ describes the so-called *circle of unity* in the Cartesian coordinate system, that is, the circle of radius 1 around the origin. Every point $P$ on the circle can

be given by the angle $\alpha$ that it has to the $y$-axis; one then has $P = (\sin(\alpha), \cos(\alpha))$. If now such a point $P$ and a further point $Q = (\sin(\beta), \cos(\beta))$ are given, one can add the angles and obtain in this way a new point $R = (\sin(\alpha + \beta), \cos(\alpha + \beta))$. Let us write $P \star Q$ for this point $R$, where the symbol "$\star$" is arbitrary and could be substituted by another symbol.

We obtain in this way a computing operation on the circle with radius 1, which might be called the "clock operation". This operation fulfills the usual rules of associativity and commutativity known from the addition or the multiplication of real numbers. Moreover, with $O := (0, 1)$ one has $P \star O = O \star P = P$ for every point $P$; the point $O$ is therefore analogous to 0 for the addition of real numbers and to 1 for the multiplication of real numbers.

One does not need angles and trigonometric functions for the computing operation on the circle: Given points $P = (x_P, y_P)$ and $Q = (x_Q, y_Q)$, the coordinates of the resulting point $R = P \star Q$ are given by the purely algebraic formulae

$$x_R = x_P y_Q + x_Q y_P \quad \text{and} \quad y_R = y_P y_Q - x_P x_Q \; . \tag{1}$$

If one now chooses a negative number $d$, the equation

$$x^2 + y^2 = 1 + dx^2 y^2 \tag{2}$$

describes an elliptic curve ([2]). Interestingly, one can also define a computing operation on such a curve. For points $P$ and $Q$, the coordinates of the resulting point $R = P \star Q$ are now given by the formulae

$$x_R = \frac{x_P y_Q + x_Q y_P}{1 + d x_P x_Q y_P y_Q} \quad \text{and} \quad y_R = \frac{y_P y_Q - x_P x_Q}{1 - d x_P x_Q y_P y_Q} \; . \tag{3}$$

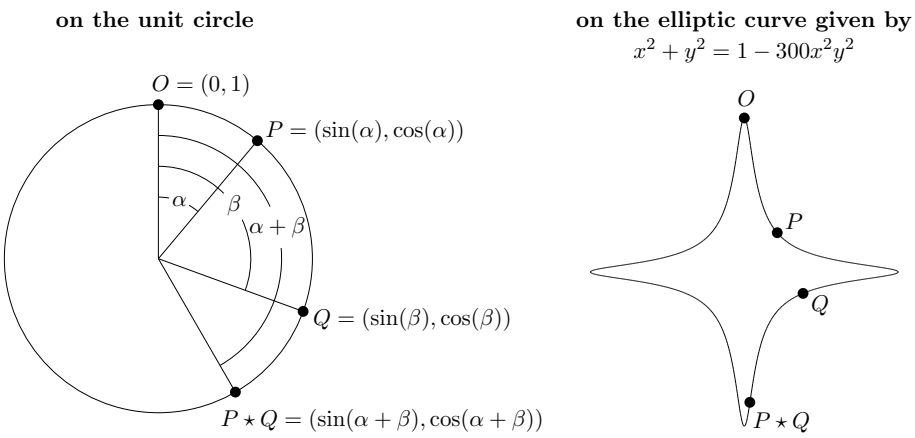
This operation is again associative and commutative and one again has $O \star P = P \star O = P$ for every point $P$.

One shall note that for $d = 0$ one would obtain again the circle with the computing operation (1), which is however not an elliptic curve. We also mention that the condition that $d$ is negative ensures that the nominators in the formulae are always non-zero and remark – because we will use this shortly – that the condition on $d$ is equivalent to $d$ not being the square of another real number.

In cryptography, not solution sets of such equations over the real of the rational numbers but over finite computing domains are considered. Most often one uses the finite prime fields $\mathbb{F}_p = \{0, 1, \ldots p-1\}$ described in Section 3.2.4, to which we want to restrict ourselves here.

So let $p$ be a prime larger than 2 and $d \in \mathbb{F}_p$ with $d \neq 0$ and let us consider the solutions to the equation (2) in $\mathbb{F}_p$, that is, the tuples $(x, y)$ with $x, y \in \mathbb{F}_p$ and $(x^2 + y^2) \bmod p = (1 + dx^2 y^2) \bmod p$. In order to obtain a computing operation on the solution set, one again has to ensure that the denominators in (3) are always non-zero. For this one uses the condition that $d$ shall not be the square of another element, which is now also adapted to the computation in $\mathbb{F}_p$. This means that there shall be no $a \in \mathbb{F}_p$ with $a^2 \bmod p = d$; there are exactly $\frac{p-1}{2}$ such elements in $\mathbb{F}_p$.

