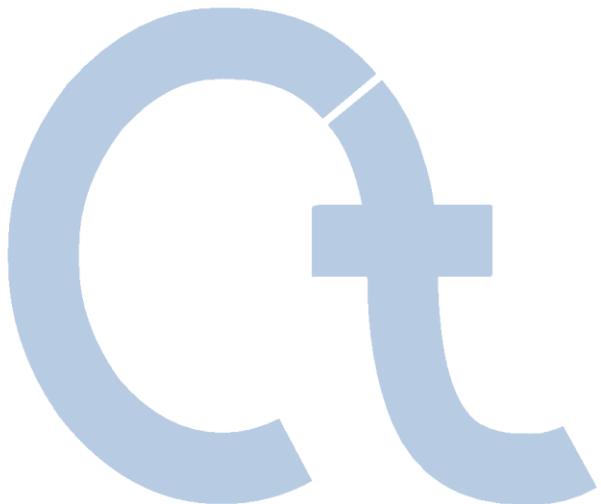


CRYPTOLOGY WITH CRYPTOTool 1



Practical Introduction to
Cryptography and Cryptanalysis

Scope, Technology, and Future of CrypTool 1.4.xx

Prof. Bernhard Esslinger and the CrypTool Team
(Updated: September 2017 with release CT 1.4.40)

www.cryptool.org

Content (I)

I. CrypTool and Cryptology – Overview

1. Definition and relevance of cryptology
2. The CrypTool project
3. Examples of classical encryption methods
4. Insights from cryptography development

II. Features of CrypTool 1

1. Overview
2. Interaction examples
3. Challenges for developers

III. Examples

1. Encryption with RSA / Prime number test / Hybrid encryption and digital certificates / SSL
2. Digital signature visualized
3. Attack on RSA encryption (small modulus N)
4. Analysis of encryption in PSION 58
5. Weak DES keys
6. Locating key material (“NSA Key”)
7. Attack on digital signature through hash collision search
8. Authentication in a client-server environment
9. Demonstration of a side channel attack (on hybrid encryption protocol)



(...)

Content (II)

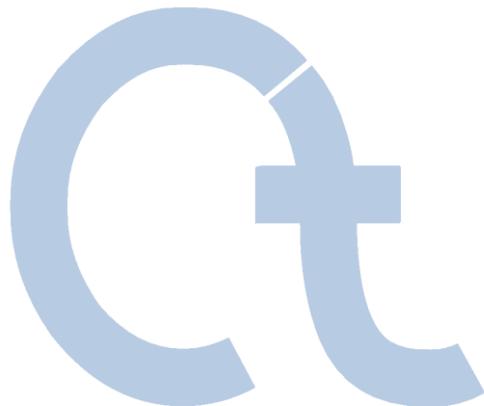
III. Examples

10. [RSA attack using lattice reduction](#)
11. [Random analysis with 3-D visualization](#)
12. [Secret Sharing using the Chinese Remainder Theorem \(CRT\) and Shamir](#)
13. [Implementation of CRT in astronomy \(solving systems of linear modular equations\)](#)
14. [Visualization of symmetric encryption methods using ANIMAL](#)
15. [Visualizations of AES](#)
16. [Visualization of Enigma encryption](#)
17. [Visualization of Secure Email with S/MIME](#)
18. [Generation of a message authentication code \(HMAC\)](#)
19. [Hash demonstration](#)
20. [Educational tool for number theory and asymmetric encryption](#)
21. [Point addition on elliptic curves](#)
22. [Password quality meter \(PQM\) and password entropy](#)
23. [Brute force analysis](#)
24. [Scytale / Rail Fence](#)
25. [Hill encryption / Hill analysis](#)
26. [CrypTool online help / Menu tree of the program](#)

IV. Project / Outlook / Contact



Content



- I. CrypTool and Cryptology – Overview
 - II. Features of CrypTool 1
 - III. Examples
 - IV. Project / Outlook / Contact
- Appendix

Relevance of Cryptography

Examples of Applied Cryptography

- Phone cards, cell phones, remote controls
- Cash machines, money transfer between banks
- Electronic cash, online banking, secure email
- Satellite TV, pay-per-view TV
- Immobilizer systems in cars
- Digital Rights Management (DRM), Cloud
- Cryptography is no longer limited to agents, diplomats, and the military.
Cryptography is a modern, mathematically characterized science.
- The breakthrough of cryptography followed the broadening usage
of the Internet
- For companies and governments it is important that systems are secure
and that



*users (i.e., clients and employees)
are aware of and understand IT security!*

Definition Cryptology and Cryptography

Cryptology (*from the Greek kryptós, "hidden," and lógos, "word") is the science of secure (or, generally speaking, secret) communication. This security requires that legitimate users, a transmitter and a receiver, are able to transform information into a cipher by virtue of a key – that is, a piece of information known only to them. Although the cipher is inscrutable and often unforgeable to anyone without this secret key, the authorized receiver can either decrypt the cipher to recover the hidden information or verify that it was sent in all likelihood by someone possessing the key.*

Cryptography was concerned initially with providing secrecy for written messages. Its principles apply equally well, however, to securing data flow between computers or to encrypting television signals. Today, the modern (mathematical) science of cryptology is not just a set of encryption mechanisms. It has since been applied to a broad range of aspects of modern life, including data and message integrity, electronic signatures, random numbers, secure key exchange, secure containers, electronic voting, and electronic money.

Source: Britannica (www.britannica.com)

A similar definition can be found on Wikipedia: <http://en.wikipedia.org/wiki/Cryptography>



Cryptography – Objectives

■ Confidentiality

Information can be made effectively unavailable or unreadable for unauthorized individuals, entities, and processes.

■ Authentication

The receiver of a message can verify the identity of the sender.

■ Integrity

Integrity ensures that data has not been altered or destroyed in an unauthorized manner.

■ Non-Repudiation

The receiver can prove that the message he or she received is precisely what the sender sent; the sender will have no means to deny any part of his or her participation.

The CrypTool Project

- Originated as an awareness program for a large bank (internal training)
→ **Employee education**
 - Developed in cooperation with universities (improvement of education)
→ **Media didactic approach and standard oriented**
 - Some milestones
 - 1998 **Project start** – over 50 person-years of effort have since been invested in CT1
 - 2000 CrypTool available as **freeware**
 - 2002 CrypTool available on the **Citizen's CD of the BSI** (German Information Security Agency)
 - 2003 CrypTool becomes **open source** – hosting by University of Darmstadt
 - 2007 CrypTool available in German, English, Polish, and Spanish
 - 2008 .NET and Java versions started – hosted by University of Duisburg and SourceForge
 - 2010 CT1 available in Serbian and Greek
 - 2010 CrypTool-Online (CTO) and MysteryTwister C3 (MTC3) published
 - 2011 .NET version (CT2) and Java version (JCT) published as 1st betas
 - 2012 New joined web portal for all 5 CT sub projects, called CrypTool portal (CTP)
 - 2014 CT 2.0 released (August 2014) – hosted by University of Kassel
 - 2017 CT1 also available in French (release 1.4.40); relaunch of the CrypTool portal and CTO
- Also see <https://en.wikipedia.org/wiki/CrypTool>

The CrypTool Project

■ Some Awards

2004 TeleTrusT (TTT Förderpreis / Sponsorship Award)



2004 NRW (IT Security Award NRW)



2004 RSA Europe (Finalist of European Information Security Award 2004)



2008 "Selected Landmark" in initiative "Germany – Land of Ideas"



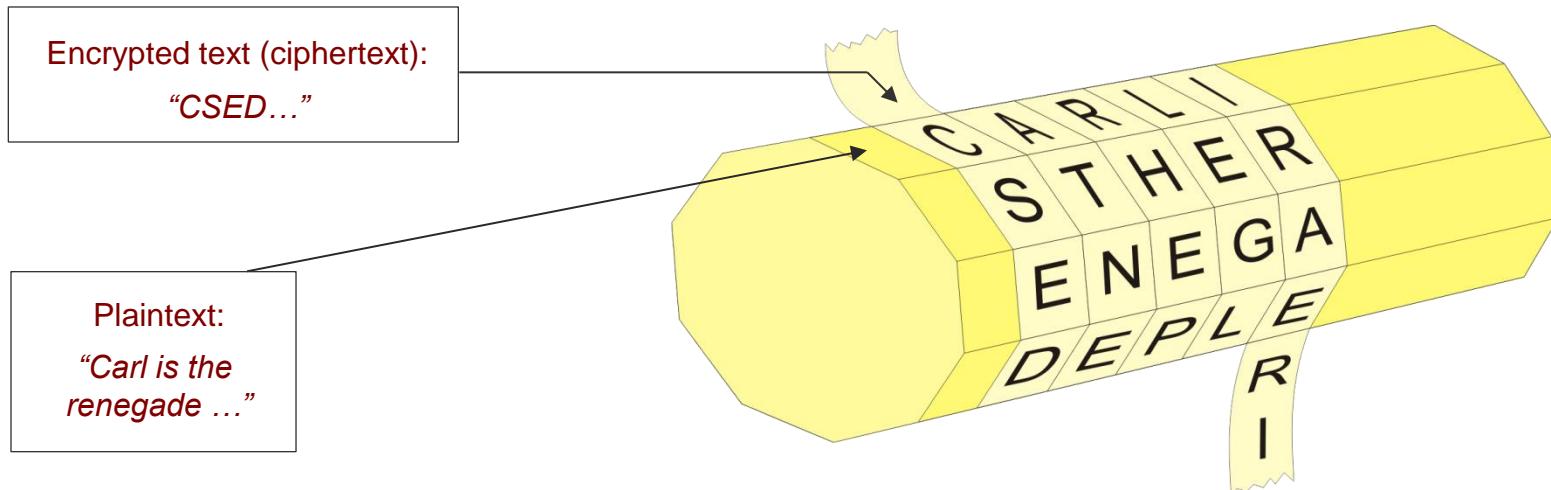
■ Developers

- Developed by people from companies and universities in many different countries.
- Currently there are about 100 people working on CrypTool worldwide.
Additional project members or applicable resources are always appreciated.

Examples of Early Cryptography (1)

Ancient encryption methods

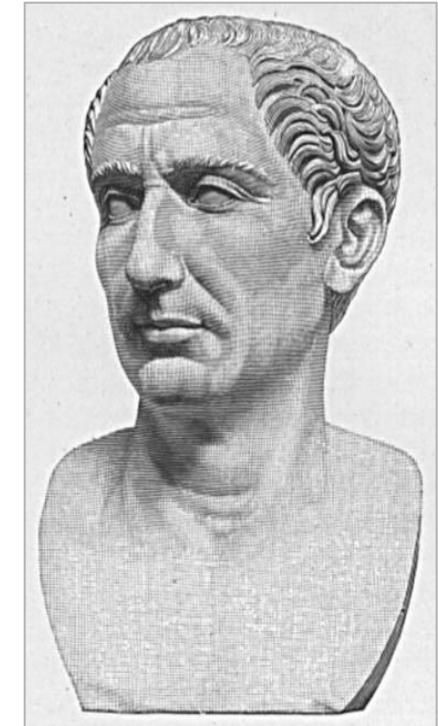
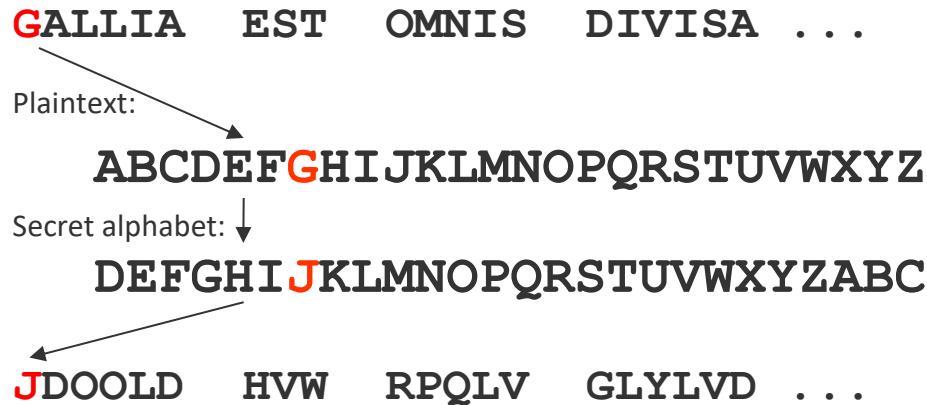
- **Tattoo on the shaven head of a slave, concealed by regrown hair**
- **Atbash (circa 600 B.C.)**
 - Hebrew secret language, reversed alphabet
- **Scytale from Sparta (circa 500 B.C.)**
 - Described by Greek historian/author Plutarch (45 - 125 B.C.)
 - The sender and receiver each need a cylinder (such as a wooden rod) with the same diameter
 - Transposition (plaintext characters are re-sorted)



Examples of Early Cryptography (2)

Caesar encryption (mono-alphabetic substitution cipher)

- **Caesar encryption** (Julius Caesar, 100 - 44 B.C.)
- Simple substitution cipher



- **Attack:** Frequency analysis (typical character allocation)

Presentation with CrypTool via the following menus:

- Animation: “Indiv. Procedures” \ “Visualization of algorithms” \ “Caesar”
- Implementation: “Crypt/Decrypt” \ “Symmetric (classic)” \ “Caesar / Rot-13”

Examples of Early Cryptography (3)

Vigenère encryption (poly-alphabetic substitution cipher)

- **Vigenère encryption** (Blaise de Vigenère, 1523-1596)

- Encryption with a keyword using a key table

- Example

Keyword: **CHIFFRE**

Encrypting: **VIGENERE** becomes **XPOJSVVG**

- The plaintext character (V) is replaced by the character in the corresponding row and in the column of the first keyword character (c). The next plaintext character (I) is replaced by the character in the corresponding row and in the column of the next keyword character (h), and so on.
- If all characters of the keyword have been used, then the next keyword character is the first key character.
- **Attack** (via Kasiski test; other tests also exist): Plaintext combinations with an identical cipher text combination can occur. The distance of these patterns can be used to determine the length of the keyword. An additional frequency analysis can then be used to determine the key.

The diagram illustrates the Vigenère square, a polyalphabetic cipher table. It consists of 26 horizontal rows and 26 vertical columns, each labeled with a letter from A to Z. The top row contains the letters a through z. The leftmost column contains the letters A through Z. The intersection of the first row and first column is at 'a'. The intersection of the second row and second column is at 'b'. The intersection of the third row and third column is at 'c'. The intersection of the fourth row and fourth column is at 'd'. The intersection of the fifth row and fifth column is at 'e'. The intersection of the sixth row and sixth column is at 'f'. The intersection of the seventh row and seventh column is at 'g'. The intersection of the eighth row and eighth column is at 'h'. The intersection of the ninth row and ninth column is at 'i'. The intersection of the tenth row and tenth column is at 'j'. The intersection of the eleventh row and eleventh column is at 'k'. The intersection of the twelfth row and twelfth column is at 'l'. The intersection of the thirteenth row and thirteenth column is at 'm'. The intersection of the fourteenth row and fourteenth column is at 'n'. The intersection of the fifteenth row and fifteenth column is at 'o'. The intersection of the sixteenth row and sixteenth column is at 'p'. The intersection of the seventeenth row and seventeenth column is at 'q'. The intersection of the eighteenth row and eighteenth column is at 'r'. The intersection of the nineteenth row and nineteenth column is at 's'. The intersection of the twentieth row and twentieth column is at 't'. The intersection of the twenty-first row and twenty-first column is at 'u'. The intersection of the twenty-second row and twenty-second column is at 'v'. The intersection of the twenty-third row and twenty-third column is at 'w'. The intersection of the twenty-fourth row and twenty-fourth column is at 'x'. The intersection of the twenty-fifth row and twenty-fifth column is at 'y'. The intersection of the twenty-sixth row and twenty-sixth column is at 'z'.

Keyword character

Plaintext character

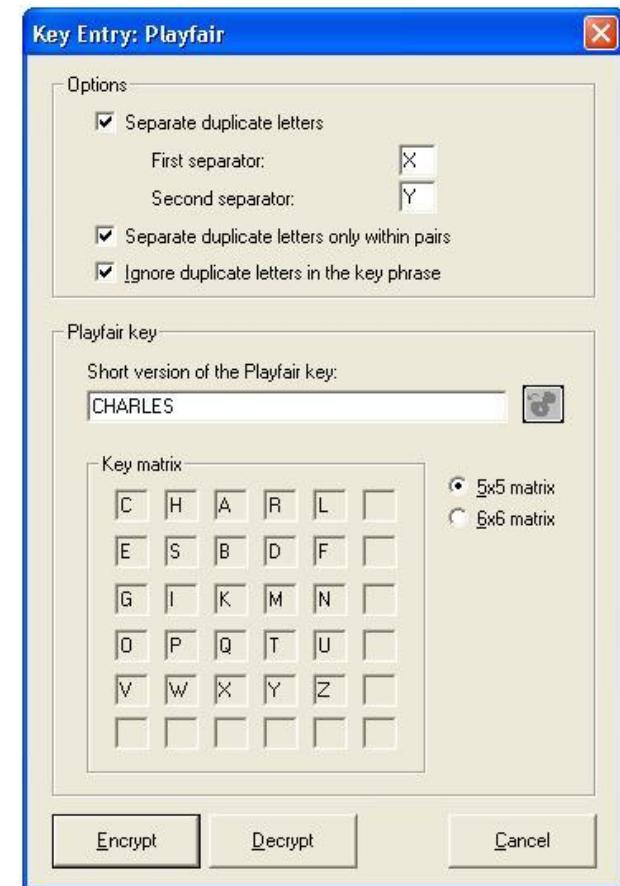
Encrypted character

Tableau carré, dit « Carré de Vigenère »

Examples of Early Cryptography (4)

Other classic encryption methods

- **Homophone substitution**
- **Playfair** (invented 1854 by Sir Charles Wheatstone, 1802-1875)
 - Published by Baron Lyon Playfair
 - Substitution of one character pair by another one based on a square-based alphabet array
- **Transfer of book pages**
 - Adaptation of the One-Time Pad (OTP)
- **Turning grille** (Fleissner)
- **Permutation encryption**
 - “Double Dice” (double column transposition)
(Pure transposition, but very effective)



Cryptography in Modern Times

Developments in cryptography from 1870-1970

Classic methods

- are still in use today
(since not everything can be done by a computer...)
- and their principles of transposition and substitution became the foundation of the design of modern symmetric algorithms, which combine simpler operations at a bit level (a type of multiple encryption or cipher cascade), use block ciphers, and/or use repeated uses of an algorithm over multiple rounds.

Encryption becomes

- more **sophisticated**,
- **mechanized or computerized**, and
- remains **symmetric**.

Example from the First Half of the 20th Century

Mechanical encryption machines (rotor machines)

Enigma Encryption (Arthur Scherbius, 1878-1929)

- More than 200,000 machines were used in WWII.
- The rotating cylinders encrypt every character of the text with a new permutation.
- The Polish Cipher Bureau broke the pre-war Enigma prototype as early as 1932.
- Based on this work, the later Enigma was broken only with massive effort. About 7000 cryptographers in the UK used decryption machines, captured Enigma prototypes, and intercepted daily status reports (such as weather reports).
- **Consequences of the successful cryptanalysis**

“The successful cryptanalysis of the Enigma cipher was a strategic advantage that played a significant role in winning the war. Some historians assert that breaking the Enigma code shortened the war by several months or even a year.”

(translated from http://de.wikipedia.org/wiki/Enigma_Machine - March 6, 2006)



Cryptography – Important Insights (1)

■ Kerckhoffs' principle (first stated in 1883)

- Separation of algorithm (method) and key e.g. Caesar encryption:
 - Algorithm: “Shift alphabet by a certain number of positions to the left”
 - Key: The “certain number of positions”
- Kerckhoffs' principle:
 - The secret lies within the key and not within the algorithm;
 - “security through obscurity” is invalid

■ One-Time Pad – Shannon / Vernam

- Theoretically completely unbreakable, but highly impractical (used by the red telephone*)

■ Shannon's concepts: Confusion and Diffusion

- Relation between M, C, and K should be as complex as possible (M=message, C=cipher, K=key)
- Every ciphertext character should depend on as many plaintext characters and as many characters of the encryption key as possible
- “Avalanche effect” (small modification, big impact)

■ Trapdoor function (one-way function)

- Fast in one direction, not in the opposite direction (without secret information)
- Possessing the secret allows the function to work in the opposite direction (access to the trapdoor)

* See http://en.wikipedia.org/wiki/Moscow-Washington_hotline



Examples of Breaches of Kerckhoffs' Principle

The secret should lie within the key, not in the algorithm

- **Cell phone encryption penetrated** (December 1999)

"Israeli researchers discovered design flaws that allow the descrambling of supposedly private conversations carried by hundreds of millions of wireless phones. Alex Biryukov and Adi Shamir describe in a paper to be published this week how a PC with 128 MB RAM and large hard drives can penetrate the security of a phone call or data transmission in less than one second. The flawed algorithm appears in digital GSM phones made by companies such as Motorola, Ericsson, and Siemens, and used by well over 100 million customers in Europe and the United States." [...]

*"Previously the GSM encryption algorithms have come under fire **for being developed in secret away from public scrutiny** -- but most experts say high security can only come from published code. Moran [GSM Association] said "it wasn't the attitude at the time to publish algorithms" when the A5 ciphers was developed in 1989, but **current ones being created will be published for peer review.**"*

[<http://www.wired.com/politics/law/news/1999/12/32900>]

- **Additional example:** In 1999, Netscape Navigator stored email server passwords using a weak proprietary encryption method.

Sample of a One-Time Pad Adaptation



Clothes hanger of a Stasi agent
with a secret one-time pad
(source: *Spiegel Spezial*, 1/1990)

Menu:
“Crypt/Decrypt” \
“Symmetric (classic)” \
“Vernam”

Key Distribution Problem

Key distribution for symmetric encryption methods

If 2 persons communicate with each other using symmetric encryption, they **need one common secret key**.

If n persons communicate with each other, then they need $S_n = n * (n-1) / 2$ keys.

That is:

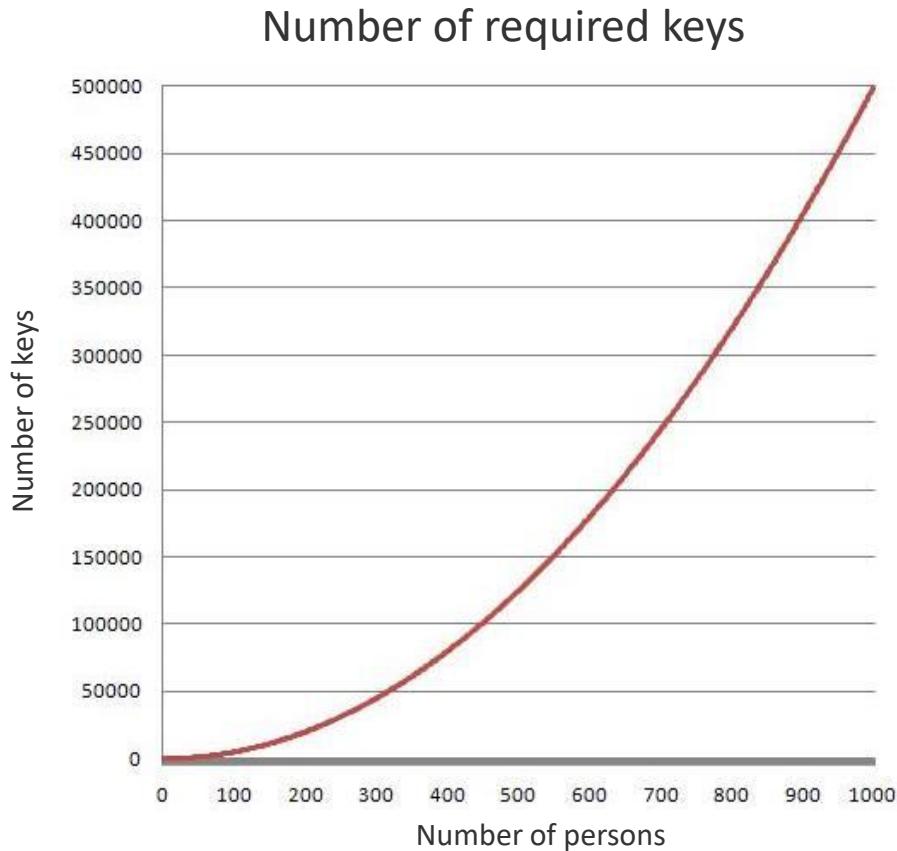
$n = 100$ persons require

$S_{100} = 4,950$ keys; and

$n = 1,000$ persons require

$S_{1000} = 499,500$ keys.

⇒ A factor of 10 more persons means a factor of 100 more keys.



Cryptography – Important Insights (2)

Solving the key distribution problem through asymmetric cryptography

Asymmetric cryptography

- For centuries it was believed that sender and receiver need to know the same secret.
- New idea: Every person needs a key pair (which also solves the key distribution problem).

Asymmetric encryption

- “Everyone can lock a padlock or drop a letter in a mail box.”
- MIT, 1977: Leonard Adleman, Ron Rivest, Adi Shamir (well known as RSA)
- GCHQ Cheltenham, 1973: James Ellis, Clifford Cocks (publicly declassified December 1997)

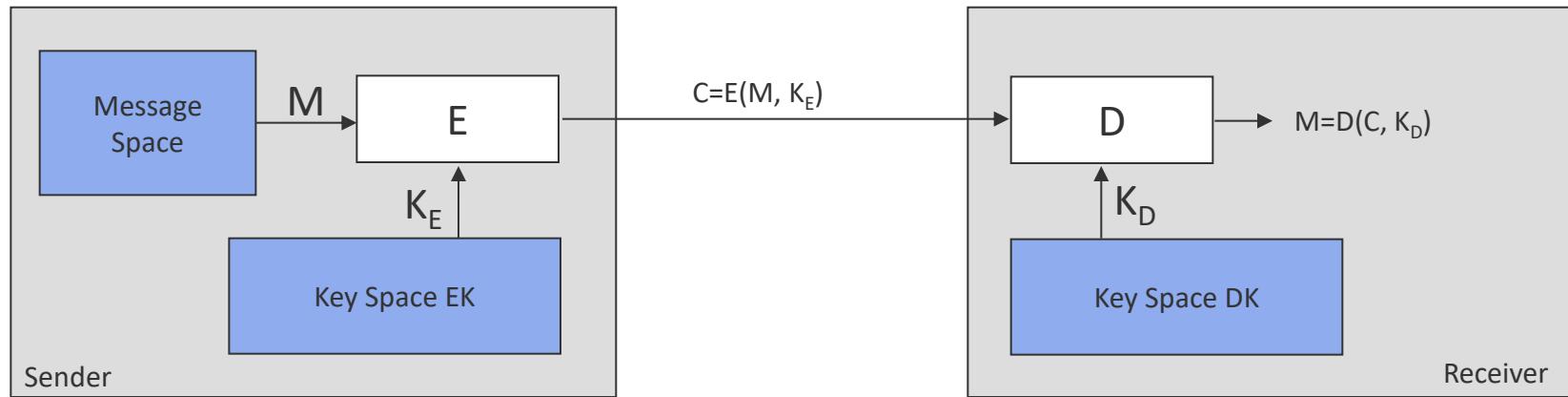
Key distribution

- Stanford, 1976: Whitfield Diffie, Martin Hellman, Ralph Merkle
(Diffie-Hellman key exchange)
- GCHQ Cheltenham, 1975: Malcolm Williamson

*Security in open networks (such as the Internet)
would be extremely expensive and complex without
asymmetric cryptography!*

Performing Encryption and Decryption

Symmetric und asymmetric encryption



a) Symmetric Encryption:

$$K_E = K_D \quad (\text{e.g. AES})$$

b) Asymmetric Encryption:

$$K_E \neq K_D \quad (\text{e.g. RSA})$$

secret
public private/secret

Cryptography – Important Insights (3)

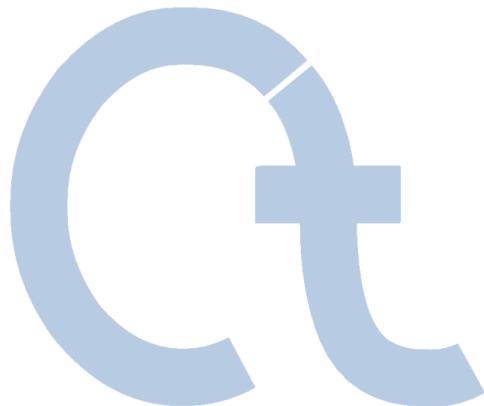
Increasing relevance of mathematics and information technology

- **Modern cryptography** is increasingly based on **mathematics**
 - There are still new symmetric encryption methods, such as AES; these often feature better performance and shorter key length compared to asymmetric methods that are based purely on mathematical problems.
- The security of encryption methods heavily depends on the current state of **mathematics** and **information technology (IT)**
 - Computation complexity (meaning processing effort in relation to key length, storage demand, and data complexity)
→ see RSA: Bernstein, TWIRL device, RSA-160, RSA-768 (CrypTool book, chapter 4.11.3)
 - Major topics in current research:
Factorization of very large numbers, non-parallelizable algorithms (to counter quantum computers), protocol weaknesses, random generators, etc.)
- Serious mistake: “Real mathematics has no effects on war.” (G.H. Hardy, 1940)
- Vendors have realized that **security** is an essential **purchase criterion**.
- Wrong believes:
Encryption /data privacy and intelligence / innovation are opposites.

Demonstration in CrypTool

- Statistic Analysis
- Encrypting twice is not always better:
 - Caesar: $C + D = G$ ($3 + 4 = 7$)
 - Vigenère: - CAT + DOG = FOZ $[(2,0,19)+(3,14,6)=(5,14,25)]$
 - "Hund" + "Katze" = "RUGCLENWGYXDATRNHNMH"
- Vernam (OTP)
- AES (output key, brute-force analysis)

Content



- I. CrypTool and Cryptology – Overview
 - II. Features of CrypTool 1**
 - III. Examples
 - IV. Project / Outlook / Contact
- Appendix

1. What is CrypTool?

- Freeware program with graphical user interface
- Cryptographic methods can be applied *and* analysed
- Comprehensive online help (understandable without a deep knowledge of cryptography)
- Contains nearly all state-of-the-art cryptography functions
- Easy entry into modern and classical cryptography
- Not a “*hacker tool*”

2. Why CrypTool?

- Originated in an awareness initiative of a financial institute
- Developed in close cooperation with universities
- Improvement of university education and in-firm training

3. Target group

- *Core group:* Students of computer science, business computing, and mathematics
- *But also for:* computer users, application developers, employees, high school students, etc.
- *Prerequisite:* PC knowledge
- *Preferable:* Interest in mathematics and/or programming

Content of the Program Package



English, German,
Polish, Spanish,
French, and Serbian

CrypTool program

- All functions integrated in a *single* program with consistent graphical interface
- Runs on Win32
- Includes cryptography libraries from Secude, cryptovision, and OpenSSL
- Long integer arithmetic via Miracl, APFLOAT and GMP/MPIR, lattice-based reduction via NTL (V. Shoup)

AES Tool

- Standalone program for AES encryption (and creation of self-extracting files)

Educational game

- “Number Shark” encourages the understanding of factors and prime numbers.

Comprehensive online help (HTML Help)

- Context-sensitive help available via F1 for all program functions (including menus)
- Detailed use cases for most program functions (tutorial)

Book (.pdf file) with background information

- Encryption methods • Prime numbers and factorization • Digital signatures • Elliptic curves
- Bit ciphers • Public-key certification • Basic number theory • Crypto 2020 • Sage

Two short stories related to cryptography by Dr. C. Elsner

- “The Dialogue of the Sisters” (features an RSA variant as key element)
- “The Chinese Labyrinth” (number theory tasks for Marco Polo)

Authorware learning tool for number theory

Features (1)

Cryptography

Classical cryptography

- Caesar (and ROT-13)
- Monoalphabetic substitution (and Atbash)
- Vigenère
- Hill
- Homophone substitution
- Playfair
- ADFGVX
- Byte Addition
- XOR
- Vernam
- Permutation / Transposition (Rail Fence, Scytale, etc.)
- Solitaire

Several options to easily comprehend cryptography samples from literature

- Selectable alphabet
- Options: handling of blanks, etc.

