# CrypTool Script*
# Mathematics and Cryptography

(c) The authors, 1998-2003
Frankfurt am Main

April 22, 2003

In this CrypTool script you will find predominantly mathematically oriented information on using cryptographic procedures. The main chapters have been written by various authors and are therefore independent from one another. At the end of most chapters you will find literature and web links.

You will obtain information about the principles of symmetrical and asymmetrical **encryption**. A large section of this script is dedicated to the fascinating topic of **prime numbers**. Using numerous examples, the **elementary number theory** and **modular arithmetic** are introduced and applied in an exemplary manner for the **RSA procedure**. By reading the following chapter you'll gain an insight into the mathematical ideas behind **modern cryptography**.

A further chapter is devoted to **digital signatures**, which are an essential component of e-business applications. The last chapter describes **elliptic curves**: they could be used as an alternative to RSA and in addition are extremely well suited for implementation on smartcards.

Whereas the *program* CrypTool teaches you how to use cryptography in practice, the *script* provides those interested in the subject with a deeper understanding of the mathematical algorithms used – trying to do it in an instructive way.

The authors Bernhard Esslinger, Matthias Büger, Bartol Filipovic, Henrik Koy, Roger Oyono and Jörg-Cornelius Schneider would like to take this opportunity to thank their colleagues in the company and at the universities of Frankfurt, Gießen, Siegen, Karlsruhe and Darmstadt. They are particularly indepted to Dr. Peer Wichmann from the Karlsruhe computer science research centre (Forschungszentrum Informatik, FZI) for his down-to-earth support.

As at CrypTool, the quality of the script is enhanced by your suggestions and ideas for improvement. We look forward to your feedback.

You will find the current version of CrypTool under
http://www.cryptool.org, http://www.cryptool.com or http://www.cryptool.de.

The contact people for this free open-source tool are listed in the "readme" file delivered within the CrypTool package.

*Background reading for the CrypTool program

# Contents

## Preface to the 6th Edition of the CrypTool Script

Starting in the year 2000 this script became a part of the CrypTool package. It is designed to accompany the program CrypTool by explaining some mathematical topics in more detail, but still in a way which is easy to understand.

In order to also enable developers/authors to work together independently the topics have been split up and for each topic an extra chapter has been written which can be read on its own. The later editorial work in TeX added cross linkages between different sections and footnotes describing where you can find the according functions within the CrypTool program (see menu tree in appendix A). Naturally there are much more interesting topics in mathematics and cryptography which could be discussed in greater depth – therefore this is only one of many ways to do it.

Rapid spread of the Internet has also lead to intensified research in the technologies involved, especially within the area of cryptography where a good deal of new knowledge has arisen.

This edition of the script mainly updates the summaries of the following topical research areas:

- the search for the largest prime numbers (generalized Mersenne and Fermat primes) (chap. 2.6),
- progress in number theory ("Primes in P") (chap. 3.11.3),
- the factorisation of big numbers (TWIRL) (chap. 3.11.3),
- progress in cryptanalysis of the AES standard (chap. 1.1.1) and
- progress in brute-force attacks against symmetric algorithms (chap. 1.1.2).

Chapter 6 of this script has also been enhanced: now we explain in greater detail how elliptic curves are defined over different number fields.

The first time the document was delivered with CrypTool was in version 1.2.01. Since then it has been expanded and revised in every new version of CrypTool (1.2.02, 1.3.00, 1.3.02, 1.3.03 and now 1.3.04).

I'd be more than happy if this also continues in the further open-source versions of CrypTool (they will be delivered by the University of Darmstadt).

I am deeply grateful to all the people helping with their impressive commitment who have made this global project so successful. Especially I would like to acknowledge the English language proof-reading of this script version done by Richard Christenson and L. Montgomery.

I hope that many readers have fun with this script and that they get out of it more interest and greater understanding of this modern but also very ancient topic.

Bernhard Esslinger

Frankfurt, March 2003

# Introduction – How do the Script and the Program Play together?

### This script

This document is delivered together with the program CrypTool.

Because the articles in this script are largely self-contained, this text can also be read independently of CrypTool.

Chapters 4 and 6 might require a deeper knowledge in mathematics, while the other chapters should be understandable with a school leaving certificate.

The authors have attempted to describe cryptography for a broad audience – without being mathematically incorrect. We believe that this didactical pretension is the best way to promote the awareness for IT security and the readiness to use standardised modern cryptography.

### The program CrypTool

CrypTool is a program with an extremely comprehensive online help enabling you to use and analyse cryptographic procedures within a unified graphical user interface.

CrypTool was developed during the end-user awareness program at Deutsche Bank in order to increase employee awareness of IT security and provide them with a deeper understanding of the term security.

A further aim has been to enable users to understand the cryptographic procedures. In this way, using CrypTool as a reliable reference implementation of the various encryption procedures (because of using the industry-proven Secude Library), you can test the encryption implemented in other programs.

CrypTool is currently been used for training in companies and teaching at school and universities, and moreover several universities are helping to further develop the project.


### Acknowledgment

At this point I'd like to thank explicitly 3 people who especially contributed to CrypTool. Without their talents and engagement CrypTool would not be what it is today:

- Mr. Henrik Koy
- Mr. Jörg-Cornelius Schneider and
- Dr. Peer Wichmann.

Also I want to thank all the many people not mentioned here for their hard work (mostly carried out in their spare time).


Bernhard Esslinger

# 1 Encryption Procedures

(Bernhard Esslinger, besslinger@web.de, May 1999, Updates Dec. 2001, Feb. 2003)

This chapter introduces the topic in a more descriptive way without using too much mathematics.

The purpose of encryption is to change data in such a way that only an authorised recipient is able to reconstruct the plaintext. This allows us to transmit data without worrying about it getting into unauthorised hands. Authorised recipients possess a piece of secret information – called the key – which allows them to decrypt the data while it remains hidden from everyone else.

One encryption procedure has been mathematically proved to be secure, the *One Time Pad*. However, this procedure has several practical disadvantages (the key used must be randomly selected and must be at least as long as the message being protected), which means that it is hardly used except in closed environments such as for the hot wire between Moscow and Washington.

For all other procedures there is a (theoretical) possibility of breaking them. If the procedures are good, however, the time taken to break them is so long that it is practically impossible to do so, and these procedures can therefore be considered (practically) secure.

The book of Bruce Schneier [Schneier1996] offers a very good overview of the different algorithms. We basically distinguish between symmetric and asymmetric encryption procedures.

## 1.1 Symmetric encryption[1]

For *symmetric* encryption sender and recipient must be in possession of a common (secret) key which they have exchanged before actually starting to communicate. The sender uses this key to encrypt the message and the recipient uses it to decrypt it.

All classical methods are of this type. Examples can be found within CrypTool or in [Nichols1996]. Now we want to consider more modern mechanisms.

The advantages of symmetric algorithms are the high speed with which data can be encrypted and decrypted. One disadvantage is the need for key management. In order to communicate with one another confidentially, sender and recipient must have exchanged a key using a secure channel before actually starting to communicate. Spontaneous communication between individuals who have never met therefore seems virtually impossible. If everyone wants to communicate with everyone else spontaneously at any time in a network of $n$ subscribers, each subscriber must have previously exchanged a key with each of the other $n-1$ subscribers. A total of $n(n-1)/2$ keys must therefore be exchanged.

The most well-known symmetric encryption procedure is the DES-algorithm. The DES-algorithm

[1] With CrypTool v1.3 you can execute the following modern symmetric encryption algorithms (using the menu **Crypt \ Symmetric**):
IDEA, RC2, RC4, DES (ECB), DES(CBC), Triple-DES(ECB), Triple-DES(CBC), MARS (AES candidate), RC6 (AES candidate), Serpent (AES candidate), Twofish (AES candidate), Rijndael (official AES algorithm).

has been developed by IBM in collaboration with the National Security Agency (NSA), and was published as a standard in 1975. Despite the fact that the procedure is relatively old, no effective attack on it has yet been detected. The most effective way of attacking consists of testing (almost) all possible keys until the right one is found (*brute-force-attack*). Due to the relatively short key length of effectively 56 bits (64 bits, which however include 8 parity bits), numerous messages encrypted using DES have in the past been broken. Therefore, the procedure can now only be considered to be conditionally secure. Symmetric alternatives to the DES procedure include the IDEA or Triple DES algorithms.

Up-to-the-minute procedure is the symmetric AES standard. The associated Rijndael algorithm was declared winner of the AES award on 2 October 2000 and thus succeedes the DES procedure.

More details about the AES algorithms can be found within the online help of CrypTool[2].

### 1.1.1 New results about cryptanalysis of AES

Below you will find some results, which have recently called into question the security of the AES algorithm – from our point of view these doubts practically still remain unfounded . The following information is based on the original papers and the articles [Wobst-iX2002] and [Lucks-DuD2002].

AES with a minimum key length of 128 bit is still in the long run sufficiently secure against brute-force attacks – as long as the quantum computers aren't powerful enough. When announced as new standard AES was immune against all known crypto attacks, mostly based on statistical considerations and earlier applied to DES: using pairs of clear and cipher texts expressions are constructed, which are not completely at random, so they allow conclusions to the used keys. These attacks required unrealistically large amounts of intercepted data.

Cryptanalysts already label methods as "academic success" or as "cryptanalytic attack" if they are theoretically faster than the complete testing of all keys (brute force analysis). In the case of AES with the maximal key length (256 bit) exhaustive key search on average needs $2^{255}$ encryption operations. A cryptanalytic attack needs to be better than this. At present between $2^{75}$ and $2^{90}$ encryption operations are estimated to be performable only just for organizations, for example a security agency.

In their 2001-paper Ferguson, Schroeppel and Whiting [Ferguson2001] presented a new method of symmetric codes cryptanalysis: They described AES with a closed formula (in the form of a continued fraction) which was possible because of the "relatively" clear structure of AES. This formula consists of around 1000 trillion terms of a sum - so it does not help concrete practical cryptanalysis. Nevertheless curiosity in the academic community was awakened. It was already known, that the 128-bit AES could be described as an over-determined system of about 8000 quadratic equations (over an algebraic number field) with about 1600 variables (some of them are the bits of the wanted key) – equation systems of that size are in practice not solvable. This special equation system is relatively sparse, so only very few of the quadratic terms (there are about 1,280,000 are possible quadratic terms in total) appear in the equation system.

[2] CrypTool online help: the index head-word **AES** leads to the 3 help pages: **AES candidates**, **The AES winner Rijndael** and **The Rijndael encryption algorithm**.

The mathematicians Courtois and Pieprzyk [Courtois2002] published a paper in 2002, which got a great deal of attention amongst the crypto community: The pair had further developed the XL-method (eXtended Linearization), introduced at Eurocrypt 2000 by Shamir et al., to create the so called XSL-method (eXtended Sparse Linearization). The XL-method is a heuristic technique, which in some cases manages to solve big non-linear equation systems and which was till then used to analyze an asymmetric algorithm (HFE). The innovation of Courtois and Pieprzyk was, to apply the XL-method on symmetric codes: the XSL-method can be applied to very specific equation systems. A 256-bit AES could be attacked in roughly $2^{230}$ steps. This is still a purely academic attack, but also a direction pointer for a complete class of block ciphers. The major problem with this attack is that until now nobody has worked out, under what conditions it is successful: the authors specify in their paper necessary conditions, but it is not known, which conditions are sufficient. There are two very new aspects of this attack: firstly this attack is not based on statistics but on algebra. So attacks seem to be possible, where only very small amounts of cipher text are available. Secondly the security of a product-algorithm does not exponentially increase with the number of rounds.

Currently there is a large amount of research in this area: for example Murphy and Robshaw presented a paper at Crypto 2002 [Robshaw2002a], which could dramatically improve cryptanalysis: the burden for a 128-bit key was estimated at about $2^{100}$ steps by describing AES as a special case of an algorithm called BES (Big Encryption System), which has an especially "round" structure. But even $2^{100}$ steps are beyond what is achievable in the foreseeable future. Using a 256 bit key the authors estimate that a XSL-attack will require $2^{200}$ operations.

More details can be found at:

`http://www.cryptosystem.net/aes`
`http://www.minrank.org/aes/`

So for 256-AES the attack is much more effective than brute-force but still far more away from any computing power which could be accessible in the short-to-long term.

The discussion is very controversial at the moment: Don Coppersmith (one of the inventors of DES) for example queries the practicability of the attack because XLS would provide no solution for AES [Coppersmith2002]. This implies that then the optimization of Murphy and Robshaw [Robshaw2002b] would not work.

### 1.1.2 Current status of brute-force attacks on symmetric algorithms (RC5)

The current status of brute-force attacks on symmetric encryption algorithms can be explained with the block cipher RC5.

Brute-force (exhaustive search, trial-and-error) means to completely examine all keys of the key space: so no special analysis methods have to be used. Instead, the cipher text is decrypted with all possible keys and for each resulting text it is checked, whether this is a meaningful clear text. A key length of 64 bit means at most $2^{64} = 18{,}446{,}744{,}073{,}709{,}551{,}616$ or about 18 trillion (GB) / 18 quintillion (US) keys to check[3].

[3] With CrypTool v1.3 you can also try brute-force attacks of modern symmetric algorithms (using the menu **Analysis**

Companies like RSA Security provide so-called cipher challenges in order to quantify the security offered by well-known symmetric ciphers as DES, Triple-DES or RC5[4]. They offer prizes for those who manage to decipher cipher texts, encrypted with different algorithms and different key lengths, and to unveil the symmetric key (under controlled conditions). So theoretical estimates can be confirmed.

It is well-known, that the "old" standard algorithm DES with a fixed key length of 56 bit is no more secure: this was demonstrated already in January 1999 by the Electronic Frontier Foundation (EFF). With their specialized computer Deep Crack they cracked a DES encrypted message within less than a day[5].

The current record for strong symmetric algorithms unveiled a key 64 bit long. The algorithm used was RC5, a block cipher with variable key size.

The RC5-64 challenge has been solved by the distributed.net team after 5 years[6]. In total 331,252 individuals co-operated over the internet to find the key. More than 15 trillion (GB) / 15 quintillion (US) keys were checked, until they found the right key.

This makes clear, that symmetric algorithms (even if they have no cryptographical weakness) using keys of size 64 bit are no more appropriate to keep sensible data private.

Similar cipher challenges are there for asymmetric algorithms (please see chapter 3.11.4).

## 1.2 Asymmetric encryption[7]

In the case of *asymmetric* encryption each subscriber has a personal pair of keys consisting of a *secret* key and a *public* key. The public key, as its name implies, is made public, e.g. in a key directory on the Internet.

If Alice[8] wants to communicate with Bob, then she finds Bob's public key in the directory and uses it to encrypt her message to him. She then sends this cipher text to Bob, who is then able to decrypt it again using his secret key. As only Bob knows his secret key, only he can decrypt messages addressed to him. Even Alice who sends the message cannot restore plaintext from the (encrypted) message she has sent. Of course, you must first ensure that the public key cannot be used to derive the private key.

Such a procedure can be demonstrated using a series of thief-proof letter boxes. If I have composed a message, I then look for the letter box of the recipient and post the letter through it. After that, I can no longer read or change the message myself, because only the legitimate recipient has the key for the letter box.

\ **Ciphertext-Only**): to achieve a result in an appropriate time with a single PC you should mark not more than 20 bit of the key as unknown.

[4] http://www.rsasecurity.com/rsalabs/challenges/secretkey/index.html

[5] http://www.rsasecurity.com/rsalabs/challenges/des3/index.html

[6] http://distributed.net/pressroom/news-20020926.html

[7] With CrypTool v1.3 you can execute RSA encryption and decryption (using the menu **Crypt \ Asymmetric**).

[8] In order to describe cryptographic protocols participants are often named Alice, Bob, ... (see [Schneier1996, p. 23]). Alice and Bob perform all 2-person-protocols. Alice will initiate all protocols and Bob answers. The attackers are named Eve (eavesdropper) and Mallory (malicious active attacker).

The advantage of asymmetric procedures is the easy key management. Let's look again at a network with $n$ subscribers. In order to ensure that each subscriber can establish an encrypted connection to each other subscriber, each subscriber must possess a pair of keys. We therefore need $2n$ keys or $n$ pairs of keys. Furthermore, no secure channel is needed before messages are transmitted, because all the information required in order to communicate confidentially can be sent openly. In this case, you simply have to pay attention to the accuracy (integrity and authenticity) of the public key. Disadvantage: Pure asymmetric procedures take a lot longer to perform than symmetric ones.

The most well-known asymmetric procedure is the RSA algorithm[9], named after its developers Ronald Rivest, Adi Shamir and Leonard Adleman. The RSA algorithm was published in 1978. The concept of asymmetric encryption was first introduced by Whitfield Diffie and Martin Hellman in 1976. Today, the ElGamal procedures also play a decisive role, particularly the Schnorr variant in the DSA (Digital Signature Algorithm).

## 1.3 Hybrid procedures[10]

In order to benefit from the advantages of symmetric and asymmetric techniques together, hybrid procedures are usually used (for encryption) in practice.

In this case the data is encrypted using symmetric procedures: the key is a session key generated by the sender randomly[11] that is only used for this message. This session key is then encrypted using the asymmetric procedure and transmitted to the recipient together with the message. Recipients can determine the session key using their secret keys and then use the session key to encrypt the message. In this way, we can benefit from the easy key management of asymmetric procedures and encrypt large quantities of data quickly and efficiently using symmetric procedures.

## 1.4 Further details

Beside the information you can find in the following chapters, many other books and on a good number of websites the online help of CrypTool also offers very many details about the symmetric and asymmetric encryption methods.

---

[9] The RSA algorithm is extensively described in chapter 3.10 and later within this script. The RSA cryptosystem can be executed in many variations with CrypTool v1.3 (using the menu **Individual Procedures \ RSA Demonstration \ RSA Cryptosystem**). The topical research results concerning RSA are described in chapter 3.11.

[10] Within CrypTool v1.3 you can get a visualization of this technique using the menu **Crypt \ Hybrid Demonstration**: this dialogue shows the single steps and its dependencies with concrete numbers.

[11] An important part of cryptographically secure techniques is to generate random numbers. Within CrypTool v1.3 you can check out different random number generators using the menu **Indiv. Procedures \ Generate Random Numbers**. Using the menu **Analysis \ Random Tests** you can apply different test methods for random data to binary documents.
Up to now CrypTool has concentrated on cryptographically strong pseudo random number generators. Only the integrated Secude generator involves a "pure" random source.

# References


[Nichols1996] Randall K. Nichols,
*Classical Cryptography Course, Volume 1 and 2*,
Aegean Park Press 1996.

[Schmeh2003] Klaus Schmeh,
*Cryptography and Public Key Infrastructures on the Internet*,
John Wiley & Sons Ltd., Chichester 2003.
A considerable, up-to-date, easy to read book, which also considers practical problems such as standardisation or real existing software.

[Schneier1996] Bruce Schneier,
*Applied Cryptography, Protocols, Algorithms, and Source Code in C*,
Wiley 1994, 2nd edition 1996.

[Coppersmith2002] Don Coppersmith,
*Re: Impact of Courtois and Pieprzyk results*,
2002-09-19, "AES Discussion Groups" at
http://aes.nist.gov/aes/

[Courtois2002] Nicolas Courtois, Josef Pieprzyk,
*Cryptanalysis of Block Ciphers with Overdefined Systems of Equations*,
received 10 Apr 2002, last revised 9 Nov 2002.
A different version, so called compact version of the first XSL attack, was published at Asiacrypt Dec 2002.
http://eprint.iacr.org/2002/044

[Ferguson2001] Niels Ferguson, Richard Schroeppel, Doug Whiting,
*A simple algebraic representation of Rijndael*, Draft 2001/05/1,
http://www.xs4all.nl/~vorpal/pubs/rdalgeq.html

[Lucks-DuD2002] Stefan Lucks, Rüdiger Weis,
*Neue Ergebnisse zur Sicherheit des Verschlüsselungsstandards AES*, in DuD Dec. 2002.

[Robshaw2002a] S.P. Murphy, M.J.B. Robshaw,
*Essential Algebraic Structure within the AES*, June 5, 2002, Crypto 2002,
http://www.isg.rhul.ac.uk/ mrobshaw/rijndael/rijndael.html

[Robshaw2002b] S.P. Murphy, M.J.B. Robshaw,
*Comments on the Security of the AES and the XSL Technique*, September 26, 2002,
http://www.isg.rhul.ac.uk/ mrobshaw/rijndael/rijndael.html

[Wobst-iX2002] Reinhard Wobst,
*Angekratzt - Kryptoanalyse von AES schreitet voran*, in iX Dec. 2002,
plus the reader's remark by Johannes Merkle in iX Feb. 2003.

## Web links

1. AES or Rijndael Cryptosystem
   http://www.cryptosystem.net/aes
   http://www.minrank.org/aes/

2. AES Discussion Groups at NIST
   http://aes.nist.gov/aes

3. distributed.net: "RC5-64 has been solved"
   http://distributed.net/pressroom/news-20020926.html

4. RSA: "The RSA Secret Key Challenge"
   http://www.rsasecurity.com/rsalabs/challenges/secretkey/index.html

5. RSA: "DES Challenge"
   http://www.rsasecurity.com/rsalabs/challenges/des3/index.html

6. Further Links can be found at the CrypTool Homepage
   http://www.cryptool.org

# 2 Prime Numbers

(Bernhard Esslinger, besslinger@web.de, May 1999, Updates Nov. 2000, Dec. 2001, Feb. 2003)

> *Albert Einstein*[12]*:*
> Progress requires exchange of knowledge.

## 2.1 What are prime numbers?

Prime numbers are whole, positive numbers greater than or equal to 2 that can only be divided by 1 and themselves. All other natural numbers greater than or equal to 2 can be formed by multiplying prime numbers.

The *natural* numbers $\mathbb{N} = \{1, 2, 3, 4, \cdots\}$ thus comprise

- the number 1 (the unit value)
- the primes and
- the composite numbers.

Prime numbers are particularly important for 3 reasons:

- In number theory, they are considered to be the basic components of natural numbers, upon which numerous brilliant mathematical ideas are based.
- They are of extreme practical importance in modern cryptography (public key cryptography). The most common public key procedure, invented at the end of the 1970's, is RSA encryption. Only using (large) prime numbers for particular parameters can you guarantee that an algorithm is secure, both for the RSA procedure and for even more modern procedures (digital signature, elliptic curves).
- The search for the largest known prime numbers does not have any practical usage known to date, but requires the best computers, is an excellent benchmark (possibility for determining the performance of computers) and leads to new calculation methods on many computers (see also: http://www.mersenne.org/prime.htm).

Many people have been fascinated by prime numbers over the past two millennia. Ambition to make new discoveries about prime numbers has often resulted in brilliant ideas and conclusions. The following section provides an easily comprehensible introduction to the basics of prime numbers. We will also explain what is known about the distribution (density, number of prime numbers in particular intervals) of prime numbers and how prime number tests work.

[12] German physicist and Nobel Prize winner, March 14, 1879 – April 14, 1955

## 2.2 Prime numbers in mathematics

Every whole number has a factor. The number 1 only has one factor, itself, whereas the number 12 has the six factors $1, 2, 3, 4, 6, 12$. Many numbers can only be divided by themselves and by 1. With respect to multiplication, these are the "atoms" in the area of numbers. Such numbers are called prime numbers.

In mathematics, a slightly different (but equivalent) definition is used.

**Definition 2.1.** *A whole number $p \in \mathbf{N}$ is called prime if $p > 1$ and $p$ only possesses the trivial factors $\pm 1$ and $\pm p$.*

By definition, the number 1 is not a prime number. In the following sections, $p$ will always denote a prime number.

The sequence of prime numbers starts with

2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97, $\cdots$ .

The first 100 numbers include precisely 25 prime numbers. After this, the percentage of primes constantly decreases. Prime numbers can be factorised in a uniquely *trivial* way:

$$5 = 1 \cdot 5, \quad 17 = 1 \cdot 17, \quad 1,013 = 1 \cdot 1,013, \quad 1,296,409 = 1 \cdot 1,296,409.$$

All numbers that have 2 or more factors not equal 1 are called *composite* numbers. These include

$$4 = 2 \cdot 2, \quad 6 = 2 \cdot 3$$

as well as numbers that *look like primes*, but are in fact composite:

$$91 = 7 \cdot 13, \quad 161 = 7 \cdot 23, \quad 767 = 13 \cdot 59.$$

**Theorem 2.1.** *Each whole number $m$ greater than 1 possesses a lowest factor greater than 1. This is a prime number $p$. Unless $m$ is a prime number itself, then: $p$ is less than or equal to the square root of $m$.*

All whole numbers greater than 1 can be expressed as a product of prime numbers — in a unique way. This is the claim of the 1st fundamental theorem of number theory (= fundamental theorem of arithmetic = fundamental building block of all positive integers).

**Theorem 2.2.** *Each element $n$ of the natural numbers greater than 1 can be written as the product $n = p_1 \cdot p_2 \dots p_m$ of prime numbers. If two such factorisations*

$$n = p_1 \cdot p_2 \cdot \dots \cdot p_m = p'_1 \cdot p'_2 \cdots p'_{m'}$$

*are given, then they can be reordered such that $m = m'$ and for all $i$: $p_i = p'_i$.*
*($p_1, p_2, \dots, p_m$ are called the prime factors of $n$)*

In other words: each natural number other than 1 can be written as a product of prime numbers in precisely one way, if we ignore the order of the factors. The factors are therefore unique (the *expression as a product of factors* is unique)! For example,

$$60 = 2 \cdot 2 \cdot 3 \cdot 5 = 2^2 \cdot 3^1 \cdot 5^1.$$

And this — other than changing the order of the factors — is the only way in which the number 60 can be factorised. If you allow numbers other than primes as factors, there are several ways of factorising integers and the uniqueness  is lost:

$$60 = 1 \cdot 60 = 2 \cdot 30 = 4 \cdot 15 = 5 \cdot 12 = 6 \cdot 10 = 2 \cdot 3 \cdot 10 = 2 \cdot 5 \cdot 6 = 3 \cdot 4 \cdot 5 = \cdots .$$

The following section is aimed more at those familiar with mathematical logic: The 1st fundamental theorem only appears to be obvious . We can construct numerous other sets of numbers (i.e. other than positive whole numbers greater than 1), for which numbers in the set cannot be expressed uniquely as a product of the prime numbers of the set: In the set $M = \{1, 5, 10, 15, 20, \cdots\}$ there is no equivalent to the fundamental theorem under multiplication. The first five prime numbers of this sequence are $5, 10, 15, 20, 30$ (note: 10 is prime, because 5 is not a factor of 10 in this set — the result is not an element of the given basic set $M$). Because the following applies in $M$:

$$100 = 5 \cdot 20 = 10 \cdot 10$$

and $5, 10, 20$ are all prime numbers in this set, the expression as a product of prime factors is not unique here.

## 2.3 How many prime numbers are there?

For the natural numbers, the primes can be compared to elements in chemistry or the elementary particles in physics (see [Blum1999, p. 22]).

Although there are only 92 natural chemical elements, the number of prime numbers is unlimited. Even the Greek, Euclid[13] knew this in the third century B.C.

**Theorem 2.3 (Euclid[14]).** *The sequence of prime numbers does not discontinue. Therefore, the quantity of prime numbers is infinite.*

His proof that there is an infinite number of primes is still considered to be a brilliant mathematical consideration and conclusion today (proof by contradiction). He assumed that there is only a

[13] Euclid, a Greek mathematician of 4th and 3rd century B.C. He worked at the Egyptian academy of Alexandria and wrote "The Elements", the most well known systematically textbook of the Greek mathematics.

[14] The common usage of the term does not denote Euclid as the inventor of the theorem rather; the true inventor is merely not as prominent. The theorem has already been distinguished and proven in Euclid's Elements (Book IX, s. 20). The phraseology is remarkable due to the fact that the word infinite is not used. The text reads as followed

*Οί πρῶτοι ἀριθμοὶ πλείους εἰσὶ παντὸς τοῦ προτεθέντος πλήθ ους πρώτων ἀριθμῶν,*

the English translation of which is: the prime numbers are more than any previously existing amount of prime numbers.

finite number of primes and therefore a largest prime number. Based on this assumption, he drew logical conclusions until he obtained an obvious contradiction. This meant that something must be wrong. As there were no mistakes in the chain of conclusions, it could only be the assumption that was wrong. Therefore, there must be an infinite number of primes!

**Euclid's proof by contradiction** goes as follows:

**Assumption:** There is a *finite* number of primes.

**Conclusion:** Then these can be listed $p_1 < p_2 < p_3 < \cdots < p_n$, where $n$ is the (finite) number of prime numbers. $p_n$ is therefore the largest prime. Euclid now looks at the number $a = p_1 \cdot p_2 \cdots p_n + 1$. This number cannot be a prime number because it is not included in our list of primes. It must therefore be divisible by a prime, i.e. there is a natural number $i$ between 1 and $n$, such that $p_i$ divides the number $a$. Of course, $p_i$ also divides the product $a - 1 = p_1 \cdot p_2 \cdots p_n$, because $p_i$ is a factor of $a - 1$. Since $p_i$ divides the numbers $a$ and $a - 1$, it also divides the difference of these numbers. Thus: $p_i$ divides $a - (a - 1) = 1$. $p_i$ must therefore divide 1, which is impossible.

**Contradiction:** Our assumption was false.

Thus there is an *infinite* number of primes (Cross-reference: overview under 2.8.1 of the number of prime numbers in various intervals).

Here we should perhaps mention yet another fact which is initially somewhat surprising. Namely, in the prime numbers sequence $p_1, p_2, \cdots$, gaps between prime numbers can have an individually determined length $n$. It is undeniable that under the $n$ succession of natural numbers

$$(n+1)! + 2, \cdots, (n+1)! + (n+1),$$

none of them is a prime number since in order, the numbers $2, 3, \cdots, (n+1)$ are comprised respectively as real divisors. ($n!$ means the product of the first $n$ natural numbers therefore $n! = n * (n-1) * \cdots * 2 * 1$).

## 2.4 The search for extremely large primes

The largest prime numbers known today have several hundred thousand digits, which is too big for us to imagine. The number of elementary particles in the universe is "only" estimated to be a 80-digit number (See: overview under 2.8.3 of various orders of magnitude / dimensions).

### 2.4.1 Special number types – Mersenne numbers

Almost all known huge prime numbers are special candidates, called *Mersenne numbers* of the form $2^p - 1$, where $p$ is a prime. Marin Mersenne (1588-1648) was a French priest and mathematician. Not all Mersenne numbers are prime:

$$
\begin{array}{ll}
2^2 - 1 = 3 & \Rightarrow \text{prime} \\
2^3 - 1 = 7 & \Rightarrow \text{prime} \\
2^5 - 1 = 31 & \Rightarrow \text{prime} \\
2^7 - 1 = 127 & \Rightarrow \text{prime} \\
2^{11} - 1 = 2.047 = 23 \cdot 89 & \Rightarrow \text{NOT prime!}
\end{array}
$$

Even Mersenne knew that not all Mersenne numbers are prime (see exponent $p = 11$). A prime Mersenne number is called Mersenne prime number.
However, he is to be thanked for the interesting conclusion that a number of the form $2^n - 1$ is not a prime number if $n$ is a composite number:

**Theorem 2.4 (Mersenne).** *If $2^n - 1$ is a prime number, then $n$ is also a prime number.*

**Proof**
The theorem of Mersenne can be proved by contradiction. We therefore assume that there exists a composite natural number $n$ (with real factorisation) $n = n_1 \cdot n_2$, with the property that $2^n - 1$ is a prime number.

From

$$
\begin{aligned}
(x^r - 1)((x^r)^{s-1} + (x^r)^{s-2} + \cdots + x^r + 1) &= ((x^r)^s + (x^r)^{s-1} + (x^r)^{s-2} + \cdots + x^r) \\
&\quad -((x^r)^{s-1} + (x^r)^{s-2} + \cdots + x^r + 1) \\
&= (x^r)^s - 1 = x^{rs} - 1,
\end{aligned}
$$

we conclude

$$2^{n_1 n_2} - 1 = (2^{n_1} - 1)((2^{n_1})^{n_2 - 1} + (2^{n_1})^{n_2 - 2} + \cdots + 2^{n_1} + 1).$$

Because $2^n - 1$ is a prime number, one of the above two factors on the right-hand side must be equal to 1. This is the case if and only if $n_1 = 1$ or $n_2 = 1$. But this contradicts our assumption. Therefore the assumption is false. This means that there exists no composite number $n$, such that $2^n - 1$ is a prime. $\square$

Unfortunately this theorem only applies in one direction (the inverse statement does not apply, no equivalence): that means that there exist prime exponent for which the Mersenne number is **not** prime (see the above example $2^{11} - 1$, where 11 is prime, but $2^{11} - 1$ not).

Mersenne claimed that $2^{67} - 1$ is a prime number. There is also a mathematical history behind this claim: it first took over 200 years before Edouard Lucas (1842-1891) proved that this number is composite. However, he argued indirectly and did not name any of the factors. Then Frank Nelson showed in 1903 which factors make up this composite number:

$$2^{67} - 1 = 147,573,952,589,676,412,927 = 193,707,721 \cdot 761,838,257,287.$$

He admitted to having worked 20 years on the factorisation (expression as a product of prime factors)[15] of this 21-digit decimal number!

[15] Using CrypTool v1.3 you can factorize numbers in the following way: menu **Indiv. Procedures \ RSA Demonstration \ Factorisation of a Number**.

Due to the fact that the exponents of the Mersenne numbers do not use all natural numbers, but only the primes, the *experimental space* is limited considerably. The currently known Mersenne prime numbers have the exponents

$$2;\ 3;\ 5;\ 7;\ 13;\ 17;\ 19;\ 31;\ 61;\ 89;\ 107;\ 127;\ 521;\ 607;\ 1,279;\ 2,203;\ 2,281;\ 3,217;\ 4,253;\ 4,423;\ 9,689;\ 9,941,\ 11,213;\ 19,937;\ 21,701;\ 23,207;\ 44,497;\ 86,243;\ 110,503;\ 132,049;\ 216,091;\ 756,839;\ 859,433;\ 1,257,787;\ 1,398,269;\ 2,976,221;\ 3,021,377;\ 6,972,593;\ 13,466,917.$$

Thus 39 Mersenne prime numbers are currently known . For the first 38 Mersenne prime numbers we know that this list is complete. The exponents until the 39th Mersenne prime number have not yet been checked completely (see chapter 2.5 prime number tests).

The 19th number with the exponent $4,253$ was the first with at least $1,000$ digits in decimal system (the mathematician Samual Yates coined the expression *titanic* prime for this; it was discovered by Hurwitz in 1961); the 27th number with the exponent $44,497$ was the first with at least $10,000$ digits in the decimal system (Yates coined the expression *gigantic* prime for this. These names are now long outdated).

These numbers can be found at the following URLs:

http://reality.sgi.com/chongo/prime/prime_press.html
http://www.utm.edu/

**M-37 – January 1998**

The 37th Mersenne prime,

$$2^{3,021,377} - 1$$

was found in January 1998 and has 909,526 digits in the decimal system, which corresponds to 33 pages in the newspaper!

**M-38 – June 1999**

The 38th Mersenne prime, called M-38,

$$2^{6,972,593} - 1$$

was discovered in June 1999 and has $2,098,960$ digits in the decimal system (that corresponds to around 77 pages in the newspaper).

Right now (Feb. 2003) all exponents smaller than $6,972,593$ have been checked: so we can be certain, that this is really the 38th Mersenne prime number. This means there is no other Mersenne prime number between the 37th Mersenne prime number and this one.