The resulting domain of computation is usually denoted by $E(\mathbb{F}_p)$. The idea is now to substitute the domain $\{1, \ldots, p-1\}$ with modular multiplication by such a




domain $E(\mathbb{F}_p)$. Indeed, this is easily possible. One can then again speak of discrete logarithms and also of the *elliptic curve discrete logarithm problem* and one can adapt schemes as the one by Diffie and Hellman to this setting.

In the meantime, a large number of cryptographic schemes have been developed which rely on modular multiplication and which can be adapted to elliptic curves. Hereby in particular a method by Taher ElGamal, which can be seen as an alternative to RSA, is worthwhile mentioning.

After 30 years of research, for most of the considered computing domains $E(\mathbb{F}_p)$ the collision method is still the most efficient method to solve the elliptic curve discrete logarithm problem. This means that for equal key length the variant of the protocol by Diffie and Hellman with elliptic curves leads to a much higher security level than the original protocol. The same holds true for other cryptographic protocols whose security rests on the discrete logarithm problem.

For example, it is recommended by the German Bundesamt für Sicherheit in der Informationstechnik (BSI) to use a key length of at least 250 bits if the discrete logarithm problem for elliptic curves is employed whereas for the classic problem and for the RSA-method a key length of at least 3000 bits is recommend ([4]).

A reader interested in public key cryptography and its number theoretic foundations might consider the book *Mathematics of Public Key Cryptography* by Steven Galbraith ([9]).

**3.2.7. Smart cards.** In the presentation of public key cryptography so far, a fundamental technical problem was put aside: How can the private key be secured from unwanted access? One can store the key by a variety of means. For example, in the 1980s the key could be saved on a floppy disk. To use the key it has however first to be loaded into the storage of a computer. This constitutes a security risk, in particular if the computer is connected to other computers, which is normally

the case nowadays.

A clean solution is to create a small device containing its own miniature computer with storage and microprocessor. Then cryptographic applications like authentication and digital signature can be realized without the key ever leaving the device.

In the beginning of the 1980s miniaturization was advanced enough that the idea could be realized with very thin and about a square centimeter large microcomputers. Applied to plastic cards in credit card size, one obtains so the so-called *smart cards*.

The secret, the private key, shall not only not leave the card in the normal mode of operation, but under no circumstances shall it be possible to obtain usable information on the internal computations of the card, even if an attacker controls the environment of the card completely. This means for example that the card must not reveal information by radiation or by power consumption.

If the card falls into the wrong hands, it should be completely useless. For this, it is secured via a password, whereby the card locks itself by repeated faulty insertion. After this, even with physical manipulation or partial destruction of the card, it should be impossible to use the card or to obtain information on the key, which is after all stored on the card.

The magnitude of potential attacks poses a challenge for card manufacturers with which these however seem to cope well.

## 4. Current developments and challenges

**4.1. Ubiquitous applications and interconnectedness.** Whereas governments and companies have been using cryptography for data security already since the 1970s, since the spread of the World Wide Web, the everyday life of many, in industrialized countries arguably of most people cannot be imagined anymore without cryptography.

Public key cryptography is automatically used if one invokes a site with an address of the form `https://...`, which in turn is regularly the case if one transacts a payment. Here the user is informed about the use of cryptography, at other places this happens "transparently" for the user, that is, without him noticing anything.

Around the automobile alone, cryptography is employed multiple times. A classic application are electronic keys and immobilizers. Also in systems for road tolls, for example in the German Toll Collect system, cryptography is used. Since 2006, manipulations of the tachograph, mandatory for buses and trucks in the European Union, are prevented with cryptography. Completely new challenges are posed by communication from car to car, particularly if this influences the automatic behavior of cars. Additionally to using cryptography inside of applications, cryptography is also used to protect software in electronic devices against manipulation. As more and more electronic devices are used inside security relevant systems such as brakes or the steering system, this protection becomes ever more important.

Three interconnected trends are visible: More and more devices and products are equipped with microprocessors, these devices are more and more interconnected and thereby access to common storage outside of the devices is ever more important.

In both, business and private domains it becomes more and more common to not store data locally but in the "cloud" and to use services such as dropbox, google drive or icloud. All these services employ cryptography, it remains however the question in how far the data is really secure against access by employees or public authorities. If however the data is encrypted before sending it off, this problem is avoided. It then nonetheless remains the question if locally installed software does exactly what it is supposed to do or if it does not send off "a bit more".