Cryptanalysis

Attack on classical methods

- Ciphertext only
 - Caesar
 - Vigenère (according to Friedman + Schroedel)
 - Addition
 - XOR
 - Substitution
 - Playfair
- Known Plaintext
 - Hill
 - Single-column transposition
- Manual (program supported)
 - Mono alphabetical substitution
 - Playfair, ADFGVX, Solitaire

Supported analysis methods

- Entropy, floating frequency
- Histogram, n-gram analysis
- Autocorrelation
- Periodicity
- Random analysis
- Base64 / UU-Encode

Features (2)

Cryptography

Modern symmetric encryption

- IDEA, RC2, RC4, RC6, DES, 3DES, DESX
- AES candidates of the last selection round (Serpent, Twofish, etc.)
- AES (=Rijndael)
- DESL, DESXL

Asymmetric encryption

- RSA with X.509 certificates
- RSA demonstration
 - For improved understanding of examples from literature
 - Alphabet and block length selectable

Hybrid encryption (RSA + AES)

- Visualized as an interactive data flow diagram

Cryptanalysis

Brute-force attack on symmetric algorithms

- For all algorithms
- Assumptions:
 - Entropy of plaintext is small,
 - Key is partially known, or
 - Plaintext alphabet is known

Attack on RSA encryption

- Factorization of RSA modulus
- Lattice-based attacks

Attack on hybrid encryption

- Attack on RSA, or
- Attack on AES (side-channel attack)

Features (3)

Cryptography

Digital signature

- RSA with X.509 certificates
 - Signature as data flow diagram
- DSA with X.509 certificates
- Elliptic Curve DSA, Nyberg-Rueppel

Hash functions

- MD2, MD4, MD5
- SHA, SHA-1, SHA-2, RIPEMD-160

Random generators

- Secude
- $x^2 \bmod n$
- Linear congruence generator (LCG)
- Inverse congruence generator (ICG)

Cryptanalysis

Attack on RSA signature

- Factorization of the RSA module
- Feasible up to 250 bits or 75 decimal places (on standard desktop PCs)

Attack on hash functions / digital signature

- Generate hash collisions for ASCII based text (birthday paradox) (up to 40 bits in about five minutes)

Analysis of random data

- FIPS-PUB-140-1 test battery
- Periodicity, Vitányi, entropy
- Floating frequency, histogram
- n-gram analysis, autocorrelation
- ZIP compression test

Features (4)

Visualizations / Demos

- Caesar, Vigenère, Nihilist, DES (all with ANIMAL)
- Enigma (Flash)
- Rijndael/AES (two versions with Flash, one with Java)
- Hybrid encryption and decryption (AES-RSA and AES-ECC)
- Generation and verification of digital signatures
- Diffie-Hellman key exchange
- Secret sharing (with CRT or Shamir)
- Challenge-response method (network authentication)
- Side-channel attack
- Secure email with the S/MIME protocol (with Java and Flash)
- Graphical 3D presentation of (random) data streams
- Sensitivity of hash functions regarding plaintext modifications
- Number theory and RSA cryptosystem (with Authorware)



Features (5)

Additional functions

- Different functions for RSA and prime numbers
- Homophone and permutation encryption (Double Column Transposition)
- PKCS #12 import and export for PSEs (Personal Security Environment)
- Hash generation of large files (without loading them)
- Flexible brute-force attacks on any modern symmetric algorithm
- ECC demonstration (as Java application)
- Password quality meter (PQM) and password entropy
- Manifold text options for the classic ciphers (see example p. 99)
- And plenty more...

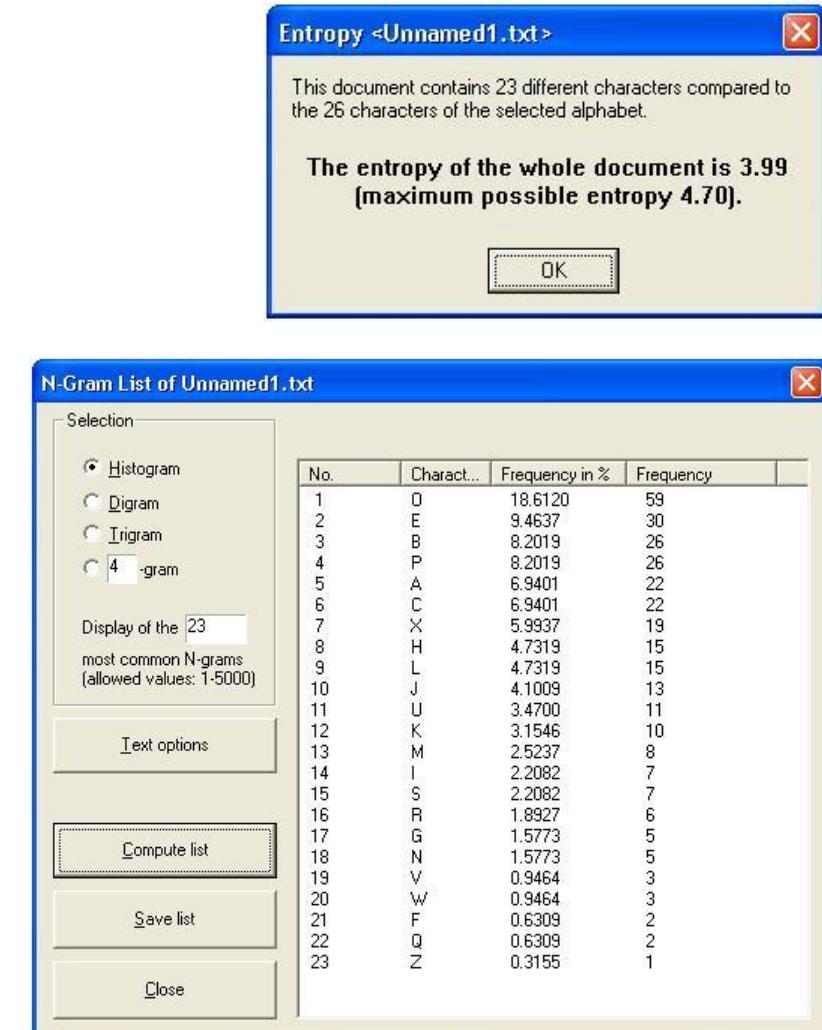
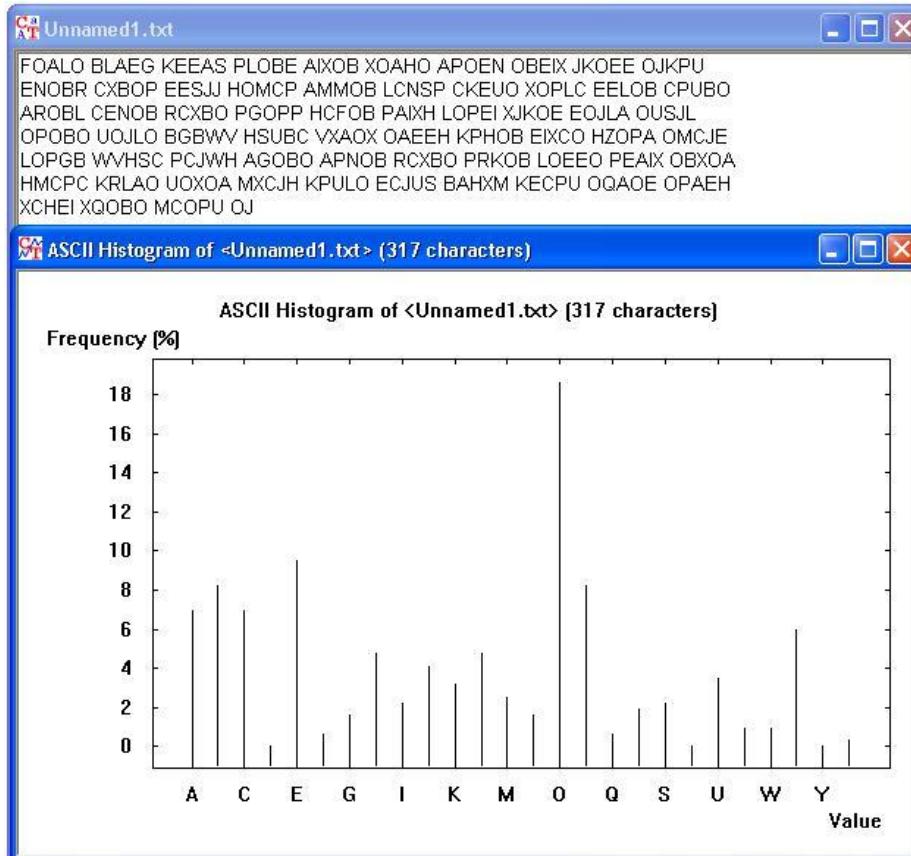


Language Structure Analysis

Language analysis options available in CrypTool 1

Number of characters, n-gram, entropy

- See menu “Analysis” \ “Tools for Analysis” \ ...



Demonstration of Interactivity (1)

Vigenère analysis

Demonstration in
CrypTool

The result of the Vigenère analysis can be manually reworked
(changing the key length)

1. Encrypt the sample file with TESTETE

- “Crypt/Decrypt” \ “Symmetric (classic)” \ “Vigenère”
- Enter TESTETE \Rightarrow “Encrypt”

Analysis of the encryption results:

- “Analysis” \ “Symmetric Encryption (classic)” \ “Ciphertext only” \ “Vigenère”
- Derived key length 7, derived key TESTETE

2. Encrypt starting sample with TEST

- “Crypt/Decrypt” \ “Symmetric (classic)” \ “Vigenère”
- Enter TEST \Rightarrow “Encrypt”

Analysis of the encryption results:

- “Analysis” \ “Symmetric Encryption (classic)” \ “Ciphertext only” \ “Vigenère”
- Derived key length 8 – incorrect
- Key length automatically set to 4 (can also be adjusted manually)
- Derived key TEST

Demonstration of Interactivity (2)

Automated factorization

*Demonstration in
CrypTool*

Factorization of a compound number with factorization algorithms

- The algorithms are executed in parallel (multi-threaded)
- Each algorithm has specific advantages and disadvantages; for example, some methods can only determine small factors

Factorization example 1

316775895367314538931177095642205088158145887517

=

3 * 1129 * 6353 * 1159777 * 22383173213963 * 567102977853788110597

48-digit decimal number

Factorization example 2

$2^{250} - 1$

=

3 * 11 * 31 * 251 * 601 * 1801 * 4051 * 229668251 * 269089806001 * 4710883168879506001 *
5519485418336288303251

75-digit decimal number

Menu: “Indiv. Procedure” \ “RSA Cryptosystem” \ “Factorization of a Number”

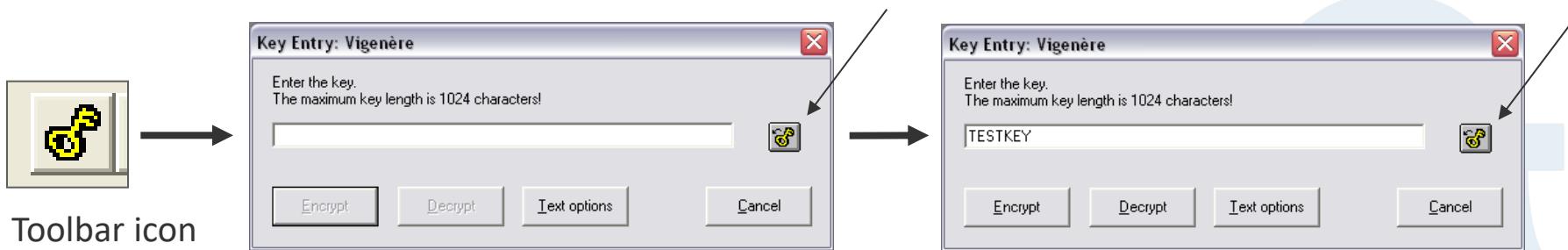
Concepts for a User-Friendly Interface

1. Context sensitive help (F1)

- F1 on a selected menu entry shows information about the algorithm/method.
- F1 in a dialog box explains the usage of the dialog.
- These assistants and the contents of the top menus are cross-linked in the online help.

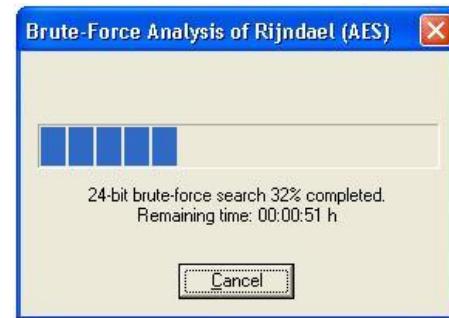
2. Copying keys to the key entry dialog

- CTRL-V can always be used to paste contents from the clipboard.
- Stored keys can be copied from ciphertext windows via an icon in the toolbar. A corresponding icon in the key entry dialog can be used to paste the key into the key field. CrypTool uses an **internal keystore**, which is available for every method of the program. (This is particularly helpful for large “specific” keys, such as in homophone encryption.)

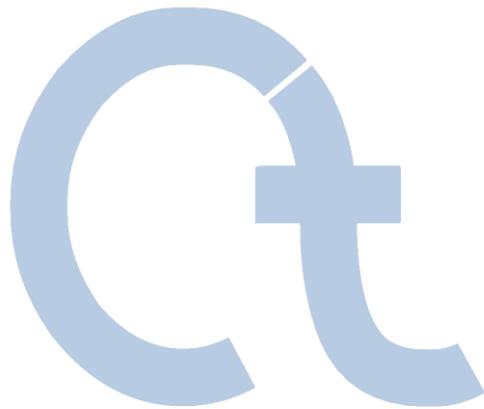


Challenges for Developers (Examples)

- 1. Allow additional functions to run in parallel**
 - Factorization already uses multi-threading to run several algorithms at once
- 2. High performance**
 - Locate hash collisions (birthday paradox) or perform brute force analysis
- 3. Consider memory limits**
 - In particular with regard to the Floyd algorithm (mappings to locate hash collisions) and quadratic sieve factorization
- 4. Time measurement and estimation**
 - Display remaining time (e.g. while using brute force)
- 5. Reusability / Integration**
 - Forms for prime number generation
 - RSA cryptosystem (switches the view after successful attack from public key user to private key owner)
- 6. Partially automate the consistency of functions, GUI, and online help**
(including different languages and the supported Windows operating systems)



Content



- I. CrypTool and Cryptology – Overview
- II. Features of CrypTool 1
- III. Examples**
- IV. Project / Outlook / Contact
- Appendix

CrypTool Examples

Overview of examples

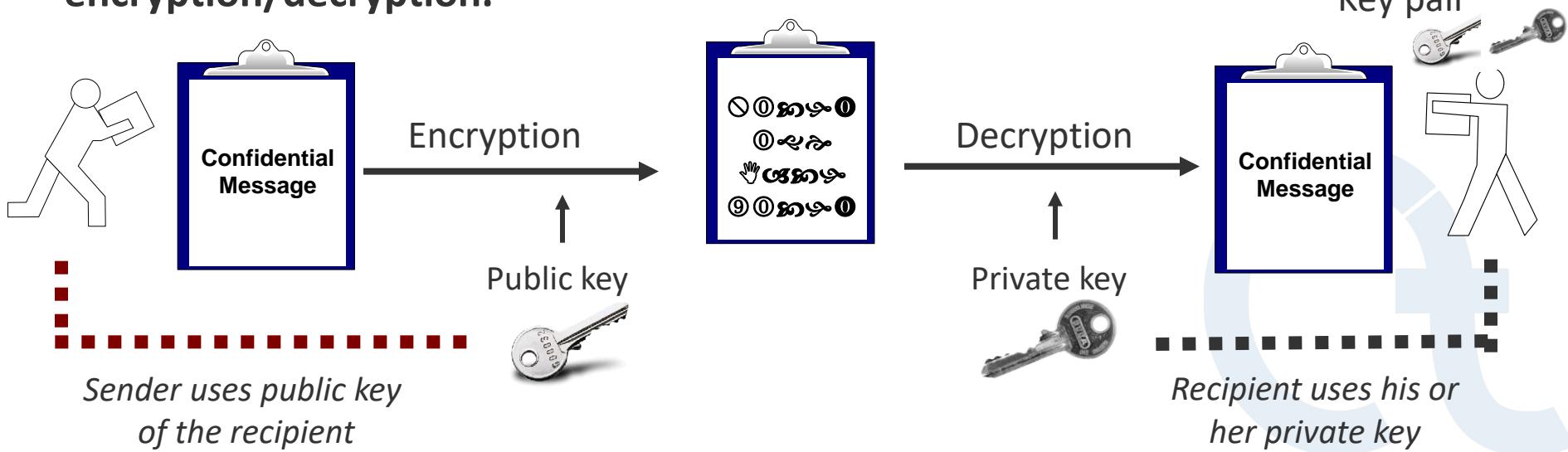
1. [Encryption with RSA / Prime number tests / Hybrid encryption and digital certificates / SSL](#)
2. [Digital signature visualized](#)
3. [Attack on RSA encryption \(small modulus N\)](#)
4. [Analysis of encryption in PSION 58](#)
5. [Weak DES keys](#)
6. [Locating key material \("NSA key"\)](#)
7. [Attack on digital signature through hash collision search](#)
8. [Authentication in a client-server environment](#)
9. [Demonstration of a side-channel attack \(on hybrid encryption protocol\)](#)
10. [Attack on RSA using lattice reduction](#)
11. [Random analysis with 3-D visualization](#)
12. [Secret Sharing using the Chinese Remainder Theorem \(CRT\) and Shamir](#)
13. [Implementation of CRT in astronomy \(solving systems of linear modular equations\)](#)
14. [Visualization of symmetric encryption methods using ANIMAL](#)
15. [Visualizations of AES](#)
16. [Visualization of Enigma encryption](#)
17. [Visualization of Secure Email with S/MIME](#)
18. [Generation of a message authentication code \(HMAC\)](#)
19. [Hash demonstration](#)
20. [Educational tool for number theory and asymmetric encryption](#)
21. [Point addition on elliptic curves](#)
22. [Password quality meter \(PQM\) and password entropy](#)
23. [Brute force analysis](#)
24. [Scytale / Rail Fence](#)
25. [Hill encryption / Hill analysis](#)
26. [CrypTool online help / Menu tree of the program](#)



Examples (1)

Encryption with RSA

- Basis of the SSL protocol (access to protected websites), among others
- Asymmetric encryption using RSA
 - Every user has a key pair – one public and one private key.
 - Sender encrypts with public key of the recipient.
 - Recipient decrypts with his or her private key.
- Usually implemented in combination with symmetric methods (hybrid encryption): The symmetric key is transmitted using RSA asymmetric encryption/decryption.



Examples (1)

Encryption using RSA – Mathematical background / algorithm

- Public key: (n, e) [the modulus N is often capitalized]
- Private key: (d)

where

p, q are large, randomly chosen prime numbers with $n = p * q$;

d is calculated under the constraints $\text{gcd}[\varphi(n), e] = 1$; $e * d \equiv 1 \pmod{\varphi(n)}$.

Encryption and decryption operation: $(m^e)^d \equiv m \pmod{n}$

- n is the modulus (its length in bits is referred to as the key length of RSA).
- gcd = greatest common divisor.
- $\varphi(n)$ is Euler's totient function.

Procedure

- Transform the message into its binary representation
- Encrypt message block-wise such that $m = m_1, \dots, m_k$ where for all $m_j: 0 \leq m_j < n$;
The maximum block size r should be chosen such that $2^r \leq n$ (and $2^r - 1 < n$)

Hint: Attractive, interactive Flash animation about the basics of the RSA cipher:

<https://www.crypto1.org/images/ct1/presentations/RSA/RSA-Flash-en/player.html>

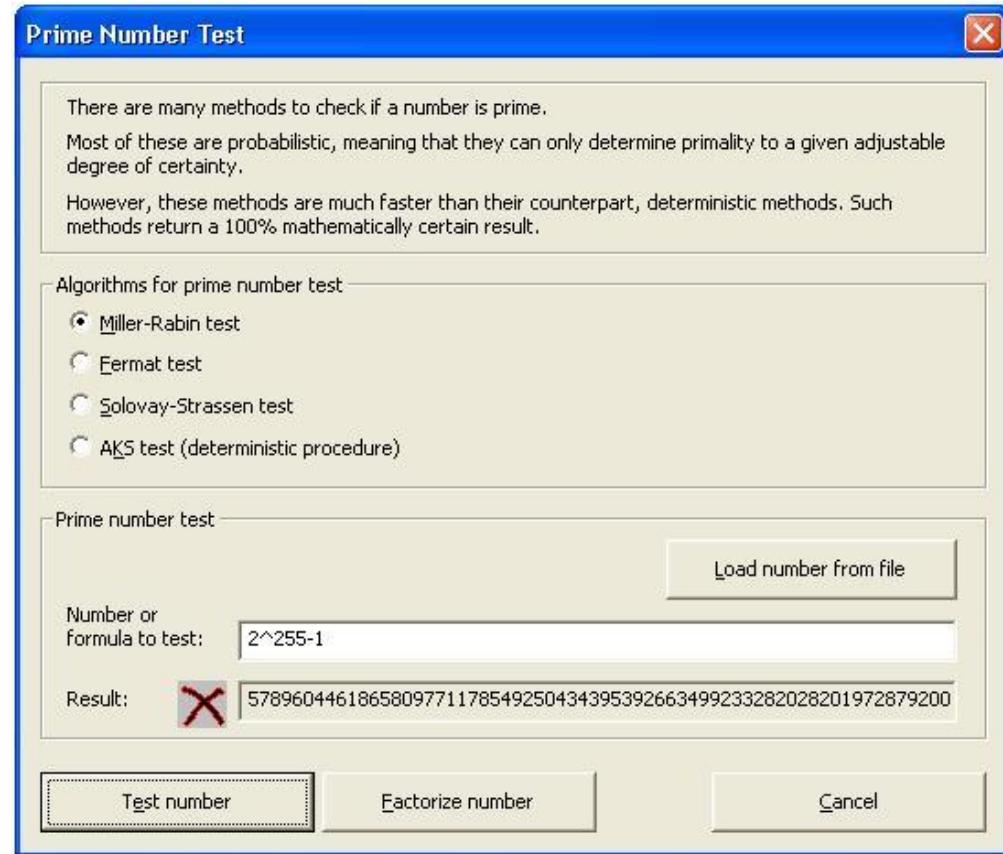
Examples (1)

Prime number tests – RSA requires the use of very large primes

- Fast probabilistic tests
- Deterministic tests

The prime number test methods can test whether a large number is prime much faster than the known factorization methods can divide a number of similar size into its prime factors.

For the AKS test the GMP / MPIR library (**GNU Multiple Precision Arithmetic Library; Multiple Precision Integers and Rationals**) was integrated into CrypTool.



Menu: "Indiv. Procedures" \ "RSA Cryptosystem" \ "Prime Number Test"

Remark: $2^{255} - 1 = 7 * 31 * 103 * 151 * 2143 * 11119 * 106591 * 131071 * 949111 * 9520972806333758431 * 5702451577639775545838643151$

Examples (1)

Printing of the current prime number records – Mersenne primes

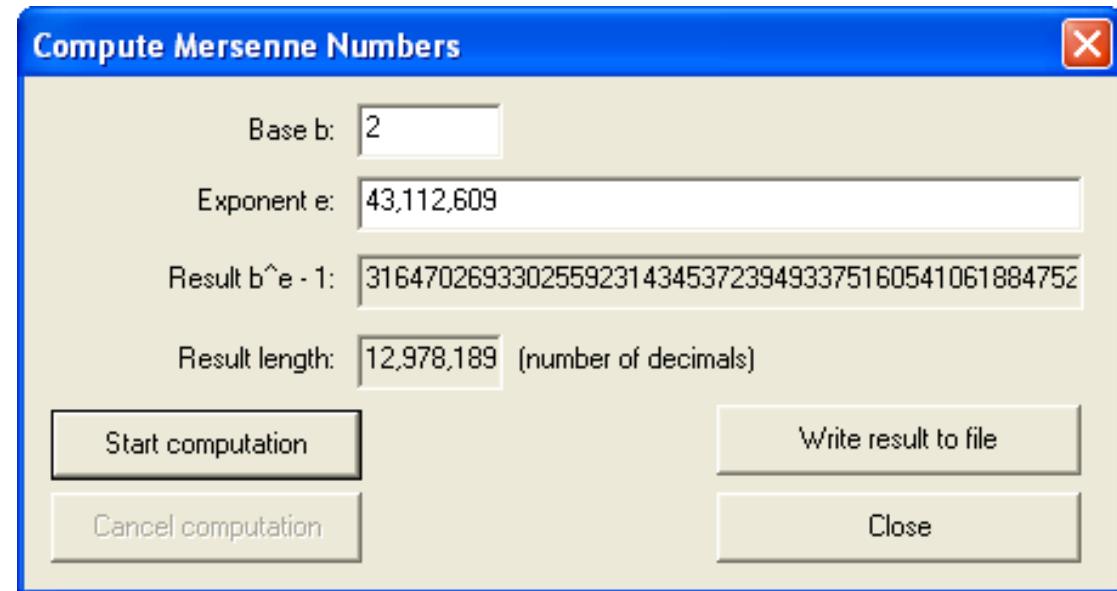
The biggest known primes are so called Mersenne primes.

The currently 4th biggest one has 12,978,189 decimal digits and was discovered in 2008 by the GIMPS project.

The adjoining dialog allows to calculate and write all digits of such numbers very fast.

To do so the APFLOAT library was integrated into CrypTool.

Within the context menu of each input or output field of this dialog you can switch on and off the thousands separator.



Remark: $2^{43,112,609} - 1 = 316,470,269 \dots 697,152,511$

Large numbers should not be marked and copied from the "Result" field – because of the performance of the GUI. Please use the button "Write result to file" in order to show the resulting number in its completeness within the CrypTool main window.

Menu: "Indiv. Procedures" \ "Number Theory – Interactive" \ "Compute Mersenne Numbers"

Examples (1)

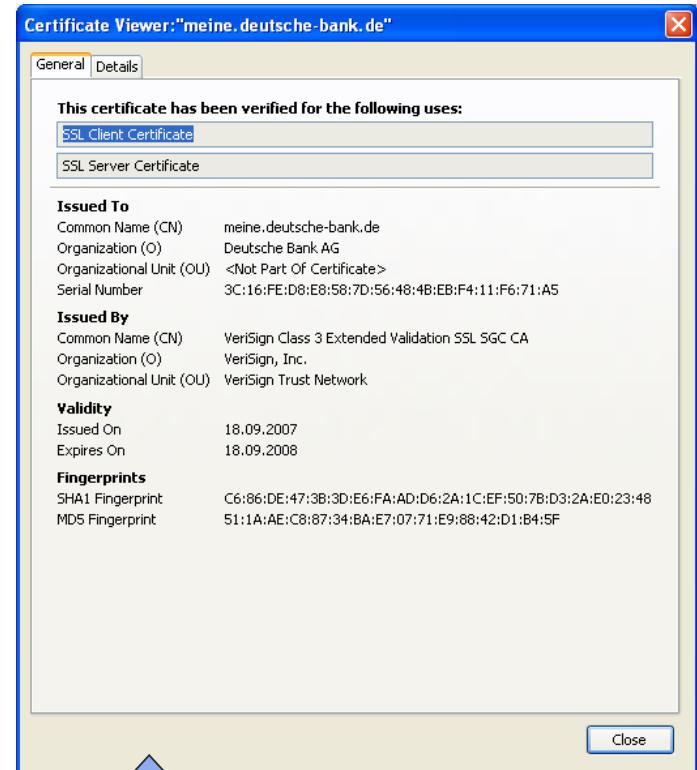
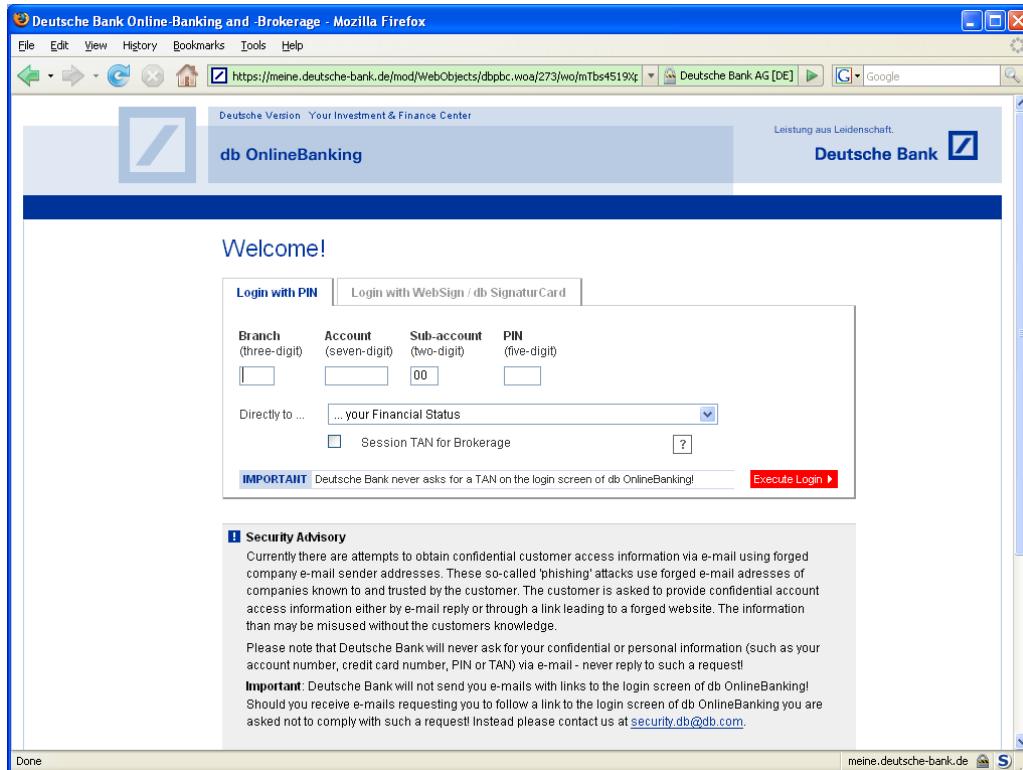
Hybrid encryption and digital certificates

- Hybrid encryption – **combination of asymmetric and symmetric encryption**
 1. Generation of a random symmetric key (session key)
 2. Session key is transferred – protected by asymmetric key
 3. Message is transferred – protected by session key
- Problem: **Man-in-the-middle attacks – does the public key of the recipient really belong to the recipient?**
- Solution: digital certificates – a central instance (e.g., GlobalSign, Telesec, VeriSign, Deutsche Bank PKI), trusted by all users, ensures the authenticity of the certificate and the associated public key (similar to a passport issued by a national government).
- Hybrid encryption based on digital certificates **is the foundation for all secured electronic communication**
 - Internet shopping and online banking
 - Secure email



Examples (1)

Secured online connection using SSL and certificates

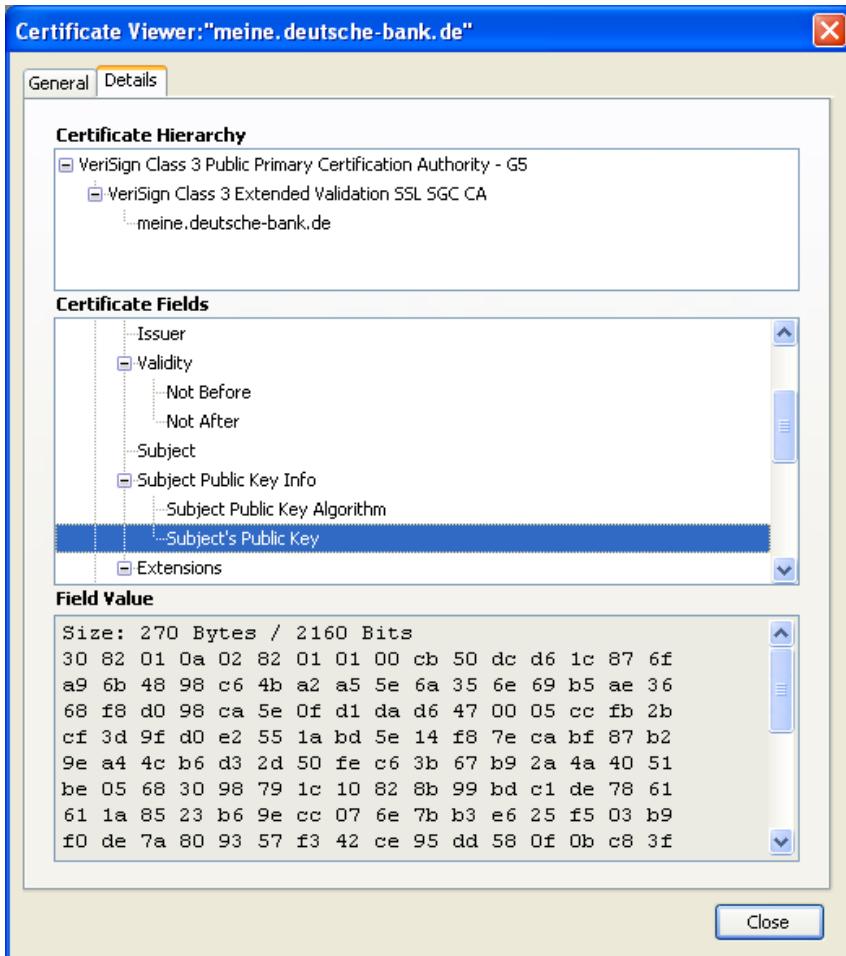


This means that the connection is authenticated (at least on one side) and that the transferred data is strongly encrypted.