#### M13466917 – December 2001

This number was discovered as 39th Mersenne prime (and already called M-39, despite it has not been proven yet, whether no further Mersenne prime numbers between M-38 und M13466917 do indeed exist),

$$2^{13,466,917} - 1,$$

at December 6, 2001 – more exactly, the verification of this number, found at November 14, 2001 by the Canadian student Michael Cameron, was successfully completed. This number has about 4 million decimal digits (exactly 4,053,946 digits). Trying only to print this number

$$(924947738006701322247758 \cdots 1130073855470256259071)$$

would require around 200 pages in the Financial Times.

#### GIMPS

Discovering the 39th Mersenne prime the GIMPS project (Great Internet Mersenne Prime Search) founded in 1996 already discovered for the 5th time the greatest Mersenne number to be proven being prime (`http://www.mersenne.org`).

Now more than 130,000 volunteers, amateurs and experts, are working for the GIMPS project. They connect their computers into the so called "primenet", organized by the company entropia, to find such numbers using distributed computer programs.

Currently the 4 largest known primes are Mersenne prime numbers. Only the 5th biggest known prime is another number type: it belongs to the so called generalized Fermat primes.

### 2.4.2 EFF

This search is also spurred on by a competition started by the non-profit organisation EFF (Electronic Frontier Foundation) using the means of an unknown donator. The participants are rewarded with a total of 500,000 USD if they find the longest prime number. In promoting this project, the unknown donator is not looking for the quickest computer, but rather wants to draw people's attention to the opportunities offered by *cooperative networking*
`http://www.eff.org/coopawards/prime-release1.html`

The discoverer of M-38 received 50,000 USD from the EFF for discovering the first prime with more than 1 million decimal digits. The next prize of 100,000 USD offered by EFF is for a proven prime with more than 10 million decimal digits.

Edouard Lucas (1842-1891) held the record for the longest prime number for over 70 years by proving that $2^{127} - 1$ is prime. No new record is likely to last that long.

## 2.5 Prime number tests

In order to implement secure encryption procedures we need extremely large prime numbers (in the region of $2^{2,048}$, i.e. numbers with 600 digits in the decimal system!).

Up to now we have looked for the prime factors in order to decide whether a number is prime. However, even if the smallest prime factor is enormous, the search takes too long. Factorising numbers using systematic computational division or using the sieve of Eratosthenes is only feasible using current computers for numbers with up to around 20 digits in the decimal system. The biggest number factorized into its 2 almost equal prime factors has 158 digits (see chapter 3.11.4).

However, if we know something about the *construction* of the number in question, there are extremely highly developed procedures that are much quicker. These procedures can determine the primality attribute of a number, but they cannot determine the prime factors of a number, if it is compound.

In the 17th century, Fermat wrote to Mersenne that he presumed that all numbers of the form

$$F(n) = 2^{2^n} + 1$$

are prime for all whole numbers $n$ greater than or equal to 0 (see below)

As early as in the 19th century, it was discovered that the 29-digit number

$$F(7) = 2^{2^7} + 1$$

is not prime. However, it was not until 1970 that Morrison/Billhart managed to factorise it.

$$\begin{aligned} F(7) &= 340,282,366,920,938,463,463,374,607,431,768,211,457 \\ &= 59,649,589,127,497,217 \cdot 5,704,689,200,685,129,054,721 \end{aligned}$$

Despite Fermat was wrong with this supposition, he is the originator of an important theorem in this area: Many rapid prime number tests are based on the (little) Fermat theorem put forward by Fermat in 1640 (see chapter 3.8.3).

**Theorem 2.5 ("little" Fermat).** *Let $p$ be a prime number and $a$ be any whole number, then for all $a$*

$$a^p \equiv a \bmod p.$$

*This could also be formulated as follows:*
*Let $p$ be a prime number and $a$ be any whole number that is not a multiple of $p$ (also $a \not\equiv 0 \bmod p$), then $a^{p-1} \equiv 1 \bmod p$.*

If you are not used to calculating with remainders (modulo), please simply accept the theorem. What is important here is that this sentence implies that if this equation is not met for any whole number $a$, then $p$ is not a prime! The tests (e.g. for the first formulation) can easily be performed using the *test basis* $a = 2$.

This gives us a criterion for non-prime numbers, i.e. a negative test, but no proof that a number $a$ is prime. Unfortunately Fermat's theorem does not apply — otherwise we would have a simple proof of the prime number property (or to put it in other words, we would have a simple prime number criterion).

Comment: Numbers n that have the property

$$2^n \equiv 2 \bmod n$$

but are not prime are called *pseudo prime numbers*. The first pseudo prime number (i.e. not a prime) is

$$341 = 11 \cdot 31.$$

There are numbers that pass the Fermat test with all bases and yet are not prime: these numbers are called *Carmichael numbers*. The first of these is

$$561 = 3 \cdot 11 \cdot 17.$$

A stronger test is provided by Miller/Rabin: it is only passed by so-called *strong pseudo prime numbers*. Again, there are strong pseudo prime numbers that are not primes, but this is much less often the case than for (simple) pseudo prime numbers. The smallest strong pseudo prime number base 2 is

$$15,841 = 7 \cdot 31 \cdot 73.$$

If you test all 4 bases, $2, 3, 5$ and 7, you will find only one strong pseudo prime number up to $25 \cdot 10^9$, i.e. a number that passes the test and yet is not a prime number.

More extensive mathematics behind the Rabin test delivers the probability that the number examined is prime (such probabilities are currently around $10^{-60}$).

Detailed descriptions of tests for finding out whether a number is prime can be found on Web sites such as:

http://www.utm.edu/research/primes/mersenne.shtml
http://www.utm.edu/research/primes/prove/index.html

## 2.6 Further special number types and the search for a formula for primes

There are currently no useful, open (i.e. not recursive) formulae known that only deliver prime numbers (recursive means that in order to calculate the function the same function is used with a smaller variable). Mathematicians would be happy if they could find a formula that leaves gaps (i.e. does not deliver all prime numbers) but does not deliver any composite (non-prime) numbers.

Ideally, we would like, for the number $n$, to immediately be able to obtain the $n$-th prime number, i.e. for $f(8) = 19$ or for $f(52) = 239$.

Ideas for this can be found at

http://www.utm.edu/research/primes/notes/faq/p_n.html.

Cross-reference: the table under 2.8.2 contains the precise values for the $n$th prime numbers for selected $n$.

The following enumeration contains the most common ideas for “prime number formulae”:

1. *Mersenne numbers* $f(n) = 2^n - 1$ for $n$ prime:
   As shown above, this formula seems to deliver relatively large prime numbers but - as for $n = 11$ [$f(n) = 2,047$] - it is repeatedly the case that the result even with prime exponents is not prime.
   Today, all the Mersenne primes having less than around 2,000,000 digits are known (M-38):

   http://perso.wanadoo.fr/yves.gallot/primes/index.html

2. $F(k, n) = k \cdot 2^n \pm 1$ for $n$ prime and $k$ small primes:
   For this generalisation of the Mersenne numbers there are (for small $k$) also extremely quick prime number tests (see [Knuth1981]). This can be performed in practice using software such as the Proths software from Yves Gallot
   http://www.prothsearch.net/index.html.

3. *Fermat numbers*[16] $F(n) = 2^{2^n} + 1$:
   As mentioned above, Fermat wrote to Mersenne regarding his assumption, that all numbers of this type are primes. Surprisingly he would have been able obtain a positive result using the negative prime number test for $n = 5$ based on his small theorem.

$$
\begin{array}{llll}
F(0) = 2^{2^0} + 1 = 2^1 + 1 & = 3 & \mapsto & \text{prime} \\
F(1) = 2^{2^1} + 1 = 2^2 + 1 & = 5 & \mapsto & \text{prime} \\
F(2) = 2^{2^2} + 1 = 2^4 + 1 & = 17 & \mapsto & \text{prime} \\
F(3) = 2^{2^3} + 1 = 2^8 + 1 & = 257 & \mapsto & \text{prime} \\
F(4) = 2^{2^4} + 1 = 2^{16} + 1 & = 65{,}537 & \mapsto & \text{prime} \\
F(5) = 2^{2^5} + 1 = 2^{32} + 1 & = 4{,}294{,}967{,}297 = 641 \cdot 6{,}700{,}417 & \mapsto & \text{NOT prime!} \\
F(6) = 2^{2^6} + 1 = 2^{64} + 1 & = 18{,}446{,}744{,}073{,}709{,}551{,}617 & & \\
 & = 274{,}177 \cdot 67{,}280{,}421{,}310{,}721 & \mapsto & \text{NOT prime!}
\end{array}
$$

   Within the project "Distributed Search for Fermat Number Dividers" offered by Leonid Durman there is also progress in finding new monster primes:
   http://www.fermatsearch.org/
   This website links to other webpages in Russian, Italian and German.

   The discovered factors can be compound integers or primes.

   On February 22, 2003 John Cosgrave discovered

   - the largest composite Fermat number to date and
   - the largest prime non-Mersenne number so far with 645,817 digits.

   The Fermat number

$$F[2, 145, 351] = 2^{(2^{2,145,351})} + 1$$

[16] The Fermat prime numbers play a role in circle division. As proven by Gauss, a regular $p$-edge can only be constructed with the use of a pair of compasses and a ruler, when $p$ is a Fermat prime number.

is divisible by the prime

$$p = 3 * 2^{2,145,353} + 1$$

This prime p is the largest known prime generalized Mersenne number. It was discovered only a few days after Michael Angel has come across the largest generalized Fermat number (see below). This moved GIMPS' first prime discovery, M-35 = M1398269, into 7th place.

This work was done using NewPGen from Paul Jobling's, PRP from George Woltman's, Proth from Yves Gallot's programs and also the Proth-Gallot group at St. Patrick's College, Dublin.

More details are in
    http://www.fermatsearch.org/history/cosgrave_record.htm/

4. *Generalized Fermat numbers*[17] $F(b,m) = b^{2^m} + 1$:
   Generalized Fermat numbers are more numerous than Mersenne numbers at equal size and many of them are waiting to be discovered to fill the big gaps between the Mersenne primes already found or still undiscovered. Progress in number theory made it possible that numbers, where the representation is not limited to the base 2, can be tested at almost the same speed than a Mersenne number.

   Yves Gallot wrote the program Proth.exe to investigate generalized Fermat numbers.

   Using this program at February 16, 2003 Michael Angel discovered the largest of them till then with 628,808 digits, which so became the 5th largest known prime number:

$$b^{2^{17}} + 1 = 62,722^{131,072} + 1.$$

   This moved GIMPS' first prime discovery, M-35 = M1398269, into 6th place.

   More details are in
   http://www.prothsearch.net/index.html
   http://perso.wanadoo.fr/yves.gallot/primes/index.html

5. Carmichael numbers: see above.

6. Pseudo prime numbers: see above.

7. Strong pseudo prime numbers: see above.

8. Idea based on Euclid's proof (infinite many prime numbers) $p_1 \cdot p_2 \cdots p_n + 1$:

$$\begin{array}{lll} 2{\cdot}3+1 & = 7 & \mapsto \text{ prime} \\ 2{\cdot}3{\cdot}5+1 & = 31 & \mapsto \text{ prime} \\ 2{\cdot}3{\cdot}5{\cdot}7+1 & = 211 & \mapsto \text{ prime} \\ 2{\cdot}3{\cdot}\cdot{\cdot}11+1 & = 2,311 & \mapsto \text{ prime} \\ 2\cdot 3 \cdots 13+1 & = 59\cdot 509 & \mapsto \text{ NOT prime!} \\ 2\cdot 3 \cdots 17+1 & = 19\cdot 97\cdot 277 & \mapsto \text{ NOT prime!} \end{array}$$

[17] The base of this power is no longer restricted to 2 !

9. As above but except +1: $p_1 \cdot p_2 \cdots p_n - 1$

$$\begin{array}{lll} 2 \cdot 3 - 1 & = 5 & \mapsto \text{prime} \\ 2 \cdot 3 \cdot 5 - 1 & = 29 & \mapsto \text{prime} \\ 2 \cdot 3 \cdots 7 - 1 & = 11 \cdot 19 & \mapsto \text{NOT prime!} \\ 2 \cdot 3 \cdots 11 - 1 & = 2,309 & \mapsto \text{prime} \\ 2 \cdot 3 \cdots 13 - 1 & = 30,029 & \mapsto \text{prime} \\ 2 \cdot 3 \cdots 17 - 1 & = 61 \cdot 8,369 & \mapsto \text{NOT prime!} \end{array}$$

10. *Euclidean numbers* $e_n = e_0 \cdot e_1 \cdots e_{n-1} + 1$ with $n$ greater than or equal to 1 and $e_0 := 1$. $e_{n-1}$ is not the $(n-1)$th prime number, but the number previously found here. Unfortunately this formula is not open but recursive. The sequence starts with

$$\begin{array}{lll} e_1 = 1 + 1 & = 2 & \mapsto \text{prime} \\ e_2 = e_1 + 1 & = 3 & \mapsto \text{prime} \\ e_3 = e_1 \cdot e_2 + 1 & = 7 & \mapsto \text{prime} \\ e_4 = e_1 \cdot e_2 \cdot e_3 + 1 & = 43 & \mapsto \text{prime} \\ e_5 = e_1 \cdot e_2 \cdots e_4 + 1 & = 13 \cdot 139 & \mapsto \text{NOT prime!} \\ e_6 = e_1 \cdot e_2 \cdots e_5 + 1 & = 3,263,443 & \mapsto \text{prime} \\ e_7 = e_1 \cdot e_2 \cdots e_6 + 1 & = 547 \cdot 607 \cdot 1,033 \cdot 31,051 & \mapsto \text{NOT prime!} \\ e_8 = e_1 \cdot e_2 \cdots e_7 + 1 & = 29,881 \cdot 67,003 \cdot 9,119,521 \cdot 6,212,157,481 & \mapsto \text{NOT prime!} \end{array}$$

$e_9, \cdots, e_{17}$ are also composite, which means that this formula is not particularly useful. Comment: However, what is special about these numbers is that any pair of them does not have a common factor other than 1. They are therefore *relatively prime*.

11. $f(n) = n^2 + n + 41$:
This sequence starts off very *promisingly*, but is far from being a proof.

$$\begin{array}{ll} f(0) = 41 & \mapsto \text{prime} \\ f(1) = 43 & \mapsto \text{prime} \\ f(2) = 47 & \mapsto \text{prime} \\ f(3) = 53 & \mapsto \text{prime} \\ f(4) = 61 & \mapsto \text{prime} \\ f(5) = 71 & \mapsto \text{prime} \\ f(6) = 83 & \mapsto \text{prime} \\ f(7) = 97 & \mapsto \text{prime} \\ \vdots & \\ f(39) = 1,601 & \mapsto \text{prime} \\ f(40) = 11 \cdot 151 & \mapsto \text{NOT prime!} \\ f(41) = 41 \cdot 43 & \mapsto \text{NOT prime!} \end{array}$$

The first 40 values are prime numbers (which have the obvious regularity that their difference starts with 2 and increases by 2 each time), but the 41th and 42th values are not prime numbers. It is easy to see that $f(41)$ cannot be a prime number: $f(41) = 41^2 + 41 + 41 = 41(41 + 1 + 1) = 41 \cdot 43$.

12. $f(n) = n^2 - 79 \cdot n + 1,601$:
This function delivers prime numbers for all values from $n = 0$ to $n = 79$. Unfortunately $f(80) = 1,681 = 11 \cdot 151$ is not a prime number. To this date, no function has been found that delivers more prime numbers in a row. On the other hand, each prime occurs twice (first in the decreasing then in the increasing sequence), which means that the algorithm delivers a total of 40 difference prime values (the same ones as the function from point 11).

| | | | | | |
|---|---|---|---|---|---|
| $f(0) = 1,601$ | $\mapsto$ | prime | $f(28) = 173$ | $\mapsto$ | prime |
| $f(1) = 1,523$ | $\mapsto$ | prime | $f(29) = 151$ | $\mapsto$ | prime |
| $f(2) = 1,447$ | $\mapsto$ | prime | $f(30) = 131$ | $\mapsto$ | prime |
| $f(3) = 1,373$ | $\mapsto$ | prime | $f(31) = 113$ | $\mapsto$ | prime |
| $f(4) = 1,301$ | $\mapsto$ | prime | $f(32) = 97$ | $\mapsto$ | prime |
| $f(5) = 1,231$ | $\mapsto$ | prime | $f(33) = 83$ | $\mapsto$ | prime |
| $f(6) = 1,163$ | $\mapsto$ | prime | $f(34) = 71$ | $\mapsto$ | prime |
| $f(7) = 1,097$ | $\mapsto$ | prime | $f(35) = 61$ | $\mapsto$ | prime |
| $f(8) = 1,033$ | $\mapsto$ | prime | $f(36) = 53$ | $\mapsto$ | prime |
| $f(9) = 971$ | $\mapsto$ | prime | $f(37) = 47$ | $\mapsto$ | prime |
| $f(10) = 911$ | $\mapsto$ | prime | $f(38) = 43$ | $\mapsto$ | prime |
| $f(11) = 853$ | $\mapsto$ | prime | $f(39) = 41$ | $\mapsto$ | prime |
| $f(12) = 797$ | $\mapsto$ | prime | $f(40) = 41$ | $\mapsto$ | prime |
| $f(13) = 743$ | $\mapsto$ | prime | $f(41) = 43$ | $\mapsto$ | prime |
| $f(14) = 691$ | $\mapsto$ | prime | $f(42) = 47$ | $\mapsto$ | prime |
| $f(15) = 641$ | $\mapsto$ | prime | $f(43) = 53$ | $\mapsto$ | prime |
| $f(16) = 593$ | $\mapsto$ | prime | $\cdots$ | | |
| $f(17) = 547$ | $\mapsto$ | prime | $f(77) = 1,447$ | $\mapsto$ | prime |
| $f(18) = 503$ | $\mapsto$ | prime | $f(78) = 1,523$ | $\mapsto$ | prime |
| $f(19) = 461$ | $\mapsto$ | prime | $f(79) = 1,601$ | $\mapsto$ | prime |
| $f(20) = 421$ | $\mapsto$ | prime | $f(80) = 11 \cdot 151$ | $\mapsto$ | NOT prime! |
| $f(21) = 383$ | $\mapsto$ | prime | $f(81) = 41 \cdot 43$ | $\mapsto$ | NOT prime! |
| $f(22) = 347$ | $\mapsto$ | prime | $f(82) = 1,847$ | $\mapsto$ | prime |
| $f(21) = 383$ | $\mapsto$ | prime | $f(83) = 1,933$ | $\mapsto$ | prime |
| $f(22) = 347$ | $\mapsto$ | prime | $f(84) = 43 \cdot 47$ | $\mapsto$ | NOT prime! |
| $f(23) = 313$ | $\mapsto$ | prime | | | |
| $f(24) = 281$ | $\mapsto$ | prime | | | |
| $f(25) = 251$ | $\mapsto$ | prime | | | |
| $f(26) = 223$ | $\mapsto$ | prime | | | |
| $f(27) = 197$ | $\mapsto$ | prime | | | |

13. Polynomial functions $f(x) = a_n x^n + a_{n-1}x^{n-1} + \cdots + a_1 x^1 + a_0$ ($a_i$ in $\mathbb{Z}$, $n \geq 1$):

    There exists no such polynomial that for all $x$ in $\mathbb{Z}$ only delivers prime values. For a proof of this, please refer to [Padberg1996, p. 83 f.], where you will also find further details about prime number formulae.

    This means there is no hope in looking for further formulae similar to that of type 11. or 12.

14. Catalan, after whom the so-called *Catalan numbers* $A(n) = (1/(n+1)) * (2n)!/(n!)^2$ are named, conjectured that $C_4$ is a prime:

$$\begin{aligned} &C_0 = 2, \\ &C_1 = 2^{C_0} - 1, \\ &C_2 = 2^{C_1} - 1, \\ &C_3 = 2^{C_2} - 1, \\ &C_4 = 2^{C_3} - 1, \cdots \end{aligned}$$

    (see http://www.utm.edu/research/primes/mersenne.shtml under Conjectures and Unsolved Problems).

    This sequence is also defined recursively and increases extremely quickly. Does it only consist of primes?

$$\begin{array}{lll} C(0) = 2 & & \mapsto \text{ prime} \\ C(1) = 2^2 - 1 & = 3 & \mapsto \text{ prime} \\ C(2) = 2^3 - 1 & = 7 & \mapsto \text{ prime} \\ C(3) = 2^7 - 1 & = 127 & \mapsto \text{ prime} \\ C(4) = 2^{127} - 1 & = 170,141,183,460,469,231,731,687,303,715,884,105,727 & \mapsto \text{ prime} \end{array}$$

    It is not (yet) known whether $C_5$ and higher elements are prime, but this is not very likely. In any case, it has not been proved that this formula delivers only primes.

## 2.7 Density and distribution of the primes

As Euclid discovered, there is an infinite number of primes. However, some infinite sets are *denser* than others. Within the set of natural numbers, there is an infinite number of even, uneven and square numbers.

The following proves that there are more even numbers than square ones:

- the size of the $n$th element:
  The $n$th element of the even numbers is $2n$; the $n$th element of the square numbers is $n^2$. Because for all $n > 2$: $2n < n^2$, the $n$th even number occurs much earlier than the $n$th square number. Thus the even numbers are distributed more densely and we can say that there are more even numbers than square ones.

- the number of values that are less than or equal to a certain *maximum value* $x$ in $\mathbb{R}$ is: There are $[x/2]$ such even numbers and $[\sqrt{x}]$ square numbers. Because for large $x$ the value $x/2$ is much greater than the square root of 2, we can again say that there are more even numbers.

**Theorem 2.6.** *For large $n$: The value of the $n$-th prime $P(n)$ is asymptotic to $n \cdot ln(n)$, i.e. the limit of the relation $P(n)/(n \cdot \ln n)$ is equal to 1 if $n$ tends to infinity.*

The definition is similar for the number of prime numbers $PI(x)$ that do not exceed the maximum value $x$:

**Theorem 2.7.** *$PI(x)$ is asymptotic to $x/ln(x)$.*

This is the **prime number theorem**. It was put forward by Legendre (1752-1833) and Gauss (1777-1855) but not proved until over 100 years later.

(Cross-reference: overview under 2.8.1 of the number of prime numbers in various intervals).

For large $n$, $P(n)$ lies between $2n$ and $n^2$. This means that there are fewer prime numbers than even natural numbers but more prime numbers than square numbers.

These formulae, which only apply when n tends to infinity, can be replaced by more precise formulae. For $x \geq 67$:

$$ln(x) - 1,5 < x/PI(x) < ln(x) - 0,5$$

Given that we know $PI(x) = x/\ln x$ only for very large $x$ ($x$ tending towards infinity), we can create the following overview:

| $x$ | $ln(x)$ | $x/ln(x)$ | $PI(x)(counted)$ | $PI(x)/(x/ln(x))$ |
|---|---|---|---|---|
| $10^3$ | 6.908 | 144 | 168 | 1.160 |
| $10^6$ | 13.816 | 72,386 | 78,498 | 1.085 |
| $10^9$ | 20.723 | 48,254,942 | 50,847,534 | 1.054 |

For a binary number (number in the binary system) of the length 250 bits ($2^{250}$ is approximately $= 1.809251 * 10^{75}$):

$$PI(250) = 2^{250}/(250 \cdot \ln 2) \text{ is approximately } = 2^{250}/173.28677 = 1.045810 \cdot 10^{73}.$$

We can therefore expect that the set of numbers with a bit length of less than 250 contains approximately $10^{73}$ primes (a reassuring result?!).

We can also express this as follows: Let us consider a *random* natural number $n$. Then the probability that this number is prime is around $1/\ln(n)$. For example, let us take numbers in the region of $10^{16}$. Then we must consider $16 \cdot \ln 10 = 36,8$ numbers (on average) until we find a prime. A precise investigation shows: There are 10 prime numbers between $10^{16} - 370$ and $10^{16} - 1$.

Under the heading *How Many Primes Are There* at

http://www.utm.edu/research/primes/howmany.shtml you will find numerous other details.

Using the following Web site:

http://www.math.Princeton.EDU/~arbooker/nthprime.html

you can easily determine $PI(x)$.

The **distribution** of primes displays several irregularities for which no system has yet been found: On the one hand, many occur closely together, like 2 and 3, 11 and 13, 809 and 811, on the other hand large gaps containing no primes also occur. For example, no primes lie between 113 and 127, 293 and 307, 317 and 331, 523 and 541, 773 and 787, 839 and 853 as well as between 887 and 907.
For details, please see:

http://www.utm.edu/research/primes/notes/gaps.html

This is precisely part of what motivates mathematicians to discover its secrets.

**Sieve of Eratosthenes**

An easy way of calculating all $PI(x)$ primes less than or equal to $x$ is to use the sieve of Eratosthenes. In the 3rd century B.C., he found an extremely easy, automatic way of finding this out. To begin with, you write down all numbers from 2 to $x$, circle 2, then cross out all multiples of 2. Next, you circle the lowest number that hasn't been circled or crossed out (3) and again cross out all multiples of this number, etc. You only need to continue until you reach the largest number whose square is less than or equal to $x$.

Apart from 2, prime numbers are never even. Apart from 2 and 5, prime numbers never end in 2, 5 or 0. So you only need to consider numbers ending in 1, 3, 7, 9 anyway (there are infinite primes ending in these numbers; see [Tietze1973, vol. 1, p. 137]).

You can now find a large number of finished programs on the Internet - often complete with source code - allowing you to experiment with large numbers yourself. You also have access to large databases that contain either a large number of primes or the factorisation of numerous composite numbers. The **Cunningham project** for example, determines the factors of all composite numbers that are formed as follows:

$$f(n) = b^n \pm 1 \quad \text{for } b = 2, 3, 5, 6, 7, 10, 11, 12$$

($b$ is not equal to multiples of bases already used, such as $4, 8, 9$).

Details of this can be found at:

http://www.cerias.purdue.edu/homes/ssw/cun

## 2.8 Notes

**Proven statements / theorems about primes**

- For each number $n$ in $\mathbf{N}$ there are $n$ consecutive natural numbers that are not primes. A proof of this can be found in [Padberg1996, p. 79].

- The Hungarian mathematician Paul Erdös (1913-1996) proved: Between each random number not equal to 1 and its double, there is at least one prime. He was not the first to prove this theorem, but proved it in a much simpler manner than those before him.
- There is a real number a such that the function $f : \mathbf{N} \to \mathbb{Z}$ where $n \mapsto a^{3^n}$ only delivers primes for all $n$ (see [Padberg1996, p. 82]). Unfortunately, problems arise when we try to determine $a$ (see below).

### Proven statements / conjectures about primes

- The German mathematician Christian Goldbach (1690-1764) conjectured: Every even natural number greater than 2 can be represented as the sum of two prime numbers. Computers have verified[18] the Goldbach conjecture for all even numbers up to $4 * 10^{14}$ but no general proof has yet been found[19].
- The German mathematician Bernhard Riemann (1826-1866) put forward a formula for the distribution of primes that would further improve the estimate. However, this has neither been proved nor disproved so far.

### Open questions

Twin primes are prime numbers whose difference is 2. Examples include 5 and 7 or 101 and 103. Triplet primes, however, only occur once: 3, 5, 7. For all other sets of three consecutive uneven numbers, one of them is always divisible by 3 and thus not a prime.

[18] It is generally accepted today, that the Goldbach Conjecture is true, i. e. valid for all even natural numbers bigger than 2. In 1999, mathematician Jörg Richstein from the computer sciences institute at the University of Giessen, studied even numbers up to 400 billion and found no contradictory example (see `http://www.informatik.uni-giessen.de/staff/richstein/de/Goldbach.html`). Nevertheless, this does not provide us with general proof.
The fact is that despite all efforts, Goldbach's conjecture has to date not been proven. This leads one to believe that since the pioneer work of the Austrian mathematician Kurt Gödel is well-known, not every true mathematical theorem is provable (see http://www.mathematik.ch/mathematiker/goedel.html). Perhaps Goldbach's conjecture was correct, but in any case the proof will never be found. Conversely, that will presumably also remain unproven.

[19] The English publisher *Faber* and the American publisher *Bloomsbury* issued in 2000 the 1992 published book "Uncle Petros and Goldbach's Conjecture" by Apostolos Doxiadis. It's the story of an old maths professor who fails to prove a more than 250 year old puzzle. To boost the sales figures the English and American publishers have offered a prize of 1 million USD, if someone can prove the conjecture – which should be published by 2004 in a well-known mathematical journal.
Surprisingly only British and American citizens are allowed to participate.
The theorem which has come closest so far to Goldbach's conjecture was proved by Chen Jing-Run in 1966 in a way which is somewhat hard to understand: Each even integer greater than 2 is the sum of one prime and of the product of two primes. E.g.: $20 = 5 + 3 * 5$.
Most of the research about the Goldbach conjecture is collected in the book: "Goldbach Conjecture", ed. Wang Yuan, 1984, World scientific Series in Pure Maths, Vol. 4.
Especially this conjecture makes it clear, that even today we do not have a complete understanding of the deeper connections between addition and multiplication of natural numbers.

- The number of twin primes is an open question: infinite or limited number? The largest twin primes known today are $1,693,965 \cdot 2^{66,443} \pm 1$.
- Does a formula exist for calculating the number of twin primes per interval?
- The above proof of the function $f: N \to Z$ with $n \mapsto a^{3^n}$ only guarantees the existence of such a number $a$. How can we determine this number $a$ and will it have a value, making the function also of some practical interest?
- Is there an infinite number of Mersenne prime numbers?
- Is there an infinite number of Fermat prime numbers?
- Does a polynomial time algorithm exist for calculating the prime factors of a number (see [Klee1997, p. 167])? This question can be divided into the two following questions:
  - Does a polynomial time algorithm exist that decides whether a number is prime?
  - Does a polynomial time algorithm exist that calculates for a composite number $n$ a non-trivial (i.e. other than 1 and $n$) factor of $n$?[20]

**Further interesting topics regarding prime numbers**

This chapter doesn't consider other number theory topics such as divisibility rules, modulus calculation, modular inverses, modular powers and roots, Chinese remainder theorem, Euler PHI function, perfect numbers. Some of these are considered in the next chapter.

[20] Please see chapters 3.11.3 and 3.11.4.

### 2.8.1 Number of prime numbers in various intervals

| Ten-sized intervals | | Hundred-sized intervals | | Thousand-sized intervals | |
|---|---|---|---|---|---|
| Interval | Number | Interval | Number | Interval | Number |
| 1-10 | 4 | 1-100 | 25 | 1-1000 | 168 |
| 11-20 | 4 | 101-200 | 21 | 1001-2000 | 135 |
| 21-30 | 2 | 201-300 | 16 | 2001-3000 | 127 |
| 31-40 | 2 | 301-400 | 16 | 3001-4000 | 120 |
| 41-50 | 3 | 401-500 | 17 | 4001-5000 | 119 |
| 51-60 | 2 | 501-600 | 14 | 5001-6000 | 114 |
| 61-70 | 2 | 601-700 | 16 | 6001-7000 | 117 |
| 71-80 | 3 | 701-800 | 14 | 7001-8000 | 107 |
| 81-90 | 2 | 801-900 | 15 | 8001-9000 | 110 |
| 91-100 | 1 | 901-1000 | 14 | 9001-10000 | 112 |

Further intervals:

| Interval | Number | Average number per 1000 |
|---|---|---|
| 1 - 10,000 | 1,229 | 122.900 |
| 1 - 100,000 | 9,592 | 95.920 |
| 1 - 1,000,000 | 78,498 | 78.498 |
| 1 - 10,000,000 | 664,579 | 66.458 |
| 1 - 100,000,000 | 5,761,455 | 57.615 |
| 1 - 1,000,000,000 | 50,847,534 | 50.848 |
| 1 - 10,000,000,000 | 455,052,512 | 45.505 |

### 2.8.2 Indexing prime numbers ($n$-th prime number)

| Index | Precise value | Rounded value | Comment |
|---|---|---|---|
| 1 | 2 | 2 | |
| 2 | 3 | 3 | |
| 3 | 5 | 5 | |
| 4 | 7 | 7 | |
| 5 | 11 | 11 | |
| 6 | 13 | 13 | |
| 7 | 17 | 17 | |
| 8 | 19 | 19 | |
| 9 | 23 | 23 | |
| 10 | 29 | 29 | |
| 100 | 541 | 541 | |
| 1,000 | 7,917 | 7,917 | |
| 664,559 | 9,999,991 | 9.99999E+06 | All prime numbers up to 1E+07 were known at the beginning of the 20th century. |
| 1E+06 | 15,485,863 | 1.54859E+07 | |
| 6E+06 | 104,395,301 | 1.04395E+08 | This prime was discovered in 1959. |
| 1E+07 | 179,424,673 | 1.79425E+08 | |
| 1E+09 | 22,801,763,489 | 2.28018E+10 | |
| 1E+12 | 29,996,224,275,833 | 2.99962E+13 | |

Comment: With gaps, extremely large prime numbers were discovered at an early stage.

Web links:
http://www.math.Princeton.EDU/~arbooker/nthprime.html.

Output of the $n$-th prime number
See `http://www.utm.edu/research/primes/notes/by_year.html`.

### 2.8.3 Orders of magnitude / dimensions in reality

In the description of cryptographic protocols and algorithms, numbers occur that are so large or so small that they are inaccessible to our intuitive understanding. It may therefore be useful to provide comparative numbers from the real world around us so that we can develop a feeling for the security of cryptographic algorithms. Some of the numbers listed below originate from [Schwenk1996] and [Schneier1996, p.18].

| | | |
|---|---|---|
| Probability that you will be hijacked on your next flight | $5.5 \cdot 10^{-6}$ | |
| Probability of 6 correct numbers in the lottery | $7.1 \cdot 10^{-8}$ | |
| Annual probability of being hit by lightning | $10^{-7}$ | |
| Risk of being hit by a meteorite | $1.6 \cdot 10^{-12}$ | |
| Time until the next ice age (in years) | 14000 | $(2^{14})$ |
| Time until the sun dies (in years) | $10^{9}$ | $(2^{30})$ |
| Age of the Earth (in years) | $10^{9}$ | $(2^{30})$ |
| Age of the universe (in years) | $10^{10}$ | $(2^{34})$ |
| Number of the Earth's atoms | $10^{51}$ | $(2^{170})$ |
| Number of the sun's atoms | $10^{57}$ | $(2^{190})$ |
| Number of atoms in the universe (without dark material) | $10^{77}$ | $(2^{265})$ |
| Volume of the universe (in $cm^3$) | $10^{84}$ | $(2^{280})$ |

### 2.8.4 Special values in the binary and decimal systems

| Dual system | Decimal system |
|---|---|
| $2^{10}$ | 1024 |
| $2^{40}$ | $1.09951 \cdot 10^{12}$ |
| $2^{56}$ | $7.20576 \cdot 10^{16}$ |
| $2^{64}$ | $1.84467 \cdot 10^{19}$ |
| $2^{80}$ | $1.20893 \cdot 10^{24}$ |
| $2^{90}$ | $1.23794 \cdot 10^{27}$ |
| $2^{112}$ | $5.19230 \cdot 10^{33}$ |
| $2^{128}$ | $3.40282 \cdot 10^{38}$ |
| $2^{150}$ | $1.42725 \cdot 10^{45}$ |
| $2^{160}$ | $1.46150 \cdot 10^{48}$ |
| $2^{250}$ | $1.80925 \cdot 10^{75}$ |
| $2^{256}$ | $1.15792 \cdot 10^{77}$ |
| $2^{320}$ | $2.13599 \cdot 10^{96}$ |
| $2^{512}$ | $1.34078 \cdot 10^{154}$ |
| $2^{768}$ | $1.55252 \cdot 10^{231}$ |
| $2^{1024}$ | $1.79769 \cdot 10^{308}$ |
| $2^{2048}$ | $3.23170 \cdot 10^{616}$ |

Calculation using GMP, for example: http://www.gnu.ai.mit.edu.