With the interconnection, in the domestic domain too cryptographic challenges have developed or are going to develop in the near future. For example, many states of the European Union want to achieve that a major part of households are equipped with so-called "smart meters" by the year 2020. The thereby occurring data security problems shall be solved with cryptography. In the context of the German *Energiewende* (energy turn) and similar policies around the globe, "smart grids" with automatized turn on/off of devices by the grid administrator are promoted. Interesting cryptographic challenges are ahead here.

Even complete industrial complexes and critical infrastructures such as pipelines, electric grids or power stations are being connected to the internet, what gives rise to all kinds of horror scenarios. Here a radical solution against threat scenarios is obviously to separate the system control completely from the communication to the exterior. However, often one wants to avoid access from the exterior for security reasons while, yet at the same time allow external monitoring. As already the existence of a canal accessible from the exterior can be seen as a weakness, the establishment of a channel which physically makes data transfer to the interior impossible might be a solution. The company Waterfall Securities Solutions offers products based on a laser. With this technology systems can be created which offers a much higher level of security than electronic devices relying on cryptographic schemes.

**4.2. The future of computers.** The computing power of computers in relation to the amount of money to be spent has increased rapidly and without interruption since World War II. With this development, over and over, cryptographic systems once thought to be secure became attackable. What will hold the future and which impact will it have on cryptology?

Since the beginning of the 1970s, the integration density of microprocessors grew exponentially. It doubled about every two years, a fact which is known as *Moore's law*. Yet due to clear physical boundaries, this increase of integration density cannot continue arbitrarily. Concretely, currently in the most modern factories the so-called 14 nanometer technology is used, which is already remarkable if one considers that a silicon atom has a radius of about 0.1 nanometers. There are plans for a 5 nanometer technology for about the year 2020 for which one already has to fight with physical boundaries. Below 7 nanometers, one has to cope with

the effect of quantum tunneling, which means that electrons can pass the logic gates unwantedly. One can therefore assume that the hitherto exponential grow will fade out relatively soon.

Yet it is thinkable that this development leads to a possibility to use quantum mechanical phenomena in a completely new way. The quantum world is a very strange world for humans. It surmounts human imagination over and over again, as can be seen with the different interpretations of quantum mechanics.

Already in the year 1981, the physicist Richard Feynman formulated the idea of a "quantum computer" and in the year 1994 the mathematician Peter Shor published a sketch of potential quantum computers for the factorization and the discrete logarithm problems. He showed that in a mathematical model in which analogously to classic computational models quantum computers are described in an idealized way, these problems can be solved with polynomial cost, that is, qualitatively efficiently. For this breakthrough result Shor received the prestigious Rolf Nevanlinna Prize in mathematical aspects of information sciences by the International Mathematical Union in 1998.

Just as the collision method described in Section 3.2.6, Shor's method for the computation of discrete logarithms relies directly on the computing operation. It is therefore not only applicable to the classic discrete logarithm problem modulo a prime number, but also to the problem in elliptic curves. Interestingly, the current main advantage of systems based on elliptic curves, the relatively short key length, could make such systems particularly vulnerable against quantum computers.

However, it is currently unclear if such a quantum computer will ever be built which can keep up with a classic computer. The hitherto tries in any case are slightly sobering: In the year 2001 with the help of a quantum computer based on Shor's ideas the number 15 was factored and the current record from the year 2012 is the number 21.[1]

In the potential quantum computers as envisioned by Shor, analogously to classic computers, the states are manipulated step by step in the course of time. There is also a different, more passive method to make quantum phenomena usable. With this method, Shor's ideas cannot be realized, but it is thinkable that for particular applications it leads to surprisingly fast computers. The company D-Wave develops computers which are based on this passive method. The computers seem to function as planned, but a prognosis on the capabilities of this technology currently seems to be hardly possible.

Interestingly, the American National Security Agency (NSA) seems to assume that quantum computers as envisioned by Shor can be realized in the coming decades. On August 19 2015, the agency announced that for publicly recommended cryptographic schemes, which includes schemes for communication with the US-government, it will "initiate a transition to quantum resistant algorithms in the not too distant future" ([26]).

[1] This is not a misprint; it is about the numbers 15 and 21, not about numbers with 15, respectively 21 bits.

