

## CRYPTOLOGY WITH CRYPTOTool v1.4.30 Beta04

Practical Introduction to  
Cryptography and Cryptanalysis

*Scope, Technology and Future of CrypTool*

[www.cryptool.org](http://www.cryptool.org)  
[www.cryptool.com](http://www.cryptool.com)  
[www.cryptool.de](http://www.cryptool.de)  
[www.cryptool.es](http://www.cryptool.es)  
[www.cryptool.pl](http://www.cryptool.pl)

Prof. Bernhard Esslinger and CrypTool-Team, June 2009

# 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. CrypTool Features

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 (modul N too short)
4. Analysis of encryption in PSION 5
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)

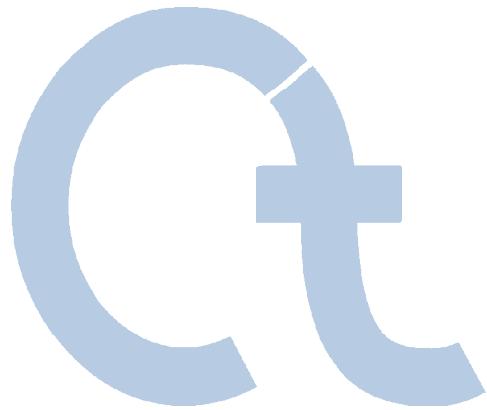
## III. Examples

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 linear modular equation systems)
14. Visualization of symmetric encryption methods using ANIMAL
15. Visualizations of AES
16. Visualization of Enigma encryption
17. Secure E-Mail with S/MIME
18. Generation of a message authentication code (MAC)
19. Hash demonstration
20. Learning 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. CrypTool online help

## IV. Project / Outlook / Contact



# Content



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

# Definition Cryptology and Cryptography

*Cryptology (from the Greek *kryptós*, "hidden," and *lógos*, "word") is the science of secure (generally secret) communications. This security obtains from legitimate users, the transmitter and the receiver, being able to transform information into a cipher by virtue of a key -- i.e., 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 contains not only mechanisms for encryption but also for integrity, electronic signatures, random numbers, secure key exchange, secure containers, electronic voting and electronic money, and has achieved to render a broad range of applications in modern life.*

Source: Britannica ([www.britannica.com](http://www.britannica.com))

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

# Relevance of Cryptography

## Examples for Cryptography Usage

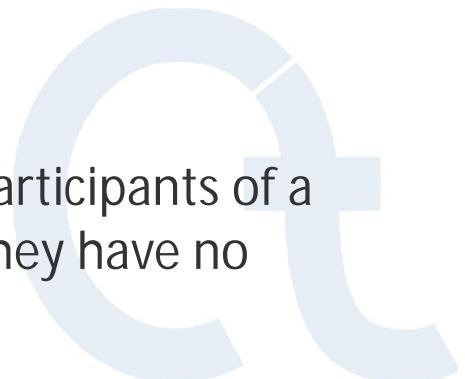
- Phone cards, cell phones, remote controls
- Cash machines, money transfer between banks
- Electronic cash, online banking, secure eMail
- Satellite TV, Pay TV
- Immobiliser systems in cars
- Digital Rights Management (DRM)
- Cryptography is no longer limited to agents, diplomats or the military.  
Cryptography is a modern, mathematically characterised science.
- Breakthrough for cryptography started with the broad use of the Internet
- For companies and governments it is important that systems are secure and

*... users (clients, employees) have a certain  
understanding and awareness for IT security!*

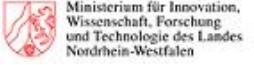
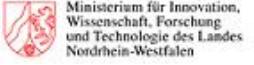


# Cryptography – Objectives

- Confidentiality
  - Information can practically not be made available or disclosed to unauthorized individuals, entities or processes.
- Authentication
  - Authentication ensures that users are identified and those identities are appropriately verified.
- Integrity
  - Integrity ensures that data has not been altered or destroyed in an unauthorized manner.
- Non-Repudiation
  - The principle that, afterwards, it can be proven that the participants of a transaction did really authorize the transaction and that they have no means to deny their participation.



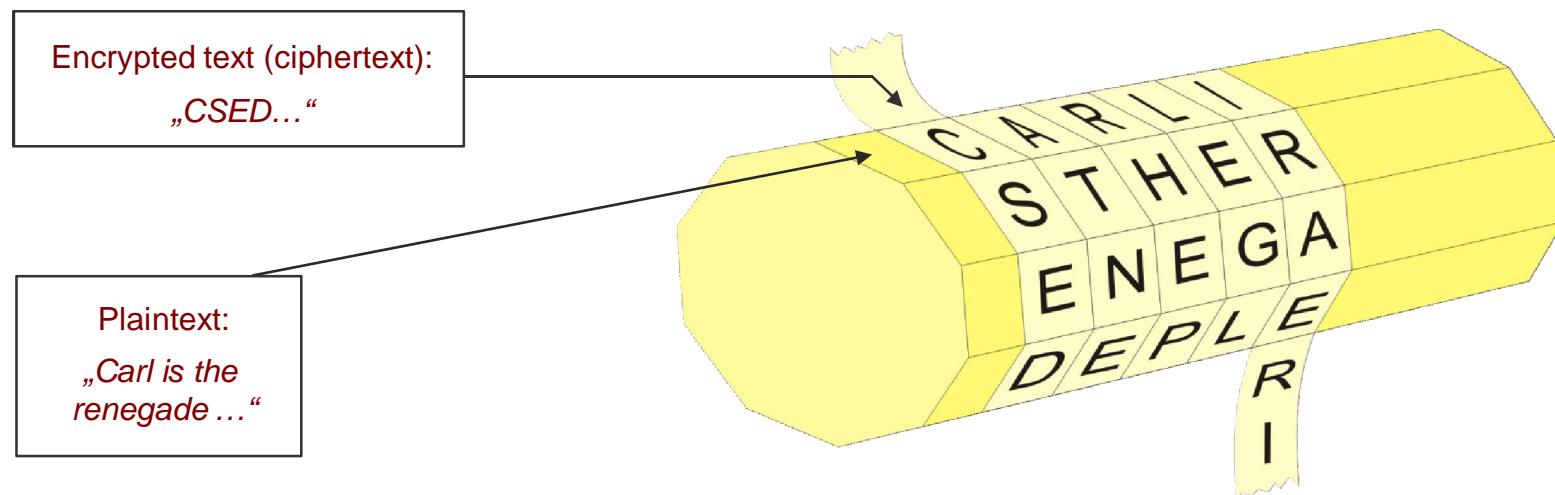
# The CrypTool Project

- Origin in awareness program of a bank (in-firm training)  
→ Awareness for employees
- Developed in co-operation with universities (improving education)  
→ Media didactic approach and standard oriented
  - 1998 Project start – effort more than 17 man-years since then
  - 2000 CrypTool available as freeware
  - 2002 CrypTool on Citizen-CD-ROM from BSI (German Information Security Agency)
  - 2003 CrypTool becomes Open-Source – Hosting by University of Darmstadt (Prof. Eckert)
  - 2007 CrypTool available in German, English, Polish und Spanish
  - 2008 .NET version and Java version – Hosted by University of Duisburg (Prof. Weis) and SourceForge
- Awards
  - 2004 TeleTrusT (TTT Förderpreis) 
  - 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"   
365 Landmarks in the Land of Ideas 2008
- Developers
  - Developed by people from companies and universities in different countries
  - Additional project members or usable sources are always appreciated  
(currently there are around 40 people working on CrypTool world wide).

# Examples of Early Cryptography (1)

## Ancient encryption methods

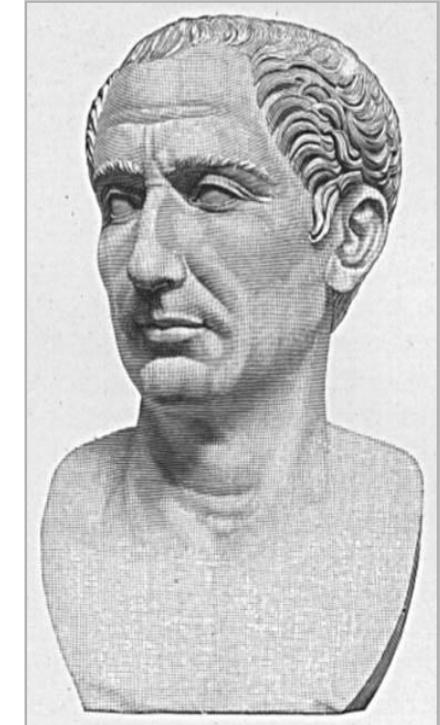
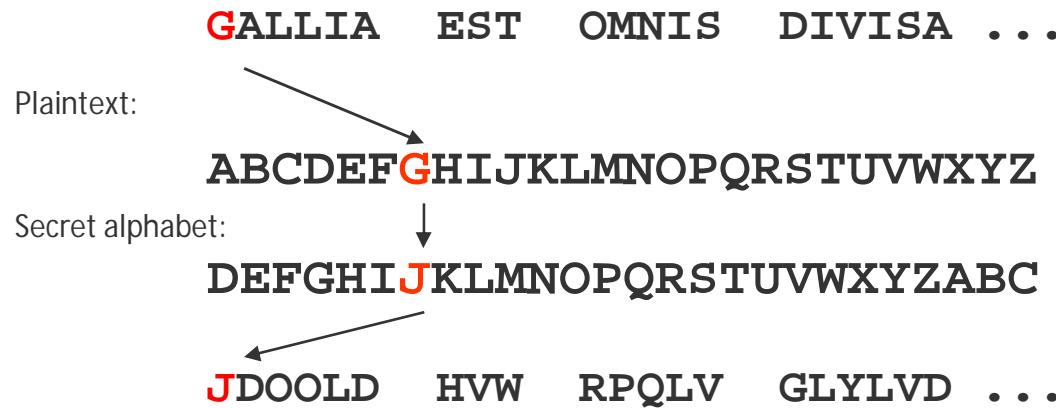
- Tattoo on a slave's head concealed by re-grown hair
- Atbash (around 600 B.C.)
  - Hebrew secret language, reversed alphabet
- Scytale from Sparta (500 B.C.)
  - Described by Greek historian/author Plutarch (45 - 125 B.C.)
  - Two cylinders (wooden rod) with identical diameter
  - Transposition (plaintext characters are re-sorted)



# Examples of Early Cryptography (2)

## Symmetric Caesar encryption

- 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)

## Symmetric Vigenère encryption

- Vigenère-Encryption (Blaise de Vigenère, 1523-1596)
- Encryption with a key word 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 key word character (c). The next plaintext character (I) is replaced by the character in the corresponding row and in the column of the next key word character (h), and so on.
- If all characters of the key word have been used, then the next key word character is the first key character.
- Attack (via Kasiski test): 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.

Keyword character

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

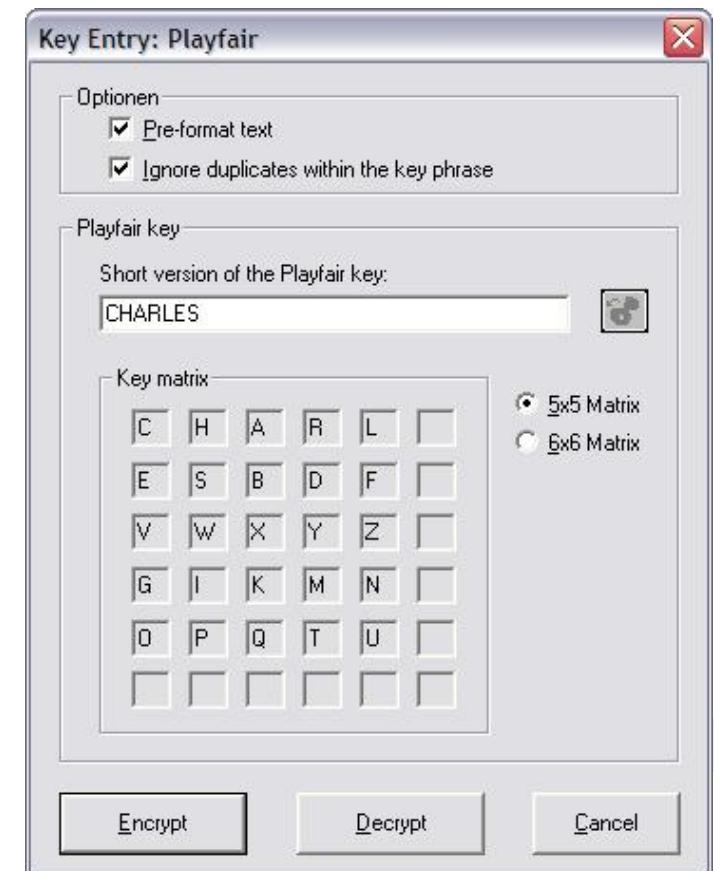
Tableau carré, dit « Carré de Vigenère »

Plaintext character      Encrypted character

# Examples of Early Cryptography (4)

Other symmetric 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)  
(Transposition / very effective)



# Cryptography in Modern Times

Cryptography developments in the last 100 years till 1970

## Classic methods

- are still in use today.  
(since, not everything can be done by a computer...)
- and their principals of transposition and substitution are inputs for the design of modern algorithms:  
combination of simple operation (a type of multiple encryption, a so called cascades of ciphers), on bit level, block cipher, rounds.

## Encryption becomes

- more sophisticated,
- mechanised or computerised and
- remains symmetric.

2003[10]06 → 2003[10]11



Robert Syrett

# Examples of the First Half of the 20th Century

Mechanical encryption machines (rotor machines)

## Enigma Encryption (Arthur Scherbius, 1878-1929)

- More than 200000 machines have been used in WW2
- The rotating cylinder set causes, that every character of the text becomes encrypted with a new permutation.
- Broken by massive effort of cryptography experts (around 7000 persons in UK) with decryption machines, captured original Enigmas and by intercepting daily status reports (e.g. weather reports).
- Consequences of this successful crypto analysis:

*"In general the successful crypto analysis of the engima encryption has been a strategic advantage, that has played a significant role in winning the war. Some historians assume that the break of the enigma code has shortened the war by several months or even a year."*

(translated from [http://de.wikipedia.org/wiki/Enigma\\_Machine](http://de.wikipedia.org/wiki/Enigma_Machine) - March 6, 2006)



# Cryptography – Important Insights (1)

- Kerckhoffs principle (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" (Caesar for example)
    - Kerckhoffs principle:
      - The secret lies within the key and not within the algorithm or „No security through obscurity“
- One-Time Pad – Shannon / Vernam
  - Demonstrably theoretically secure, but not usable in reality (only red phone)
- Shannons concepts: Confusion and Diffusion
  - Relation between M, C and K has to be as complex as possible (M=message, C=cipher, K=key)
  - Every cipher text character should depend on as many plaintext characters and as many character of encryption key
  - „Avalanche effect“(small modification, big impact)
- Trapdoor function (one-way function)
  - Fast in one direction, not in the opposite direction (without secret information)
  - Having the secret the opposite direction works (access to the trapdoor)



# Examples for a Breach of the Kerckhoffs Principle

Secret lies within the key and not within 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 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://wired.lycos.com/news/politics/0,1283,32900,00.html>]



# Sample of a One-Time Pad Adaptation



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



# 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:

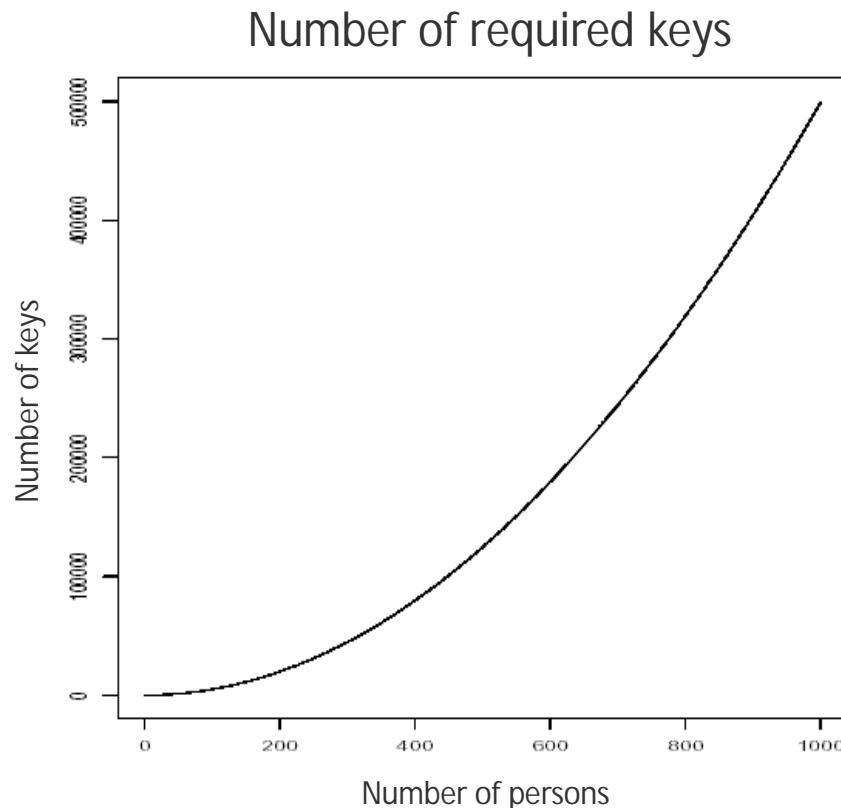
$n = 100$  persons require

$S_{100} = 4.950$  keys; and

$n = 1.000$  persons require

$S_{1000} = 499.500$  keys.

⇒ factor 10 more persons mean  
factor 100 more keys



# Cryptography – Important Insights (2)

Solving the key distribution problem through asymmetric cryptography

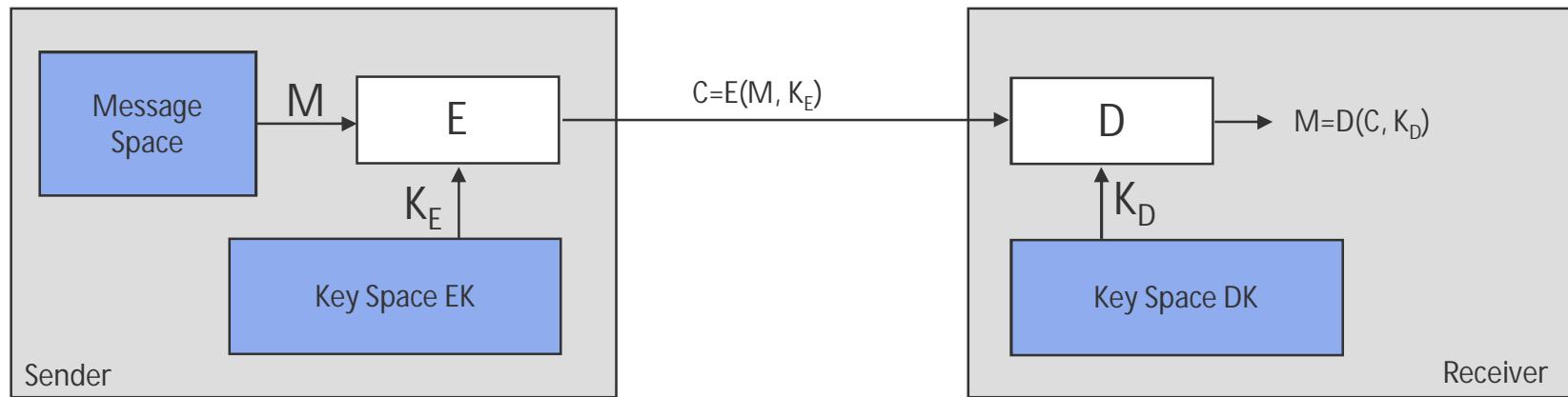
## Asymmetric cryptography

- For centuries it was believed that: Sender and receiver need same secret.
- New: Every member needs a key pair (Solution of the key distribution problem)
- Asymmetric encryption
  - „Everyone can lock a padlock or can 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 (admitted in public 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!*

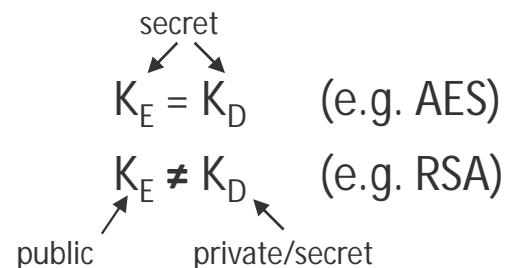
# Encryption and Decryption

Symmetric und asymmetric encryption



a) Symmetric Encryption:

b) Asymmetric Encryption:



# Cryptography – Important Insights (3)

Increasing relevance of mathematics and information technology

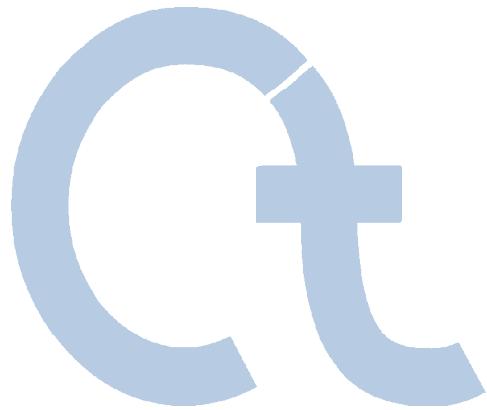
- Modern cryptography is based on mathematics
  - Still new symmetric encryption methods such as AES (better performance and shorter key length compared to the asymmetric methods purely based on mathematical problems).
- The security of encryption methods heavily depends on the current status of mathematics 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-200 (CrypTool script, chapter 4.11.3)
  - Very high activity in current research:  
Factorization, non-parallelizable algorithm (because of quantum computing), better understanding of protocol weaknesses and random generators, ...).
- Serious mistake: “Real mathematics has no effects on the war.”  
(G.H. Hardy, 1940)
- Vendors discover security as an essential purchase criterion.



# 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. CrypTool Features
- III. Examples
- IV. Project / Outlook / Contact

## 1. What is CrypTool?

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

## 2. Why CrypTool?

- Origin in 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
- *Prerequisite:* PC knowledge
- *Preferable:* Interest in mathematics and/or programming



# Content of the Program Package



German, English,  
Polish and Spanish

## CrypTool program

- All functions integrated in a *single* program with consistent graphical interface
- Runs on Win32
- Cryptography libraries from Secude and OpenSSL
- Long integer arithmetic from Miracl and GMP, Lattice base reduction via NTL (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 a lot of program functions (tutorial)

## Script (.pdf file) with background information

- Encryption methods • Prime factorization • Digital signature
- Elliptic curves • Public-key certification • Basic number theory • Crypto 2020 • SAGE

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

- „The Dialogue of the Sisters“ (a RSA variant as key element)
- „The Chinese Labyrinth“ (Numbers theory tasks for Marco Polo)

## 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, ...)
- Solitaire

Several options to easily comprehend cryptography samples from literature:

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

## Cryptanalysis

### Attack on classical methods

- Cipher text only
  - Caesar
  - Vigenère
  - Addition
  - XOR
  - Substitution
  - Playfair
- Known-plaintext
  - Hill
  - Single-column transposition
- Manually (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, ...)
- AES (=Rijndael)
- DESL, DESXL

### Asymmetric encryption

- RSA with X.509 certificates
- RSA demonstration
  - Understanding of examples
  - Alphabet and block length selectable

### Hybrid encryption (RSA + AES)

- Interactive data flow diagram

## Cryptanalysis

### Brute-force attack on symmetric algorithm

- For all algorithms
- Assumptions:
  - Entropy of plaintext is small or key is partly known or plaintext alphabet is known

### Attack on RSA encryption

- Factorization of RSA module
- 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 bit in around 5 min)

### Analysis of random data

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

# Features (4)

## Animation / Demos

- Caesar, Vigenère, Nihilist, DES (all with ANIMAL)
- Enigma (Flash)
- Rijdael/AES (twice with Flash, once 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 (authentication)
- Side-channel attack
- Secure e-mail 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 crypto system (with Authorware)



# Features (5)

## Additional functions

- Homophone and permutation encryption (Double Column Transposition)
- PKCS #12 import and export for PSEs (Personal Security Environment)
- Generate hashes 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
- And a lot more ...

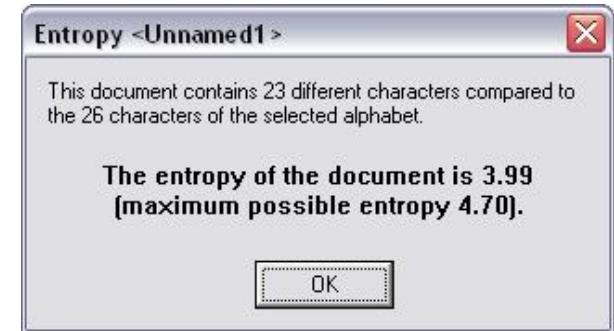
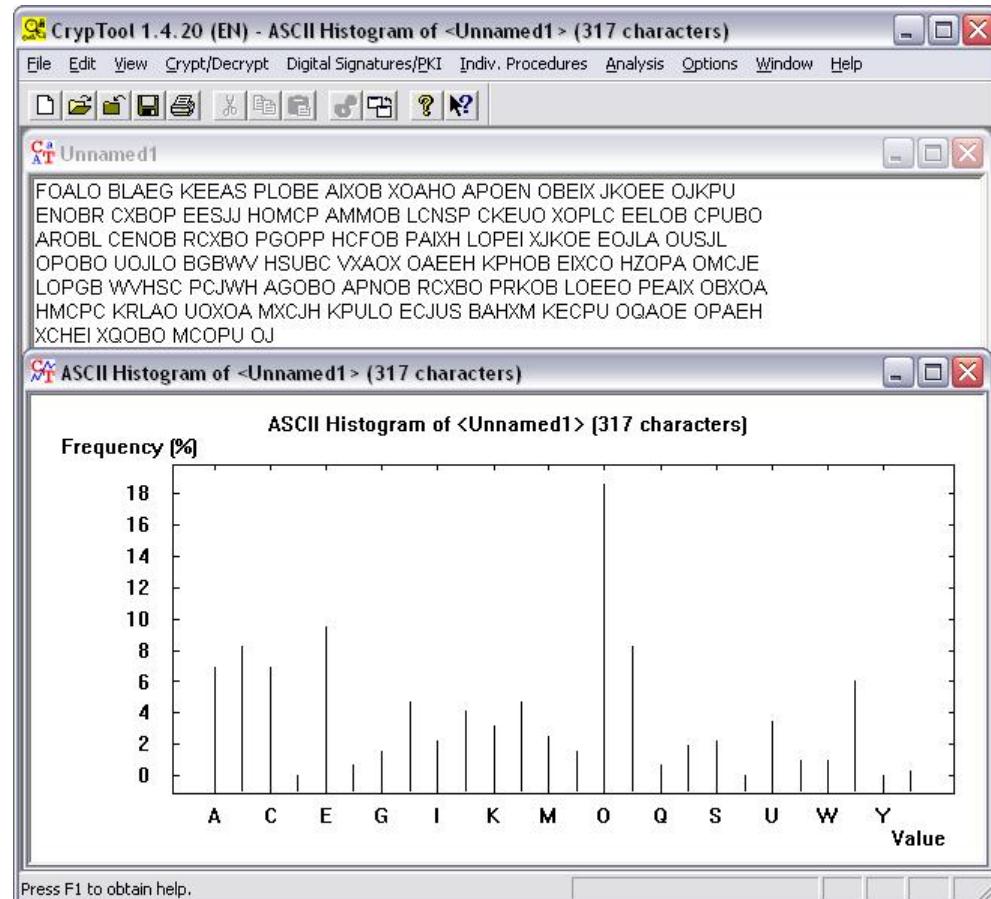


# Language Structure Analysis

Language analysis options available in CrypTool

Number of characters, n-gram, entropy

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



This dialog box is titled "N-Gram List of Unnamed1". It has a "Selection" section with radio buttons for "Histogram", "Digram", "Trigram", and "4-gram", where "Histogram" is selected. Below this is a text input field for "Display of the [26] most common N-grams (allowed values: 1-5000)". There are three buttons at the bottom: "Determine list", "Save list", and "Close". To the right is a table showing the frequency of characters:

No.	Charact...	Frequency in %	Frequency
1	O	18.6120	59
2	E	9.4637	30
3	B	8.2019	26
4	P	8.2019	26
5	A	6.9401	22
6	C	6.9401	22
7	X	5.9937	19
8	H	4.7319	15
9	L	4.7319	15
10	J	4.1009	13
11	U	3.4700	11
12	K	3.1546	10
13	M	2.5237	8
14	I	2.2082	7
15	S	2.2082	7
16	R	1.8927	6
17	G	1.5773	5
18	N	1.5773	5
19	V	0.9464	3
20	W	0.9464	3
21	F	0.6309	2
22	Q	0.6309	2
23	Z	0.3155	1

# 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 starting example with TESTETE

- „Crypt/Decrypt“ \ „Symmetric (classic)“ \ „Vigenère“
- Enter TESTETE ⇒ „Encrypt“

Analysis of the encryption results:

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

## 2. Encrypt starting example with TEST

- „Crypt/Decrypt“ \ „Symmetric (classic)“ \ „Vigenère“
- Enter TEST ⇒ „Encrypt“

Analysis of the encryption results:

- „Analysis“ \ „Symmetric Encryption (classic)“ \ „Ciphertext only“ \ „Vigenère“
- Derived key length 8 – not correct
- 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

- Menu: „Indiv. Procedures“ \ „RSA Cryptosystem“ \ „Factorization of a Number“
- Some methods are executed in parallel (multi-threaded)
- Methods have specific advantages and disadvantages (e.g. some methods can only determine small factors)

Factorization example 1 :

316775895367314538931177095642205088158145887517

48-digit decimal number

=

3 \* 1129 \* 6353 \* 1159777 \* 22383173213963 \* 567102977853788110597

Factorization example 2:

$2^{250} - 1$

75-digit decimal number

=

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

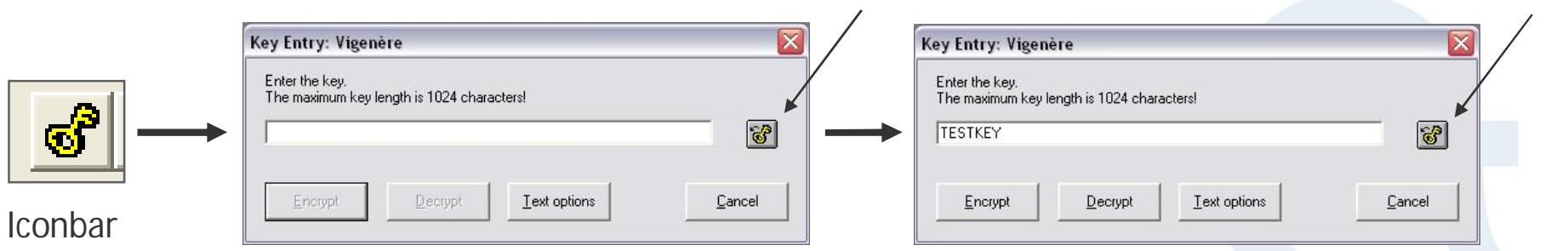
# Concepts for a User-Friendly Interface

## 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 assistances and the contents of the super ordinate menus are cross linked in the online help.