Examples (1)

Attributes / fields of a certificate



General attributes / fields

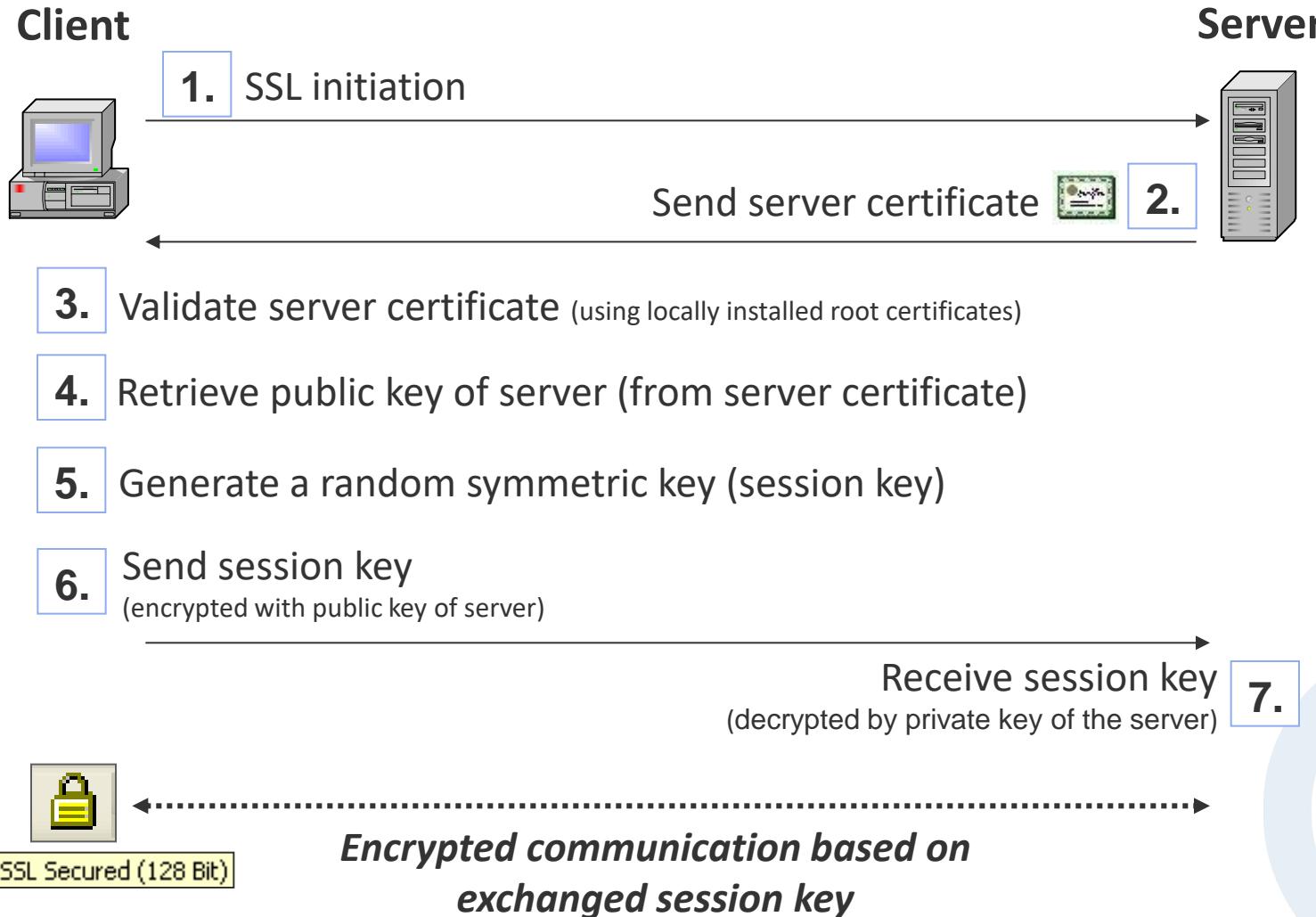
- Issuer (e.g., VeriSign)
- Requestor
- Validity period
- Serial number
- Certificate type / version (X.509v3)
- Signature algorithm
- Public key (and method)

Public key



Examples (1)

Establishing a secure SSL connection (server authentication)



Examples (1)

Establishing a secure SSL connection (server authentication)

General

- The example shows the typical SSL connection establishment in order to transfer sensitive data over the internet (e.g. online shopping).
- During SSL connection establishment only the server is authenticated using a digital certificate (authentication of the user usually occurs through user name and password after the SSL connection has been established).
- SSL also offers the option for client authentication based on digital certificates.

Remarks on establishing an SSL connection (see previous slide)

- Step 1: SSL Initiation – the characteristics of the session key (e.g. bit size) as well as the symmetric encryption algorithm (e.g. 3DES, AES) are negotiated.
- Step 2: In a multi-level certificate hierarchy, the required intermediate certificates are also passed to the client.
- Step 3: The root certificates installed in the browser's certificate store are used to validate the server certificate.
- Step 5: The session key is based on the negotiated characteristics (see step 1).

Examples (2)

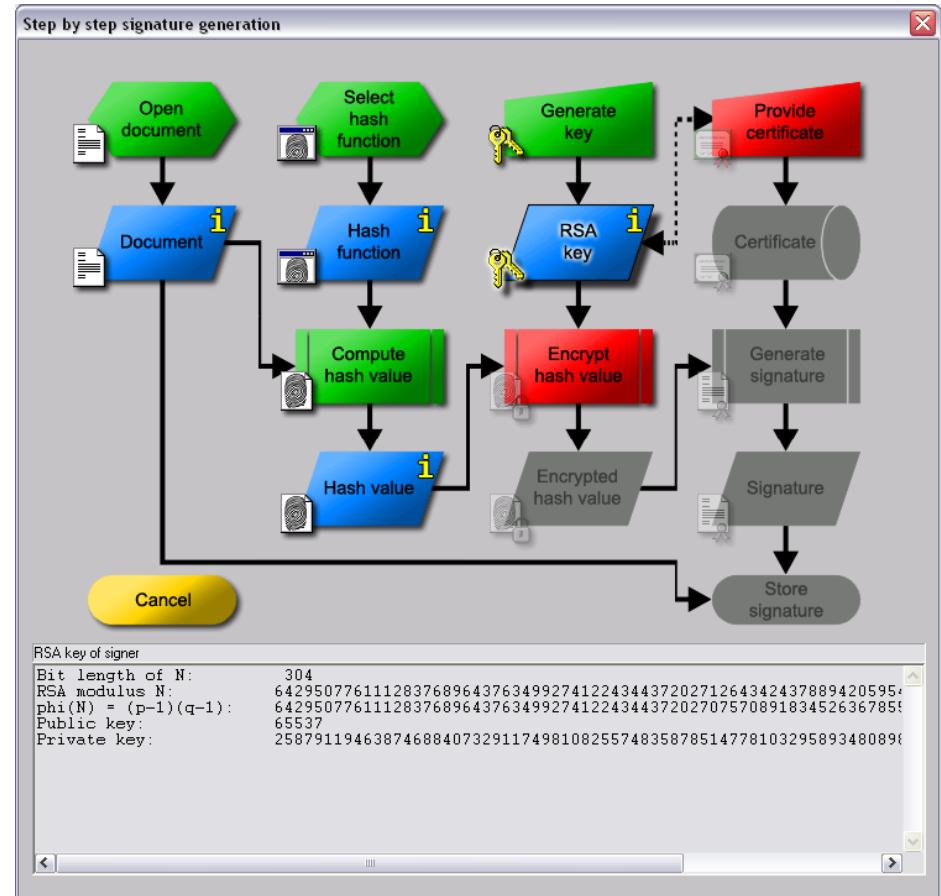
Digital signature visualized

Digital signature

- Increasingly important
 - Equivalent to a handwritten signature (digital signature law)
 - increasingly used by companies, governments, and consumers
- Few actually know how it works

Visualization in CrypTool

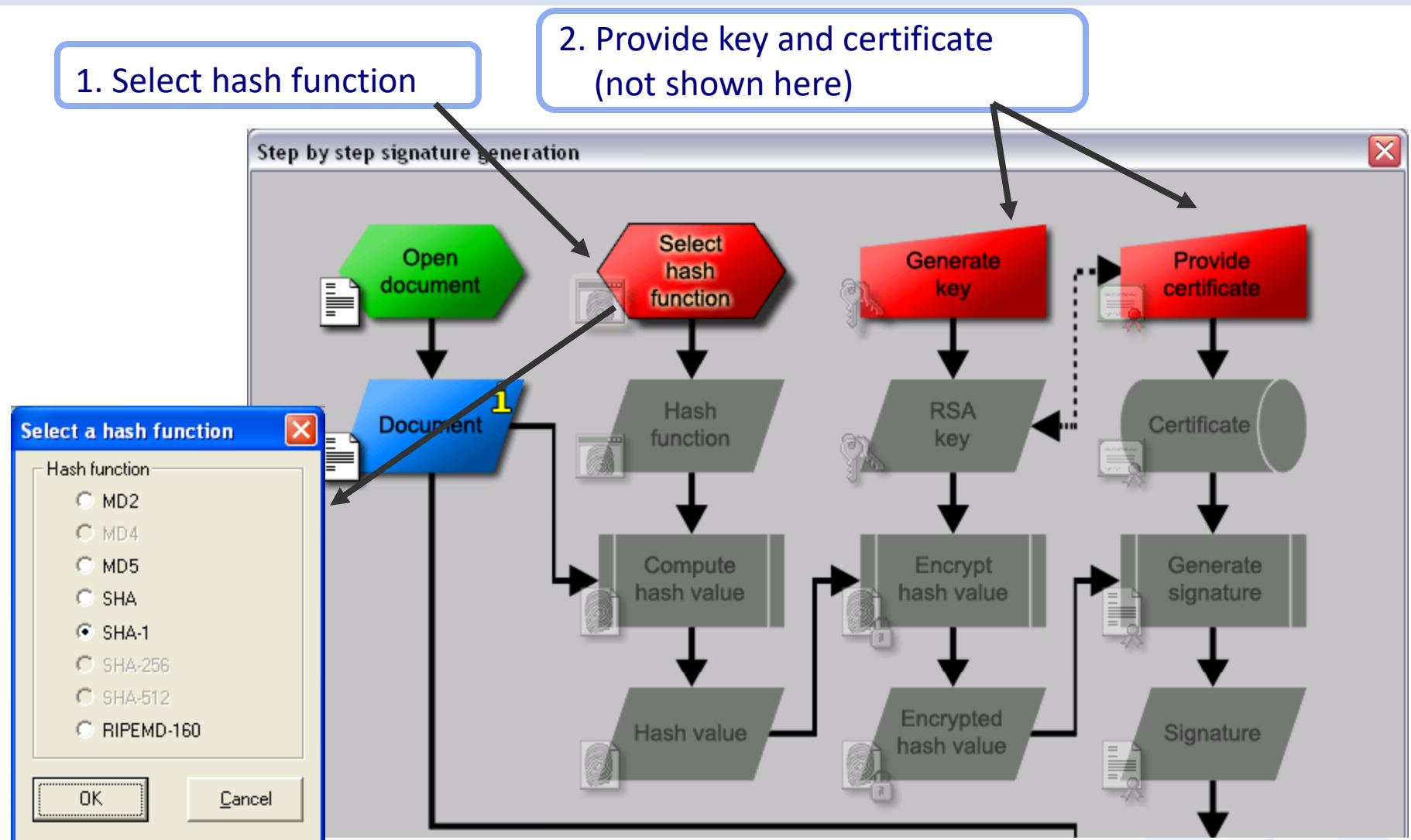
- Interactive data flow diagram
- Similar to the visualization of hybrid encryption



Menu: “Digital Signatures/PKI” \
“Signature Demonstration (Signature Generation)”

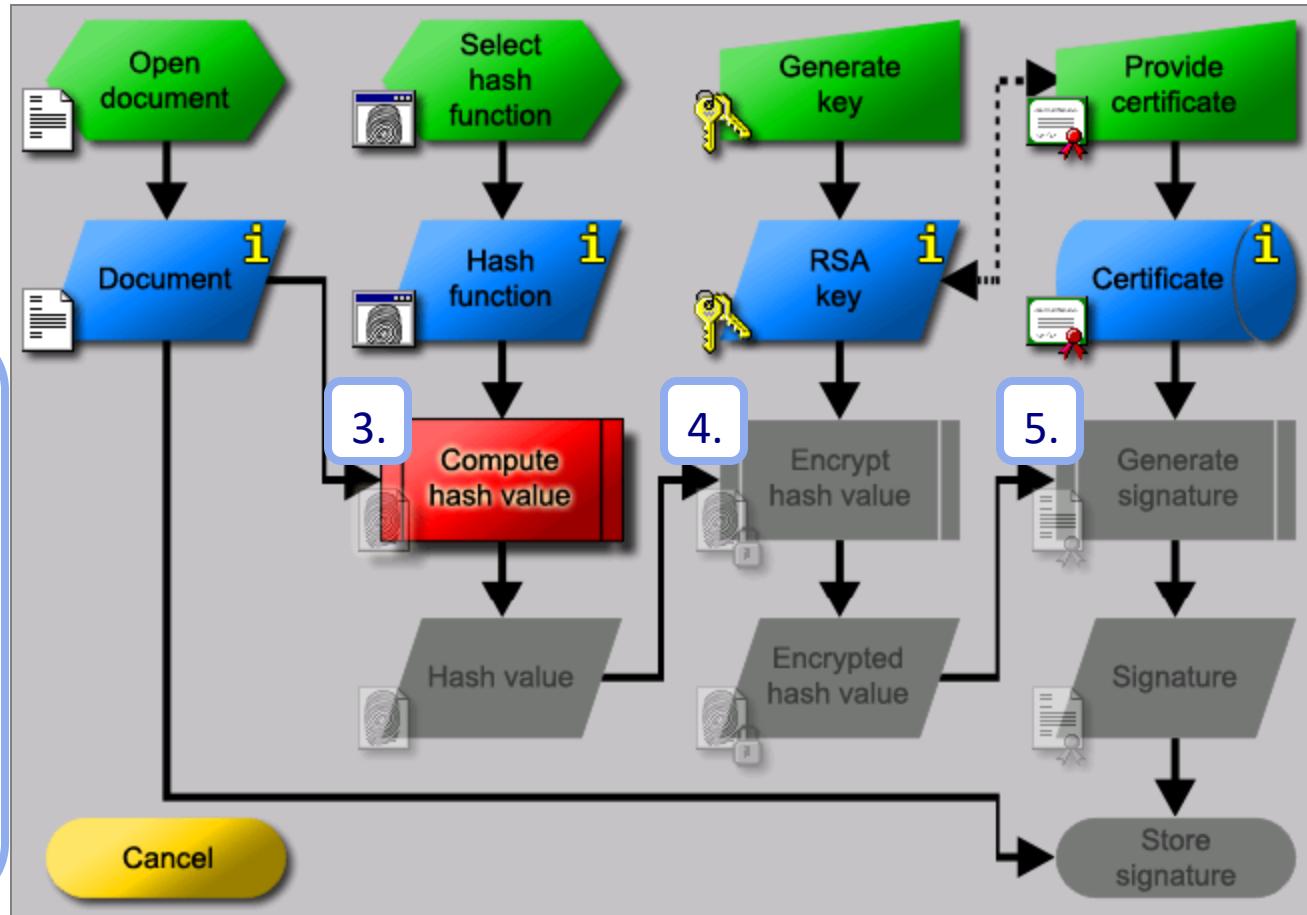
Examples (2)

Digital signature visualized: a) Preparation



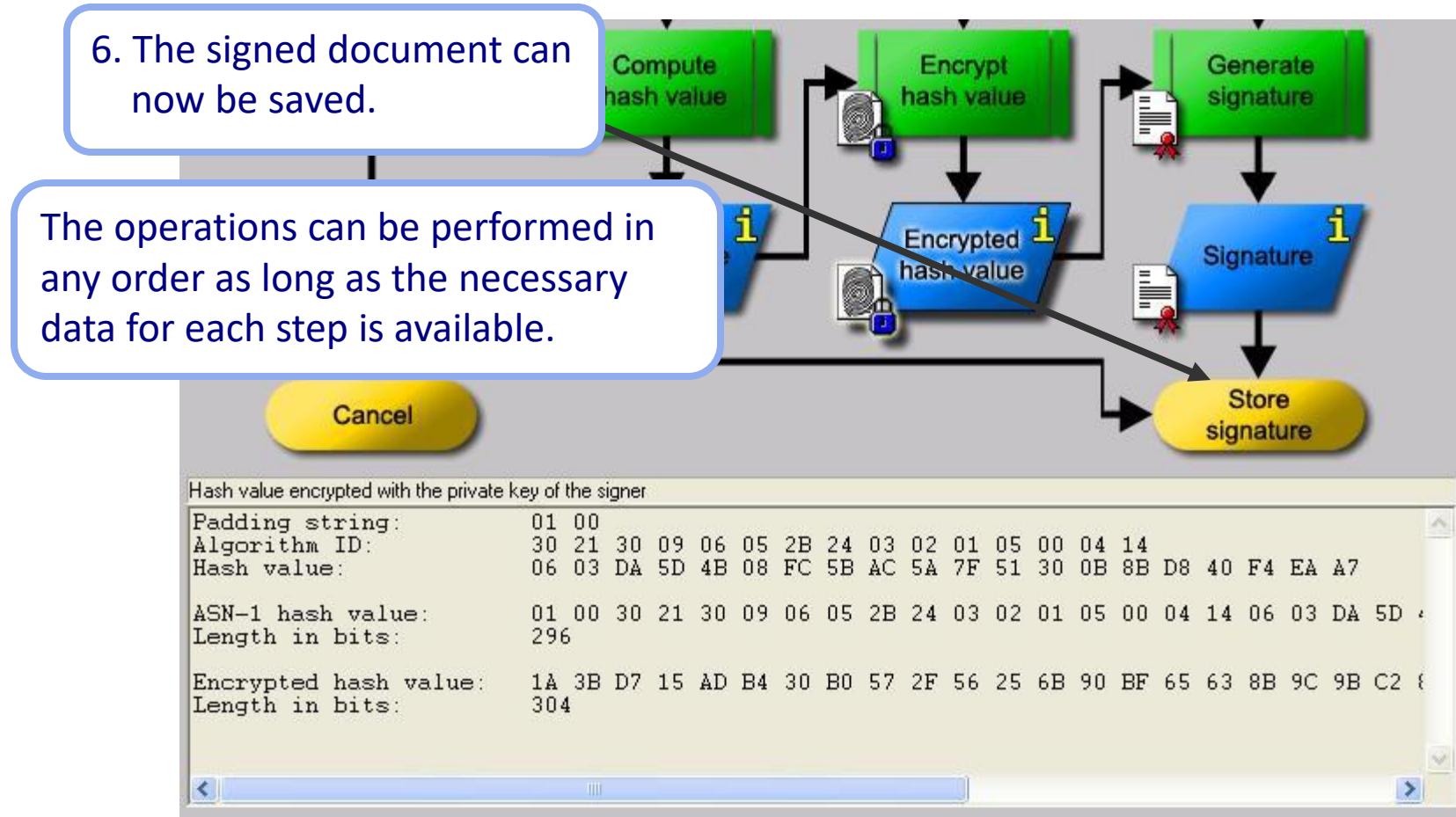
Examples (2)

Digital signature visualized: b) Cryptography



Examples (2)

Digital signature visualized: c) Result



Examples (3)

Attack on RSA encryption with short RSA modulus

Example from Song Y. Yan, *Number Theory for Computing*, Springer, 2000

- Public key
 - RSA modulus **N = 63978486879527143858831415041** (95 bits, 29 decimal digits)
 - public exponent **e = 17579**
- Ciphertext (block length = 8):
 $C_1 = 45411667895024938209259253423,$
 $C_2 = 16597091621432020076311552201,$
 $C_3 = 46468979279750354732637631044,$
 $C_4 = 32870167545903741339819671379$
- This text must be deciphered!

To perform the actual cryptanalysis (revealing the private key), the ciphertext is not actually necessary!

Solution using CrypTool (further details in the examples section of the online help)

- Enter public parameters into “RSA cryptosystem” (menu: “Indiv. Procedures”)
- Clicking the button “Factorize the RSA modulus” yields the two prime factors $pq = N$
- Based on that information the private exponent $d = e^{-1} \bmod (p-1)(q-1)$ can be determined
- Decrypt the ciphertext with d : $M_i = C_i^d \bmod N$

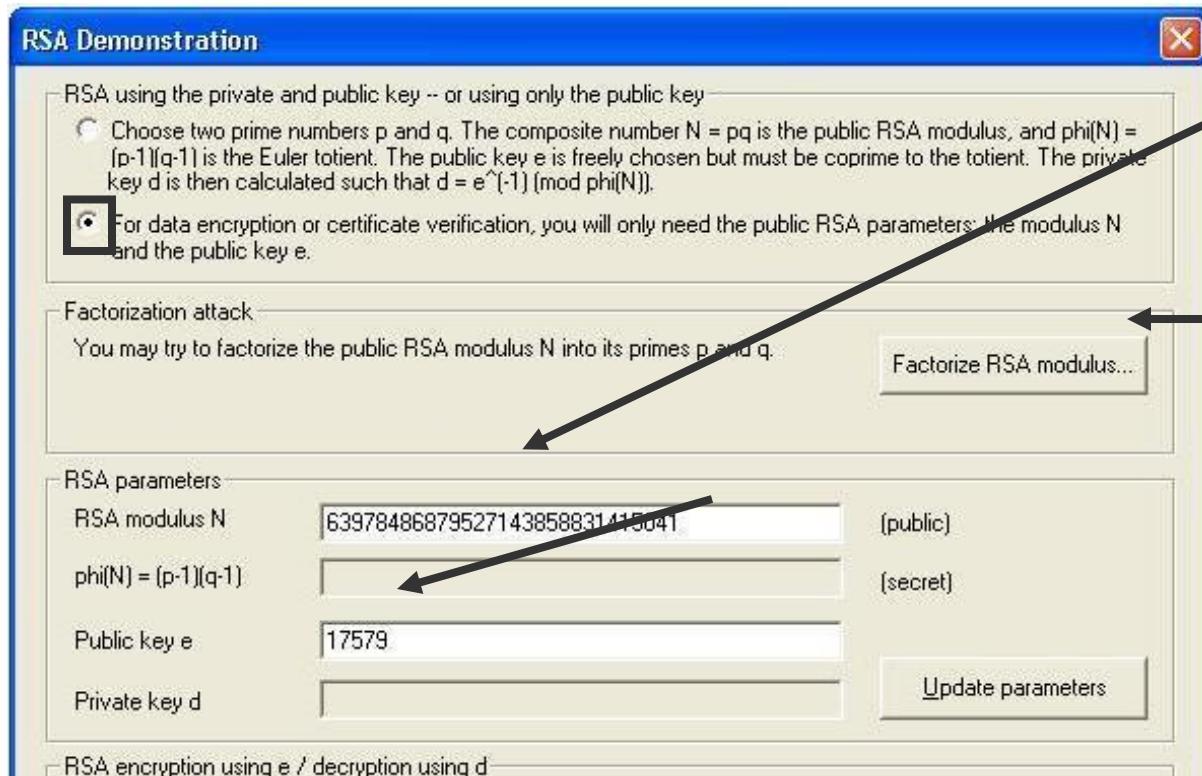
In CrypTool 1, this attack is only practical for RSA key sizes up to about 250 bits.

A successful attack means you could then digitally sign in someone else's name!

Examples (3)

Short RSA modulus: Enter public RSA parameters

Menu: "Indiv. Procedures" \ "RSA Cryptosystem" \ "RSA Demonstration ..."



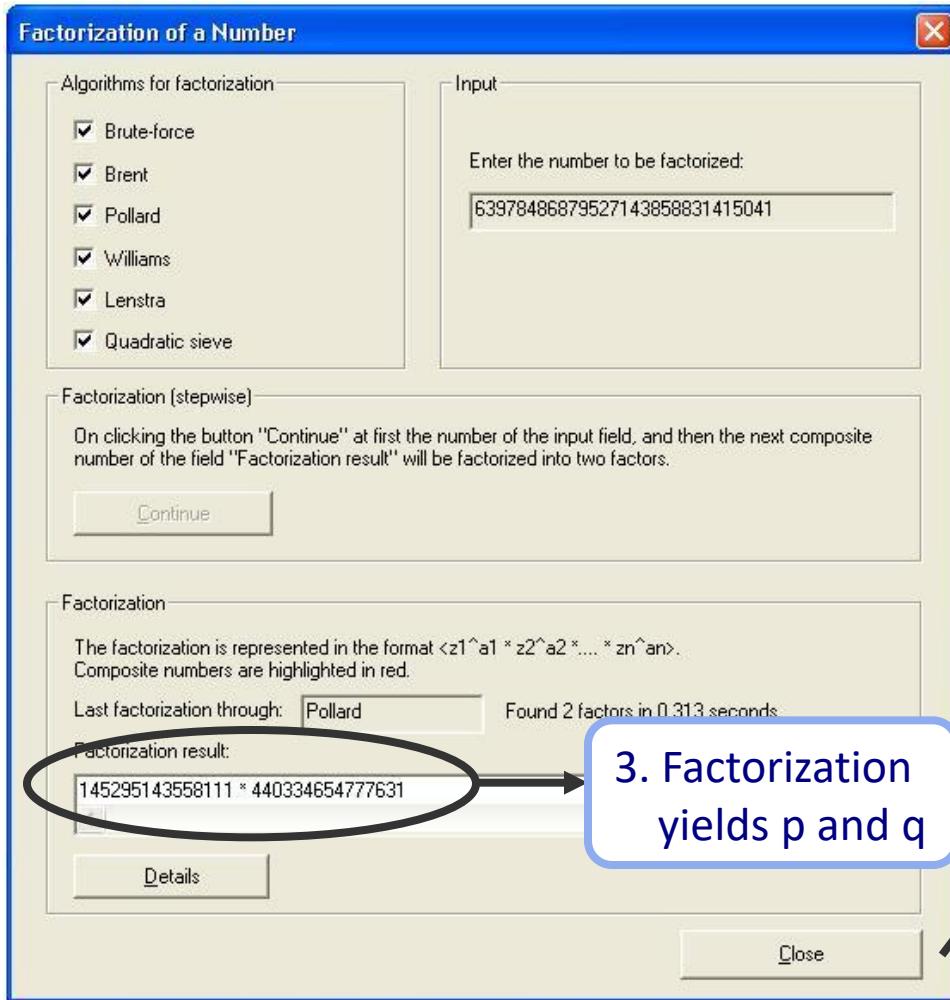
1. Enter public RSA parameters N and e

2. Factorize



Examples (3)

Short RSA modulus: Factorize RSA modulus



Examples (3)

Short RSA modulus: Determine private key d

RSA Demonstration

RSA using the private and public key -- or using only the public key

Choose two prime numbers p and q. The composite number $N = pq$ is the public RSA modulus, and $\phi(N) = (p-1)(q-1)$ is the Euler totient. The public key e is freely chosen but must be coprime to the totient. The private key d is then calculated such that $d = e^{-1} \pmod{\phi(N)}$.

For data encryption or certificate verification, you will only need the public RSA parameters: the modulus N and the public key e.

Prime number entry

Prime number p: 145295143558111

Prime number q: 440334654777631

RSA parameters

RSA modulus N: 63978486879527143858831415041
(public)

$\phi(N) = (p-1)(q-1)$: 63978486879526558229033079300
(secret)

Public key e: 17579

Private key d: 10663687727232084624328285019

Update parameters

RSA encryption using e / decryption using d

Input as: text numbers

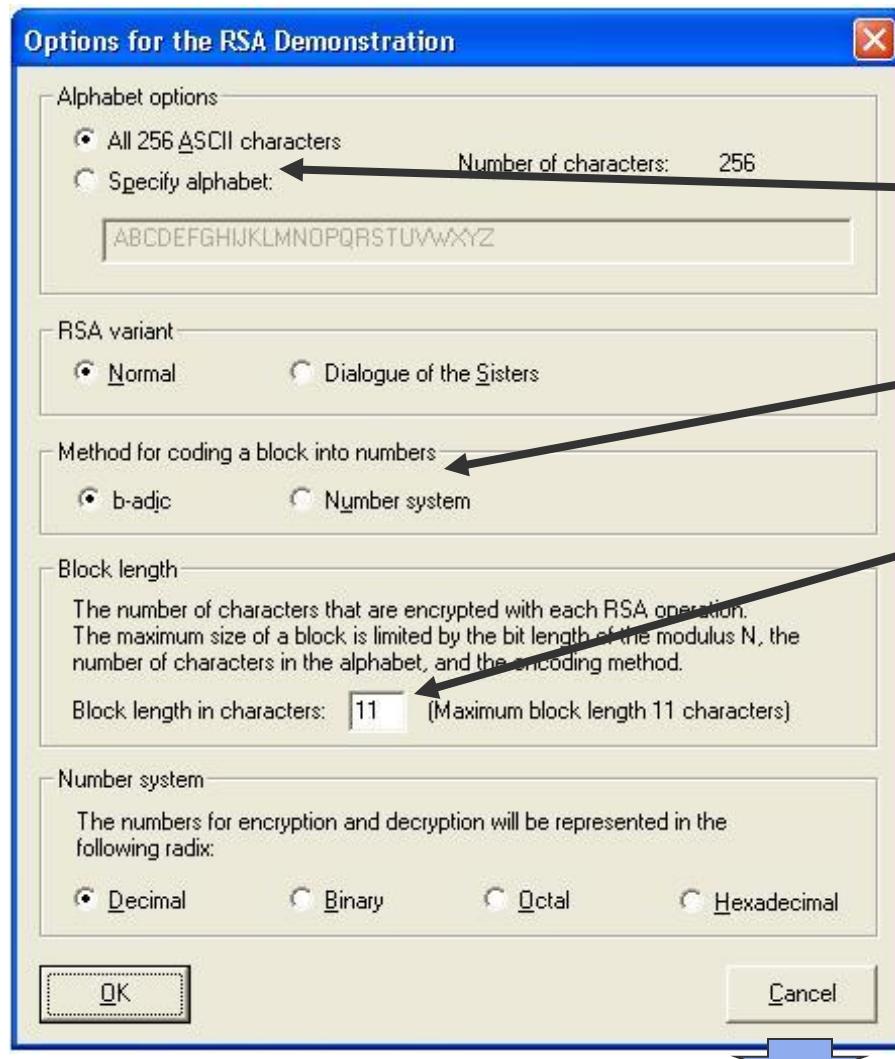
Change the view to the owner of the secret key

4. p and q have been entered automatically, and private key d has been calculated

5. Change settings

Examples (3)

Short RSA modulus: Change settings



6. Select alphabet

7. Select coding method

8. Select block length

Examples (3)

Short RSA modulus: Decrypt ciphertext

RSA parameters

RSA modulus N	63978486879527143858831415041	(public)
$\phi(N) = (p-1)(q-1)$	63978486879526558229033079300	(secret)
Public key e	17579	
Private key d	10663687727232084624328285019	Update parameters

RSA encryption using e / decryption using d

Input as text numbers [Options for alphabet and number system...](#)

Ciphertext coded in numbers of base 10

```
9012514888519448364025235 # 34010413691723826674267175419 # 23969594359517745619250974441
```

Decryption into plaintext $m[i] = c[i]^d \pmod{N}$

```
00088649797025753466833030724 # 00039059986781466364221017938 # 0009220007821449890658414
```

Output text from the decryption (into segments of size 11; the symbol '#' is used as separator).

```
ITS THE END # OF THE WOR # LD AS WE KN # OW IT AND I # FEEL FINE
```

Plaintext

```
ITS THE END OF THE WORLD AS WE KNOW IT AND I FEEL FINE
```

[Encrypt](#) [Decrypt](#) [Close](#)

9. Enter ciphertext

10. Decrypt

Examples (4)

Analysis of encryption used in the PSION 5

Practical application of cryptanalysis

*Attack on the encryption option in the
PSION 5 PDA word processing application*

Starting point: an encrypted file on the PSION

Requirements

- Encrypted English or German text
- Depending on method and key length, text of at least 100 bytes up to several kB

Procedure

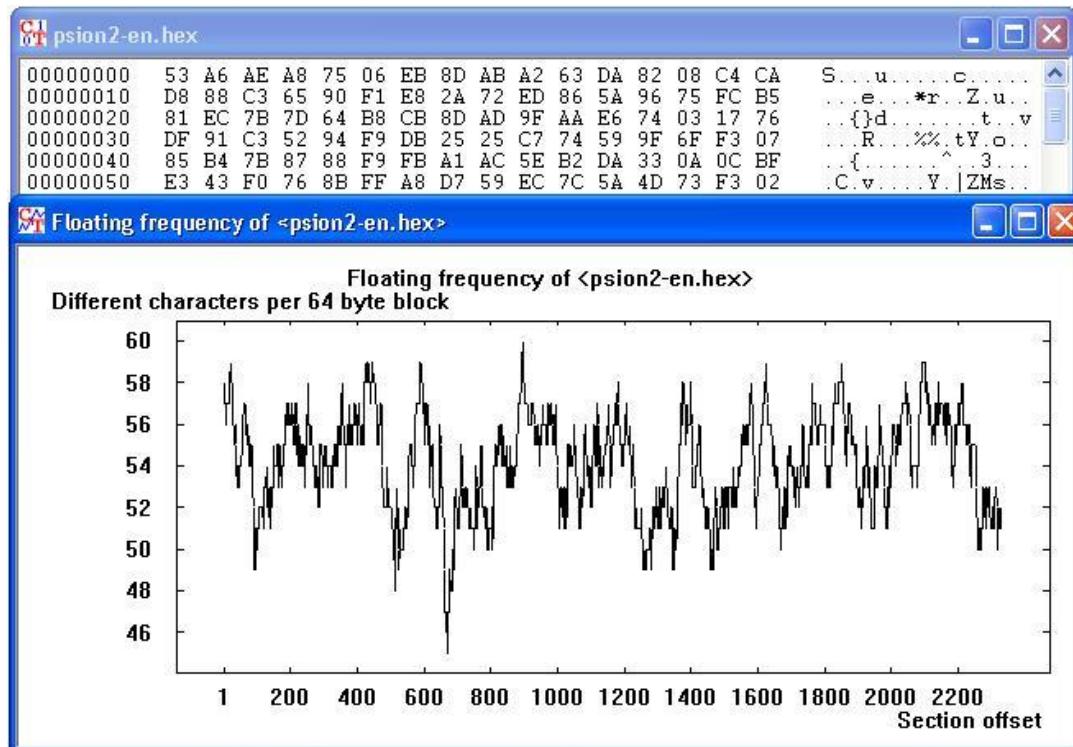
- Pre-analysis
 - entropy
 - floating entropy
 - compression test
- Auto-correlation
- Automated analysis with classical methods

} *probably classical
encryption algorithm*



Examples (4)

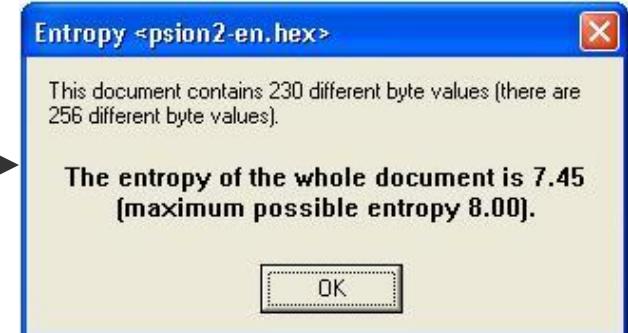
PSION 5 PDA – determine entropy, compression test



Entropy: not all possible values are present, but this does not indicate a specific encryption method.