## References

[Aaronson2003] Scott Aaronson,
*The Prime Facts: From Euclid to AKS*,
http://www.cs.berkeley.edu/~aaronson/prime.ps.
Only after I had completed this article, did I come across the extremely well-written paper by Scott Aaronson, which also offers a didactically well done intoduction to this topic. It is humorous and easy to read but at the same time precise and erudite.

[Bartholome1996] A. Bartholom, J. Rung, H. Kern,
*Zahlentheorie für Einsteiger*, Vieweg 1995, 2nd edition 1996.

[Blum1999] W. Blum,
*Die Grammatik der Logik*, dtv, 1999.

[Bundschuh1998] Peter Bundschuh,
*Einführung in die Zahlentheorie*, Springer 1988, 4th edition 1998.

[Doxiadis2000] Apostolos Doxiadis,
*Uncle Petros and the Goldbach's Conjecture*,
Faber/Bloomsbury, 2000.

[Graham1989] R.E. Graham, D.E. Knuth, O. Patashnik,
*Concrete Mathematics*, Addison-Wesley, 1989.

[Klee1997] V. Klee, S. Wagon,
*Ungelöste Probleme in der Zahlentheorie und der Geometrie der Ebene*,
Birkhäuser Verlag, 1997.

[Knuth1981] Donald E. Knuth,
*The Art of Computer Programming, vol 2: Seminumerical Algorithms*,
Addison-Wesley, 1969, 2nd edition 1981.

[Lorenz1993] F. Lorenz,
*Algebraische Zahlentheorie*, BI Wissenschaftsverlag, 1993.

[Padberg1996] F. Padberg,
*Elementare Zahlentheorie*, Spektrum Akademischer Verlag 1988, 2nd edition 1996.

[Pieper1983] H. Pieper,
*Zahlen aus Primzahlen*, Verlag Harri Deutsch 1974, 3rd edition 1983.

[Richstein1999] J. Richstein,
*Verifying the Goldbach Conjecture up to* $4 * 10^{14}$, Mathematics of Computation.

[Schneier1996] Bruce Schneier,
*Applied Cryptography, Protocols, Algorithms, and Source Code in C*,
Wiley and Sons, 2nd edition 1996.

[Schroeder1999] M.R. Schroeder,
*Number Theory in Science and Communication*,
Springer 1984, 3rd edition 1997, Corrected Printing 1999.

[Schwenk1996] J. Schwenk
*Conditional Access*, in taschenbuch der telekom praxis 1996,
Hrgb. B. Seiler, Verlag Schiele und Schön, Berlin.

[Tietze1973] H. Tietze,
*Gelöste und ungelöste mathematische Probleme*,
Verlag C. H. Beck 1959, 6th edition 1973.

## Web links

1. GIMPS (Great Internet Mersenne-Prime Search)
   www.mersenne.org is the home page of the GIMPS project,
   http://www.mersenne.org/prime.htm

2. The Proth Search Page with the Windows program by Yves Gallot
   http://prothsearch.net/index.html

3. Generalized Fermat Prime Search
   http://perso.wanadoo.fr/yves.gallot/primes/index.html

4. Distributed Search for Fermat Number Dividers
   http://www.fermatsearch.org/

5. At the University of Tennessee you will find extensive research results about prime numbers.
   http://www.utm.edu/

6. The best overview about prime numbers is offered from my point of view by "The Prime Pages" from Chris Caldwell.
   http://www.utm.edu/research/primes

7. Descriptions e.g. about prime number tests
   http://www.utm.edu/research/primes/mersenne.shtml
   http://www.utm.edu/research/primes/prove/index.html

8. Showing the $n$-th prime number
   http://www.utm.edu/research/primes/notes/by_year.html

9. The supercomputer manufacturer SGI Cray Research not only employed brilliant mathematicians but also used the prime number tests as benchmarks for their machines.
   http://reality.sgi.com/chongo/prime/prime_press.html

10. http://www.eff.org/coop-awards/prime-release1.html

11. http://www.math.Princeton.EDU/~arbooker/nthprime.html

12. http://www.cerias.purdue.edu/homes/ssw/cun

13. http://www.informatik.uni-giessen.de/staff/richstein/de/Goldbach.html

14. http://www.mathematik.ch/mathematiker/goedel.html

15. http://www.mscs.dal.ca/~dilcher/goldbach/index.html

### Acknowledgments

I would like to take this opportunity to thank Mr. Henrik Koy and Mr. Roger Oyono for their very constructive proof-reading of this article.

# 3 Introduction to Elementary Number Theory with Examples

(Bernhard Esslinger, besslinger@web.de, July 2001, Updates: Dec. 2001, June 2002, Feb. 2003)

This "introduction" is for people with a mathematical interest. There is no more pre-knowledge necessary than what you learn in the secondary school.

We intentionally had "beginners" in mind; we did not take the approach of mathematical text-books, called "introduction", which cannot be understood at the first reading further than page 5 and which have the real purpose to deliver all information that special monographs can be read.

## 3.1 Mathematics and cryptography

A large proportion of modern, asymmetric cryptography is based on mathematical knowledge – on the properties ("laws") of whole numbers, which are investigated in elementary number theory. Here, the word "elementary" means that questions raised in number theory are essentially rooted in the set of natural and whole numbers.

Further mathematical disciplines currently used in cryptography include (see [Bauer1995, p. 2], [Bauer2000, p. 3]) :

- Group theory
- Combination theory
- Complexity theory
- Ergodic theory
- Information theory.

Number theory or arithmetic (the emphasis here is more on the aspect of performing calculations with numbers) was established by Carl Friedrich Gauss as a special mathematical discipline. Its elementary features include the greatest common divisor[21] (gcd), congruence (remainder classes), factorisation, the Euler-Fermat theorem and primitive roots. However, the most important aspect is prime numbers and their multiplicative operation.

For a long time, number theory was considered to be the epitome of pure research, the ideal example of research in the ivory tower. It delved into "the mysterious laws of the realm of numbers", giving rise to philosophical considerations as to whether it described elements that exist everywhere in nature or whether it artificially constructed elements (numbers, operators and properties).

We now know that patterns from number theory can be found everywhere in nature. For example, the ratio of rotating counterclockwise and rotating clockwise spirals in a sunflower is equal to two

[21] This article deals with the gcd (greatest common divisor) in Appendix A of this chapter.

consecutive Fibonacci numbers[22], for example 21 : 34.

Also, at the latest when number theory was applied in modern cryptography, it became clear that a discipline that had been regarded as purely theoretical for centuries actually had a practical use. Today, experts in this field are in great demand on the job market.

Applications in (computer) security now use cryptography because this mathematical discipline is simply better and easier to prove than all other "creative" substitution procedures that have been developed over the course of time and better than all sophisticated physical methods such as those used to print bank notes [Beutelspacher1996, p. 4].

This article explains the basics of elementary number theory in a way that you can easily understand. It provides numerous examples and very rarely goes into any proofs (these can be found in mathematical textbooks).

The goal is not to exhaustively explain the number theory findings, but to show the essential procedures. The volume of the content is so oriented that the reader can understand and apply the RSA method.

For this purpose we will use both theory and examples to explain how to perform calculations in finite sets and describe how these techniques are applied in cryptography. Particular attention will be paid to the traditional Diffie-Hellman (DH) and RSA public key procedures.

Additionally I added some qualified statements about the security of the RSA algorithm.

> *Carl Friedrich Gauss (30.4.1777–23.2.1855):*
> Mathematics is the queen of sciences and number theory is the queen of mathematics.

### 3.2 Introduction to number theory

Number theory arose from interest in positive whole numbers $1, 2, 3, 4, \cdots$, also referred to as the set of natural numbers *natural numbers* $\mathbb{N}$. These are the first mathematical constructs used by human civilisation. According to Kronecker[23], they are a creation of God. In Dedekind's[24] opinion, they are a creation of the human intellect. Dependent upon one's ideology, this is an unsolvable contradiction or one and the same thing.

In ancient times, no distinction was made between number theory and numerology, which attributed a mystical significance to specific numbers. In the same way as astronomy and chemistry gradually detached themselves from astrology and alchemy during the Renaissance (from the 14th century), number theory also separated itself from numerology.

[22] The sequence of Fibonacci numbers $(a_i)_{i\in\mathbb{N}}$ is defined by the "recursive" rule $a_1 := a_2 := 1$ and for all numbers $n = 1, 2, 3, \cdots$ we define $a_{n+2} := a_{n+1} + a_n$. This historical sequence can be found in many interesting forms in nature (for example, see [Graham1994, p. 290 ff] or the website of Ron Knott, which is devoted to Fibonacci numbers). A lot is known about the Fibonacci sequence and it is used today as an important tool in mathematics.

[23] Leopold Kronecker, German mathematician, 1823−1891.

[24] Julius Wilhelm Richard Dedekind, German mathematician, 06.10.1831−12.02.1916.

Number theory has always been a source of fascination – for both amateurs and professional mathematicians. In contrast to other areas of mathematics, many of the problems and theorems in number theory can be understood by non-experts. On the other hand, mathematicians often take a long time to find solutions to the problems or prove the theorems. It is therefore one thing to pose good questions but quite another matter to find the answer. One example of this is what is known as Fermat's Last (or large) theorem[25].

Up until the mid 20th century, number theory was considered to be the purest area of mathematics, an area that had no practical use in the real world. This changed with the development of computers and digital communication, as number theory was able to provide several unexpected solutions to real-life tasks. At the same time, advances in information technology allowed specialists in number theory to make huge progress in factorising large numbers, finding new prime numbers, testing (old) conjectures and solving numerical problems that were previously impossible to solve. Modern number theory is made up of areas such as:

- Elementary number theory
- Algebraic number theory
- Analytic number theory
- Geometric number theory
- Combinatorial number theory
- Numeric number theory
- Probability theory.

All of the different areas are concerned with questions regarding whole numbers (both positive and negative whole numbers plus zero). However, they each have different methods of dealing with them.

This article only deals with the area of elementary number theory.

[25] One of the things you learn in mathematics at school is Pythagoras theorem, which states the following for a right-angle triangle: $a^2 + b^2 = c^2$, where $a$ and $b$ are the lengths of the sides containing the right angle and $c$ is the length of the hypotenuse. Fermat famously proposed that $a^n + b^n \neq c^n$ for whole-number exponents $n > 2$. Unfortunately, the letter in which Fermat made the claim did not have enough space for him to prove it. The theorem was not proved until over 300 years later [Wiles1994, p. 433-551].

### 3.2.1 Convention

Unless stated otherwise:

- The letter $a, b, c, d, e, k, n, m, p, q$ are used to present whole numbers.
- The letter $i$ and $j$ represent natural numbers.
- The letter $p$ always represents a prime number.
- Te sets $\mathbb{N} = \{1, 2, 3, \cdots\}$ and $\mathbb{Z} = \{\cdots, -3, -2, -1, 0, 1, 2, 3, \cdots\}$ are the *natural* and *whole* numbers respectively.

*Joanne K. Rowling*[26]
This isn't magic – it's logic – a puzzle. A lot of the greatest wizards haven't got an ounce of logic.

## 3.3 Prime numbers and the first fundamental theorem of elementary number theory

Many of the problems in elementary number theory are concerned with prime numbers.

Every whole number has divisors or factors. The number 1 has just one itself, whereas the number 12 has the six factors 1, 2, 3, 4, 6 and 12[27]. Many numbers are only divisible by themselves and by 1. When it comes to multiplication, these can be regarded as the "atoms" in the realm of numbers.

**Definition 3.1. Prime numbers** *are natural numbers greater than 1 that can only be divided by 1 and themselves.*

By definition, 1 is not a prime number.

If we write down the prime numbers in ascending order (prime number sequence), then we get:

$$2,\ 3,\ 5,\ 7,\ 11,\ 13,\ 17,\ 19,\ 23,\ 29,\ 31,\ 37,\ 41,\ 43,\ 47,\ 53,\ 59,\ 61,\ 67,\ 71,\ 73,\ 79,\ 83,\ 89,\ 97, \cdots.$$

The first 100 numbers include precisely 25 prime numbers. After this, the percentage of primes decreases, but never reaches zero.

We come across whole numbers that are prime fairly often. In the last decade only, three years were prime: $1993, 1997$ and $1999$. If they were rare, cryptography would not be able to work with them to the extent it does.

Prime numbers can be factorised in a unique ("*trivial*") way:

$$\begin{aligned} 5 &= 1 * 5 \\ 17 &= 1 * 17 \\ 1,013 &= 1 * 1,013 \\ 1,296,409 &= 1 * 1,296,409. \end{aligned}$$

**Definition 3.2.** *Natural numbers greater than* 1 *that are not prime are called* **composite numbers**. *These have at least two factors other than* 1.

[26] Joanne K. Rowling, "Harry Potter and the Philosopher's Stone", Bloomsbury, (c) 1997, chapter "Through the trapdoor", p. 307, by Hermine.

[27] Due to the fact that 12 has so many factors, this number – and multiples of this number – is often found in everyday life: the 12-hour scale on clocks, the 60 minutes in an hour, the 360-degree scale for measuring angles, etc. If we divide these scales into segments, the segments often turn out to be whole numbers. These are easier to use in mental arithmetic than fractions.

Examples of the decomposition of such numbers into prime factors:

$$\begin{aligned}
4 &= 2*2 \\
6 &= 2*3 \\
91 &= 7*13 \\
161 &= 7*23 \\
767 &= 13*59 \\
1,029 &= 3*7^3 \\
5,324 &= 22*11^3.
\end{aligned}$$

**Theorem 3.1.** *Each composite number $a$ has a lowest factor greater than 1. This factor is a prime number $p$ and is less than or equal to the square root of $a$.*

All whole numbers greater than 1 can be expressed as a product of prime numbers — in a *unique* way.

This is the claim of the 1st *fundamental theorem of number theory* (= fundamental theorem of arithmetic = fundamental building block of all positive integers). This was formulated precisely for the first time by Carl Friedrich Gauss in his Disquisitiones Arithmeticae (1801).

**Theorem 3.2. Gauss 1801** *Every even natural number greater than 1 can be written as the product of prime numbers. Given two such decompositions $a = p_1 * p_2 * \cdots * p_n = q_1 * q_2 * \cdots * q_m$, these can be resorted such that $n = m$ and, for all $i, p_i = q_i$.*

In other words: Each natural number other than 1 can be written as a product of prime numbers in precisely one way, if we ignore the order of the factors. The factors are therefore unique (the "expression as a product of factors" is unique)!

For example, $60 = 2*2*3*5 = 2^2*3*5$. And this — other than changing the order of the factors — is the only way in which the number 60 can be factorised.

If you allow numbers other than primes as factors, there are several ways of factorising integers and the *uniqueness* is lost:

$$60 = 1*60 = 2*30 = 4*15 = 5*12 = 6*10 = 2*3*10 = 2*5*6 = 3*4*5 = \cdots$$

The 1st fundamental theorem only appears to be obvious. We can construct numerous other sets of numbers[28] for which numbers in the set *cannot* be expressed uniquely as a product of the prime numbers of the set.

In order to make a mathematical statement, therefore, it is important to state not only the operation for which it is defined but also the basic set on which the operation is defined.

For more details on prime numbers (e.g. how "Fermat's Little Theorem" can be used to test extremely large numbers to determine whether they are prime), please refer to the article on prime numbers in this script.

[28] These sets are formed especially from the set of natural numbers. An example of this can be found in this script on page 17 at the end of chapter 2.2

## 3.4 Divisibility, modulus and remainder classes

If whole numbers are added, subtracted or multiplied, the result is always another whole number.

The division of two whole numbers does not always result in a whole number. For example, if we divide 158 by 10 the result is the decimal number 15.8, which is not a whole number!

If, however, we divide 158 by 2 the result 79 is a whole number. In number theory we express this by saying that 158 is *divisible* by 2 but not by 10. In general, we say:

**Definition 3.3.** *A whole number $n$ is* **divisible** *by a whole number $d$ if the quotient $n/d$ is a whole number $c$ such that $n = c * d$.*

$n$ is called a *multiple* of $d$, whereas $d$ is called a *divisor* or *factor* of $n$.

The mathematical notation for this is $d|n$ (read "$d$ divides $n$"). The notation $d \nmid n$ means that $d$ does not divide the number $n$.

In our example therefore: $10 \nmid 158$ but $2|158$.

### 3.4.1 The modulo operation – working with congruence

When we investigate divisibility, it is only the remainder of the division that is important. When dividing a number $n$ by $m$, we often use the following notation:

$$\frac{n}{m} = c + \frac{r}{m},$$

where $c$ is a whole number and $r$ is a number with the values $0, 1, \cdots, m-1$. This notation is called division with remainder, whereby $c$ is called the whole-number "quotient" and $r$ is the "remainder" of the division.

**Example:**

$$\frac{19}{7} = 2 + \frac{5}{7} \quad (m = 7,\ c = 2,\ r = 5)$$

What do the numbers $5, 12, 19, 26, \cdots$ have in common for division by 7? The remainder is always $r = 5$. For division by 7, only the following remainders are possible:

$$r = 0, 1, 2, \cdots, 6.$$

The numbers that result in the same remainder $r$ when divided by 7 are combined to form the "remainder class $r$ modulo 7". Two numbers $a$ and $b$ belonging to the same remainder class modulo 7 are said to be "congruent modulo 7". Or in general:

**Definition 3.4.** *The* **remainder class r modulo m** *is the set of all whole numbers $a$ that have the same remainder $r$ when divided by $m$.*

**Examples:**

Remainder class 0 modulo 4 =
$\{x|x = 4 * n;\ n \in \mathbb{N}\} = \{\dots, -16, -12, -8, -4, 0, 4, 8, 12, 16, \dots\}$

Remainder class 3 modulo 4 =
$\{x|x = 4 * n + 3;\ n \in \mathbb{N}\} = \{\dots, -13, -9, -5, -1, 3, 7, 11, 15, \dots\}$

As only the remainders $0, 1, 2, \cdots, m-1$ are possible for division modulo $m$, modular arithmetic works with finite sets. For each modulo $m$ there are precisely $m$ remainder classes.

**Definition 3.5.** *Two numbers $a, b \in \mathbb{N}$ are said to be congruent modulo $m \in \mathbb{N}$ if and only if they have the same remainder when divided by $m$.*

We write: $a \equiv b \pmod{m}$ (read $a$ is congruent $b$ modulo $m$), which means that $a$ and $b$ belong to the same remainder class. The modulo is therefore the divisor. This notation was introduced by Gauss. Although the divisor is usually positive, $a$ and $b$ can also be any whole numbers.

**Examples:**

$19 \equiv 12 \pmod{7}$, because the remainders are equal: $19/7 = 2$ remainder 5 and $12/7 = 1$ remainder 5.

$23103 \equiv 0 \pmod{453}$, because $23103/453 = 51$ remainder 0 and $0/453 = 0$ remainder 0.

**Theorem 3.3.** *$a \equiv b \pmod{m}$ if and only if, the difference $(a - b)$ is divisible by $m$, i.e. if $q \in \mathbb{Z}$ exists with $(a - b) = q * m$.*

These two statements are therefore equivalent.

Therefore: If $m$ divides the difference, there exists a whole number $q$ such that: $a = b + q * m$. As an alternative to the congruence notation, we can also use the divisibility notation: $m|(a - b)$.

**Example of equivalent statements:**
$35 \equiv 11 \pmod{3} \iff 35 - 11 \equiv 0 \pmod{3}$, where $35 - 11 = 24$ is divisible by 3 without remainder while $35 : 3$ and $11 : 3$ leave the remainder 2.

**Comment:**
The above equivalence does not apply to the sum $(a + b)$!

**Example:**
$11 \equiv 2 \pmod{3}$, therefore $11 - 2 \equiv 9 \equiv 0 \pmod{3}$; but $11 + 2 = 13$ is not divisible by 3. The statement in theorem 3.3 does not even apply to sums in one direction. It is correct for sums only if the remainder is 0 and only in the following direction: if a divisor divides both summands with no remainder, it also divides the sum with no remainder.

We can apply the above equivalence in theorem 3.3 if we need a quick and easy method of determining whether large numbers are divisible by a certain number.

**Example:**
Is $69,993$ divisible by 7?

The number can be written in the form of a difference in which it is clear that each operand is divisible by 7: $69,993 = 70,0007$. Therefore, the difference is also divisible by 7.

Although these considerations and definitions may seem to be rather theoretical, we are so familiar with them in everyday life that we no longer think about the formal procedure. For example, the 24 hours on a clock are represented by the numbers $1, 2, \cdots, 12$. We obtain the hours after 12 noon as the remainder of a division by 12 and know immediately that 2 o'clock in the afternoon is the same as 14.00.

This "modular" arithmetic (based on division remainders) forms the basis of asymmetric encryption procedures. Cryptographic calculations are therefore not based on real numbers, as the calculations you performed at school, but rather on character strings with a limited length, in other words on positive whole numbers that cannot exceed a certain value. This is one of the reasons why we choose a large number $m$ and "calculate modulo $m$". That is, we ignore whole-number multiples of $m$ and, rather than working with a number, we only work with the remainder when this number is divided by $m$. The result is that all results are in the range 0 to $m - 1$.

## 3.5 Calculations with finite sets

### 3.5.1 Laws of modular calculations

From algebra theorems it follows that essential parts of the conventional calculation rules are kept when we proceed to modular calculations over a basic set $\mathbb{Z}$. For example, addition remains commutative. The same goes for multiplication modulo $m$. The result of a division[29] is not a fraction but rather a whole number between 0 and $m - 1$.

The known laws apply:

1. **Associative law:**
   $((a + b) + c) \pmod m \equiv (a + (b + c)) \pmod m$.
   $((a * b) * c) \pmod m \equiv (a * (b * c)) \pmod m$.

2. **Commutative law:**
   $(a + b) \pmod m \equiv (b + a) \pmod m$.
   $(a * b) \pmod m \equiv (b * a) \pmod m$.

The associative law and commutative law apply to both addition and multiplication.

3. **Distributive law:**
   $(a * (b + c)) \pmod m \equiv (a * b + a * c) \pmod m$.

4. **Reducibility:**
   $(a + b) \pmod m \equiv (a \pmod m + b \pmod m) \pmod m$.
   $(a * b) \pmod m \equiv (a \pmod m * b \pmod m) \pmod m$.

[29] When dividing modulo $m$ we cannot use every number because some numbers have the same property as zero. See footnote 33 in chapter 3.6.1.

The order in which order the modulo operation is performed is irrelevant.

5. **Existence of an identity (neutral element):**
$(a + 0) \pmod{m} \equiv (0 + a) \pmod{m} \equiv a \pmod{m}$.
$(a * 1) \pmod{m} \equiv (1 * a) \pmod{m} \equiv a \pmod{m}$.

6. **Existence of an inverse element:**
For all whole numbers $a$ and $m$ there exists a whole number $-a$ such that:
$(a + (-a)) \pmod{m} \equiv 0 \pmod{m}$ (additive inverse).
For each $a$ ($a \not\equiv 0 \pmod{p}$ ) where $p$ is prime there exists a whole number $a^{-1}$, such that:
$(a * a^{-1}) \pmod{p} \equiv 1 \pmod{p}$ (multiplicative inverse).

7. **Closeness**[30]:
$a, b \in G \Longrightarrow (a + b) \in G$.
$a, b \in G \Longrightarrow (a * b) \in G$.

8. **Transitivity**:
$[a \equiv b \bmod m, \; b \equiv c \bmod m] \Longrightarrow [a \equiv c \bmod m]$.

### 3.5.2 Patterns and structures

In general mathematicians investigate “Structures”. They ask e.g. at $a * x \equiv b \bmod m$, which values $x$ can take for given values of $a, \; b, \; m$.

Especially the case is investigated, where the result $b$ of this operation is the neutral element. Then $x$ is the inverse of $a$ regarding this operation.

[30] The property of closeness is always defined in relation to an operation in a set. See Appendix B of this chapter.

*Seneca*[31]*:*
The way of theory is long, it is short and effective by examples.

## 3.6 Examples of modular calculations

As we have already seen:

For two natural numbers $a$ and $m$, $a$ mod $m$ denotes the remainder obtained when we divide $a$ by $m$. This means that $a \pmod{m}$ is always a number between 0 and $m-1$.

For example, $1 \equiv 6 \equiv 41 \equiv 1 \pmod{5}$ because the remainder is always 1. Another example is: $2000 \equiv 0 \pmod{4}$ because 4 divides 2000 with no remainder.

Modular arithmetic only contains a limited quantity of non-negative numbers. The number of these is specified by a modulus $m$. If the modulo is $m = 5$, then only the 5 numbers in the set $\{0, 1, 2, 3, 4\}$ are used.

A calculation result larger than 4 is then reduced "modulo 5". In other words, it is the remainder when the result is divided by 5. For example, $2 * 4 \equiv 8 \equiv 3 \pmod{5}$ because 3 is the remainder when we divide 8 by 5.

### 3.6.1 Addition and multiplication

The following shows

- the addition table[32] $\pmod{5}$ and
- the multiplication tables[33] for mod 5 and mod 6.

#### Example of an addition table

The result when we add 3 and 4 $\pmod{5}$ is calculated as follows: Calculate $3 + 4 = 7$ and keep subtracting 5 from the result until the result is less than the modulo: $7 - 5 = 2$. Therefore: $3 + 4 \equiv 2 \pmod{5}$.

---

[31] Lucius Annaeus Seneca, philosophical writer and poet, 4 B. C. – 65 A. D.

[32] Comment on subtraction modulo 5:
$2 - 4 \equiv -2 \equiv 3$ mod 5.
It is therefore not true modulo 5 that $-2 = 2$ (see also Appendix C of this chapter).

[33] Comment on division modulo 6:
Due to the special role of zero as the identity for addition, division by zero is not permitted:
for all $a$ it is $a * 0 = 0$, because $a * 0 = a * (0 + 0) = a * 0 + a * 0$. Obviously 0 has no inverse regarding the multiplication, because if there would be one, it must be $0 = 0 * 0^{-1} = 1$. Also see footnote 29.

Addition table modulo 5:

| + | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| 0 | 0 | 1 | 2 | 3 | 4 |
| 1 | 1 | 2 | 3 | 4 | 0 |
| 2 | 2 | 3 | 4 | 0 | 1 |
| 3 | 3 | 4 | 0 | 1 | 2 |
| 4 | 4 | 0 | 1 | 2 | 3 |

**Example of an multiplication table:**

The result of the multiplication $4 * 4 \pmod 5$ is calculated as follows: $4 * 4 = 16$ and subtract 5 until the result is less than the modulus.

$$16 - 5 = 11;\ 11 - 5 = 6;\ 6 - 5 = 1.$$

The table directly shows that $4 * 4 \equiv 1 \pmod 5$ because $16 : 5 = 3$ remainder 1. Multiplication is defined on the set $\mathbb{Z}$ excluding 0.

Multiplication table modulo 5:

| * | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 1 | 1 | 2 | 3 | 4 |
| 2 | 2 | **4** | **1** | 3 |
| 3 | 3 | **1** | **4** | 2 |
| 4 | 4 | 3 | 2 | 1 |

### 3.6.2 Additive and multiplicative inverses

You can use the tables to read the inverses for each number in relation to addition and multiplication.

The inverse of a number is the number that gives the result 0 when the two numbers are added and 1 when they are multiplied. Thus, the inverse of 4 for addition mod 5 is 1 and the inverse of 4 for multiplication mod 5 is 4 itself, because

$$\begin{aligned} 4 + 1 &= \ \ 5 \equiv 0 \pmod 5; \\ 4 * 4 &= 16 \equiv 1 \pmod 5. \end{aligned}$$

The inverse of 1 for multiplication mod 5 is 1, while the inverse modulo 5 of 2 is 3 and, since multiplication is commutative, the inverse of 3 is again 2.

If we take a random number and add or multiply another number (here 4) and then add[34] or multiply the corresponding inverse (1 or 4) to the interim result (1 or 3), then the end result is the same as the initial value.

**Example:**

$$\begin{aligned} 2 + 4 \equiv 6 \equiv 1 \pmod 5; &\quad 1 + 1 \equiv 2 \equiv 2 \pmod 5, \\ 2 * 4 \equiv 8 \equiv 3 \pmod 5; &\quad 3 * 4 \equiv 12 \equiv 2 \pmod 5. \end{aligned}$$

[34] In general $x + y + (-y) \equiv x \pmod m$ $[(-y) =$ additive inverse of $y \pmod m]$.

In the set $\mathbb{Z}_5 = \{0, 1, 2, 3, 4\}$ for the addition and in the set $\mathbb{Z}_5^*$ for the multiplication, all numbers have a **unique** inverse modulo 5.

In the case of modular addition, this is true for every modulo (not just for 5).

This is not the case, however, for modular multiplication.

**Theorem 3.4.** *A natural number a from the set* $\{1, \cdots, m-1\}$ *has one inverse if and only if it and the modulo m are co-prime*[35]*, in other words if a and m have no common prime factors.*

Since $m = 5$ is prime, the numbers 1 to 4 are relatively prime to 5 and **each** of these numbers has a multiplicative inverse in mod 5.

A counterexample shows the multiplication table for mod 6 (since the modulus $m = 6$ is not prime, not all elements from $\mathbb{Z}_6 \setminus \{0\}$ are relatively prime to 6):

| Multiplication table modulo 6: | * | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| | 1 | 1 | 2 | 3 | 4 | 5 |
| | 2 | 2 | **4** | **0** | **2** | 4 |
| | 3 | 3 | **0** | **3** | **0** | 3 |
| | 4 | 4 | **2** | **0** | **4** | 2 |
| | 5 | 5 | 4 | 3 | 2 | 1 |

In addition to 0, the numbers 2, 3 and 4 also have no unique inverse (we can also say they have **no** inverse, because the elementary property of an inverse is uniqueness).

The numbers 2, 3 and 4 have the factor 2 or 3 in common with the modulus 6. Only the numbers 1 and 5, which are relatively prime to 6, have multiplicative inverses, namely themselves.

The number of numbers that are relatively prime to the modulus $m$ is the same as the number of numbers that have a multiplicative inverse (see the Euler function $J(m)$ below).

For the two moduli 5 and 6 used in the multiplication tables, this means: the modulus 5 is a prime number itself. In mod 5, therefore, there are exactly $J(5) = 5 - 1 = 4$ numbers that are relatively prime to the modulus, that is all numbers from 1 to 4.

Since 6 is not a prime number, we write it as a product of its factors: $6 = 2 * 3$. In mod 6, therefore, there are exactly $J(6) = (2-1)*(3-1) = 1*2 = 2$ numbers that have a multiplicative inverse, that is 1 and 5.

Although it may seem difficult to calculate the table of multiplicative inverses for large moduli (this only applies to the areas of the table shaded dark grey), we can use Fermat's Little Theorem

---

[35] Two whole numbers $a$ and $b$ are co-prime if and only if $\gcd(a, b) = 1$.
If $p$ is prime and $a$ is a random whole number that is not a multiple of $p$, then $p$ and a are co-prime.
Further name to the topic co-prime (with $a_i \in \mathbb{Z}, i = 1, \cdots, n$):

1. $a_1, a_2, \cdots, a_n$ are *relatively prime* , if $\gcd(a_1, \cdots, a_n) = 1$.
2. An even stronger request for more than two numbers is :
$a_1, \cdots, a_n$ are *in pairs relatively prime*, if for all $i = 1, \cdots, n$ and $j = 1, \cdots, n$ with $i \neq j$: $\gcd(a_i, a_j) = 1$.

Example: $2, 3, 6$ are relatively prime, because $\gcd(2, 3, 6) = 1$. They are not in pairs relatively prime, because $\gcd(2, 6) = 2 > 1$.

to create a simple algorithm for this [Pfleeger1997, p. 80]. Quicker algorithms are described, for instance, in [Knuth1998][36].

Cryptographically not only the unique nature of the inverse is important, but also that the set of possible values has been exhausted.

**Theorem 3.5.** *For $a, i \in \{1, \cdots, m-1\}$ with* $\gcd(a, m) = 1$)*, then takes the product* $a*i \bmod m$ *for a certain number $a$ all values from $\{1, \cdots, m-1\}$ (exhaustive permutation of the length $m-1$)*[37].

The following three examples[38] illustrate the properties of multiplicative inverses.

In the multiplication table mod 17, the following was calculated for $i = 1, 2, \cdots, 18$:

$(5*i)/17 = a$ remainder $r$ and high-lighted $5*i \equiv 1 \pmod{17}$,

$(6*i)/17 = a$ remainder $r$ and high-lighted $6*i \equiv 1 \pmod{17}$.

We need to **find** the $i$ for which the product remainder $a*i$ modulo 17 with $a = 5$ or $a = 6$ has the value 1.

**Table 1: Multiplication table modulo 17 (for $a = 5$ and $a = 6$)**

| i | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $5*i$ | 5 | 10 | 15 | 20 | 25 | 30 | 35 | 40 | 45 | 50 | 55 | 60 | 65 | 70 | 75 | 80 | 85 | 90 |
| remainder | 5 | 10 | 15 | 3 | 8 | 13 | **1** | 6 | 11 | 16 | 4 | 9 | 14 | 2 | 7 | 12 | 0 | 5 |
| $6*i$ | 6 | 12 | 18 | 24 | 30 | 36 | 42 | 48 | 54 | 60 | 66 | 72 | 78 | 84 | 90 | 96 | 102 | 108 |
| remainder | 6 | 12 | **1** | 7 | 13 | 2 | 8 | 14 | 3 | 9 | 15 | 4 | 10 | 16 | 5 | 11 | 0 | 6 |

Between $i = 1, \cdots, m$, all values between $0, \cdots, m-1$ occur for the remainders, because both 5 and 6 are also relatively prime to the modulus $m = 17$.

**The multiplicative inverse of 5 (mod 17) is 7, while the inverse of 6 (mod 17) is 3.**

**Table 2: Multiplication table modulo 13 (for $a = 5$ and $a = 6$)**

| i | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $5*i$ | 5 | 10 | 15 | 20 | 25 | 30 | 35 | 40 | 45 | 50 | 55 | 60 | 65 | 70 | 75 | 80 | 85 | 90 |
| remainder | 5 | 10 | 2 | 7 | 12 | 4 | 9 | **1** | 6 | 11 | 3 | 8 | 0 | 5 | 10 | 2 | 7 | 12 |
| $6*i$ | 6 | 12 | 18 | 24 | 30 | 36 | 42 | 48 | 54 | 60 | 66 | 72 | 78 | 84 | 90 | 96 | 102 | 108 |
| remainder | 6 | 12 | 5 | 11 | 4 | 10 | 3 | 9 | 2 | 8 | **1** | 7 | 0 | 6 | 12 | 5 | 11 | 4 |

[36] Using Euclid's extended theorem (extended gcd), we can calculate the multiplicative inverse and determine whether numbers have an inverse (see appendix A of this chapter). Alternatively, we can also use the primitive roots.

[37] See also theorem 3.14 in chapter 3.9 Multiplicative order and primitive roots.

[38] See Appendix D of this chapter for the source code to compute the tables using Mathematica and Pari-GP.

Between $i = 1, \cdots, m$, all values between $0, \cdots, m-1$ occur for the remainders, because both 5 and 6 are relatively prime to the modulus $m = 13$.

**The multiplicative inverse of 5 (mod 13) is 8, while the inverse of 6 (mod 13) is 11.**

The following table contains an example, where the modulus $m$ and the number $a = 6$ are *not* relatively prime.