**4.3. Crypto currencies and crypto contracts.** The crypto currency *bitcoin* is on everyone's lips. The development of its exchange rate is impressive. 10 000 bitcoins were offered by a programmer in the year 2010 for two pizzas, the prompt delivery of which led to the first payment in bitcoins. For the all time high up to now of 905 Euros per bitcoin in December 2013, one obtains, purely arithmetically, a solid price of about 9 million Euros. Also with the rate of around 400 Euros per bitcoin at the beginning of 2016, the pizzas were rather expensive.

Even though this development is impressive, at the moment there are no indications that bitcoins could indeed become relevant in day-to-day use.

The bitcoin payment system together with the anonymization software TOR is however one of the prominent technologies for anonymized trading platforms in the internet. The most well known of these markets was Silk Road, operating from 2011 to 2013. As this site was closed by the FBI in 2013, the agency declared that in about 1 million transactions a turnover of over 9 million bitcoins had been achieved. The physicist Ross Ulbricht was identified as the administrator of the site known under the pseudonym Dread Pirate Roberts and sentenced because of drug trafficking and other felonies to life long imprisonment without eligibility for parole. Ulbricht declared in the trail that he established the site because of idealism, "to empower people to be able to make choices in their life, for themselves in privacy and anonymity" and not being involved at a later time. One did not believe him.

Despite the drastic judgment, Silk Road will surely not be the last successful anonymous market place, by whatever motivation it will be established. A particularly interesting situation could occur if a state declared the operation of such a worldwide reachable marketplace legal. Due to the outlook of a turnover in billions of Euros, particularly for smaller and poorer states there is an incentive to do so, if only the profit is taxed.

Even if the bitcoin system apparently does not prepare for a giant leap, this could be the case for an aspect of the bitcoin protocol: the *blockchain technology*.

Superficially, with the bitcoin protocol "electronic coins" are transferred. A first idea would be to administrate the rights on all coins in a single ledger. The administrator of the ledger would then be a kind of deposit bank (without lending). With the bitcoin protocol a more sophisticated decentralized scheme is realized.

Public key cryptography on the basis of elliptic curves is used in about the following way: If Alice wants to transfer a coin to Bob, she signs with her private key a composition of the coin and Bob's public key. It must now be ensured that Alice indeed has the right to transfer the coin. For this, (essentially) all transactions since the beginning of the bitcoin system are stored. At first sight, this solution seems to contradict the desired anonymity of the bitcoin system. This is achieved by "Alice", "Bob" et cetera being only virtual concepts which are created anew over and which can act as fronts for arbitrary persons.

The ledger of transactions is not only stored once but in many different so-called knots. More concretely, new transactions are arranged in blocks and every 10 minutes in all knots the same new block is attached to the stored ledger. The resulting multiply stored ledger is called the *blockchain*.

By modifying the bitcoin protocol slightly, one can build systems for the decentralized storage and transfer of different categories of rights. To start with, one might use such a system for the management of bonds.

Bonds are administrated by so-called central security depositories and from a transaction to the settlement generally one to three days are passed during which the contracting partners have a mutual counterparty risk. With a scheme based on the blockchain technology, the central security depositories would be omitted and the settlement could occur a few minutes after the deal. This would be more efficient as well less risky.

Currently, there is a literal hype about the blockchain technology and it seems that in contrast to the anarchic bitcoin system this technology will indeed change the world of finance.

**4.4. Edward Snowden and the NSA.** The "Edward Snowden Story" is usually perceived by the public as a "real life thriller". The disclosures themselves have led to a vague feeling that "they surveil everything anyway".

With the disclosures indeed a substantial acquisitiveness became evident. On the other hand: That a secret service responsible for the surveillance and analysis of electronic communication surveils – presumably in line with the laws of its home country – exactly this communication and analyzes it with filter technologies should be obvious. It should be even more obvious that this technology is also used to surveil target subjects abroad, in particular after the corresponding country has suffered a massive terror attack.

It should have also been generally known that the sending of an email can be compared to the sending of a postcard, which everybody can read who gets it into his hand.

More interesting is the question concerning the abilities of the NSA regarding encrypted communication, a question on which speculations were made for years. The documents now in the public domain allow for the first time a look at the capabilities of the NSA in this area.

According to the documents, the NSA pursues a large-scale cryptanalytic program with the name *Bullrun*. In a presentation of the British partner service GCHQ it is written that Bullrun "covers the ability to defeat encryption used in specific network communications" and "includes multiple, extremely sensitive, sources and methods". It would offer "groundbreaking capabilities", would be "extremely fragile" and the addressees must "not ask about or speculate on sources or methods underpinning Bullrun successes" ([12]).