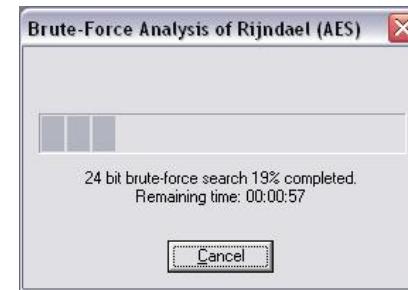
## 2. Paste of keys in key-input dialog

- CTRL-V can be used to paste contents from the clipboard.
- Used keys can be taken out of cipher text windows via an icon in the icon bar. A corresponding icon in the key-input dialog can be used to paste the key into the key field. A CrypTool-internal memory which is available for every method is used (helpful for large „specific“ keys – e.g. homophone encryption).

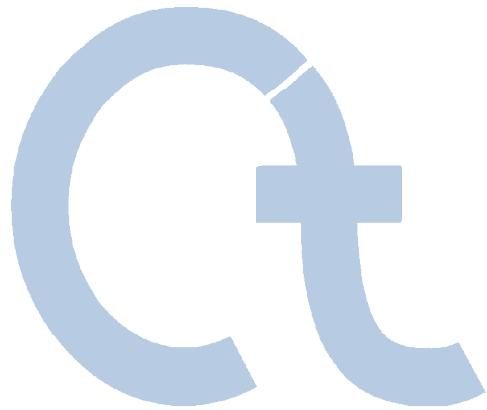


# Challenges for Developers (Examples)

1. Different functions running in parallel
  - Factorization runs with multi-threaded algorithms
2. High performance
  - Locate hash collisions (birthday paradox) or perform brute-force analysis
3. Consider memory limits
  - Floyd algorithm (mappings to locate hash collisions) or factorization with quadratic sieve
4. Time measurement and estimates
  - Display of elapsed time 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. Partly automate the consistency of functions, GUI and online help (including different languages)



# Content



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

# CrypTool Examples

## Overview of examples

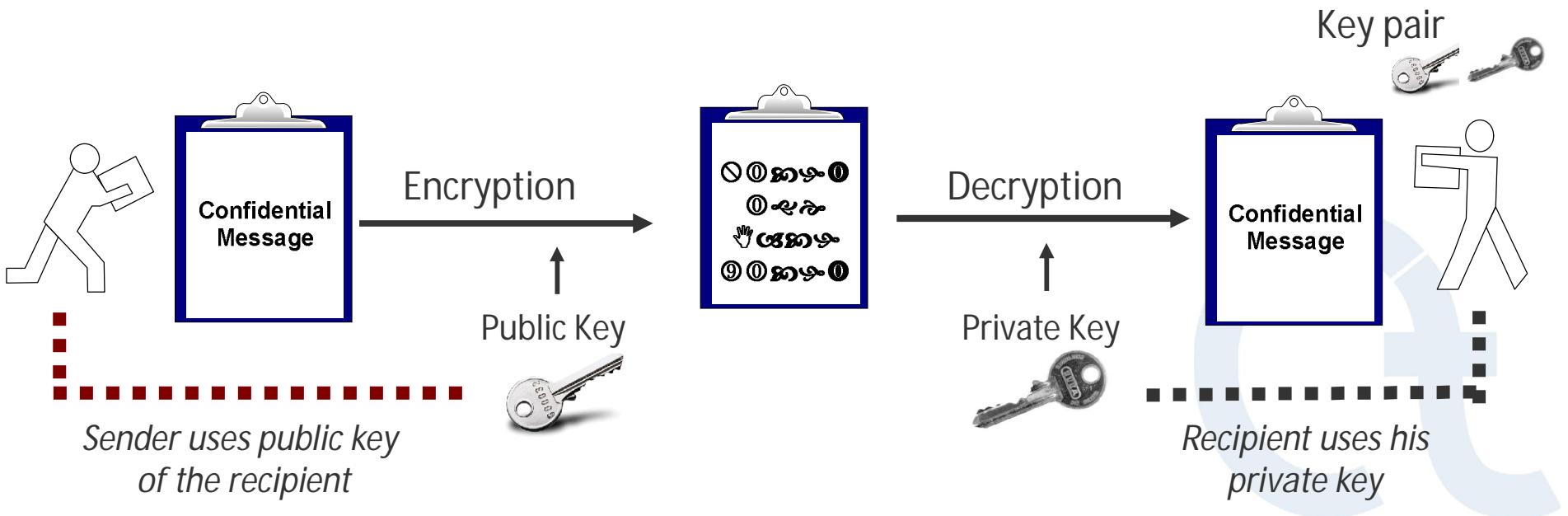
1. [Encryption with RSA / Prime number tests / Hybrid encryption and digital certificates / SSL](#)
2. [Digital signature visualized](#)
3. [Attack on RSA encryption \(modul N too short\)](#)
4. [Analysis of encryption in PSION 5](#)
5. [Weak DES keys](#)
6. [Locating key material \("NSA key"\)](#)
7. [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 \(solving linear modular equation systems\)](#)
14. [Visualization of symmetric encryption methods using ANIMAL](#)
15. [Visualizations of AES](#)
16. [Visualization of Enigma encryption](#)
17. [Secure E-Mail with S/MIME](#)
18. [Generation of a message authentication code \(MAC\)](#)
19. [Hash demonstration](#)
20. [Learning 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. [CrypTool online help](#)



# Examples (1)

Encryption with RSA (in reality mostly hybrid encryption)

- Basis for e.g. SSL protocol (access to protected web sites)
- 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 private key
- Implemented usually in a combination with symmetric methods (transfer of the symmetric key through RSA asymmetric encryption/decryption)



# Examples (1)

Encryption using RSA – Mathematical background / algorithm

- Public key:  $(n, e)$
- Private key:  $(d)$

**where:**

$p, q$  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 module, which length in bits is referred to as RSA key length.
- $\text{gcd}$  = greatest common divisor.
- $\varphi(n)$  is the Euler phi function.

**Procedure :**

- Transformation of message in binary representation
- Encrypt message  $m = m_1, \dots, m_k$  block wise, with for all  $m_j$ :  
 $0 \leq m_j < n$ ; maximum block size  $r$ , so that:  $2^r \leq n$  ( $2^r - 1 < n$ )



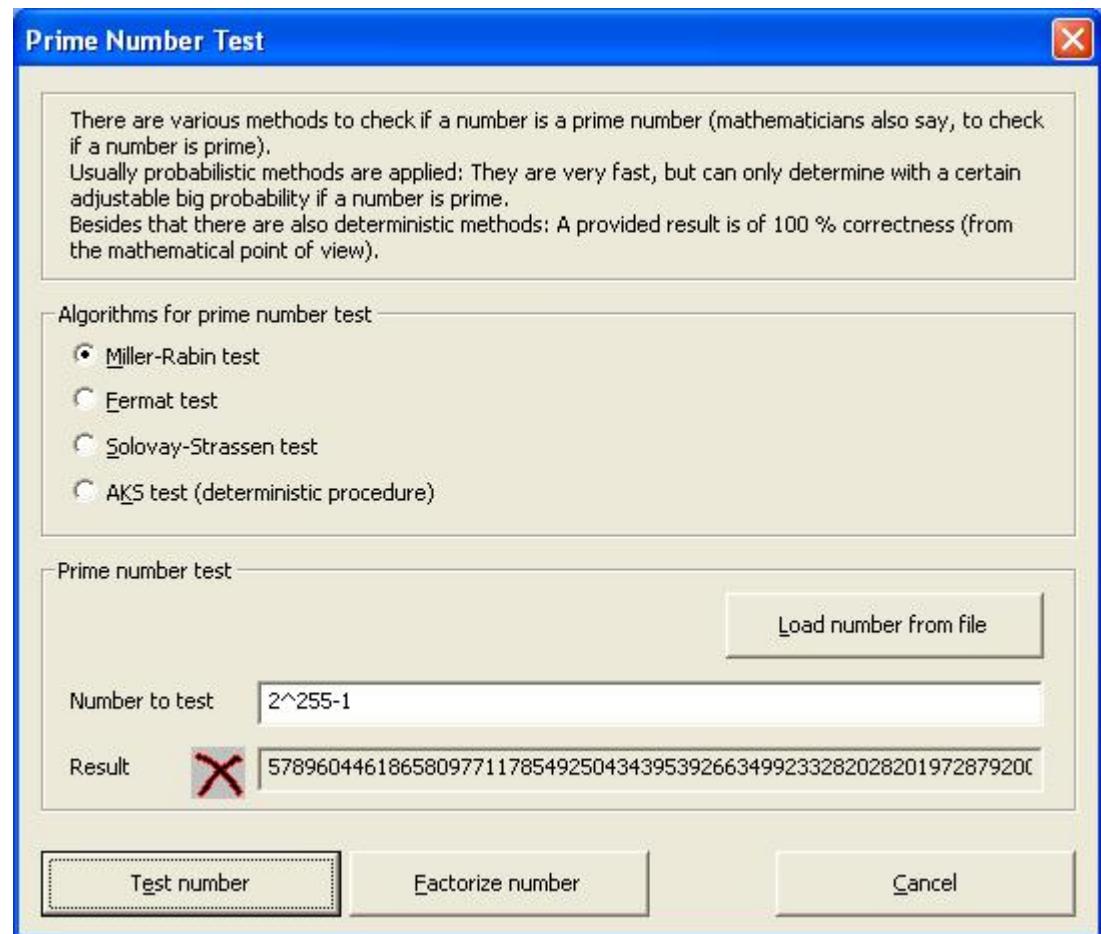
# Examples (1)

Prime number tests – For RSA huge primes are needed

- Fast probabilistic tests
- Deterministic tests

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

For the AKS test the GMP library (**GNU Multiple Precision Arithmetic Library**) was integrated into CrypTool.



Menu: „Indiv. Procedure“ \ „RSA Cryptosystem“ \ „Prime Number Test“

# 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. Telesec, VeriSign, Deutsche Bank PKI), that is being trusted by all users, ensures the authenticity of the certificate and the contained public key (similar to a passport issued by the state).
- 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

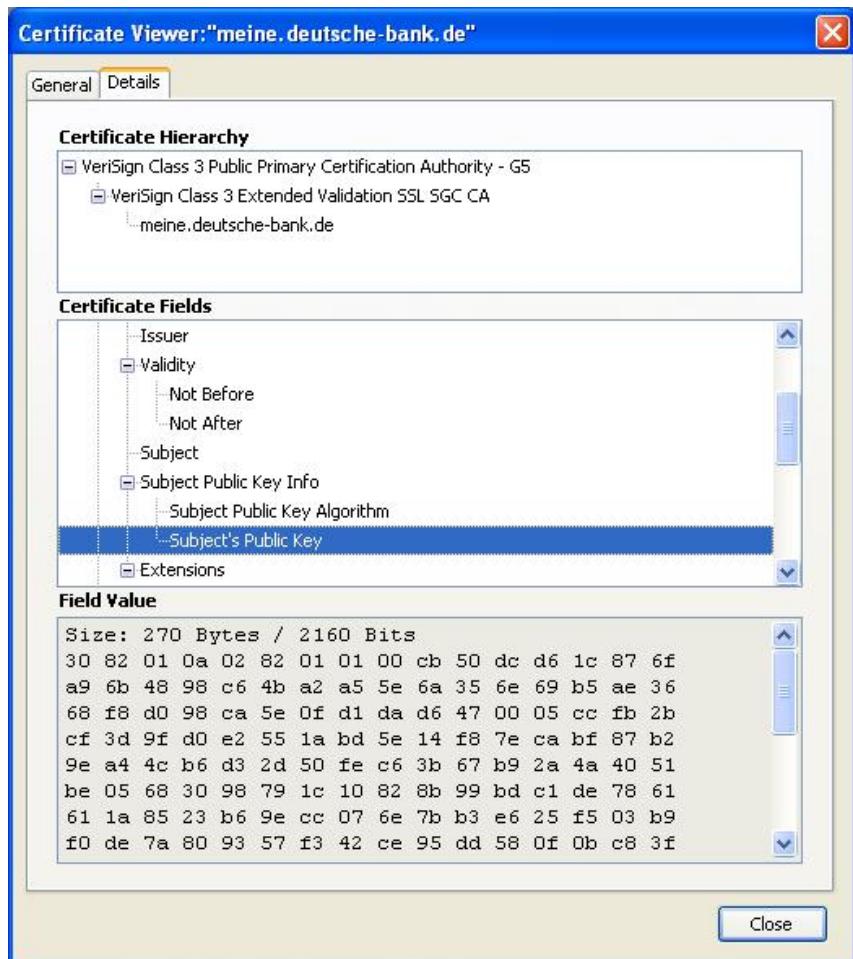
The image shows two windows side-by-side. On the left is a Mozilla Firefox browser window displaying the Deutsche Bank Online Banking login page. The page has fields for 'Branch (three-digit)', 'Account (seven-digit)', 'Sub-account (two-digit)', and 'PIN (five-digit)'. It also includes a dropdown for 'Directly to ...' and a checkbox for 'Session TAN for Brokerage'. A security advisory message at the bottom cautions against phishing attacks. On the right is a 'Certificate Viewer' window for the URL 'meine.deutsche-bank.de'. The 'General' tab is selected, showing that the certificate is verified for 'SSL Client Certificate' and 'SSL Server Certificate'. The 'Issued To' section lists the common name as 'meine.deutsche-bank.de', organization as 'Deutsche Bank AG', and serial number as '3C:16:FE:D8:E8:58:7D:56:4B:EB:F4:11:F6:71:A5'. The 'Issued By' section shows it was issued by 'VeriSign Class 3 Extended Validation SSL SGC CA' from 'VeriSign, Inc.' The 'Validity' section indicates it was issued on 18.09.2007 and expires on 18.09.2008. The 'Fingerprints' section provides SHA1 and MD5 fingerprints.

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

SSL-secured (128 Bit)

# Examples (1)

Attributes or 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)

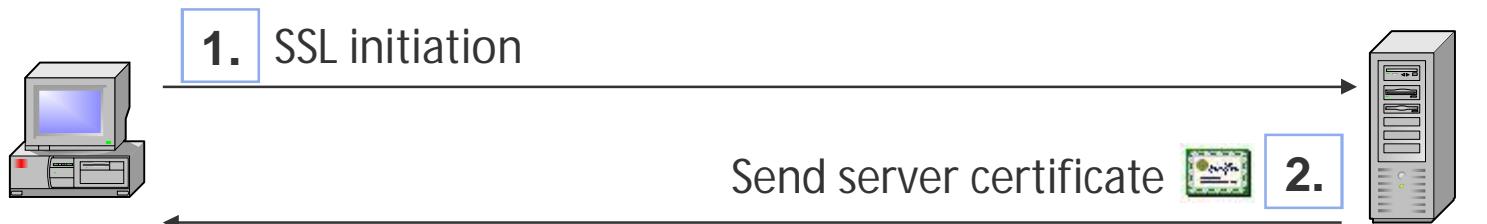
## Public Key



# Examples (1)

Establishing a secure SSL connection (server authentication)

Client



3. Validate server certificate (using locally installed root certificates)

4. Retrieve public key of server (from server certificate)

5. Generate a random symmetric key (session key)

6. Send session key  
(encrypted with public key of server)

Receive session key  
(decrypted by private key of the server)

7.



*Encrypted communication based on  
exchanged session key*



# 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 the 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.

## Comments to the SSL connection establishment

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

# Examples (2)

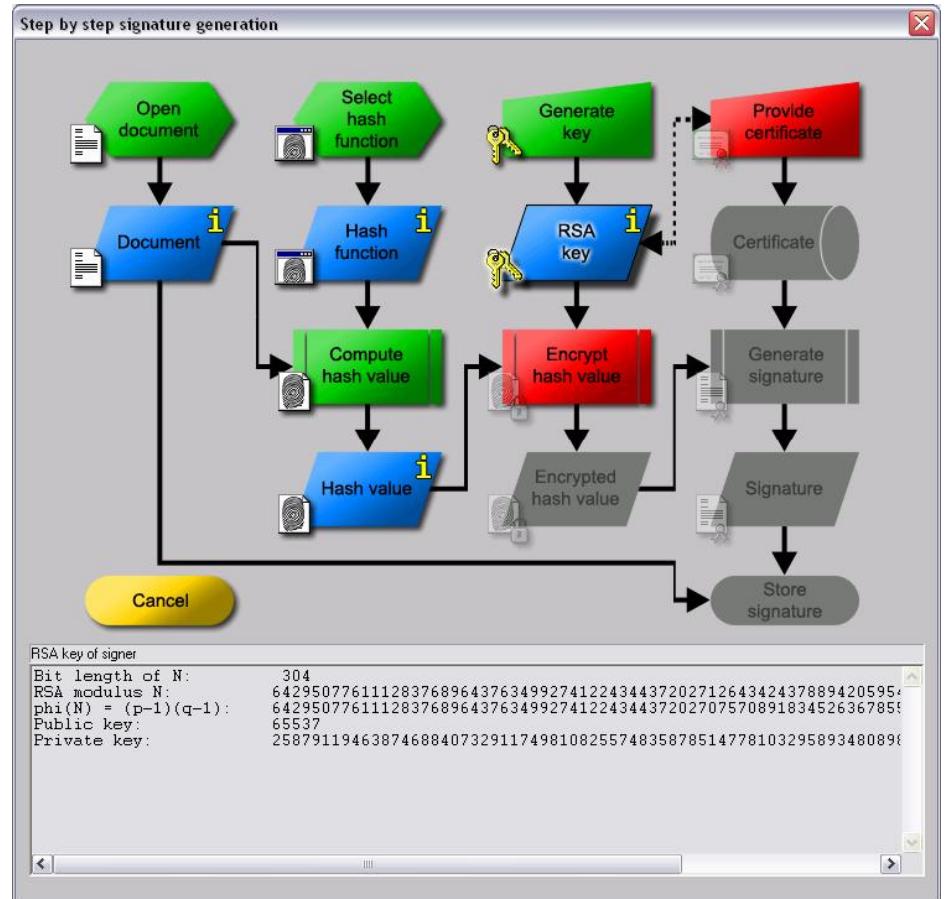
## Digital signature visualized

### Digital signature

- Increasingly important
  - equivalence with manual signature (digital signature law)
  - increasingly used by industry, government and consumers
- Few people know how it works exactly

### Visualization in CrypTool

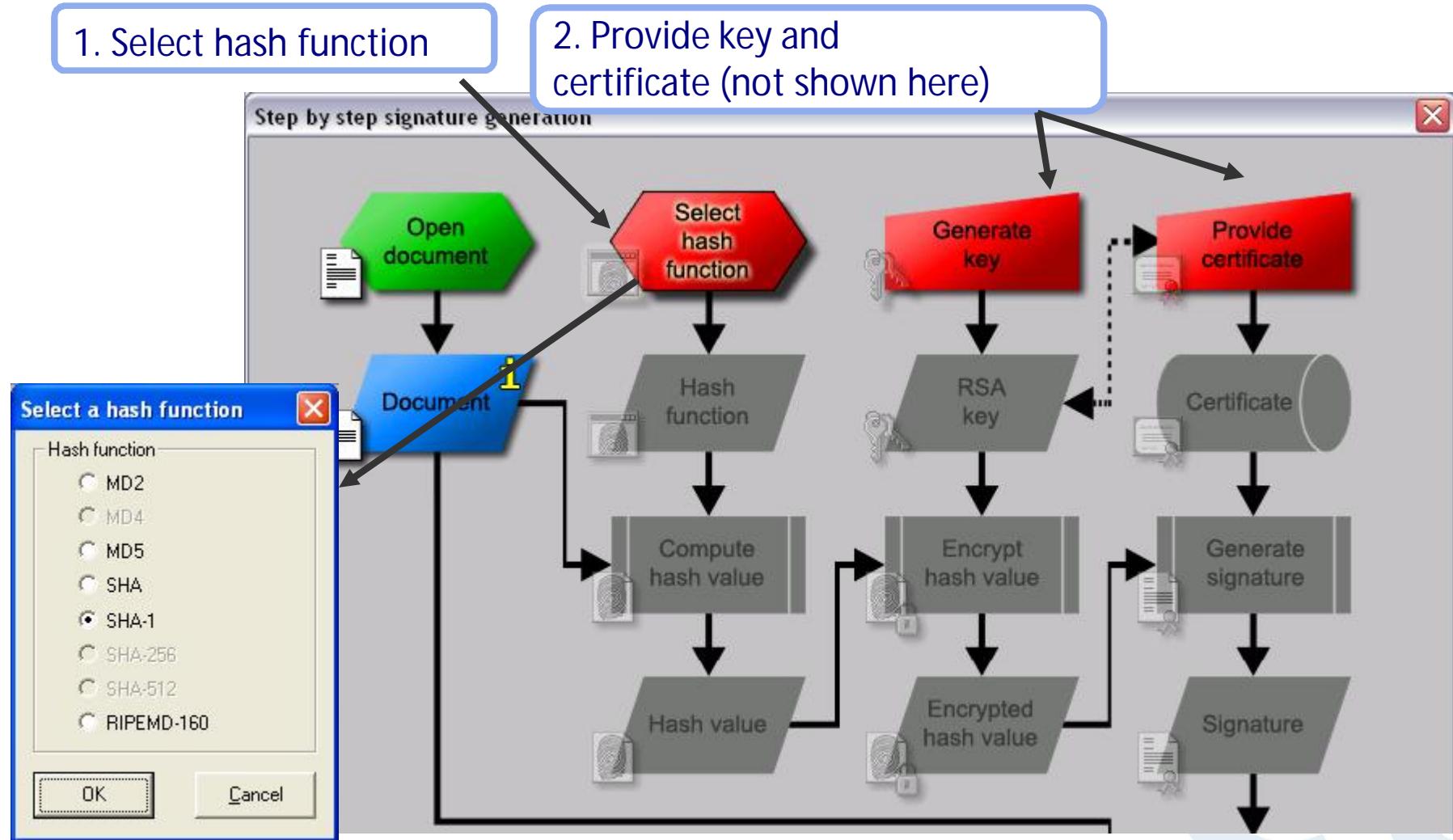
- Interactive data flow diagram
- Similar to the visualization of hybrid encryption



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

# Examples (2)

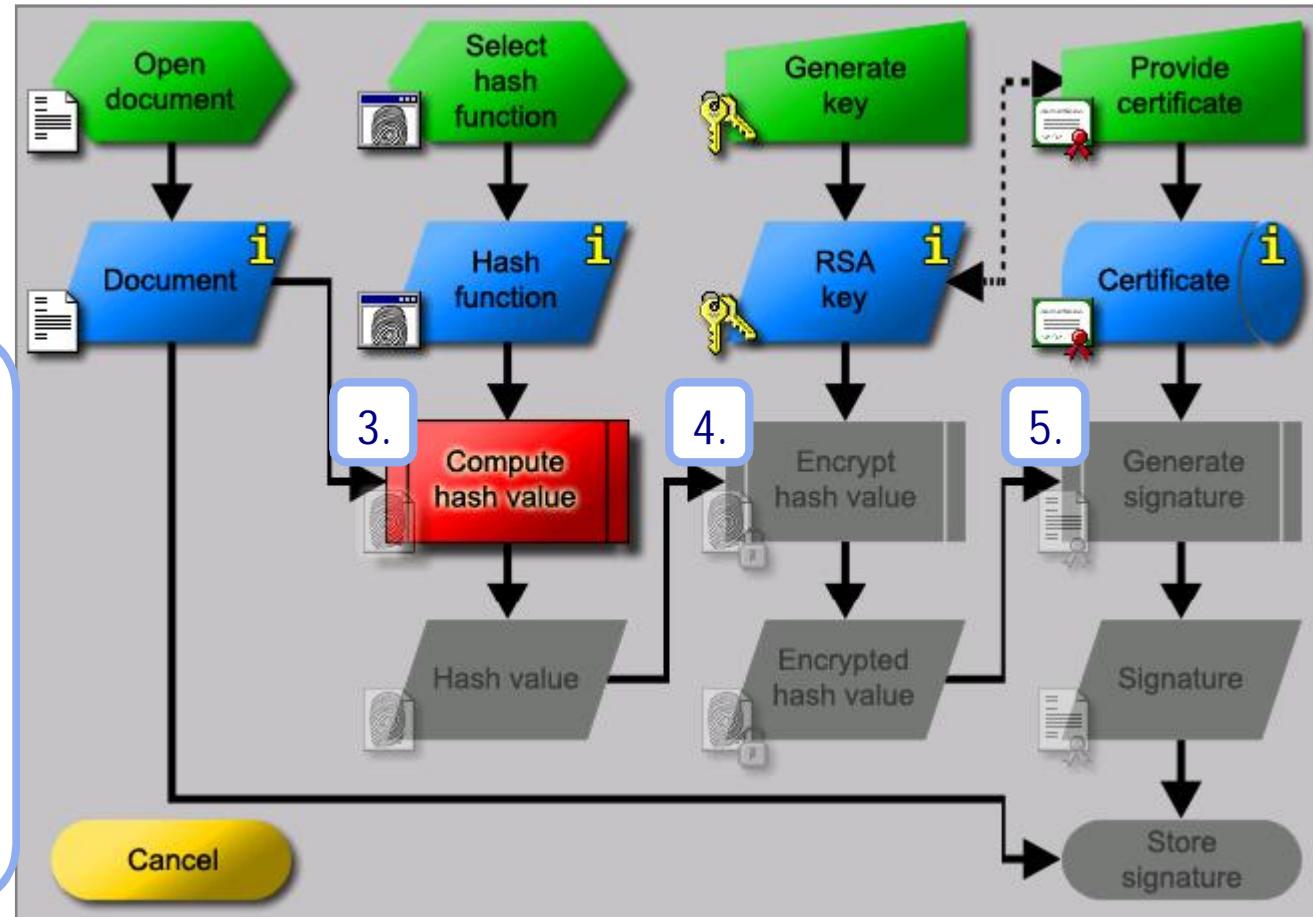
Digital signature visualized: a) Preparation



# Examples (2)

Digital signature visualized: b) Cryptography

- 3. Calculate hash value
- 4. Encrypt hash value with private key (sign)
- 5. Generate signature

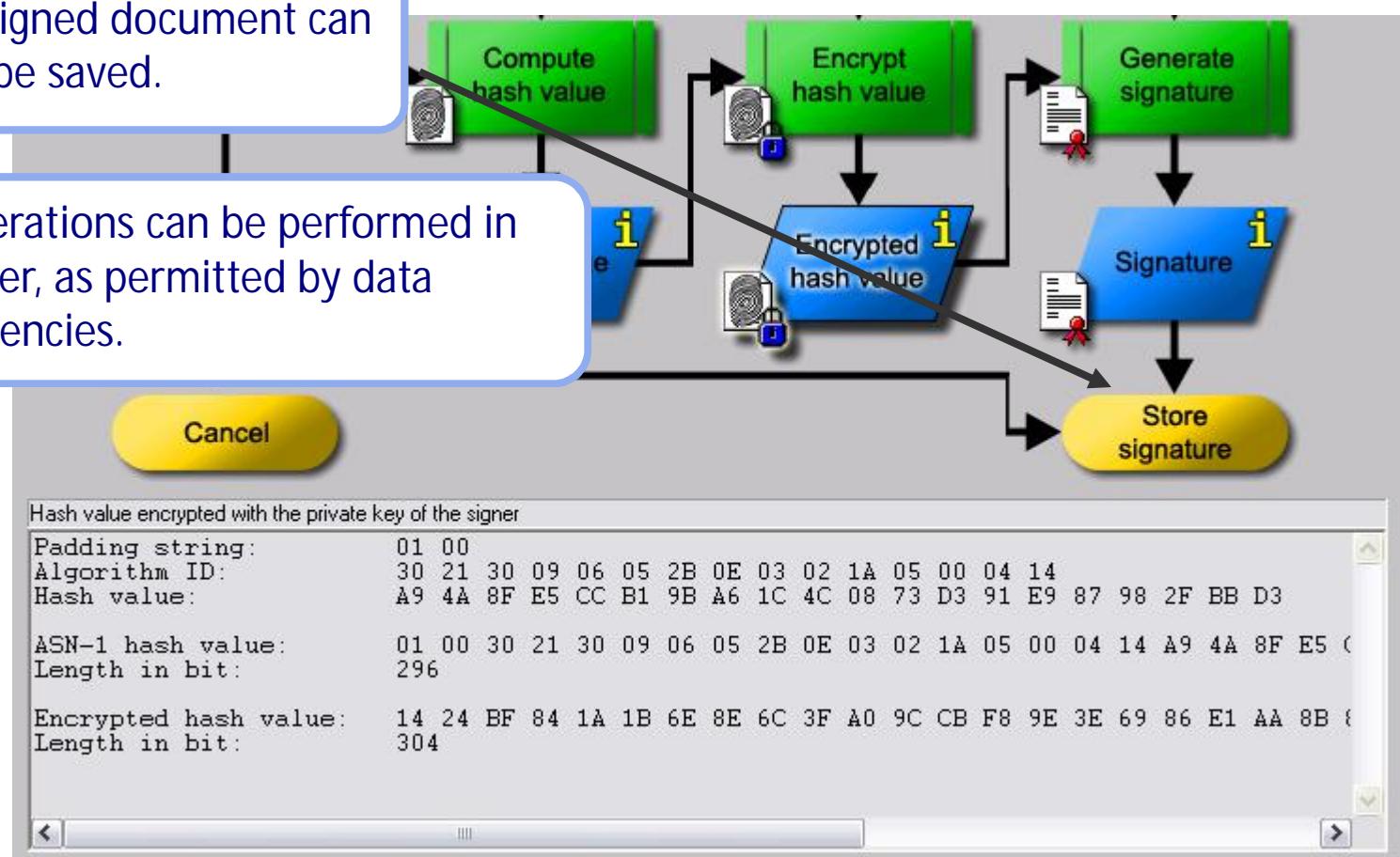


# Examples (2)

Digital signature visualized: c) Result

6. The signed document can now be saved.

The operations can be performed in any order, as permitted by data dependencies.