**Table 3: Multiplication table modulo 12 (for $a = 5$ and $a = 6$)**

| i | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $5 * i$ | 5 | 10 | 15 | 20 | 25 | 30 | 35 | 40 | 45 | 50 | 55 | 60 | 65 | 70 | 75 | 80 | 85 | 90 |
| remainder | 5 | 10 | 3 | 8 | **1** | 6 | 11 | 4 | 9 | 2 | 7 | 0 | 5 | 10 | 3 | 8 | 1 | 6 |
| $6 * i$ | 6 | 12 | 18 | 24 | 30 | 36 | 42 | 48 | 54 | 60 | 66 | 72 | 78 | 84 | 90 | 96 | 102 | 108 |
| remainder | 6 | 0 | 6 | 0 | 6 | 0 | 6 | 0 | 6 | 0 | 6 | 0 | 6 | 0 | 6 | 0 | 6 | 0 |

We have calculated $(5 * i) \pmod{12}$ and $(6 * i) \pmod{12}$. Between $i = 1, \cdots, m$, not all values between $0, \cdots, m-1$ occur and 6 does not have an inverse mod 12, because 6 and the modulus $m = 12$ are not co-prime.

**The multiplicative inverse of 5 (mod 12) is 5. The number 6 has no inverse (mod 12).**

### 3.6.3 Raising to the power

In modular arithmetic, raising to the power is defined as repeated multiplication – as usual except that multiplication is now slightly different. We can even apply the usual rules, such as:

$$\begin{aligned} a^{b+c} &= a^b * a^c, \\ (a^b)^c &= a^{b*c} = a^{c*b} = (a^c)^b. \end{aligned}$$

Modular powers work in the same way as modular addition and modular multiplication:

$$3^2 \equiv 9 \equiv 4 \pmod 5.$$

Even consecutive powers work in the same way:

**Example 1:**

$$(4^3)^2 \equiv 64^2 \equiv 4096 \equiv 1 \pmod 5.$$

(1) We can speed up[39] the calculation by reducing the **interim results** modulo 5 but we need to take care because not everything will then work in the same way as

[39] The time required to calculate the multiplication of two numbers normally depends on the length of the numbers. We can observe this if we use the school method to calculate, for instance, $474 * 228$. The time required increases in a quadratic square manner , because we need to multiply $3 * 3$ numbers. The numbers become considerably smaller if we reduce the interim result.

in standard arithmetic.

$$\begin{aligned}(4^3)^2 &\equiv (4^3 \pmod 5)^2 \pmod 5\\ &\equiv (64 \pmod 5)^2 \pmod 5\\ &\equiv 4^2 \pmod 5\\ &\equiv 16 \equiv 1 \pmod 5.\end{aligned}$$

(2) In standard arithmetic, consecutive powers can be reduced to a single power by multiplying the exponents:

$$(4^3)^2 = 4^{3*2} = 4^6 = 4096.$$

This is not quite as simple in modular arithmetic because this would give:

$$(4^3)^2 \equiv 4^{3*2 \pmod 5} \equiv 4^{6 \pmod 5} \equiv 4^1 \equiv 4 \pmod 5.$$

But as we saw above, the correct result is 1!!

(3) Therefore, the rule is slightly different for consecutive powers in modular arithmetic: we do not multiply the exponents in $\pmod m$ but rather in $\pmod{J(m)}$.

Using $J(5) = 4$ gives:

$$(4^3)^2 \equiv 4^{3*2 \pmod{J(5)}} \equiv 4^{6 \bmod 4} \equiv 4^2 \equiv 16 \equiv 1 \pmod 5.$$

This delivers the correct result.

**Theorem 3.6.** $(a^b)^c \equiv a^{b*c \pmod{J(m)}} \pmod m$.

**Example 2:**

$$3^{28} \equiv 3^{4*7} \equiv 3^{4*7 \pmod{10}} \equiv 3^8 \equiv 6561 \equiv 5 \pmod{11}.$$

### 3.6.4 Fast calculation of high powers

RSA encryption and decryption[40] entails calculating high powers modulo $m$. For example, the calculation $(100^5) \pmod 3$ exceeds the 32-bit long integer number range provided we calculate $a^n$ by actually multiplying a with itself $n$ times in line with the definition. In the case of extremely large numbers, even a fast computer chip would take longer than the age of the universe to calculate a single exponential. Luckily, there is an extremely effective shortcut for calculating exponentials (but not for calculating logarithms).

If the expression is divided differently using the rules of modular arithmetic, then the calculation does not even exceed the 16-bit short integer number range:

$$(a^5) \equiv (((a^2 \pmod m))^2 \pmod m)) * a) \pmod m.$$

[40] See chapter 3.10 Proof of the RSA procedure with Euler-Fermat and chapter 3.13 The RSA procedure with actual numbers.

We can generalise this by representing the exponent as a binary number. For example, the naive method would require 36 multiplications in order to calculate $a^n$ for $n = 37$. However, if we write $n$ in the binary representation as $100101 = 1 * 2^5 + 1 * 2^2 + 1 * 2^0$, then we can rewrite the expression as: $a^{37} = a^{2^5+2^2+2^0} = a^{2^5} * a^{2^2} * a^1$

**Example 3:** $87^{43} \pmod{103}$.

Since $43 = 32 + 8 + 2 + 1$ , 103 is prime, $43 < J(103)$

and the squares (mod 103) can be calculated beforehand

$$
\begin{aligned}
87^2 &\equiv 50 \pmod{103}, \\
87^4 &\equiv 50^2 \equiv 28 \pmod{103}, \\
87^8 &\equiv 28^2 \equiv 63 \pmod{103}, \\
87^{16} &\equiv 63^2 \equiv 55 \pmod{103}, \\
87^{32} &\equiv 55^2 \equiv 38 \pmod{103}.
\end{aligned}
$$

we have[41]:

$$
\begin{aligned}
87^{43} &\equiv 87^{32+8+2+1} \pmod{103} \\
&\equiv 87^{32} * 87^8 * 87^2 * 87 \pmod{103} \\
&\equiv 38 * 63 * 50 * 87 \equiv 85 \pmod{103}.
\end{aligned}
$$

The powers $(a^2)^k$ can be determined easily by means of repeated squaring. As long as $a$ does not change, a computer can calculate them beforehand and – if enough memory is available – save them. In order to then find $a^n$ in each individual case, it now only needs to multiply those $(a^2)^k$ for which there is a one in the k-th position of the binary representation of $n$. The typical effort is then reduced from $2^{600}$ to $2 * 600$ multiplications! This frequently used algorithm is called "Square and Multiply".

### 3.6.5 Roots and logarithms

The inverses of the powers are also defined. The roots and logarithms are again whole numbers. Yet in contrast to the usual situation, they are not only difficult to calculate but, in the case of large numbers, cannot be calculated at all within a reasonable space of time.

Let us take the equation $a \equiv b^c \pmod{m}$.

**a) Taking the logarithm (determining $c$) – Discrete logarithm problem :**

If we know $a$ and $b$ of the three numbers $a$, $b$ and $c$ that meet this equation, then every known method of finding $c$ is approximately just as time-consuming as trying out all $m$ possible values for $c$ one after the other. For a typical $m$ of the order of magnitude of $10^{180}$ for 600-digit binary numbers, this is a hopeless task. More precisely, for suitably

[41] See Appendix D of this chapter for source code implementing the square and multiply method in Mathematica and Pari-GP, which can be used to reproduce the calculations above.

large numbers $m$, the time required according to current knowledge is proportional to $\exp\left(C * (\log m[\log\log m]^2)^{1/3}\right)$ with a constant $C > 1$.

**b) Calculating the root (determining $b$):**

The situation is similar if $b$ is the unknown variable and we know the values of $a$ and $c$: If we know the Euler function of $m$, $J(m)$, then we can easily[42] calculate $d$ with $c * d \equiv 1 \pmod{J(m)}$ and use theorem 3.6 to obtain:

$$a^d \equiv (b^c)^d \equiv b^{c*d} \equiv b^{c*d \ (mod\ J(m))} \equiv b^1 \equiv b \pmod{m}$$

the *c-th root* $b$ of $a$.

If $J(m)$ cannot be determined[43], it is difficult to calculate the $c$-th root. This forms the basis for the security assumption used by the RSA encryption system (see Sub-section 4.3.1: the RSA procedure and Sub-section 3.10 Proof of the RSA procedure).

The time required for inverting addition and multiplication, on the other hand, is simply proportional to $\log m$ or $(\log m)^2$. Powers (for a number $x$ calculate $x^a$ with $a$ fixed) and exponents (for a number $x$ calculate $a^x$ with $a$ fixed) are therefore typical one way functions (See Overview of the one way functions in this Script and article).

## 3.7 Groups and modular arithmetic in $\mathbb{Z}_n$ and $\mathbb{Z}_n^*$

Mathematical "*groups*" play a decisive role in number theory and cryptography. We only talk of groups if, for a defined set and a defined relation (an operation such as addition or multiplication), the following properties are fulfilled:

- The set is closed
- A neutral element exists
- An inverse element exists
- The associative law applies.

The abbreviated mathematical notation is $(G, +)$ or $(G, *)$.

**Definition 3.6.** *$\mathbb{Z}_n$:*

$$\mathbb{Z}_n \textit{ comprises all numbers from } 0 \textit{ to } n-1: \ \mathbb{Z}_n = \{0, 1, 2, \cdots, n-2, n-1\}.$$

$\mathbb{Z}_n$ is an often used finite group of the natural numbers. It is sometimes also called the *remainder set $R$* modulo $n$.

For example, 32-bit computers (standard PCs) only directly work with whole numbers in a finite set, that is the value range $0, 1, 2, \cdots, 2^{32} - 1$.

This value range is equivalent to the set $\mathbb{Z}_{2^{32}}$.

[42] See Appendix A of this chapter: the greatest common divisor (gcd) of whole numbers.

[43] According to the first fundamental theorem of number theory and theorem 3.11, we determine $J(m)$ by reducing $m$ to prime factors.

### 3.7.1 Addition in a group

If we define the operation mod+ on such a set where

$$a \bmod + \ b := (a + b) \pmod n,$$

then the set $\mathbb{Z}_n$ together with the relation mod+ is a group because the following properties of a group are valid for all elements in $\mathbb{Z}_n$:

- $a \bmod + \ b$ is an element of $\mathbb{Z}_n$ (the set is closed),
- $(a \bmod + \ b) \bmod + \ c \equiv a \bmod + \ (b \bmod + \ c)$ (mod+ is associative),
- the neutral element is 0.
- each element $a \in \mathbb{Z}_n$ has an inverse for this operation, namely $n - a$ (because $a \bmod + \ (n - a) \equiv a + (n - a) \pmod n \equiv n \equiv 0 \pmod n$).

Since the operation is commutative, i.e. $(a \bmod + \ b) = (b \bmod + \ a)$, this structure is actually a "commutative group".

### 3.7.2 Multiplication in a group

If we define the operation mod* on the set $\mathbb{Z}_n$ where

$$a \bmod * \ b := (a * b) \pmod n,$$

then $\mathbb{Z}_n$ together with this operation is **usually not a group** because not all properties are fulfilled for each $n$.

**Examples:**

a) In $\mathbb{Z}_{15}$, for example, the element 5 does not have an inverse. That is to say, there is no $a$ with
$5 * a \equiv 1 \pmod{15}$. Each modulo product with 5 on this set gives $5, 10$ or 0.

b) In $\mathbb{Z}_{55} \setminus \{0\}$, for example, the elements 5 and 11 do not have multiplicative inverses. That is to say, there is no $a \in \mathbb{Z}_{55}$ such that $5 * a \equiv 1 \ ( \ mod \ 55 \ )$ and no $a$ such that $11 * a \equiv 1 \ ( \ mod \ 55 \ )$. This is because 5 and 11 are not relatively prime to 55. Each modulo product with 5 on this set gives $5, 10, 15, \ldots, 50$ or 0. Each modulo product with 11 on this set gives $11, 22, 33, 44$ or 0.

On the other hand, there are subsets of $\mathbb{Z}_n$ that form a group with the operation mod*. If we choose all elements in $\mathbb{Z}_n$ that are relatively prime to $n$, then this set forms a group with the operation mod*. We call this set $\mathbb{Z}_n^*$.

**Definition 3.7.** $\mathbb{Z}_n^*$ :

$$\mathbb{Z}_n^* = \{a \in \mathbb{Z}_n | \gcd(a, n) = 1\}.$$

$\mathbb{Z}_n^*$ is sometimes also called the reduced remainder set $R'$ modulo $n$.

**Example:** For $n = 10 = 2 * 5$ the following applies:

$$\text{full remainder set } R = \mathbb{Z}_n = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$$

$$\text{reduced remainder set } R' = \mathbb{Z}_n^* = \{1, 3, 7, 9\} \longrightarrow J(n) = 4.$$

**Comment:** $R'$ or $\mathbb{Z}_n^*$ is always a genuine subset of $R$ or $\mathbb{Z}_n$ because 0 is always an element of $R$ but never an element of $R'$. Since 1 and $n-1$ are always relatively prime to $n$, they are always elements of both sets.

If we select a random element in $\mathbb{Z}_n^*$ and multiply it by every other element in $\mathbb{Z}_n^*$, then the products[44] are all in $\mathbb{Z}_n^*$, and the results are also a unique permutation of the elements in $\mathbb{Z}_n^*$. Since 1 is always an element of $\mathbb{Z}_n^*$, there is a unique "partner" in this set such that the product is 1. In other words:

**Theorem 3.7.** *Each element in $\mathbb{Z}_n^*$ has a multiplicative inverse.*

Example for $a = 3$ modulo 10 with $\mathbb{Z}_n^* = \{1, 3, 7, 9\}$ :

$$\begin{aligned} 3 &\equiv 3 * 1 \pmod{10}, \\ 9 &\equiv 3 * 3 \pmod{10}, \\ 1 &\equiv 3 * 7 \pmod{10}, \\ 7 &\equiv 3 * 9 \pmod{10}. \end{aligned}$$

The unique inverse is an essential condition for cryptography (see Section 3.10: Proof of the RSA procedure with Euler-Fermat).

[44] This is due to the fact that $\mathbb{Z}_n^*$ is closed with respect to the multiplication and due to the gcd property: $[a, b \in \mathbb{Z}_n^*] \Rightarrow [((a * b) \pmod n) \in \mathbb{Z}_n^*]$, exactly:
$[a, b \in \mathbb{Z}_n^*] \Rightarrow [\gcd(a, n) = 1, \gcd(b, n) = 1] \Rightarrow [\gcd(a * b, n) = 1] \Rightarrow [((a * b) \pmod n) \in \mathbb{Z}_n^*]$.

*Eric Berne*[45]*:*
Mathematical game theory postulates players who respond rationally. Transactional game theory, on the other hand, deals with games that are not rational, perhaps even **irrational and thereby closer to reality**.

## 3.8 Euler function, Fermat's little theorem and Euler-Fermat

### 3.8.1 Patterns and structures

As mathematicians investigate the structure $a * x \equiv b \bmod m$ (see chapter 3.5.2), so they are interested in the structure $x^a \equiv b \bmod m$.

Again here they are interested in the case, if $b = 1$ (value of the multiplicative inverse) and if $b = x$ (the function has a fixpoint).

### 3.8.2 The Euler function

Given $n$, the number of numbers from the set $\{1, \cdots, n-1\}$ that are relatively prime to $n$ is equal to the value of the Euler[46] function $J(n)$.

**Definition 3.8.** *The Euler function*[47] *$J(n)$ specifies the number of elements in $\mathbb{Z}_n^*$.*

$J(n)$ also specifies how many whole numbers have multiplicative inverses in mod $n$. $J(n)$ can be calculated if we know the prime factors of $n$.

**Theorem 3.8.** *For a prime number, the following is true: $J(p) = p - 1$.*

**Theorem 3.9.** *If $m$ is the product of two distinct primes, then:*

$$J(p * q) = (p-1) * (q-1) \quad \text{or} \quad J(p * q) = J(p) * J(q).$$

This case is important for the RSA procedure.

**Theorem 3.10.** *If $n = p_1 * p_2 * \cdots * p_k$ where $p_1$ to $p_k$ are distinct prime numbers (i.e. no factor occurs more than once), then the following is true (as a generalisation of theorem 3.9):*

$$J(n) = (p_1 - 1) * (p_2 - 1) * \cdots * (p_k - 1).$$

**Theorem 3.11.** *In general, the following is true for every prime number $p$ and every $n$ in $\mathbb{N}$:*

1. $J(p^n) = p^{n-1} * (p-1)$.

[45] Eric Berne, "Games People Play", rororo, (c) 1964, page 235.
[46] Leonhard Euler, Swiss mathematician, 15.4.1707 – 18.9.1783
[47] Is often described as the Euler phi function $\Phi(n)$.

2. *If $n = p_1^{e_1} * p_2^{e_2} * \cdots * p_k^{e_k}$, where $p_1$ to $p_k$ are distinct prime numbers, then:*
$J(n) = [(p_1^{e_1-1}) * (p_1 - 1)] * \cdots * [(p_k^{e_k-1}) * (p_k - 1)] = n * ([(p_1 - 1)/p_1] * \cdots * [(p_k - 1)/p_k])$.

**Examples:**

- $n = 70 = 2 * 5 * 7 \Longrightarrow$ using theorem 3.10: $J(n) = 1 \cdot 4 \cdot 6 = 24$.
- $n = 9 = 3^2 \Longrightarrow$ using theorem 3.11: $J(n) = 3^1 \cdot 2 = 6$, because $\mathbb{Z}_9^* = \{1, 2, 4, 5, 7, 8\}$.
- $n = 2,701,125 = 3^2 * 5^3 * 7^4 \Longrightarrow$ using theorem 3.11:
$$J(n) = [3^1 * 2] * [5^2 * 4] * [7^3 * 6] = 1,234,800.$$

### 3.8.3 The theorem of Euler-Fermat

In order to prove the RSA procedure, we need Fermat's theorem and its generalisation (Euler-Fermat theorem) – please also see chapter 2.5.

**Theorem 3.12. Fermat's Little Theorem**[48] *Let $p$ be a prime number and $a$ be a random whole number, then:*
$$a^p \equiv a \pmod{p}.$$

An alternative formulation of Fermat's Little Theorem is as follows: Let $p$ be a prime number and $a$ be a random whole number that is relatively prime to $p$, then:
$$a^{p-1} \equiv 1 \pmod{p}.$$

**Theorem 3.13. Euler-Fermat theorem (generalisation of Fermat's Little Theorem)** *For all elements $a$ in the group $\mathbb{Z}_n^*$ (i.e. $a$ and $n$ are natural numbers that are co-prime):*
$$a^{J(n)} \equiv 1 \pmod{n}.$$

This theorem states that if we raise a group element (here $a$) to the power of the order of the group (here $J(n)$), we always obtain the neutral element for multiplication (the number 1).

The 2nd formulation of Fermat's Little Theorem is derived directly from Euler's theorem if $n$ is a prime number.

If $n$ is the product of two prime numbers, we can - in certain cases - use Euler's theorem to calculate the result of a modular power very quickly. We have: $a^{(p-1)*(q-1)} \equiv 1 \pmod{pq}$.

**Examples for calculating a modular power:**

- With $2 = 1 * 2$ and $6 = 2 * 3$ where 2 and 3 are both prime; $J(6) = 2$ because only 1 and 5 are relatively prime to 6, we obtain the equation $5^2 \equiv 5^{J(6)} \equiv 1 \pmod{6}$, without having to calculate the power.
- With $792 = 22 * 36$ and $23 * 37 = 851$ where 23 and 37 are both prime, it follows that $31^{792} \equiv 31^{J(23*37)} \equiv 31^{J(851)} \equiv 1 \pmod{851}$.

[48] Pierre de Fermat, French mathematician, 17.8.1601 – 12.1.1665.

### 3.8.4 Calculation of the multiplicative inverse

Another interesting application is a special case of determining the multiplicative inverses using the Euler-Fermat theorem (multiplicative inverses are otherwise determined using the extended Euclidean algorithm).

**Example:**

Find the multiplicative inverse of 1579 modulo 7351.
According to Euler-Fermat: $a^{J(n)} = 1 \pmod n$ for all $a$ in $\mathbb{Z}_n^*$. If we divide both sides by $a$, we get: $a^{J(n)-1} \equiv a^{-1} \pmod n$. For the special case that the modulo is prime, we have $J(n) = p-1$. Therefore, the modular inverse is

$$a^{-1} = a^{J(n)-1} \equiv a^{(p-1)-1} \equiv a^{p-2} \pmod p.$$

For our example, this means:

> Since the modulus 7351 is prime, $p - 2 = 7349$.
> $1579^{-1} \equiv 1579^{7349} \pmod p$.

By cleverly breaking down the exponent, we can calculate this power relatively easily (see Section 3.6.4 Fast calculation of high powers):

> $7349 = 4096 + 2048 + 1024 + 128 + 32 + 16 + 4 + 1$
>
> $1579^{-1} \equiv 4716 \pmod{7351}$.

### 3.8.5 Fixpoints modulo 26

According to theorem 3.6, the arithmetic operations of modular expressions are performed in the exponents modulo $J(n)$ rather than modulo $n$[49].

In $a^{e*d} \equiv a^1 \pmod n$, if we wish to determine the inverses for the factor $e$ in the exponent, we need to calculate modulo $J(n)$.

**Example (with reference to the RSA algorithm):**
If we calculate modulo 26, which set can $e$ and $d$ come from?

Solution: we have $e * d \equiv 1 \pmod{J(26)}$.

> The reduced remainder set $R' = \mathbb{Z}_{26}^* = \{1, 3, 5, 7, 9, 11, 15, 17, 19, 21, 23, 25\}$ are the elements in $\mathbb{Z}_{26}$, which have a multiplicative inverse, that is which are relatively prime to 26.
>
> The reduced remainder set $R''$ contains only the elements of $R'$ that are relatively prime to $J(n) = 12 : R'' = \{1, 5, 7, 11\}$.
>
> For every $e$ in $R''$ there exists a $d$ in $R''$ such that $a \equiv (a^e)^d \pmod n$.

[49] For the following example, we will adopt the usual practice for the RSA procedure of using "$n$" rather than "$m$" to denote the modulus.

For every $e$ in $R''$, there exists therefore precisely one element (not necessarily different from $e$) such that $e * d \equiv 1 \pmod{J(26)}$.

For all $e$ that are relatively prime to $J(n)$ we could calculate $d$ as follows using the Euler-Fermat theorem: For $a^{J(n)} \equiv 1 \pmod{n}$ is the same as saying $a^{J(n)-1} \equiv a^{-1} \pmod{n}$. Therefore

$$d \equiv e^{-1} \pmod{J(n)} \equiv e^{J(J(n))-1} \pmod{J(n)}.$$

The problems of factorising $n = pq$ with $q \neq p$ and finding $J(n)$ have a similar degree of difficulty and if we find a solution for one of the two problems, we also have a solution for the other[50] (please compare requisition 3 in section 3.10.1).

[50] If we know the factors of $n = p * q$ with $p \neq q$, then $J(n) = (p-1) * (q-1) = n - (p+q) + 1$. Additionally the factors $p$ and $q$ are solutions of the quadratic equation $x^2 - (p+q)x + pq = 0$.
If only $n$ and $J(n)$ are known, then it is: $pq = n$ and $p + q = n - J(n) + 1$. So you get $p$ and $q$ by solving the equation

$$x^2 + (J(n) - n - 1)x + n = 0.$$

## 3.9 Multiplicative order and primitive roots

Mathematicians often ask, in which conditions the repeated application of an operation results in the neutral element (compare patterns and structures before).

For the $i$-times successive modular multiplication of a number $a$ with $i = 1, \cdots, m-1$ the product is the neutral element of the multiplication (1) if and only if $a$ and $m$ are relatively prime. The value of $i$, for which the product $a^i = 1$, is called the multiplicative order of $a$.

The multiplicative order and the primitive root are two useful constructs (concepts) in elementary number theory.

**Definition 3.9.** *The* **multiplicative order** $\mathrm{ord}_m(a)$ *of a whole number a (mod m) (where a and m are co-prime) is the smallest whole number e for which* $a^e \equiv 1 \pmod{m}$.

The following table shows that in a multiplicative group (here $\mathbb{Z}_{11}^*$) not all numbers necessarily have the same order. The orders in this case are $1, 2, 5$ and $10$ and we notice that:

1. The orders are all factors of 10.
2. The numbers $a = 2, 6, 7$ and $8$ have the order 10 - we say that these numbers have the **maximum order** in $\mathbb{Z}_{11}^*$.

**Example 1:**
The following table[51] shows the values $a^i \bmod 11$ for the exponents $i = 1, 2, \cdots, 10$ and for the bases $a = 1, 2, \cdots, 10$ as well as the resulting value $ord_{11}(a)$ for each $a$:

**Table 4: Values of $a^i \bmod 11, 1 \leq a, i < 11$ and according order of $a$ mod $m$:**

| | i=1 | i=2 | i=3 | i=4 | i=5 | i=6 | i=7 | i=8 | i=9 | i=10 | $ord_{11}(a)$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $a = 1$ | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| $a = 2$ | 2 | 4 | 8 | 5 | 10 | 9 | 7 | 3 | 6 | 1 | 10 |
| $a = 3$ | 3 | 9 | 5 | 4 | **1** | 3 | 9 | 5 | 4 | 1 | 5 |
| $a = 4$ | 4 | 5 | 9 | 3 | **1** | 4 | 5 | 9 | 3 | 1 | 5 |
| $a = 5$ | 5 | 3 | 4 | 9 | **1** | 5 | 3 | 4 | 9 | 1 | 5 |
| $a = 6$ | 6 | 3 | 7 | 9 | 10 | 5 | 8 | 4 | 2 | 1 | 10 |
| $a = 7$ | 7 | 5 | 2 | 3 | 10 | 4 | 6 | 9 | 8 | 1 | 10 |
| $a = 8$ | 8 | 9 | 6 | 4 | 10 | 3 | 2 | 5 | 7 | 1 | 10 |
| $a = 9$ | 9 | 4 | 3 | 5 | **1** | 9 | 4 | 3 | 5 | 1 | 5 |
| $a = 10$ | 10 | 1 | 10 | 1 | 10 | 1 | 10 | 1 | 10 | 1 | 2 |

The table shows, for example, that the order of 3 modulo 11 has the value 5.

**Definition 3.10.** *If a and m are co-prime and if* $ord_m(a) = J(m)$ *(i.e. a has maximum order), then we say that a is a* **primitive root** *of m.*

[51] See Appendix D of this chapter for the source code to generate the table using Mathematica and Pari-GP.

A number $a$ is not a primitive root for every modulo $m$. In the above table, only $a = 2, 6, 7$ and 8 is a primitive root with respect to mod 11 ($J(11) = 10$).

Using the primitive roots, we can clearly establish the conditions for which powers modulo $m$ have a unique inverse and the calculation in the exponents is manageable.

The following two tables show the multiplicative orders and primitive roots modulo 45 and modulo 46.

**Example 2:**
The following table[52] shows the values $a^i$ mod 45 for the exponents $i = 1, 2, \cdots, 12$ and for the bases $a = 1, 2, \cdots, 12$ as well as the resulting value $ord_{45}(a)$ for each $a$:

**Table 5: Values of** $a^i$ mod $45, 1 \leq a, i < 13$**:**

| $a \setminus i$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | $ord_{45}(a)$ | $J(45)$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 24 |
| 2 | 2 | 4 | 8 | 16 | 32 | 19 | 38 | 31 | 17 | 34 | 23 | 1 | 12 | 24 |
| 3 | 3 | 9 | 27 | 36 | 18 | 9 | 27 | 36 | 18 | 9 | 27 | 36 | — | 24 |
| 4 | 4 | 16 | 19 | 31 | 34 | 1 | 4 | 16 | 19 | 31 | 34 | 1 | 6 | 24 |
| 5 | 5 | 25 | 35 | 40 | 20 | 10 | 5 | 25 | 35 | 40 | 20 | 10 | — | 24 |
| 6 | 6 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | — | 24 |
| 7 | 7 | 4 | 28 | 16 | 22 | 19 | 43 | 31 | 37 | 34 | 13 | 1 | 12 | 24 |
| 8 | 8 | 19 | 17 | 1 | 8 | 19 | 17 | 1 | 8 | 19 | 17 | 1 | 4 | 24 |
| 9 | 9 | 36 | 9 | 36 | 9 | 36 | 9 | 36 | 9 | 36 | 9 | 36 | — | 24 |
| 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | — | 24 |
| 11 | 11 | 31 | 26 | 16 | 41 | 1 | 11 | 31 | 26 | 16 | 41 | 1 | 6 | 24 |
| 12 | 12 | 9 | 18 | 36 | 27 | 9 | 18 | 36 | 27 | 9 | 18 | 36 | — | 24 |

$J(45)$ is calculated using theorem 3.11: $J(45) = J(3^2 * 5) = 3^1 * 2 * 4 = 24$.

Since 45 is not a prime, there is no "multiplicative order" for all values of $a$ (e. g. for the numbers that are not relatively prime to $45 : 3, 5, 6, 9, 10, 12, \cdots$, because $45 = 3^2 * 5$).

**Example 3:**
Is 7 a primitive root modulo 45?
The requirement/condition $gcd(7, 45) = 1$ is fulfilled. The table 'values of $a^i$ mod 45' shows that the number 7 is not a primitive root of 45, because $ord_{45}(7) = 12 \neq 24 = J(45)$.

**Example 4:**
The following table[53] answers the question as to whether the number 7 is a primitive root of 46. The requirement/condition $gcd(7, 46) = 1$ is fulfilled.

[52]See Appendix D of this chapter for the source code to generate the table using Mathematica and Pari-GP.
[53]See Appendix D of this chapter for the source code to generate the table using Mathematica and Pari-GP.

**Table 6: Values of $a^i$ mod $46, 1 \leq a, i < 23$:**

| $a \backslash i$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | ord |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 2 | 2 | 4 | 8 | 16 | 32 | 18 | 36 | 26 | 6 | 12 | 24 | 2 | 4 | 8 | 16 | 32 | 18 | 36 | 26 | 6 | 12 | 24 | 2 | – |
| 3 | 3 | 9 | 27 | 35 | 13 | 39 | 25 | 29 | 41 | 31 | 1 | 3 | 9 | 27 | 35 | 13 | 39 | 25 | 29 | 41 | 31 | 1 | 3 | 11 |
| 4 | 4 | 16 | 18 | 26 | 12 | 2 | 8 | 32 | 36 | 6 | 24 | 4 | 16 | 18 | 26 | 12 | 2 | 8 | 32 | 36 | 6 | 24 | 4 | – |
| 5 | 5 | 25 | 33 | 27 | 43 | 31 | 17 | 39 | 11 | 9 | 45 | 41 | 21 | 13 | 19 | 3 | 15 | 29 | 7 | 35 | 37 | 1 | 5 | 22 |
| 6 | 6 | 36 | 32 | 8 | 2 | 12 | 26 | 18 | 16 | 4 | 24 | 6 | 36 | 32 | 8 | 2 | 12 | 26 | 18 | 16 | 4 | 24 | 6 | – |
| 7 | 7 | 3 | 21 | 9 | 17 | 27 | 5 | 35 | 15 | 13 | 45 | 39 | 43 | 25 | 37 | 29 | 19 | 41 | 11 | 31 | 33 | **1** | 7 | 22 |
| 8 | 8 | 18 | 6 | 2 | 16 | 36 | 12 | 4 | 32 | 26 | 24 | 8 | 18 | 6 | 2 | 16 | 36 | 12 | 4 | 32 | 26 | 24 | 8 | – |
| 9 | 9 | 35 | 39 | 29 | 31 | 3 | 27 | 13 | 25 | 41 | 1 | 9 | 35 | 39 | 29 | 31 | 3 | 27 | 13 | 25 | 41 | 1 | 9 | 11 |
| 10 | 10 | 8 | 34 | 18 | 42 | 6 | 14 | 2 | 20 | 16 | 22 | 36 | 38 | 12 | 28 | 4 | 40 | 32 | 44 | 26 | 30 | 24 | 10 | – |
| 11 | 11 | 29 | 43 | 13 | 5 | 9 | 7 | 31 | 19 | 25 | 45 | 35 | 17 | 3 | 33 | 41 | 37 | 39 | 15 | 27 | 21 | 1 | 11 | 22 |
| 12 | 12 | 6 | 26 | 36 | 18 | 32 | 16 | 8 | 4 | 2 | 24 | 12 | 6 | 26 | 36 | 18 | 32 | 16 | 8 | 4 | 2 | 24 | 12 | – |
| 13 | 13 | 31 | 35 | 41 | 27 | 29 | 9 | 25 | 3 | 39 | 1 | 13 | 31 | 35 | 41 | 27 | 29 | 9 | 25 | 3 | 39 | 1 | 13 | 11 |
| 14 | 14 | 12 | 30 | 6 | 38 | 26 | 42 | 36 | 44 | 18 | 22 | 32 | 34 | 16 | 40 | 8 | 20 | 4 | 10 | 2 | 28 | 24 | 14 | – |
| 15 | 15 | 41 | 17 | 25 | 7 | 13 | 11 | 27 | 37 | 3 | 45 | 31 | 5 | 29 | 21 | 39 | 33 | 35 | 19 | 9 | 43 | 1 | 15 | 22 |
| 16 | 16 | 26 | 2 | 32 | 6 | 4 | 18 | 12 | 8 | 36 | 24 | 16 | 26 | 2 | 32 | 6 | 4 | 18 | 12 | 8 | 36 | 24 | 16 | – |
| 17 | 17 | 13 | 37 | 31 | 21 | 35 | 43 | 41 | 7 | 27 | 45 | 29 | 33 | 9 | 15 | 25 | 11 | 3 | 5 | 39 | 19 | 1 | 17 | 22 |
| 18 | 18 | 2 | 36 | 4 | 26 | 8 | 6 | 16 | 12 | 32 | 24 | 18 | 2 | 36 | 4 | 26 | 8 | 6 | 16 | 12 | 32 | 24 | 18 | – |
| 19 | 19 | 39 | 5 | 3 | 11 | 25 | 15 | 9 | 33 | 29 | 45 | 27 | 7 | 41 | 43 | 35 | 21 | 31 | 37 | 13 | 17 | 1 | 19 | 22 |
| 20 | 20 | 32 | 42 | 12 | 10 | 16 | 44 | 6 | 28 | 8 | 22 | 26 | 14 | 4 | 34 | 36 | 30 | 2 | 40 | 18 | 38 | 24 | 20 | – |
| 21 | 21 | 27 | 15 | 39 | 37 | 41 | 33 | 3 | 17 | 35 | 45 | 25 | 19 | 31 | 7 | 9 | 5 | 13 | 43 | 29 | 11 | 1 | 21 | 22 |
| 22 | 22 | 24 | 22 | 24 | 22 | 24 | 22 | 24 | 22 | 24 | 22 | 24 | 22 | 24 | 22 | 24 | 22 | 24 | 22 | 24 | 22 | 24 | 22 | – |
| 23 | 23 | 23 | 23 | 23 | 23 | 23 | 23 | 23 | 23 | 23 | 23 | 23 | 23 | 23 | 23 | 23 | 23 | 23 | 23 | 23 | 23 | 23 | 23 | – |

$J(46)$ is calculated using theorem 3.9: $J(46) = J(2 * 23) = 1 * 22 = 22$. The number 7 is a primitive root of 46, because $ord_{46}(7) = 2 = J(46)$.

**Theorem 3.14.** [54,55] *For a modulus $n$ and a relative prime to $n$ the following holds:* $\{a^i(mod\ n) | i = 1, \ldots, J(n)\}$ *equals the multiplicative group* $Z_n^*$ *if and only if* $ord_n(a) = J(n)$.