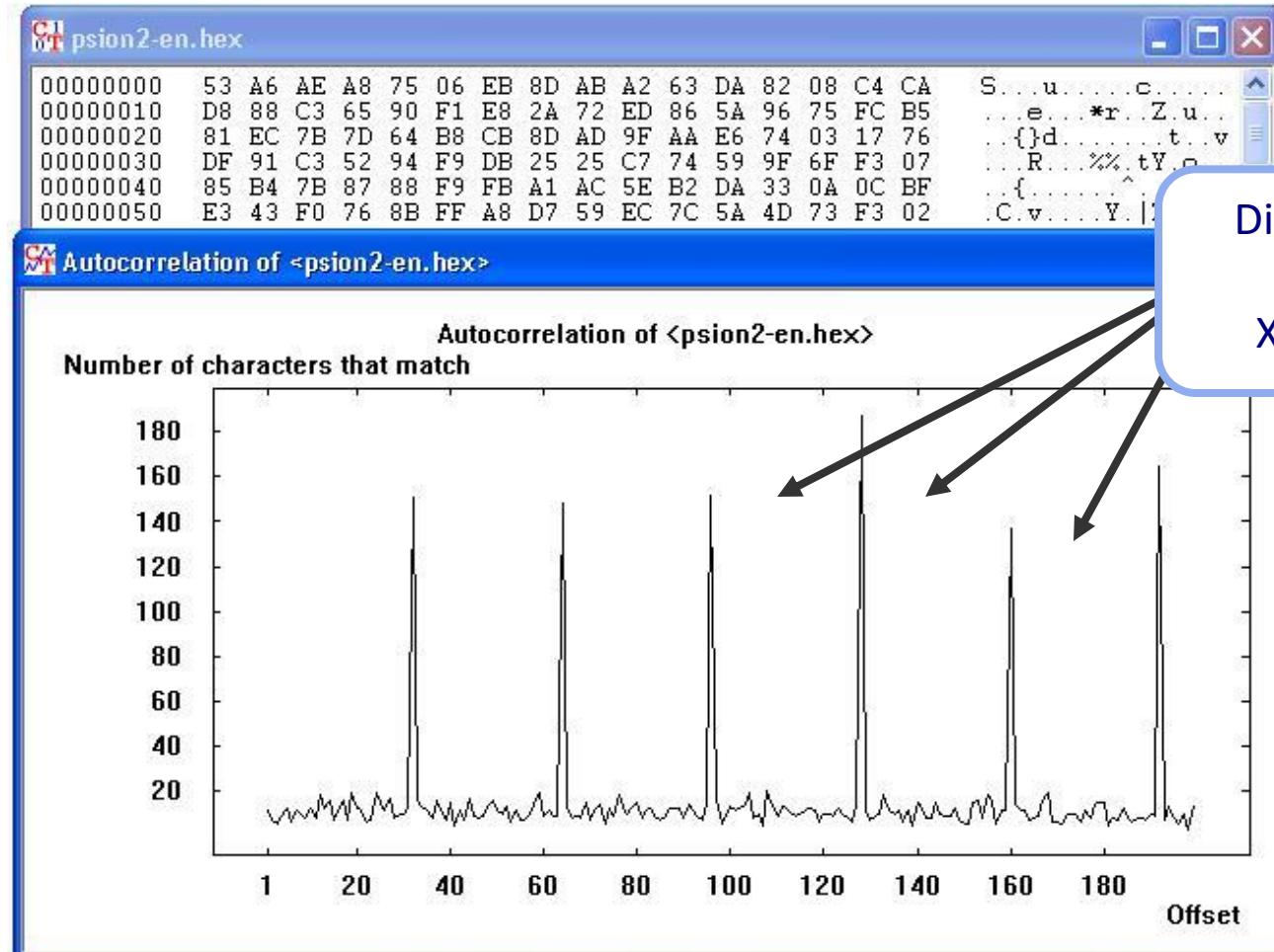


Compressibility:
not indicative. A larger value would be a clear indication of weak cryptography.



Examples (4)

PSION 5 PDA – determine auto-correlation



Distinctive comb pattern:
typical for Vigenère,
XOR, and byte addition

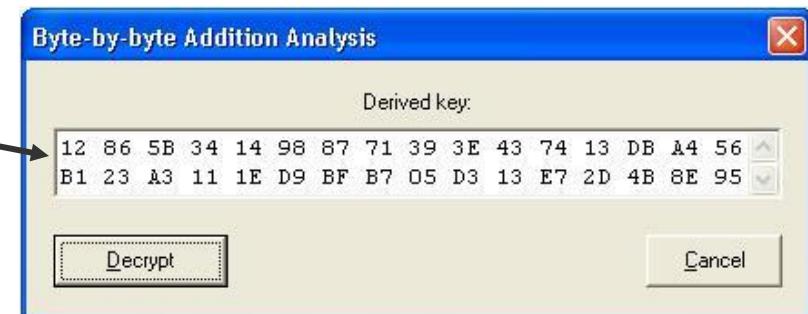
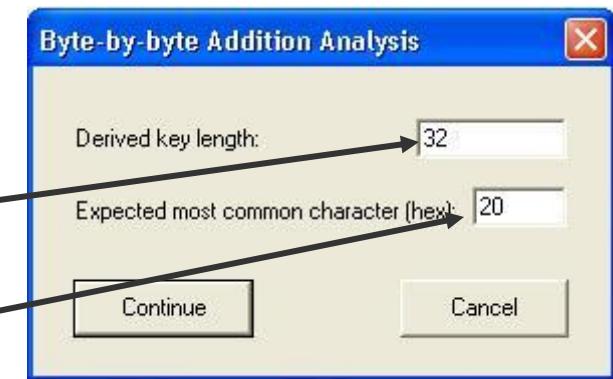
* The encrypted file is available in CrypTool (see CrypTool\examples\psion-en-enc.hex).

Examples (4)

PSION 5 PDA – automatic analysis

Automatic analysis using

- Vigenère: no success
- XOR: no success
- Byte addition
 - CrypTool calculates the key length using auto-correlation: 32 bytes
 - The user can choose which character is expected to occur most frequently: the empty space = 0x20 (ASCII code)
 - Analysis calculates the most likely key (based on assumptions regarding distribution)
 - Result: good, but not perfect



Examples (4)

PSION 5 PDA – results of automatic analysis

Results of automatic analysis under the assumption of “byte addition”

- Result is good, but not perfect: 25 out of 32 key bytes correct.
- The key length 32 was correctly determined.

	Automatic Addition Analysis of <psion2-en.hex>, key: <12 86 5B 34 14 98 87 71 39 3...																													
000000000	41	20	53	74	61	6E	64	1C	72	64	20	66	6F	2D	20	74														
000000010	27	65	20	54	72	18	29	73	6D	1A	73	73	69	2A	6E	20														
000000020	6F	66	20	49	50	20	44	1C	74	61	67	72	61	28	73	20														
000000030	2E	6E	20	41	76	20	1C	6E	20	F4	61	72	72	24	65	72														
000000040	73	2E	20	53	74	61	74	30	73	20	6F	66	20	2F	68	69														
000000050	32	20	4D	65	6D	26	E9	20	54	19	69	73	20	28	65	6D														
000000060	6F	20	64	65	73	63	72	24	62	65	73	20	61	29	20	65														
000000070	37	70	65	72	69	24	20	6E	74	12	6C	20	6D	20	74	68														
000000080	6F	64	20	66	6F	72	20	2F	68	65	20	65	6E	1E	61	70														
000000090	32	75	6C	61	74	20	2A	6E	20	20	66	20	49	0B	20	64														
0000000A0	61	74	61	67	72	61	6D	2E	20	69	6E	20	61	31	69	61														
0000000B0	2D	20	63	61	72	29	24	65	72	24	2E	20	54	23	69	73														
0000000C0	20	73	70	65	63	69	66	24	63	61	74	69	6F	29	20	69														
0000000D0	32	20	70	72	69	24	1C	72	69	1D	79	20	75	2E	65	66														
0000000E0	75	6C	20	69	6E	20	4D	20	74	72	6F	70	6F	27	69	74														
0000000F0	20	6E	20	41	72	1C	1C	20	4E	16	74	77	6F	2D	6B	73														
00000100	2E	20	54	68	69	73	20	24	73	20	61	6E	20	20	78	70														
00000110	24	72	69	6D	65	25	2F	61	6C	DD	20	6E	6F	2F	20	72														

- The password entered was not 32 bytes long.
→ PSION Word derives the actual key from the password.
- Manual post-processing produces the encrypted text (not shown).

Examples (4)

PSION 5 PDA – determining the remaining key bytes

First, copy the key to the clipboard during automatic analysis.

Then, in the automatic analysis hex dump:

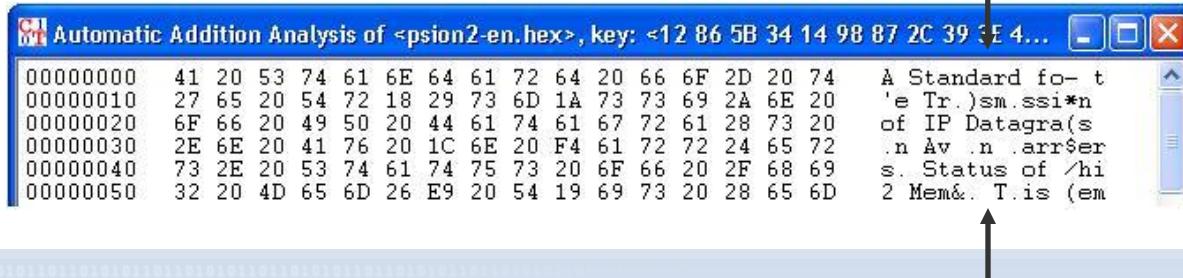
- Determine incorrect byte positions, e.g. 0x1C at position 8
- Guess and write down corresponding correct bytes: “a” = 0x61

Next, in the encrypted initial file hex dump:

- Determine initial bytes from the calculated byte positions: 0x8D
- Calculate correct key bytes with CALC.EXE: $0x8D - 0x61 = 0x2C$

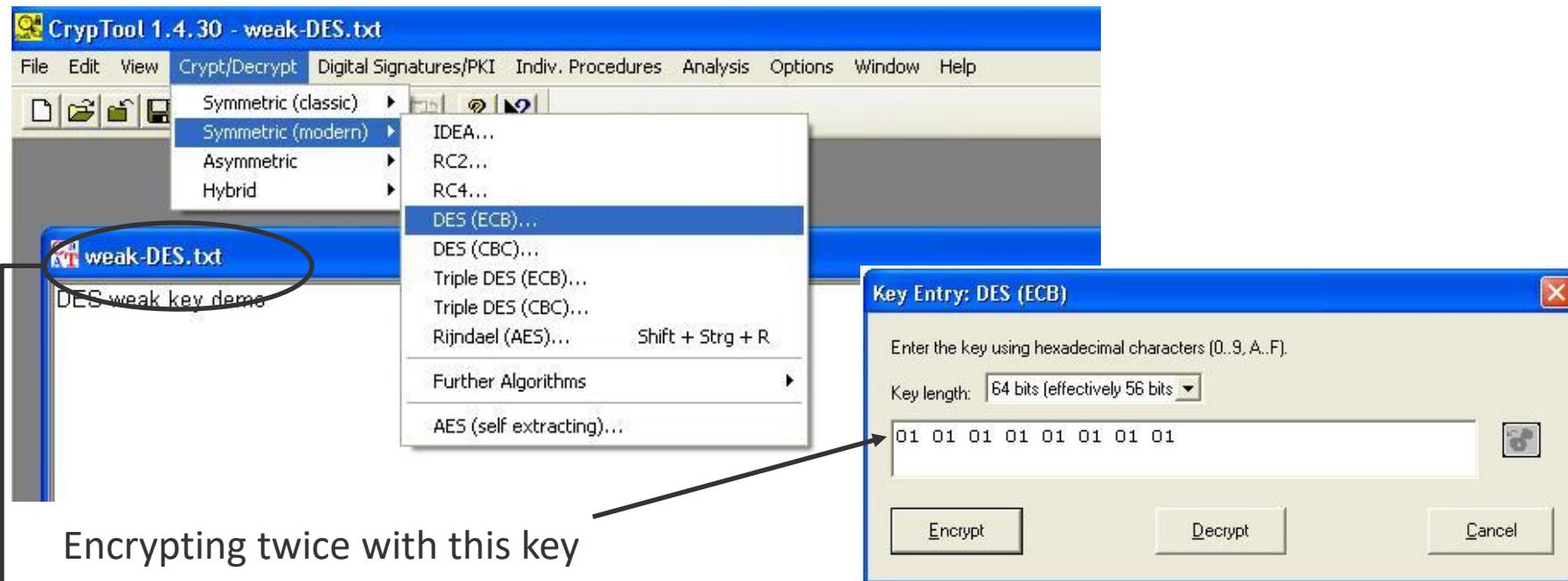
Finally, get the key from the clipboard:

- Correct 12865B34149887**2C**393E437413DBA456B123A3111ED9BFB705D313E72D4B8E95
- Decrypt encrypted initial document using byte addition
- Bytes at position 3, 3+32, 3+2*32, etc. are now correct



Examples (5)

Weak DES key



Encrypting twice with this key
returns the plaintext.



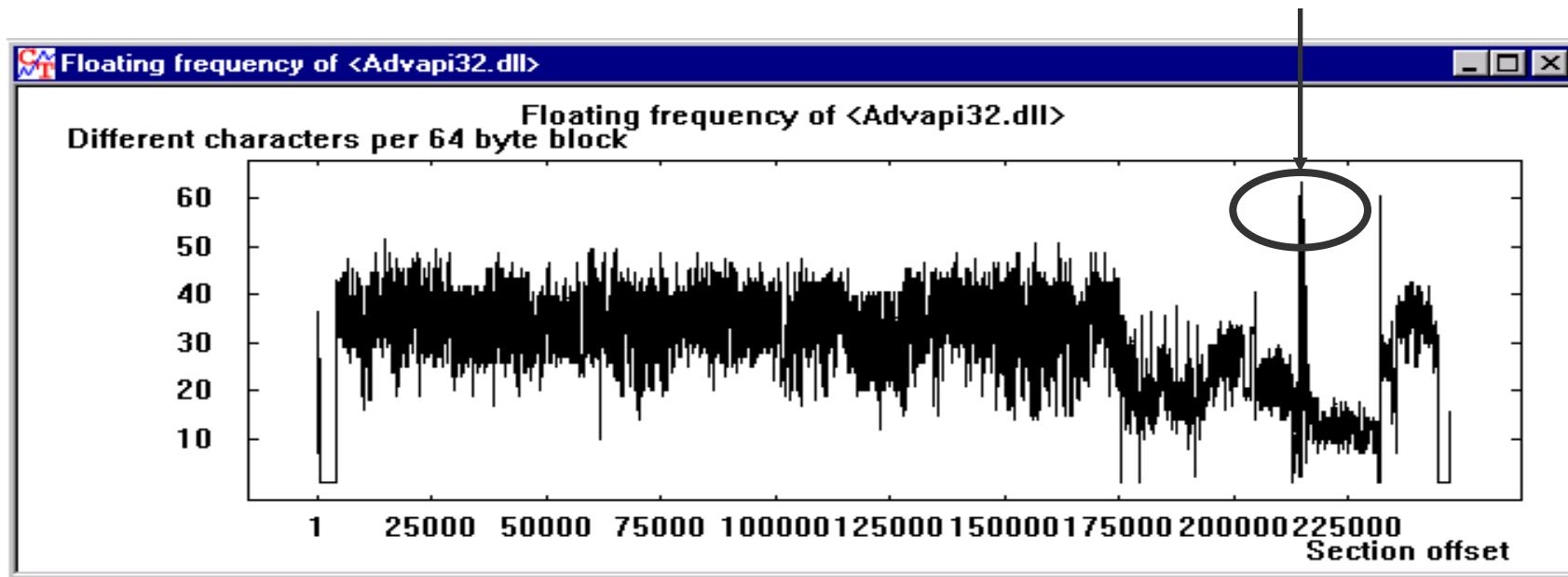
Examples (6)

Locate key material

The function “Floating frequency” is suitable for locating key material and encrypted areas in files.

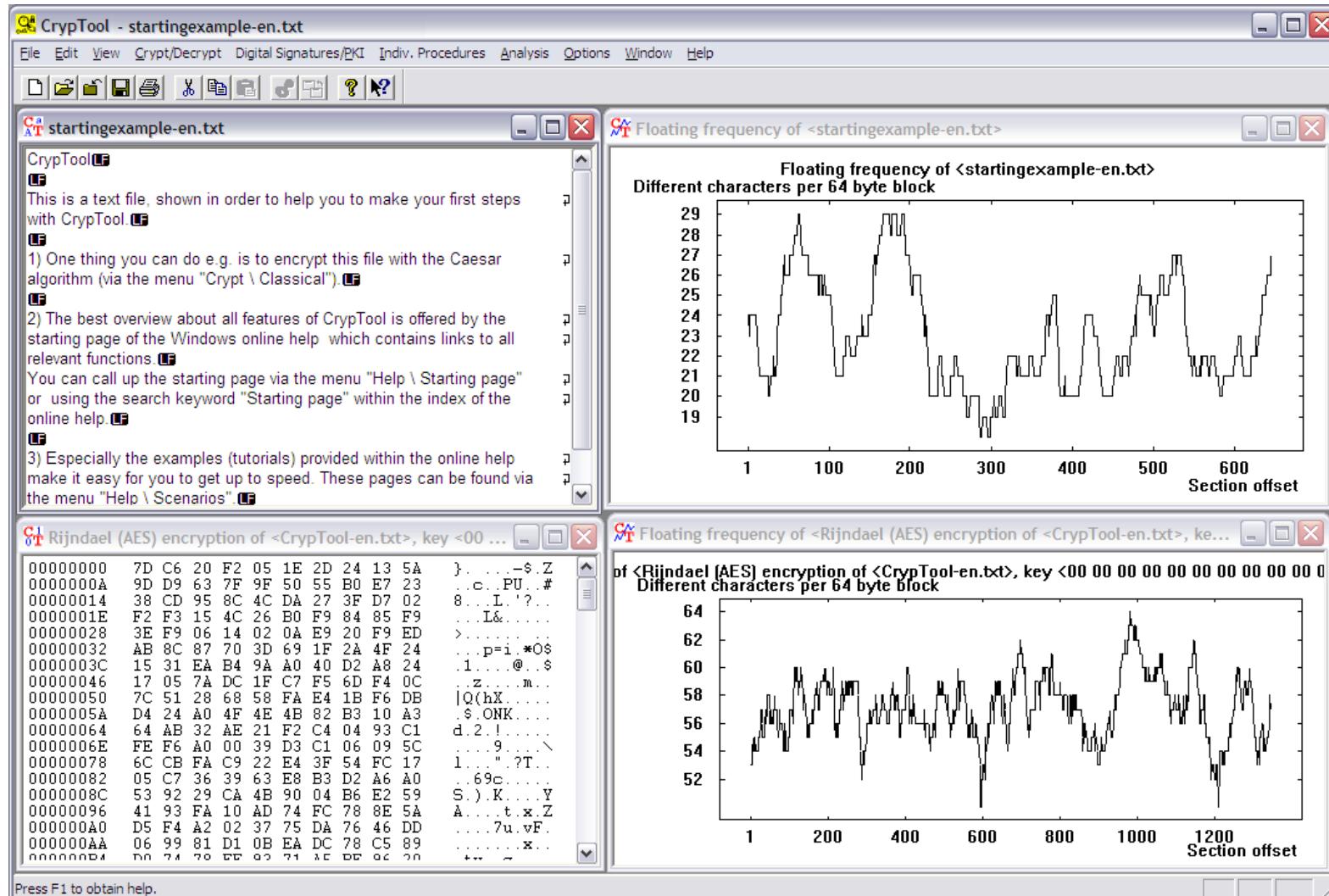
Background

- Key data is “more random” than text or program code
- Can be recognized as peaks in the “floating frequency”
- Example: the “NSA key” in advapi32.dll (Windows NT)



Examples (6)

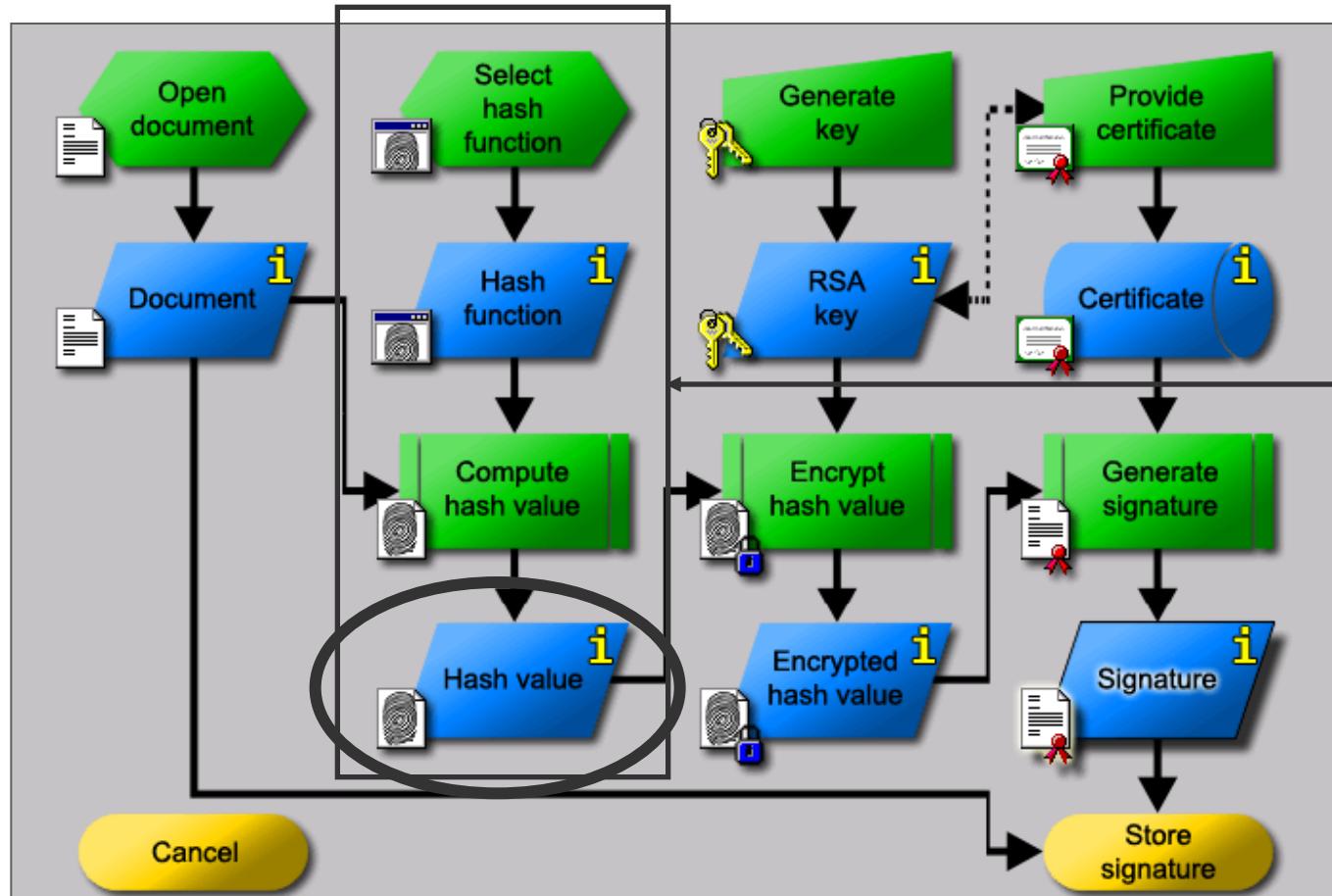
Floating frequency comparison



Press F1 to obtain help.

Examples (7)

Attack on digital signatures



Attack

Find two messages with the same hash value!

Menu: "Analysis" \ "Hash" \ "Attack on the Hash Value of the Digital Signature"

Examples (7)

Attack on digital signature – idea (I)

Attack on the digital signature of an ASCII text by means of a hash collision search.

Idea:

- ASCII texts can be modified by changing/inserting **non-printable** characters without changing the visible content
- Modify two texts in parallel until a hash collision is found
- Exploit the birthday paradox (birthday attack)
- Generic attack applicable to all hash functions
- Can parallelized across many machines (not implemented in CrypTool)
- Implemented in CrypTool as part of the bachelor thesis

“Methods and Tools for Attacks on Digital Signatures” (German), 2003.

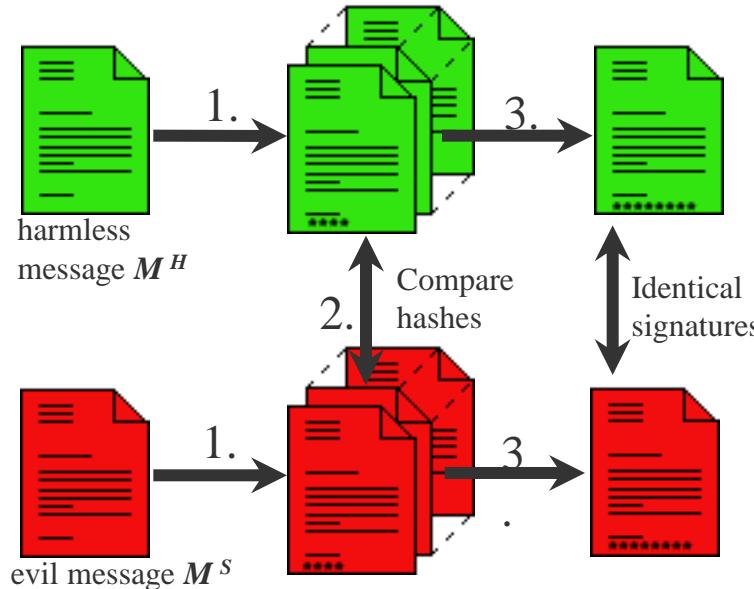
Concepts :

- Mappings
- Modified Floyd algorithm (constant memory consumption)



Examples (7)

Attack on digital signature – idea (II)



1. **Modification:** starting from a message M create N different messages M_1, \dots, M_N with the same “content” as M .
2. **Search:** find modified messages M_i^H and M_j^S with the same hash value.
3. **Attack:** the signatures of those two documents M_i^H and M_j^S are the same.

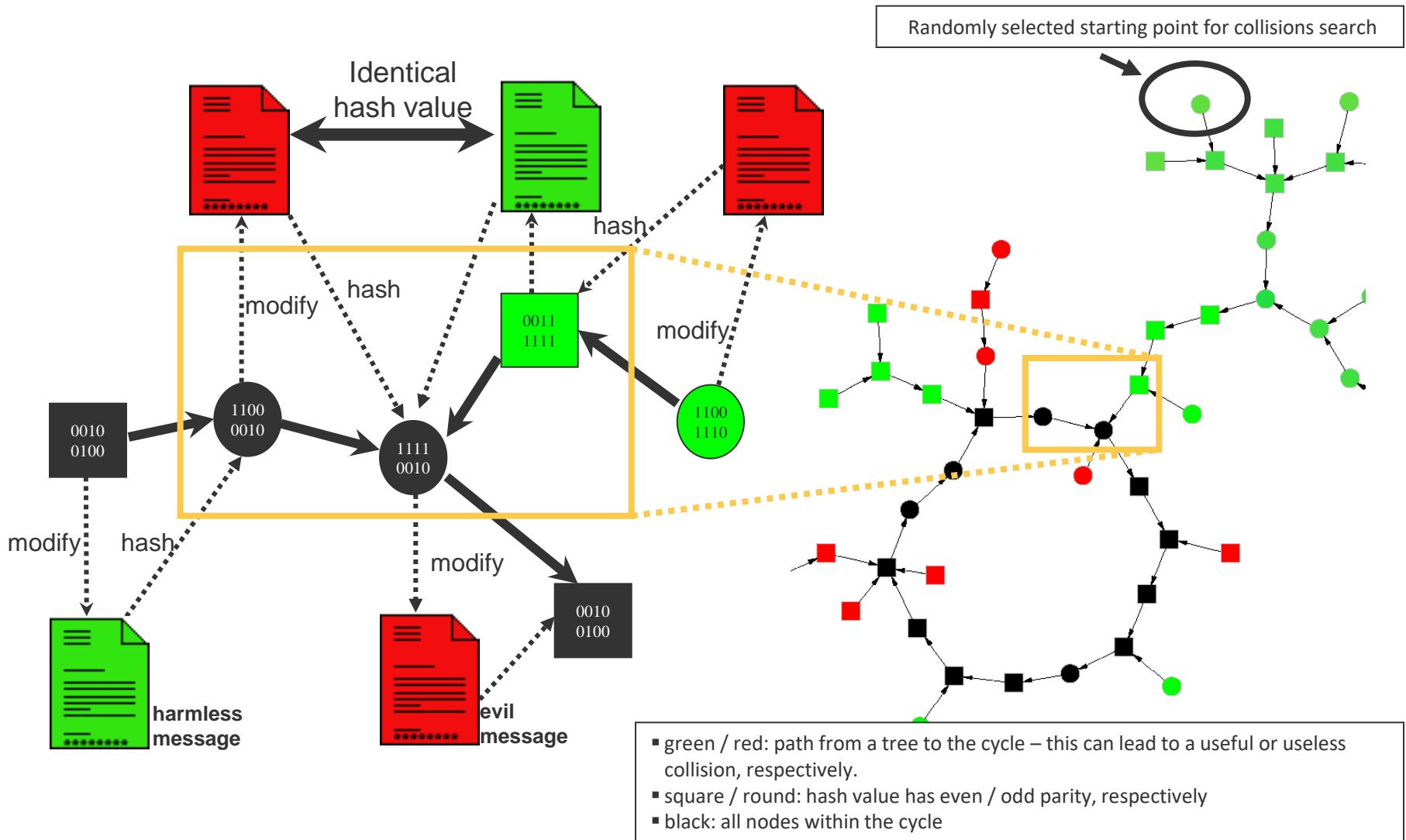
We know from the birthday paradox that for hash values of bit length n :

- search collision between M^H and M_1^S, \dots, M_N^S :
 $N \approx 2^n$
- search collision between M_1^H, \dots, M_N^H and M_1^S, \dots, M_N^S :
 $N \approx 2^{n/2}$

Estimated number of generated messages in order to find a hash collision.