# 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 bit, 29 decimal digits)
  - public exponent  $e = 17579$

- Ciphertext (block length = 8):

$C_1 = 45411667895024938209259253423,$   
 $C_2 = 16597091621432020076311552201,$   
 $C_3 = 46468979279750354732637631044,$   
 $C_4 = 32870167545903741339819671379$

- The text shall be deciphered!

Solution using CrypTool (more detailed in online help examples section)

- Enter public parameters into "RSA cryptosystem" (menu: „Indiv. Procedures")
- Button "Factorize the RSA modulus" yields the two prime factors  $pq = N$
- Based on that information the private exponent  $d = e^{-1} \bmod (p-1)(q-1)$  is determined
- Decrypt the cipher text with  $d$ :  $M_i = C_i^d \bmod N$

The attack with CrypTool works for RSA moduli up to 250 bit.

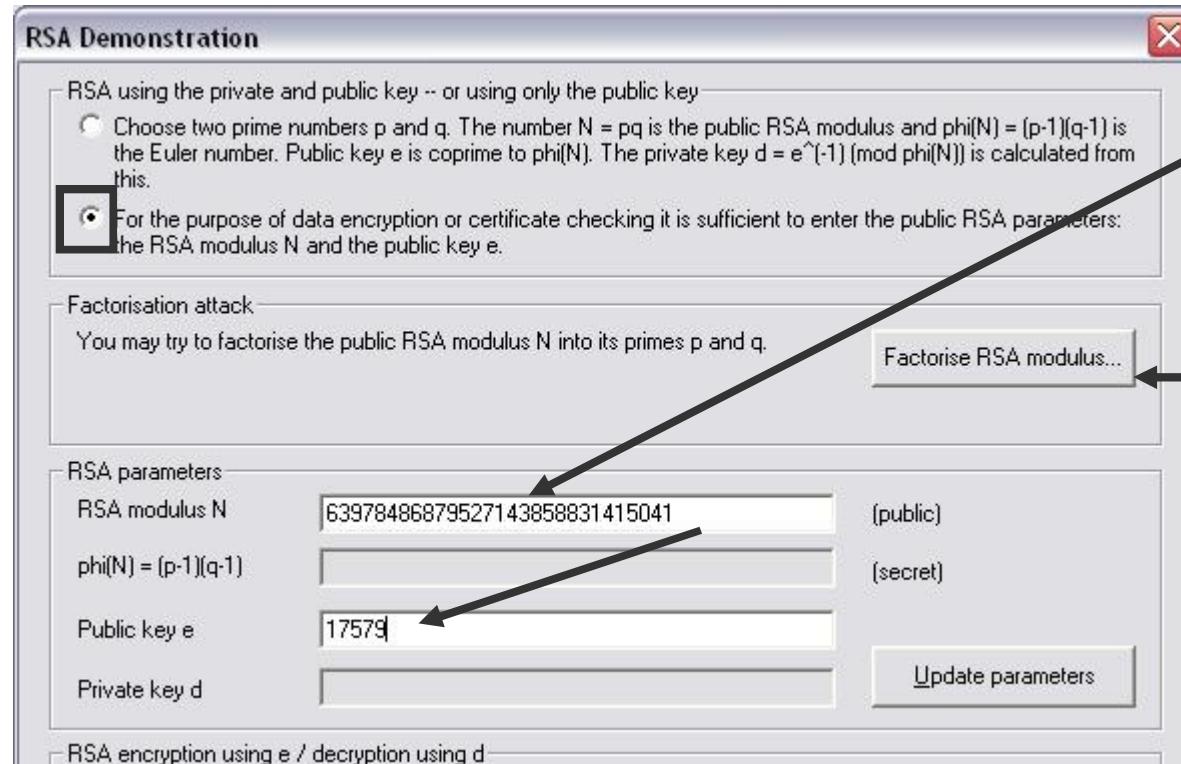
Then you could digitally sign for someone else !

The ciphertext is not  
necessary for the actual  
cryptanalysis  
(locating the private key) !

# Examples (3)

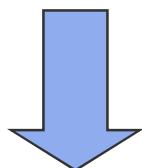
Short RSA modulus: enter public RSA parameters

Menu: „Indiv. Procedures“ \ „RSA Cryptosystem“ \ „RSA Demonstration ...“



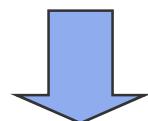
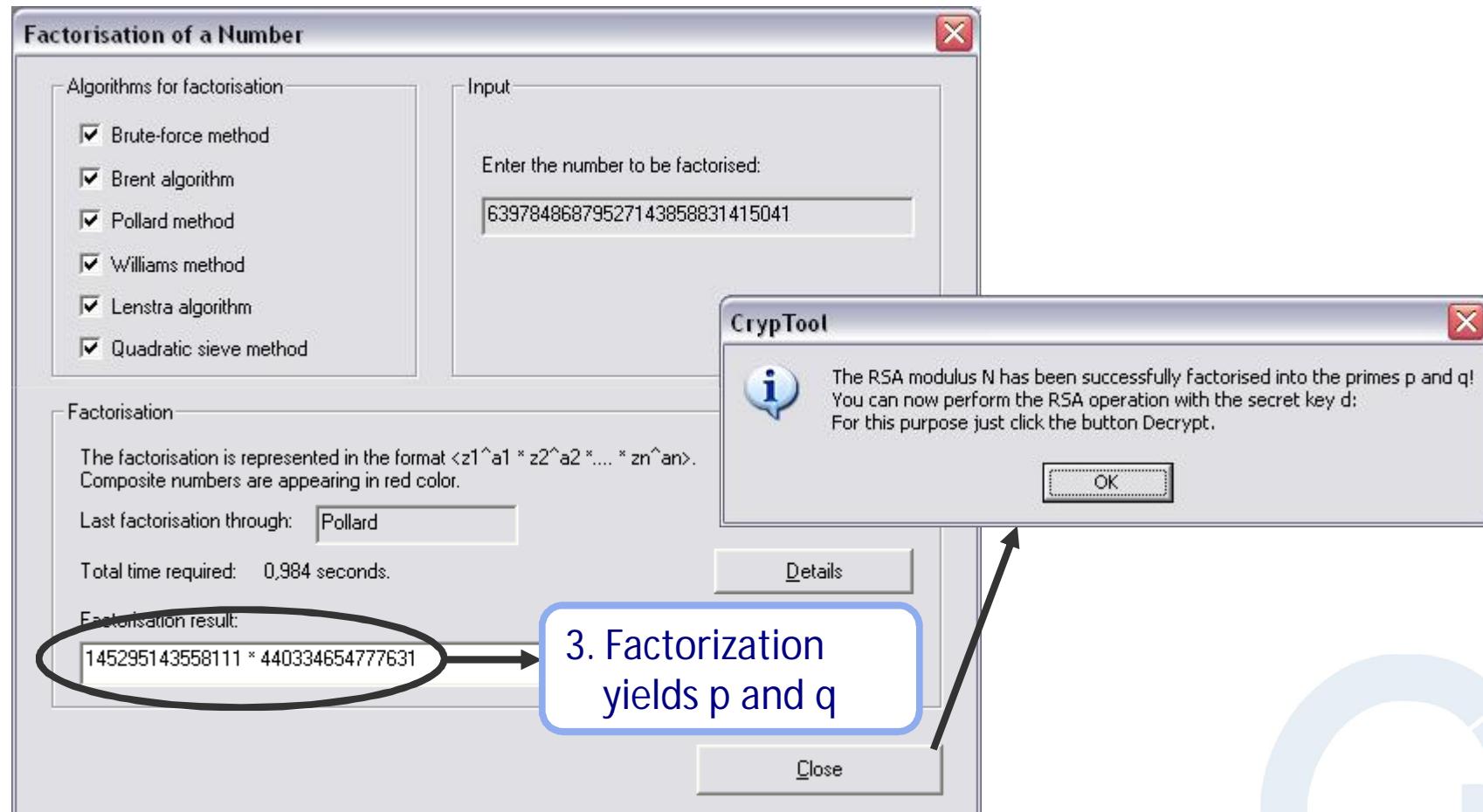
1. Enter RSA parameters N and e

2. Factorize



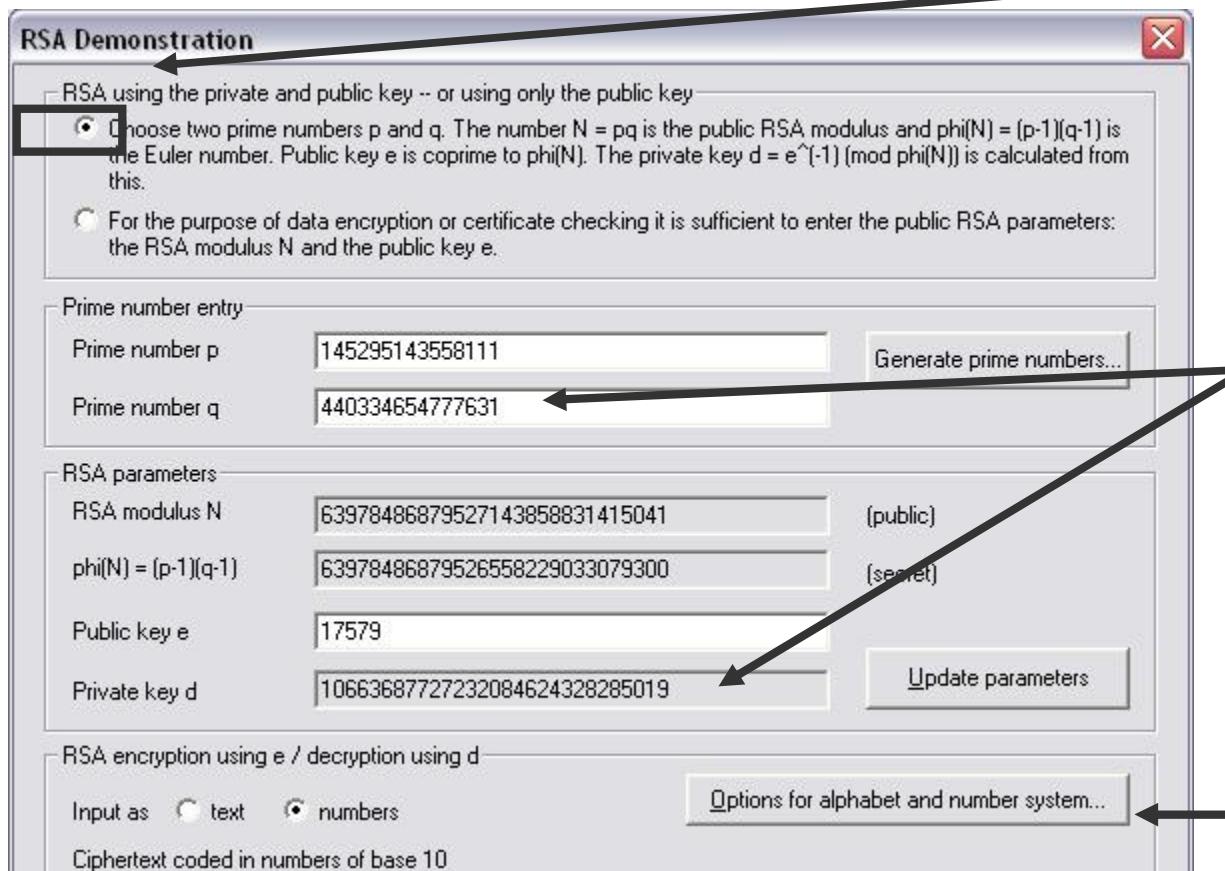
# Examples (3)

Short RSA modulus: factorize RSA modulus



# Examples (3)

Short RSA modulus: determine private key d



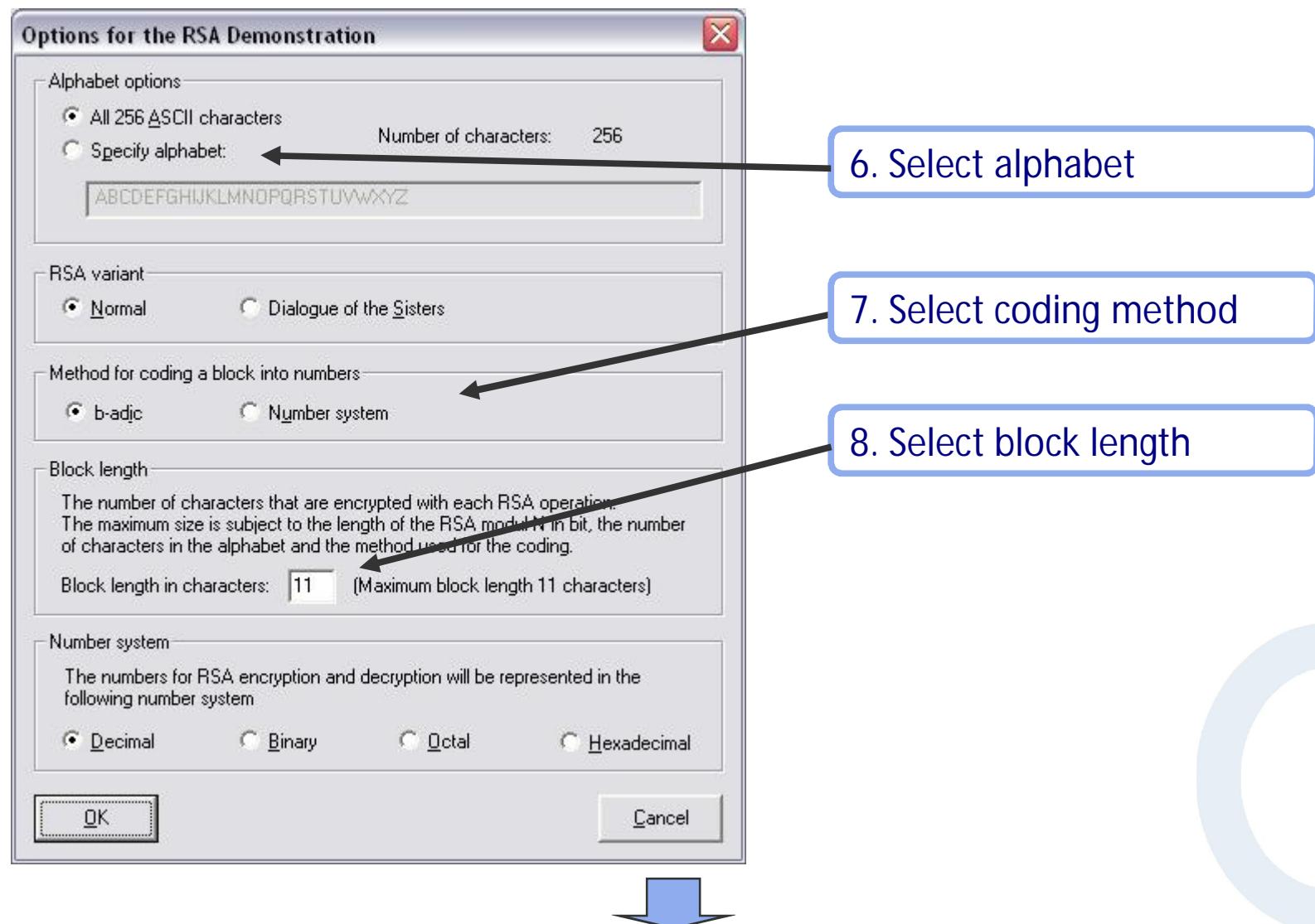
Change the view  
to the owner of the  
secret key

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

5. Adjust options

# Examples (3)

Short RSA modulus: adjust options



# Examples (3)

Short RSA modulus: decrypt cipher text

RSA parameters

RSA modulus N	63978486879527143858831415041	(public)
phi(N) = (p-1)(q-1)	63978486879526558229033079300	(secret)
Public key e	17579	
Private key d	10663687727232084624328285019	<a href="#">Update parameters</a>

RSA encryption using e / decryption using d

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

Ciphertext coded in numbers of base 10

```
5069057070940529666287522 # 40486038748314205283477353228 # 26906996968026845590696474185
```

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

```
00094604723425878030697780557 # 00080116466004756901239213377 # 0008253339409781111818054
```

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

```
NATURAL NUM # BERS ARE MA # DE BY GOD
```

Plaintext

```
NATURAL NUMBERS ARE MADE BY GOD
```

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

9. Enter cipher text

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, 100 bytes up to several kB of text

### Procedure

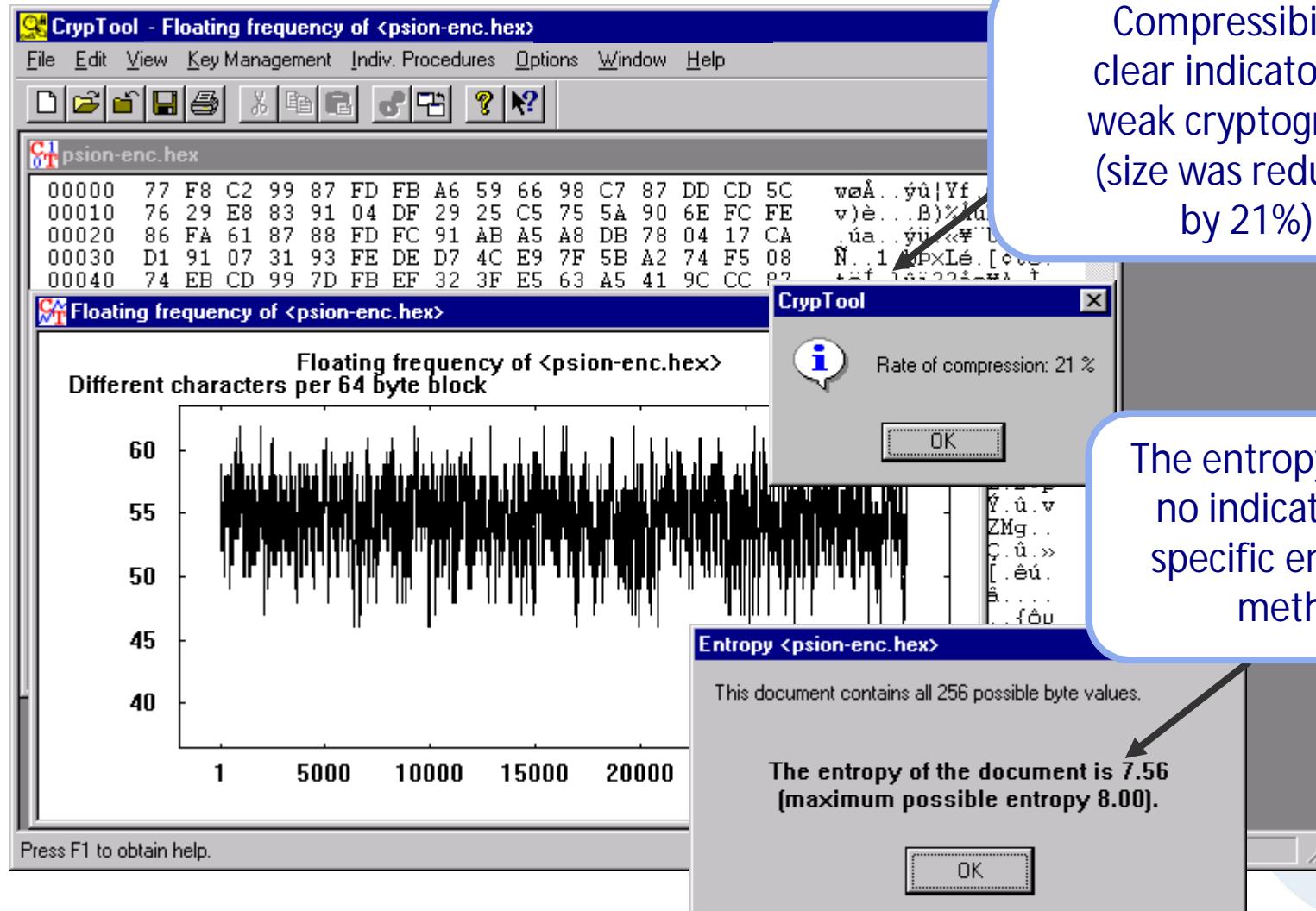
- Pre-analysis
  - entropy
  - floating entropy
  - compression test
- Auto-correlation
- Try out automatic analysis with classical methods

} *probably classical  
encryption algorithm*



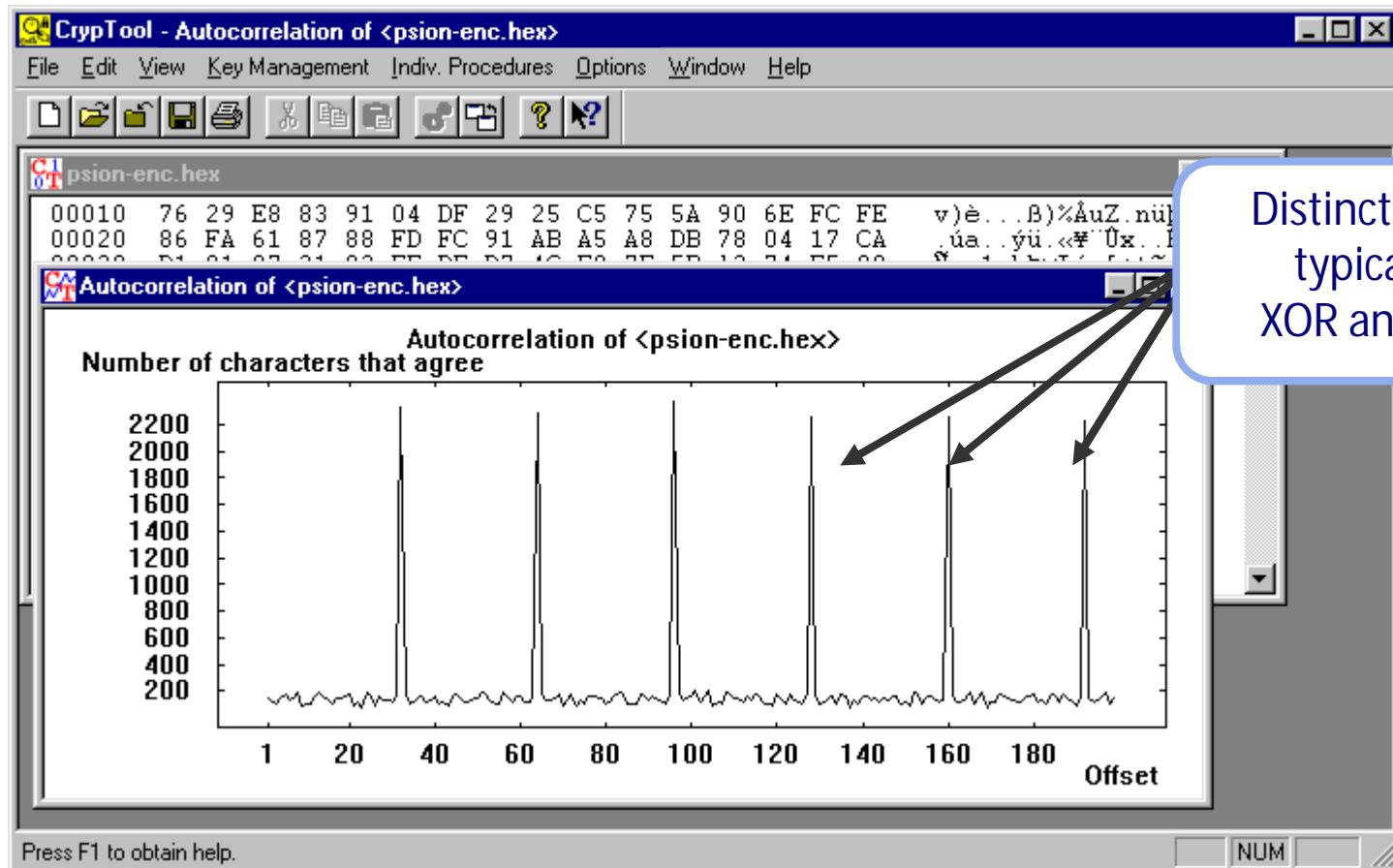
# Examples (4)

PSION 5 PDA – determine entropy, compression test



# Examples (4)

PSION 5 PDA – determine auto-correlation



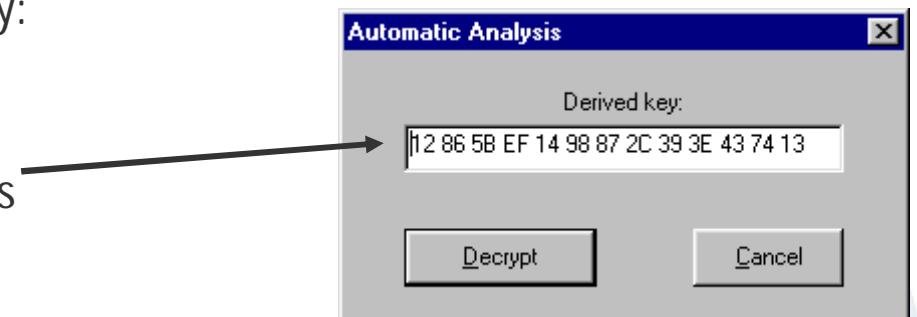
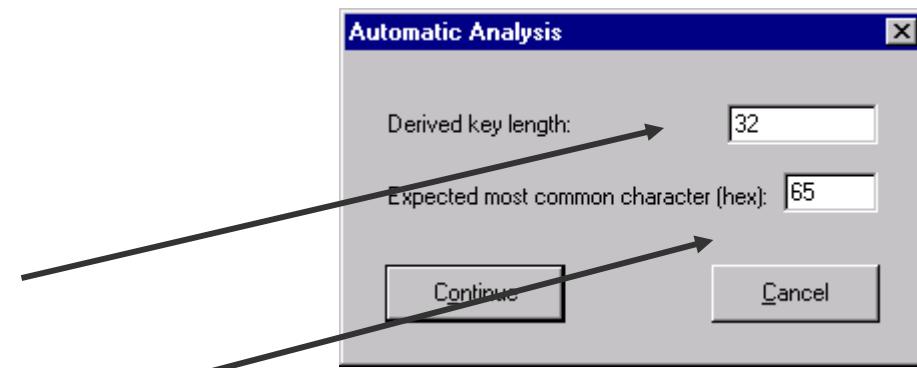
\* The encrypted file is available with CrypTool (see CrypTool\examples\psion-enc.hex)

# Examples (4)

PSION 5 PDA – automatic analysis

## Automatic analysis using

- Vigenère: no success
- XOR: no success
- binary addition
  - CrypTool calculates the key length using auto-correlation: 32 bytes
  - The user can choose which character is expected to occur most frequently: "e" = 0x65 (ASCII code)
  - Analysis calculates the most likely key (based on the assumptions about distribution)
  - Result: good, but not perfect



# Examples (4)

PSION 5 PDA – results of automatic analysis

Results of automatic analysis with assumption “binary addition”:

- Result is good, but not perfect: 24 out of 32 key bytes correct.
- The key length 32 was correctly determined.