[54] For prime moduli $p$ all $a$ with $0 < a < p$ are of order $J(p) = p - 1$. Compare table 5 for an example. In this case $a^i(\text{mod } n)$ goes through all the values $1, \ldots, p-1$. Exhausting all possible values of the set is an important cryptographic proposition (compare theorem 3.5). This determines a permutation $\pi(p-1)$.

[55] Table 6 demonstrates that for composite moduli $n$ not all $a$ are of maximal order $J(n)$. In this example only $5, 7, 11, 15, 17, 19$ and $21$ are of order 22.

## 3.10 Proof of the RSA procedure with Euler-Fermat

Using the Euler-Fermat theorem, we can "prove" the RSA[56,57] procedure in the group $\mathbb{Z}_n^*$.

### 3.10.1 Basic idea of public key cryptography

The basic idea behind public key cryptography is that all participants possess a different pair of keys ($P$ and $S$) and the public keys for all recipients are published. You can retrieve the public key $P$ for a recipient from a directory just as you would look up someone's phone number in the phone book. Furthermore, each recipient has a secret key $S$ that is needed in order to decrypt the message and that is not known to anyone else. If the sender wishes to send a message $M$, he encrypts it using the public key $P$ of the recipient before sending it:

The cipher text $C$ is determined as $C = E(P; M)$, where $E$ (encryption) is the encryption rule. The recipient uses his private key $S$ to decrypt the message with the decryption rule $D : M = D(S; C)$.

In order to ensure that this system works for every message $M$, the following four **requirements** must be met:

**1.** $D(S; E(P; M)) = M$ for every $M$ (invertibility) and $M$ takes "very many" of its possible values.

**2.** All $(S, P)$ pairs are different for all participants (i.e. lots of them are needed).

**3.** The time required to derive $S$ from $P$ is at least as high as the time required to decrypt $M$ with no knowledge of $S$.

**4.** Both $C$ and $M$ can be calculated relatively easily.

The 1st requirement is a general condition for all cryptographic encryption algorithms.

The 2nd requirement can easily be met because there is a "very" large number of prime numbers[58] and because this can be ensured by a central office that issues certificates.

---

[56] The RSA procedure is the most common asymmetric cryptography procedure. Developed in 1978 by Ronald Rivest, Adi Shamir and Leonard Adleman, it can be used both for signatures and for encryption. Cryptographers always associate this procedure with the abbreviation "RSA". However, please note that the abbreviation may be used in other fields to mean something completely different.

[57] In literature and movies not only classical methods are used (like the secret message solved by Sherlock Holmes in "The Adventure of the Dancing Men" by Arthur Conan Doyle ), but also modern schemes. Examples are

- the movie "House of Cards", 1992, where autistic children communicate via primes,
- the story "The Dialogue of the Sisters" by Dr. C. Elsner, 1999 (included in the CrypTool package as PDF file), where the sisters communicate confidentially using a variant of RSA.

[58] According to the prime number theorem of Legendre and Gauss there are approximately $n/\ln(n)$ prime numbers up to the number $n$. This means, for example, that there are $6.5 * 10^{74}$ prime numbers under $n = 2^{256}$ ($= 1.1 * 10^{77}$) and $3.2 * 10^{74}$ prime numbers under $n = 2^{255}$. Between $2^{255}$ and $2^{256}$ there are therefore $3.3 * 10^{74}$ prime numbers with precisely 256 bits. This large number is also the reason why we cannot simply save them all.

It is this last requirement that makes the procedure actually usable. This is because it is possible to calculate the powers in a linear amount of time (because there is a restriction on the length of the numbers).

Although Whitfield Diffie and Martin Hellman formulated the general method as early as 1976, the actual procedure that met all four requirements was only discovered later by Rivest, Shamir and Adleman.

### 3.10.2 How the RSA procedure works

The individual steps for implementing the RSA procedure can be described as follows (see [Eckert2003, p. 213 ff] and [Sedgewick1990, p. 338 ff]). Steps 1 to 3 constitute key generation, steps 4 and 5 are the encryption, and steps 6 and 7 are the decryption:

**1.** Select two distinct random prime numbers[59,60] $p$ and $q$ and calculate $n = p * q$[61]. The value $n$ is called the RSA modulus[62].

**2.** Select a random $e \in \{2, \cdots, n-1\}$ such that:
$e$ is relatively prime to $J(n) = (p-1) * (q-1)$. For example, we can select $e$ such that: $\max(p, q) < e < J(n) - 1$[63]. We can then "throw away" $p$ and $q$.

**3.** Select $d \in \{1, \cdots, n-1\}$ with $e * d \equiv 1 \pmod{J(n)}$, i.e. $d$ is the multiplicative inverse of $e$ modulo $J(n)$[64,65]. We can then "throw away" $J(n)$.

$\rightarrow (n, e)$ is the public key $P$.

$\rightarrow (n, d)$ is the secret key $S$ (only $d$ must be kept secret).

---

[59] Compaq introduced the so-called multi-prime method with high marketing effort in 2000. $n$ was the product of two big and one relative small prime: $n = o * p * q$. With theorem 3.10 we get: $J(n) = (o-1) * (p-1) * (q-1)$. This method did not assert itself yet.
One reason probably is, that Compaq claimed a patent on it. Generally there is less understanding in Europe and with the Open Source Initiative , that one can claim patents on algorithms. But there is really no understanding outside the U.S., that one can get a patent for a special case (3 factors) of an algorithm (RSA), although the patent for the general case is almost expired.

[60] If the two primes $p$ and $q$ are equal then $(m^e)^d \equiv m \bmod n$ is not true for all $m < n$ (although $e * d \equiv 1 \bmod J(n)$ is fulfilled). Example: If $n = 5^2$ then according to theorem 3.5 it is $J(n) = 5 * 4 = 20,\ e = 3,\ d = 7,\ e * d = 21 \equiv 1 \bmod J(n)$. But it is $(5^3)^7 \equiv 0 \bmod 25$.

[61] The GISA (German Information Security Agency) recommends, to choose the prime factors $p$ and $q$ almost the same, but not too close:

$$0.5 < |\log_2(p) - \log_2(q)| < 30.$$

They recommend to generate the primes independently and check that the restriction is fulfilled (see [GISA2002]).

[62] In CrypTool the RSA modulo is denoted with a capital "$N$" .

[63] The procedure also allows us to select $d$ freely and then calculate $e$. However, this has practical disadvantages. We usually want to be able to encrypt messages "quickly", which is why we choose a public exponent $e$ such that it has a short bit length compared to the modulus $n$ and as few binary ones as possible. A number often used is $65537 = 2^{16} + 1$, or in binary: $10 \cdots 0 \cdots 01$. We want to select the publicly known $e$ to be an advantageous value that allows the exponential calculation to be performed quickly during encryption. The prime numbers $3, 17$ and $65537$ have proved to be particularly practical for this purpose.

[64] For reasons of security, $d$ should not be too small.

[65] We start by determining either $d$ or $e$ depending on the implementation.

4. For encryption, the message represented as a (binary) number is divided into parts such that each part of the number is less than $n$.
5. Encryption of the plaintext (or the parts of it) $M \in \{1, \cdots, n-1\}$:

$$C = E((n,e); M) := M^e \pmod n.$$

6. For decryption, the cipher text represented as a binary number is divided into parts such that each part of the number is less than $n$.
7. Decryption of the cipher text (or the parts of it) $C \in \{1, \cdots, n-1\}$:

$$M = D((n,d); C) := C^d \pmod n.$$

The numbers $d, e$ and $n$ are usually extremely large (e. g. $d$ and $e$ 300 bits, $n$ 600 bits).

**Comment:**

The security of the RSA algorithm depends as with all public key methods on the difficulty to calculate the private key $d$ from the public key $(n, e)$.

Concrete for the RSA method does this mean:

1. it is hard to calculate $J(n)$ for big compounds $n$ and
2. it is hard to calculate the prime factors of big compounds $n$ (Factorisation!factorisation problem).

### 3.10.3 Proof of requirement 1 (invertibility)

For pairs of keys $(n, e)$ and $(n, d)$ that possess fixed properties in steps 1 to 3 of the RSA procedure, the following must be true for all $M < n$:

$$M \equiv (M^e)^d \pmod n \quad \text{with} \quad (M^e)^d = M^{e*d}.$$

This means that the deciphering algorithm above works correctly.

We therefore need to show that:

$$M^{e*d} \equiv M \pmod n.$$

We will show this in 3 steps (see [Beutelspacher1996, p. 131ff]).

**Step 1**:

In the first step we show that: $M^{e*d} \equiv M \pmod p$. This results from the requirements and from Euler-Fermat (theorem 3.13). Since $n = p * q$ and $J(p * q) = (p-1) * (q-1)$ and since $e$ and

$d$ are selected in such a way that $e * d \equiv 1 \pmod{J(n)}$, there is a whole number $k$ such that: $e * d = 1 + k * (p-1) * (q-1)$.

$$\begin{aligned} M^{e*d} &\equiv M^{1+k*J(n)} \equiv M * M^{k*J(n)} \equiv M * M^{k*(p-1)*(q-1)} \pmod{p} \\ &\equiv M * (M^{p-1})^{k*(q-1)} \pmod{p} \quad \text{based on little Fermat}: \ M^{p-1} \equiv 1 \pmod{p} \\ &\equiv M * (1)^{k*(q-1)} \pmod{p} \\ &\equiv M \pmod{p} \end{aligned}$$

The requirement for using the simplified Euler-Fermat theorem (theorem 3.12) was that $M$ and $p$ are relatively prime.

Since this is not true in general, we need to consider the case when $M$ and $p$ are not relatively prime. Since $p$ is a prime number, this implies that $p$ is a factor of $M$. But this means:

$$M \equiv 0 \pmod{p}.$$

If $p$ is a factor of $M$, then $p$ is also a factor of $M^{e*d}$. Therefore:

$$M^{e*d} \equiv 0 \pmod{p}.$$

Since $p$ is a factor of both $M$ and $Me * d$, it is also a factor of their difference:

$$(M^{e*d} - M) \equiv 0 \pmod{p}.$$

And therefore our conjecture is also true in this special case.

**Step 2**:

In exactly the same way we prove that: $M^{e*d} \equiv M \pmod{q}$.

**Step 3**:

We now combine the conjectures from (a) and (b) for $n = p * q$ to show that:

$$M^{e*d} \equiv M \pmod{n} \text{ for all } M < n.$$

From (a) and (b) we have $(M^{e*d} - M) \equiv 0 \pmod{p}$ and $(M^{e*d} - M) \equiv 0 \pmod{q}$. Therefore, $p$ and $q$ are both factors of the same number $z = (M^{e*d} - M)$. Since $p$ and $q$ are **distinct** prime numbers, their product must also be a factor of this number $z$. Thus:

$$(M^{e*d} - M) \equiv 0 \pmod{p * q} \quad \text{or} \quad M^{e*d} \equiv M \pmod{p * q} \quad \text{or} \quad M^{e*d} \equiv M \pmod{n}.$$

□

**1st comment:**

We can also condense the three steps if we use the theorem 3.13 (Euler-Fermat) - i.e. not the simplified theorem where $n = p$ and which corresponds to Fermat's Little Theorem:

$$(M^e)^d \equiv M^{e*d} \equiv M^{(p-1)(q-1)*k+1} \equiv (\underbrace{M^{(p-1)(q-1)}}_{\equiv M^{J(n)} \equiv 1 \pmod{n}})^k * M \equiv 1^k * M \equiv M \pmod{n}.$$

**2nd comment:**
When it comes to signing messages, we perform the same operations but first use the secret key $d$, followed by the public key $e$. The RSA procedure can also be used to create digital signatures, because:

$$M \equiv (M^d)^e \pmod{n}.$$

## 3.11 Considerations regarding the security of the RSA algorithm[66]

There have always been discussions about the suitability of the RSA algorithm for digital signatures and encryption, e. g. after publications of breakthroughs in factorisation. Nevertheless the RSA algorithm has become a de-facto standard since it was published more than 20 years ago (compare 6.1).

The security of the RSA algorithm rests — as with all cryptographic methods — on the following 4 central pillars:

- the complexity of the number theoretical problem on which the algorithm is based (here factorisation of big numbers),
- the election of fitting parameters (here the length of the module $N$),
- the adequate usage of the algorithm and key generation and
- the correct implementation of the algorithm.

Usage and key generation are well understood today. Implementation based on long integer arithmetic is very easy.

The following sections examine the RSA algorithm with respect to the first two points.

### 3.11.1 Complexity

Successful decryption or forgery of a signature — without knowing the private key — requires calculating the $e$-th root mod $n$. The private key, this is the multiplicative inverse of $e$ mod $J(n)$, can be easily determined if $J(n)$ is known. $J(n)$ again can be calculated from the prime factors of $n$. Breaking of RSA therefore cannot be more difficult than factorisation of the module $n$.

The best factorisation method known today is a further development of the General Number Field Sieve (GNFS), which was originally devised to factor only numbers of a special form (like Fermat numbers). The complexity of solving the factorisation problem with the GNFS is asymptotically

$$O(l) = e^{c\cdot(l\cdot\ln 2)^{1/3}\cdot(\ln(l\cdot\ln(2))^{2/3}+o(l)}$$

Please refer to:

- A. Lenstra, H. Lenstra: *The development of the Number Field Sieve* [Lenstra1993].

[66] Major parts of chapters 3.11.1 and 3.11.2 follow the article "Vorzüge und Grenzen des RSA-Verfahrens" written by F. Bourseau, D. Fox and C. Thiel [Bourseau2002]

- Robert D. Silverman: *A Cost-Based Security Analysis of Symmetric and Asymmetric Key Lengths* [Silverman2000].

This formula shows, that the factorisation problem belongs to the class of problems with sub-exponential time complexity (i. e. time complexity grows asymptotically not as fast as exponential functions like $e^l$ or $2^l$, but strictly slower, e. g. like $e^{\sqrt{l}}$). This classification is all that is currently known; it does not preclude the possibility that the factorisation problem can be solved in polynomial time (see 3.11.3).

$O(l)$ is the average number of processor steps depending on the bit length $l$ of the number $n$ to be factorised. For the best currently known factorisation algorithm the constant $c = (64/9)^{1/173} = 1,923$.

The inverse proposition, that the RSA algorithm can be broken only by factorisation of $n$, is still not proven. Most number theorists consider the "RSA problem" and the factorisation problem equivalent in terms of time complexity.

Please refer to: *Handbook of Applied Cryptography* [Menezes2001].

### 3.11.2 Security parameters because of progress in factorisation

The complexity is basically determined by the length $l$ of the module $n$.

- In 1994 a 129-digit RSA module (428 bit), published in 1977, was factorised by a distributed implementation of the Quadratic Sieve algorithm (QS), developed 1982 by Pomerance. This effort took 8 months.

  Please refer to:

  C. Pomerance: *The quadratic sieve factoring algorithm* [Pomerance1984].

- In 1999 a 155-digit module (512 bit) was factored with an implementation of the General Number Field Sieve algorithm (GNFS), developed by Buhler, Lenstra and Pomerance. The GNFS is more efficient than QS if $n$ is longer than about 116 decimal digits. This effort took 5 months.

  Please refer to:

  J.P. Buhler, H.W. Lenstra, C. Pomerance: *Factoring integers with the number field sieve* [Buhler1993].

This made evident that a module length of 512 bit no longer prevents from attackers.

Within the last 20 years a lot of progress has been made. Estimations about the future development of the ability to factor RSA modules vary and depend on some assumptions:

- progression in computing performance (Moore's law: every 18 month the computing power will double) and in grid computing.

- development of new algorithms.

Within the last years the module bit length feasible for factorisation increased — even without new algorithms — by 10 bit per year. Larger numbers require not only more time to be factored, but also huge RAM storage for the solutions matrix being used by the best algorithms known today. This need for storage grows like the square root of the computation time, i. e. also sub-exponentially. Because RAM availability increased exponentially in the recent decades, it seems that this should not be the limiting factor.

Within the above mentioned article [Bourseau2002] Dirk Fox[67] prognosticates an almost linear progression with the factorisation, if all factors are included: Each year the module length feasible for factorisation increases by 20 bit on average.

### 3.11.3 Further current research in factorisation and number theory

Prime numbers are part of very many topical research areas in number theory and computer science. Progress made with factorisation is bigger than estimated 5 years ago – this is not only founded on faster computers but also in new knowledge.

The security of the RSA algorithm is based on the empirical observation that factoring large numbers is a hard problem. A Module $n$ (typically, 1024 bit) can be easily constructed as the product of two large primes $p$, $q$ (typically, 500−600 bit each), by calculating $n = pq$. However, it is a hard problem to extract $p$, $q$ from $n$. Without knowing $p$ or $q$, the private key cannot be calculated.

Thus, any progress in efficiency of factorising large integers will effect the security of the RSA. As a consequence, the underlying primes $p$, $q$ and, thus, the module n (1024 bit as of today) have to be increased. In the worst case of a breakthrough in factorisation, the RSA algorithm might be compromised.

**Bernstein's paper and its implication on the security of the RSA algorithm**

In his paper "Circuits for integer factorisation: a proposal" (`http://cr.yp.to/djb.html`), published November 2001, D. J. Bernstein [Bernstein2001] addresses the problem of factorising large integers. Therefore, his results are of relevance from a RSA point of view. As a main result Bernstein claims that the implementation of the General Number Field Sieve algorithm (GNFS) can be improved to factor, with the same effort as before, integers with three times more digits.

We note that the definition of *effort* is a crucial point: Bernstein claims that effort is the product of time and costs of the machine (including the memory used). The gist of the paper lies in the fact that he can reduce a big part of factorising to sorting. Using Schimmler's scheme, sorting

[67] His company Secorvo Ltd made a statement on the recommendation for key length selection published by the GISA (German Information Security Agency). Chapter 2.3.1 of their statement contains a very competent and understandable discussion of RSA security (this document exists – to my knowledge – only in German): `http://www.secorvo.de/publikat/stellungnahme-algorithmenempfehlung-020307.pdf`

can be optimized by massive parallel computing. At the end of section 3 Bernstein explains this effect: The costs of $m^2$ parallel computers with a constant amount of memory is a constant times $m^2$. The costs of a computer with a single processor and memory of size $m^2$ is also of the order of $m^2$, but with a different constant factor. With $m^2$ processors in parallel, sorting of $m^2$ numbers (with Schimmler's scheme) can be achieved in time $m$, while a $m^2$-memory computer needs time of the order of $m^2$. Decreasing memory and increasing the number of processors, the computing time can be reduced by a factor $1/m$ without additional effort in terms of total costs. In section 5 it is said that massive parallel computing can also increase efficiency of factorising using Lenstra's elliptic-curve-method (a search algorithm has costs that increase in a quadratic square manner instead of cubically).

We note that all results achieved so far are asymptotic results. This means that they only hold in the limit n to infinity. Unfortunately, there is no upper limit for the residual error (i.e. the difference between the real and the asymptotic value) for finite n — a problem which has already been addressed by the author. As a consequence, one cannot conclude whether the costs (in the sense of Bernstein) for factorising 1024−2048-bit RSA modules can be significantly reduced.

There is no doubt that Bernstein's approach is innovative. However, the reduction of computing time under constant costs comes along with a massive use of parallel computing — a scenario which seems not to be realistic yet. For example, formally 1 sec computing time on one machine and 1/1,000,000 sec time parallel computing time on 1,000,000 machines might have same costs. In reality, it is much harder to realize the second situation, and Bernstein does not take into account the fixed costs, in particular for building a network between all these computers.

Although distributed computing over a large network might help to overcome this problem, realistic costs for data transfer has to be taken into account — a point which was not addressed in Bernstein's proposal.

As long as there is neither (low cost) hardware nor a distributed computing approach (based on Bernstein's ideas), there should not be a problem for RSA. It has to be clarified from which magnitude of n on Bernstein's method could lead to a significant improvement (in the sense of the asymptotic result).

Arjen Lenstra, Adi Shamir et. al. analyzed the paper of Bernstein [Lenstra2002]. In summary they expect a factorisation improvement on how much longer the bit length of the keys could be with a factor of 1.17 (instead of factor 3 as proposed by Bernstein).

The abstract of their paper "Analysis of Bernstein's Factorization Circuit" says:

"... Bernstein proposed a circuit-based implementation of the matrix step of the number field sieve factorisation algorithm. We show that under the non-standard cost function used in [1], these circuits indeed offer an asymptotic improvement over other methods but to a lesser degree than previously claimed: for a given cost, the new method can factor integers that are 1.17 times larger (rather than 3.01). We also propose an improved circuit design based on a new mesh routing algorithm, and show that for factorisation of 1024-bit integers the matrix step can, under an optimistic assumption about the matrix size, be completed within a day by a device that costs a few thousand dollars. We conclude that from a practical standpoint, the security of RSA relies exclusively on the hardness of the relation collection step of the number field sieve."

RSA Security's analysis of the Bernstein paper [RSA Security 2002] from April, 8 2002 also – as expected – concludes, that RSA is still not compromised.

This is still an ongoing discussion.

When this section was written (June 2002) nothing was publicly known, how far there exist implementations of his theoretical onsets and how much financing there was for his research project.

Links:
 http://cr.yp.to/djb.html
 http://www.counterpane.com/crypto-gram-0203.html#6
 http://www.math.uic.edu

**The TWIRL Device**

In January 2003 Adi Shamir and Eran Tromer from the Weizmann Institute of Science published a preliminary draft called *"Factoring Large Numbers with the TWIRL Device"* raising concerns about the security of key sizes till 1024 bits [Shamir2003].

Their abstract summarizes their results very well: "The security of the RSA cryptosystem depends on the difficulty of factoring large integers. The best current factoring algorithm is the Number Field Sieve (NFS), and its most difficult part is the sieving step. In 1999 a large distributed computation involving thousands of workstations working for many months managed to factor a 512-bit RSA key, but 1024-bit keys were believed to be safe for the next 15-20 years. In this paper we describe a new hardware implementation of the NFS sieving step ... which is 3-4 orders of magnitude more cost effective than the best previously published designs ... . Based on a detailed analysis of all the critical components (but without an actual implementation), we believe that the NFS sieving step for 1024-bit RSA keys can be completed in less than a year with a $10M device, and that the NFS sieving step for 512-bit RSA keys can be completed in less than ten minutes with a $10K device. Coupled with recent results about the difficulty of the NFS matrix step ... this raises some concerns about the security of these key sizes."

So recommendations like the ones from the GISA (German Information Security Agency) to use higher key lengths are very valid.

**"Primes in P": Primality testing is polynomial**

In August 2002 the three Indian researchers M. Agrawal, N. Kayal and N. Saxena published the paper *"PRIMES in P"* about a new primality testing algorithm [Agrawal2002]. They discovered a polynomial time deterministic algorithm for determining if a number is prime or not.

The importance of this discovery is that it provides number theorists with new insights and opportunities for further research. Lots of people over centuries have been looking for a polynomial time test for primality, and this result is a major theoretic breakthrough. It shows that new results can be generated from already known facts.

But even its authors note that other known algorithms may be faster (for example ECPP). The new algorithm works on any integer. For example the GIMPS project uses the Lucas-Lehmer primality test which takes advantage of the special properties of Mersenne numbers. This makes the Lucas-Lehmer test much faster, allowing to test numbers with millions of digits while general purpose algorithms are limited to numbers with a few thousand digits.

Current research results on this topic can be found at:

http://www.mersenne.org/
http://fatphil.org/maths/AKS/

| |
|---|
| *Hermann Hesse*[68]*:*<br>To let the possible happen, you again and again have to try the impossible. |

### 3.11.4 Status regarding factorisation of concrete large numbers

The web page http://www.crypto-world.com contains an excellent overview about the factoring records of composed integers using different methods. The current record (as of June 2002) obtained using the GNFS method (General Number Field Sieve) factorised a 158-digit into its both prime factors (these are build with 73 and 86 decimal digits).

This record, established on January 18, 2002 by researchers of the German University of Bonn[69], got much less attention within the press than the solution of the RSA challenge[70], established on August 22, 1999, when researchers from the Netherlands factorised the 155-digit number into its both 78-digit primes (see chapter 3.11.2). One reason is that 512 bit RSA-155 was a kind of *magic* border.

The task of the researchers from Bonn was not initiated by a challenge, but they wanted to find the last prime factors of the integer $2^{953} - 1$ (see "Wanted List" of the Cunningham Project[71]).

The 6 smaller prime factors, already found before have been:

$$\begin{gathered} 3, 1907, 425796183929, \\ 1624700279478894385598779655842584377, \\ 3802306738549441324432139091271828121 \quad \text{and} \\ 128064886830166671444802576129115872060027. \end{gathered}$$

The first 3 factors can be easily computed[72]. The next three prime factors were found by P. Zimmerman [73], T. Grandlund[74] and R. Harley during the years 1999 and 2000 using the elliptic curve factorisation method.

[68] German/Swiss writer and Nobel Prize winner, July 2, 1877 – August 9, 1962.
[69] http://www.ercim.org/publication/Ercim_News/enw49/franke.html, 2002-01
[70] http://www.rsasecurity.com/rsalabs/challenges/factoring/numbers.html
[71] http://www.cerias.purdue.edu/homes/ssw/cun/
[72] e.g. using CrypTool via menu **Indiv. Procedures \ RSA Demonstration \ Factorisation of a Number**.
[73] http://www.loria.fr/~zimmerma/ecmnet
[74] http://www.swox.se/gmp/

The last remaining factor, called “C158”, was known to be composite by then, but its factors where not known (the following 3 lines contain one number):

395058745832651445264197678006144819960207764603049364541393760515793556265294506836097278424682195350935443058704902519956553357102097992264849779494429556O3

The factorisation of C158 resulted in the following two prime factors:

3388495837466721394368393204672181522815830368604993048084925840555281177

and

11658823406671259903148376558383270818131012258146392600439520994131344334162924536139.

So now all 8 prime factors of $2^{953} - 1$ have been found.

Links:
- http://www.loria.fr/ zimmerma/records/gnfs158
- http://www.crypto-world.com/FactorRecords.html
- http://www.crypto-world.com/announcements/c158.txt

As you notice the factorised compound numbers built of 2 prime factors are much smaller than the especially structured numbers, for which primality tests are able to decide whether they are prime or not (see chapters 2.4 and 2.5).

*Joanne K. Rowling*[75]*:*
It is our choices, that show what we truly are, far more than our abilities.

## 3.12 Further applications of number theory in cryptography

The results of modular arithmetic are used extensively in modern cryptography. Here we will provide a few examples from cryptography using small[76] numbers.

Enciphering a text entails applying a function (mathematical operation) to a character string (number) to generate a different number. Deciphering entails reversing this function, in other words using the distorted image that the function has created from the plaintext in order to restore the original image. For example, the sender could take the plaintext $M$ of a confidential message and add a secret number, the key $S$, to obtain the cipher text $C$:

$$C = M + S.$$

The recipient can reconstruct the plaintext by reversing this operation, in other words by subtracting $S$:

$$M = C - S.$$

Adding $S$ reliably makes the plaintext impossible to read. However, this encryption is rather weak, because all an interceptor needs to do to calculate the key is obtain a plaintext and the associated cipher text

$$S = C - M,$$

and can then read any subsequent messages encrypted using $S$.
The essential reason for this is that subtraction is just as simple an operation as addition.

### 3.12.1 One way functions

If the key is to be impossible to determine even with knowledge of both the plaintext and the cipher text, we need a function that is, on the one hand, relatively easy to calculate – we don't want to have problems encrypting messages. On the other hand, the inverse function should exist (otherwise information would be lost during encryption), but should be de facto incalculable.

What are possible candidates for such a **one way function**? We could take multiplication rather than addition, but even primary school children know that the inverse function, division, is only slightly more difficult than multiplication itself. We need to go one step higher in the hierarchy of calculation methods. It is still relatively simple to calculate the power of a number, but the corresponding two reverse functions – *taking roots* (find $b$ in the equation $a = b^c$ when $a$ and $c$

[75] Joanne K. Rowling, "Harry Potter and the Chamber of Secrets", Bloomsbury, 1998, last chapter "Dobby's reward", p. 245, by Dumbledore.

[76] In the RSA procedure, we call numbers "small" if the bit lengths are much less than 1024 bits (i.e. 308 decimal points). In practice, 1024 bits is currently the minimum length for a secure Certification Authority RSA modulus.

are known) and *calculating logarithms* (find $c$ in the above equation when $a$ and $b$ are known) are so complicated that pupils normally do not learn them at school.

Although a certain structure can still be recognised for addition and multiplication, raising numbers to the power of another or calculating exponentials totally mixes up all the numbers. Knowing a few values of the function doesn't tell us much about the function as a whole (in contrast to addition and multiplication).

### 3.12.2 The Diffie-Hellman key exchange protocol

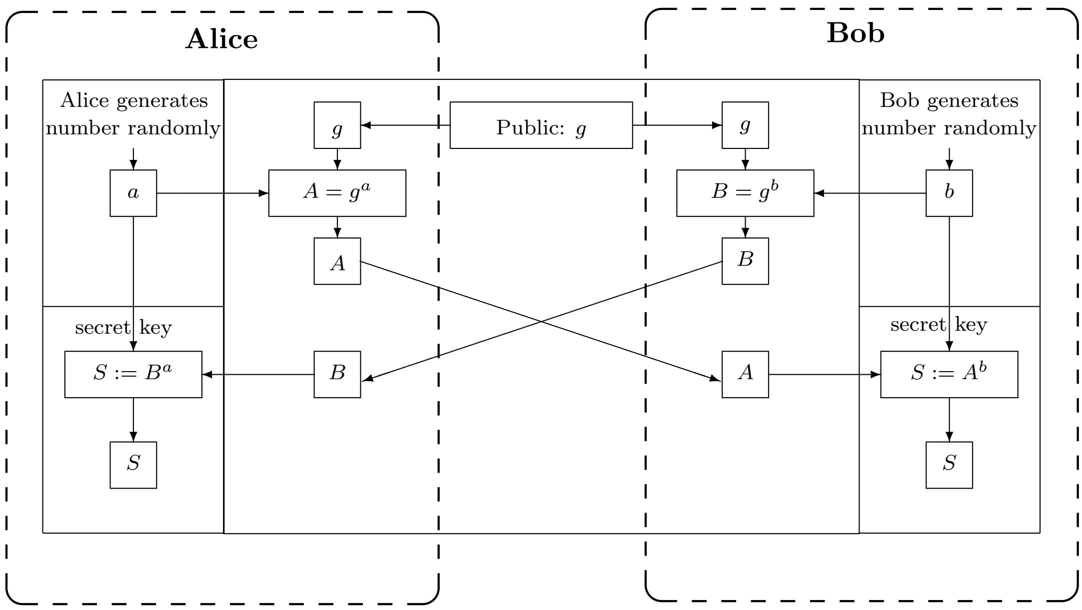
Whitfield Diffie, Martin E. Hellman and Ralph Merkle developed this DH key exchange protocol in Stanford in 1976[77].

Alice and Bob[78] use a one way function to obtain a key $S$, the session key, for subsequent correspondence. This is then a secret that is only known to the two of them. Alice selects a random number $a$ and keeps it secret. She applies a one way function to $a$ to calculate the number $A = g^a$ and sends it to Bob. He does the same, by selecting a secret random number $b$, calculating $B = g^b$ and sending it to Alice. The number $g$ is random and can be publicly known. Alice applies the one way function together with her secret number $a$ to $B$, while Bob does the same with his secret number $b$ and the received number $A$.

The result $S$ is the same in each case because the one way function is commutative: $(g^a)^b = (g^b)^a$. But even Bob cannot reconstruct Alice's secret number $a$ from the data available to him, while Alice cannot determine Bob's secret number $b$. And a perpetrator who knows $g$ and has intercepted both $A$ and $B$ cannot use this knowledge to determine $a, b$ or $S$.

[77] With CrypTool v1.3.04 this exchange protocol has been visualized: you can execute the single steps with concrete numbers using menu **Indiv. Procedures \ Diffie-Hellman Demonstration**.

[78] Bob and Alice are the default names used for the two authorised participants in a protocol (see [Schneier1996, p. 23]).




**Procedure:**

Alice and Bob want to negotiate a secret session key $S$ via a channel that may be intercepted.

1. They select a prime number $p$ and a random number $g$ and exchange this information openly.

2. Alice now selects $a$, a random number less than $p$ and keeps it secret.

   Similarly, Bob selects $b$, a random number less than $p$ and keeps it secret.

3. Alice now calculates $A \equiv g^a \pmod{p}$.
   Bob calculates $B \equiv g^b \pmod{p}$.

4. Alice sends the result $A$ to Bob.
   Bob sends the result $B$ to Alice.

5. In order to now determine the session key to be used by both, they both separately raise the respective results they have received to the power of their secret random number modulo $p$. This means:

   - Alice calculates $S \equiv B^a \pmod{p}$ and
   - Bob calculates $S \equiv A^b \pmod{p}$.

Even if a spy intercepts $g, p$, and the interim results $A$ and $B$, he cannot use these to determine the session key used due to the difficulty of calculating the discrete logarithm.

We will now use an example with (unrealistically) small numbers to illustrate this.

**Example using numbers:**

1. Alice and Bob select $g = 11, p = 347$.
2. Alice selects $a = 240$, Bob selects $b = 39$ and they keep $a$ and $b$ secret.
3. Alice calculates $A \equiv g^a \equiv 11^{240} \equiv 49 \pmod{347}$.
   Bob calculates $B \equiv g^b \equiv 11^{39} \equiv 285 \pmod{347}$.
4. Alice sends Bob: $A \equiv 49$,
   Bob sends Alice: $B \equiv 285$.
5. Alice calculates $B^a \equiv 285^{240} \equiv 268 \pmod{347}$,
   Bob calculates $A^b \equiv 49^{39} \equiv 268 \pmod{347}$.

Alice and Bob can now communicate securely using their shared session key. Even if spies were to intercept everything transferred via the connection: $g = 11, p = 347, A = 49$ and $B = 285$, they would not be able to calculate the secret key.

**Comment:**
In this example using such small numbers, it would be possible, but with large numbers the discrete logarithm problem[79,80] is extremely difficult to solve.

Here, we need to calculate:
For Alice: $11^x \equiv 49 \pmod{347}$, that means $\log_{11}(49) \pmod{347}$.
For Bob: $11^y \equiv 285 \pmod{347}$, that means $\log_{11}(285) \pmod{347}$.

## 3.13 The RSA procedure with actual numbers

Having described above how the RSA procedure works, we will now work through the steps using actual, but small, numbers.

---

[79] If you try to determine the discrete logarithm $x$ that solves the equation $11^x \equiv 49 \pmod{347}$ with Mathematica by means of Solve, you obtain the em tdep message "The equations appear to involve the variables to be solved for in an essentially non-algebraic way". Mathematica therefore claims not to know a direct algebraic procedure for solving the equation. Yet Mathematica is able to calculate this with the general function for the multiplicative order (here for Alice): `MultiplicativeOrder[11, 347, 49]` delivers the value 67.
The syntax with Pari-GP is: `znlog(Mod(49,347),Mod(11,347))`.
Such number-theory tasks can also be solved using other tools such as the LiDIA or BC package (see web links in appendix). The dl function in the **LC** user interface for LiDIA also delivers the value 67 for `dl(11,49,347)`.