Locate Hash Collisions (1)

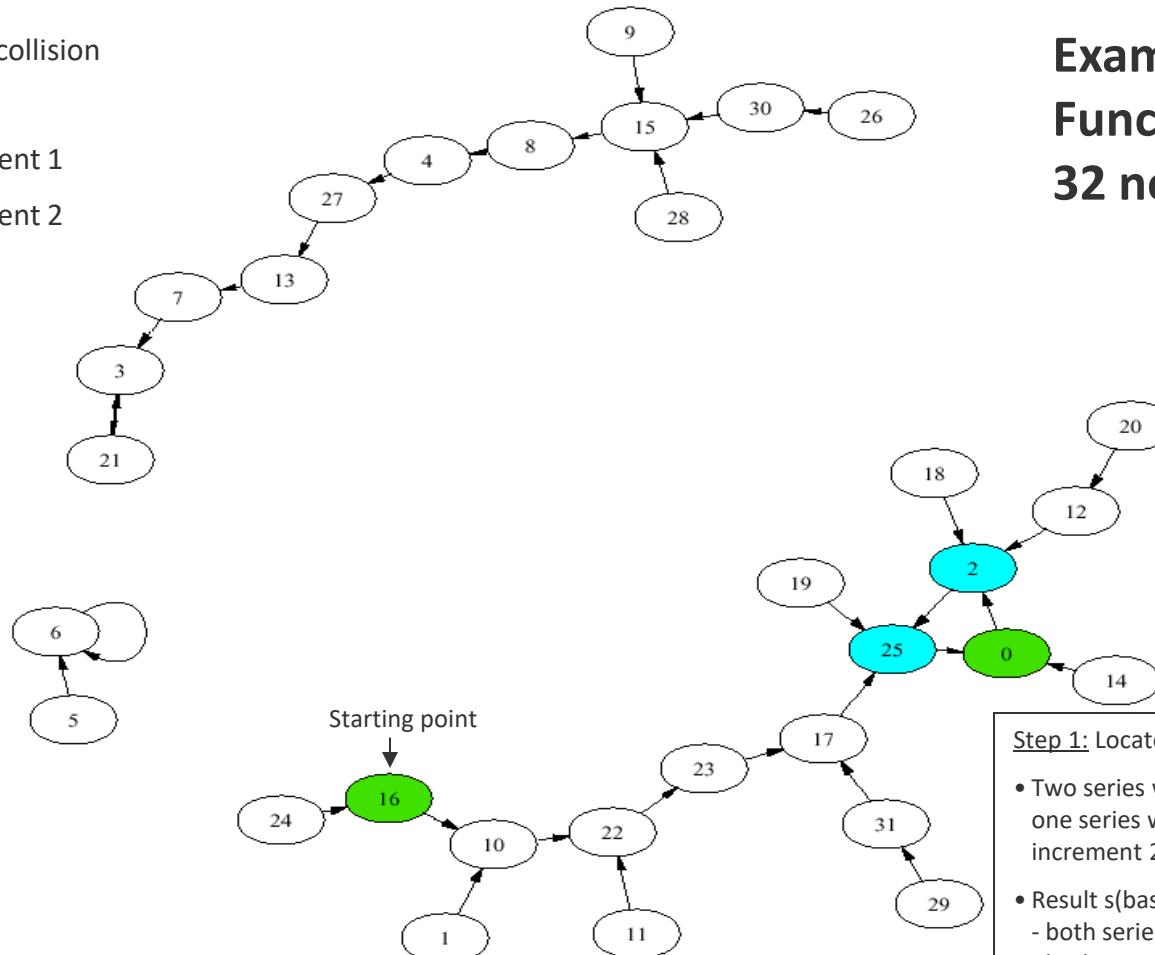
Mapping via text modifications



Locate Hash Collisions (2)

Floyd Algorithm: Meet within the cycle

-  start / collision
-  cycle
-  increment 1
-  increment 2



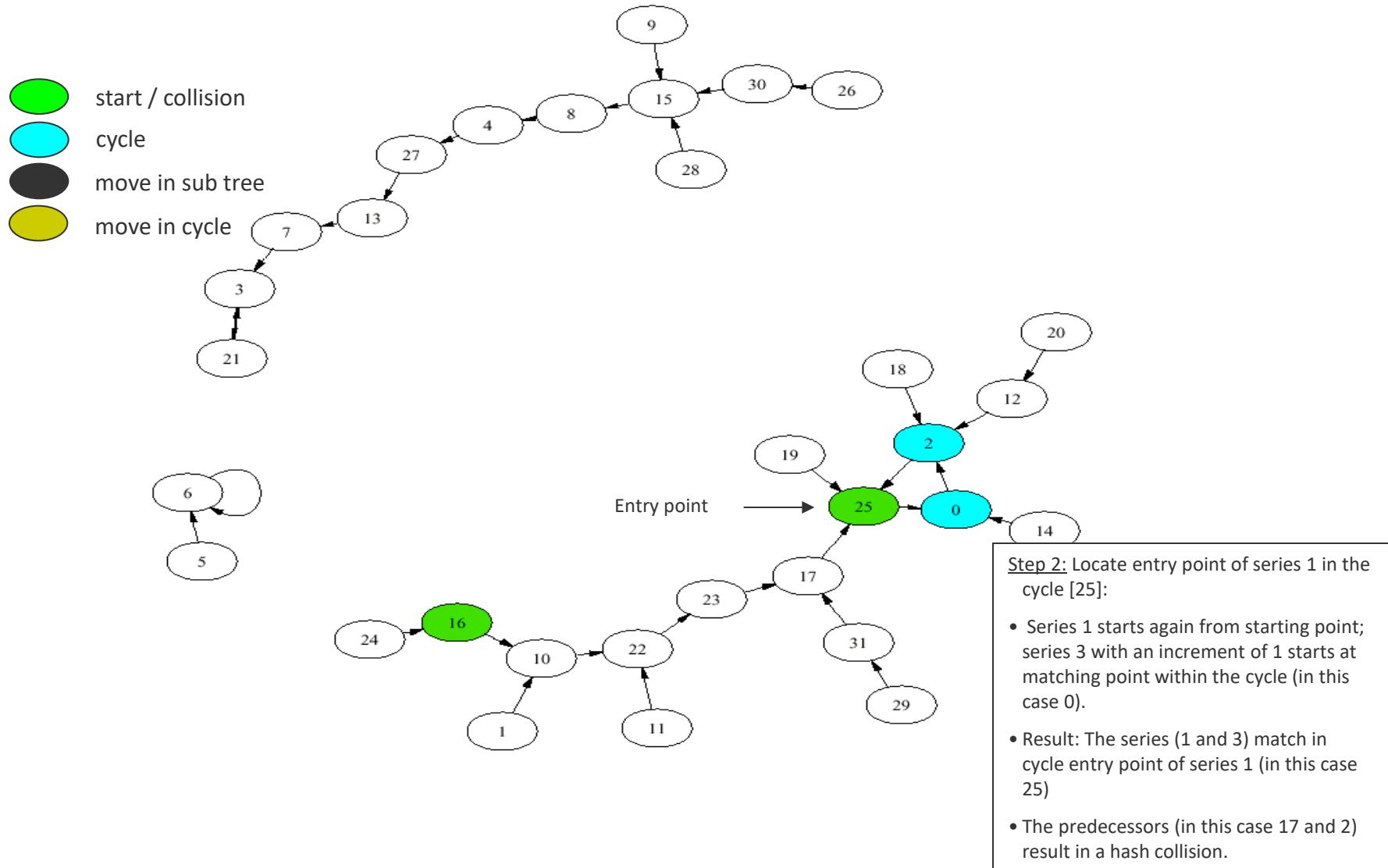
Example:
Function graph with 32 nodes

Step 1: Locate matching point within cycle:

- Two series with identical starting point [16]: one series with increment 1, the other with increment 2.
- Result s(based on graph theory):
 - both series always end up in a cycle.
 - both series match in a node within the cycle (in this case 0).

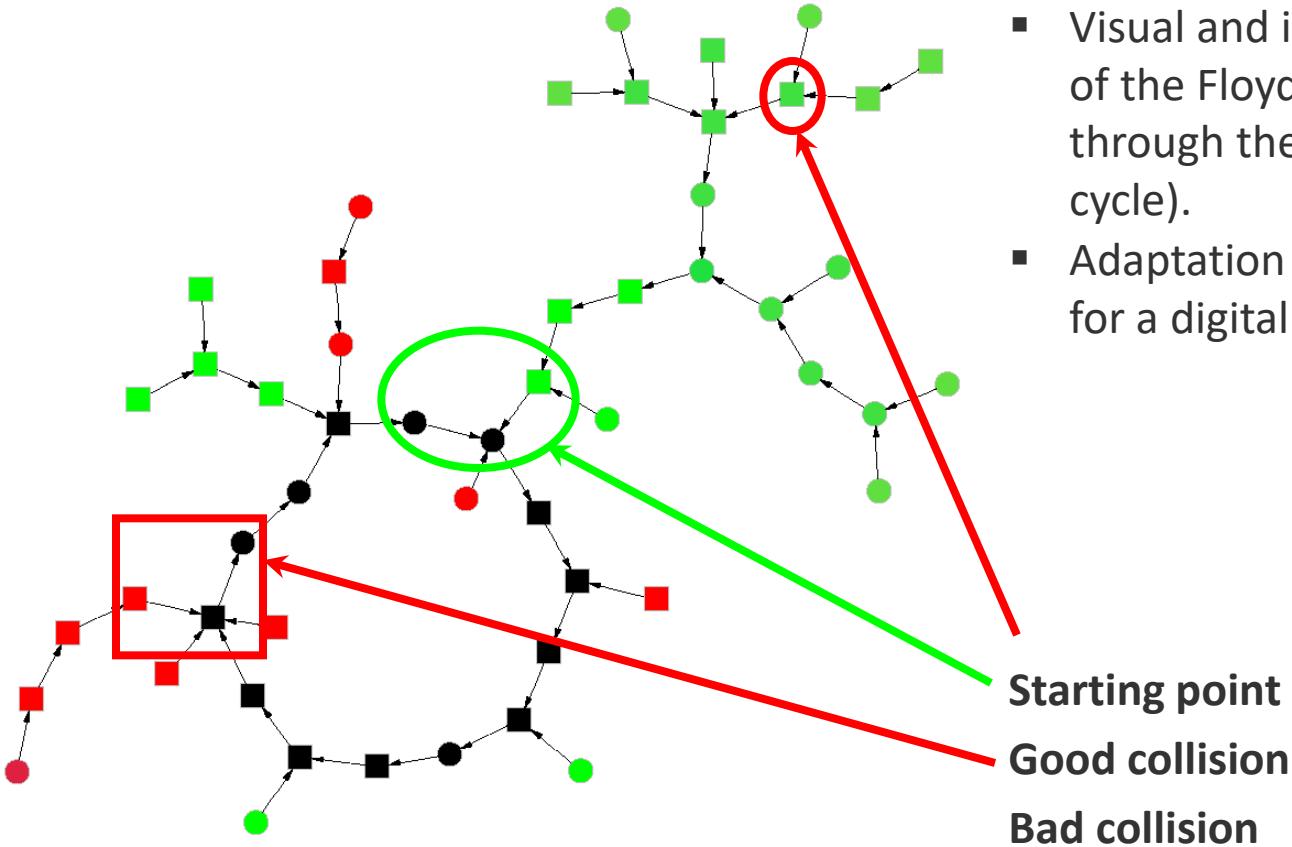
Locate Hash Collisions (3)

Step into cycle (extension of Floyd): Find entry point



Locate Hash Collisions (4)

Birthday paradox attack on digital signature



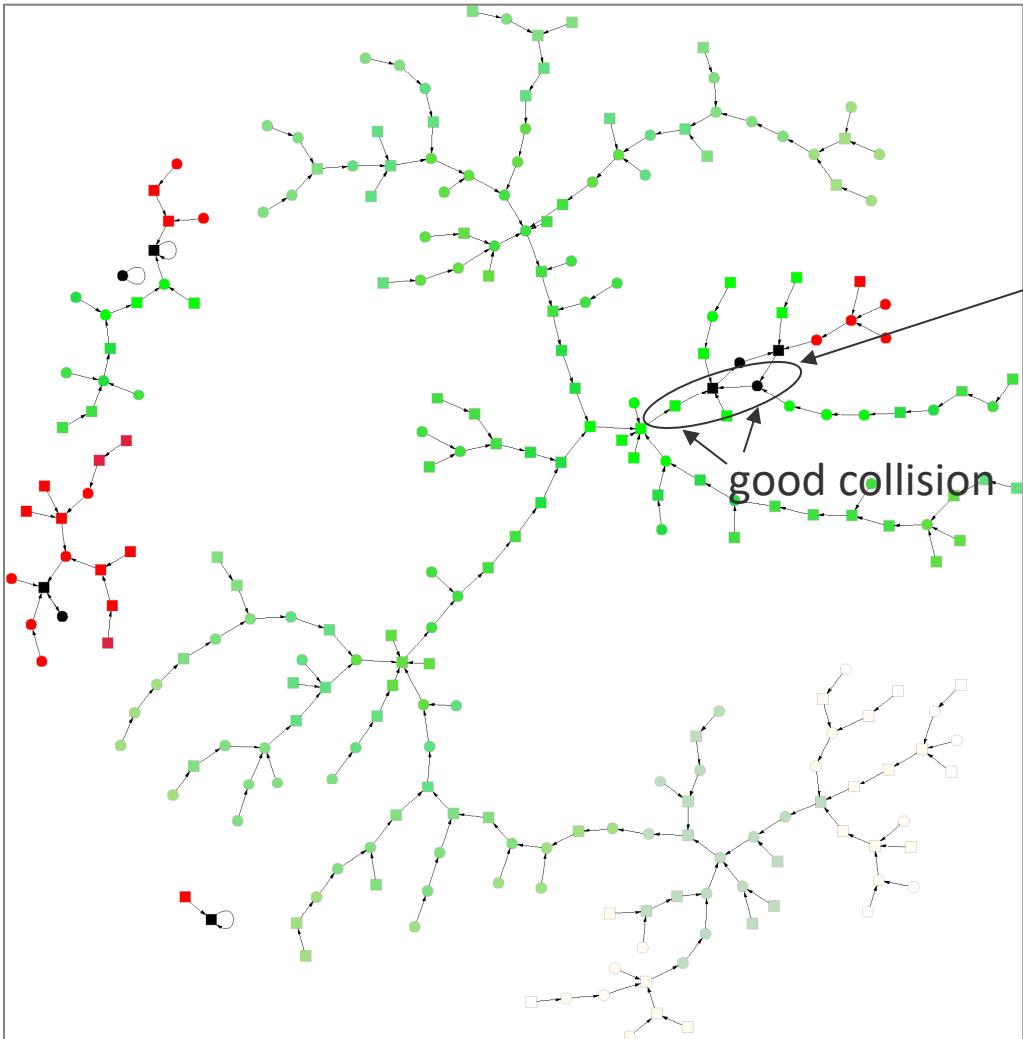
Examination of Floyd algorithm

- Visual and interactive presentation of the Floyd algorithm (“Moving through the mapping” into a cycle).
- Adaptation of the Floyd algorithm for a digital signature attack.

*The Floyd algorithm is implemented in CrypTool, but the visualization of the algorithm has not yet been implemented.

Examples (7)

Attack on digital signature

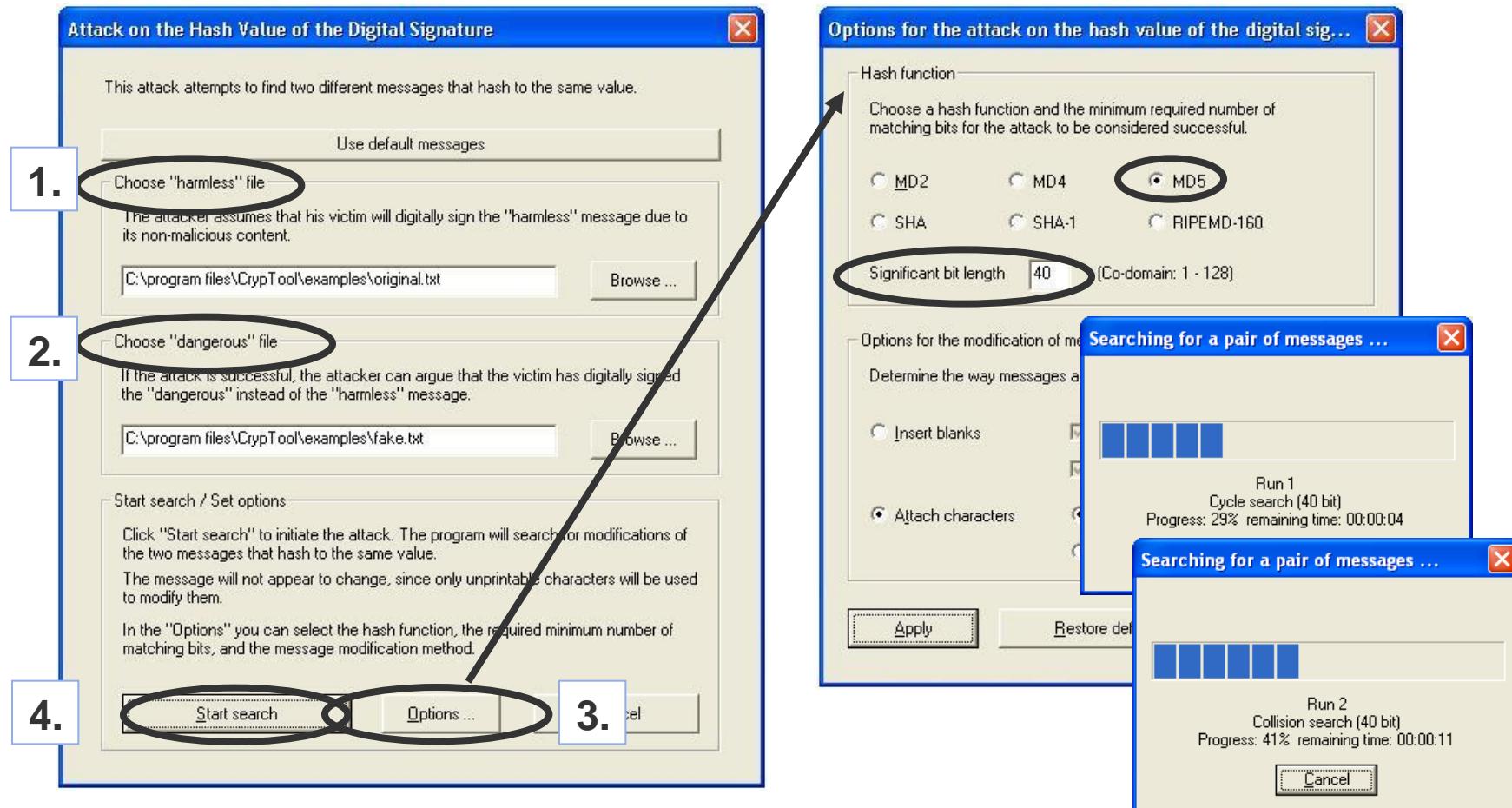


An example of a “good” mapping (nearly all nodes are green). In this graph almost all nodes belong to a big tree, which leads into the cycle with an even hash value and where the entry point predecessor within the cycle is odd. That means that the attacker finds a useful collision for nearly all starting points.



Examples (7)

Attack on digital signature: attack



Menu: “Analysis” \ “Hash” \ “Attack on the Hash Value of the Digital Signature”

Examples (7)

Attack on digital signature: results

The screenshot shows two windows from the CrypTool interface. The top window is titled "Harmless message: MD5, <A9 76 34 AB>" and contains the message: "Dear Mr Shopaholic, please order a typewriter. Regards Honest John". The bottom window is titled "Dangerous message: MD5, <A9 76 34 AB>" and contains the message: "Dear Mr Shopaholic, please order a Porsche and a prepaid insurance scheme for Mr. Dodgy. Regards Honest John". Both messages have their last few characters circled in yellow. Below each message is its corresponding MD5 hash value. A large arrow points from the bottom window's hash to the text "The first 32 bits of the hash values are identical.".

Message	MD5 Hash
Harmless message: MD5, <A9 76 34 AB>	MD5: 4F 47 DF 1F D2 DE CC BE 4B 52 86 29 F7 A8 1A 9A
Dangerous message: MD5, <A9 76 34 AB>	MD5: 4F 47 DF 1F 30 38 BB 6C AB 31 B7 52 91 DC D2 70

The first 32 bits of the hash values are identical.

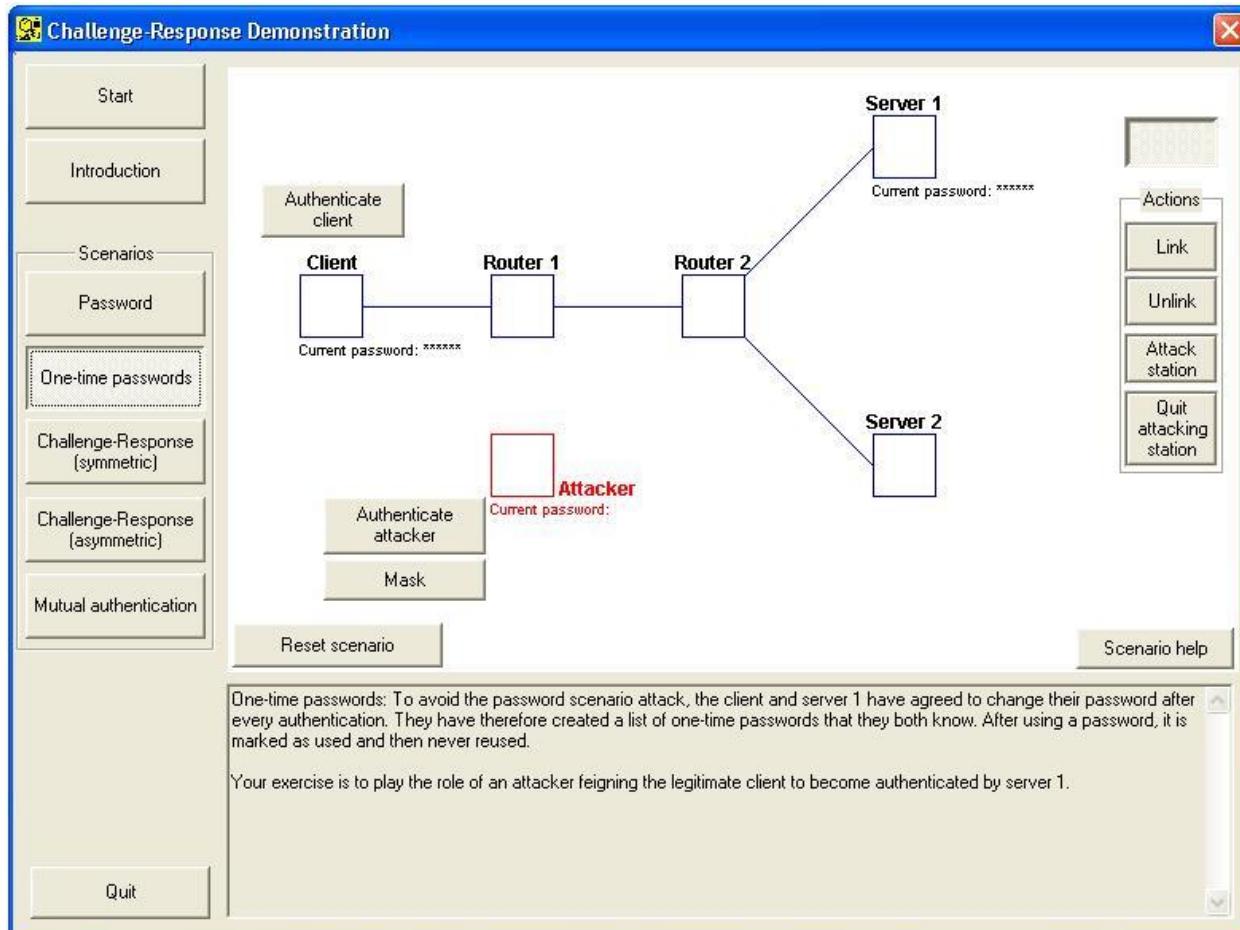
Experimental results

- A 72-bit *partial collision* (i.e., the first 72 hash value bits are identical) was found in a couple of days using a single PC.
- Today, signatures with hash values of 128 bits or less are vulnerable to a massive parallel search!
- It is therefore recommended to use hash values with a length of at least 160 bits.

In addition to the interactive tool, CrypTool also includes a command-line feature to execute and log the results for entire sets of parameter configurations.

Examples (8)

Authentication in a client-server environment

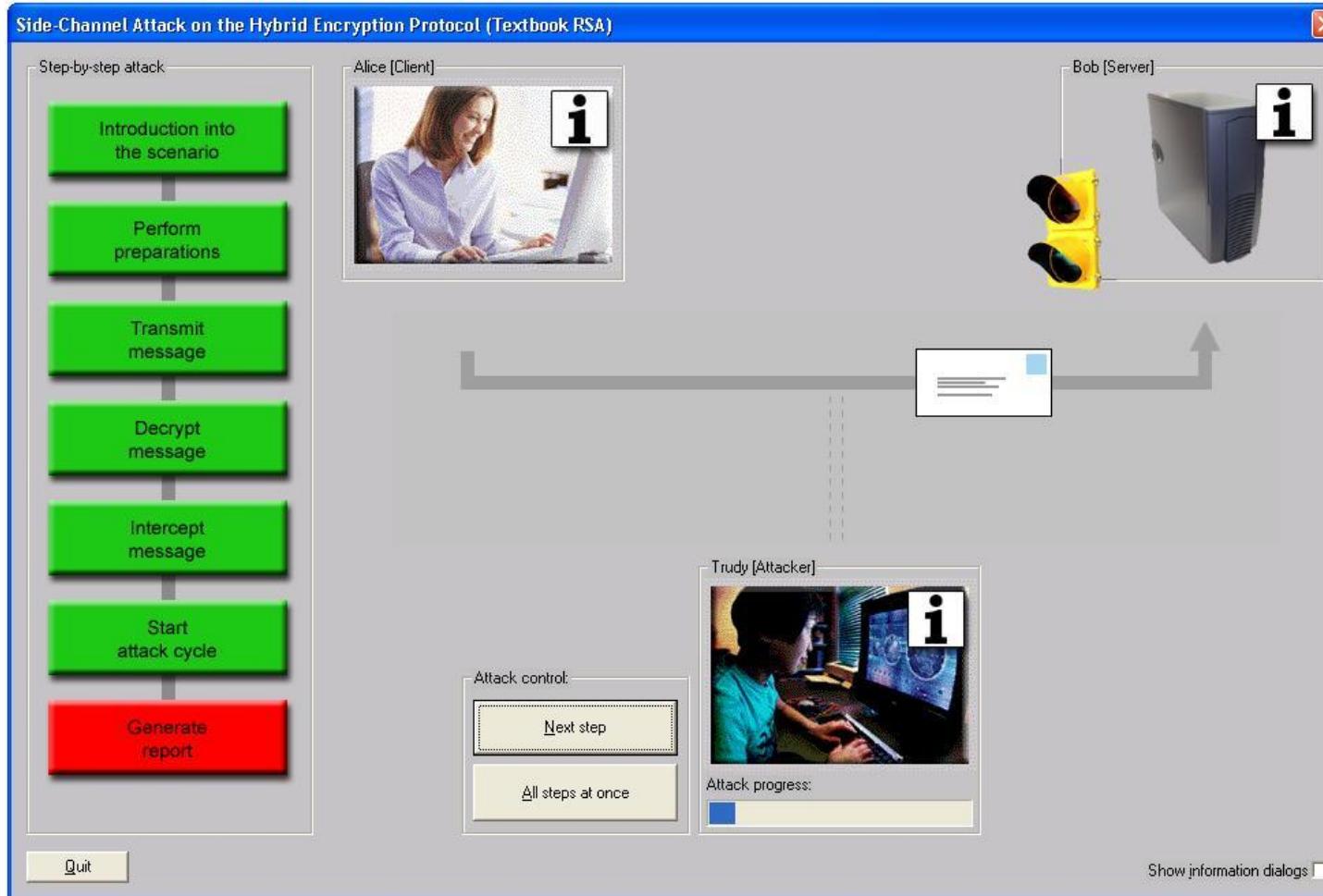


- Interactive demo for different authentication methods.
- Specifies vulnerabilities that an attacker could take advantage of.
- Allows the user to play the role of an attacker.
- **Learning outcome:** Only mutual authentication is secure.

Menu: “Indiv. Procedures” \ “Protocols” \ “Network Authentication”

Examples (9)

Demonstration of a side-channel attack (on a hybrid encryption protocol)



Menu: "Analysis" \ "Asymmetric Encryption" \ "Side-Channel Attack on Textbook RSA"

Examples (9)

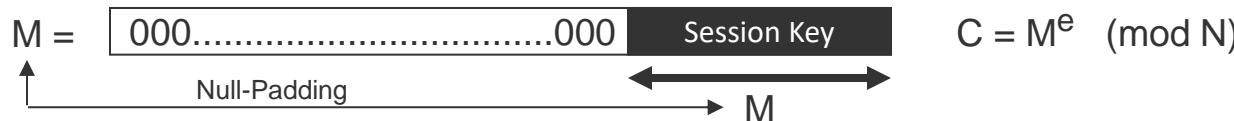
Concept of this side channel attack

Ulrich Kuehn: "Side-channel attacks on textbook RSA and ElGamal encryption", 2003

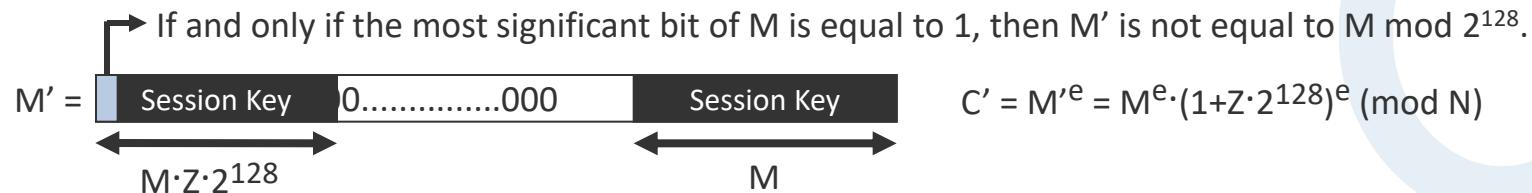
Prerequisites [CCA (Chosen-ciphertext attack) against deciphering oracle]

- RSA encryption: $C = M^e \pmod{N}$ and decryption: $M = C^d \pmod{N}$.
- 128-bit session keys (in M) are encoded according to textbook RSA (null padding).
- The server knows the secret key d and
 - uses after decryption only the least significant 128 bits without validating the null-padded bits, meaning that the server does not recognize if there is something there other than zero.
 - An error message is prompted if the encryption attempt results in an “incorrect” session key (decrypted text cannot be interpreted by the server). In all other cases there will be no message.

Idea for attack: Approximation of Z in 129 bits from the equation $N = M * Z$ per $M = \lfloor |N/Z| \rfloor$

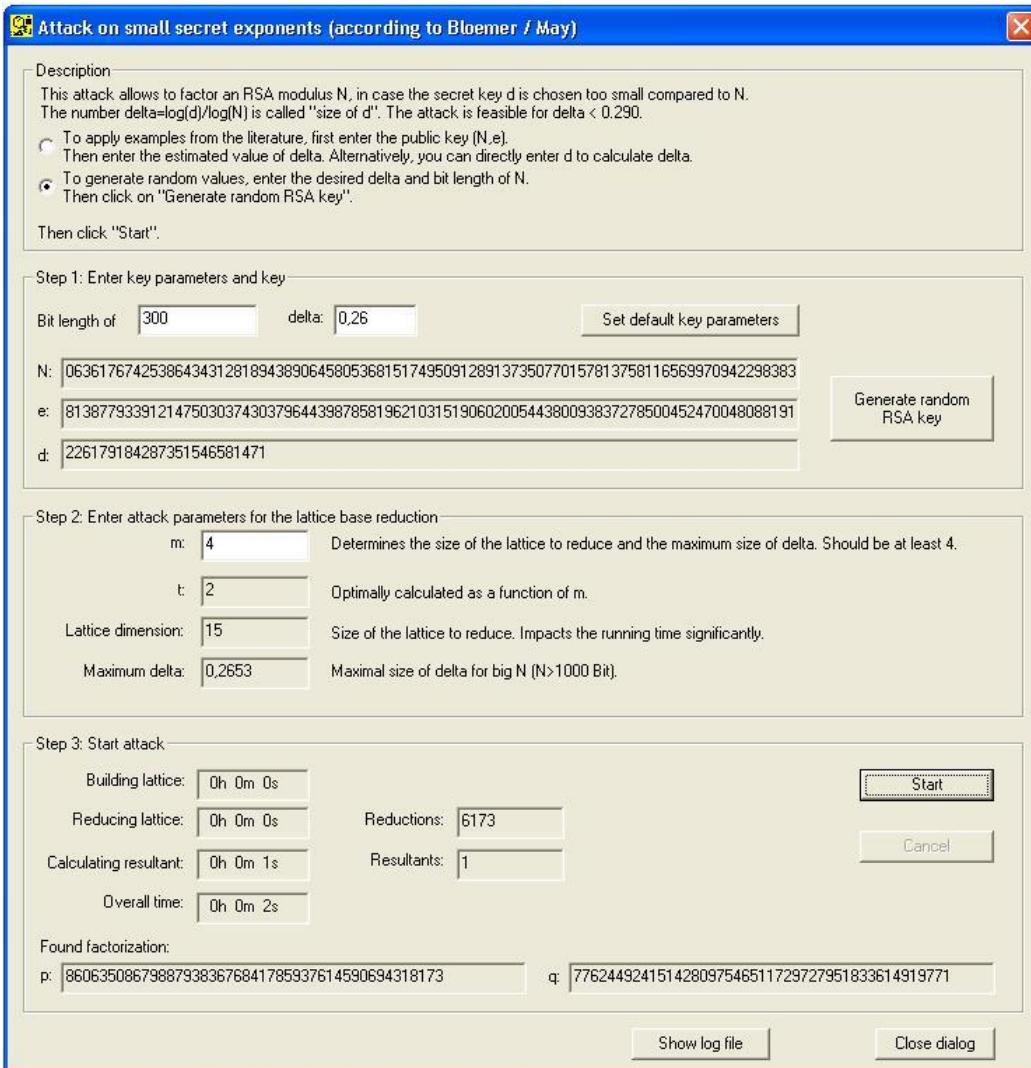


All bit positions for Z are successively calculated: for each step the attacker gets one additional bit. He or she then modifies C to C' (see below). If a bit overflow occurs while calculating M' on the server (recipient), the server sends an error message. Based on this information, the attacker can determine a single bit of Z .