As however the further documents and known facts invite to such speculations, we will now do exactly this.

**4.4.1. Proactive approach.** The NSA does not wait until schemes are established but rather attempts to steer the development into a direction favorable to the agency.

As per one of the revealed documents, in the years 2011 to 2013 an amount between 250 and 300 million US dollars was provided for the "SIGINT Enabling

Project" ([27]). According to the project description, the "project actively engages the US and foreign IT industries to covertly influence and/or overtly leverage their commercial products' designs. These design changes make the systems in question exploitable through SIGINT collection [...] with foreknowledge of the modification. To the consumer and other adversaries, however, the systems' security remains intact." The resources of the project shall among others be used to "insert vulnerabilities" into systems and to "influence policies, standards and specifications for commercial public key technologies".

This apparently has been successful at least once, with the so-called *Dual-EC-Deterministic Random Bit Generator*, *Dual_EC_DRBG* for short.

Cryptographic protocols often need "randomness", which must first be generated in the computer. One can indeed generate "true randomness" by physical means, but this is rather time consuming. One therefore uses what is called a *pseudo random number generator* or a *deterministic random bit generator* to generate from a truly random bit-string a substantially longer bit-string. The essential requirements to such a generator are that it is very fast and that only with unrealistically large computing power the output can be distinguished from a truly random bit-string.

In the year 2006, the US-American standardization agency NIST published a "Recommendation for Random Number Generation Using Deterministic Random Bit Generators". As already remarked in Section 3.2.1, all standards and "recommendations" of NIST formally only apply to the US-government and its contractors, but often become de facto industry standards.

One class of generators in the document is the Dual_EC_DRBG. For this class of generators, also an exact specification with concrete parameters is contained in the document. This specification can be seen as the first standard with secret, but in hindsight obvious backdoor.

Already two years before the publication of the document of NIST the such specified generator was implemented by the company RSA Security as the default option in the widely used cryptographic software library BSAFE and only removed after the disclosures by Edward Snowden.

The "Dual_EC" stands for "dual elliptic curve" and indeed elliptic curves are of particular relevance for this class of generators. In contrast to other generators, the construction principle is particularly clear and mathematically elegant. This class of generators is a more efficient variant of the best known and most well studied class of generators in complexity oriented cryptography. For appropriate choices of parameters, the qualitative security of the generators can be reduced to a problem similar to the Diffie-Hellman problem.

In practice however, these generators have the disadvantage of being much slower than competing generators. Furthermore, as was already pointed out in 2005, the concrete generator specified by NIST can be distinguished from a truly random sequence, even if the derivation is small ([13]). Obviously, from the point of view of security, the parameters where chosen deliberately in the wrong way.

As argued in [3], the choice of parameters had its reasons: The generator is still reasonably secure for the user and at the same time it has an obvious backdoor for

whoever chose the concrete parameters. This corresponds exactly to the the goals in the document on "Bullrun" cited above. As the parameters for the concrete generator specified by NIST were officially computed by the NSA, one therefore cannot but assume that the NSA has a backdoor to this generator.

How could the NSA accomplish the masterpiece that a scheme with an obvious backdoor, which is in addition slow and has a security weakness independently of the backdoor, is built into a wildly used cryptographic software library as default option?

Obviously, no final answer is possible here, but there is some interesting information. According to Reuters, documents revealed Edward Snowden show that RSA Security received 10 million dollars by the NSA to make the standard the default method in the software library ([23]). RSA Security responded by stating that it "categorically" denies having "entered into a 'secret contract' with the NSA to incorporate a known flawed random number generator into its BSAFE encryption libraries". The company points out that it did indeed work with the NSA, which had "a trusted role in the community-wide effort to strengthen, not weaken, encryption" ([29]).

The NSA is indeed not a monolithic organization aimed at information gathering but also has a defensive arm, the Information Assurance Directorate. There is thus an internal conflict in the organization itself. This conflict was also addressed by a "Review Group on Intelligence and Communications Technologies" which was appointed by the American president Barack Obama after the disclosures by Edward Snowden ([6]). The committee asserted that the "NSA now has multiple missions and mandates, some of which are blurred, inherently conflicting, or both" and recommended to split off the Information Assurance Directorate from the NSA. However, not only was this recommendation not put into practice, but somewhat ironically the NSA announced in February 2016 a reorganization, called NSA21, which goes in the opposite direction: The Information Assurance and the Signals Intelligence directorates shall be merged in a single directorate called Operations with employees working officially "on both sides" ([25]). It is not hard to predict that this merger will not help in regaining confidence by the industry in advice from the NSA.