[80] Why have the functions delivered the value 67 rather than 240 for the dl problem for Alice? The discrete logarithm is the smallest natural exponent that solves the equation $11^x \equiv 49 \pmod{347}$. Both $x = 67$ and $x = 240$ (the number selected in the example) satisfy the equation and can therefore be used to calculate the session key: $285^{240} \equiv 285^{67} \equiv 268 \pmod{347}$. If Alice and Bob had selected a primitive root modulo $p$ as base $g$, then for every remainder from the set $\{1, 2, \cdots, p-1\}$ there is exactly one exponent from the set $\{0, 1, \cdots, p-2\}$.
For info: there are 172 different primitive roots for modulo 347, 32 of which are prime (not necessary). Since the number 11 selected for $g$ in the example is not a primitive root of 347, the remainders do not take all values from the set $\{1, 2, \cdots, 346\}$. Thus, for a particular remainder there may be more than one exponent or even no exponent at all in the set $\{0, 1, \cdots, 345\}$ that satisfies the equation.
`PrimeQ[347] = True; EulerPhi[347] = 346; GCD[11, 347] = 1; MultiplicativeOrder[11, 347] = 173`
The syntax with Pari-GP is: `isprime(347); eulerphi(347); gcd(11,347); znorder(Mod(11,347))`.

### 3.13.1 RSA with small prime numbers and with a number as message

Before applying the RSA procedure to a text, we will first demonstrate it directly using a single number as message[81].

1. Let the selected prime numbers be $p = 5$ and $q = 11$.
   Thus, $n = 55$ and $J(n) = (p-1) * (q-1) = 40$.
2. $e = 7$ (should lie between 11 and 40 and must be relatively prime to 40).
3. $d = 23$ (since $23 * 7 \equiv 161 \equiv 1 \pmod{40}$),

   $\rightarrow$ Public key of the recipient: $(55, 7)$,

   $\rightarrow$ Private key of the recipient: $(55, 23)$.
4. Let the message be the number $M = 2$ (so no division into blocks is required).
5. Encryption: $C \equiv 2^7 \equiv 18 \pmod{55}$.
6. The cipher text is simply the number $C = 18$ (we therefore do not need to divide it into blocks).
7. Decryption: $M \equiv 18^{23} \equiv 18^{(1+2+4+16)} \equiv 18 * 49 * 36 * 26 \equiv 2 \pmod{55}$.

We will now apply the RSA procedure to a text, first using the upper case alphabet (26 characters), then using the entire ASCII character set as the basis for the messages.

| i | $11^i$ mod 347 | |
|---|---|---|
| 0 | 1 | |
| 1 | 11 | |
| 2 | 121 | |
| 3 | 290 | |
| 67 | 49 | searched exponent |
| 172 | 284 | |
| 173 | 1 | = multiplicative order of 11 (mod 347) |
| 174 | 11 | |
| 175 | 121 | |
| 176 | 290 | |
| 240 | 49 | searched exponent |

[81] Using CrypTool v1.3 you can solve this with the **Indiv.Procedures \ RSA Demonstration \ RSA Cryptosystem**.

### 3.13.2 RSA with slightly larger primes and a text of upper case letters

We have the text "ATTACK AT DAWN" and the characters are coded in the following simple manner[82]:

**Table 7: capital letters alphabet**

| Character | Numerical value | Character | Numerical value |
|---|---|---|---|
| Blank | 0 | M | 13 |
| A | 1 | N | 14 |
| B | 2 | O | 15 |
| C | 3 | P | 16 |
| D | 4 | Q | 17 |
| E | 5 | R | 18 |
| F | 6 | S | 19 |
| G | 7 | T | 20 |
| H | 8 | U | 21 |
| I | 9 | V | 22 |
| J | 10 | W | 23 |
| K | 11 | X | 24 |
| L | 12 | Y | 25 |
| | | Z | 26 |

**Key generation (steps 1 to 3)**:
**1.** $p = 47, q = 79$ ($n = 3,713$; $J(n) = (p-1)*(q-1) = 3,588$).
**2.** $e = 37$ (should lie between 79 and $3,588$ and must be relatively prime to $3,588$).
**3.** $d = 97$ (since $e * d = 1 \bmod J(n); 37 * 97 \equiv 3,589 \equiv 1 \pmod{3,588}$ )[83].
**4. Encryption**:

| Text: | A | T | T | A | C | K | | A | T | | D | A | W | N |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Number: | 01 | 20 | 20 | 01 | 03 | 11 | 00 | 01 | 20 | 00 | 04 | 01 | 23 | 14 |

This 28-digit number is divided into 4-digit parts (because $2,626$ is still smaller than $n = 3,713$):
`0120 2001 0311 0001 2000 0401 2314`

All 7 parts are encrypted using: $C \equiv M^{37} \pmod{3,713}$[84]:
`1404 2932 3536 0001 3284 2280 2235`

**5. Decryption**:
Cipher text: `1404 2932 3536 0001 3284 2280 2235`
This 28-digit number is divided into 4-digit parts.

[82] Using CrypTool v1.3 you can solve this with the **Indiv.Procedures \ RSA Demonstration \ RSA Cryptosystem**. This is also described in the tutorial/scenario in CrypTool's online help [Options, specify alphabet, number system, block length 2 and decimal representation].

[83] How to compute $d = 97$ using the *extended* gcd algorithm is shown in appendix A of this chapter

[84] See Appendix D of this chapter for source code to do RSA encryption using Mathematica and Pari-GP. You can also encrypt the message with CrypTool v1.3 **Indiv. Procedures \ RSA Demonstration \ Factorisation of a Number**.

All 7 parts are decrypted using: $M \equiv C^{97} \pmod{3,713}$:
`0120 2001 0311 0001 2000 0401 2314`
The 2-digit numbers are transformed into capital letters and blanks.

Using the selected values it is easy for a cryptanalyst to derive the secret values from the public parameters $n = 3,713$ and $e = 37$ by revealing that $3,713 = 47 * 79$.
If $n$ is a 768-bit number, there is, according to present knowledge, little chance of this.

### 3.13.3 RSA with slightly more larger primes and a text made up of ASCII characters

In real life, the ASCII alphabet is used to code the individual characters of the message as 8-bit numbers.

The idea for this task[85] is taken from the example in [Eckert2003, p. 271].

Coded in decimal notation, the text "RSA works!" is as follows:

| Text: | R | S | A | | w | o | r | k | s | ! |
|---|---|---|---|---|---|---|---|---|---|---|
| Number: | 82 | 83 | 65 | 32 | 119 | 111 | 114 | 107 | 115 | 33 |

We will work through the example in 2 variants. The steps 1 to 3 are common for both.

**Key generation (steps 1 to 3):**
**1.** $p = 503, q = 509 \quad (n = 256,027;\ J(n) = (p-1)(q-1) = 255,016 = 2^3 * 127 * 251)$[86].
**2.** $e = 256,027$ (should lie between 509 and 255,016 and must be relatively prime to 255,016)[87].
**3.** $d = 231,953$
(since $e \equiv d^{-1} \bmod J(n):\ 65,537 * 231,953 \equiv 15,201,503,761 \equiv 1 \pmod{67,000}$)[88].

**Variant 1: All ASCII characters are encrypted and decrypted separately (no blocks are formed).**

**4. Encryption:**

| Text: | R | S | A | | w | o | r | k | s | ! |
|---|---|---|---|---|---|---|---|---|---|---|
| Number: | 82 | 83 | 65 | 32 | 119 | 111 | 114 | 107 | 115 | 33 |

The letters are not combined[89]!

---

[85] Using CrypTool v1.3 you can solve this with the **Indiv.Procedures \ RSA Demonstration \ RSA Cryptosystem**.

[86] See Appendix D of this chapter for the source code to factorise the number $J(n)$ using Mathematica and Pari-GP.
Using CrypTool v1.3 you can solve this with the **Indiv.Procedures \ RSA Demonstration \ Factorisation of a Number**.

[87] $e$ cannot, therefore, be 2, 127 or 251 ($65537 = 2^{16} + 1$).
In real life, $J(n)$ is not factorised but rather the Euclidean algorithm is used for the selected e to guarantee that $\gcd(d, J(n)) = 1$.

[88] Other possible combinations of $(e, d)$ include: $(3, 170, 011)$, $(5, 204, 013)$, $(7, 36, 431)$.

[89] For secure procedures we need large numbers that assume – as far as possible – all values up to $n-1$. If the possible value set for the numbers in the message is too small, even large prime numbers cannot make the procedure secure. An ASCII character is represented by 8 bits. If we want larger values we must combine several numbers. Two characters need 16 bits, whereby the maximum value that can be represented is 65536. The modulus $n$ must then be greater than $2^{16} = 65536$. This is applied in variant 2. When the numbers are combined, the leading zeros are

Each character is encrypted using: $C = M^{65,537} \pmod{256,027}$[90]:

```
212984  025546  104529  031692  248407
100412  054196  100184  058179  227433
```

**5. Decryption**:

Cipher text:

```
212984  025546  104529  031692  248407
100412  054196  100184  058179  227433
```

Each character is decrypted using: $M \equiv C^{231,953} \mod 256,027$:

`82 83 65 32 119 111 114 107 115 33`

**Variant 2: The ASCII characters are encrypted and decrypted two at a time as blocks.**

In variant 2 the block formation is done in two different sub-variants: (4./5. and 4'./5'.).

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Text: | R | S | A | | w | o | r | k | s | ! |
| Number: | 82 | 83 | 65 | 32 | 119 | 111 | 114 | 107 | 115 | 33 |

**4. Encryption**:

Blocks are formed[91] (each ASCII character is encoded into a 8 digit binary number below):

`21075 16672 30575 29291 29473`[92]

Each block is encrypted using: $C \equiv M^{65,537} \pmod{256,027}$[93]:

`158721 137346 37358 240130 112898`

**5. Decryption:**

Cipher text:

`158721 137346 37358 240130 112898`

Each block is decrypted using: $M \equiv C^{231,953} \pmod{256,027}$:

`21075 16672 30575 29291 29473`

---

kept in binary notation (just as if we were to write all numbers with 3 digits in decimal notation above and were then to obtain the sequence `082 083, 065 032, 119 111, 114 107, 115 033`).

[90] See Appendix D of this chapter for the source code for RSA exponentiation using Mathematica and Pari-GP.

[91]

```
                 binary representation   decimal representation
01010010, 82     01010010 01010011       =21075
01010011, 83
01000001, 65     01000001 00100000       =16672
00100000, 32
01110111, 119    01110111 01101111       =30575
01101111, 111
01110010, 114    01110010 01101011       =29291
01101011, 107
01110011, 115    01110011 00100001       =29473
00100001, 33:
```

[92] Using CrypTool v1.3 you can solve this with the **Indiv.Procedures \ RSA Demonstration \ RSA Cryptosystem** with the following options: all 256 ASCII characters, b-adic, block length 2 and decimal representation.

[93] See Appendix D of this chapter for the source code for RSA exponentiation using Mathematica and Pari-GP.

**4'. Encryption:**
Blocks are formed: (each ASCII character is encoded into a 3 digit decimal number below):
`82083 65032 119111 114107 115033`[94]

Each block is encrypted using: $C \equiv M^{65,537} \pmod{256,027}$[95]:
`198967 051405 254571 115318 014251`

**5'. Decryption:**
Cipher text:
`198967 051405 254571 115318 014251`

Each block is decrypted using: $M \equiv C^{2,473} \pmod{67,519}$:
`82083 65032 119111 114107 115033`

### 3.13.4 A small RSA cipher challenge (1)

The task is taken from [Stinson1995, Exercise 4.6]: The pure solution has been published by Prof. Stinson at http://www.cacr.math.uwaterloo.ca/~dstinson/solns.html.[96]

However, it is not the result that is important here but rather the individual steps of the solution, that is, the explanation of the cryptanalysis[97].

Two samples of RSA cipher text are presented in Tables 4.1 and 4.2. Your task is to decrypt them. The public parameters of the system are

$n = 18,923$ and $e = 1,261$ (for Table 4.1) and
$n = 31,313$ and $e = 4,913$ (for Table 4.2).

This can be accomplished as follows. First, factor $n$ (which is easy because it is so small). Then compute the exponent $d$ from $J(n)$, and, finally, decrypt the cipher text. Use the square-and-multiply algorithm to exponentiate modulo $n$.

In order to translate the plaintext back into ordinary English text, you need to know how alphabetic characters are "encoded" as elements in $\mathbb{Z}_n$. Each element of $\mathbb{Z}_n$ represents three alphabetic characters as in the following examples:

$$\begin{array}{lcl} \texttt{DOG} & \mapsto & 3 * 26^2 + 14 * 26 + 6 = 2,398 \\ \texttt{CAT} & \mapsto & 2 * 26^2 + 0 * 26 + 19 = 1,371 \\ \texttt{ZZZ} & \mapsto & 25 * 26^2 + 25 * 26 + 25 = 17,575\,. \end{array}$$

You will have to invert this process as the final step in your program.

The first plaintext was taken from "The Diary of Samuel Marchbanks", by Robertson Davies, 1947, and the second was taken from "Lake Wobegon Days", by Garrison Keillor, 1985.

[94] The RSA encryption works correctly with the modulus $n = 256.027$ because each ASCII block of two characters will be encoded into a number that is smaller or equal than the number $255,255$.
[95] See Appendix D of this chapter for the source code for RSA exponentiation using Mathematica and Pari-GP.
[96] or http://bibd.unl/~stinson/solns.html.
[97] The method of solving the problem is outlined in the scenario of the online help to CrypTool and in the presentation on the website. If anyone sends us a well prepared exact method of solving the problem, we would be pleased to include it in the documentation.

TABLE 4.1[98]: RSA cipher text

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 12423 | 11524 | 7243 | 7459 | 14303 | 6127 | 10964 | 16399 |
| 9792 | 13629 | 14407 | 18817 | 18830 | 13556 | 3159 | 16647 |
| 5300 | 13951 | 81 | 8986 | 8007 | 13167 | 10022 | 17213 |
| 2264 | 961 | 17459 | 4101 | 2999 | 14569 | 17183 | 15827 |
| 12693 | 9553 | 18194 | 3830 | 2664 | 13998 | 12501 | 18873 |
| 12161 | 13071 | 16900 | 7233 | 8270 | 17086 | 9792 | 14266 |
| 13236 | 5300 | 13951 | 8850 | 12129 | 6091 | 18110 | 3332 |
| 15061 | 12347 | 7817 | 7946 | 11675 | 13924 | 13892 | 18031 |
| 2620 | 6276 | 8500 | 201 | 8850 | 11178 | 16477 | 10161 |
| 3533 | 13842 | 7537 | 12259 | 18110 | 44 | 2364 | 15570 |
| 3460 | 9886 | 8687 | 4481 | 11231 | 7547 | 11383 | 17910 |
| 12867 | 13203 | 5102 | 4742 | 5053 | 15407 | 2976 | 9330 |
| 12192 | 56 | 2471 | 15334 | 841 | 13995 | 17592 | 13297 |
| 2430 | 9741 | 11675 | 424 | 6686 | 738 | 13874 | 8168 |
| 7913 | 6246 | 14301 | 1144 | 9056 | 15967 | 7328 | 13203 |
| 796 | 195 | 9872 | 16979 | 15404 | 14130 | 9105 | 2001 |
| 9792 | 14251 | 1498 | 11296 | 1105 | 4502 | 16979 | 1105 |
| 56 | 4118 | 11302 | 5988 | 3363 | 15827 | 6928 | 4191 |
| 4277 | 10617 | 874 | 13211 | 11821 | 3090 | 18110 | 44 |
| 2364 | 15570 | 3460 | 9886 | 9988 | 3798 | 1158 | 9872 |
| 16979 | 15404 | 6127 | 9872 | 3652 | 14838 | 7437 | 2540 |
| 1367 | 2512 | 14407 | 5053 | 1521 | 297 | 10935 | 17137 |
| 2186 | 9433 | 13293 | 7555 | 13618 | 13000 | 6490 | 5310 |
| 18676 | 4782 | 11374 | 446 | 4165 | 11634 | 3846 | 14611 |
| 2364 | 6789 | 11634 | 4493 | 4063 | 4576 | 17955 | 7965 |
| 11748 | 14616 | 11453 | 17666 | 925 | 56 | 4118 | 18031 |
| 9522 | 14838 | 7437 | 3880 | 11476 | 8305 | 5102 | 2999 |
| 18628 | 14326 | 9175 | 9061 | 650 | 18110 | 8720 | 15404 |
| 2951 | 722 | 15334 | 841 | 15610 | 2443 | 11056 | 2186 |

[98] The numbers of this table can be worked with via Copy and Paste.

TABLE 4.2[99]: RSA cipher text

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 6340 | 8309 | 14010 | 8936 | 27358 | 25023 | 16481 | 25809 |
| 23614 | 7135 | 24996 | 30590 | 27570 | 26486 | 30388 | 9395 |
| 27584 | 14999 | 4517 | 12146 | 29421 | 26439 | 1606 | 17881 |
| 25774 | 7647 | 23901 | 7372 | 25774 | 18436 | 12056 | 13547 |
| 7908 | 8635 | 2149 | 1908 | 22076 | 7372 | 8686 | 1304 |
| 4082 | 11803 | 5314 | 107 | 7359 | 22470 | 7372 | 22827 |
| 15698 | 30317 | 4685 | 14696 | 30388 | 8671 | 29956 | 15705 |
| 1417 | 26905 | 25809 | 28347 | 26277 | 7897 | 20240 | 21519 |
| 12437 | 1108 | 27106 | 18743 | 24144 | 10685 | 25234 | 30155 |
| 23005 | 8267 | 9917 | 7994 | 9694 | 2149 | 10042 | 27705 |
| 15930 | 29748 | 8635 | 23645 | 11738 | 24591 | 20240 | 27212 |
| 27486 | 9741 | 2149 | 29329 | 2149 | 5501 | 14015 | 30155 |
| 18154 | 22319 | 27705 | 20321 | 23254 | 13624 | 3249 | 5443 |
| 2149 | 16975 | 16087 | 14600 | 27705 | 19386 | 7325 | 26277 |
| 19554 | 23614 | 7553 | 4734 | 8091 | 23973 | 14015 | 107 |
| 3183 | 17347 | 25234 | 4595 | 21498 | 6360 | 19837 | 8463 |
| 6000 | 31280 | 29413 | 2066 | 369 | 23204 | 8425 | 7792 |
| 25973 | 4477 | 30989 | | | | | |

### 3.13.5 A small RSA cipher challenge (2)

The following task is a corrected version from the excellent book written by Prof. Yan [Yan2000, Example 3.3.7, p. 318]. However, it is not the result that is important here but rather the individual steps of the solution, that is, the explanation of the cryptanalysis[100].

There are three tasks with completely different degrees of difficulty here. In each case we know the cipher text and the public key $(e, n)$:

**(a)** Known plaintext: find the secret key $d$ using the additionally known original message.

**(b)** Cipher text only: find $d$ and the plaintext.

**(c)** Calculate the RSA modulus, in other words factorisation (with no knowledge of the message).

[99] The numbers of this table are in the online-help "Example illustrating the RSA demonstration" of CrypTool.

[100] The method of solving the problem is outlined in the scenario of the online help to CrypTool and in the CrypTool presentation. If anyone sends us a well prepared exact method of solving the problem, we would be pleased to include it in the documentation.

$n = 63978486879527143858831415041, \; e = 17579$

**Message[101]:**

```
1401202118011200,
1421130205181900,
0118050013010405,
0002250007150400
```

**Cipher:**

```
45411667895024938209259253423,
16597091621432020076311552201,
46468979279750354732637631044,
32870167545903741339819671379
```

**Comments:**

The original message consisted of a sentence containing 31 characters (coded with the capital letters alphabet from section 3.12.2). Each group of 16 decimal numbers is then combined to form one number (the last number is filled with zeros). These numbers are raised to the power of $e$.

When you decrypt the message you must fill the calculated numbers with leading zeros in order to obtain plaintext.

This needs to be stressed because the type of padding is extremely important during implementation and standardisation for interoperable algorithms.

[101] The numbers of this table are in the online help "Example illustrating the RSA demonstration" of CrypTool.

# References


[Agrawal2002] M. Agrawal, N. Kayal, N. Saxena,
*PRIMES in P*, August 2002
http://www.cse.iitk.ac.in/news/primality.html.

[Bartholome1996] A. Bartholome, J. Rung, H. Kern,
*Zahlentheorie für Einsteiger*, Vieweg 1995, 2nd edition 1996.

[Bauer1995] Friedrich L. Bauer,
*Entzifferte Geheimnisse*, Springer, 1995.

[Bauer2000] Friedrich L. Bauer,
*Decrypted Secrets*, Springer 1997, 2nd edition 2000.

[Bernstein2001] D. J. Bernstein,
*Circuits for integer factorization: a proposal*,
http://cr.yp.to/papers/nfscircuit.ps
http://cr.yp.to/djb.html.

[Beutelspacher1996] Albrecht Beutelspacher,
*Kryptologie*, Vieweg 1987, 5th edition 1996.

[Bourseau2002] F. Bourseau, D. Fox, C. Thiel,
*Vorzüge und Grenzen des RSA-Verfahrens*,
in Datenschutz und Datensicherheit (DuD) 26/2002, pp 84-89.

[Brands2002] Gilbert Brands,
*Verschlüsselungsalgorithmen – Angewandte Zahlentheorie rund um Sicherheitsprotokolle*,
Vieweg, 2002.

[Buchmann1999] Johannes Buchmann,
*Einführung in die Kryptographie*, Springer, 1999.

[Buhler1993] J.P. Buhler, H.W. Lenstra, C. Pomerance,
*Factoring integers with the number field sieve*,
In: A.K. Lenstra, H.W. Lenstra (Hrsg.): The Development of the Number Field Sieve,
Lecture Notes in Mathematics, Vol. 1554, Springer, Heidelberg 1993, pp 50−94.

[Eckert2003] Claudia Eckert,
*IT-Sicherheit: Konzepte-Verfahren-Protokolle*, Oldenbourg 2001, 2nd edition 2003.

[Ertel2001] Wolfgang Ertel,
*Angewandte Kryptographie*, Fachbuchverlag Leipzig FV 2001.

[GISA2002] GISA (German Information Security Agency)
*Recommendation for key length selection*, Bonn, Sep. 2002,
http://www.bsi.bund.de/esig/basics/techbas/krypto/bund02v7.pdf

A statement on these recommendations:
http://www.secorvo.de/publikat/stellungnahme-algorithmenempfehlung-020307.pdf

[Graham1994] Graham, Knuth, Patashnik,
*Concrete Mathemathics, a Foundation of Computer Science*,
Addison Wesley 1989, 6th printing 1990.

[Kippenhahn1997] Rudolph Kippenhahn,
*Verschlüsselte Botschaften – Geheimschrift, Enigma und Chipkarte*, Rowohlt, 1997.

[Kippenhahn1999] Rudolph Kippenhahn,
*Code Breaking – A History and Exploration*, Constable, 1999.

[Knuth1998] Donald E. Knuth,
*The Art of Computer Programming, vol 2: Seminumerical Algorithms*,
Addison-Wesley, 2nd edition 1998.

[Lenstra1993] A. Lenstra, H. Lenstra:
*The development of the Number Field Sieve*,
Lecture Notes in Mathematics 1554, Springer, New York 1993

[Lenstra2002] Arjen K. Lenstra, Adi Shamir, Jim Tomlinson, Eran Tromer,
*Analysis of Bernstein's Factorization Circuit*,
http://www.cryptosavvy.com/mesh.pdf

[Menezes2001] Alfred J. Menezes, Paul C. van Oorschot, Scott A. Vanstone
*Handbook of Applied Cryptography*, CRC Press 1997, 5th printing 2001.

[Pfleeger1997] Charles P. Pfleeger,
*Security in Computing*, Prentice-Hall, 2nd edition 1997.

[Pomerance1984] C. Pomerance,
*The quadratic sieve factoring algorithm*,
In: G.R. Blakley, D. Chaum (Hrsg.): Proceedings of Crypto '84, LNCS 196, Springer Berlin 1995, pp 169-182.

[RSA Security 2002] RSA Security,
*Has the RSA algorithm been compromised as a result of Bernstein's Paper?*,
April 8th, 2002,
http://www.rsasecurity.com/.

[Schneier1996] Bruce Schneier,
*Applied Cryptography, Protocols, Algorithms, and Source Code in C*,
Wiley and Sons, 2nd edition 1996.

[Schwenk2002] Jörg Schwenk,
*Sicherheit und Kryptographie im Internet*, Vieweg 2002.

[Sedgewick1990] Robert Sedgewick,
*Algorithms in C*, Addison-Wesley, 1990.

[Shamir2003] Adi Shamir, Eran Tromer,
*Factoring Large Numbers with the TWIRL Device*, Januar 2003,
http://www.wisdom.weizmann.ac.il/~tromer/.

[Silverman2000] Robert D. Silverman:
*A Cost-Based Security Analysis of Symmetric and Asymmetric Key Lengths*
In: RSA Laboratories Bulletin, No. 13, April 2000, p. 1-22

[Stinson1995] Douglas R. Stinson,
*Cryptography - Theory and Practice*, CRC Press, 1995.

[Welschenbach2001] Welschenbach, Michael,
*Kryptographie in C und C++*, Springer 2001.

[Wiles1994] Wiles, Andrew,
*Modular elliptic curves and Fermat's Last Theorem*,
in Annals of Mathematics 141 (1995).

[Wolfenstetter1998] Albrecht Beutelspacher, Jörg Schwenk, Klaus-Dieter Wolfenstetter,
*Moderne Verfahren in der Kryptographie*, Vieweg 1995, 2nd edition 1998.

[Yan2000] Song Y. Yan,
*Number Theory for Computing*, Springer, 2000.

### Web links

1. Ron Knott's Fibonacci page,
   Here, everything revolves around Fibonacci numbers.
   http://www.mcs.surrey.ac.uk/personal/R.Knott/Fibonacci/fib.html

2. CrypTool (Version 1.3.04, 2003),
   Freeware to illustrate cryptography
   http://www.cryptool.de,
   http://www.cryptool.org,
   http://www.cryptool.com

3. Mathematica,
   Commercial mathematics package
   http://www.wolfram.com

4. LiDIA,
   Extensive library containing number-theory functions and the LC interpreter
   http://www.informatik.tu-darmstadt.de/TI/LiDIA

5. BC,
   Interpreter with number-theory functions
   http://www.maths.uq.edu.au/~krm/gnubc.html

6. Pari-GP,
   Excellent, fast, free interpreter with number theoretical functions
   http://www.parigp-home.de and http://www.parigp-home.com

7. Only after I had completed this article, did I come across the website of Mr. Münchenbach, which interactively and didactically uses elementary number theory to provide a sophisticated description of the fundamental mathematical thought processes. It was created for a teaching project in the 11th grade of the technical grammar school (unfortunately only available in German):
   http://www.hydrargyrum.de/kryptographie

8. Once again only after finishing this I happened upon the web site of Mr. Wagner, who is responsible for the development of the curriculum of computer science in one of the German federal states (Länder). Here you can get hold of (but only in German) a collection of texts and (Java-)programs:
   http://www.hom.saar.de/~awa/kryptolo.htm

9. GISA,
   German Information Security Agency
   http://www.bsi.bund.de

10. Factorisation records and challenges
    http://www.crypto-world.com/

http://www.uni-bonn.de/Aktuelles/Pressemitteilungen/pm02/pm035-02.html
http://www.ercim.org/publication/Ercim_News/enw49/franke.html, 2002-01
http://www.rsasecurity.com/rsalabs/challenges/factoring/numbers.html

11. The Cunningham Project,
http://www.cerias.purdue.edu/homes/ssw/cun/

### Acknowledgments

I would like to take this opportunity to thank

- Mr. Henrik Koy for making many very useful suggestions, for the very constructive proof-reading this article and for helping with TeX.
  Mr. Koy developed the functions and the complex dialog box of the RSA cryptosystem, which enable you to execute the RSA samples of this article.

- Jörg-Cornelius Schneider for his TeX support und for the many cases where he helped when facing programming or design problems.

- Dr. Illies for pointing me to Pari-GP .

## Appendix A: the greatest common divisor (gcd) of whole numbers and the two Algorithms of Euclid

1. The greatest common divisor of two natural numbers $a$ and $b$ is an important value that can be calculated very quickly. Here we make use of the fact that if a number $c$ divides the numbers $a$ and $b$ (i.e. there exists an $a'$ and a $b'$ such that $a = a' * c$ and $b = b' * c$), then $c$ also divides the remainder $r$ of $a/b$. In short notion we can write: If $c$ divides $a$ and $b$ it follows that $c$ divides $r = a - \lfloor a/b \rfloor * b$[102]. As the latter statement is valid for each common divisor $c$ of $a$ and $b$ it follows that:

$$gcd(a, b) = gcd(a - \lfloor a/b \rfloor * b, b).$$

   Using this information, the algorithm for calculating the gcd of two numbers can be written as follows (in pseudo code):

```
INPUT: a,b != 0
1. if ( a < b ) then  x = a; a = b; b = x; // Swap a and b (a > b)
2. a = a - int(a/b) * b                    // a is smaller than b, the
                                           // gcd(a, b) is unchanged
3. if ( a != 0 ) then goto 1.              // a falls after each step and
                                           // the algorithm ends when a==0.
OUTPUT "gcd(a,b) = " b    // b is the gcd of the original a and b
```

2. However, to other relationships can be derived from the gcd: For this, we need the set of equations for $a$ and $b$:

$$\begin{aligned} a &= 1 * a + 0 * b \\ b &= 0 * a + 1 * b, \end{aligned}$$

   or, in matrix notation:

$$\begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} * \begin{pmatrix} a \\ b \end{pmatrix}.$$

   We summarise this information in the extended matrix:

$$\begin{pmatrix} a & | & 1 & 0 \\ b & | & 0 & 1 \end{pmatrix}$$

   If we apply the above gcd algorithm to this matrix, we obtain the extended gcd algorithm:

[102] The Gauss bracket $\lfloor x \rfloor$ of a real number $x$ is defined via: $\lfloor x \rfloor$ is the next integer less or equal $x$.

`INPUT:` $a, b \neq 0$

`0.` $x_{1,1} := 1, x_{1,2} := 0, x_{2,1} := 0, x_{2,2} := 1$

`1.` $\begin{pmatrix} a & | & x_{1,1} & x_{1,2} \\ b & | & x_{2,1} & x_{2,2} \end{pmatrix} := \begin{pmatrix} 0 & 1 \\ 1 & -\lfloor a/b \rfloor * b \end{pmatrix} * \begin{pmatrix} a & | & x_{1,1} & x_{1,2} \\ b & | & x_{2,1} & x_{2,2} \end{pmatrix}$.

`2. if (b != 0) then goto 1.`

`OUTPUT:` "gcd$(a,b) = a * x + b * y$: ", "gcd$(a,b)$ = " $b$, "$x$ =" $x_{2,1}$, "$y$ =" $x_{2,2}$

Since this algorithm only performs linear transformations, the same equations always apply

$$\begin{aligned} a &= x_{1,1} * a + x_{1,2} * b \\ b &= x_{2,1} * a + x_{2,2} * b, \end{aligned}$$

and we have the extended gcd equation at the end of the algorithm[103]:

$$gcd(a,b) = a * x_{2,1} + b * x_{2,2}.$$

**Example:**
Using the extended gcd we can determine for $e = 37$ the multiplicative inverse number $d$ to modulo 3588 (i.e. $37 * d \equiv 1 \pmod{3588}$):

`0.` $\begin{pmatrix} 3588 & | & 1 & 0 \\ 37 & | & 0 & 1 \end{pmatrix}$

`1.` $\begin{pmatrix} 37 & | & 1 & 0 \\ 36 & | & 0 & -96 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & -(\lfloor 3588/36 \rfloor = 96) * 37 \end{pmatrix} * \begin{pmatrix} 3588 & | & 1 & 0 \\ 37 & | & 0 & 1 \end{pmatrix}$.

`2.` $\begin{pmatrix} 36 & | & 1 & -96 \\ 1 & | & -1 & 97 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & -(\lfloor 37/36 \rfloor = 1) * 36 \end{pmatrix} * \begin{pmatrix} 37 & | & 1 & 0 \\ 36 & | & 0 & -96 \end{pmatrix}$.

`3.` $\begin{pmatrix} \mathbf{1} & | & \mathbf{-1} & \mathbf{97} \\ 0 & | & 37 & -3588 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & -(\lfloor 36/1 \rfloor = 36) * 1 \end{pmatrix} * \begin{pmatrix} 36 & | & 1 & -96 \\ 1 & | & -1 & 97 \end{pmatrix}$.

`OUTPUT:`
gcd$(37, 3588) = a * x + b * y$: gcd$(37, 3588) = 1$, $x = -1$, $y = 97$.

Thus

(a) 37 and 3588 are relatively prime (37 has an inverse modulo 3588).

(b) $37 * 97 = (1 * 3588) + 1$ in other words $37 * 97 \equiv 1 \pmod{3588}$. and therefore the number 97 is the multiplicative inverse to 37 modulo 3588.

[103] By termination of the gcd algorithm, the program variables $a$ and $b$ contain the values $a = 0$ and $b = gcd(a,b)$. Please keep in mind, that the program variables are different to the numbers $a$ and $b$ and that they are only relevant for the scope of the algorithm.

## Appendix B: Forming closed sets

The property of closeness is always defined in relation to an operation in a set. The following shows how to construct the "closed set" $G$ with respect to the operation + (mod 8) for a given initial set $G_0$:

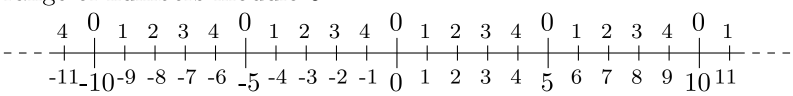
$$
\begin{aligned}
G_0 &= \{2,3\} \text{ addition of the numbers in } G_0 \text{ determines further numbers :} \\
& \quad 2+3 \equiv 5 \pmod 8 = 5 \\
& \quad 2+2 \equiv 4 \pmod 8 = 4 \\
& \quad 3+3 \equiv 6 \pmod 8 = 6 \\
G_1 &= \{2,3,4,5,6\} \text{ addition of the numbers in } G_1 \text{ determines :} \\
& \quad 3+4 \equiv 7 \pmod 8 = 7 \\
& \quad 3+5 \equiv 8 \pmod 8 = 0 \\
& \quad 3+6 \equiv 9 \pmod 8 = 1 \\
G_2 &= \{0,1,2,3,4,5,6,7\} \text{ addition of the numbers in } G_2 \textit{ does not extend the set}! \\
G_3 &= G_2 \text{ we say : } G_2 \text{ is closed for addition } \pmod 8.
\end{aligned}
$$

End of forming a closed set.

## Appendix C: Comments on modulo subtraction

Comment on subtraction modulo 5: $2-4 \equiv -2 \equiv 3$ mod 2. It is therefore not true modulo 5 that $-2 = 2$ ! People often make the mistake of equating this. You can show this clearly if you place the permutation $(0,1,2,3,4)$ in $\mathbb{Z}_5$, for example from $-11$ to $+11$, over the range of numbers in $\mathbb{Z}$.

range of numbers modulo 5




range of numbers in $\mathbb{Z}$

## Appendix D: Examples using Mathematica and Pari-GP

This appendix gives you the source code to compute the tables and examples using Mathematica or the free software Pari-GP.