Examples (10)

Mathematics: Attacks on RSA using lattice reduction



- Demonstrates that the parameters of RSA should be chosen in a way to withstand the lattice reduction attacks described in current literature.

- **3 variants which are *not* resistant:**

1. The secret exponent d is too small in comparison to N .
2. One of the factors of N is partially known.
3. A part of the plaintext is known.

- These assumptions are realistic.

Menu: “Analysis” \
“Asymmetric Encryption” \
“Lattice Based Attacks on RSA” \ ...

Examples (11)

Random analysis with 3-D visualization

3-D visualization for random analysis

Example 1

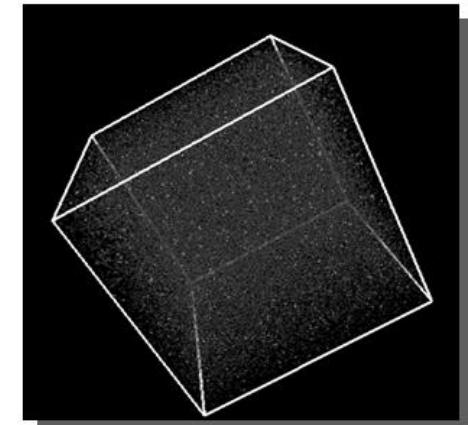
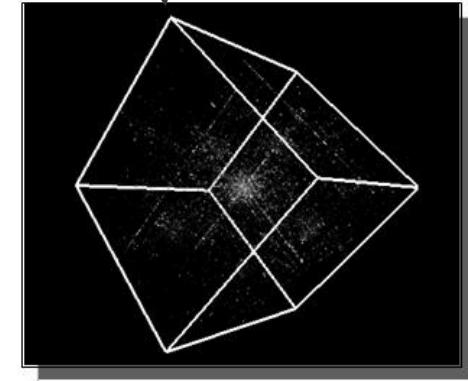
- Open an arbitrary file (e.g. report in Word or PowerPoint presentation)
- It is recommended to select a file with at least 100 kB
- 3-D analysis
- Result: **structures are easily recognizable**

Example 2

- Generation of random numbers via menu:
“Indiv. Procedures” \ “Tools” \ “Generate Random Numbers”
- It is recommended to generate at least 100,000 random bytes
- 3-D analysis
- Result: **uniform distribution (no structures are recognizable)**

Menu: “Analysis” \ “Analyze Randomness” \ “3-D Visualization”

You can turn the cube with the mouse to the perspective you wish.

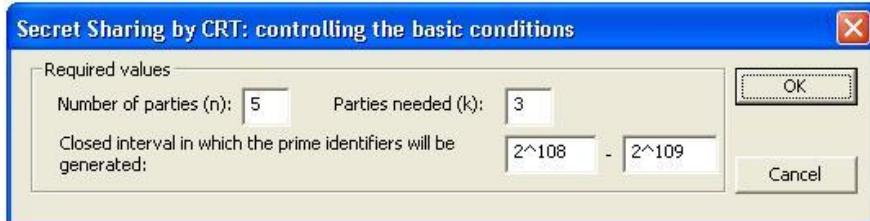


Examples (12)

Secret sharing with CRT – implementation of the Chinese remainder theorem (CRT)

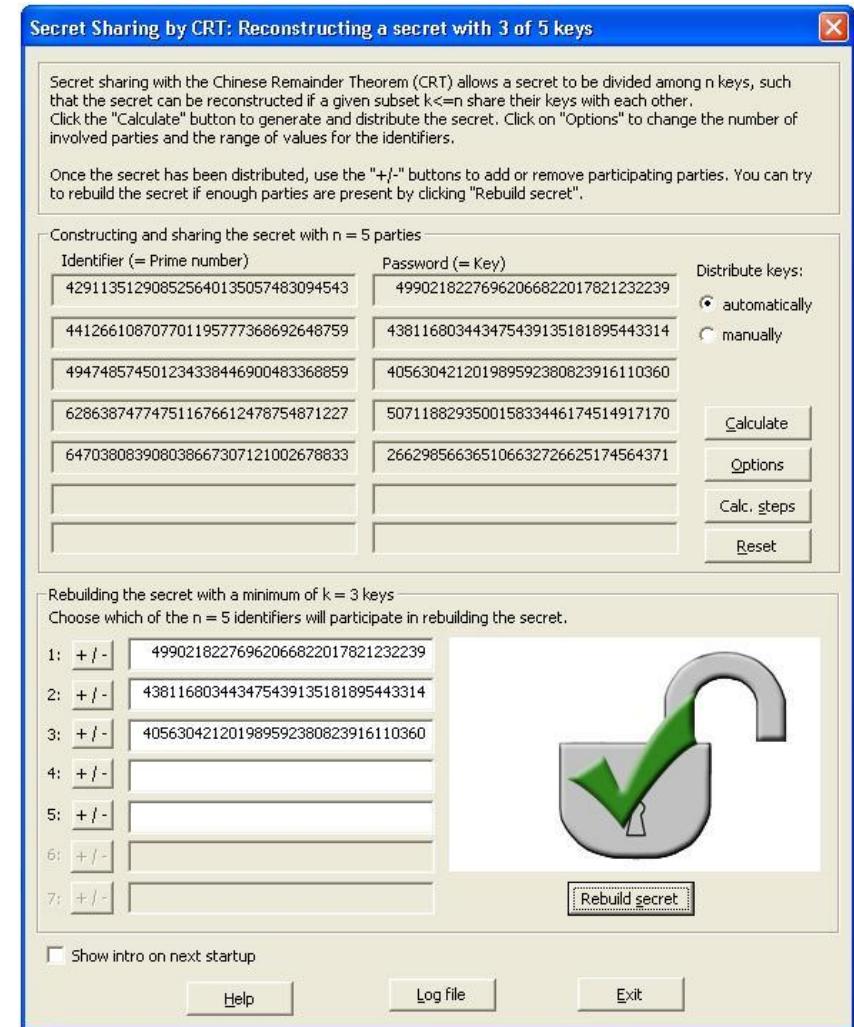
Secret sharing example (1)

- **Problem**
 - 5 people each receive a single key
 - To gain access, at least 3 of the 5 people must be present
- **“Options”** allows the user to configure additional settings.



- **“Calc. steps”** shows all of the steps in key generation.

Menu: “Indiv. Procedures” \
“Chinese Remainder Theorem Applications” \
“Secret Sharing by CRT”



Examples (12)

Shamir secret sharing

Secret sharing example (2)

■ Problem

- A secret value is to be divided among n people.
- t out of n people are required to restore the secret value K.
- (t, n) threshold scheme

■ Perform it in the dialog:

1. Enter the secret K, number of persons n and threshold t
2. Generate polynomial
3. Select parameters
4. Click “Reconstruction” to restore the secret.

Menu: “Indiv. Procedures” \
“Secret Sharing Demonstration (Shamir)”

Secret Sharing: Initializing the threshold scheme

By means of a (t, n) Shamir scheme a secret S can be distributed among n persons. Afterwards, t persons (t <= n) will be able to reconstruct the original secret by combining their individual secrets (shares). To set up such a scheme, a polynomial f(x) of degree at most t-1 [with t-1 coefficients a[i] chosen at random] and a random prime p must be generated. Each participant receives a randomly chosen public value x and his or her share, the corresponding secret value y=f(x). For further details please check the CrypTool online help by pressing F1.

Choose your secret and the parameters (whole numbers) to set up a scheme

Secret S with S >= 0	1244	Set default parameters
Number of participants n, with n > 0	8	Options
Threshold (minimum) t with t > 0	3	

Parameters concerning the polynomial f(x) of degree t-1
All computations take place in the discrete space GF(p)

Polynomial f(x): 1244+42x+571x^2

Prime p: 1627

Participants' values, calculated from chosen parameters:

Participant	Public value x	Share [secret value f(x)]
<input checked="" type="checkbox"/> participant 1	1454	1564
<input type="checkbox"/> participant 2	469	1257
<input checked="" type="checkbox"/> participant 3	1273	995
<input type="checkbox"/> participant 4	1082	673
<input checked="" type="checkbox"/> participant 5	90	1309
<input type="checkbox"/> participant 6	73	1425
<input type="checkbox"/> participant 7	931	1445
<input type="checkbox"/> participant 8	60	1209

Please check the appropriate boxes to select the participants who will attempt to reconstruct the secret.

Show information dialog at startup

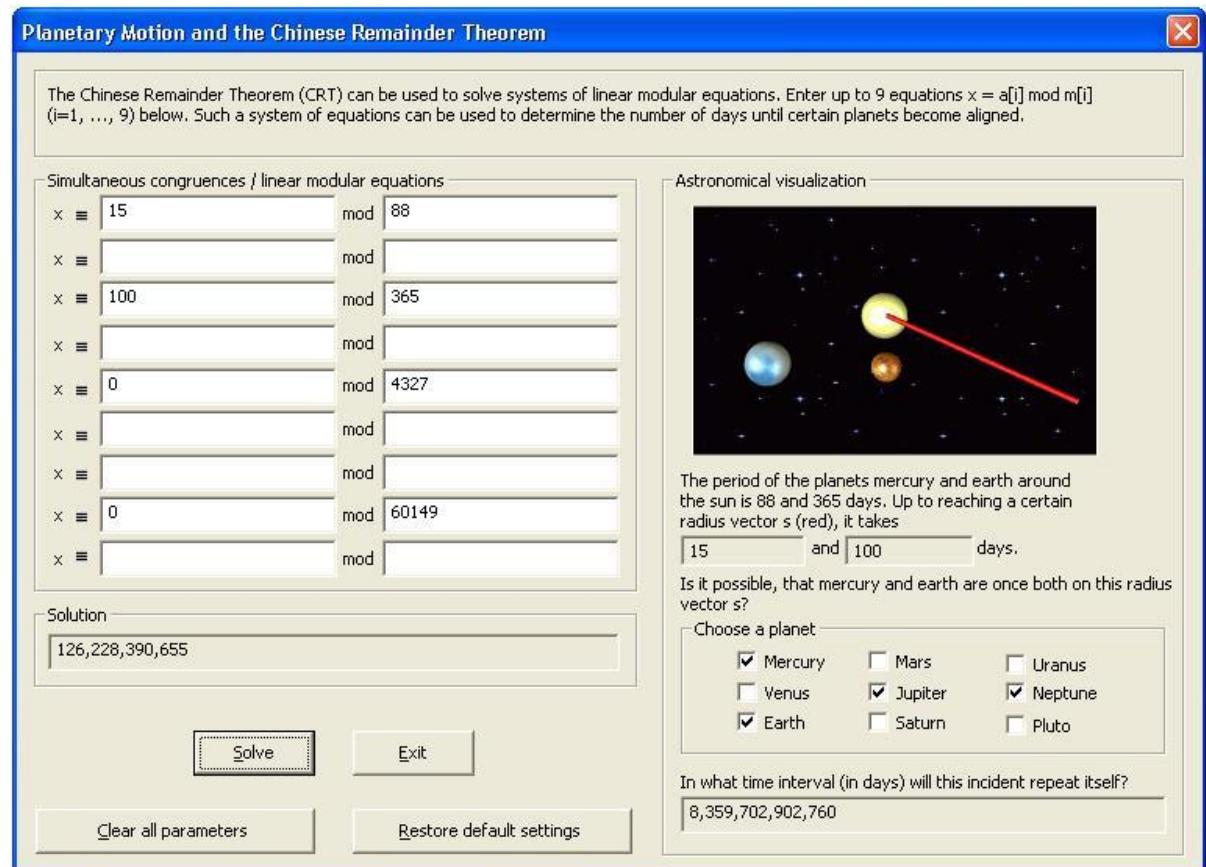
Cancel **Reconstruction**

Examples (13)

Implementation of CRT to solve linear modular equation systems

Astronomical scenario

- How long would it take for a given number of planets (with different rotation times) to become aligned?
- The result is a linear modular equation system that can be solved with the Chinese remainder theorem (CRT).
- In this demo you can enter up to 9 equations and compute a solution using the CRT.



Menu: “Indiv. Procedures” \ “Chinese Remainder Theorem Applications” \ “Astronomy and Planetary Motion”

Examples (14)

Visualization of symmetric encryption methods using ANIMAL (1)

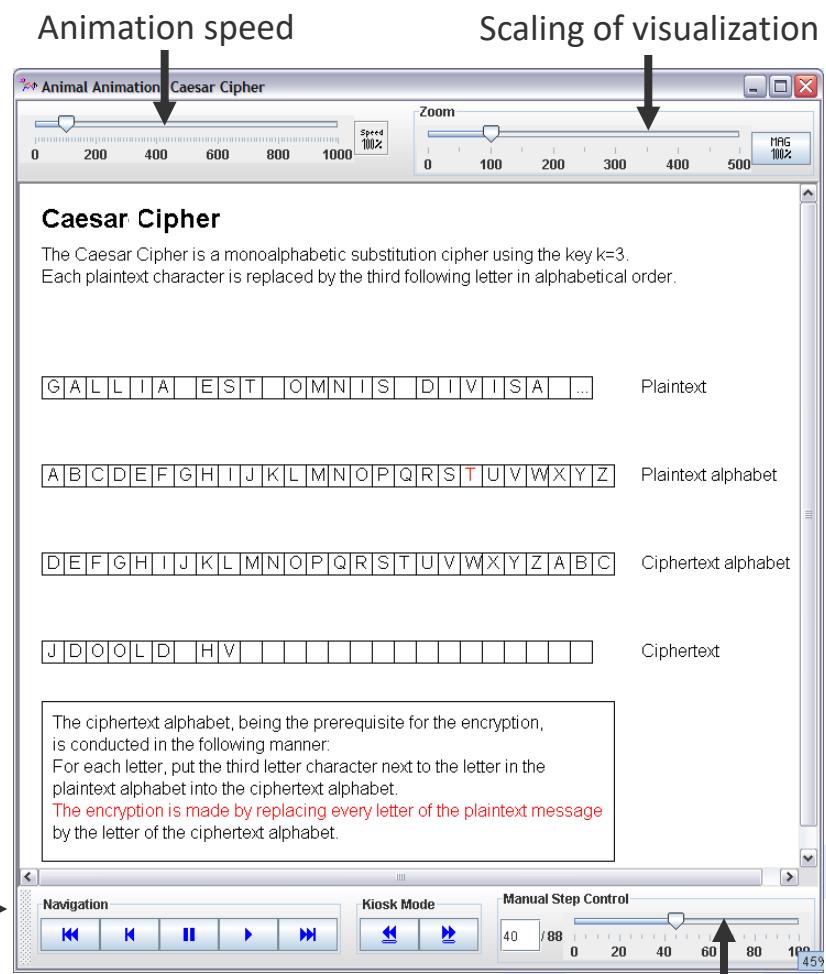
Animated visualization of several symmetric algorithms

- Caesar
- Vigenère
- Nihilist
- DES

CrypTool

- Menu: “Indiv. Procedures” \ “Visualization of Algorithms” \ ...
- Interactive animation control using integrated control center window.

Animation controls (next, forward, pause, etc.)

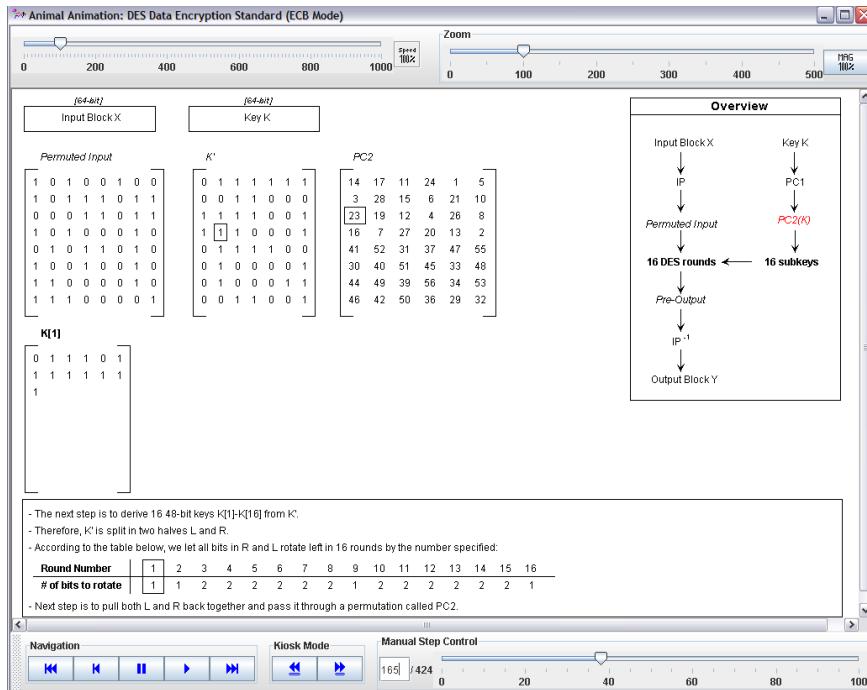


Direct selection of an animation step

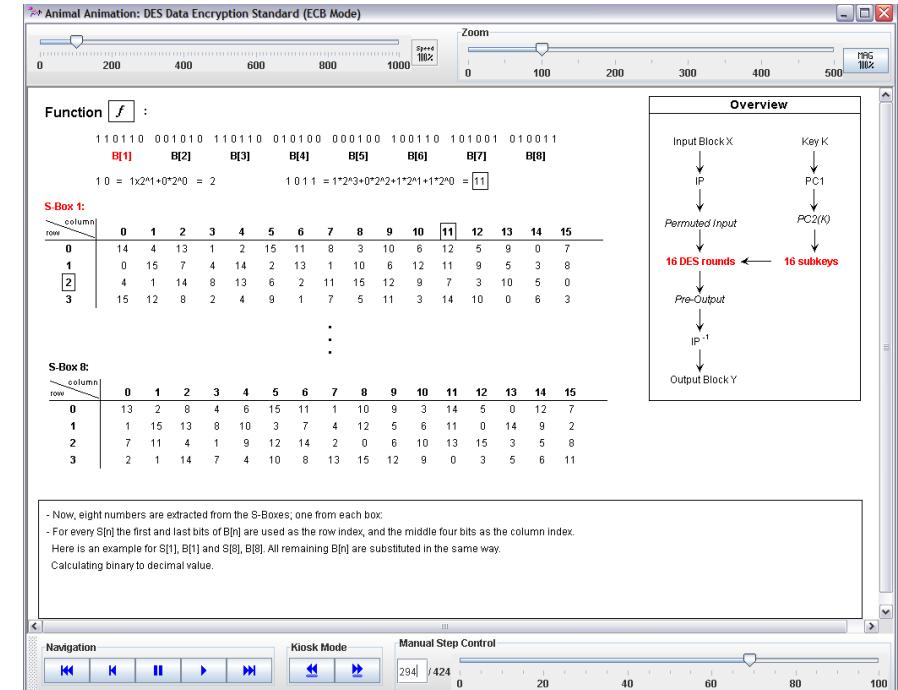
Examples (14)

Visualization of symmetric encryption methods using ANIMAL (2)

Visualization of DES encryption



After the permutation of the input block with the initialization vector (IV), the key K is permuted with PC1 and PC2.



The core function f of DES, which links the right half of the block R_{i-1} with the partial key K_i .

Examples (15)

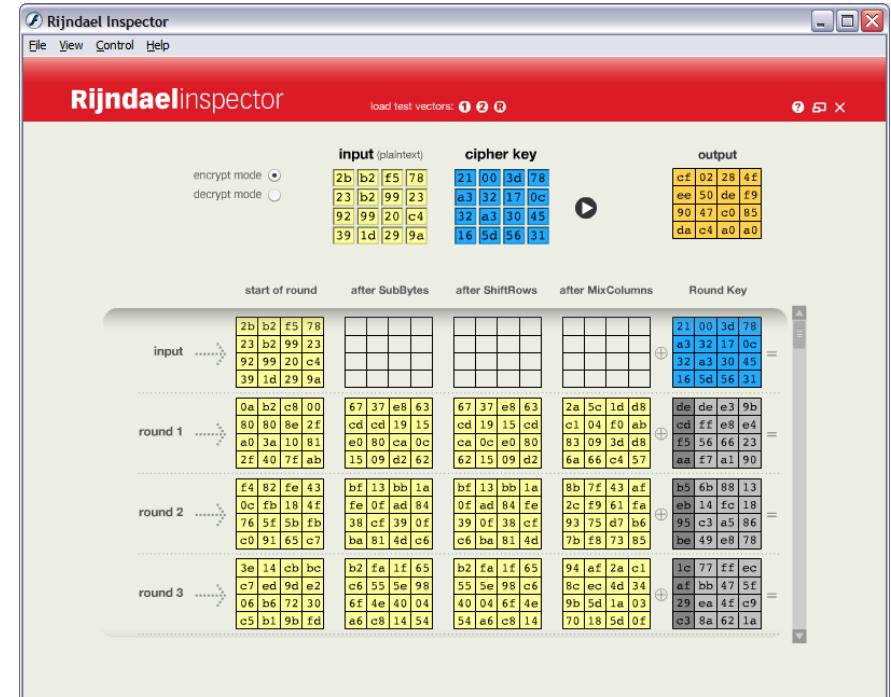
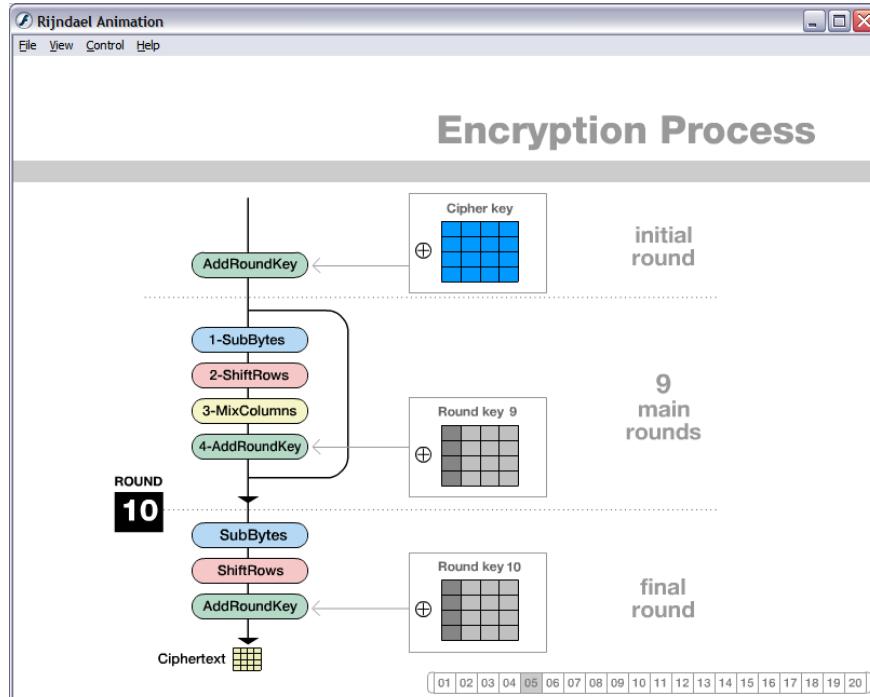
Visualizations of AES (Rijndael cipher) – in Flash

Rijndael Animation (the Rijndael cipher was the winner of the AES selection competition)

- Shows the encryption processes of each round (using fixed initial data)

Rijndael Inspector

- Test with your own data (shows the contents of the matrix after each round)



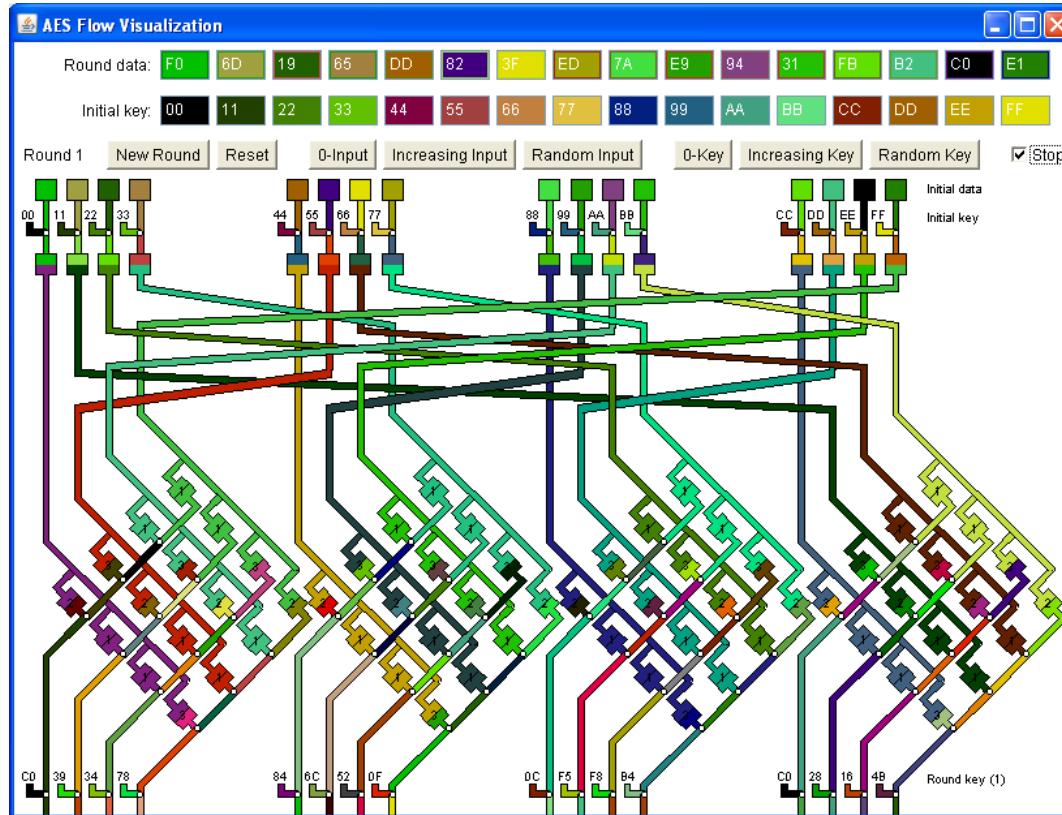
Menu: "Indiv. Procedures" \ "Visualization of Algorithms" \ "AES" \ "Rijndael Animation" or "Rijndael Inspector"

Examples (15)

Flow visualization of AES (Rijndael cipher) – in Java

Rijndael flow visualization

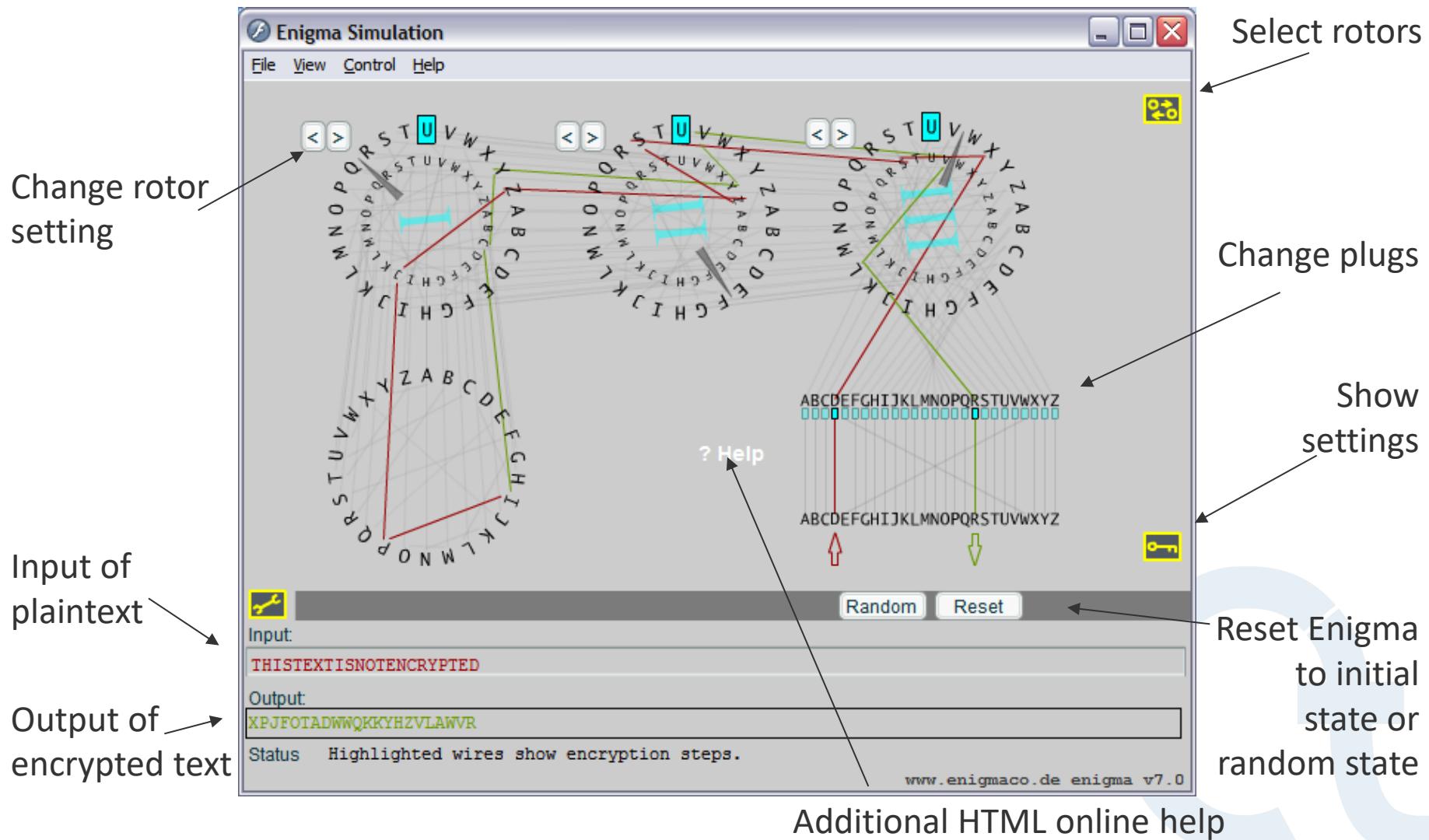
- Visualization of data changes per round using color gradient



Menu: "Indiv. Procedures" \ "Visualization of Algorithms" \ "AES" \ "Rijndael Flow Visualization..."

Examples (16)

Visualization of the Enigma encryption – in Flash



Examples (17)

Visualization of secure Email via S/MIME

S/MIME visualization

- Control Center: Sign/Encrypt messages with different parameters
- Animation: From the sender's creation of the message until it is read by the receiver

The image shows two windows related to S/MIME visualization.

S/MIME Visualization Control Center v1.0

In this window you can dynamically configure parameters for secure email messaging.

The visualisation is then done in two steps (control center & flash animation):

- At the control center you choose whether to encrypt or sign an email and the appropriate parameters.
- After clicking the start button the chosen procedure is visualized with a flash animation.

You can open more than one flash animation at once with different parameter from the control center.

Signing or encrypting

Signing
 Encrypting

Text of the message

Receiver: bob@web.com
Sender: alice@wonderland.com
Subject: Message will be signed

Donec consequat, ipsum non volutpat placerat, ...

Note: In this demonstration the text field can only handle 50 characters, longer texts will be shortened.

Load message text from file

Start signing

Choose sender's PSE

Internal PSE
 Personal PSE
Load existing PSE

Control parameters

Signature algorithm: RSA
Hash function: SHA-1
transfer encoding: quoted-printable
MIME type: multipart/signed

S/MIME Animation

File View Control Help

A cartoon character with glasses and yellow hair stands next to a computer monitor. The monitor displays a screenshot of Microsoft Outlook showing an email message. Below the monitor is a keyboard.

To ensure authenticity she makes use of the e-mail client's S/MIME features. One of these features enables her to attach a digital signature. Alice normally doesn't see her signature when she has composed the message, so let's take a look behind the scenes.

<< Prev. Chapter < Prev. Step Next Step >> Next Chapter >> Close

Menu: "Indiv. Procedures" \ "Protocols" \ "Secure E-Mail with S/MIME..."