Automatic Addition Analysis of <psion-enc.hex>, key: <12 86 5B EF 14 98 87 2C 39 3E 43 74 13 ...>																
00000	65	72	67	AA	73	65	74	7A	20	28	55	53	74	47	29	06
00010	06	06	8A	72	73	74	65	72	65	41	62	B8	A8	68	6E	AE
00020	74	74	06	98	74	65	75	65	72	67	65	67	65	6E	73	74
00030	61	6E	A9	20	75	6E	64	20	8C	65	6C	B9	BA	6E	67	B8
00040	62	65	72	AA	69	63	68	06	06	A7	20	31	2E	06	28	31
00050	29	20	89	65	72	20	55	6D	B8	61	74	BF	B8	74	65	BA
00060	65	72	20	BA	6E	74	65	72	6C	69	65	67	65	6E	20	64
00070	69	65	65	66	6F	6C	67	65	B3	64	65	B3	65	55	6D	B8
00080	E4	74	7A	AA	3A	06	31	2E	20	64	69	65	20	4C	69	65
00090	66	65	B7	75	6E	67	65	6E	65	75	6E	A9	65	73	6F	B3
000A0	73	74	69	AC	65	6E	20	4C	65	69	73	74	75	6E	67	65
000B0	6E	2C	65	64	69	65	20	65	AE	6E	20	9A	B3	74	65	B7
000C0	6E	65	68	B2	65	72	20	69	6D	20	49	6E	6C	61	6E	64
000D0	20	67	AA	67	65	6E	20	45	B3	74	67	AA	B1	74	20	AE
000E0	6D	20	52	A6	68	6D	65	6E	20	73	65	69	6E	65	73	20
000F0	55	6E	B9	65	72	6E	65	68	B2	65	6E	B8	65	61	75	B8

- 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

Copy key to clipboard during automatic analysis

In automatic analysis hex dump,

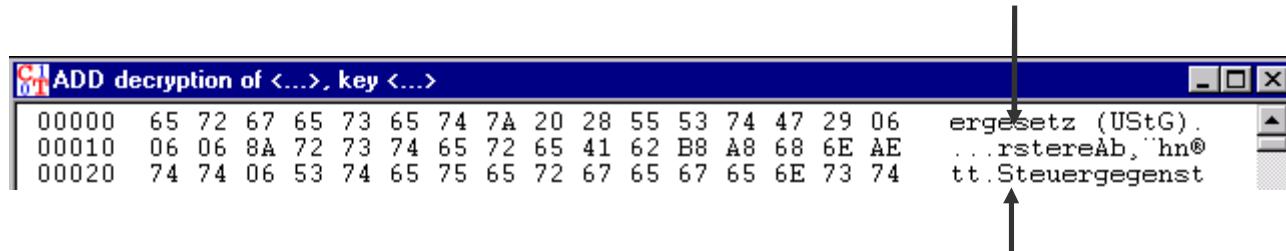
- Determine incorrect byte positions, e.g. 0xAA at position 3
- Guess and write down corresponding correct bytes: „e“ = 0x65

In encrypted initial file hex dump,

- Determine initial bytes from the calculated byte positions: 0x99
- Calculate correct key bytes with CALC.EXE:  $0x99 - 0x65 = 0x34$

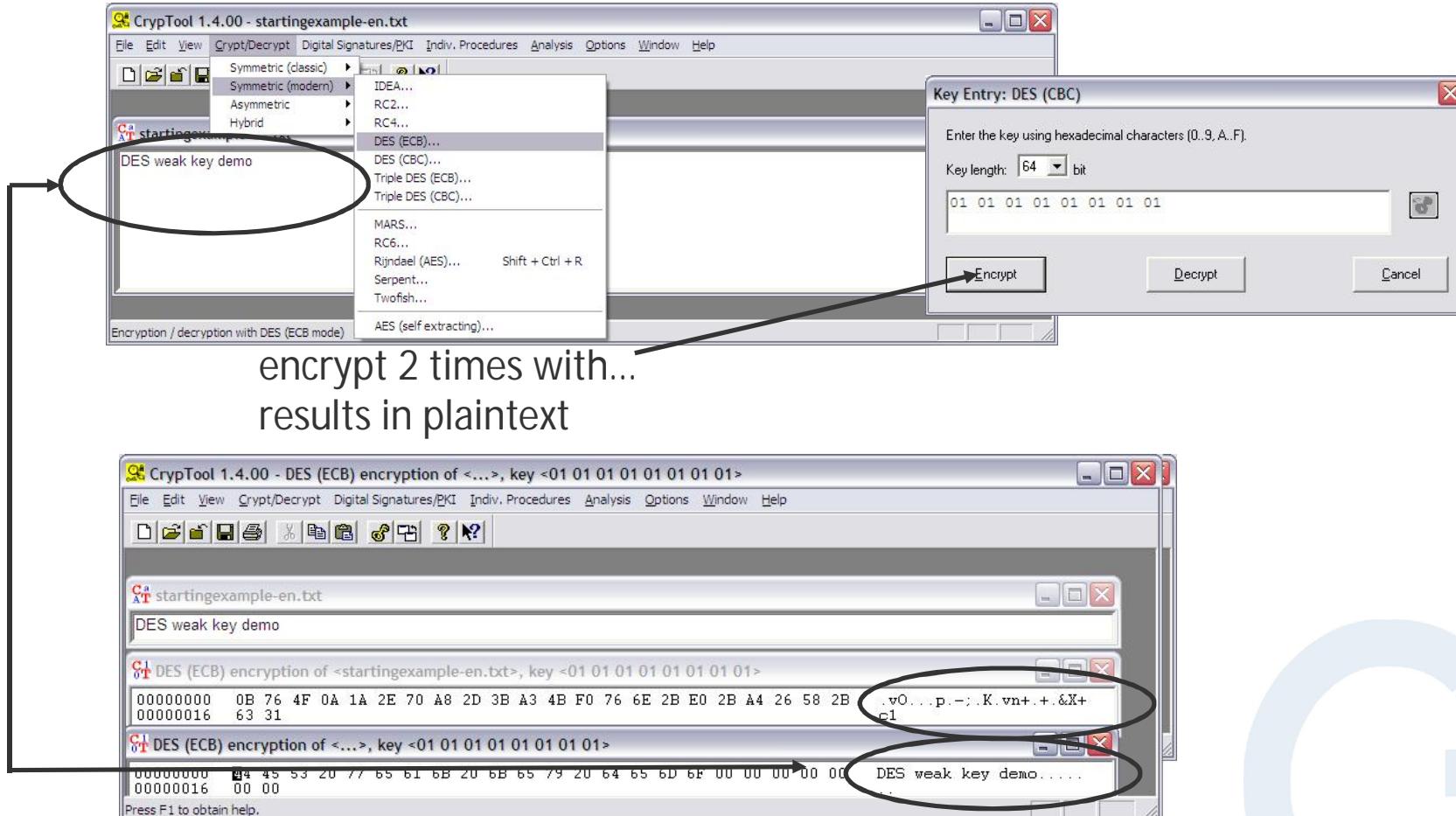
Key from the clipboard

- Correct 12865B341498872C393E43741396A45670235E111E907AB7C0841...
- Decrypt encrypted initial document using binary addition
- Bytes at position 3, 3+32, 3+2\*32, ... are now correct



# Examples (5)

## Weak DES key



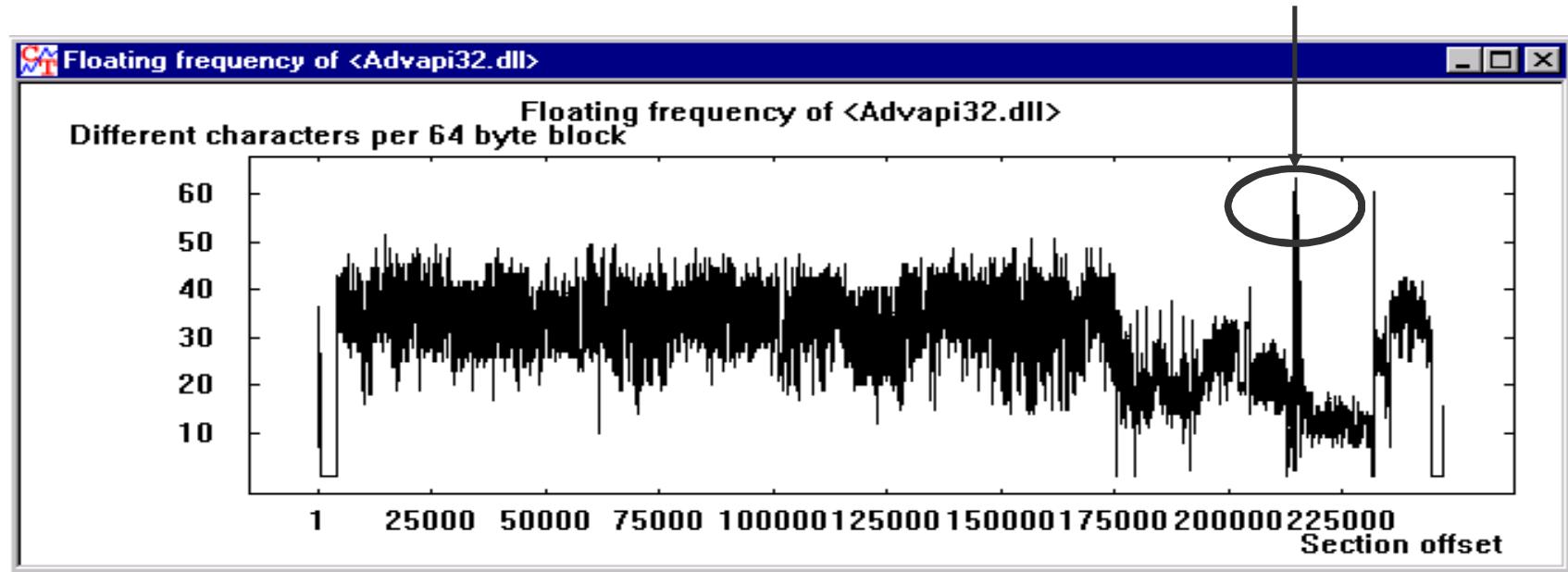
# Examples (6)

Locate key material

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

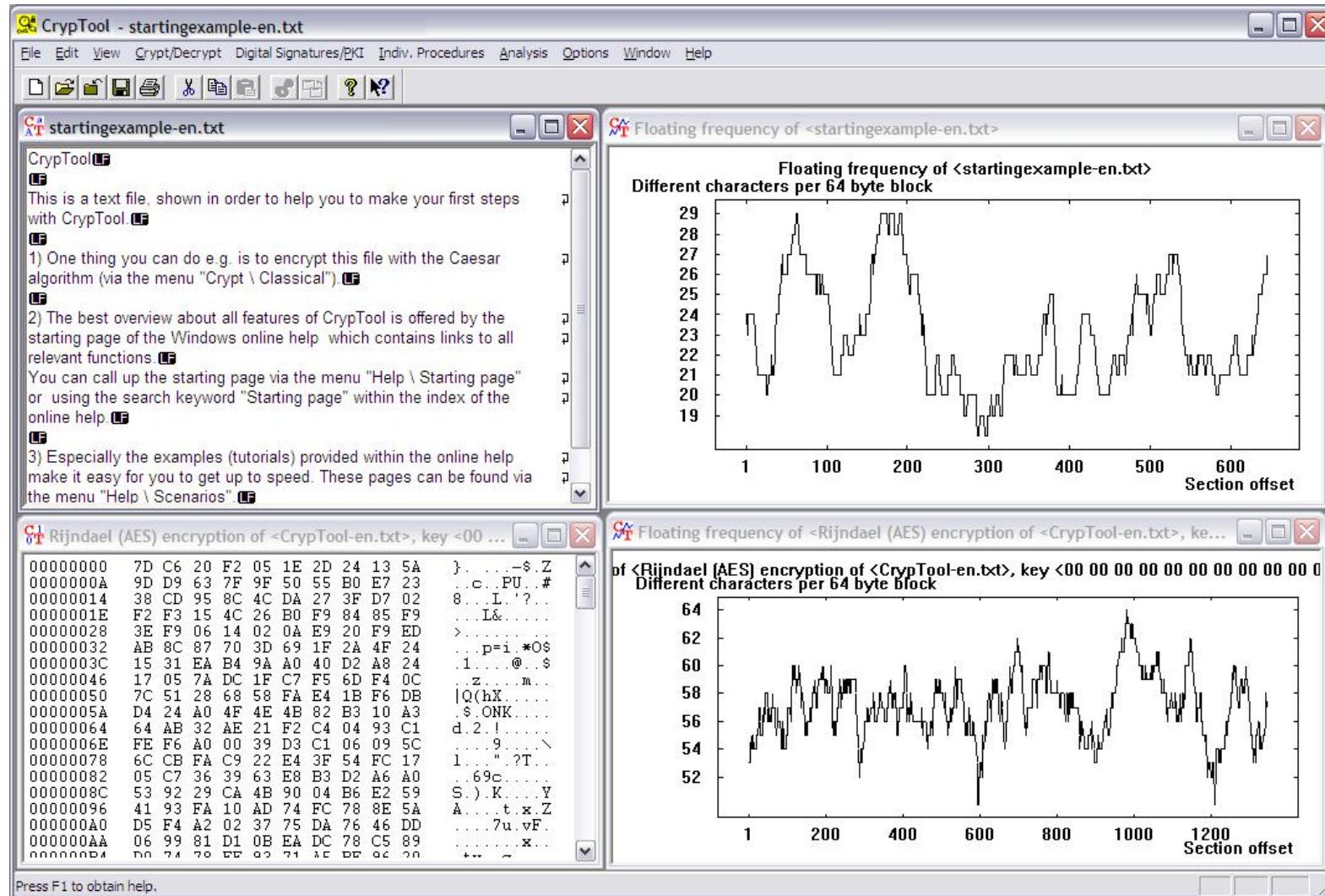
Background:

- 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)



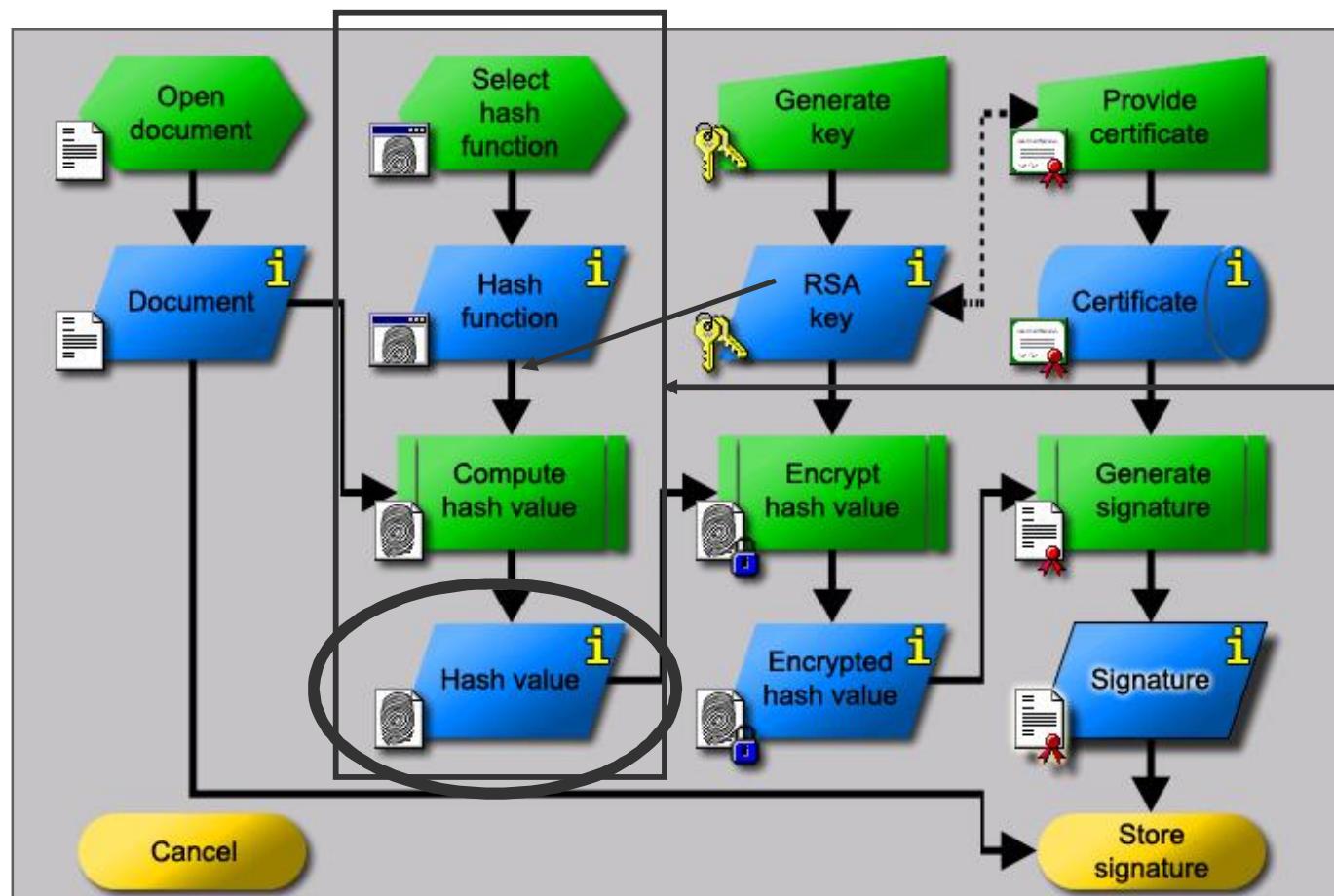
# Examples (6)

Comparison on floating frequency with other files



# Examples (7)

## Attack on digital signature



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 based on 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 be run in parallel on many machines (not implemented)
- Implemented in CrypTool as part of the bachelor thesis "*Methods and Tools for Attacks on Digital Signatures*" (German), 2003.

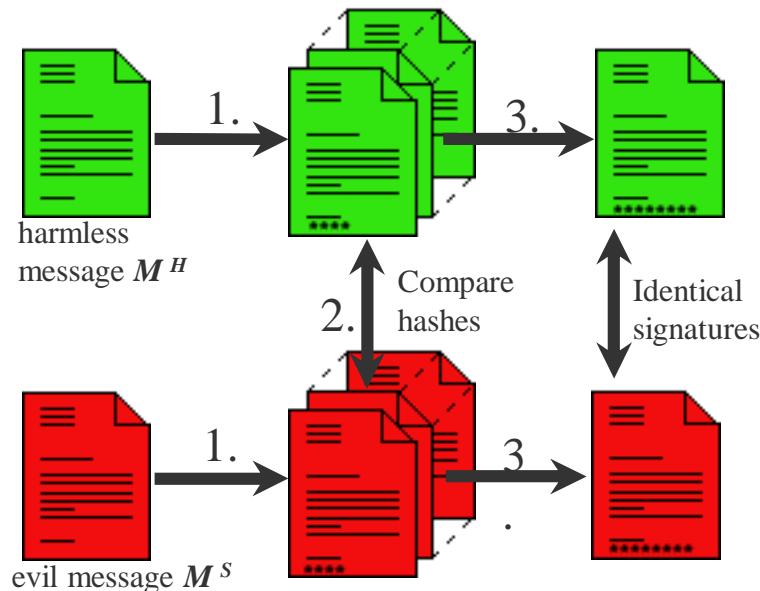
Concepts:

- 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.

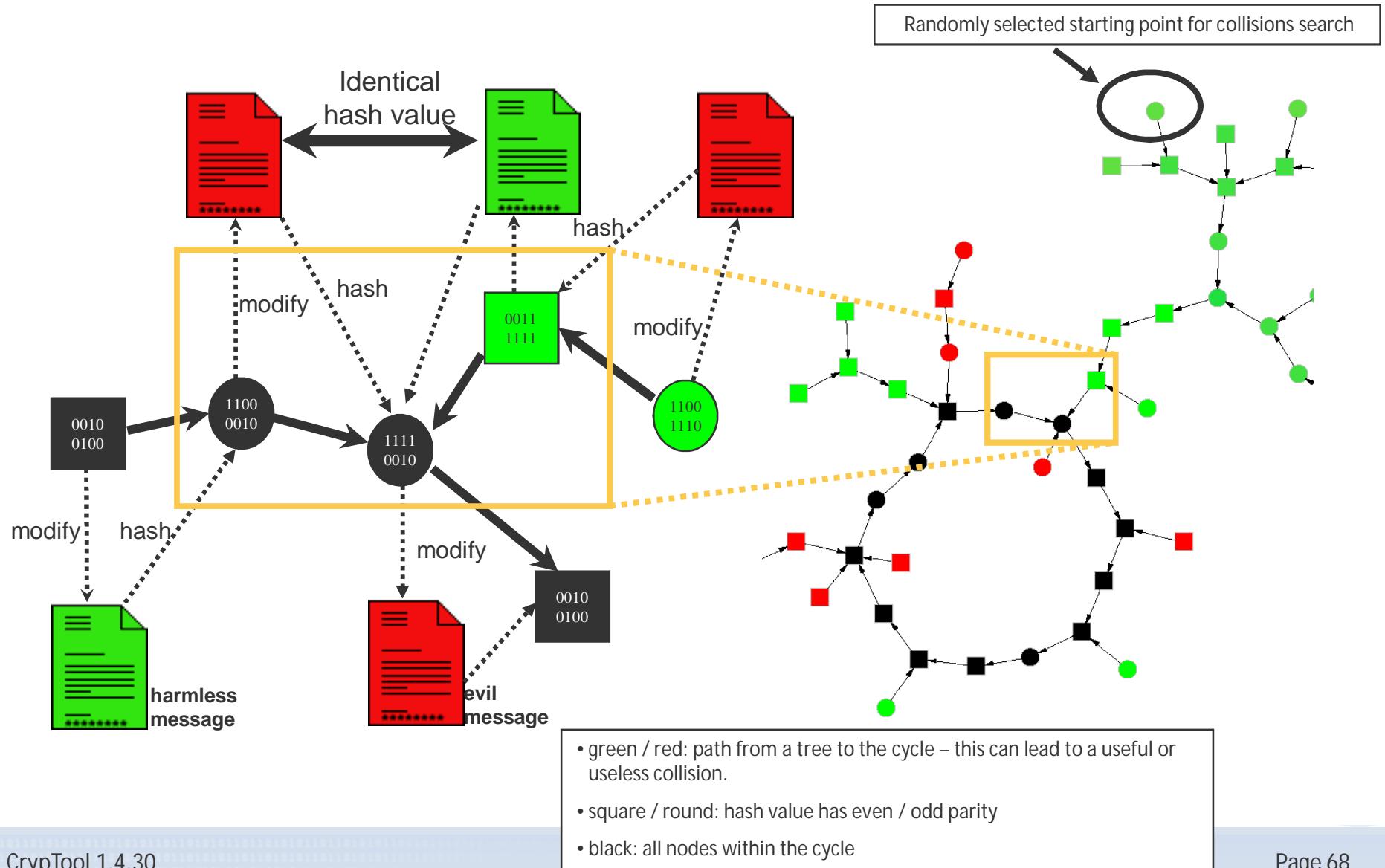
We know from the birthday paradox that for hash values of bit length  $n$ :

- 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)

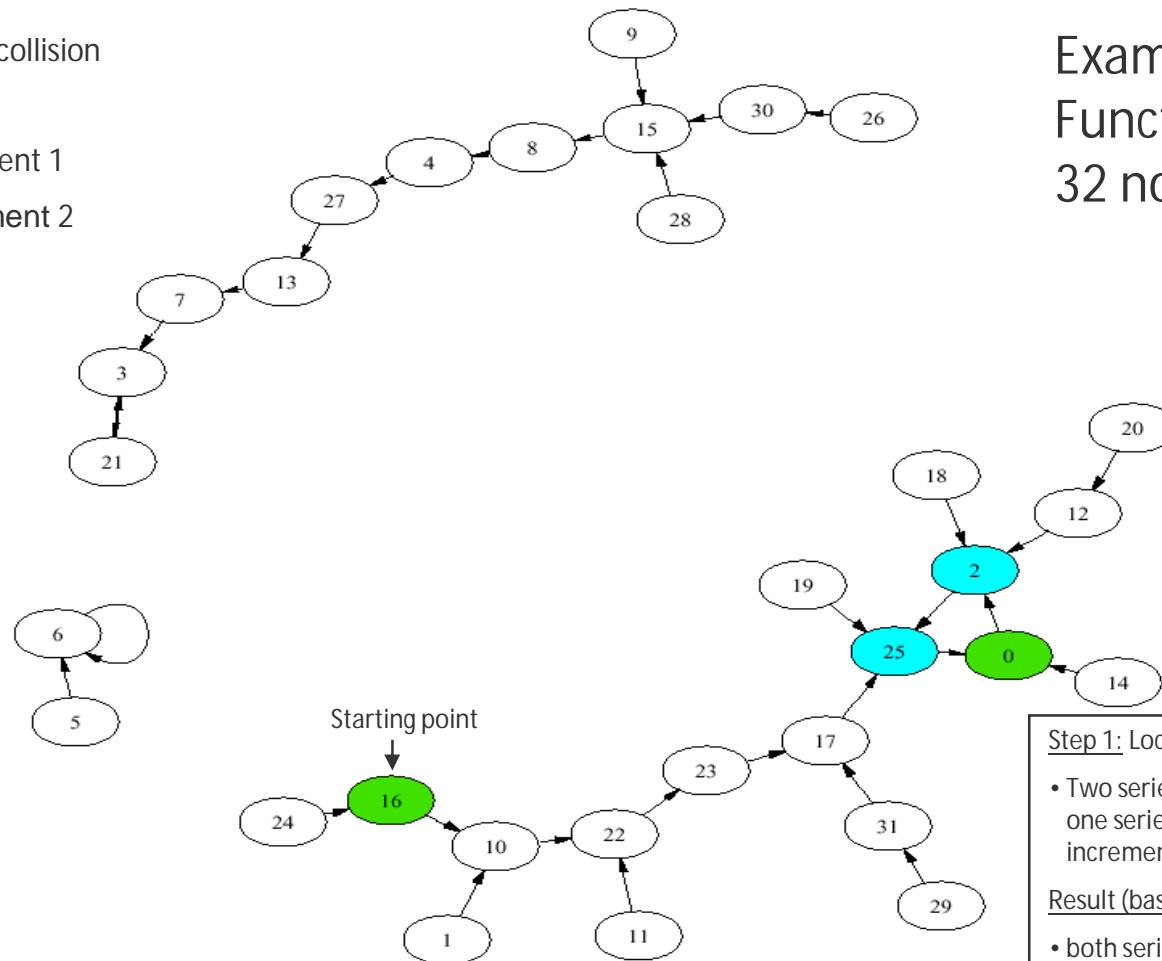
Mapping via text modifications



# 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.

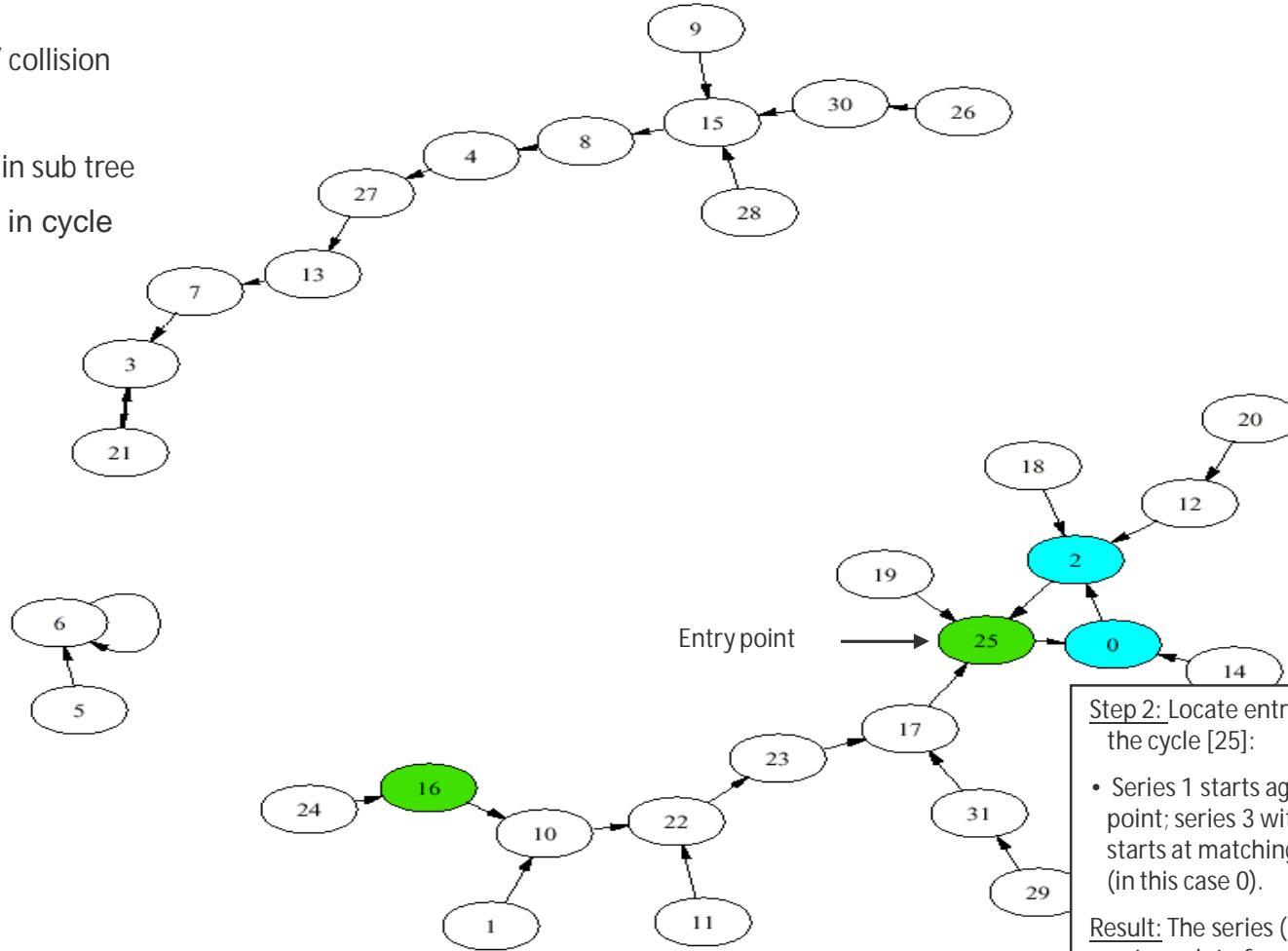
Result (based on graph theory):