### Multiplication table modulus $m$

The multiplication tables modulo $m = 17$ for $a = 5$ and $a = 6$ on page 52 can be computed in Mathematica with the following commands:

```
m = 17; iWidth = 18; iFactor1 = 5; iFactor2 = 6;
Print[ ''i '', Table[ i, {i, 1, iWidth} ] ];
Print[ iFactor1, ''*i '', Table[ iFactor1*i, {i, 1, iWidth } ] ];
Print[ ''Remainder '', Table[ Mod[iFaktor1*i, m], {i, 1, iWidth } ] ];
Print[ iFactor2, ''*i '', Table[ iFactor2*i, {i, 1, iWidth } ] ];
Print[ ''Remainder '', Table[ Mod[iFactor2*i, m], {i, 1, iWidth } ] ];
```

Pari-GP computes the tables via:

`m=17; iWidth=18; iFactor1=5; iFactor2=6;`

`matrix(1,iWidth, x,y, iFactor1*y)` yields
`[5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80 85 90]`

`matrix(1,iWidth, x,y, (iFactor1*y)%m )` yields
`[5 10 15 3 8 13 1 6 11 16 4 9 14 2 7 12 0 5]`

Note: Pari-GP generates when using the `Mod` function compound `Mod` objects, which are displayed as shown below:

```
matrix(1,iWidth, x,y, Mod(iFactor1*y, m))
[Mod(5, 17) Mod(10, 17) Mod(15, 17) Mod(3, 17) Mod(8, 17) Mod(13, 17) Mod(1, 17)
 Mod(6, 17) Mod(11, 17) Mod(16, 17) Mod(4, 17) Mod(9, 17) Mod(14, 17) Mod(2, 17)
 Mod(7, 17) Mod(12, 17) Mod(0, 17) Mod(5, 17)]
```

From a `Mod` object you can get back the components with the `component` or `lift` function:

```
component(Mod(5, 17),1) → 17
component(Mod(5, 17),2) → 5
component(Mod(17,5), 1) → 5
component(Mod(17,5), 2) → 2
lift(Mod(17,5)) → 2
```

The other multiplication table examples modulo 13 and modulo 12 on page 52 can computed by replacing `m=17` with `m=13` and `m=12` respectively.

**Fast exponentiation**

The fast exponentiation modulo $m$ belongs to the built in functions of Mathematica and Pari-GP. Using those programs you can comprehend the idea of the square and multiply method. With Mathematica you can compute the exponentiations of the example on page 55 as follows:

```
Mod[{87^43, 87^2, 87^4, 87^8, 87^16, 87^32}, 103] = {85, 50, 28, 63, 55, 38}.
```

and in Pari-GP the syntax is:

```
Mod([87^43,87^2,87^4,87^8,87^16,87^32],103)
```

**Multiplicative order and primitive roots**

The order $ord_m(a)$ of a number $a$ in the multiplicative group $Z_m^*$ is the smallest number $i \geq 1$, for with $a^i \equiv 1 \bmod m$ holds. For the example on page 63 you can make Mathematica print all exponentiations $a^i$ mod 11 using the following syntax:

```
m=11;  Table[ Mod[a^i, m], {a, 1, m-1}, {i, 1, m-1} ]
```

Equivalent Pari-GP syntax:

```
m=11; matrix(10,10, x,y, (x^y)%m )
```

The table on page 64 gives examples for the order modulo 45 $ord_{45}(a)$ and the Euler number $J(45)$. Mathematica can be used to create this table with the following program (please note that Print cannot be used inside of Do-loops and each Print outputs a newline).

```
m = 45;
Do[ Print[ Table[ Mod[a^i, m], {i, 1, 12} ],
'', '', MultiplicativeOrder[a, m, 1],
'', '', EulerPhi[m] ],
{a, 1, 12} ];
```

Here is the corresponding Pari-GP syntax:

```
m=45;
matrix(12,14, x,y,
       if( y<=12, (x^y)%m,
       if( y==13, if( gcd(x,m)==1, znorder(Mod(x,m)), "--"),
       eulerphi(m))))
```

`znorder(Mod(x,m))` can only be calculated if $x$ is relatively prime to $m$, which can be checked with `gcd(x,m)`.

Performance can be improved by using `Mod(x,m)^y` instead of `(x^y)%m`.

Loops are also supported by Pari-GP. When you remove the table formatting the result looks like this:

```
for( x=1,12,
     for(y=1,12, print(Mod(x^y,m)));
     if(gcd(x,m)==1, print(znorder(Mod(x,m))), print("--"));
     print(eulerphi(m)))
```

The third example on page 65 displays exponentiations $a^i$ mod 46 as well as the order $ord_{46}(a)$.

Mathematica can create this table with the following loop:

```
m = 46;
Do[ Print[ Table[ Mod[a^i, m], {i, 1, 23} ],
'', '', MultiplicativeOrder[a, m, 1]
{a, 1, 23} ] ];
```

In Pari-GP the syntax looks like this:

```
m=46;
matrix(23,24, x,y,
       if( y<=23, (x^y)%m,
       if( y==24, if( gcd(x,m)==1, znorder(Mod(x,m)), "--"))))
```

### RSA examples

This section list the source code of the RSA examples in section "The RSA procedure with actual numbers" using Mathematica and Pari-GP syntax.

**Example on page 82.**
The RSA exponentiation $M^{37}$ mod 3713 on message $M = 120$ can be calculated in Mathematica like this: `PowerMod[120, 37, 3713]`.
Here is the corresponding Pari-GP syntax:
`Mod(120,3713)^37 or Mod(120^37,3713)`.

**Example on page 83.**
The factorisation of $J(256, 027) = 255, 016 = 2^3 * 127 * 251$ can be calculated with Mathematica like this: `FactorInteger[255016]= {{2,3}, {127,1}, {251,1}}`.
Pari-GP does the same with:
`factor(255016)`.

**Example on page 83.**
Mathematica can do RSA encryption with the command:

```
PowerMod[{82, 83, 65, 32, 119, 111, 114, 107, 115, 33}, 65537, 256027]}
```
Pari-GP needs the following syntax:
```
vecextract( [Mod(82,256027)^65537, Mod(83,256027)^65537, Mod(65,256027)^65537,
Mod(32,256027)^65537, Mod(119,256027)^65537, ...])
```

**Remarks on using `Mod` in Pari-GP:**
`Mod(82,256027)^65537` is much faster than
– `Mod(82^65537, 256027)` and
– `(82^65537) % 256027`.

**Example on page 84.**
Mathematica can do RSA encryption with the following command:
```
PowerMod[{21075, 16672, 30575, 29291, 29473}, 65537, 256027]
```
The same calculation with Pari-GP:
```
vecextract( [Mod(21075,256027)^65537, Mod(16672,256027)^65537,
Mod(30575,256027)^65537, Mod(29291,256027)^65537,
Mod(29473,256027)^65537], 31)
```

**Example on page 85.**
RSA encryption using Mathematica:
```
PowerMod[{82083, 65032, 119111, 114107, 115033}, 65537, 256027]
```
RSA encryption with Pari-GP:
```
vecextract( [Mod(82083,256027)^65537, Mod(65032,256027)^65537,
Mod(119111,256027)^65537, Mod(114107,256027)^65537,
Mod(115033,256027)^65537], 31)
```

**Appendix E: List of the formulated definitions and theorems**

| | Short description | Page |
|---|---|---|
| Definition 3.1 | prime numbers | 43 |
| Definition 3.2 | composite numbers | 43 |
| Theorem 3.1 | factors of composite numbers | 44 |
| Theorem 3.2 | 1. fundamental theorem of number theory | 44 |
| Definition 3.3 | divisibility | 45 |
| Definition 3.4 | remainder class $r$ modulo $m$ | 45 |
| Definition 3.5 | congruent | 46 |
| Theorem 3.3 | congruence with difference | 46 |
| Theorem 3.4 | multiplicative inverse | 51 |
| Theorem 3.5 | exhaustive permutation | 52 |
| Theorem 3.6 | power mod $m$ | 54 |
| Definition 3.6 | $\mathbb{Z}_n$ | 56 |
| Definition 3.7 | $\mathbb{Z}_n^*$ | 57 |
| Theorem 3.7 | multiplicative inverse in $\mathbb{Z}_n^*$ | 58 |
| Definition 3.8 | Euler function $J(n)$ | 59 |
| Theorem 3.8 | $J(p)$ | 59 |
| Theorem 3.9 | $J(p * q)$ | 59 |
| Theorem 3.10 | $J(p_1 * \cdots * p_k)$ | 59 |
| Theorem 3.11 | $J(p_1^{e_1} * \cdots * p_k^{e_k})$ | 59 |
| Theorem 3.12 | little Fermat | 60 |
| Theorem 3.13 | Euler-Fermat theorem | 60 |
| Definition 3.9 | multiplicative order $\mathrm{ord}_m(a)$ | 63 |
| Definition 3.10 | primitive root of $m$ | 63 |
| Theorem 3.14 | exhausting of all possible values | 65 |

# 4 The Mathematical Ideas behind Modern Cryptography

(Oyono R. / Esslinger B., Sep. 2000, Updates Nov. 2000, Feb. 2003)

## 4.1 One way functions with trapdoor and complexity classes

A **one way function** is a (one-to-one) function that can be calculated efficiently, but whose inverse is extremely complicated and practically impossible to calculate.

To put it more precisely: A one way function is a mapping $f$ from a set $X$ to a set $Y$, such that $f(x)$ can be calculated easily for each element $x$ of $X$, whereas for (almost) every $y$ from $Y$ it is practically impossible to find an inverse image $x$ (i.e. an $x$ where $f(x) = y$).

An everyday example of a one way function is a telephone book: the function to be performed is to assign a name to the corresponding telephone number. This can be done easily due to the fact that the names are sorted alphabetically. However, the inverse function - assigning a name to a given number - is obviously difficult if you only have a telephone book available.

One way functions play a decisive role in cryptography. Almost all cryptographic terms can be rephrased using the term one way function. Let's take for example public key encryption (asymmetric cryptography):

Each subscriber $T$ to the system is assigned a private key $d_T$ and what is known as a public key $e_T$. These keys must have the following property (public key property):
For an opponent who knows the public key $e_T$, it is practically impossible to determine the private key $d_T$.

In order to construct useful public key procedures, therefore, we look for a one way function that is "easy" to calculate in one direction , but is "difficult" (practically impossible) to calculate in the other direction, provided that a particular piece of additional information (trapdoor) is not available. This additional piece of information allows the inverse to be found efficiently. Such functions are called **trapdoor one way functions**. In the above case, $d_T$ is the trapdoor information.

In this process, we describe a problem as "easy" if it can be solved in polynomial time as a function of the length of the input. If the length of the input is $n$ bits, then the time for calculating the function is proportional to $n^a$, where $a$ is a constant. We say that the complexity of such problems is $O(n^a)$ t (Landau- or Big-O notation).

If you compare 2 functions $2^n$ and $n^a$, where $a$ is a constant, then there always exists a value for $n$, from which for all further $n$ applies: $n^a < 2^n$. The function $n^a$ has a lower complexity. Sample: for $a = 5$ the following applies: from the length $n = 23$, $2^n$ is greater than $n^5$; for further $n$ $2^n$ clearly increases more quickly [($2^{22} = 4,194,304$, $22^5 = 5,153,632$); ($2^{23} = 8,388,608$, $23^5 = 6,436,343$); ($2^{24} = 16,777,216$, $24^5 = 7,962,624$);].

The term "practically impossible" is slightly less precise. In general, we can say that a problem cannot be solved efficiently, if the time required to solve it increases more quickly than the polynomial time as a function of the size of the input. If, for example, the length of the input is $n$ bits and the time required for calculating the function is proportional to $2^n$, then the following currently applies: the function practically cannot be calculated for $n > 80$.

In order to develop a public key procedure that can be implemented in practice, it is therefore necessary to discover a suitable trapdoor one way function.

In order to tidy things up among this confusing multitude of possible problems and their complexities, we group problems with similar complexities into classes.

The most important complexity classes are the classes **P** and **NP**:

- The class **P**: This class contains those problems that can be solved in a polynomial amount of time.
- The class **NP**: The definition of this class doesn't look at the time required to solve a problem, but rather at the time required to verify a given solution. The class **NP** consists of those problems for which a given solution can be verified in a polynomial amount of time. Hereby, the term **NP** "non-deterministic" means polynomial and is based on a calculation model, i.e. on a computer that only exists in theory and can "guess" correct solutions non-deterministically then verify them in polynomial time.

The class **P** is contained in the class **NP**. A well-known unsolved problem is the question whether or not $\mathbf{P} \neq \mathbf{NP}$ is true, i.e. whether or not **P** is a true subset. An important property of the class **NP** is that it also contains what are known as "**NP**-complete" problems. These are problems that represent the class **NP** as follows: If a "good" algorithm for such a problem exists, then "good" algorithms exist for all problems from **NP**. In particular: if **P** only contained one complete problem, i.e. if a polynomial solution algorithm existed for this problem, then **P**would be equal to **NP**. In this sense, the **NP**-complete problems are the most difficult problems in **NP**.

Many cryptographic protocols are formed in such a way that the "good" subscribers only have to solve problems from **P**, whereas a perpetrator is faced with problems from **NP**.

Unfortunately, we do not yet know whether one way functions actually exist. However, we can prove that one way functions exist if and only if $\mathbf{P} \neq \mathbf{NP}$ [Balcazar1988, S.63].

Mathematicians have again and again claimed to have proven the equivalence, e.g.

http://www.geocities.com/st_busygin/clipat.html),

but so far the claims have always turned out to be false.

A number of algorithms have been suggested for public key procedures. In many cases - although they at first appeared promising - it was discovered that they could be solved polynomially. The most famous failed applicant is the knapsack with trapdoor, suggested by Ralph Merkle [Merkle1978].

## 4.2 Knapsack problem as a basis for public key procedures

### 4.2.1 Knapsack problem

You are given $n$ objects $G_1, \ldots, G_n$ with the weights $g_1, \ldots g_n$ and the values $w_1, \cdots, w_n$. The aim is to carry away as much as possible in terms of value while restricted to an upper weight limit $g$. You therefore need to find a subset of $\{G_1, \cdots, G_n\}$, i.e. $\{G_{i_1}, \ldots, G_{i_k}\}$, so that $w_{i_1} + \cdots + w_{i_k}$ is maximised under the condition $g_{i_1} + \cdots + g_{i_k} \leq g$.

Such questions are called **NP**-complete problems (not deterministically polynomial) that are difficult to calculate.

A special case of the knapsack problem is:
Given the natural numbers $a_1, \ldots, a_n$ and $g$., find $x_1, \ldots, x_n \in \{0, 1\}$ where $g = \sum_{i=1}^{n} x_i a_i$ (i.e. where $g_i = a_i = w_i$ is selected). This problem is also called a **0-1 knapsack problem** and is identified with $K(a_1, \ldots, a_n; g)$.

Two 0-1 knapsack problems $K(a_1, \ldots, a_n; g)$ and $K(a'_1, \ldots, a'_n; g')$ are called congruent if two co-prime numbers $w$ and $m$ exist in such a way that

1. $m > \max\{\sum_{i=1}^{n} a_i, \sum_{i=1}^{n} a'_i\}$,
2. $g \equiv wg' \mod m$,
3. $a_i \equiv wa'_i \mod m$ for all $i = 1, \ldots, n$.

**Comment:** Congruent 0-1 knapsack problems have the same solutions. No quick algorithm is known for clarifying the question as to whether two 0-1 knapsack problems are congruent.

A 0-1 knapsack problem can be solved by testing the $2^n$ possibilities for $x_1, \ldots, x_n$. The best method requires $O(2^{n/2})$ operations, which for $n = 100$ with $2^{100} \approx 1.27 \cdot 10^{30}$ and $2^{n/2} \approx 1.13 \cdot 10^{15}$ represents an insurmountable hurdle for computers. However, for special $a_1, \ldots, a_n$ the solution is quite easy to find, e.g. for $a_i = 2^{i-1}$. The binary representation of $g$ immediately delivers $x_1, \ldots, x_n$. In general, the a 0-1 knapsack problem can be solved easily if a permutation[104] $\pi$ of $1, \ldots, n$ exists with $a_{\pi(j)} > \sum_{i=1}^{j-1} a_{\pi(i)}$. If, in addition, $\pi$ is the identity, i.e. $\pi(i) = i$ for $i = 1, 2, \ldots, n$, then the sequence $a_1, \ldots, a_n$ is said to be super-increasing. The following algorithm solves the knapsack problem with a super-increasing sequence in the timeframe of $O(n)$.

[104] A permutation $\pi$ of the numbers $1, \ldots, n$ is a change in the order in which these numbers are listed. For example, a permutation $\pi$ of $(1, 2, 3)$ is $(3, 1, 2)$, i.e. $\pi(1) = 3$, $\pi(2) = 1$ and $\pi(3) = 2$.

```
for i = n to 1 do
    if T ≥ a_i then
        T := T − s_i
        x_i := 1
    else
        x_i := 0
if T = 0 then
    X := (x_1, ..., x_n) is the solution.
else
    No solution exists.
```

**Algorithm 1.** Solving knapsack problems with super-increasing weights

### 4.2.2 Merkle-Hellman knapsack encryption

In 1978, Merkle and Hellman [Merkle1978] specified a public key encryption procedure that is based on "defamiliarising" the easy 0-1 knapsack problem with a super-increasing sequence into a congruent one with a super-increasing sequence. It is a block ciphering that ciphers an $n$-bit plaintext each time it runs. More precisely:

Let $(a_1, \dots, a_n)$ be super-increasing. Let $m$ and $w$ be two co-prime numbers with $m > \sum_{i=1}^{n} a_i$ and $1 \leq w \leq m-1$. Select $\bar{w}$ with $w\bar{w} \equiv 1 \mod m$ the modular inverse of $w$ and set $b_i := wa_i \mod m$, $0 \leq b_i < m$ for $i = 1, \dots, n$, and verify whether the sequence $b_1, \dots b_n$ is not super-increasing. A permutation $b_{\pi(1)}, \dots, b_{\pi(n)}$ of $b_1, \dots, b_n$ is then published and the inverse permutation $\mu$ to $\pi$ is defined secretly. A sender writes his/her message in blocks $(x_1^{(j)}, \dots, x_n^{(j)})$ of binary numbers $n$ in length, calculates

$$g^{(j)} := \sum_{i=1}^{n} x_i^{(j)} b_{\pi(i)}$$

and sends $g^{(j)}, (j = 1, 2, \dots)$.
The owner of the key calculates

$$G^{(j)} := \bar{w} g^{(j)} \mod m, \quad 0 \leq G^{(j)} < m$$

and obtains the $x_{\mu(i)}^{(j)} \in \{0, 1\}$ (and thus also the $x_i^{(j)}$) from

$$\begin{aligned} G^{(j)} &\equiv \bar{w} g^{(j)} = \sum_{i=1}^{n} x_i^{(j)} b_{\pi(i)} \bar{w} \equiv \sum_{i=1}^{n} x_i^{(j)} a_{\pi(i)} \mod m \\ &= \sum_{i=1}^{n} x_{\mu(i)}^{(j)} a_{\pi(\mu(i))} = \sum_{i=1}^{n} x_{\mu(i)}^{(j)} a_i \mod m, \end{aligned}$$

by solving the easier 0-1 knapsack problems $K(a_1, \dots, a_n; G^{(j)})$ with super-increasing sequence $a_1, \dots, a_n$.

**Merkle-Hellman procedure** (based on knapsack problems).

In 1982, Shamir [Shamir1982] specified an algorithm for breaking the system in polynomial time without solving the general knapsack problem. Len Adleman [Adleman1982] and Jeff Lagarias [Lagarias1983] specified an algorithm for breaking the twice iterated Merkle-Hellman knapsack encryption procedure in polynomial time. Ernst Brickell [Brickell1985] then specified an algorithm for breaking multiply iterated Merkle-Hellman knapsack encryption procedures in polynomial time. This made this procedure unsuitable as an encryption procedure. It therefore delivers a one way function whose trapdoor information (defamiliarisation of the 0-1 knapsack problem) could be discovered by an evesdropper.

## 4.3 Decomposition into prime factors as a basis for public key procedures

### 4.3.1 The RSA procedure[105]

As early as 1978, R. Rivest, A. Shamir, L. Adleman [RSA1978] introduced the most important asymmetric cryptography procedure to date.

Key generation:

Let $p$ and $q$ be two different prime numbers and $N = pq$. Let $e$ be any prime number relative to $\phi(N)$ , i.e. $\gcd(e, \phi(N)) = 1$. Using the Euclidean algorithm, we calculate the natural number $d < \phi(N)$, such that

$$ed \equiv 1 \mod \phi(N).$$

whereby $\phi$ is the **Euler phi Function**.
The output text is divided into blocks and encrypted, whereby each block has a binary value $x^{(j)} \leq N$.

Public key:

$$N, e.$$

Private key:

$$d.$$

Encryption:

$$y = e_T(x) = x^e \mod N.$$

Decryption:

$$d_T(y) = y^d \mod N.$$

**RSA procedure** (based on the factorisation problem).

**Comment:** The Euler phi function is defined as: $\phi(N)$ is the number of natural numbers that do not have a common factor with $N \quad x \leq N$. Two natural numbers $a$ and $b$ are co-prime if $\gcd(a, b) = 1$.

For the Euler phi function: $\phi(1) = 1, \phi(2) = 1, \phi(3) = 2, \phi(4) = 2, \phi(6) = 2, \phi(10) = 4, \phi(15) = 8$.
For example, $\phi(24) = 8$, because $|\{x < 24 : \gcd(x, 24) = 1\}| = |\{1, 5, 7, 11, 13, 17, 19, 23\}|$.

If $p$ is a prime number, then $\phi(p) = p - 1$.

If we know the various prime factors $p_1, \dots, p_k$ of $N$, then $\phi(N) = N \cdot (1 - \frac{1}{p_1}) \cdots (1 - \frac{1}{p_k})$[106].

[105] Using CrypTool v1.3 you gain practical experience with the RSA procedure via the menu **Indiv.Procedures \ RSA Demonstration \ RSA Cryptosystem**.

[106] Further formulas for the Euler phi function are in the article Introduction to Elementary Number Theory with Examples, chapter 3.8.

In the case of $N = pq$, $\phi(N) = pq(1 - 1/p)(1 - 1/q) = p(1 - 1/p)q(1 - 1/q) = (p - 1)(q - 1)$.

| $n$ | $\phi(n)$ | The natural numbers that are co-prime to $n$ and less than $n$. |
|---|---|---|
| 1 | 1 | 1 |
| 2 | 1 | 1 |
| 3 | 2 | 1, 2 |
| 4 | 2 | 1, 3 |
| 5 | 4 | 1, 2, 3, 4 |
| 6 | 2 | 1, 5 |
| 7 | 6 | 1, 2, 3, 4, 5, 6 |
| 8 | 4 | 1, 3, 5, 7 |
| 9 | 6 | 1, 2, 4, 5, 7, 8 |
| 10 | 4 | 1, 3, 7, 9 |
| 15 | 8 | 1, 2, 4, 7, 8, 11, 13, 14 |

The function $e_T$ is a one way function whose trapdoor information is the decomposition into primes of $N$.

At the moment, no algorithm is known that can factorise two prime numbers sufficiently quickly for extremely large values (e.g. for several hundred decimal places). The quickest algorithms known today [Stinson1995] factorise a compound whole number $N$ in a time period proportional to $L(N) = e^{\sqrt{\ln(N)\ln(\ln(N))}}$.

| $N$ | $10^{50}$ | $10^{100}$ | $10^{150}$ | $10^{200}$ | $10^{250}$ | $10^{300}$ |
|---|---|---|---|---|---|---|
| $L(N)$ | $1.42 \cdot 10^{10}$ | $2.34 \cdot 10^{15}$ | $3.26 \cdot 10^{19}$ | $1.20 \cdot 10^{23}$ | $1.86 \cdot 10^{26}$ | $1.53 \cdot 10^{29}$ |

As long as no better algorithm is found, this means that values of the order of magnitude 100 to 200 decimal places are currently safe. Estimates of the current computer technology indicate that a number with 100 decimal places could be factorised in approximately two weeks at justifiable cost. Using an expensive configuration (e.g. of around 10 million US dollars), a number with 150 decimal places could be factorised in about a year. A 200−digit number should remain impossible to factorise for a long time to come, unless there is a mathematical breakthrough. For example, it would take about 1000 years to decompose a 200-digit number into prime factors using the existing algorithms; this applies even if $10^{12}$ operations can be performed per second, which is beyond the performance of current technology and would cost billions of dollars in development costs. However, you can never be sure that there won't be a mathematical breakthrough tomorrow.

To this date, it has not been proved that the problem of breaking RSA is equivalent to the factorisation problem. Nevertheless, it is clear that the RSA procedure will no longer be safe if the factorisation problem is "solved".

### 4.3.2 Rabin public key procedure (1979)

In this case it has been shown that the procedure is equivalent to breaking the factorisation problem. Unfortunately, this procedure is susceptible to chosen-cipher text attacks.

Let $p$ and $q$ be two different prime numbers with $p, q \equiv 3 \mod 4$ and $n = pq$. Let $0 \leq B \leq n-1$.
Public key:
$$e = (n, B).$$
Private key:
$$d = (p, q).$$
Encryption:
$$y = e_T(x) = x(x+B) \mod n.$$
Decryption:
$$d_T(y) = \sqrt{y + B^2/4} - B/2 \mod n.$$

**Rabin procedure** (based on the factorisation problem).

Caution: Because $p, q \equiv 3 \mod 4$ the encryption is easy to calculate (if the key is known). This is not the case for $p \equiv 1 \mod 4$. In addition, the encryption function is not injective: There are precisely four different source codes that have $e_T(x)$ as inverse image: $x, -x-B, \omega(x+B/2)-B/2, -\omega(x+B/2)-B/2$, where $\omega$ is one of the four roots of unity. The source codes therefore must be redundant for the encryption to remain unique!

Backdoor information is the decomposition into prime numbers of $n = pq$.

## 4.4 The discrete logarithm as a basis for public key procedures

Discrete logarithms form the basis for a large number of algorithms for public- key procedures.

### 4.4.1 The discrete logarithm in $\mathbb{Z}_p$

Let $p$ be a prime number and let $g \in \mathbb{Z}_p^* = \{0, 1, \ldots, p-1\}$. Then the discrete exponential function base $g$ is defined as

$$e_g : k \longrightarrow y := g^k \mod p, \quad 1 \leq k \leq p-1.$$

The inverse function is called a discrete logarithm function $\log_g$; the following holds:

$$\log_g(g^k) = k.$$

The problem of the discrete logarithm (in $\mathbb{Z}_p^*$) is understood to be as follows:

$$\text{Given } p, g \text{ and } y, \text{ determine } k \text{ such that } y = g^k \mod p.$$

It is much more difficult to calculate the discrete logarithm than to evaluate the discrete exponential function (see chapter 3.9). There are several procedures for calculating the discrete logarithm [Stinson1995]:

| Name | Complexity |
|---|---|
| Baby-Step-Giant-Step | $O(\sqrt{p})$ |
| Silver-Pohlig-Hellman | polynomial in $q$, the greatest prime factor of $p-1$. |
| Index-Calculus | $O(e^{(1+o(1))\sqrt{\ln(p)\ln(\ln(p))}})$ |

### 4.4.2 Diffie-Hellman key agreement[107]

The mechanisms and algorithms of classical cryptography only take effect when the subscribers have already exchanged the secret key. In classical cryptography you cannot avoid exchanging secrets without encrypting them. Transmission safety here must be achieved using non-cryptographic methods. We say that we need a secret channel for exchanging secrets. This channel can be realised either physically or organisationally.

What is revolutionary about modern cryptography is, amongst other things, that you no longer need secret channels: You can agree secret keys using non-secret, i.e. public channels.

One protocol that solves this problem is that of Diffie and Hellman.

Two subscribers $A$ and $B$ want to agree on a joint secret key.
Let $p$ be a prime number and $g$ a natural number. These two numbers do not need to be secret.
The two subscribers then select a secret number $a$ and $b$ from which they calculate the values $\alpha = g^a \mod p$ and $\beta = g^b \mod p$. They then exchange the numbers $\alpha$ and $\beta$. To end with, the two subscribers calculate the received value to the power of their secret value to get $\beta^a \mod p$ and $\alpha^b \mod p$.
Thus

$$\beta^a \equiv (g^b)^a \equiv g^{ba} \equiv g^{ab} \equiv (g^a)^b \equiv \alpha^b \mod p$$

**Diffie-Hellman key agreement.**

[107] With CrypTool v1.3.04 this exchange protocol has been visualized: you can execute the single steps with concrete numbers using menu **Indiv. Procedures \ Diffie-Hellman Demonstration**.

The safety of the **Diffie-Hellman protocol** is closely connected to calculating the discrete logarithm mod $p$. It is even thought that these problems are equivalent.

### 4.4.3 ElGamal public key encryption procedure in $\mathbb{Z}_p^*$

By varying the Diffie-Hellman key agreement protocol slightly, you can obtain an asymmetric encryption algorithm. This observation was made by Taher ElGamal.

Let $p$ be a prime number such that the discrete logarithm in $\mathbb{Z}_p$ is difficult to compute. Let $\alpha \in \mathbb{Z}_p^*$ be a primitive element. Let $a \in \mathbb{N}$ and $\beta = \alpha^a \mod p$.

Public key:

$$p, \alpha, \beta.$$

Private key:

$$a.$$

Let $k \in \mathbb{Z}_{p-1}$ be a random number and $x \in \mathbb{Z}_p^*$ the plaintext.

Encryption:

$$e_T(x,k) = (y_1, y_2),$$

where

$$y_1 = \alpha^k \mod p$$

and

$$y_2 = x\beta^k \mod p.$$

Decryption:

$$d_T(y_1, y_2) = y_2(y_1^a)^{-1} \mod p$$

**ElGamal procedure** (based on the factorisation problem).

### 4.4.4 Generalised ElGamal public key encryption procedure

The discrete logarithm can be generalised in any number of finite groups $(G, \circ)$. The following provides several properties of $G$, that make the discrete logarithm problem difficult.

**Calculating the discrete exponential function** Let $G$ be a group with the operation $\circ$ and $g \in G$. The (discrete) exponential function base $g$ is defined as

$$e_g : k \longmapsto g^k, \quad \text{for all } k \in \mathrm{N}.$$

where

$$g^k := \underbrace{g \circ \ldots \circ g}_{k \text{ times}}.$$

The exponential function is easy to calculate:

**Lemma.**

*The power $g^k$ can be calculated in at most $2\log_2 k$ group operations.*

**Proof**

Let $k = 2^n + k_{n-1}2^{n-1} + \cdots + k_1 2 + k_0$ be the binary representation of $k$. Then $n \leq \log_2(k)$, because $2^n \leq k < 2^{n+1}$. $k$ can be written in the form $k = 2k' + k_0$ with $k' = 2^{n-1} + k_{n-1}2^{n-2} + \cdots + k_1$. Thus

$$g^k = g^{2k'+k_0} = (g^{k'})^2 g^{k_0}.$$

We therefore obtain $g^k$ from $g^{k'}$ by squaring and then multiplying by $g$. The claim is thus proved by induction to $n$. □

**Problem of the discrete logarithm**

| |
|---|
| Let $G$ by a finite group with the operation $\circ$. Let $\alpha \in G$ and $\beta \in H = \{\alpha^i : i \geq 0\}$.<br>We need to find a unique $a \in \mathrm{N}$ with $0 \leq a \leq \|H\| - 1$ and $\beta = \alpha^a$.<br>We define $a$ as $\log_\alpha(\beta)$. |

**Calculating the discrete logarithm** A simple procedure for calculating the discrete logarithm of a group element, that is considerably more efficient than simply trying all possible values for $k$, is the Baby-Step-Giant-Step algorithm.

**Theorem 4.1.** *[Baby-Step-Giant-Step algorithm] Let $G$ be a group and $g \in G$. Let $n$ be the smallest natural number with $|G| \leq n^2$. Then the discrete logarithm of an element $h \in G$ can be calculated base $g$ by generating two lists each containing $n$ elements and comparing these lists. In order to calculate these lists, we need $2n$ group operations.*

**Proof**

First create the two lists

Giant-Step list: $\{1, g^n, g^{2n}, \ldots, g^{n \cdot n}\}$,

Baby-Step list: $\{hg^{-1}, hg^{-2}, \ldots, hg^{-n}\}$.

If $g^{jn} = hg^{-i}$, i.e. $h = g^{i+jn}$, then the problem is solved. If the lists are disjoint, then $h$ cannot be represented as $g^{i+jn}, i, j \leq n,$. As all powers of $g$ are thus recorded, the logarithm problem does not have a solution. □

You can use the Baby-Step-Giant-Step algorithm to demonstrate that it is much more difficult to calculate the discrete logarithm than to calculate the discrete exponential function. If the numbers that occur have approximately 1000 bits in length, then you only need around 2000

multiplications to calculate $g^k$ but around $2^{500} \approx 10^{150}$ operations to calculate the discrete logarithm using the Baby-Step-Giant-Step algorithm.
In addition to the Baby-Step-Giant-Step algorithm, there are also numerous other procedures for calculating the discrete logarithm [Stinson1995].

**The theorem from Silver-Pohlig-Hellman** In finite Abelian groups, the discrete logarithm problem can be reduced to groups of a lower order.

**Theorem 4.2.** *[Silver-Pohlig-Hellman] Let $G$ be a finite Abelian group with $|G| = p_1^{a_1} p_2^{a_2} \cdot \ldots \cdot p_s^{a_s}$. The discrete logarithm in $G$ can then be reduced to solving logarithm problems in groups of the order $p_1, \ldots, p_s$.*

If $|G|$ contains a "dominant" prime factor $p$, then the complexity of the logarithm problem is approximately

$$O(\sqrt{p}).$$

Therefore, if the logarithm problem is to be made difficult, the order of the group used $G$ should have a large prime factor. In particular, if the discrete exponential function in the group $\mathbb{Z}_p^*$ is to be a one way function, then $p-1$ must be a large prime factor.

Let $G$ be a finite group with operation $\circ$, and let $\alpha \in G$, so that the discrete logarithm in $H = \{\alpha^i : i \geq 0\}$ is difficult, Let $a$ with $0 \leq a \leq |H| - 1$ and let $\beta = \alpha^a$.
Public key:

$$\alpha, \beta.$$

Private key:

$$a.$$

Let $k \in \mathbb{Z}_{|H|}$ be a random number and $x \in G$ be a plaintext.
Encryption:

$$e_T(x, k) = (y_1, y_2),$$

where

$$y_1 = \alpha^k$$

and

$$y_2 = x \circ \beta^k.$$

Decryption:

$$d_T(y_1, y_2) = y_2 \circ (y_1^a)^{-1}$$

**Generalised ElGamal procedure** (based on the factorisation problem).

Elliptic curves provide useful groups for public key encryption procedures.

## References


[Adleman1982] Adleman L.: *On breaking the iterated Merkle-Hellman public key Cryptosystem.* Advances in Cryptology, Proceedings of Crypto 82, Plenum Press 1983, 303-308.

[Balcazar1988] Balcazar J.L., Daaz J., Gabarr´ J.: *Structural Complexity I.* Springer Verlag, pp 63.

[Brickell1985] Brickell E.F.: *Breaking Iterated Knapsacks.* Advances in Cryptology: Proc. CRYPTO8́4, Lecture Notes in Computer Science, vol. 196, Springer-Verlag, New York, 1985, pp. 342-358.

[Lagarias1983] Lagarias J.C.: *Knapsack public key Cryptosystems and diophantine Approximation.* Advances in Cryptology, Proseedings of Crypto 83, Plenum Press.

[Merkle1978] Merkle R. and Hellman M.: *Hiding information and signatures in trapdoor knapsacks.* IEEE Trans. Information Theory, IT-24, 1978.

[RSA1978] Rivest R.L., Shamir A. and Adleman L.: *A Method for Obtaining Digital Signatures and Public Key Cryptosystems.* Commun. ACM, vol 21, April 1978, pp. 120-126.

[Shamir1982] Shamir A.: *A polynomial time algorithm for breaking the basic Merkle-Hellman Cryptosystem.* Symposium on Foundations of Computer Science (1982), 145-152.

[Stinson1995] Stinson D.R.: *Cryptography.* CRC Press, Boca Raton, London, Tokyo, 1995.