Examples (18)

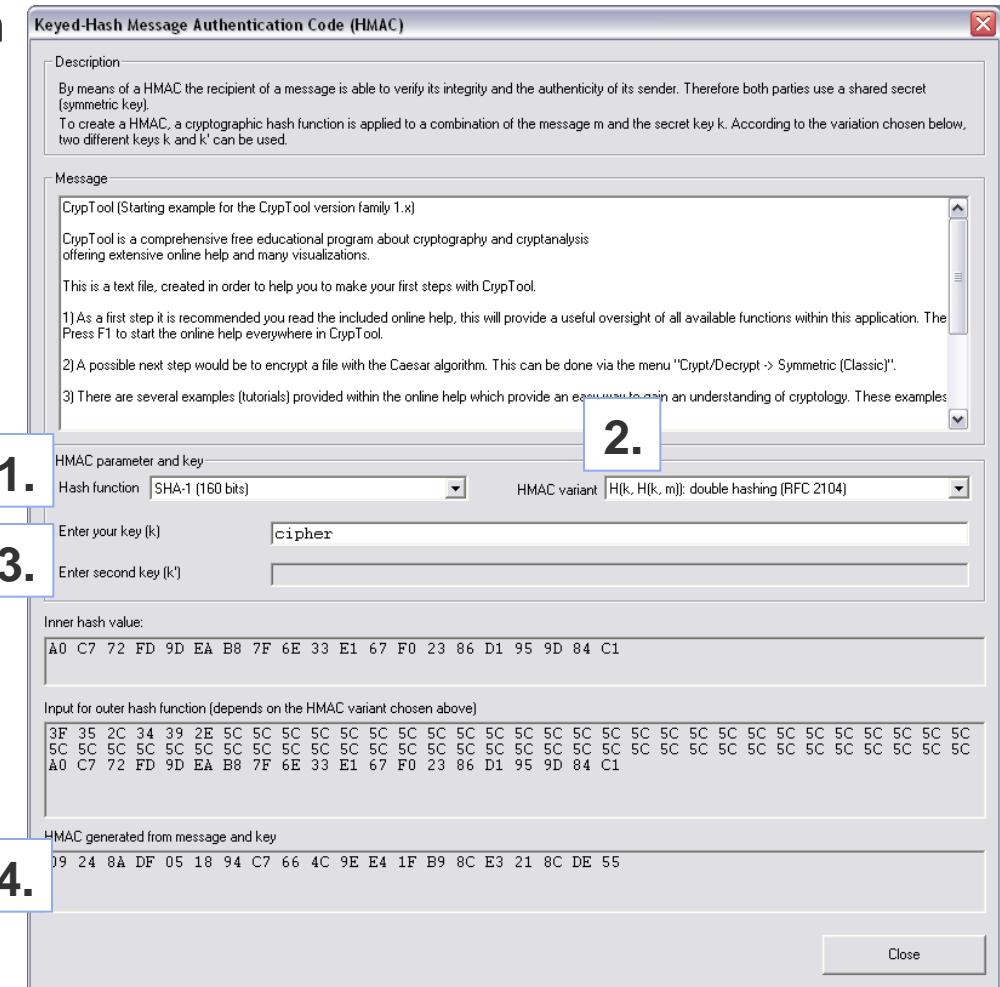
Generation of a keyed-hash message authentication code (HMAC)

Keyed-Hash Message Authentication Code (HMAC)

- Ensures
 - Integrity of a message
 - Authentication of the message
- Basis: a common key for sender and recipient
- Alternative: Digital signature

Generation of a MAC in CrypTool

1. Choose a hash function
2. Select HMAC variant
3. Enter a key (or keys, depending on the HMAC variant)
4. Generation of the HMAC (automatic)



Menu: "Indiv. Procedures" \ "Hash" \ "Generation of HMACs"

Examples (19)

Hash demonstration

Sensitivity of hash functions to plaintext modifications

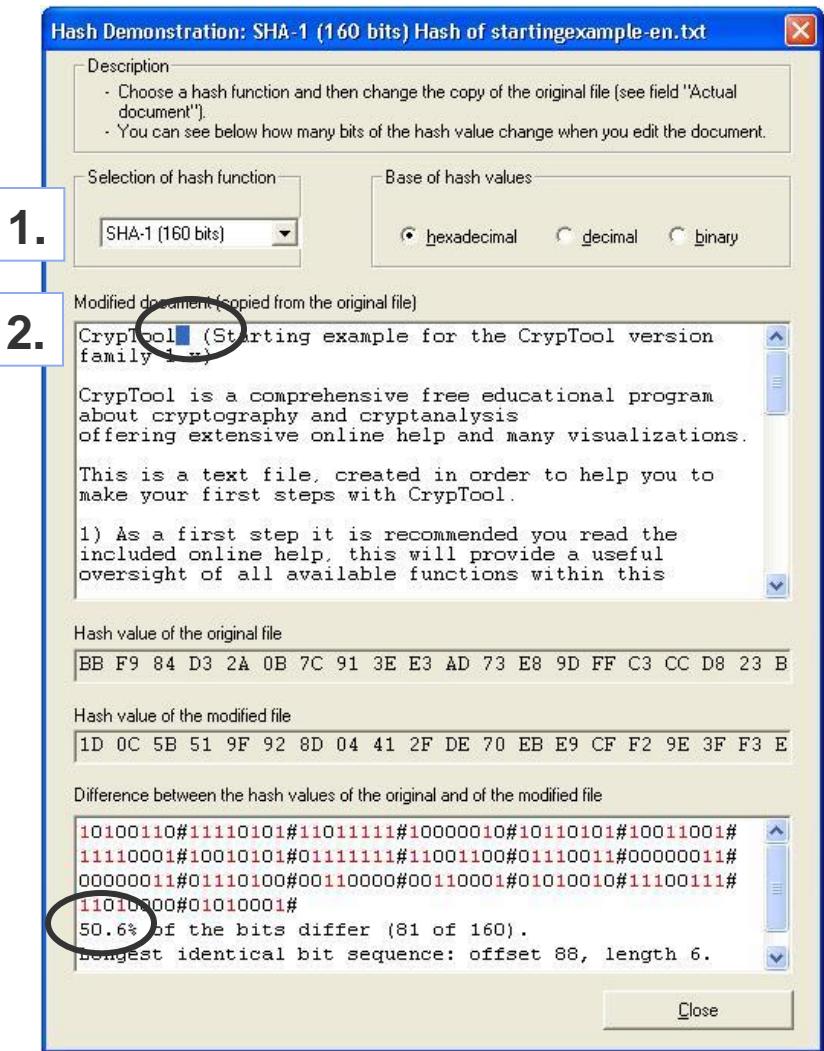
1. Select a hash function
2. Modify characters in plaintext

Example:

By adding a space after the word “CrypTool” in the example text, 50.6 % of the bits in the resulting hash value will change.

A good hash function should react highly sensitively to even the smallest change in the plaintext – “Avalanche effect” (small change, big impact).

Menu: “Indiv. Procedures” \ “Hash” \ “Hash Demonstration”



Examples (20)

Educational tool for number theory

- **Number theory**
supported by graphical elements and interactive tools
- **Topics**
 1. Integers
 2. Residue classes
 3. Prime generation
 4. Public-key cryptography
 5. Factorization
 6. Discrete logarithms

NT

Calculators Navigation Glossaries Help

3.2 Fermat Test page 4 of 11

Each prime p passes a test that results from Fermat's [Little Theorem](#):
For $b \in \{2, \dots, p-1\}$, test if $b^{p-1} \equiv 1 \pmod{p}$.

This test is called **Fermat Test**. Unfortunately some composite numbers pass it as well.

Example: $341 = 11 \cdot 31$, and yet $2^{340} \equiv 1 \pmod{341}$.

Passing the test provides no information. It must be repeated with a different base b :

$n = 341$ $2^{n-1} \equiv 1 \pmod{n}$ Test passed
 $\text{GCD}(b, n) = 1$

Definition: Let n be a composite number coprime to b .
If $b^{n-1} \equiv 1 \pmod{n}$, then it is said that

- n is **pseudoprime to b** ,
- and • b is a **liar for** (the primality of) **n** ,

otherwise it is said that

- b is a **witness against** (the primality of) **n** .

Theorem: If there are any witnesses against n ,
then they make up at least 50% of all $b \in \{1, \dots, n\}$ coprime to n . [Proof](#)

(Go on to the next page.)

Menu: "Indiv. Procedures" \ "Number Theory – Interactive" \ "Learning tool for number theory"

Examples (21)

Point addition on elliptic curves

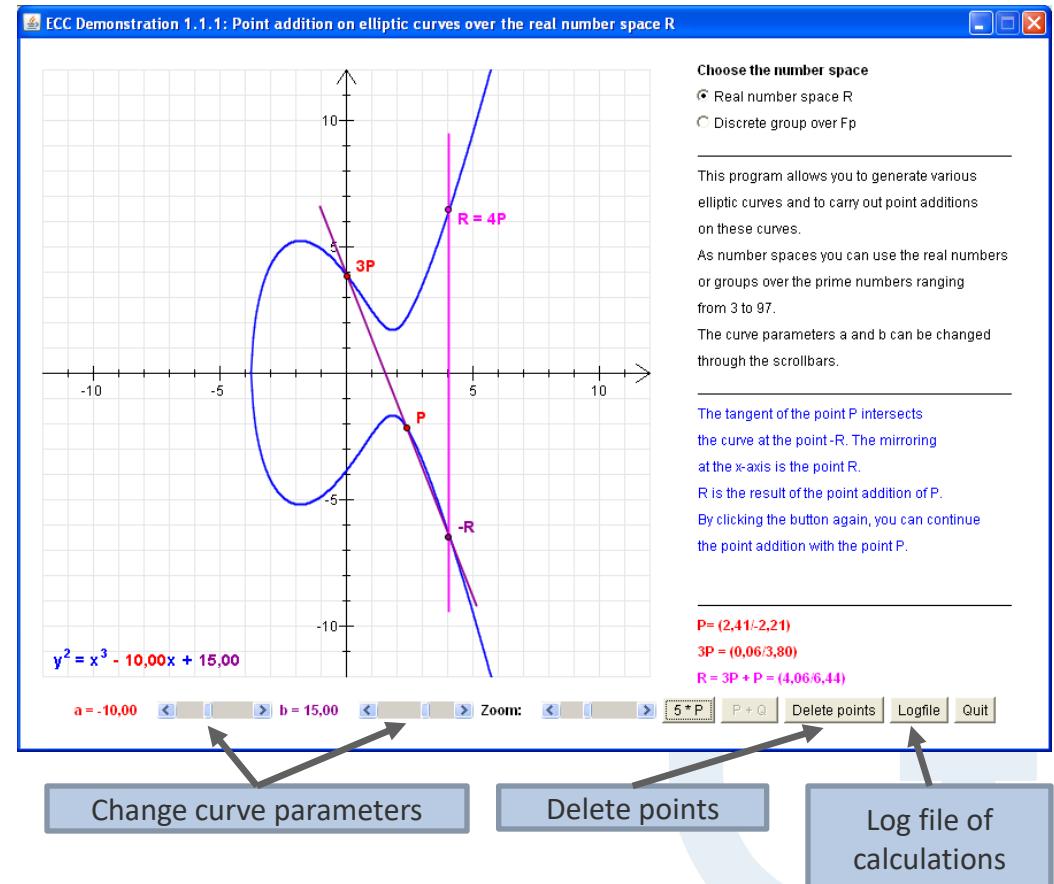
- Visualization of point addition on elliptic curves (both real and discrete)
- Foundation of elliptic curve cryptography (ECC)

Example 1: Add two different points

- Mark point P on the curve
- Mark point Q on the curve
- Pressing button “P+Q” creates point R:
 - The straight line through P and Q intersects the curve at point -R.
 - Mirroring -R over the X-axis produces the point R.

Example 2: Multiply a single point

- Mark point P on the curve
- Pressing button “2*P” creates point R:
 - The tangent of point P intersects the curve at point -R.
 - Mirroring -R over the X-axis produces the point R.



Menu: “Indiv. Procedures” \ “Number Theory – Interactive” \ “Point Addition on Elliptic Curves”

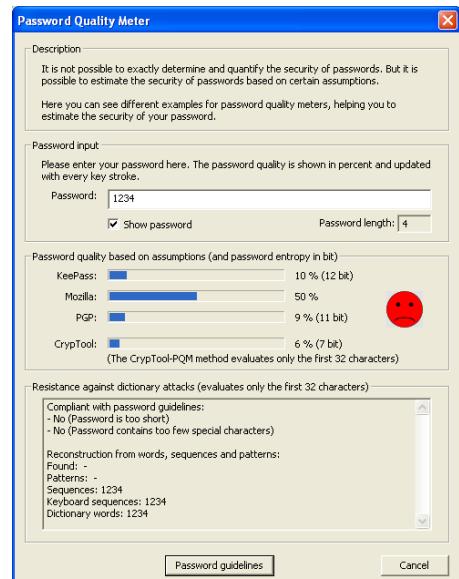
Examples (22)

Password quality meter (PQM) and password entropy (1)

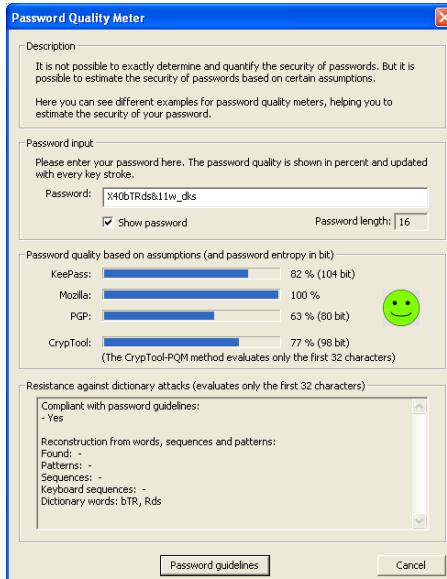
Functions

- Measure the quality of passwords
- Compare with PQMs in other applications: KeePass, Mozilla und PGP
- Experimental evaluation with the CrypTool algorithm
- Example: Input of a password in cleartext

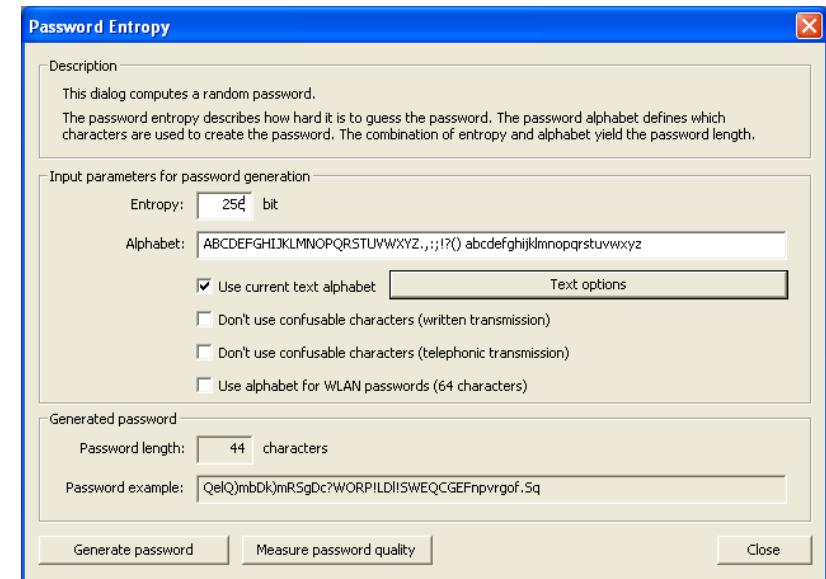
Password: **1234**



Password: **X40bTRds&11w_dks**



Menu: "Indiv. Procedures" \ "Tools" \ "Password Quality Meter"



Menu: "Indiv. Procedures" \ "Tools" \ "Password Entropy"

Examples (22)

Password quality meter (PQM) and password entropy (2)

Insights from the Password Quality Meter

- Password quality depends primarily on the **length of the password**.
- A higher quality of the password can be achieved by using **different types of characters**: upper/lower case, numbers, and special characters (**password space**)
- **Password entropy** is an indicator of the randomness of the password characters within the password space (higher password entropy results in improved password quality)
- Passwords should **not exist in a dictionary** (remark: here, a dictionary check is not yet implemented in CrypTool 1).

Quality of a password from an attacker's perspective

- Attack on a password (if any number of attempts are possible):
 1. Classical **dictionary attack**
 2. Dictionary attack **with variants** (e.g., 4-digit number combinations: "Summer2007")
 3. **Brute-force attack** by testing all combinations (with additional parameters such as limitations on the types of character sets)
- ⇒ A good password should be chosen so that attacks 1 and 2 do not compromise the password. Regarding brute-force attacks, the most important factors are the length of the password (recommended at least 8 characters) and the character set that was used.

Examples (23)

Brute-force analysis (1)

Brute-force analysis

Optimized brute-force analysis with the assumption that the key is partially known.

Example – Analysis with DES (ECB)

Attempt to find the remainder of the key in order to decrypt an encrypted text.
(Assumption: the plaintext is a block of 8 ASCII characters.)

Key (Hex)

68ac78dd40bbefd*
0123456789ab****
98765432106*****
0000000000*****
000000000000****
abacadaba*****
ddddddddd*****

Encrypted text (Hex)

66b9354452d29eb5
1f0dd05d8ed51583
bcf9ebd1979ead6a
8cf42d40e004a1d4
0ed33fed7f46c585
d6d8641bc4fb2478
a2e66d852e175f5c

Examples (23)

Brute-force analysis (2)

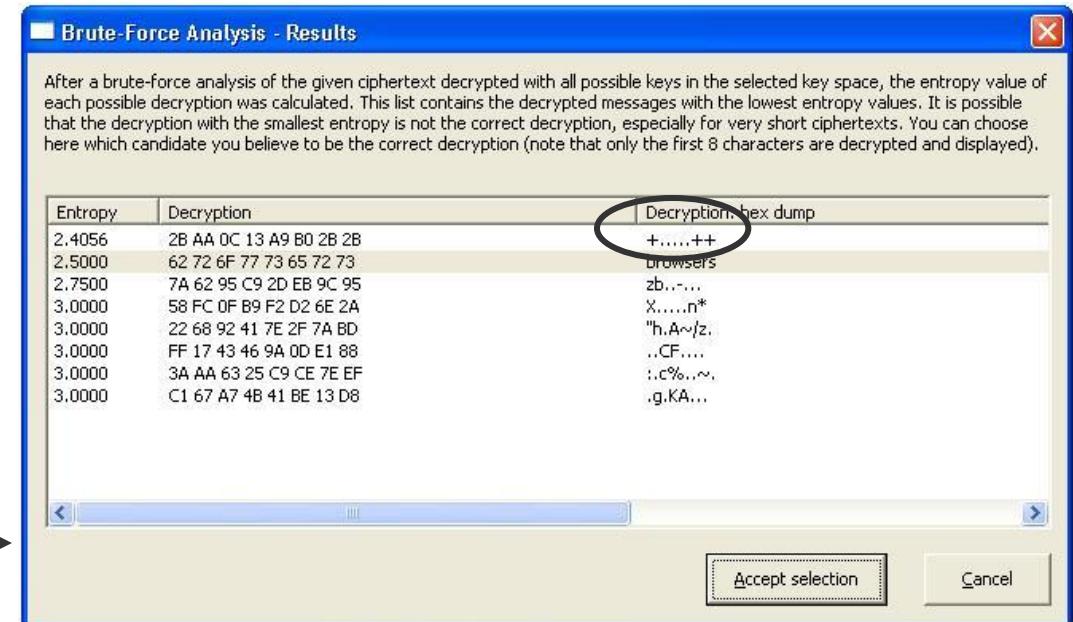
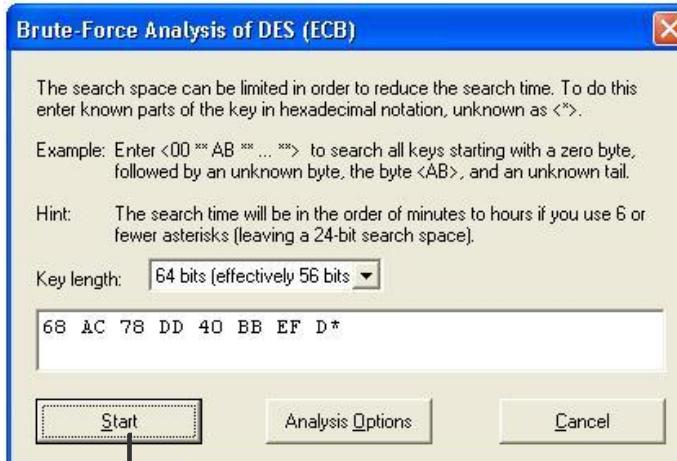
1. Input of encrypted text
2. Use brute-force analysis
3. Input partially known key
4. Start brute-force analysis

Select "View" \ "Show as HexDump"



5. Analysis of the results: the correct decryption usually has relatively low entropy. However, because a very short plaintext has been used in this example, the correct result does not have the lowest entropy.

Menu: "Analysis" \ "Symmetric Encryption (modern)" \ "DES (ECB)"



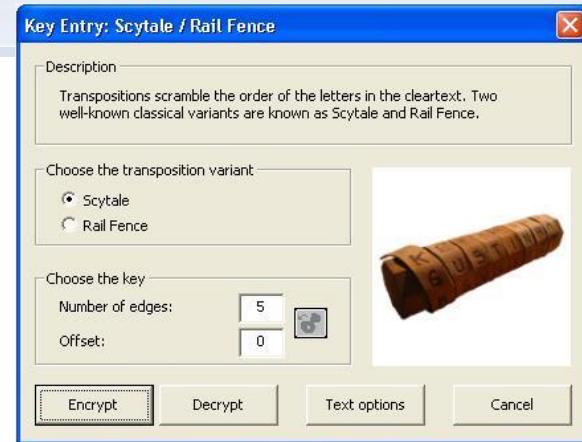
Examples (24)

Scytale / Rail Fence

Scytale and Rail Fence

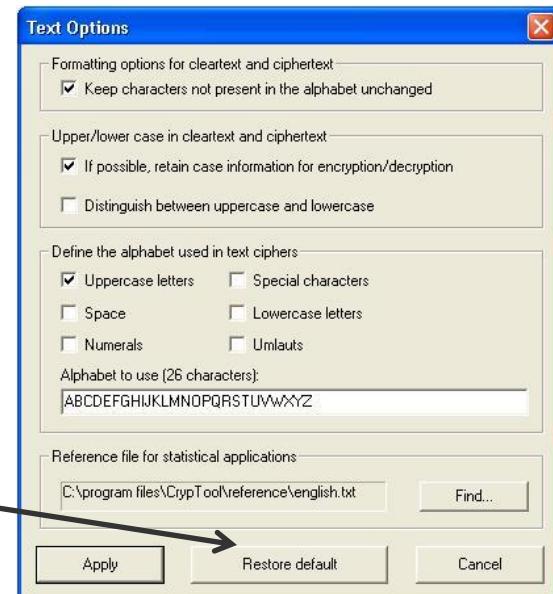
- Transpositions scramble the order of letters in the cleartext
- **Transposition variant**
 - Number of edges (Scytale)
 - Number of rows (Rail Fence)
 - Offset

Menu: “Crypt/Decrypt” \ “Symmetric (classic)” \ “Scytale / Rail Fence...”



Text options

- General text options (Menu: “Options” \ “Text Options...”)
- Formatting options for cleartext and ciphertext
- Processing of upper/lower case
- Alphabet for text processing (i.e., what set of characters should be encrypted/decrypted)
- Return to the default settings by clicking the “Restore Standard” button
- Creates the statistical reference patterns dynamically



Examples (25)

Hill encryption / Hill analysis (1)

Hill encryption

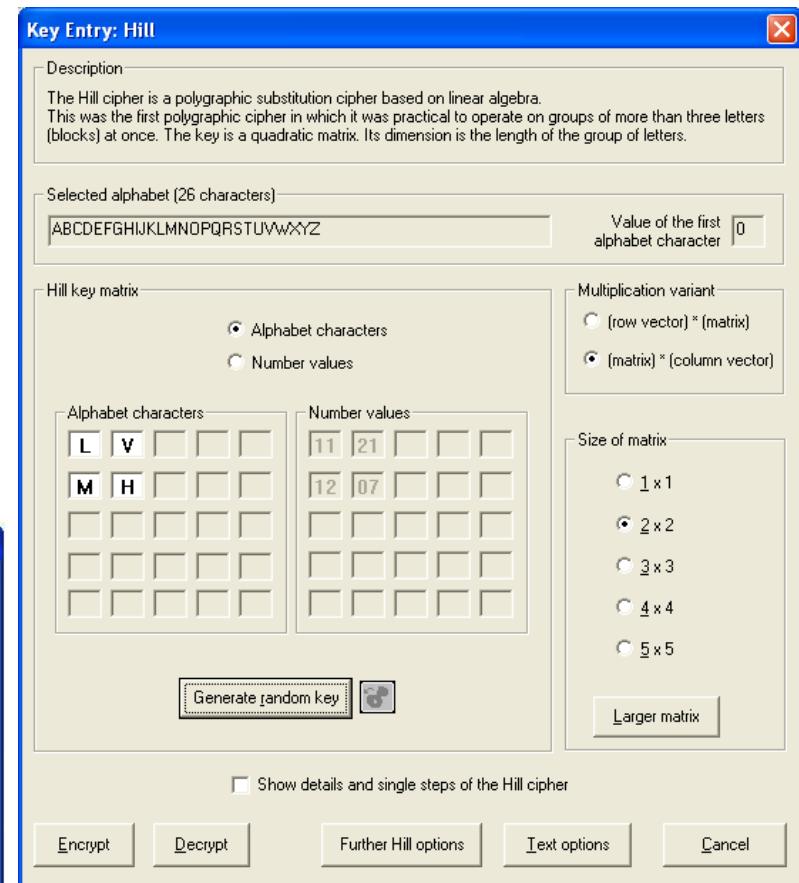
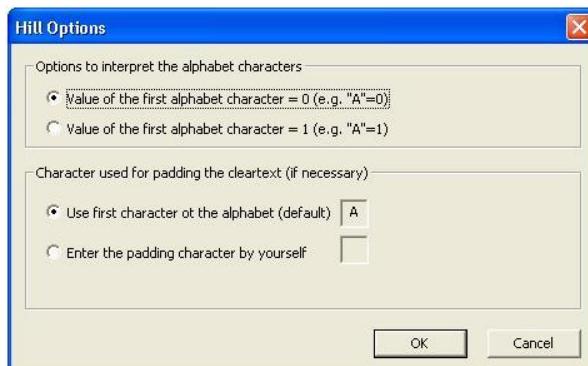
- Polygraphic substitution cipher
- Based on linear algebra

Key

- Alphabet characters (see text options) or number values
- Enter or generate random key
- Select multiplication variant
- Size of matrix
- Hill options

Menu:

“Crypt/Decrypt” \
“Symmetric (classic)” \
“Hill ...”



Examples (25)

Hill encryption / Hill analysis (2)

Hill encryption

- Sample text with key LVMH

Hill analysis (with known plaintext)

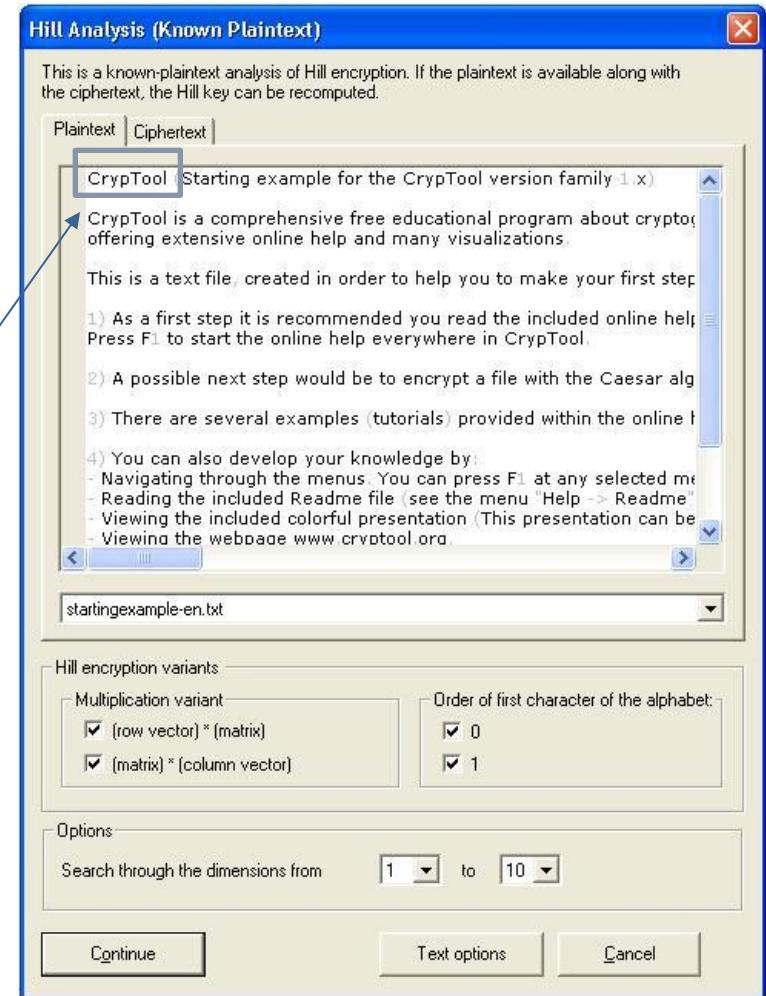
1. Long plaintext/ciphertext

- Select plaintext (startingexample-en.txt)
- Select ciphertext
(Hill encryption of <startingexample-en.txt>)
- Click “Continue” to search for the key

2. Reduced plaintext/ciphertext

- Clear all of the plaintext except the first word (“CrypTool”)
- Clear all of the ciphertext except for the first eight characters (“PnhdJovl”)
- Click “Continue” to reveal the key!

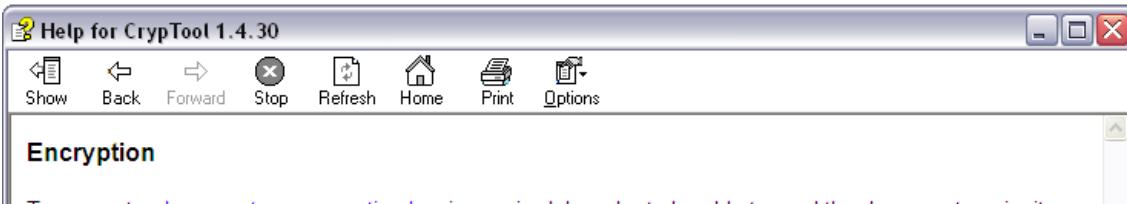
Which length of plaintext/ciphertext is required to find the correct encryption key?



Menu: “Analysis” \ “Symmetric Encryption (classic)” \ “Known Plaintext” \ “Hill...”

Examples (26)

CrypTool online help (1)



Several different [encryption algorithms](#) are available in [CrypTool](#). These are accessed via the [Crypt/Decrypt](#) menu.

Encryption algorithms

An encryption algorithm is required in order to transmit confidential information over insecure channels, for example, over a network. The information is [encrypted](#) by the originator prior to transmission and [decrypted](#) by the recipient following transmission.

A symmetric encryption algorithm is one in which the originator's and recipient's [keys](#) are identical. Encryption algorithms in which the originator and recipient have different keys are called asymmetric.

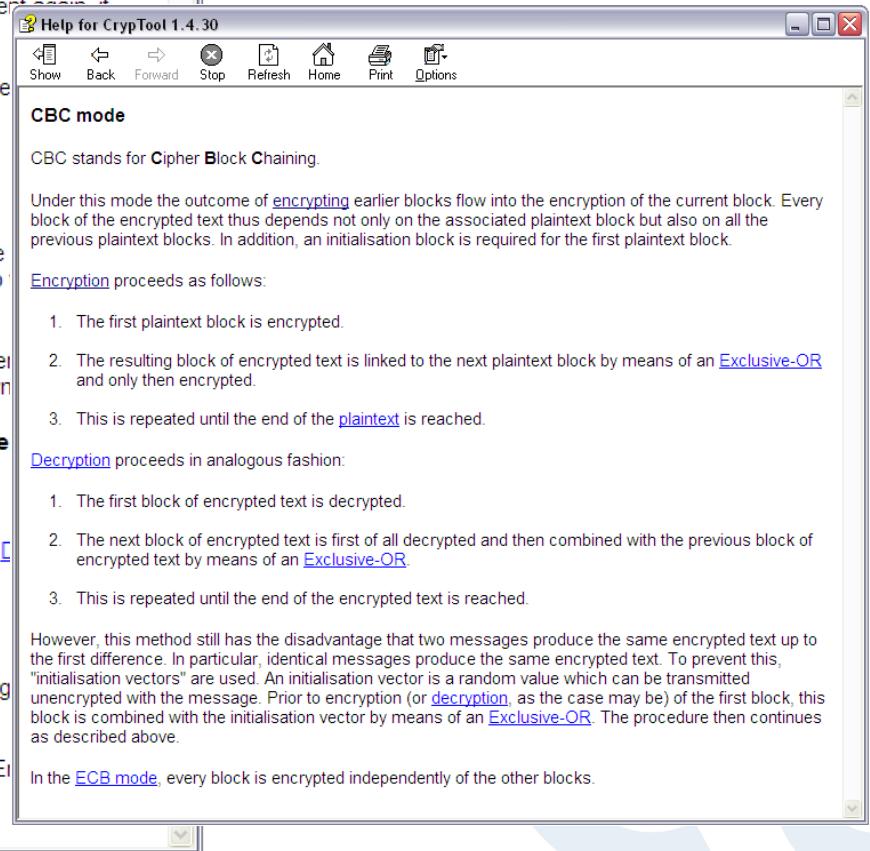
Modern symmetric encryption algorithms can be divided in **block ciphers** and **stream ciphers**.

- [Block ciphers](#) encrypt blocks of fixed length (e.g. 64 or 128 bit). Available in CrypTool are [IDEA](#), [RC2](#), [DES \(ECB\)](#), [DES \(CBC\)](#), [Triple DES \(ECB\)](#), [Triple DES \(CBC\)](#), [Rijndael \(AES\)](#), [MARS](#), [RC6](#), [Serpent](#), [Twofish](#), [DESX](#), [DESL](#) and [Loki97](#).
- [Stream ciphers](#) encrypt messages bit by bit. In this category CrypTool provides [RC4](#).