- 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

-  start / collision
-  cycle
-  move in sub tree
-  move in cycle



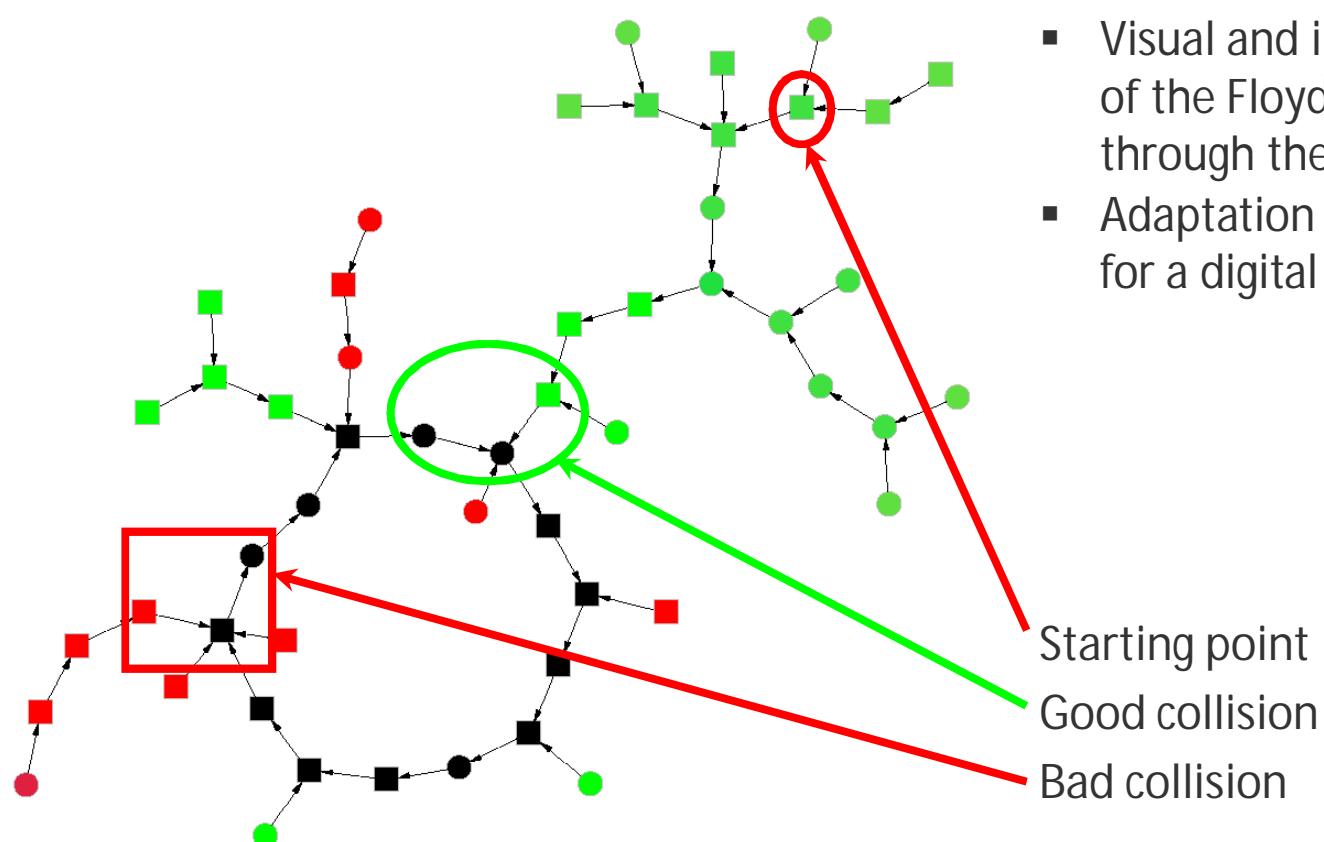
Step 2: Locate entry point of series 1 in the cycle [25]:

- Series 1 starts again from starting point; series 3 with an increment of 1 starts at matching point within the cycle (in this case 0).

Result: The series (1 and 3) match in cycle entry point of series 1 (in this case 25)

- The predecessors (in this case 17 and 2) result in a hash collision.

# Birthday Paradox Attack on Digital Signature



## Examination of Floyd algorithm

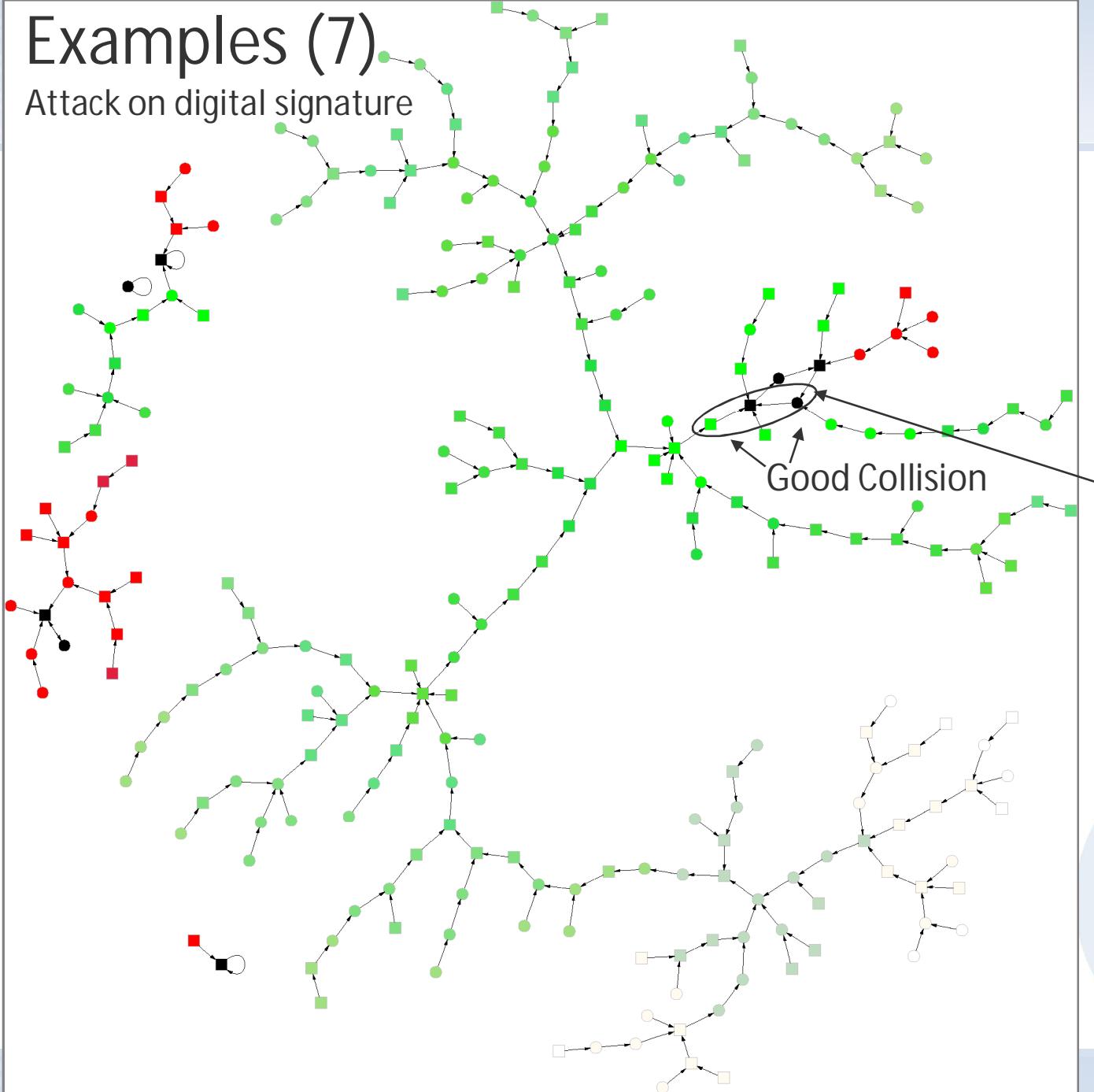
- Visual and interactive presentation of the Floyd algorithm („Moving through the mapping“ into a cycle).
- Adaptation of the Floyd algorithm for a digital signature attack.

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



# Examples (7)

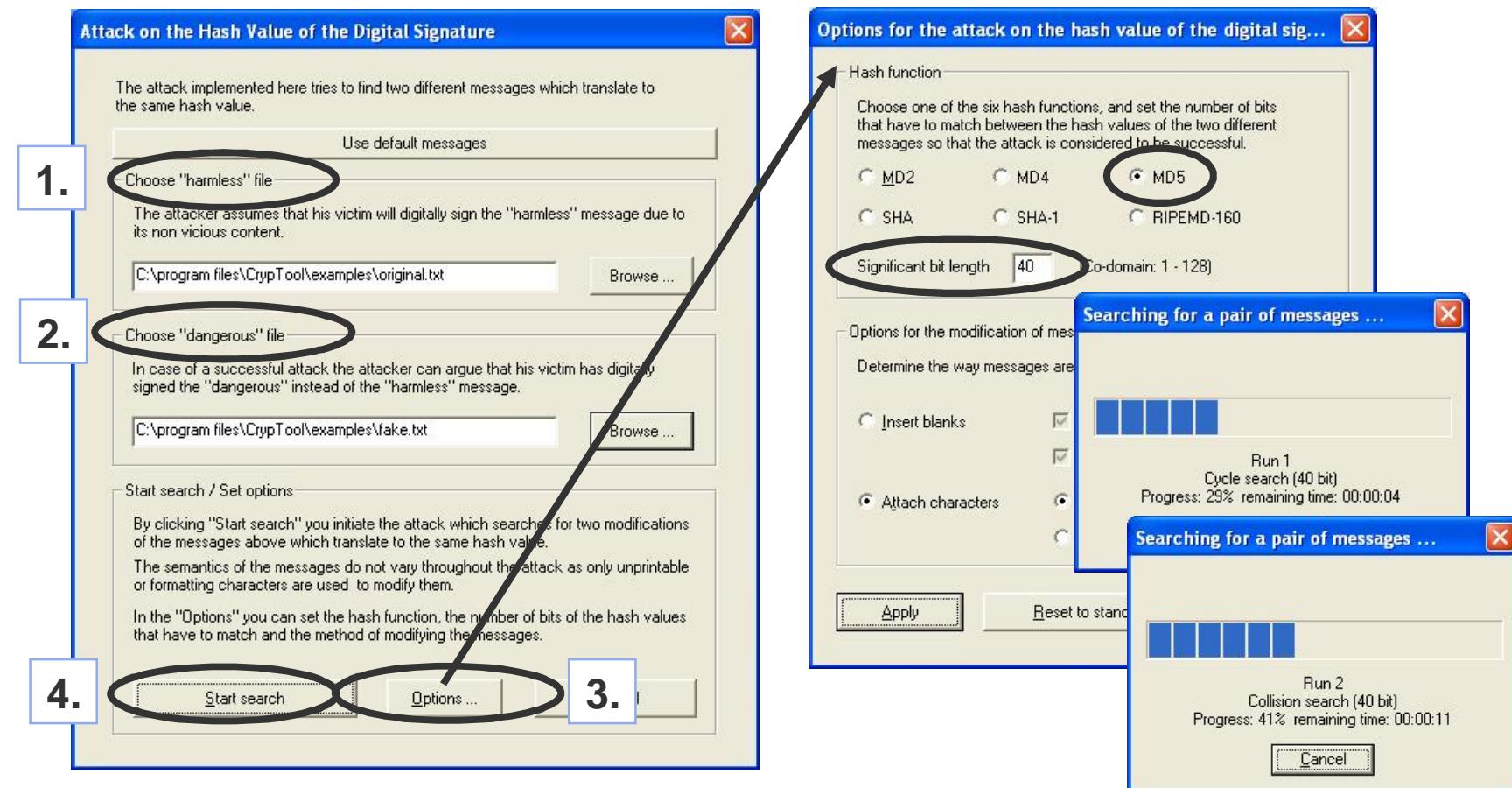
## Attack on digital signature



An example for 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)

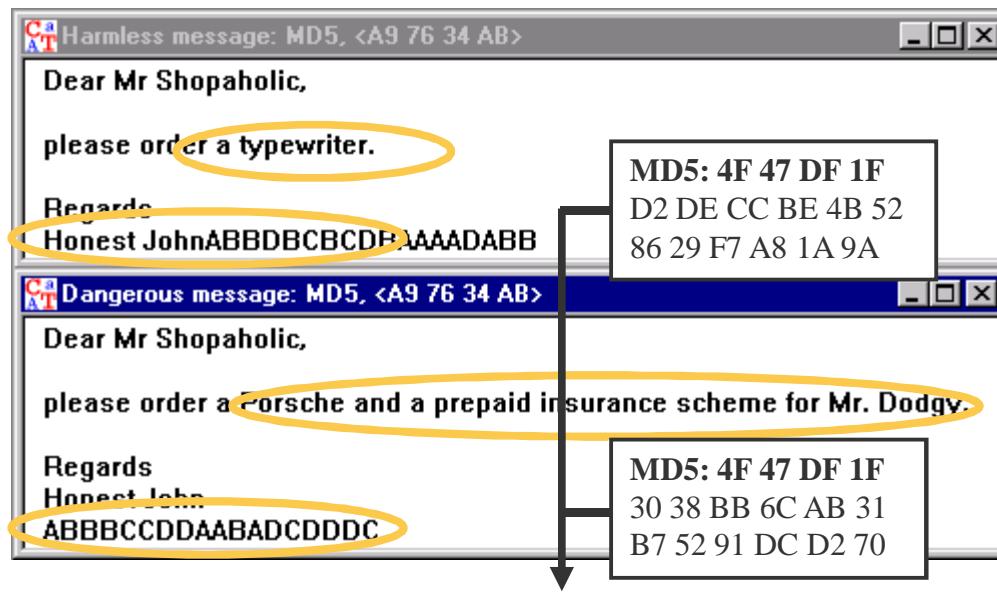
## Attack on digital signature: attack



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

# Examples (7)

Attack on digital signature: results



The first 32 bits of the hash values are identical.

In addition to the interactive handling:

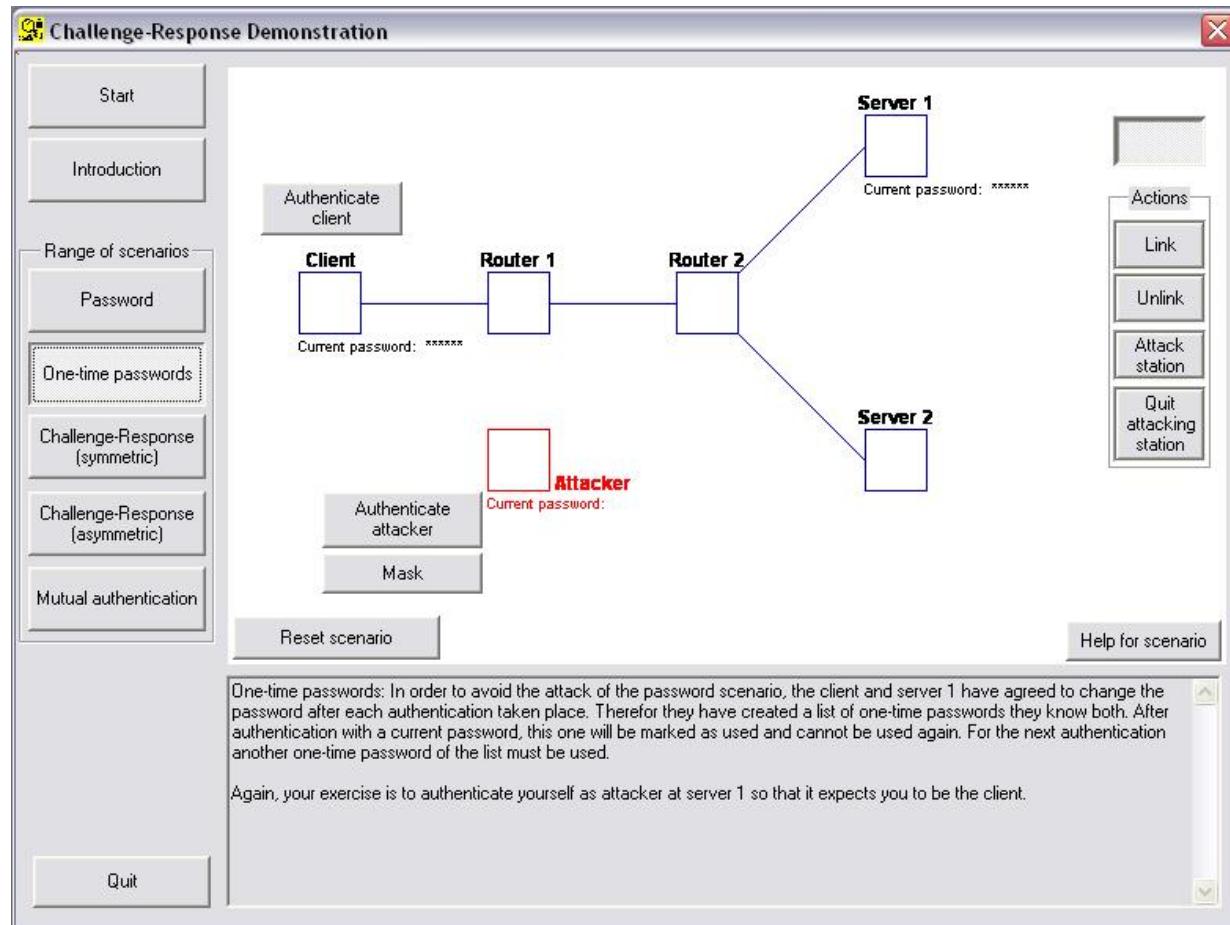
Automated offline feature in CrypTool: Execute and log the results for entire sets of parameter configurations. Available through command line execution of CrypTool.

## Experimental results

- 72 Bit *partial collision* (equality of the first 72 hash value bits) were found in a couple of days on a single PC.
- Signatures using hash values of up to 128 bit can be attacked today using massive parallel search!
- Use hash values of at least 160 bit length.

# Examples (8)

## Authentication in a client-server environment

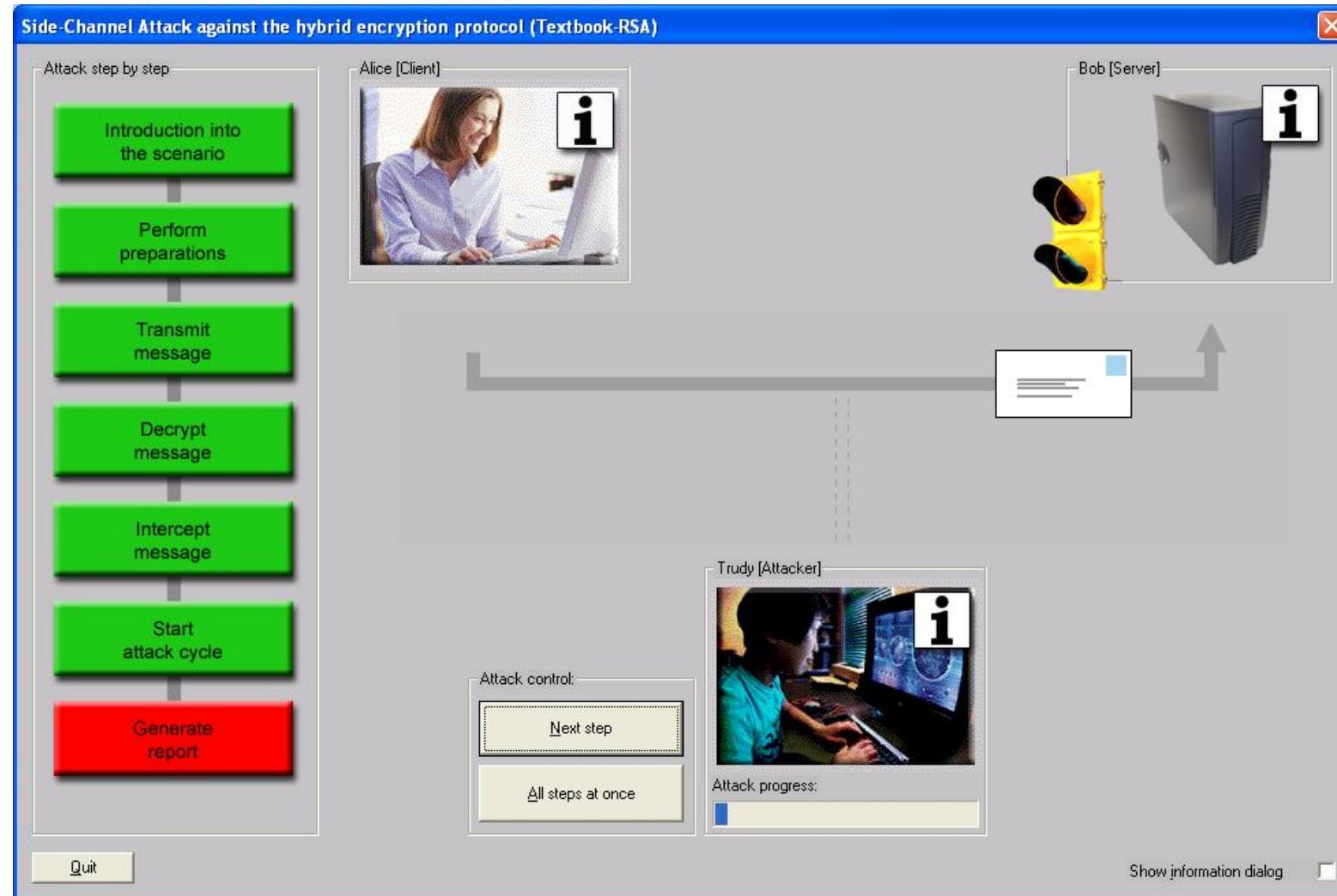


Menu: „Indiv. Procedures“ \ „Protocols“ \ „Network Authentication“

- Interactive demo for different authentication methods.
- Defined opportunities of the attacker.
- You can play the role of an attacker.
- Learning effect:  
Only mutual authentication is secure.

# 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)

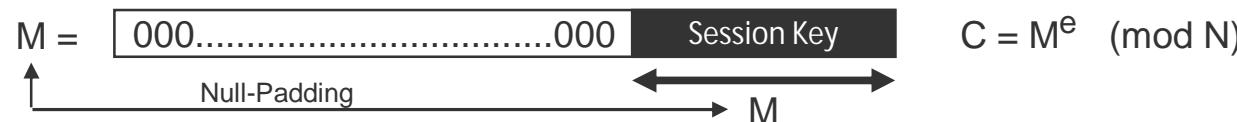
Idea for this side channel attack

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

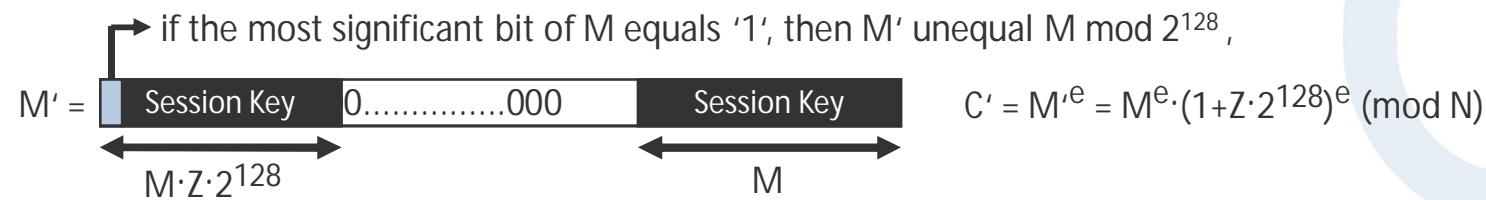
Prerequisites:

- RSA encryption:  $C = M^e \pmod{N}$  and decryption:  $M = C^d \pmod{N}$ .
- 128-Bit session keys (in M) are „word book encoded“ (null padding).
- The server knows the secret key d and
  - uses after decryption the 128 least significant bits only (no validation of zero padding bits) (that means the server does not recognize if there is something other than zero).
  - Prompts an error message, if the encryption attempt results in a wrong session key (decrypted text can not be interpreted by the server). In all other cases there will be no message.