**4.4.2. Cryptanalysis.** Which "classic" cryptanalysis the NSA conducts exactly is not made clear by the documents. It would be a breakdown of security without comparison if an external staffer like Edward Snowden could get hold of such information. One can however get a general impression of applications of cryptanalysis and the general structure of systems.

*Virtual private networks (VPNs)* are cryptographically secured connections over the internet, which are for example used for external access to company nets. Often hereby the so-called *IPsec protocol* is used. According to the documents, the NSA has built a system for the analysis of data of such connections. Even if it is unknown how the system operates, in the article [1] the following is argued credibly:

Before the publication of the article, most connections could be started with a

Diffie-Hellman key exchange modulo a prime $p$ of 1024 bits. Now, already in the year 2005 researchers argued that it should be possible to build at a cost of one billion Euros a machine which can compute one instance of this size of the classic discrete logarithm in one year ([8]).[2]

By itself such a machine would be rather pointless for a secret service; after all, one not only wants to break one key exchange per year. Yet the problem to solve many discrete logarithms is in practice often much easier than one might think: According to information in [1], about two thirds of all key exchanges in VPNs using IPsec are conducted with respect to the same prime $p$. This means that one has to compute over and over again discrete logarithms modulo a single prime. Even if it is costly to start the computation of discrete logarithms modulo a fixed prime, after the initial phase it is surprisingly easy to compute arbitrarily many discrete logarithms.

Due to the falling of prices for computing power, the NSA could have built at the costs of some hundred million Euros a system with which it can surveil these VPNs and also the traffic with about one fifth of the most popular `https`-internet sites. This speculation is consistent with budget items for the NSA.

As this thought is now public, maybe exactly the case occurred against which it was warned in the presentation of the GCHQ mentioned in the beginning and the possibilities have been diminished already.

**4.4.3. What one should not forget.** Since the disclosures by Edward Snowden, the NSA was very present in the media. In the crypto-scene there is a long tradition to call the top hacker "NSA". Also in this section capabilities of the NSA were discussed.

One should however not forget that besides the United States and its secret service partners Canada, United Kingdom, Australia and New Zealand, there are other countries with considerable means. And the five mentioned countries are after all states of the law, in which secret services face clear cut constraints, and lively democracies.

**4.5. Specialization as a security threat.** In all fields of science, the scientific progress leads to the necessity of more and more knowledge to comprehend or even to obtain a basic understanding of new works. There is therefore a general trend towards specialization. The formation of computer science and then of cryptology as a subdicipline of computer science are aspects of this trend, a trend which now continues inside cryptology.

There are three aspects of the general trend towards specialization: First, works build on previous works; even if not directly results are used, a certain familiarity with definitions and techniques is necessary to read a work. This necessary background knowledge is constantly increasing. Second, works get more difficult to read even if one is familiar with the area. Third, not only are constantly new

[2] In [8] the factorization problem is treated. The discrete logarithm problem can be attacked with similar machines.

works added to the literature but the number of works published each year is also increasing.

Nowadays, even for experts it is difficult to judge new works. To read a work without preparation, generally speaking it has to be very near to the personal research and even then one probably has to face several days of work. Sometimes, months or even a year of preparations are necessary before a work can be read and understood in detail.

This description fits to many areas of science, but for cryptography the implications are different than for pure mathematics, for example, because of the very goals of cryptography.

Why did the obvious backdoor in the Dual_EC_DRBG and in implementations like the one by RSA Securities not already cause a stir before the disclosures by Edward Snowden? Looking back, one notices that already in 2007, at the most prominent annual cryptologic conference, CRYPTO, two employees of Microsoft drew attention to the obvious backdoor in a short informal presentation ([32]). Despite this information, nobody seemed to have bothered to check whether NIST's "Recommendation" had been implemented. The author does not have an explanation of this, except maybe that he had never even heard about the "Recommendation" and the generator, even though he works in the very area of elliptic curve cryptography, and many of his colleagues had not either.