A summary of all the encryption algorithms available in CrypTool is contained on the help page under the [Crypt/Decrypt](#) menu.

Further information on encryption algorithms can be found in the [script](#), e.g. in the chapter "Encryption Procedures".

Menu: "Help" \ "Starting Page"



Examples (26)

CrypTool online help (2)

Help for CrypTool 1.4.30

Hide Back Forward Stop Refresh Home Print Options

Contents Index Search

Type in the keyword to find:

Lattice reduction

Lattice reduction
Liability (exclusion)
License terms
Line wrap
Links
Literature
MARS encryption algorithm
MD2 hash value
MD4 hash value
MD5 hash value
Menu (overview of all menus)
Miracl
Modular transformation
Modulo operator
Monoalphabetic substitution encryp
Network authentication
N-gram
Nihilist encryption algorithm
NIST
Normal distribution
NSA
NTL
Number Shark
Number system
Number theory
Offset
One-time pad
OpenGL
OpenPGP
OpenSSL
Options
Overview / Subsumption / Broader C
Padding
Parent window
Password
Pattern search

Display

Menu Lattice Based Attacks on RSA (Menu Individual Procedures \ RSA Cryptosystem)

The menu **Lattice Based Attacks on RSA** contains the following commands:

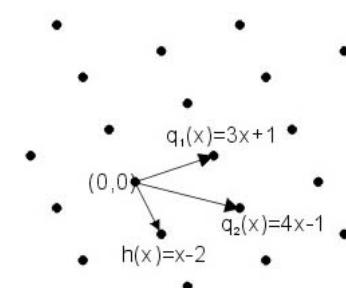
Factoring with a Hint	Attacks RSA with lattice reduction algorithms, if a part of one of the primes of N is known.
Attack on Stereotyped Messages	Attacks RSA with lattice reduction algorithms, if a part of the original cleartext of an intercepted ciphertext is known and if e is small.
Attack on Small Secret Keys	Attacks RSA with lattice reduction algorithms, if d is too small compared to N.

All attacks presented here are based on a common approach: first the task of breaking RSA is transformed into finding the root of a polynomial modulo an integer (mostly N) but to find such a root is a difficult problem.

To solve this problem further polynomials are generated which are known to have the same root. From the coefficients of these polynomials a latticebase is built. This is then reduced with, i.e. the LLL-algorithm to find a small vector.

From this newly found short vector a new polynomial is built. It can be proven that if the vector is short enough, the polynomial has the desired root not only modulo N, but also over the integers.

Example:



The polynomial $q_1(x) = 3x+1$ has a root x_0 modulo 7. It is supposed, that the polynomial $q_2(x) = 4x-1$ has the same root x_0 modulo 7. From these polynomials the vectors $b_1=[3 1]$ and $b_2=[4 -1]$ are built. All integer linear combinations of these vectors form points in a lattice. The Figure on the left shows a part of this lattice. Each point of the lattice now can again be interpreted as a polynomial having the desired root. A short vector of the lattice is $b_3=[1 -2]$ from which the polynomial $h(x) = x-2$ is built. This polynomial has a root in $x_0=2$ over the integers as well as modulo 7. That $x_0=2$ is also a root of the polynomials $q_1(x)$ and $q_2(x)$ modulo 7 can be easily established. ($3x_0+1=7, 7 \text{ modulo } 7 = 0$)

Examples (26)

CrypTool online help (3)

Help for CrypTool 1.4.30

Hide Back Forward Stop Refresh Home Print Options

Contents Index Search

Type in the keyword to find:

base

Base64 coding

- BC
- Binary exclusive-OR
- Birthday attack / birthday paradox
- Bit length
- Block cipher
- Blocks
- Books
- Bounding box
- Brute-force attack
- Byte addition
- Caesar encryption algorithm
- Card game
- Cascade
- Cascading cipher
- CBC mode
- Certificate
- Challenge
- Challenge-response demonstration
- Chi² distribution
- Chinese remainder theorem
- Chosen-plaintext attack
- Ciphertext
- Ciphertext-only attack
- Clipboard
- Codings
- Coin toss
- Column transposition
- Compress
- Congruence generator
- Contact
- Context / Subsumption / Overview
- Copyright
- Correlation
- Cryptanalysis
- Crypto competitions / Crypt

Display

Comparison of Base64 and UU coding

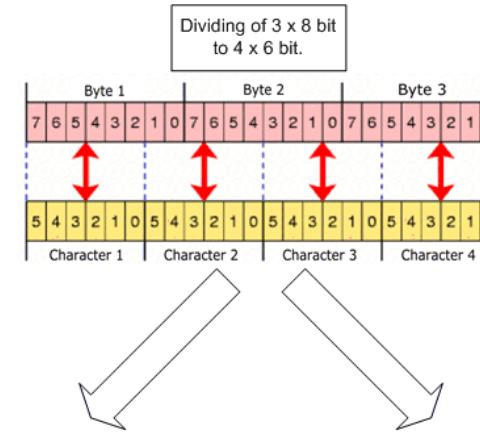
The encoding procedures of [Base64](#) and [UUencode](#) are quite similar, which is shown by the following figure:

Base64 UUencode

Step 1: Splitting the data stream – same procedure in both encodings.

Step 2: Representation of the 6 bit values – different procedures.

Dividing of 3 x 8 bit to 4 x 6 bit.



Get the characters from Base64 coding table.
(defined in an IETF standard)

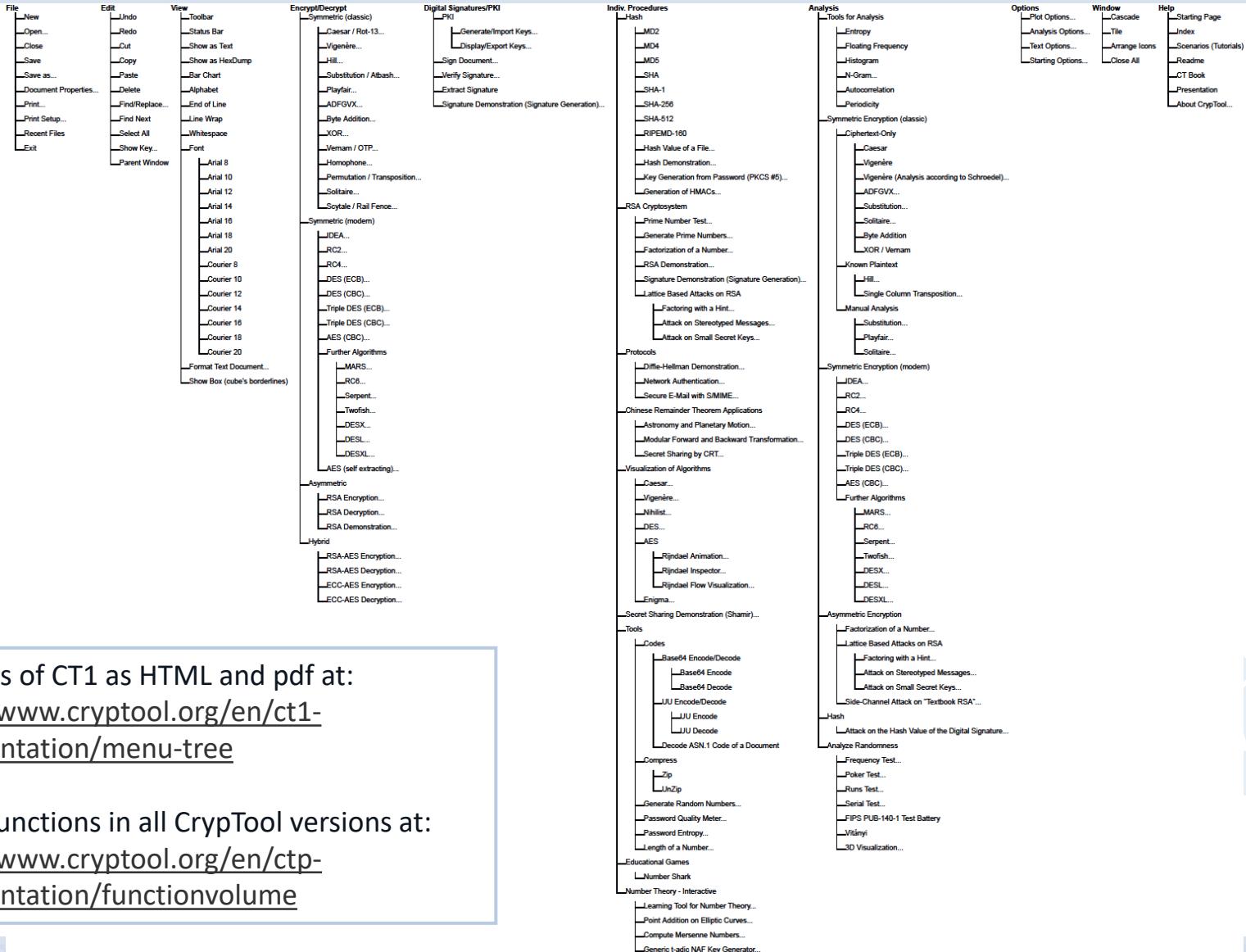
Get the characters, increased by decimal 32, from the ASCII char set.

Because of the similar encoding procedure, there are also shared advantages and drawbacks:

Advantages	Drawbacks
• Arbitrary binary data can be represented with a 6-bit	

Examples (26)

Menu tree of the program CrypTool 1.4.40



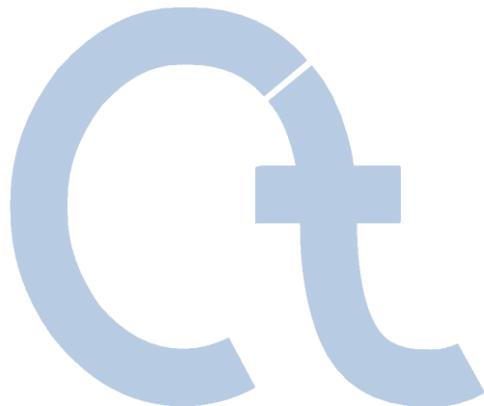
Menu trees of CT1 as HTML and pdf at:

<https://www.cryptool.org/en/ct1-documentation/menu-tree>

List of all functions in all CrypTool versions at:

<https://www.cryptool.org/en/ctp-documentation/functionvolume>

Content



- I. CrypTool and Cryptology – Overview
- II. Features of CrypTool 1
- III. Examples
- IV. Project / Outlook / Contact**
- Appendix



Future CrypTool Development (1)

Examples of what is coming after the release of CrypTool 1.4.40 (see readme for details)

CT1 FIPS test with the ability to analyze packets with lengths other than 2500 bytes, etc.

JCT Tri-partite key agreements

JCT Visualization of the interoperability between S/MIME and OpenPGP formats

JCT Entropy analysis

JCT Fleissner grille, Autokey Vigenère, interactive cryptanalysis of classic ciphers

JCT Analysis of transposition ciphers using the ACO algorithm

JCT Visualization of zero-knowledge proofs

JCT+CT2 Visualization of Quantum Key Agreement, BB84 protocol

JCT Visualization of the SETUP attack against RSA key generation (Kleptography)

JCT Action history with the ability to create and replay any given cipher cascade

CT2 Comprehensive visualization on the topic of prime numbers

CT2 GNFS (General number field sieve)

CT2 Encryption and automated cryptanalysis of the Enigma machine (and possibly of M-138 and Sigaba as well)

CT2 Cube attack (I. Dinur and A. Shamir: "Cube Attacks on Tweakable Black Box Polynomials", 2008)

CT2 Demonstration of Bleichenbacher's and Kuehn's RSA signature forgery

CT2 Demonstration of virtual credit card numbers (as an educational tool against credit card abuse)

CT2 WEP encryption and WEP analysis

CT2 Mass pattern search

CT2 Framework for distributed cryptanalysis → CrypCloud

CT2 Demonstration of SOA security (SOAP messages with WS-Security)

CT2 Framework to create and analyze LFSR stream ciphers

CT2/JCT Creation of a command-line interface for batch processing

CT2/JCT Modern pure plugin architecture with plugin reloading capability

All Expanded parameterization and flexibility of present algorithms

Ideas Visualization of the SSL protocol // Demonstration of visual cryptography // Post-quantum computing // Cryptography as web application

CT1 = CrypTool 1.x

New versions of CT:

CT2 = CrypTool 2

JCT = JCrypTool

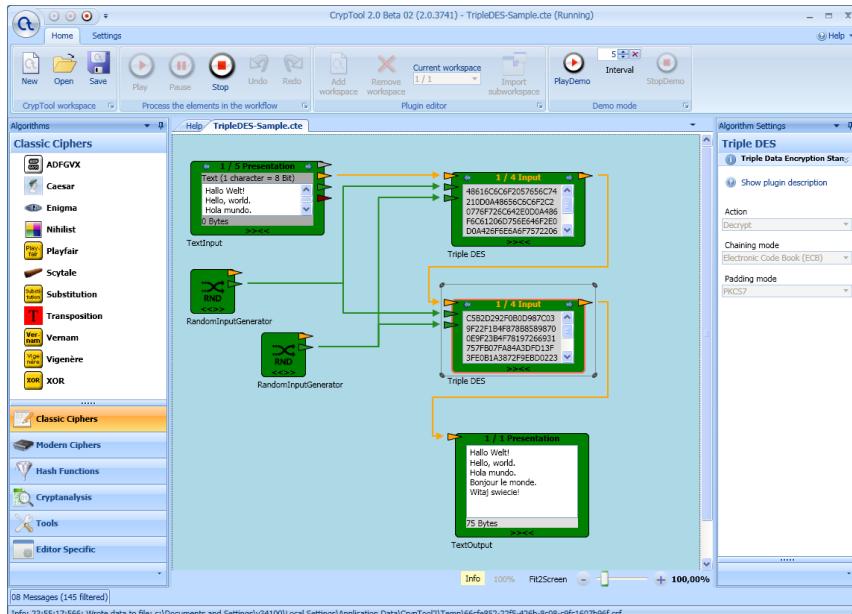
(both introduced on the next slides)



Future CrypTool Development (2)

In Progress: the two successor versions of CT v1 (see readme file)

1. JCT: Port and redesign of the C++ version with Java / SWT / Eclipse / RCP
 - see: <https://github.com/jcryptool/core/wiki>
 - Release Candidate RC8 is available since October 2016 (since 2010, weekly builds are created each week).
2. CT2: Port and redesign of the C++ version with C# / WPF / Visual Studio / .NET
 - Allows visual programming and distributed calculations (CrypCloud)
 - see: <https://www.cryptool.org/en/ct2-documentation>
 - Release 2.0 is available since August 2014 (since July 2008, nightly builds are created each day).

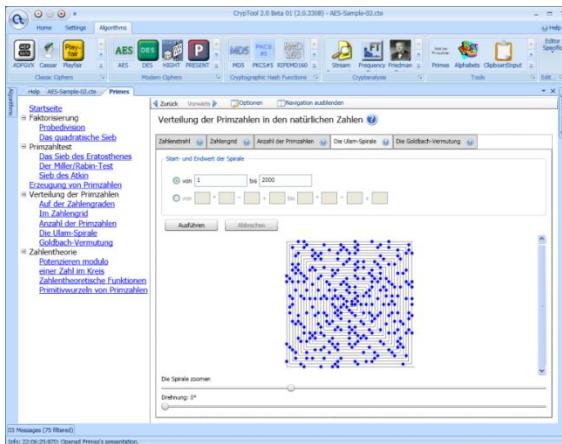
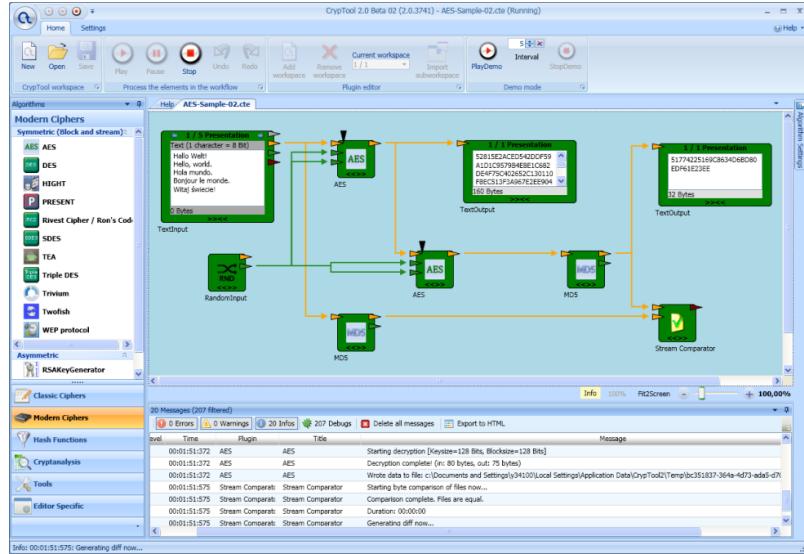


CrypTool 2 (CT2) (screenshot from 2010)

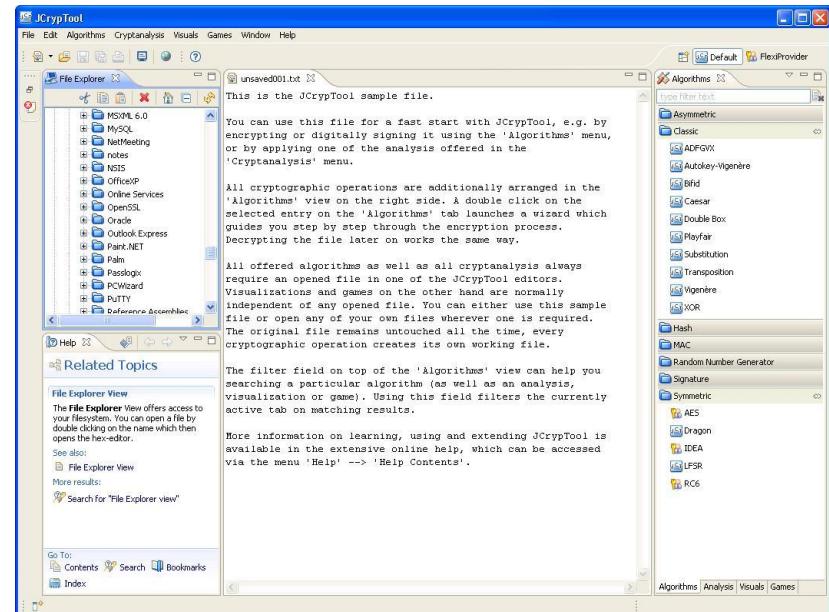


JCrypTool (JCT) (screenshot from 2010)

Future CrypTool Development (3)



CrypTool 2 (CT2) (screenshots from 2010)



JCrypTool (JCT) (screenshots from 2010)

CrypTool as a Framework for your Own Work

Proposal

- Reuse the comprehensive set of algorithms, included libraries, and interface elements as a foundation.
- Free training to help getting started with CrypTool development.
- Advantage: code written for university theses or other projects will not simply disappear, but rather be further maintained.

Current development environment for CT1: Microsoft Visual Studio C++ , Perl, Subversion Source Code Management

- CrypTool 1.4.40: Visual C++ .NET (= VC++ 9.0)(= Visual Studio 2008 Standard)
- Description for developers: see CrypToolDeveloperReadme.pdf within the code repository
- Sources and binaries of release versions are available for download.
To get sources of current betas, anyone has read access to the Subversion repository.

Development environments for CT2 and JCT

- CT2 – C# version: .NET 4.0 with Visual Studio 2015 Express Edition (free), and WPF
- Java – Java version: Eclipse 4.6, RCP, SWT (free)



CrypTool – Request for Contribution

Every contribution to the project is highly appreciated

- Feedback, criticism, suggestions, and ideas
- Integration of additional algorithms, protocols, analysis (consistency and completeness)
- Development assistance (programming, layout, translation, testing)
- CT1: for the current C/C++ project, and
- For the new projects (preferred):
 - C# project: “CrypTool 2” = CT2
 - Java project: “JCrypTool” = JCT
- In particular, university faculties that use CrypTool for educational purposes are invited to contribute to the further development of CrypTool.
- Samples of open tasks are on the following developer pages:
 - CT2: See the list <https://www.cryptool.org/trac/CrypTool2/wiki/WikiStart>
 - JCT: See the wiki <https://github.com/jcryptool/core/wiki/Project-Ideas>
- Users that make a significant contribution can request to be referenced by name in the online help, the readme file, the about dialog, and/or on the CrypTool website.
- CrypTool 1 is currently downloaded over 6,000 times per month from the CrypTool website. Just over half of these downloads are of the English version.
The two successors are already being downloaded over 2,000 times a month each.

CrypTool – Summary

THE e-learning program for cryptology

- Successfully active as an open-source project for over 15 years
- Over 600,000 total downloads
- Widespread international usage in schools, universities, companies, and government agencies
- Extensive online help and documentation
- Available for free
- Multilingual

CT: The worldwide most wide-spread e-learning program for cryptography and cryptanalysis.



Contact

Prof. Bernhard Esslinger

University of Siegen
Institute for Economics and Business Computing

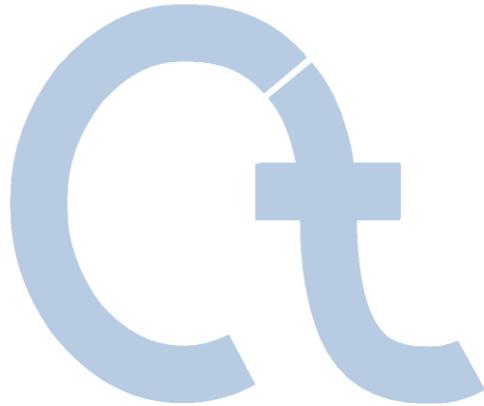
bernhard.esslinger@uni-siegen.de

www.cryptool.org

Additional contacts: See readme within the CrypTool 1 package



Content



- I. CrypTool and Cryptology – Overview
- II. Features of CrypTool 1
- III. Examples
- IV. Project / Outlook / Contact

Appendix (Literature, CrypTool-related Websites, Download)



Additional Literature

As an introduction to cryptology – and more

- Klaus Schmeh, “*Codeknacker gegen Codemacher. Die faszinierende Geschichte der Verschlüsselung*”, 2nd edition, 2007, W3L [German]
- Simon Singh, “*The Codebook*”, 1999, Doubleday
- Johannes Buchmann, “*Introduction to Cryptography*”, 2nd edition, 2004, Springer
- Paar / Pelzl: “*Understanding Cryptography – A Textbook for Students and Practitioner*”, 2009, Springer
- [HAC] Menezes / van Oorschot / Vanstone, “*Handbook of Applied Cryptography*”, 1996, CRC Press
- van Oorschot / Wiener, “*Parallel Collision Search with Application to Hash Functions and Discrete Logarithms*”, 1994, ACM
- Antoine Joux, “*Algorithmic Cryptanalysis*”, 2009, Chapman & Hall/CRC Cryptography and Network Security Series
- Additional cryptography literature – see also the links at the CrypTool web page and the literature in the CrypTool online help (e.g. by Wätjen, Salomaa, Brands, Schneier, Shoup, Stamp/Low, Oppiger, Martin, etc.)
- **Importance of cryptography in the broader context of IT security and risk management**
 - See e.g. Kenneth C. Laudon / Jane P. Laudon / Detlef Schoder, “*Wirtschaftsinformatik*”, 3rd edition 2016, Pearson, chapter 15 about IT Security [German]
 - Wikipedia: http://en.wikipedia.org/wiki/Risk_management
 - CrypTool site: <https://www.cryptool.org/en/ctp-education/awareness>

The CrypTool Portal: www.cryptool.org

The screenshot shows the homepage of the CrypTool 1 website. At the top, there's a blue header with the CrypTool 1 logo and the tagline "Cryptography for everybody". Below the header is a large image of a person working at a desk covered in many papers and documents. The main content area has a white background. On the left, a sidebar lists "CRYPTOOL 1", "CRYPTOOL 2", "JCT JCRYPTOOL", "CRYPTOOL ONLINE", and "MYSTERY TWISTER C3". In the center, a section titled "What is CrypTool 1" describes it as an open-source Windows program for cryptography and cryptanalysis. Below this are download links for "FREE DOWNLOADS": "CrypTool 1" (highlighted in blue), "CrypTool 2", and "JCrypTool". At the bottom of the page, there are navigation links for "About CrypTool 1", "Documentation", "Downloads", and "Screenshots".

CRYPTOTOOL 1 NEWS

THIRD PUBLIC BETA OF CRYPTOTOOL 1.4.31

The new CrypTool 1.4.31 beta is ready. This version is now available in 6 languages German, English, Spanish, Polish, Serbian and Greek. We would appreciate if you test this beta in detail and give us feedback. The release version of CT 1.4.31 is scheduled for

About CrypTool 1

CrypTool 1 (CT1) is a free, open-source Windows program for cryptography and cryptanalysis. It is available in 5 languages and the most wide-spreaded e-learning software of its kind. It supports both contemporary teaching methods at schools and universities as well as awareness training for employees and civil servants. The program can be downloaded [here](#). Originally designed as an internal business application for information security training, CrypTool 1 has since developed into an important open-source project in the field of cryptology and IT security awareness. CrypTool 1 is written in C++.

About

- [CrypTool Introduction](#)
- [CrypTool in Education](#)
- [CrypTool for Awareness](#)
- [Coverage in Print Media](#)
- [Awards](#)
- [Contributors](#)
- [Related Projects](#)
- [Contact](#)

Features

- [CrypTool Features](#)
- [Roadmap](#)

Media

- [Screenshots](#)
- [Screencast](#)

Documentation

- [Presentations](#)
- [Script](#)
- [Crypto History](#)
- [Links / Books](#)

www.cryptool-online.org

Experiment
with
cryptography
from within
your smart
phone

The screenshot shows a web browser window with three tabs open: "Awareness - CrypTool Portal", "Padding oracle attack - Wiki", and "About CrypTool-Online - Cry...". The main content area displays the "CRYPTOOL-ONLINE" logo and a "DID YOU KNOW..." box stating "that you can create Barcodes with CrypTool-Online?". Below this is a navigation bar with links: "About CrypTool-Online", "Ciphers", "Codings", "Cryptanalysis", "Highlights", and "Documentation". On the left, there's a sidebar with sections for "CIPHERS" (illustrated with a historical cipher wheel) and "CODINGS" (illustrated with a barcode). The right side contains the "About CrypTool-Online (CTO)" page content.

CRYPTOOL-ONLINE

Cryptography for everybody

DID YOU KNOW...
that you can create Barcodes with CrypTool-Online?

About CrypTool-Online Ciphers Codings Cryptanalysis Highlights Documentation

CTO OVERVIEW

CIPHERS

How do classical ciphers work?

CODINGS

Where are codings used and how do they work?

About CrypTool-Online (CTO)

Encrypt directly within your browser

CrypTool-Online provides an exciting insight into the world of cryptology. A variety of ciphers, coding methods, and analysis tools are introduced together with illustrated examples. Our emphasis is on making explanations easy to understand in order to further the general interest in cryptography and cryptanalysis. Therefore, you can experiment with the introduced methods in an interactive way directly on the website.

So you can learn the fundamentals of historically relevant ciphers in a little while, and also use the tools under "**Ciphers**" to encrypt messages yourself. You can also decrypt and analyze already encrypted messages and discover weaknesses of different ciphers. Under **Highlights**, you can for instance check the modern cipher AES or let the site generate good passwords for you.

CrypTool-Online is the online version of the e-learning program CrypTool. The so-called download (or offline) versions of CrypTool are also free and suitable for working with longer texts and conducting high performance analyses on encrypted messages.

Developers who like to join and enhance CTO with self-written plugins, find a good guidance in the Wiki. Especially the page How-to-Start leads you step-by-step.

Further Offers from the CT Project

Members in the family of CrypTool-related websites:

- **CrypTool 1** site (CT1)
- **CT2** (download and developer site)
- **JCT** (download and developer site)
- **CrypTool-Online**
(allows to experiment with cryptography from within your browser, at the PC or with your smart phone)
- **CryptoPortal** for teachers (currently only in German)
- **Schülerkrypto** for pupils & teachers (currently only in German)
- **MysteryTwister C3 (MTC3)** is an international crypto challenge contest.



The screenshot shows the homepage of the CRYPTOPORTAL für Lehrer website. At the top, there is a navigation bar with links for "Über", "Unterrichtsmaterial", "Linksammlung", "Registrierung", "Cryptool", and "Einloggen". The main content area features a search form titled "Filterkriterien" with dropdown menus for "Land" (set to "alle Länder"), "Schultyp" (set to "alle Schultypen"), and "Autor" (set to "alle Autoren"). Below the search form is a list of teaching materials:

- [1] Die Stromchiffre A5**
Autor: PS
Land: Deutschland - alle Bundesländer
Schultyp: Gymnasien
In dieser Ausarbeitung zum Seminar IT-Sicherheit wird der auf der Verschaltung von linear rückgekoppelten Schieberegistern (LFSR) basierende Algorithmus A5 und die bisher gefundenen [...]
[a5_thesis.pdf](#) 8 mal heruntergeladen
- [2] Die wichtigsten Verfahren der Kryptologie**
Autor: HW
Land: Deutschland - Berlin
Schultyp: alle Schultypen
Die Präsentation besteht aus zwei Folien. In der ersten wird die Entwicklung der klassischen Kryptographie (von Caesar bis zum one-time-pad) dargestellt. In der zweiten wird ein Überblick zur [...]
[Krypto-Entwicklung.ppt](#) 15 mal heruntergeladen
- [3] Kryptografie für Jedermann**
Autor: Consultant
Land: Deutschland - alle Bundesländer
Schultyp: alle Schultypen
Einführung in die Kryptografie, Erläuterungen zu populären kryptografischen Primitiven und Protokollen [...]
[Originalpräsentation.pdf](#) 14 mal heruntergeladen

The teacher's portal is currently only available in German. We would greatly welcome any help to build an English version too.

MysteryTwister C3

THE CRYPTO CHALLENGE CONTEST

Search... All Search!

Start Challenges Forum MysteryTwister I

About MTC3 Partners News

NUMBER OF ACTIVE MEMBERS: 7886 Register here

MTC3 PARTNERS

CITS

Follow us: [f](#) [t](#)

Login DE EN

CONNECT TO OTHER USERS

Discuss the challenges with other MTC3 users in the forum. Share your ideas and help bring each other closer to the solution.

Register here

Who is online

In total there are 21 user online :: 21 registered, 0 hidden
Most users ever online was 25 on Wed May 26, 2010 3:3

Registered users:

Welcome to MTC3 — The Cipher Contest

You like riddles? You always loved to solve the crosswords in your newspaper? Or maybe you are just curious and want to find out about some of the ways to hide a secret (and possibly even to uncover it)? This is your place! Here at MysteryTwister C3 you can solve crypto challenges, starting from the simple Caesar cipher all the way to modern AES we have challenges for everyone. Our challenges range from level I to III, and an additional level X for "mystery" challenges (they may have been unsolved for a long time, mostly we don't know their solution or have no idea whether there is a solution at all). If you are a beginner its probably best if you start trying those challenges that have been solved mostly (see table below). Additional information regarding MTC3 can be found on our [about page](#).

art 1' +++ [19:59 - 01.03.2017] Zylius solved the Level II challenge 'Cracking SHA1-Hashed Passwords' +++ [15:27 - 01.03.2017] capiaghi solved the Level I challenge 'Number Se

MysteryTwister C3 (MTC3) is an international crypto challenge contest.

The CrypTool Book (the pdf is for free)

A screenshot of a PDF viewer interface. The top bar includes icons for file operations (cloud, print, email, search), navigation (up, down, first, last, search), page number (1 / 551), zoom (43.3%), and other document controls. The left sidebar is titled "Lesezeichen" (Bookmarks) and lists ten chapters, each with a bookmark icon and a brief description. The main content area displays the title page of the book "The CrypTool Book: Learning and Experiencing Cryptography with CrypTool and SageMath". The title is in large blue serif font. Below it, the subtitle reads "Prof. Bernhard Esslinger and the Development Team of the Open-Source Software CrypTool". At the bottom of the page, it says "Edition 12 (2017)". In the top right corner of the main content area, there is a small whiteboard with handwritten mathematical notes: $2 - 1 = 3$ prim, $2^3 - 1 = 7$ prim, $2^5 - 1 = 31$ prim, $2^7 - 1 = 127$ prim, and $2^{11} - 1 = 2047 = 23 \cdot 89$.

- Introduction – How do the Book and the Programs Play together?
- 1 Security Definitions and Encryption Procedures
- 2 Paper and Pencil Encryption Methods
- 3 Prime Numbers
- 4 Introduction to Elementary Number Theory with Examples
- 5 The Mathematical Ideas behind Modern Cryptography
- 6 Hash Functions and Digital Signatures
- 7 Elliptic Curves
- 8 Introduction to Bitblock and Bitstream Ciphers
- 9 Homomorphic Ciphers
- 10 Survey on Current Academic Results for Solving Discrete Logarithms and for Factoring

The CrypTool Book:
Learning and Experiencing
Cryptography with
CrypTool and SageMath

Prof. Bernhard Esslinger
and the Development Team
of the Open-Source Software CrypTool

Edition 12 (2017)