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

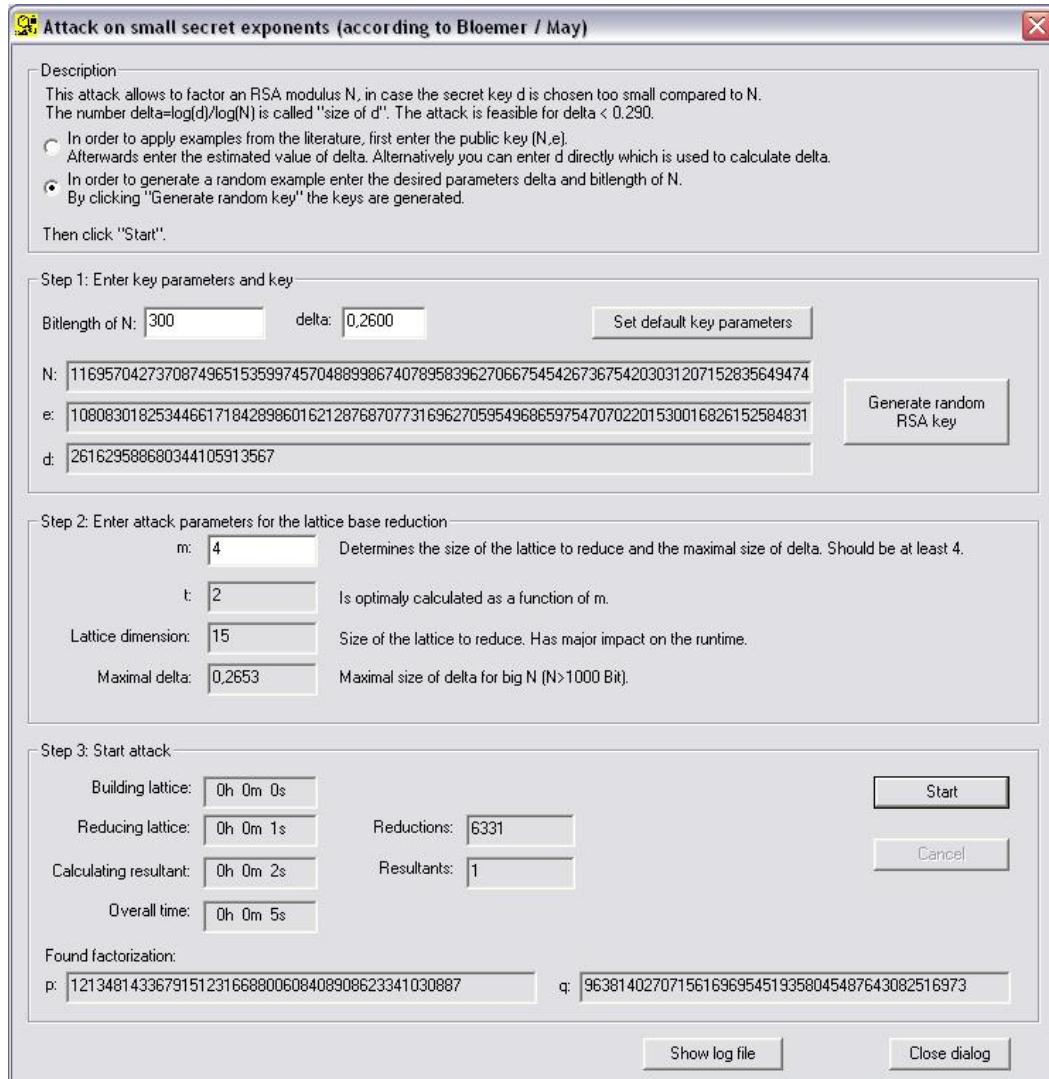


All bit positions for Z are successively calculated: For every step one gets 1 further bit. The attacker modifies C to  $C'$  (see below). If a bit overflow occurs while calculating  $M'$  on the server (recipient), the server sends an error message. Based on this information the attacker gets a bit for Z.



# Examples (10)

## Mathematics: Attacks on RSA using lattice reduction



- Shows how the parameters of the RSA method have to be chosen, so that the algorithm resists the lattice reduction attacks described in current literature.
- 3 variants
  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“ \ ...

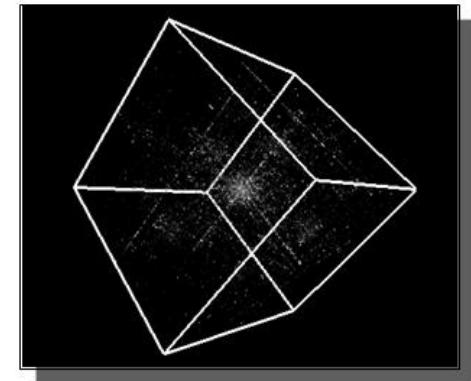
# Examples (11)

Random analysis with 3-D visualization

## 3-D visualization for random analysis

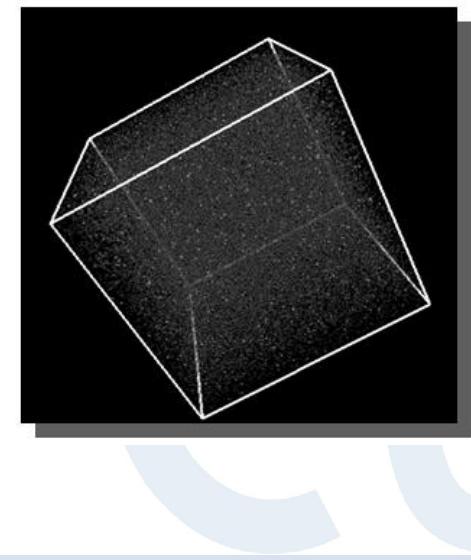
### Example 1

- 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 using menu: „Analysis“ \ „Analyse Randomness“ \ „3-D Visualization“
- Result: structures are easily recognisable



### Example 2

- Generation of random numbers: „Indiv. Procedures“ \ “Tools“ \ „Generate Random Numbers“
- It is recommended to generate at least 100,000 random bytes
- 3-D analysis using menu: „Analysis“ \ „Analyse Randomness“ \ „3-D Visualization“
- Result: uniform distribution (no structures are recognisable)

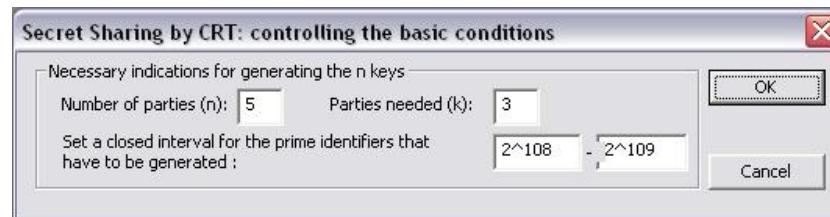


# Examples (12)

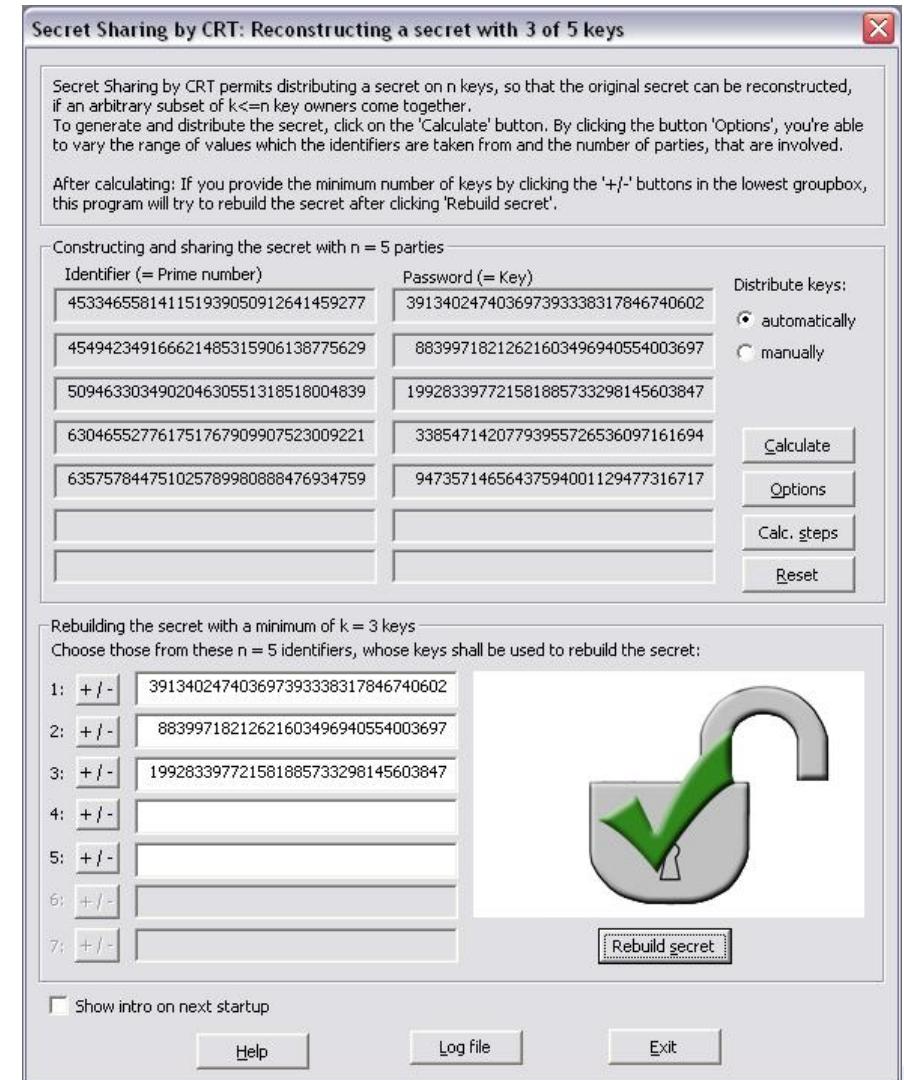
Secret sharing with CRT – implementation of the Chinese remainder theorem (CRT)

## Secret sharing example (1):

- Problem:
  - 5 people get a single key
  - To gain access at least 3 of the 5 people have to be present
- Menu: „Indiv. Procedures“ \ „Chinese Remainder Theorem Applications“ \ „Secret Sharing by CRT“
- „Options“ allows to configure more details of the method.



- „Calc. steps“ shows all steps to generate the key.

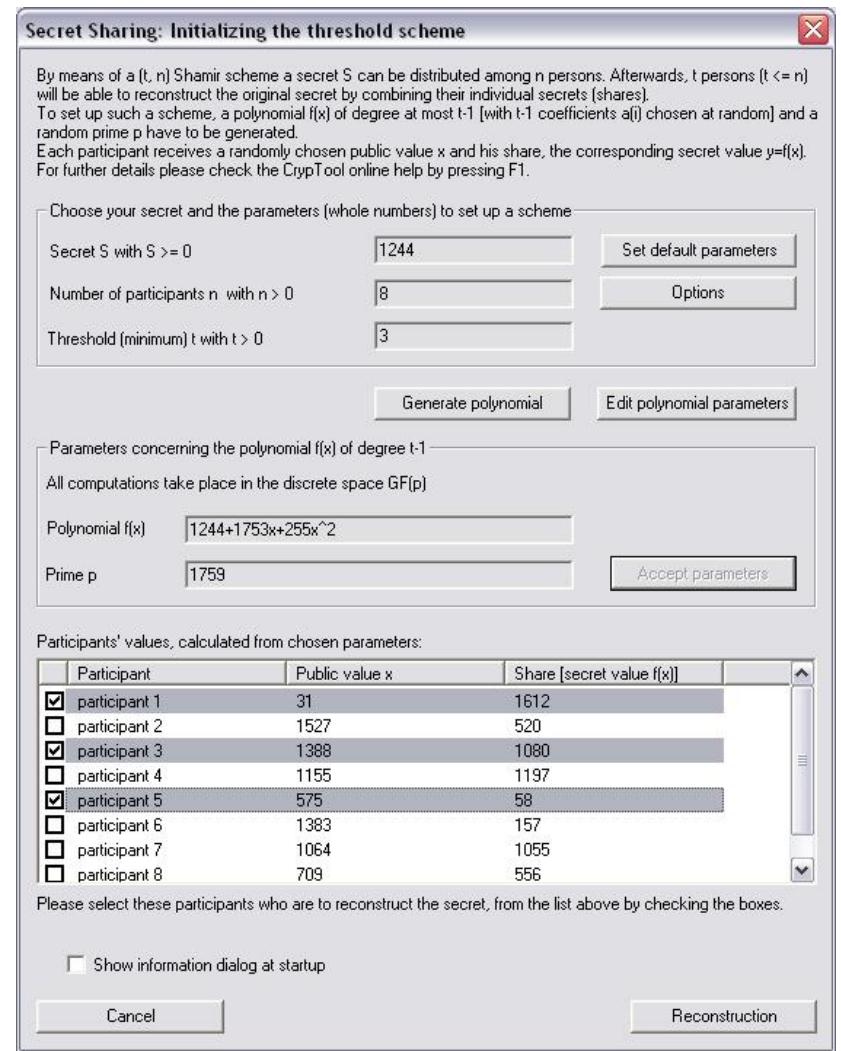


# Examples (12)

## Shamir secret sharing

### Secret sharing example (2)

- Problem
  - A secret value should be split for n people.
  - t out of n people are required to restore the secret value K.
  - (t, n) threshold scheme
- Menu: „Indiv. Procedures“ \ „Secret Sharing Demonstration (Shamir)“
  1. Enter the secret K, number of persons n and threshold t
  2. Generate polynom
  3. Use parameters
- Using „Reconstruction“ the secret can be restored

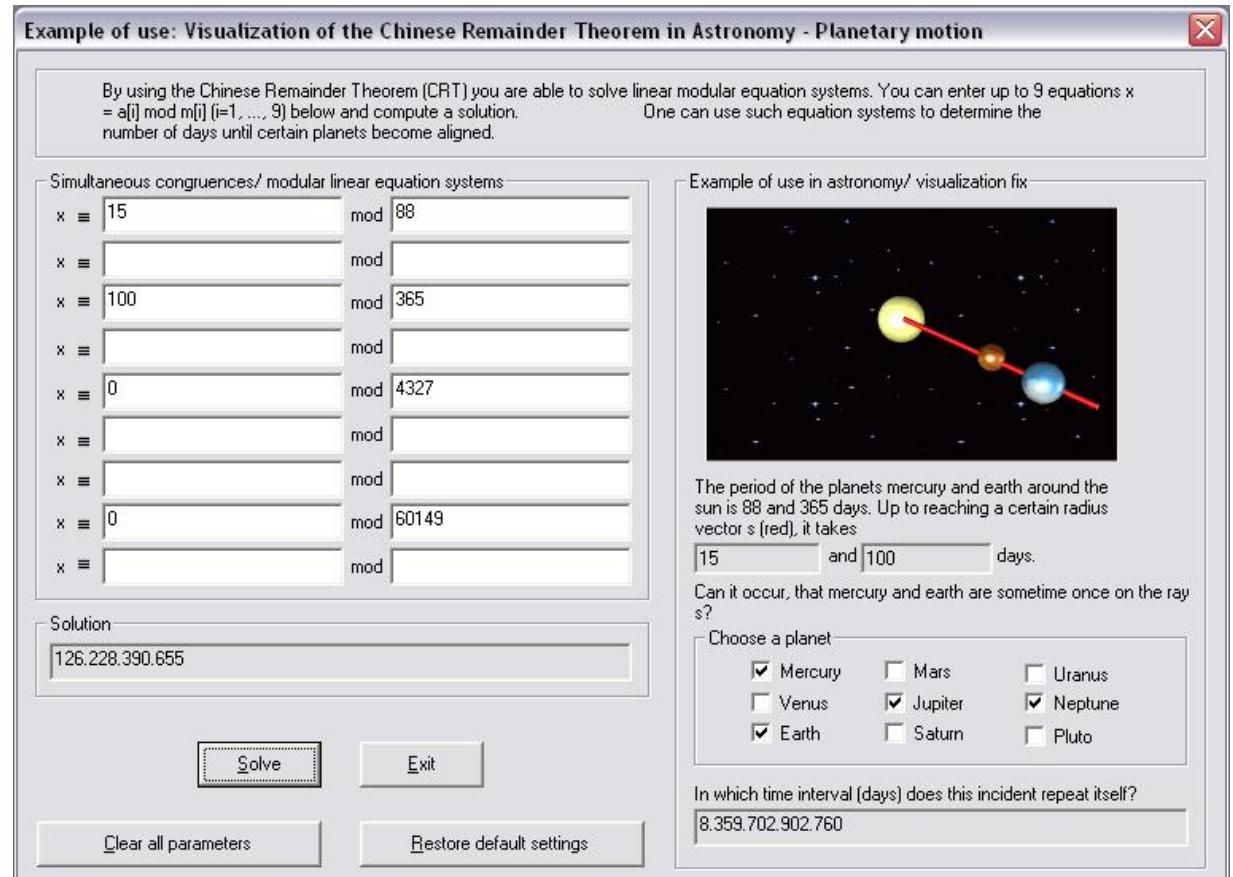


# Examples (13)

Implementation of CRT to solve linear modular equation systems

## Scenario in astronomy

- How long does it take until 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)

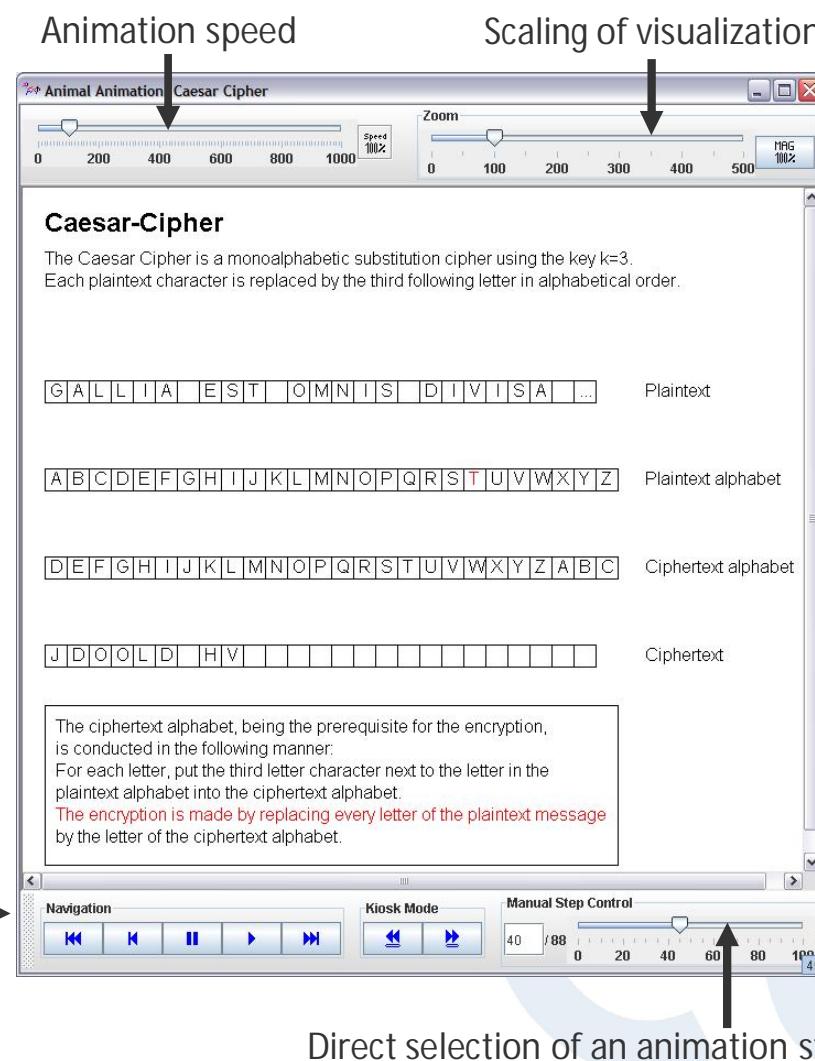
Animated visualization of several symmetric algorithms

- Caesar
- Vigenère
- Nihilist
- DES

### CrypTool

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

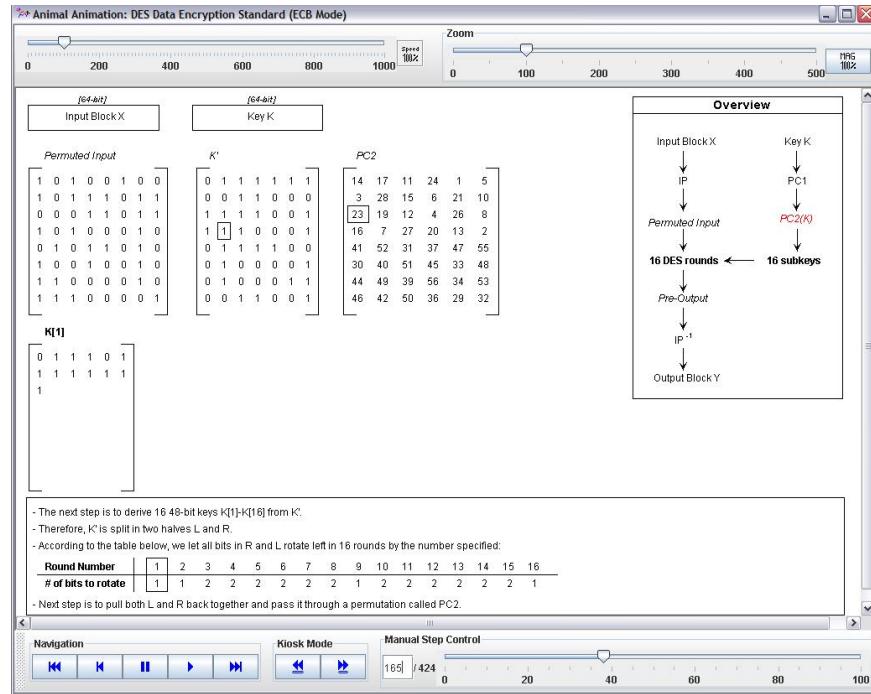
Animation controls (next,  
forward, pause, etc.)



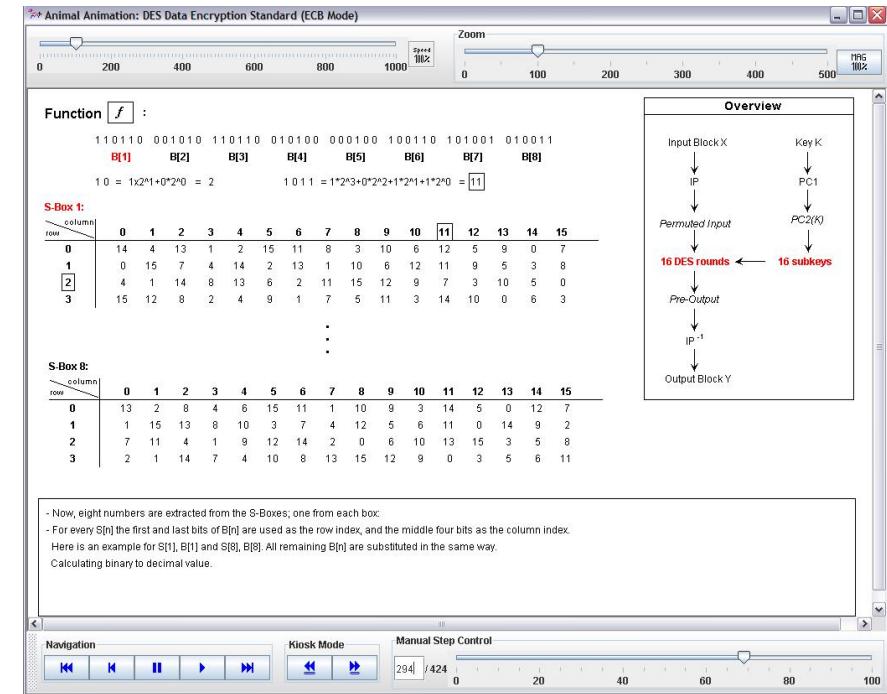
# Examples (14)

## Visualization of symmetric encryption methods using ANIMAL (2)

### Visualization of DES encryption



After the permutation of the input block using the initialisation vector IV the key K is being permuted with PC1 and PC2.



# Examples (15)

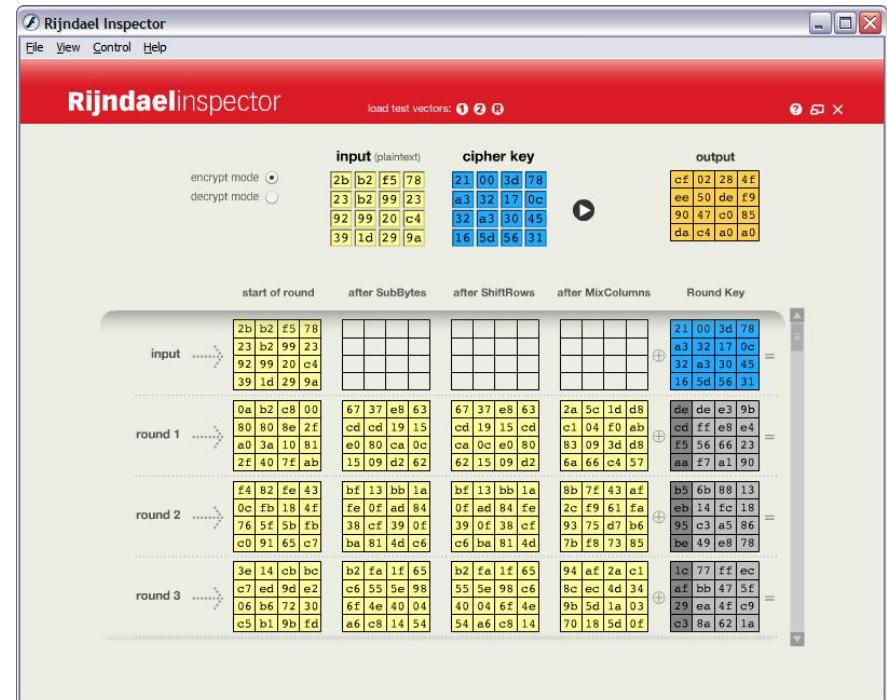
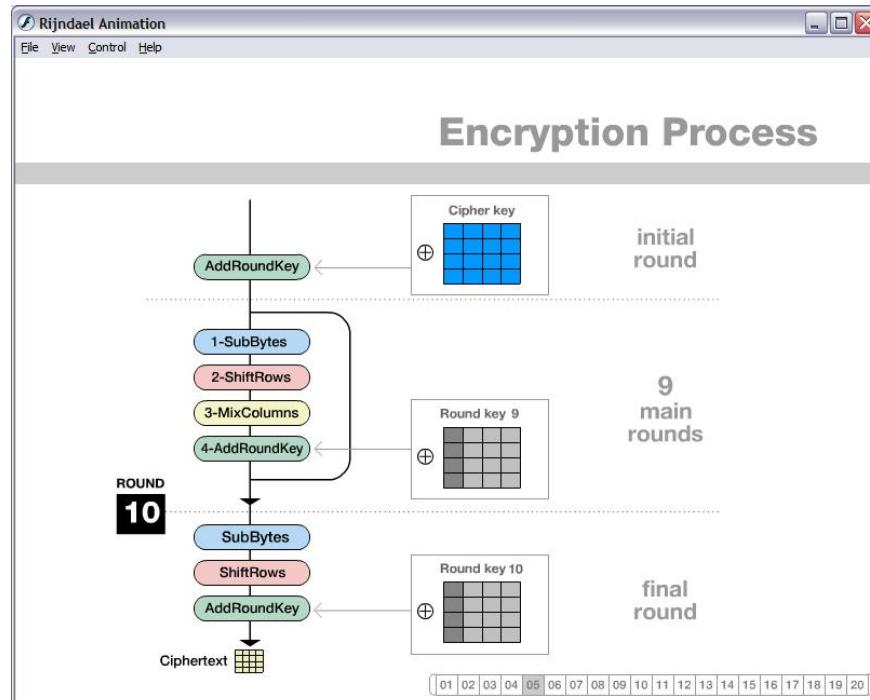
Visualizations of AES (Rijndael cipher) – using Flash

Rijndael Animation (the Rijndael cipher was the winner of the AES submission)

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

Rijndael Inspector

- Encryption process for testing (using your own data, showing each step's matrices)



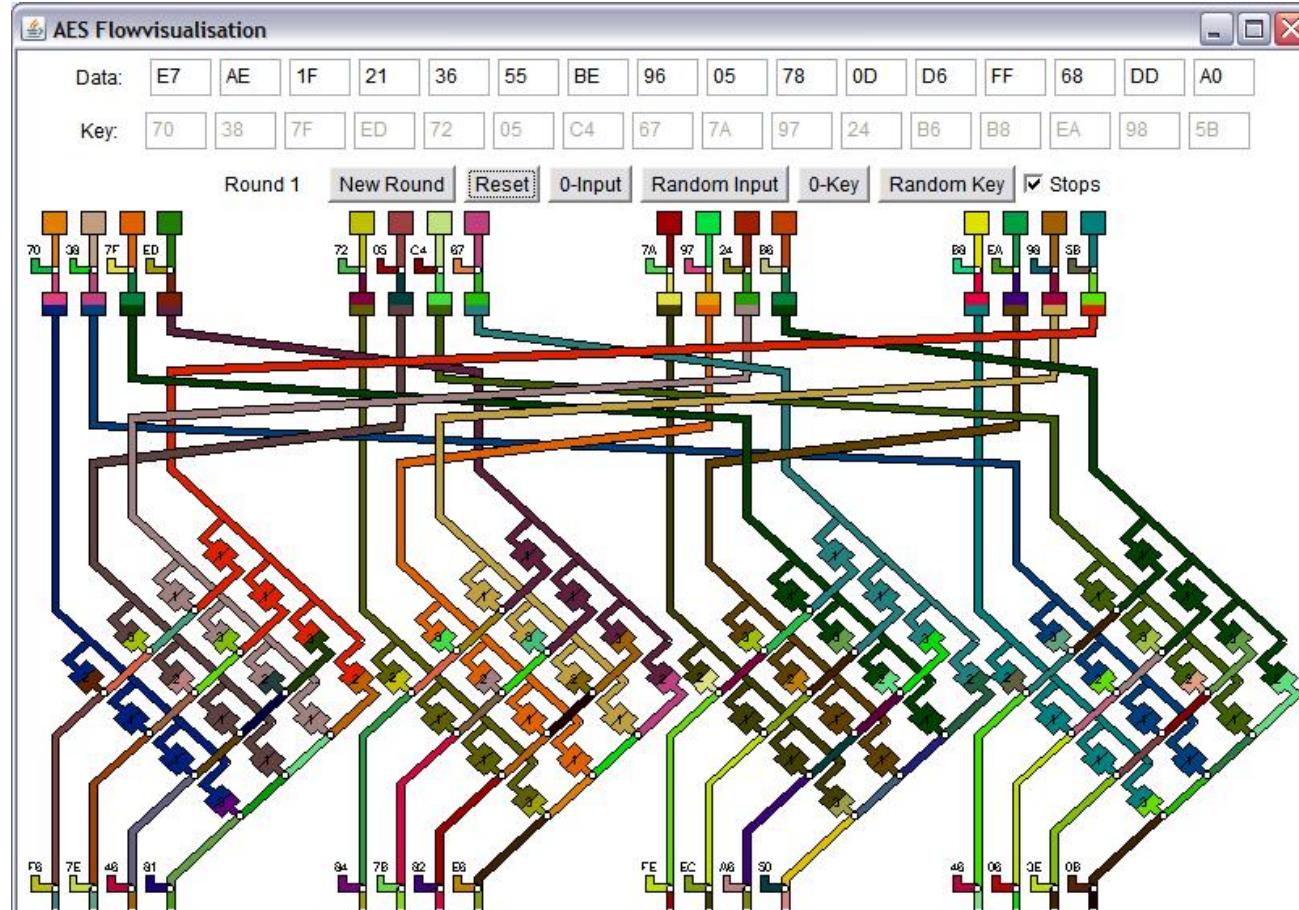
Menu: "Indiv. Procedures" \ "Visualization of Algorithms" \ "AES" \ "Rijndael Animation" or "Rijndael Inspector"