The real-life goal of the rigorous, mathematical approach to cryptographic schemes based on solid foundations, as described in Section 3.2.5, is to have an assurance against unexpected attacks. As discussed at the end of Section 3.2.5, there is always a gap between theory and praxis, which is not easy to close. Unfortunately, the factual carrying out of the rigorous approach to cryptographic schemes goes along with further problems:

The exact statements of scientific works in cryptography are often hard to understand and to interpret even for researchers working in the area and it is even harder to check whether the alleged proofs are correct. Not only this, but it does happen not too seldom that works with alleged reductive security results with respect to seemingly strong attack scenarios do not hold what they seem to promise. Sometimes, the exact contributions are misrepresented in the introduction; sometimes, the allegedly rigorous definitions on which the analysis relies are in fact unclear or not appropriate for the situation to be studied; sometimes, to obtain a result for a particular attack scenario a scheme is designed which is then weak with respect to a straightforward attack outside of the attack scenario; sometimes, the underlying computational problem seems to be a contrived and artificial one which was only invented to obtain some kind of result; and last but not least, sometimes the alleged proofs are plainly wrong. Often such a problem has been discovered only after a scheme has been attacked successfully.

The fact that a number of protocols which were advertised as being "provably secure" (which actually means that some reductive security result had allegedly been established) have had unexpected real-life security holes has been critically assessed by Neal Koblitz, the coinventor of elliptic curve cryptography, and Alfred Menezes in an unusual article with the title "Another look at provable security" in

the Journal of Cryptology and further articles with similar titles ([21]). According to Koblitz, "As with many other over-hyped ideas – fallout shelters in the 1950s, missile shields in the 1980s – 'proofs' of the security of a cryptographic protocol often give a false confidence that blinds people to the true dangers" ([20]).

The critique by Koblitz and Menezes has led to a fierce dispute in which the contrary position was particularly advocated by Oded Goldreich, the author of the *Foundations of Cryptography* ([15]). In his opinion, "Misconceptions about the meaning of results in cryptography are unfortunately all too common. But Koblitz and Menezes, besides pointing out some already known flaws in published purported proofs, only added to the confusion with an article which is full of such misconceptions as well." Flaws, misconceptions and misunderstandings would in any case only highlight the importance of a scientific approach to cryptography, an approach which is based on rigorous terminology and analyses ([16]).

Without further addressing the assessments by Koblitz and Menezes as well as by Goldreich, the author wants to emphasize that it is THE SCIENTIFIC PROGRESS ITSELF which goes along with increased specialization which in turn goes along with the careful reading and thinking through of many important works by a very small number of people. As humans have only limited time and limited intellectual capabilities and do err, there is no easy remedy against the security threat of misleading and wrong statements and publications or misconceptions by readers.

**4.6. The picture at large.** After existing in the penumbra and the influence of strongmen for centuries, cryptology has now stepped into the public.

There is an active research community with results in the public domain, there are established scientific principles, a never-ending stream of results, ideas for new applications as well as technical progress which makes new applications possible. Cryptographic schemes like crypto currencies or crypto contracts could have large consequences on the economy or even society as a whole.

But as in the past centuries so today there is the question if the used schemes and products in their daily use are really secure.

Like for other aspects of modern life, the layman has to rely here on specialists, who themselves only have a limited knowledge, can make mistakes or have other interests as the ones pretended. How can at least a partial remedy be found?

Well, a single specialist might make misleading statements or be misunderstood by a layman. Wrong statements by single persons are however rather irrelevant if there is a process in which worse ideas are refuted and better ideas can succeed.

Like for other areas it is also valid for cryptology: It is the right social institutions on which progress is built. Values like integrity, self critique, openness towards the new, a conduct based on transparency, factuality, cooperation and competition as well as the right formal institutions with clear goals free from conflicts of interest lead in a continuous improvement process to good ideas and products.

**Acknowledgments.** I thank Marianne Diem, Werner Diem, Oded Goldreich, Neal Koblitz, Wolfgang König, Alfred Menezes, Eric Noeth and Sebastian Sterl for comments, discussions and help.

## References

[1] David Adrian, Karthikeyan Bhargavan, Zakir Durumeric, Pierrick Gaudry, Matthew Green, J. Alex Halderman, Nadia Heninger, Drew Springall, Emmanuel Thomé, Luke Valenta, Benjamin VanderSloot, Eric Wustrow, Santiago Zanella-Béguelin and Paul Zimmermann, Imperfect Forward Secrecy: How Diffie-Hellman Fails in Practice, In *Procedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, 2015.

[2] Daniel Bernstein and Tanja Lange, Faster Addition and Doubling on Elliptic curves. In *Advances in Cryptology – ASIACRYPT 2007*, Springer, Berlin, 2007, 29–50.

[3] Daniel Bernstein, Tanja Lange and Ruben Niederhagen, Dual EC: A Standardized Back Door. Cryptology ePrint Archive: Report 2015/767, 2015.

[4] Bundesamt für Sicherheit in der Informationstechnik, BSI Technische Richtlinie TR-02102-1, Kryptographische Verfahren: Empfehlungen und Schlüssellängen, 2015.