## Web links

1. http://www.geocities.com/st_busygin/clipat.html

# 5 Digital Signatures

(Schneider J. / Esslinger B. / Filipovics B. / Koy H., June 2002, Update: Feb. 2003)

The aim of digital signatures is to guarantee the following two points:

- User authenticity:
  It can be checked whether a message really does come from a particular person.
- Message integrity:
  It can be checked whether the message has been changed (on route).

An asymmetric technique is used again (see encryption procedures). Participants who wish to generate a digital signature for a document must possess a pair of keys. They use their secret key to generate signatures and the recipient uses the sender's public key to verify whether the signature is correct. As before, it must be impossible to use the public key to derive the secret key[108].

In detail, a *Signature procedure* looks like this:
Senders use their message and secret key to calculate the digital signature for the message. Compared to hand-written signatures, digital signatures therefore have the advantage that they also depend on the document to be signed. Signatures from one and the same participant are different unless the signed documents are completely identical. Even inserting a blank in the text would lead to a different signature. The recipient of the message would therefore detect any injury to the message integrity as this would mean that the signature no longer matches the document and is shown to be incorrect when verified.

The document is sent to the recipient together with the signature. The recipient can then use the sender's public key, the document and the signature to establish whether or not the signature is correct. In practice, however, the procedure we have just described has a decisive disadvantage. The signature is approximately as long as the document itself. To prevent an unnecessary increase in data traffic, and also for reasons of performance, we use a cryptographic hash function[109].

A cryptographic *hash function* maps a message of any length to a string of characters with a constant size (usually 128 or 160 bits), the hash value. It should be practically impossible, for a given number, to find a message that has precisely this number as hash value. Furthermore, it

---

[108] With CrypTool v1.3 you can also generate and check digital signatures:
using the submenus of the main menu **Digital Signatures** or
using menu **Indiv. Procedures \ RSA Demonstration \ Signature Demonstration**.

[109] Hash functions are implemented within CrypTool v1.3.04 at several places.
Using menu **Indiv. Procedures \ Hash** you have the possibilities

- to apply one of 6 hash functions to the content of the current window,
- to calculate the hash value of a file,
- to test, how changes to a text change the according hash value and
- to perform a simulation, how digital signatures could be attacked by a targeted search for hash value collisions.

should be practically impossible to find two messages with the same hash value. In both cases the final signature procedure would display weak points.

So far, no formal proof has been found that perfectly secure cryptographic hash functions exist. However, there are several good candidates that have not yet shown any weak points in practice (e.g. SHA-1 or RIPEMD-160).

The hash function procedure is as follows:
Rather than signing the actual document, the sender now first calculates the hash value of the message and signs this. The recipient also calculates the hash value of the message (the algorithm used must be known), then verifies whether the signature sent with the message is a correct signature of the hash value. If this is the case, the signature is verified to be correct. This means that the message is authentic, because we have assumed that knowledge of the public key does not enable you to derive the secret key. However, you would need this secret key to sign messages in another name.

Some digital signature schemes are based on asymmetric *encryption* procedures, the most prominent example being the RSA system, which can be used for signing by performing the private key operation on the hash value of the document to be signed.

Other digital signature schemes where developed exclusively for this purpose, as the DSA (Digital Signature Algorithm), and are not directly connected with a corresponding encryption scheme.

Both, RSA and DSA signature are discussed in more detail in the following two sections. After that we go one step further and show how digital signatures can be used to create the digital equivalent of ID cards. This is called Public Key Certification.

## 5.1 RSA signatures

As mentioned in the comment at the end of section 3.10.3 it is possible to perform the RSA private and public key operation in reverse order, i. e. raising $M$ to the power of $d$ and then to the power of $e$ (mod $N$) yields $M$ again. Based on this simple fact, RSA can be used as a signature scheme.

The RSA signature $S$ for a message $M$ is created by performing the private key operation:

$$S \equiv M^d \pmod N$$

In order to verify, the corresponding public key operation is performed on the signature $S$ and the result is compared with message $M$:

$$S^e \equiv (M^d)^e \equiv (M^e)^d \equiv M \pmod N$$

If the result matches the message $M$, then the signature is accepted by the verifier, otherwise the message has been tampered with, or was never signed by the holder of $d$.

As explained above, signatures are not performed on the message itself, but on a cryptographic hash value of the message. To prevent certain attacks on the signature procedure (alone or in combination with encryption) it is necessary to format the hash value before doing the exponentiation, as described in the PKCS#1 (Public Key Cryptography Standard #1 [PKCS1]). The fact that this standard had to be revised recently, after being in use for several years, can serve as an example of how difficult it is to get the details of cryptography right.

## 5.2 DSA signatures

In August of 1991, the U.S. National Institute of Standards and Technology (NIST) proposed a digital signature algorithm (DSA), which was subsequently adopted as a U.S. Federal Information Processing Standard (FIPS 186 [FIPS186]).

The algorithm is a variant of the ElGamal scheme. Its security is based on the Discrete Logarithm Problem. The DSA public and private key and its procedures for signature and verification are summarised below.

**Public Key**
$p$ prime
$q$ 160-bit prime factor of $p-1$
$g = h^{(p-1)/q} \bmod p$, where $h < p-1$ and $h^{(p-1)/q} > 1 \pmod p$
$y \equiv g^x \bmod p$
*Remark:* Parameters $p, q$ and $g$ can be shared among a group of users.

**Private Key**
$x < q$ (a 160-bit number)

**Signing**
$m$ the message to be signed
$k$ choose at random, less than $q$
$r = (g^k \bmod p) \bmod q$
$s = (k^{-1}(\text{SHA-1}(m) + xr)) \bmod q$
*Remark:*

- $(s, r)$ is the signature.
- The security of the signature depends not only on the mathematical properties, but also on using a good random source for $k$.
- SHA-1 is a 160-bit hash function specified in FIPS186.

**Verifying**
$w = s^{-1} \bmod q$
$u_1 = (\text{SHA-1}(m)w) \bmod q$
$u_2 = (rw) \bmod q$
$v = (g^{u_1} y^{u_2}) \bmod p) \bmod q$

*Remark:* If $v = r$, then the signature is verified.

While DSA was specifically designed, so that it can be exported from countries regulating export of encryption soft and hardware (like the U.S. at the time when it was specified), it has been noted [Schneier1996, p. 490], that the operations involved in DSA can be used to emulate RSA and ElGamal encryption.

## 5.3 Public key certification

The aim of public key certification is to guarantee the connection between a public key and a user and to make it traceable for external parties. In cases in which it is impossible to ensure that a public key really belongs to a particular person, many protocols are no longer secure, even if the individual cryptographic modules cannot be broken.

### 5.3.1 Impersonation attacks

Assume Charlie has two pairs of keys (PK1, SK1) and (PK2, SK2), where SK denotes the secret key and PK the public key. Further assume that he manages to palm off PK1 on Alice as Bob's public key and PK2 on Bob as Alice's public key (by falsifying a public key directory).

Then he can attack as follows:

- Alice wants to send a message to Bob. She encrypts it using PK1 because she thinks that this is Bob's public key. She then signs the message using her secret key and sends it.
- Charlie intercepts the message, removes the signature and decrypts the message using SK1. If he wants to, he can then change the message in any way he likes. He then encrypts the message again, but this time using Bob's genuine public key, which he has taken from a public key directory, signs the message using SK2 and forwards it to Bob.
- Bob verifies the signature using PK2 and will reach the conclusion that the signature is correct. He then decrypts the message using his secret key.

In this way Charlie can listen in on communication between Alice and Bob and change the exchanged messages without them noticing. The attack will also work if Charlie only has one pair of keys.

Another name for this type of attack is "man-in-the-middle attack". Users are promised protection against this type of attack by publickey certification, which is intended to guarantee the authenticity of public keys. The most common certification method is the X.509 standard.

### 5.3.2 X.509

All participants who want to have a X.509 certificate verifying that their public key belongs to a real person consult what is known as a certification authority (CA). They prove their identity to this CA (for example by showing their ID). The CA then issues them an electronic document (certificate) which essentially contains the names of the certificate-holder and the CA, the certificate-holder's public key and the validity period of the certificate. The CA then signs the certificate using its secret key.

Anyone can now use the CA's public key to verify whether a certificate is falsified. The CA therefore guarantees that a public key belongs to a particular user.

This procedure is only secure as long as it can be guaranteed that the CA's public key is correct. For this reason, each CA has its public key certified by another CA that is superior in the hierarchy. In the upper hierarchy level there is usually only one CA, which can of course then have its key certified by another CA. It must therefore transfer its key securely in another way. In the case of many software products that work with certificates (such as the Microsoft and Netscape Web browsers), the certificates of these root CAs are permanently embedded in the program right from the start and cannot be changed by users at a later stage. However, (public) CA keys, in particularly those of the root entity, can also be secured by means of making them available publicly.

## References

[Schneier1996] Bruce Schneier,
Applied Cryptography, Protocols, Algorithms, and Source Code in C,
Wiley, 2nd edition, 1996.

[PKCS1] RSA Laboratories
PKCS #1 v2.1 Draft 3: RSA Cryptography Standard, April 19, 2002.

[FIPS186] U.S. Department of Commerce/N.I.S.T.
Entity authentication using public key cryptography, February 18, 1997.

# 6 Elliptic Curves

(Filipovics B. / Büger M. / Esslinger B. / Oyono R., April 2000, Updates: Dec. 2001, June 2002, Mar. 2003)

## 6.1 Elliptic curve cryptography – a high-performance substitute for RSA?

In many business sectors secure and efficient data transfer is essential. In particular, the RSA algorithm is used in many applications. Although the security of RSA is beyond doubt, the evolution in computing power has caused a growth in the necessary key length. Today, 1024-bit RSA keys are standard, but the GISA (German Information Security Agency) recommends the usage of 2048-bit keys from 2006 on (compare section 3.11). The fact that most chips on smart cards cannot process keys extending 1024 bit shows that there is a need for alternatives. Elliptic curve cryptography (ECC) can be such an alternative in the field of asymmetric cryptography.

The efficiency of a cryptographic algorithm depends on the key length and the calculation effort that is necessary to provide a prescribed level of security. The major advantage of ECC compared to RSA is that it requires much shorter key lengths. If we assume that the computing power increases by Moore's law (i. e. it doubles every 18 months), then the evolution of the key lengths for secure communication will be as figure 1 (source: Arjen Lenstra und Eric Verheul: http://cryptosavvy.com/table.htm).

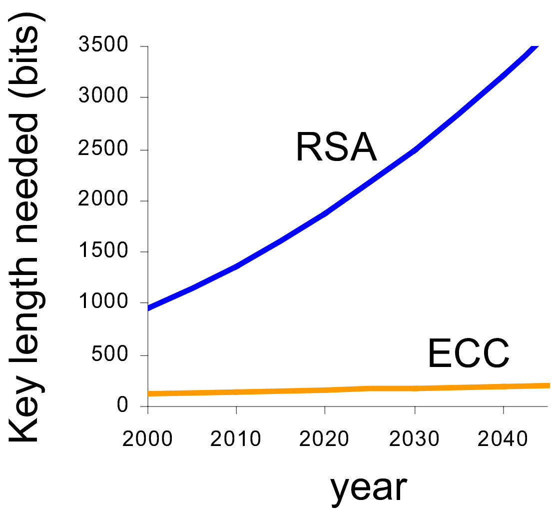



Figure 1: Prognosis of the key lengths to be regarded safe for RSA and Elliptic Curves

In addition, a digital signature can be processed 10-times faster with ECC than with RSA.

However, verification of a given signature is still more efficient with RSA than with ECC. Refer to figure 2 (source: Dr. J. Merkle, Elliptic Curve Cryptography Workshop, 2001) for a comparison. The reason is that RSA public keys can be chosen relatively small as long as the secret key is sufficiently long.

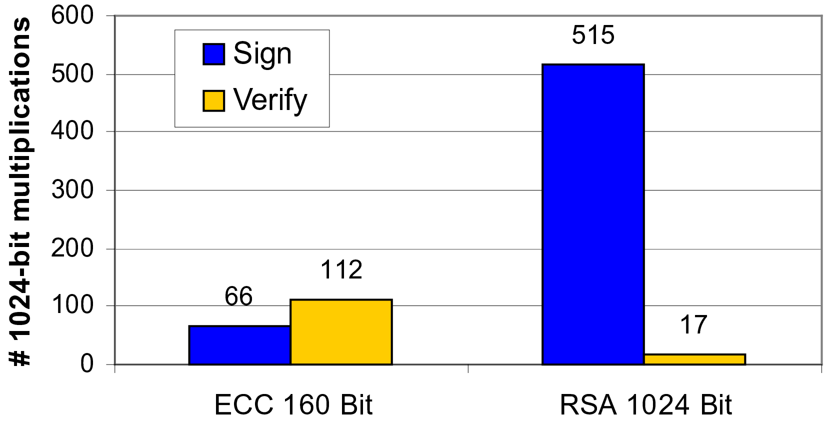



Figure 2: Comparison of signing and verification time for RSA and Elliptic Curves

Nevertheless, thin clients like smart cards usually have to store the (long) secret key and have to process a digital signature rather than to verify one. Therefore, there is a clear advantage for ECC in terms of efficiency.

Today, the major problem with ECC-implementations is the lack of standardization. There is only one way to implement RSA, but there are many ways for ECC: One can work with different sets of numbers, different (elliptic) curves described by up to 6 parameters, and a variety of representations of the elements on the curve. Each choice has its advantages and disadvantages, and one can certainly construct the most efficient for each application. However, this causes problems in interoperability. But if all ECC-tools should be able to communicate with each other, they will have to support all different algorithms, which might put the advantage of efficient computation and the need of less storage capacity to the contrary.

Therefore, international standardization organizations like IEEE (P1363), ASC (ANSI X9.62, X9.63), ISO/IEC as well as major players like RSA labs or Certicom have recently started standardization initiatives. While the IEEE only describes the different implementations, the ASC has explicitly stated 10 elliptic curves and recommends their usage. The advantage of the ASC approach is that one needs only a single byte to indicate which curve is meant. However, it is not clear yet whether the ASC-curves will become a de facto standard.

Although we see no need to replace RSA in any application today, one should take the usage of ECC-based tools into consideration whenever a new system is set up — in particular, when the tool should be available beyond 2005.

## 6.2 Elliptic curves – history

Mathematicians have been researching elliptic curves for over 100 years. In the course of time, many lengthy and mathematically complex results have been found and published in connection with elliptic curves. A mathematician would say that elliptic curves (or the mathematics behind them) are widely understood. This research was originally purely mathematical. That is to say, elliptic curves were investigated, for example, in the mathematical areas of number theory and algebraic geometry, which are generally highly abstract. Even in the recent past, elliptic curves played an important role in pure mathematics. In 1993 and 1994, Andrew Wiles published mathematical works that triggered enthusiasm far beyond the specialist audience. In these works, he proved a conjecture put forward in the 1960's. To put it short, this conjecture was concerned with the connection between elliptic curves and what are called module forms. That which is interesting for most people is that the works of Wiles also proved the famous second theorem of Fermat. Mathematicians had spent centuries (Fermat lived from 1601 to 1665) trying to find a strict proof of this theorem. Understandably, therefore, Wiles' proof got a good response. Fermat formulated his theorem as follows (written in the border of a book):

> *Cubum autem in duos cubos, aut quadratoquadratum in duos quadratoquadratos, et generaliter nullam in infinitum ultra quadratum potestatem in duos ejusdem nominis fas est dividere: cujus rei demonstrationem mirabilem sane detexi. Hanc marginis exiguitas non caperet.*

Translated freely, using the denotation of modern mathematics, this means:
No positive whole numbers $x, y$ and $z$ greater than zero exist such that $x^n + y^n = z^n$ for $n > 2$. I have found an amazing proof of this fact, but there is too little space in the border [of the book] to write it down.

This is truly amazing: A statement that is relatively simple to understand (we are referring to Fermat's second theorem here) could only be proved after such a long period of time, although Fermat himself claimed to have found a proof. What's more, the proof found by Wiles is extremely extensive (all of Wiles publications connected with the proof made up a book in themselves). This should therefore make it obvious that elliptic curves are generally based on highly complex mathematics.

Enough about the role of elliptic curves in pure mathematics. In 1985 Neal Koblitz and Victor Miller independently suggested using elliptic curves in cryptography. Elliptic curves have thus also found a concrete practical application. Another interesting field of application for elliptic curves is for factorising whole numbers. (For example the RSA cryptography system is based on the difficulty/complexity of finding prime factors of an extremely large number.) In this area, procedures based on elliptic curves have been investigated and partially used since 1987 (a study by H.W. Lenstra). There are also prime number tests based on elliptic curves.

Elliptic curves are used differently in the various areas. Encryption procedures based on elliptic curves are based on the difficulty of a problem known as elliptic curve discrete logarithm. The factorisation of whole numbers uses the fact that a large number of elliptic curves can be generated for a natural composite number $n$ with several prime factors; however, these curves are not then groups for composite $n$. More information about this can be found under Factorisation using elliptic curves.

## 6.3 Elliptic curves – mathematical basics

This section provides information about *groups* and *fields*.

### 6.3.1 Groups

Because the term *group* is used differently in everyday language than in mathematics, we will, for reasons of completeness, begin by introducing the essential statement of the formal definition of a group:

- A group is a non-empty set $G$ and an operation $+$. The set $G$ is closed under the operation $+$. Regardless of which two elements $a, b$ from $G$ are taken, performing the operation on them gives an element in $G$ (i.e. $a + b = c$, and $c$ lies in $G$).
- For all elements $a, b$ and $c$ in $G$: $(a + b) + c = a + (b + c)$.
- There exists an element $e$ in $G$ that behaves neutrally with respect to the operation $+$. That means that for all a in the set $G:\ a + e = e + a = a$.
- For each element $a$ in $G$ there exists a so-called inverse element $-a$ ($-a$ also lies in $G$) such that: $a + (-a) = (-a) + a = e$.

If also $a+b = b+a$ for all $a, b$ in $G$ then we call the group an *Abelian* group. An operation denoted as $+$ indicates an *additive* group; if the operation is denoted as $\cdot$, we speak of a *multiplicative* group.

The simplest example of an (Abelian) group is the group of whole numbers under the standard operation of addition. The set of whole numbers is denoted as $\mathbb{Z}$. $\mathbb{Z}$ has an infinite number of elements, because $\mathbb{Z} = \{\cdots, -4, -3, -2, -1, 0, 1, 2, 3, 4, \cdots\}$. For example, the operation of $1+2$ lies in $\mathbb{Z}$, for $1 + 2 = 3$ and 3 lies in $\mathbb{Z}$. The neutral element in the group $\mathbb{Z}$ is 0. The inverse element of 3 is $-3$, for $3 + (-3) = 0$.

There are also *finite* groups. This means that these exists a set $\mathcal{M}$ with a fixed number of elements and an operation $+$ such that the above conditions are fulfilled. One example of this is any set $\mathbb{Z}_n$ where $\mathbb{Z}_n = \{0, 1, 2, 3, \cdots, n-1\}, n$ is a positive whole number and the operation is addition mod $n$, i.e. $a$ and $b$ in $\mathbb{Z}_n$ are subject to the operation $a + b \bmod n$.

**Cyclic groups** Cyclic groups are those groups $G'$ that possess an element $g$ from which the group operation can be used to generate all other elements in the group. This means that for each element $a$ in $G'$ there exists a positive whole number $i$ such that if $g$ is subject to the operation $i$ times (i.e. "$g \cdot i$"), $g + g + \cdots + g = a$ (additive group) or $g^i = g \cdot g \cdots g = a$ (multiplicative group). The element $g$ is the *generator* of the cyclic group — each element in $G$ can be generated using $g$ and the operation.

Now to the order of an element of the group: Let $a$ be in $G$. The smallest positive whole number $r$ for which $a$ subject to the operation with itself $r$ times is the neutral element of the group $G$ (i.e.: $r \cdot a = a + a + \cdots + a = e$ bzw. $a^r = e$), is called the *order* of $a$.

The order of the group is the number of elements in the set $G$.

#### 6.3.2 Fields

In mathematics, a field is understood to be a set $K$ with two operations (denoted as $+$ and $\cdot$) which fulfils the following conditions:

- The set $K$ forms an Abelian group together with the operation $+$ (addition), where 0 is the neutral element of the operation $s$.
- The set $K$ (without the element 0) also forms an Abelian group together with the operation $\cdot$ (multiplication).
- For all elements $a, b$ and $n$ in $K$, $n \cdot (a + b) = n \cdot a + n \cdot b$ and $(a + b) \cdot n = a \cdot n + b \cdot n$.

There are *infinite* fields, i.e. the set on which the field is based contains an infinite number of elements (e.g.: the field of real numbers). And there are also finite fields, such as $\mathbb{Z}_p = \{0, 1, 2, 3, \cdots, p-1\}$ , where $p$ is a prime. $\mathbb{Z}_p$ with addition mod $p$ and multiplication mod $p$ is a finite field.

**Characteristic of a field** Let $K$ be a field and 1 be the neutral element of $K$ with respect to the multiplicative operation "$\cdot$". For positive natural numbers $n$, let us understand $n_1$ to be $n_1 = 1 + 1 + \cdots + 1$ ($n$ summands and $n_1$ is an element in $K$). If $n_1$ is then unequal to 0 for all $n > 0$, then we call $K$ a field with characteristic zero. Otherwise, the characteristic of $K$ is defined to be the smallest positive natural number $p$ for which $p_1 = 0$ (note: $p$ is then a prime). Comment: The field of real numbers has the characteristic 0; the field $\mathbb{Z}_p$ has the characteristic $p$.

### 6.4 Elliptic curves in cryptography

An elliptic curve is described by an equation. In order to keep it simple, we restrict our explanation to elliptic curves over

$$\mathbb{Z}_p = \{0, 1, 2, 3, \cdots, p-1\}$$

where $p$ is a prime greater than 3. $\mathbb{Z}_p$ with addition mod $p$ and multiplication mod $p$ is a finite field. However, we must mention that elliptic curves can be defined over any (finite) field. In particular, elliptic curves over fields with characteristic 2 are extremely interesting from a practical point of view because computers can be used to represent the elements from these fields as bit strings. This leads to an efficient implementation of the arithmetic in such fields, which means that a computer can perform the operations of the field particularly quickly.

Because these points actually refer to the same thing, it is seldom necessary to distinguish between exact meanings.

An elliptic curve over $\mathbb{Z}_p$ is defined by an equation of the following form:

$$y^2 \text{ (mod } p) = x^3 + ax + b(\text{mod } p)$$

(thus: equality in the field $\mathbb{Z}_p$), where $a, b$ are in $\mathbb{Z}_p$ and $4a^3 + 27b^2$ mod $p$ is not equal to zero. For fixed chosen numbers $a$ and $b$ in $\mathbb{Z}_p$, this equation has the pair of solutions

$$\mathbf{E} = \left\{ (x, y) \;\middle|\; \begin{array}{c} x \text{ and } y \text{ are in } \mathbb{Z}_p \text{ and} \\ y^2 \equiv x^3 + ax + b \text{ (mod } p) \text{ and} \\ 4a^3 + 27b^2 \not\equiv 0 \text{ (mod } p) \end{array} \right\},$$

i.e. the set $\mathbf{E}$ consists of all pairs $x$ and $y$ that are a solution (in $\mathbb{Z}_p$) of the above equation. It must be noted that the numbers $a, b$ and $p$ determine which pairs $(x, y)$ lie in the set $\mathbf{E}$. This means that $a, b$ and $p$ specify this set. The elements $(x, y)$ in $\mathbf{E}$ are called points on the elliptic curve. In addition, $\mathbf{E}$ has one more element $O$ (the so-called point in infinity). The set $\mathbf{E}$ is usually called an elliptic curve.

We can now define an operation[110] (also denoted as $+$, although it is not the standard/usual addition of real numbers) on two elements in $\mathbf{E}$ such that the operation delivers an element that also lies in $\mathbf{E}$. The set $\mathbf{E}$ is therefore closed under the operation $+$. We can show that $\mathbf{E}$ is a group. The neutral element of the group $\mathbf{E}$ is the point in infinity $O$. Thus, for every two points $(x_1, y_1)$ and $(x_2, y_2)$ on the elliptic curve $\mathbf{E}$, there exists a point $(x_3, y_3)$ on $\mathbf{E}$ such that the operation $+$ complies with the following: $(x_1, y_1) + (x_2, y_2) = (x_3, y_3)$. Under certain circumstances, these points may also be equal to the point in infinity. Thus, when we speak of a point $P$ on an elliptic curve $\mathbf{E}$, we mean that $P = (x, y)$ and $(x, y)$ lies in the set $\mathbf{E}$. Any two points on an elliptic curve specified by $a, b$ and $p$ can therefore be added and the result is a point that also lies on the same elliptic curve.

[110] A animated addition of elliptic curve points can be found at the web page of Certicom: http://www.certicom.com/resources/ecc_tutorial/ecc_tutorial.html.

### Adding of two points on an elliptic curve

The following two figures show an elliptic curve in the affine plane and shows how points on an elliptic curve are added. Note that the point in infinity $O$ cannot be represented in the affine plane.

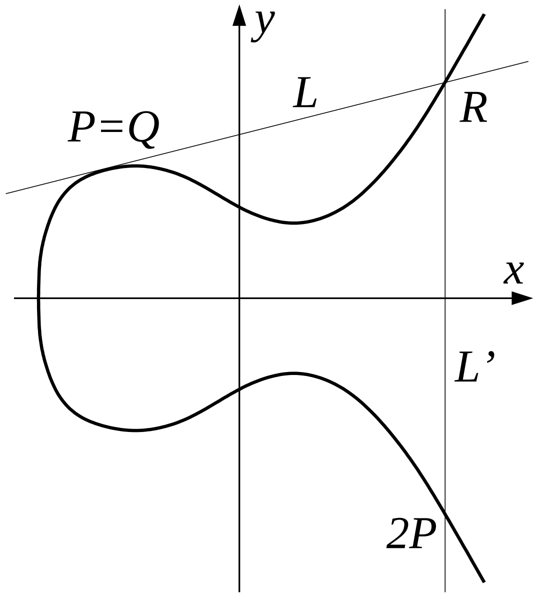



Figure 3: Doubling the point P

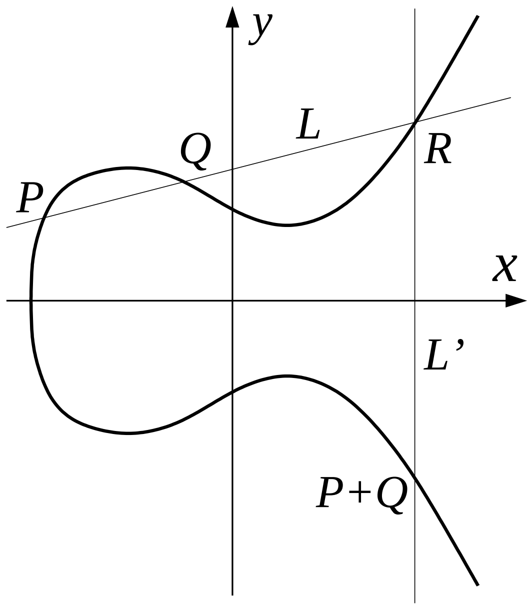



Figure 4: Adding distinct points P and Q

We must note that $\mathbf{E}$ can have the following meanings:

- the set $\mathbf{E}$ of solutions pairs $(x, y)$ for an equation including the point $O$
- the group $\mathbf{E}$ (with the operation "addition of $(x_1, y_1)$ and $(x_2, y_2)$")
- the elliptic curve $\mathbf{E}$ (which is actually the same as the group $\mathbf{E}$)

For cryptography, the important fact is that, for very large numbers, it appears to be extremely difficult to use a given point $Q$ on an elliptic curve to determine which two points have to be added to obtain $Q$.

For large numbers $a, b$ and $p$ ($p$, for example, has a length of more than 160 bits), the computer can easily add the point $P$ $m$ times after another, i.e. to determine the point $P + P + \cdots + P = Q$ in an incredibly short space of time (in a few fractions of a second) ($m$ summands $P$). Rather than $P + P + \cdots + P = Q$ ($m$ summands $P$) we also write $mP = Q$. If we have a point $P$ and a point $Q$, which both lie on the same elliptic curve, no procedure is known that enables us — within an acceptable space of time — to determine the number $m$ (assuming it actually exists) for which $mP = Q$. This is referred to as the "elliptic curve discrete logarithm problem" (abbreviated to ECDLP).

We must note that not all elliptic curves are equally secure. This means that we must choose the parameters $a$ and $b$ carefully when defining a curve. For certain classes of elliptic curves, it is possible to solve the ECDLP easier than in the general case. Cryptographically unsuitable elliptic curves are called *abnormal* curves (these are curves over $\mathbb{Z}_p$ for which the set $\mathbf{E}$ has precisely $p$ elements) and the *super-singular* curves (the curves for which we can reduce the calculation of the ECDLP to calculating the "standard" discrete logarithm in other finite fields, i.e. simplify the calculation). There are therefore cryptographically good and bad curves. However, for given parameters $a$ and $b$ we can, with some difficulty, establish whether or not the resulting elliptic curve is useful for cryptographic purposes. The curves used in cryptography are usually provided by experts. They ensure that the elliptic curves they classify as secure satisfy the current security requirements.

For secure curves, the parameter $p$ determines how long it takes to solve the ECDLP on this curve. The larger the parameter $p$, the longer it takes to solve the problem. Experts recommend a bit length of over 200 bits for the parameter $p$. This makes it clear why elliptic curves are so interesting for cryptography. Because the parameter $p$ also determines the time required to perform the signature/encryption procedure when using elliptic curves in cryptography. The time taken to generate a pair of keys also depends on $p$. Thus, small values (few bits) are desirable here (in order to minimise the run times for the procedures); however, the required security must still be maintained. For example, with a length of 200 bits for $p$, a *good* elliptic curve is just as secure as an RSA module of over 1024 bits in length (at least according to the current state of research). The reason for this is that the quickest algorithms for solving the *elliptic curve discrete logarithm* problem have an exponential run time — unlike the sub-exponential run times that the best factorisation algorithms currently have (number sieve, quadratic sieve or factorisation with elliptic curves). Therefore, the parameters for cryptographic procedures based on the problem of *factorising whole numbers* must be greater than the parameters for cryptographic procedures based on the ECDLP problem.

### 6.4.1 Digital signatures using elliptic curves

The *elliptic curve discrete logarithm problem* (ECDLP) forms the basis for elliptic-curve cryptography. Various signature procedures are based on this. What they have in common is how they use the public parameters and how these parameters are used to generate secret and public keys:

- The parameters of the elliptic curve E, i.e. a prime number $p$ that determines over which field $\mathbb{Z}_p$ the elliptic curve $\mathbf{E}$ is defined, as well as the two numbers $a$ and $b$ in $\mathbb{Z}_p$.
- A point $G = (x, y)$ that lies on the elliptic curve $\mathbf{E}$.
- A prime number $r < p$ for which $rG = O$ (i.e. $G$ added $r$ times gives the neutral element of the group $\mathbf{E}$) and $r$ is a factor of $\#\mathbf{E}$ (where $\#\mathbf{E}$ is the number of elements in the set $\mathbf{E}$). The point $G$ therefore has the order $r$ and is the generator of a cyclic subgroup of $\mathbf{E}$ with the order $r$.
- The number $k = \#\mathbf{E}/r$ ($k$ is called the *cofactor*).

The parameters $a, b, p, G, r$ and $k$ listed above are called *domain parameters*. They determine on which elliptic curve $\mathbf{E}$ and in which cyclic subgroup of $\mathbf{E}$ a signature procedure has been "used".

The secret key $s$ of the signature generator is a (random) whole number $s$ in the interval $[1, r-1]$. The public key of the signature generator is a point $W = (x, y)$ on the elliptic curve $\mathbf{E}$. The public key $W$ and secret key $s$ are interrelated as follows: $W = sG$. This means that the domain parameters (particularly $G$) and the secret key $s$ are used to calculate the public key $W$ (by adding $G$ $s$ times on $\mathbf{E}$). The ECDLP is obviously used here: If $W$ and $G$ (as well as the other domain parameters used) are known, then it is difficult to use these to calculate $s$. (If the parameters are chosen correctly, this currently appears to be practically impossible).

In order to verify a signature, the recipient of the signature must know the following:

1. The signature procedure used,
2. The hash function used,
3. The domain parameters used to generate the signature
4. The public key $W$ of the signature generator.

### 6.4.2 Factorisation using elliptic curves

There are factorisation algorithms based on elliptic curves [111]. More precisely, these procedures exploit the fact that elliptic curves can be defined over $\mathbb{Z}_n$ ($n$ composite number). Elliptic curves

[111] In 1987 H.W. Lenstra published a factorisation algorithm, based on elliptic curves (see [Lenstra1987]). The biggest compound number currently factorised with elliptic curves is the number $628^{59} - 1$, which has 55 decimal digits. It was found Oct. 6th, 2001 by M. Izumi (See ECMNET).

over $\mathbb{Z}_n$ do not form a group, because not every point on such an elliptic curve has an inverse point. This is connected with the fact that - if $n$ is a composite number - there exist elements in $\mathbb{Z}_n$ that do not have an inverse with respect to multiplication mod $n$. In order to add two points on an elliptic curve over $\mathbb{Z}_n$, we can calculate in the same way as on elliptic curves over $\mathbb{Z}_p$. Addition of two points (on an elliptic curve over $\mathbb{Z}_n$), however, fails if and only if a factor of $n$ has been found. The reason for this is that the procedure for adding points on elliptic curves gives elements in $\mathbb{Z}_n$ and calculates the inverse elements for these (with respect to multiplication mod $n$) in $\mathbb{Z}_n$. The extended Euclidean algorithm is used here. If the addition of two points (that lie of an elliptic curve over $\mathbb{Z}_n$) gives an element in $\mathbb{Z}_n$ that does not have an inverse element in $\mathbb{Z}_n$, then the extended Euclidean algorithm delivers a genuine factor of $n$.

Factorisation using elliptic curves thus principally works as follows: You select random curves over $\mathbb{Z}_n$, as well as random points (that lie on this curve) and add them; you thus obtain points that also lie on the curve or find a factor of $n$. Factorisation algorithms based on elliptic curves therefore work probabilistically. The opportunity of defining large number of elliptic curves over $\mathbb{Z}_n$ allows you to increase the probability of finding two points which you can add to obtain a factor of $n$. These procedures are therefore highly suitable for parallelisation.

## 6.5 Implementing elliptic curves

CrypTool also offers elliptic curves for the digital signature function.

It implements the basic algorithms for group operations, for generating elliptic curves, for importing and exporting parameters for elliptic curves over finite fields with $p$ ($p$ prime) elements. The algorithms have been implemented in ANSI C and comply with draft no. 8 of the IEEE P1363 work group *Standard Specifications for Public Key Cryptography*

`http://grouper.ieee.org/groups/1363`.

The procedure implements the cryptographic primitives for generating and verifying signatures for the variations of Nyberg-Rueppel signatures and DAS signatures based on elliptic curves (in accordance with draft no. 8 of the IEEE P1363 work group). This was done in collaboration with the Secude GmbH — using the above library and the Secude SDK.

## References

[Lenstra1987] H.W. Lenstra
Factoring integers with elliptic curves, Annals of Mathematics 126, pp. 649-673, 1987.

## Web links

1. Certicom Online Tutorial,
   http://www.certicom.com/resources/ecc_tutorial/ecc_tutorial.html
2. IEEE P1363 http://grouper.ieee.org/groups/1363
3. An informative web page about factorisation with elliptic curves.
   http://www.loria.fr/~zimmerma/records/ecmnet.html
   It contains literature related to the topic factorisation with elliptic curves as well as links to other web page.
4. Key length comparison by Arjen Lenstra and Eric Verheul
   http://cryptosavvy.com/table.htm

# Index