# Examples (15)

Flow visualization of AES (Rijndael Cipher) – using Java

## Rijndael Flow Visualization

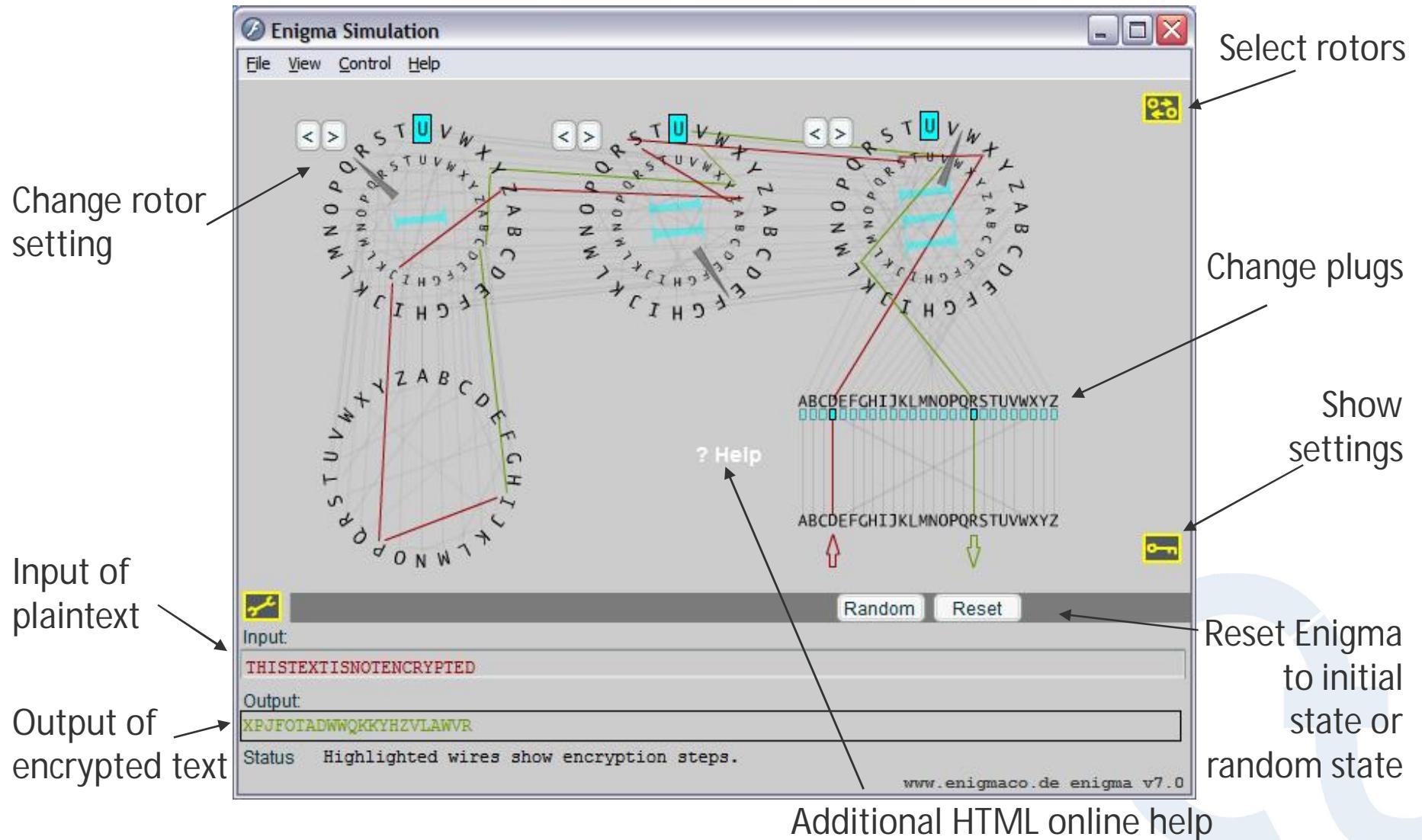
- Visualization of data changes per round using colour gradient



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

# Examples (16)

Visualization of the Enigma encryption

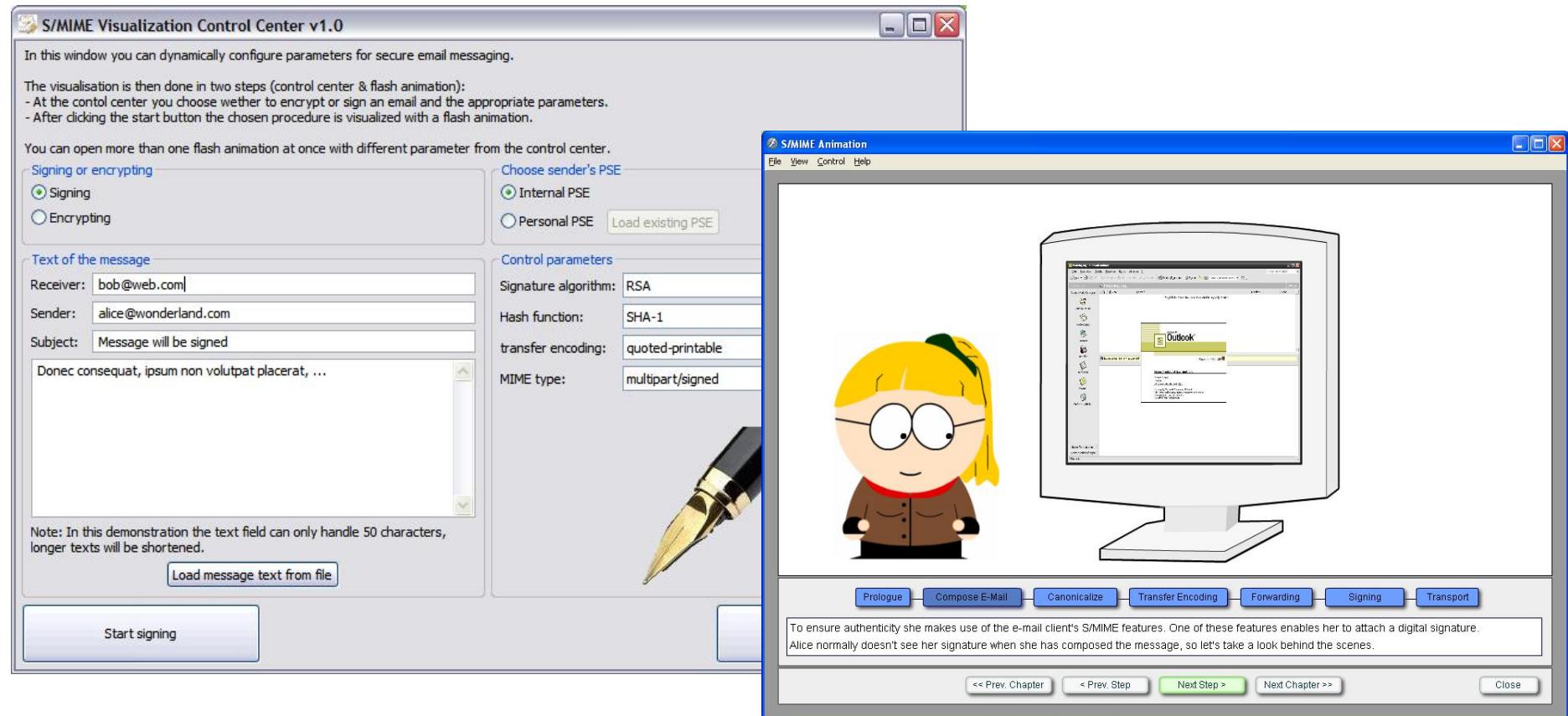


# Examples (17)

## Visualization of secure E-Mail using S/MIME

### S/MIME Visualization

- Control Center: Sign/Encrypt messages with different parameters
- Animation: From the creation at the sender to the reading at the receiver



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

# Examples (18)

## Generation of a message authentication code (MAC)

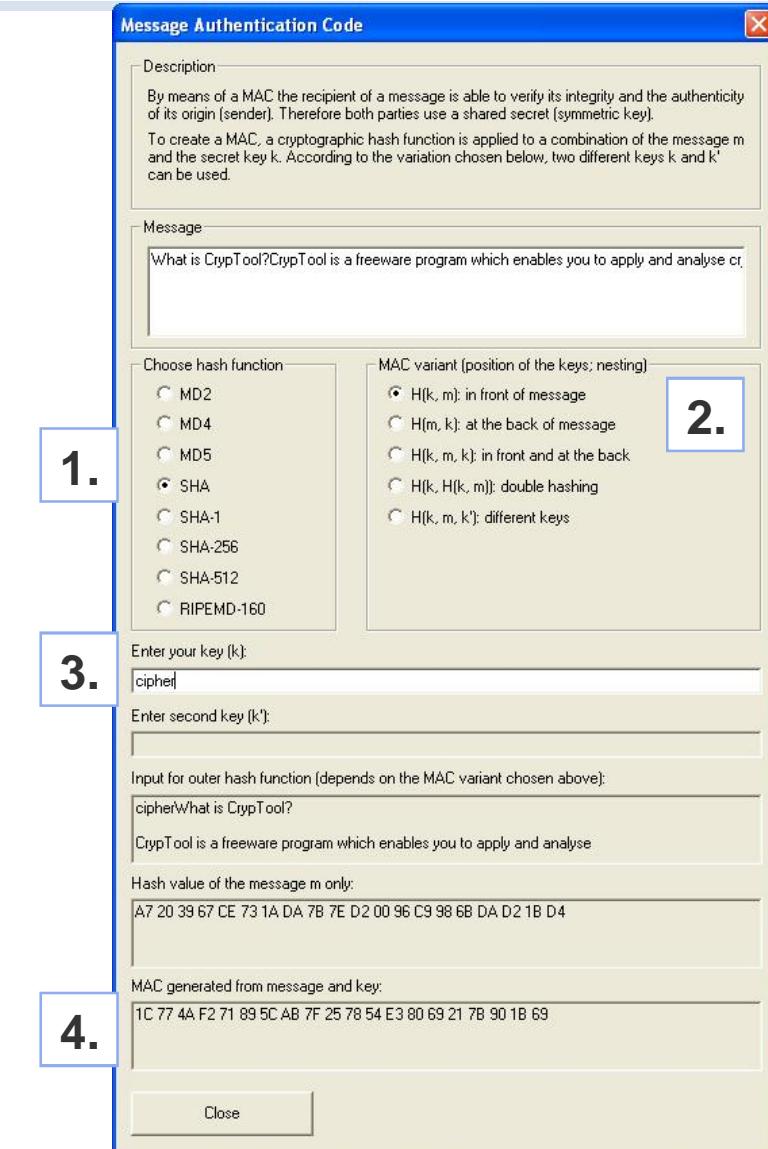
### Message Authentication Code (MAC)

- Ensures integrity of a message
- Authentication of the message
- Basis: a common key

### Generation of a MAC in CrypTool

- Choose a hash function
- Select MAC variant
- Enter a key (depending on MAC variant also two keys)
- Generation of the MAC (automatic)

Menu: „Indiv. Procedures“ \ „Hash“ \ „Generation of MACs“



# Examples (19)

## Hash demonstration

Sensitivity of hash functions to plaintext modifications

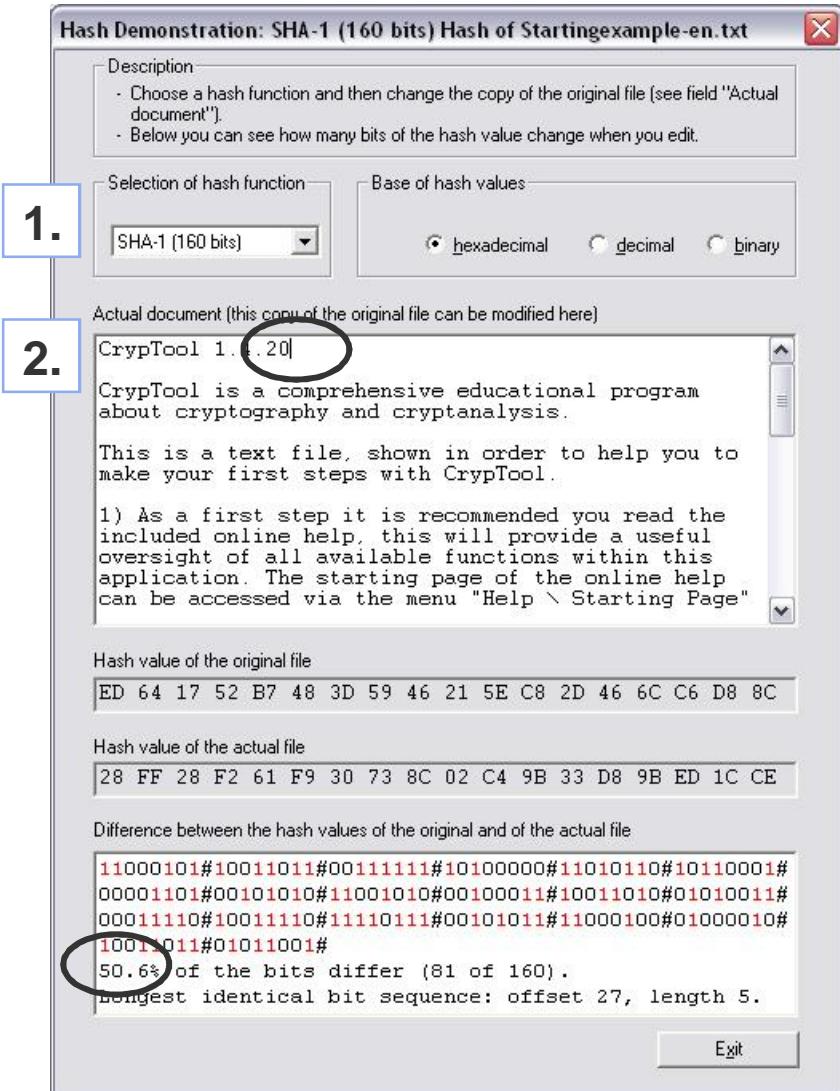
1. Select a hash function
2. Modification of characters in plaintext

Example:

Entering a blank after „CrypTool“ in the example text results in a 50.6 % change of the bits of the generated hash value.

A good hash function should react sensitive to even the smallest change within the plaintext – „Avalanche effect“ (small change, big impact).

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



# Examples (20)

Learning tool for number theory

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

The screenshot shows a window titled "NT" with a menu bar: Calculators, Navigation, Glossaries, Help. The main content area is titled "3.2 Fermat Test". On the right, it says "page 4 of 11". The text explains Fermat's Little Theorem: "Each prime p passes a test that results from Fermat's Little Theorem: Try for a  $b \in \{2, \dots, p-1\}$ , if  $b^{p-1} \equiv 1 \pmod{p}$ ". It notes that some composite numbers also pass this test. An example is given: "Example:  $341 = 11 \cdot 31$ , even so is  $2^{340} \equiv 1 \pmod{341}$ ". A user interface allows testing other numbers: "n =   $2^{n-1} \equiv 1 \pmod{n}$  Test passed". Below it, "GCD(b, n) = 1" and buttons for " $\langle$ ", " $b$ ", and " $\rangle$ ". A definition states: "Definition: Let  $n$  be a composite number,  $b$  coprime to  $n$ . If  $b^{n-1} \equiv 1 \pmod{n}$ , then one calls

- $n$  Pseudo Prime to Base  $b$ ,
- $b$  Liar for (the primality of)  $n$ ,

otherwise one calls  $b$  Witness against (the primality of)  $n$ ". A theorem follows: "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](#)". At the bottom are navigation icons (back, forward, search, etc.) and the text "(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
- Foundation of elliptic curve cryptography (ECC)

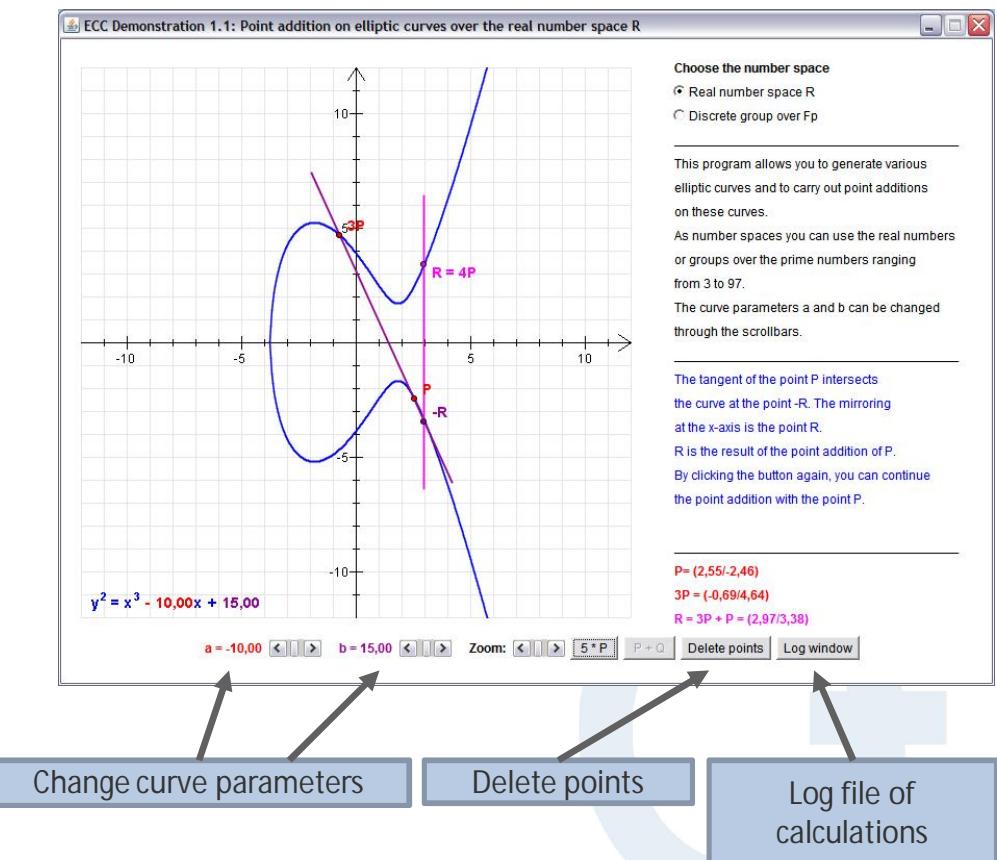
### Example 1

- Mark point P on the curve
- Mark point Q on the curve
- Press button „P+Q“: The straight line through P and Q intersects the curve in point -R
- Mirroring on the X-axis results in point R

### Example 2

- Mark point P on the curve
- Press button „2\*P“: The tangent of point P intersects the curve in point -R
- Mirroring on the X-axis results in point R

Menu: „Indiv. Procedures“ \ „Number Theory - Interactive“ \ „Point Addition on Elliptic Curves“



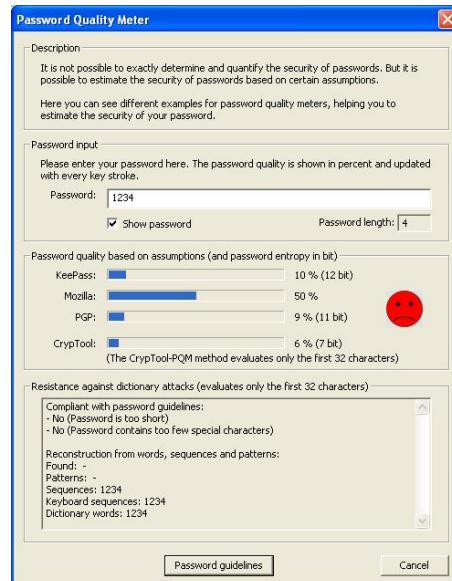
# Examples (22)

## Password Quality Meter (PQM) 1

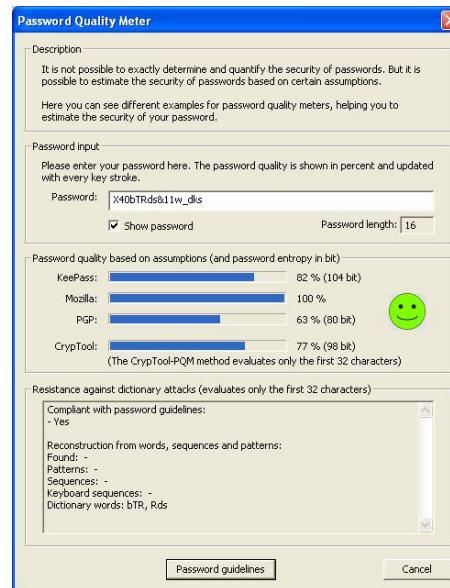
### Functions

- Measuring the quality of passwords
- Compare with PQMs in other applications: KeePass, Mozilla und PGP
- Experimental measuring through CrypTool algorithm
- Example: Input of a password (while showing the password)

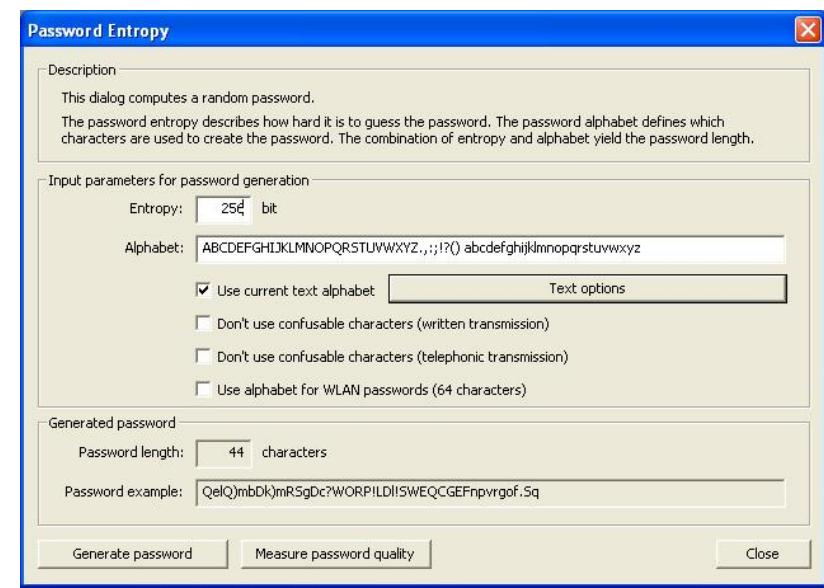
Password: **1234**



Password: **X40bTRds&11w\_dks**



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



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

# Examples (22)

## Password Quality Meter (PQM) 2

### Findings of 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 as indicator of the randomness of password characters of the password space (higher password entropy results in improved password quality)
- Passwords should not exist in a dictionary (remark: a dictionary check is not yet implemented in CrypTool).

### 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 attack 1. and 2. do not compromise the password. Regarding brute-force attacks the length of the password (at least 8 characters) as well as the used character sets are important.

# Examples (23)

## Brute-force analysis 1

### Brute-force analysis

Optimised brute-force analysis under the assumption that the key is partly 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*****

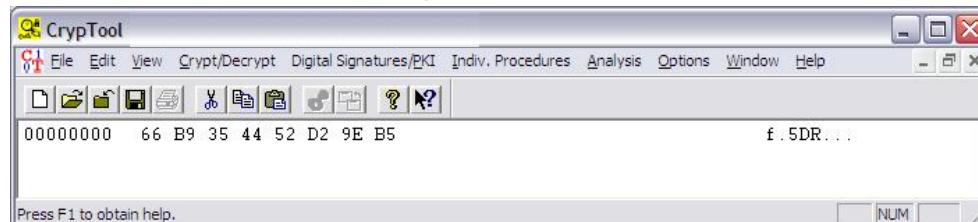
Encrypted text (Hex)
66b9354452d29eb5
1f0dd05d8ed51583
bcf9ebd1979ead6a
8cf42d40e004a1d4
0ed33fed7f46c585
d6d8641bc4fb2478
a2e66d852e175f5c

# Examples (23)

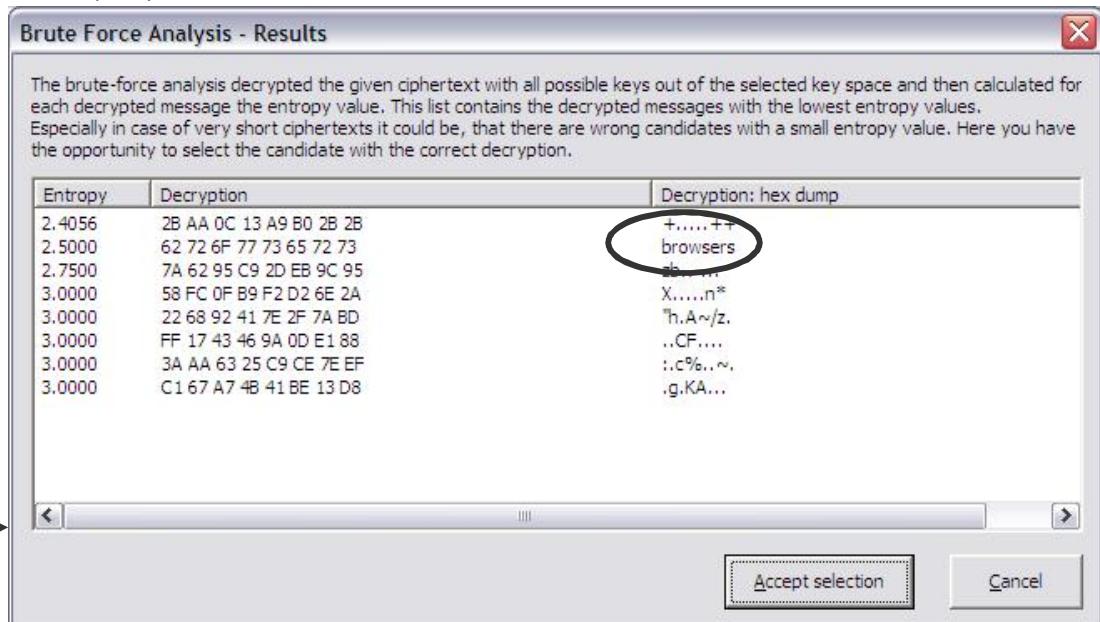
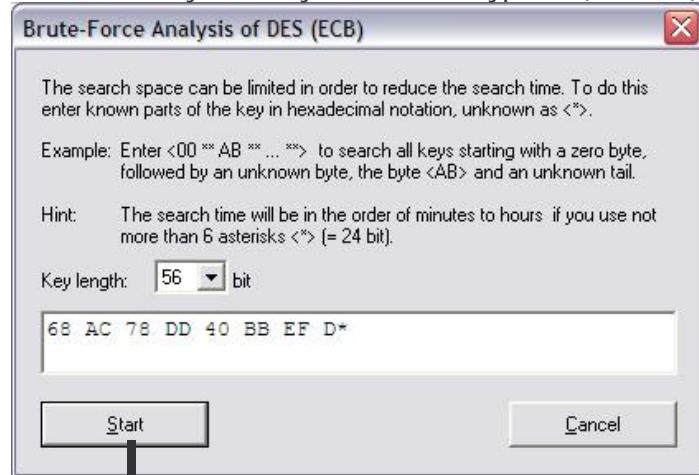
## Brute-force analysis 2

1. Input of encrypted text
2. Use brute-force analysis
3. Input partly known key
4. Start brute-force analysis
5. Analysis of the results: Low entropy as evidence of a possible decryption. However, because a very short plaintext has been used in this example, the correct result does not have the lowest entropy.