[5] Anne Canteaut, Céderic Lauradoux and André Seznec, Understanding cache attacks. INRIA Rapport de recherche No. 5881, 2006.

[6] Richard Clarke, Michael Morell, Geoffrey Stone, Cass Sunstein and Peter Swire, Liberty and Security in a Changing World, Report and Recommendations of The President's Review Group on Intelligence and Communications Technologies. The White House, 2013.

[7] Whitfield Diffie and Martin Hellman, New Directions in Cryptography. *IEEE Transactions on Information Theory* **2** (1976), 644 – 654.

[8] Jens Franke, Thorsten Kleinjung, Christof Paar, Jan Pelzl, Christine Priplata and Colin Stahlke, SHARK. Presented at SHARCS - Special-purpose Hardware for Attacking Cryptographic Systems 2005. `http://www.sharcs.org`

[9] Steven Galbraith, Mathematics of Public Key Cryptography. Cambridge University Press, Cambridge, UK, 2012.

[10] Joachim von zur Gathen, CryptoSchool. Springer, Berlin, 2015

[11] Carl Friedrich Gauß, Disquisitiones Arithmeticae. Gerhard Fleischer, Leipzig, 1801.

[12] GCHQ, BULLRUN. Internal presentation. `http://www.spiegel.de/media/media-35532.pdf`

[13] Kristian Gjøsteen, Comments on the Dual-EC-DRBG / NIST SP 800-900, 2005. `http://www.math.ntnu.no/~kristiag/drafts/dual-ec-drbg-comments.pdf`

[14] Oded Goldreich, Foundations of Cryptography I and II. Cambridge University Press, Cambridge, UK, 2001 and 2004.

[15] Oded Goldreich, On Post-Modern Cryptography. Cryptology ePrint Archive: Report 2006/461, 2006.

[16] Oded Goldreich, Conversation with the author, 2016.

[17] Godfrey Harold Hardy, A Mathematician's Apology. Cambridge University Press, Cambridge, UK, 1940.

[18] David Kahn, The Codebreakers – The Comprehensive History of Secret Communication from Ancient Times to the Internet. Scribner, New York, 1996.

[19] Auguste Kerckhoffs, La cryptographie militaire. *Journal des sciences militaires* **9** (1883), 5–38 & 161–191.

[20] Neal Koblitz, The uneasy relationship between mathematics and cryptography. *Notices of the AMS* **54** (2007), 972 – 979.

[21] Neal Koblitz and Alfred Menezes, Another Look at Provable Security. *Journal of Cryptology* **20** (2007), 3 – 37.

[22] Maurice Kraitchik, Théorie des Nombres, Gauthier-Villars, Paris, 1922.

[23] Joseph Menn, Exclusive: Secret contract tied NSA and security industry pioneer. Reuters, Dec. 20 2013.

[24] Mohammed Mrayati, Y. Meer Alam and M. H. Tayyan, Series on Arabic Origins of Cryptology 1 – 3. KFCRIS & KACST, Riyadh, 2002 – 2003

[25] Ellen Nakashima, National Security Agency plans major reorganization. Washington Post, Feb. 2 2016.

[26] National Security Agency, Suite B Cryptography, Aug. 19 2015. `https://www.nsa.gov/ia/programs/suiteb_cryptography/index.shtml`

[27] New York Times, Secret Documents Reveal N.S.A. Campaign Against Encryption, Sep. 5 2013.

[28] Ron Rivest, Adi Shamir, Leonard Adleman, A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Communications of the ACM* **21** (1987), 120–126.

[29] RSA Security, RSA response to media claims regarding NSA relationship, 2013. `https://blogs.rsa.com/rsa-response`

[30] Klaus Schmeh, Codeknacker und Codemacher, W3L, Bochum, 2014.

[31] Claude Shannon, Communication Theory of Secrecy Systems. Bell System Technical Journal **28** (1948), 656 – 715.

[32] Dan Shumow and Niels Ferguson, On the Possibility of a Back Door in the NIST SP800-90 Dual EC Prng. CRYPTO Rump Session, 2007. `http://rump2007.cr.yp.to/15-shumow.pdf`

[33] Peter Shor, Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. *SIAM Journal on Computing* **26** (1997), 1484–1509.

[34] Maurice Wilkes, Time-Sharing Computer Systems. American Elsevier, 1968.

Claus Diem, Mathematical Institute, University of Leipzig, Leipzig, Germany
E-mail: diem@math.uni-leipzig.de