Use „View“ \ „Show as HexDump“

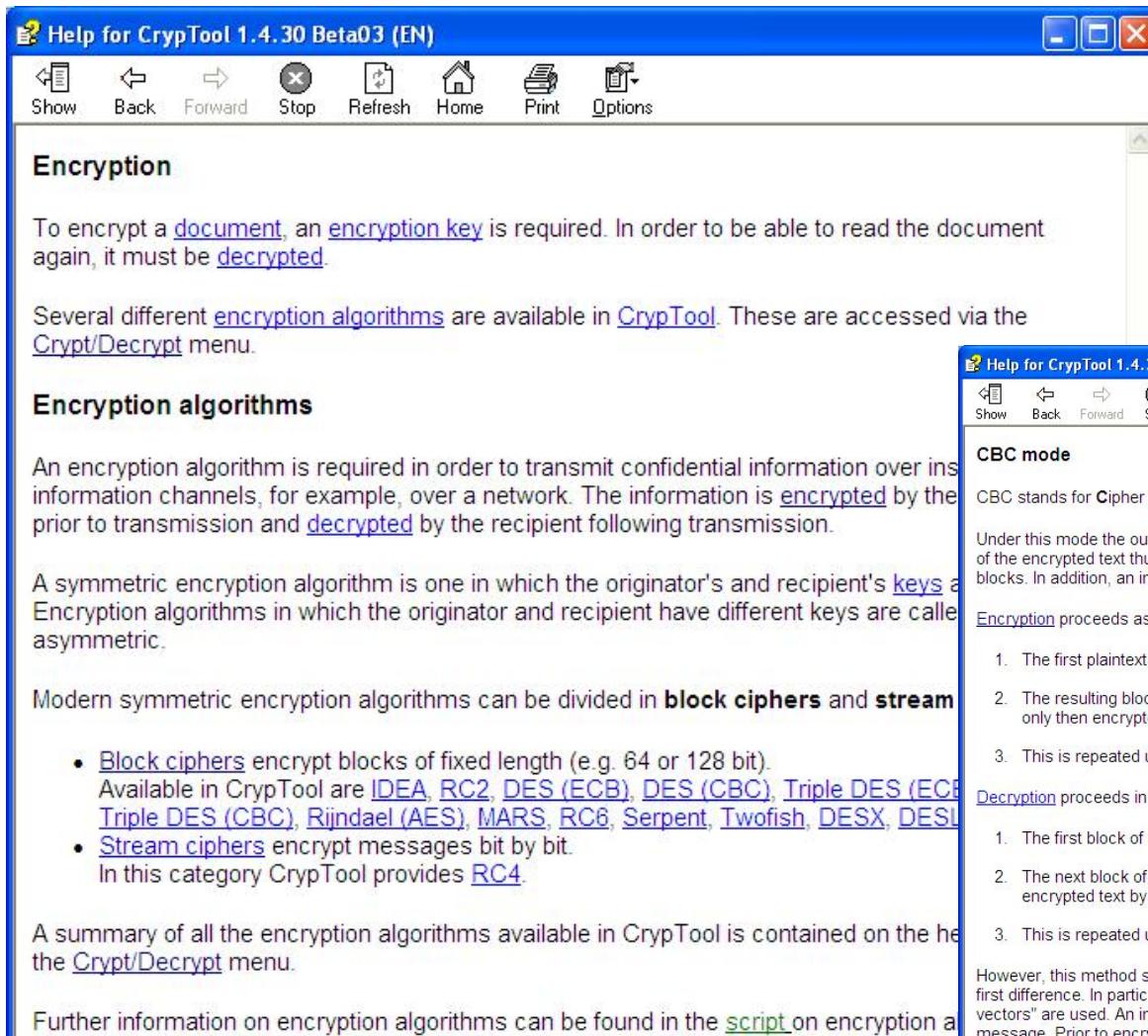


Menu: „Analysis“ \ „Symmetric Encryption (modern)“ \ „DES (ECB)“

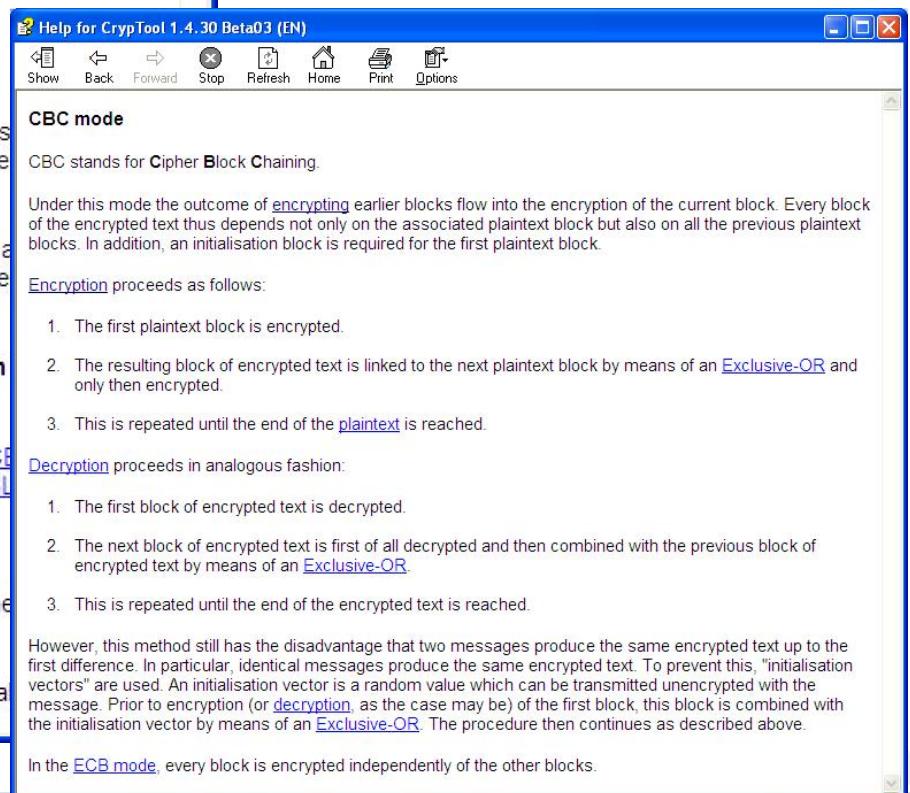


# Examples (24)

## CrypTool online help 1

A screenshot of the CrypTool 1.4.30 Beta03 online help window. The title bar says "Help for CrypTool 1.4.30 Beta03 (EN)". The menu bar includes "Show", "Back", "Forward", "Stop", "Refresh", "Home", "Print", and "Options". The main content area has a green header "Encryption". A text block states: "To encrypt a [document](#), an [encryption key](#) is required. In order to be able to read the document again, it must be [decrypted](#)." Another text block says: "Several different [encryption algorithms](#) are available in [CrypTool](#). These are accessed via the [Crypt/Decrypt](#) menu." Below this is a section titled "Encryption algorithms" with a text block: "An encryption algorithm is required in order to transmit confidential information over [information channels](#), for example, over a network. The information is [encrypted](#) by the sender prior to transmission and [decrypted](#) by the recipient following transmission." A note below states: "A symmetric encryption algorithm is one in which the originator's and recipient's [keys](#) are the same. Encryption algorithms in which the originator and recipient have different keys are called asymmetric." A section titled "Modern symmetric encryption algorithms can be divided in **block ciphers** and **stream ciphers**" follows. A bulleted list includes: "• [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](#), [DESX \(CBC\)](#), [AES \(GCM\)](#), [AES \(CCM\)](#), [AES \(CTR\)](#), [AES \(EAX\)](#), [AES \(OCB\)](#), [AES \(GCM\)](#), [AES \(CCM\)](#), [AES \(CTR\)](#), [AES \(EAX\)](#), [AES \(OCB\)](#)". "• [Stream ciphers](#) encrypt messages bit by bit. In this category CrypTool provides [RC4](#)". A summary of all encryption algorithms is mentioned as being available in the "Crypt/Decrypt" menu.

Menu: „Help“ \ „Starting Page“

A screenshot of the CrypTool 1.4.30 Beta03 online help window. The title bar says "Help for CrypTool 1.4.30 Beta03 (EN)". The menu bar includes "Show", "Back", "Forward", "Stop", "Refresh", "Home", "Print", and "Options". The main content area has a green header "CBC mode". A text block states: "CBC stands for **Cipher Block Chaining**. Under this mode the outcome of [encrypting](#) earlier blocks flow into the encryption of the current block. Every block of the encrypted text thus depends not only on the associated plaintext block but also on all the previous plaintext blocks. In addition, an initialisation block is required for the first plaintext block." A note below states: "Encryption proceeds as follows: 1. The first plaintext block is encrypted. 2. The resulting block of encrypted text is linked to the next plaintext block by means of an [Exclusive-OR](#) and only then encrypted. 3. This is repeated until the end of the [plaintext](#) is reached." A note below states: "Decryption proceeds in analogous fashion: 1. The first block of encrypted text is decrypted. 2. The next block of encrypted text is first of all decrypted and then combined with the previous block of encrypted text by means of an [Exclusive-OR](#). 3. This is repeated until the end of the encrypted text is reached." A note at the bottom states: "However, this method still has the disadvantage that two messages produce the same encrypted text up to the first difference. In particular, identical messages produce the same encrypted text. To prevent this, "initialisation vectors" are used. An initialisation vector is a random value which can be transmitted unencrypted with the message. Prior to encryption (or [decryption](#), as the case may be) of the first block, this block is combined with the initialisation vector by means of an [Exclusive-OR](#). The procedure then continues as described above." Another note at the bottom states: "In the [ECB mode](#), every block is encrypted independently of the other blocks."

# Examples (24)

## CrypTool online help 2

**Help for CrypTool**

Ausblenden Zurück Vorwärts Abbrechen Aktualisieren Startseite Drucken Optionen

Inhalt Index Suchen | Zu suchendes Schlüsselwort:

Lattice reduction

- Known-plaintext attack
- 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)
- Message authentication code (MAC)
- 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 Con
- Parent window
- Password
- Pattern search
- Periodicity analysis
- Permutation encryption algorithm
- PGP
- Phase space visualization
- Phi function of Euler
- PIN
- PKCS #11

Anzeigen

**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. ( $3x_0+1=7, 7 \bmod 7 = 0$ )

**Annotations:**

In 1988 by Johan Håstad [Hås88] presented this method for the first time and it was further developed by Don Coppersmith [Cop96a, Cop96b] in 1996. Nick Howgrave-Graham [HG97] showed a more intuitive approach.

**Sources:**

[Conrad, Coppersmith, Don: Finding a Small Root of a Bivariate Integer Equation: Factoring with High Bits Known. In: ...]

Menu: „Help“ \ „Starting Page“

# Examples (24)

## CrypTool online help 3

**Help for CrypTool**

Ausblenden Zurück Vorwärts Abbrechen Aktualisieren Startseite Drucken Optionen

Inhalt Index Suchen

Zu suchendes Schlüsselwort: base

base

- Base64 coding
- BC
- Binary exclusive OR
- Birthday attack/birthday paradox
- Bit length
- Blocks
- Books
- Bounding box
- Brute Force Analysis
- Brute-force attack
- Byte addition
- Caesar encryption algorithm
- Card game
- Cascade
- Cascading cipher
- CBC mode
- Certificate
- Challenge
- Challenge-response demonstration
- Chi<sup>2</sup> distribution
- Chinese remainder theorem (CRT)
- Chosen-plaintext attack
- Ciphertext
- Ciphertext-only attack
- Clipboard
- Codings
- Coin toss
- Compress
- Configuration file
- Congruence generator
- Contact
- Context/Subsumption/Overview
- Copyright
- Correlation
- Cryptanalysis
- Crypto libraries
- Cryptography
- Cryptology
- CryptTool
- Curve chart
- cv cryptovision
- Decompress
- Decryption
- Default settings

Anzeigen

### 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**

Dividing of  $3 \times 8 to  $4 \times 6$  bit.$

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

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

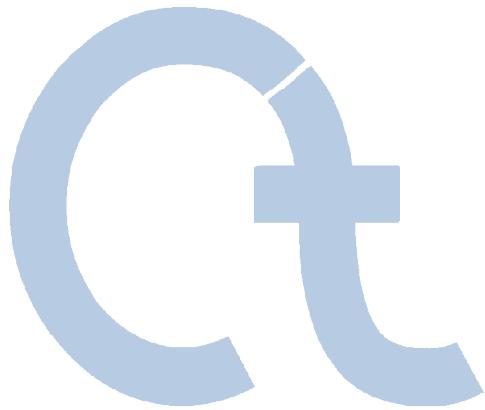
Get the characters from Base64 coding table. (defined in an IETF standard)

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
<ul style="list-style-type: none"><li>Arbitrary binary data can be represented with a 6-bit char set.<ul style="list-style-type: none"><li>No problems with 7-bit char set restrictions.</li><li>No problems with line length restrictions or special control characters.</li></ul></li><li>Only an enlargement of about 33 % (instead of an enlargement of 100 % when encoding to hexadecimal values).</li></ul>	<ul style="list-style-type: none"><li>No support for distribution of big files.</li><li>Enlargement of about 33 % (in comparison to the original file). Only UUencode:<ul style="list-style-type: none"><li>No EBCDIC support.</li><li>No defined standards.</li></ul></li></ul>

# Content



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



# Future CrypTool Development (1)

Planned after release CrypTool 1.4.30 (see readme file)

CT1 = CrypTool 1.x  
CT2 = CrypTool 2.0  
JCT = JCrypTool

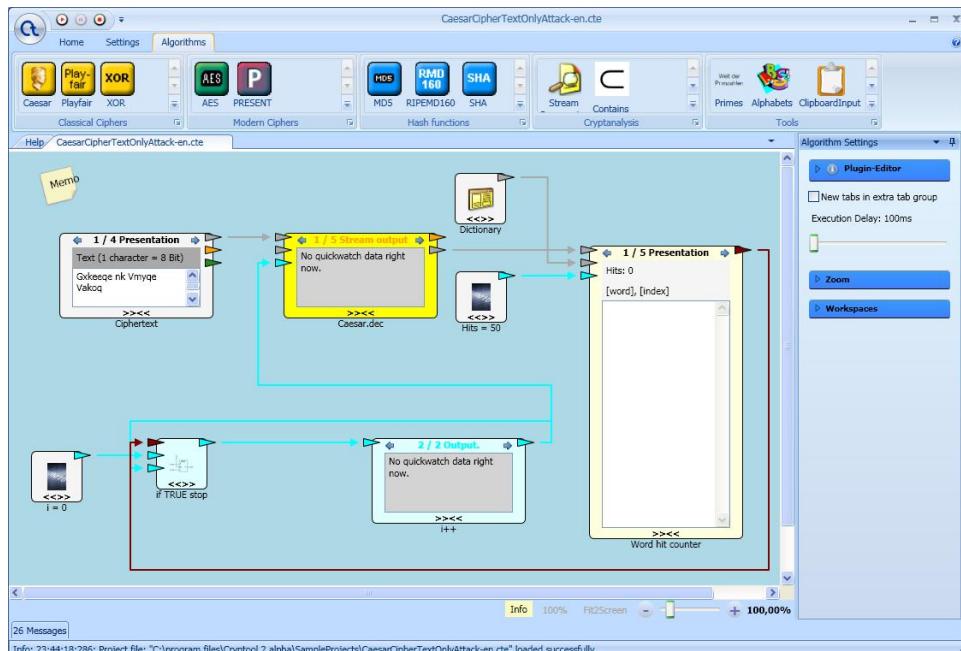
- CT1 Sigaba, FIPS test to investigate more than 2500 bytes, ...
- JCT Tri-partite key agreements
- JCT Visualization of the interoperability between S/MIME and OpenPGP formats
- JCT Analysis of entropy
- JCT Grille, Autokey Vigenère, interactive cryptanalysis of classic ciphers
- JCT Analysis of transposition ciphers using the ACO algorithm
- JCT Visualization of zero-knowledge proofs
- JCT Cube attack (I. Dinur and A. Shamir, "Cube Attacks on Tweakable Black Box Polynomials", 2008)
- JCT Action-History with the additional feature, to create any product cipher
- CT2 Comprehensive visualization on the topic of prime numbers
- CT2 Demonstration of Bleichenbacher's RSA signature forgery
- CT2 Demonstration of virtual credit card numbers (approach against credit card abuse)
- CT2 WEP encryption and WEP analysis
- CT2 Mass pattern search
- CT2 Demonstration of SOA security (SOAP messages according WS Security between the participants)
- CT2 Framework to create and analyze LFSR stream ciphers
- CT2 Graphical design oriented mode for beginners plus expert mode
- CT2/JCT Creation of a command line version for batch processing
- CT2/JCT Modern pure plugin architecture with loading of plugins
- All Further parameterization / Increasing the flexibility of present algorithms
- Idea Visualization of the SSL protocol
- Idea Demonstration of visual cryptography
- Idea Integration of crypto library crypto++ from Wei Dai



# Future CrypTool Development (2)

## In Progress (see readme file)

1. JCT: Port and redesign of CrypTool in Java / SWT / Eclipse 3.5 / RPC
  - see: <http://jcryptool.sourceforge.net>
  - Milestone 4a available for users and developers (February 2009) // M5: planned for August 2009
2. CT2: Port and redesign of the C++ version with C# / WPF / VS2008 / .NET 3.5
  - direct successor of current releases: allows visual programming, ...
  - Beta available for users and developers (since July 2008, permanently updated)
3. C2L: Direct port of the C++ version to Linux with Qt4 (very slow progress)
  - see: <http://www.cryptoolinux.net>

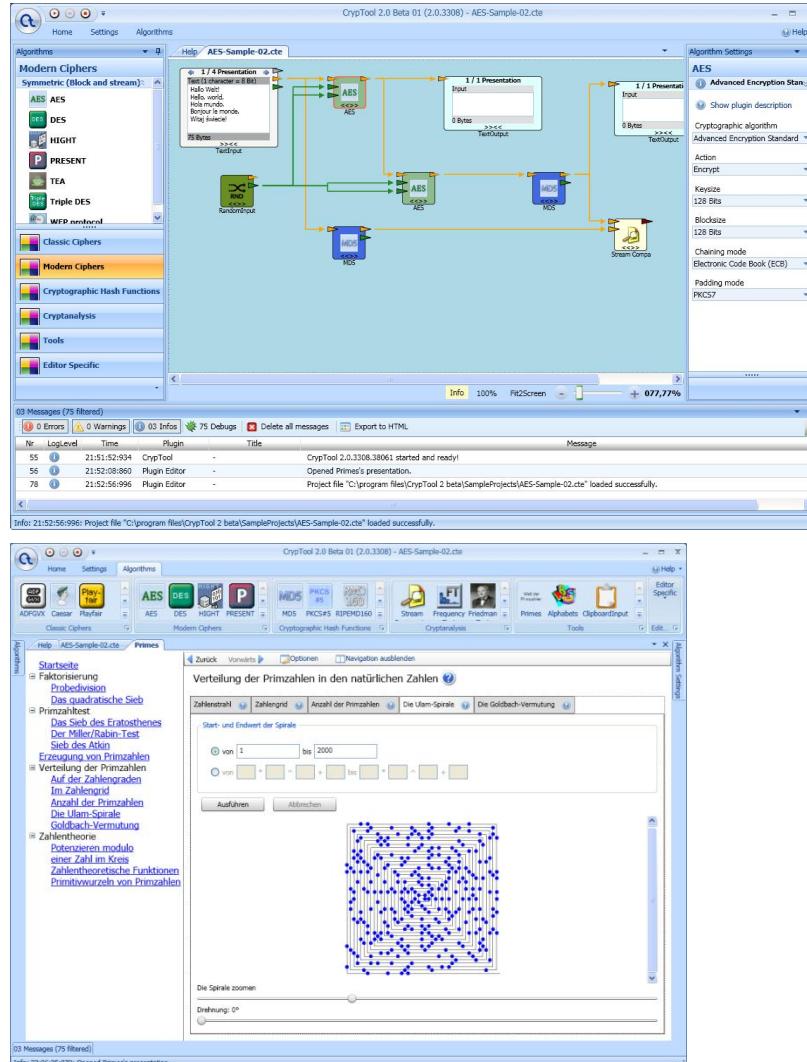


CrypTool 2 (CT2)



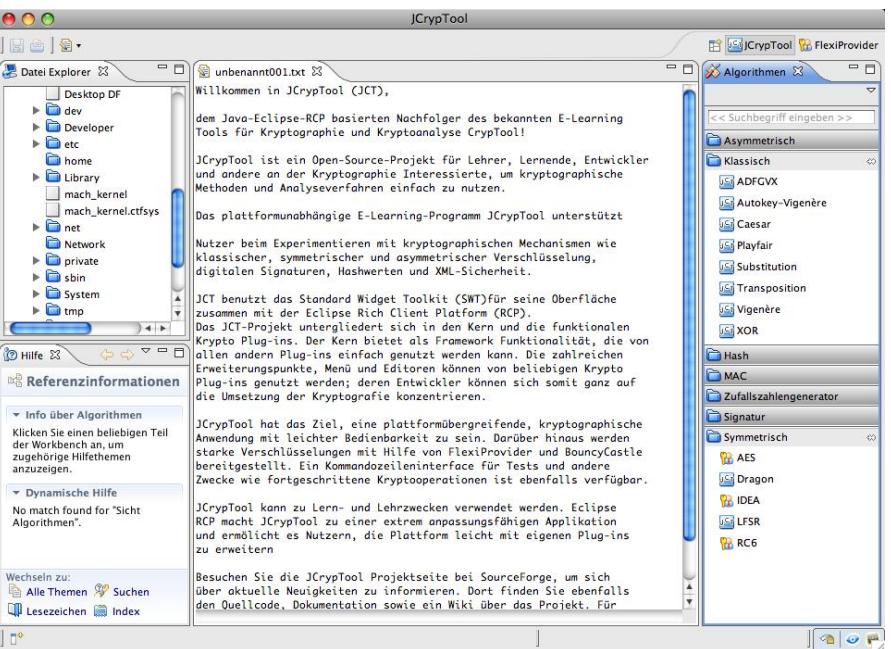
JCrypTool (JCT)

# Future CrypTool Development (3)



CrypTool 2 (CT2)

CrypTool 1.4.30



JCrypTool (JCT)

Page 103

# CrypTool as a Framework

## Proposal

- Re-use the comprehensive set of algorithms, included libraries and interface elements as foundation
- Free of charge training in Frankfurt, how to start with CrypTool development
- Advantage: Your own code does not „disappear“, but will be maintained

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

- CrypTool 1.4.30: Visual C++ .net (= VC++ 9.0)(= Visual Studio 2008 Standard)
- Description for developers: see readme-source.txt
- Download: Sources and binaries of releases.  
To get sources of current betas, please see subversion repository.

Development environments for CT2 and JCT:

- CT2 – C# version: .NET with Visual Studio 2008 Express Edition (free), WPF und Perl
- Java – Java version: Eclipse 3.5, 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, test)
  - For the current C/C++ project
  - For the new projects (preferred):
    - C# project: „CrypTool 2.0“ = CT2
    - Java project: „JCrypTool“ = JCT
  - Especially University faculties using CrypTool for educational purposes are invited to contribute to the further development of CrypTool.
- Significant contributions can be referenced by name (in help, readme, about dialog and on the CrypTool web site).
- Currently CrypTool is being downloaded more than 4000 times a month (with a little bit more than 50 % getting the English version).



# CrypTool – Summary

- *THE* e-learning program for cryptology
- Over more than 10 years a successful open source project
- More than 200,000 downloads
- International utilisation in schools, universities as well as companies and government agencies
- Extensive online help and documentation
- Available for free and multi-language support

# Contact

## Prof. Bernhard Esslinger

University of Siegen  
Faculty 5, Economics and Business Computing

Deutsche Bank AG  
Director, IT Security Manager

[esslinger@fb5.uni-siegen.de](mailto:esslinger@fb5.uni-siegen.de)

[www.cryptool.org](http://www.cryptool.org)  
[www.cryptool.com](http://www.cryptool.com)  
[www.cryptool.de](http://www.cryptool.de)  
[www.cryptool.es](http://www.cryptool.es)  
[www.cryptool.pl](http://www.cryptool.pl)

Additional contacts: See readme within the CrypTool folder



# Additional Literature

As 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
- Udo Ulfkotte, "*Wirtschaftsspionage*", 2001, Goldmann [German]
- Johannes Buchmann, "*Introduction to Cryptography*", 2nd edition, 2004, Springer
- Claudia Eckert, "*IT-Sicherheit*", 5th edition, 2008, Oldenbourg [German]
- A. Beutelspacher / J. Schwenk / K.-D. Wolfenstetter, "*Moderne Verfahren der Kryptographie*", 5th edition, 2004, Vieweg [German]
- [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
- 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, ...)
- 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*", 2009, Pearson, chapter 14 [German]
  - Wikipedia: [http://en.wikipedia.org/wiki/Risk\\_management](http://en.wikipedia.org/wiki/Risk_management)
  - CrypTool Site: <http://cryptool.com/index.php/en/cryptool-for-awareness-aboutmenu-74.html>

# www.cryptool.org / .com / .de / .es / .pl

The screenshot shows the official website for CrypTool. At the top, there's a navigation bar with links for About, Features, Screenshots, Documentation, and Download, along with language icons for German, English, French, Spanish, and Polish. Below the navigation bar, a yellow banner displays the latest stable version (1.4.21) and a download link. The main content area features a large section titled "CrypTool Introduction" with a sub-section "Selected Landmark 2008 in: Germany Land of Ideas". The introduction text describes CrypTool as a free e-learning application for Windows used for applying and analyzing cryptographic algorithms. It highlights various features like classic and modern algorithms, visualization, cryptanalysis, and entropy measuring. A sidebar on the left contains a "About" menu with links to the introduction, education, awareness, print media coverage, awards, contributors, related projects, and contact information.

## CrypTool Introduction

The application CrypTool is a free e-learning application for Windows. You can use it to apply and analyze cryptographic algorithms. The current version of CrypTool is used all over the world. It supports both contemporary teaching methods at schools and universities as well as awareness training for employees.

The current version offers **beside others** the following highlights:

- Numerous classic and modern cryptographic algorithms (encryption and decryption, key generation, secure passwords, authentication, secure protocols, ...)
- Visualisation of several methods (e.g. Caesar, Enigma, RSA, Diffie-Hellman, digital signatures, AES)
- Cryptanalysis of certain algorithms (e.g. Vigenère, RSA, AES)
- Crypt-analytical measuring methods (e.g. entropy, n-grams, autocorrelation)
- Auxiliary methods (e.g. primality tests, factorisation, base64 coding)
- Tutorial about number theory
- Comprehensive online help
- Supportive script with further information about cryptology

From its original use of information security training for a company, CrypTool has developed into an outstanding open source project for cryptology related topics.

Since spring 2008, the CrypTool project has been operating the [Crypto Portal for Teachers](#). Thus far, the portal is only available in German and is intended to act as a platform for teachers to share teaching materials about cryptology and related links.

Currently the CrypTool team is working on two future projects intended to become the successors of the current

**About**

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

**Selected Landmark 2008 in:**  
**Germany**  
**Land of Ideas**

Especially the didactical brilliant tutorials and the use of new media

# www.cryptool-online.org

The newest member in the family of CrypTool related websites:

- CrypTool site (CT1)
- CT2 developer site
- JCT developer site
- CryptoPortal for teachers (currently only in German)
- CrypTool-Online (try cryptographic methods within your browser).



# www.cryptoportal.org

The screenshot shows the homepage of the CRYPTOPORTAL website for teachers. The header features a background image of a child writing on a chalkboard with various mathematical and technical drawings. The title "CRYPTOPORTAL" is prominently displayed in large white letters, with "für Lehrer" underneath. A navigation bar below the title includes links for "Über", "Unterrichtsmaterial", "Linksammlung", "Registrierung", "Cryptoool", and "Einloggen".

**Filterkriterien**

Land: alle Länder

Schultyp: alle Schultypen

Autor: alle Autoren

Material enthält folgenden Text:

Filtern Zurücksetzen

**Unterrichtsmaterial**

**[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 Protokolle [...] [\[...\]](#)

[Orginalpraesentation.pdf](#) 14 mal heruntergeladen

**The teacher's portal currently exists in German only. Help for an English version of this portal is welcome.**

# Download Software and CrypTool Script

CrypTool - Educational Tool for Cryptography and Cryptanalysis - Script - Mozilla Firefox  
Datei Bearbeiten Ansicht Chronik Lesezeichen Extras Hilfe  
CrypTool - Educa... SourceForge.net: Pr... CrypTool 2 - Archite... CrypTool-Online / Lin... Cryptoportal für Leh... SourceForge.net: jcr...  
http://www.cryptool.com/index.php/en/script-documentationmenu-69.html

**CRYpTOOL**  
About Features Screenshots Documentation Download Latest stable version: 1.4.21 Download  
Stable Beta 1.4.30 download & test now

**Documentation**  
▪ Presentations  
▪ Script  
▪ Crypto History  
▪ Links  
▪ [Download script](#)

**Script**  
Current version: 9th edition, July 2008  
Download script

In this script delivered with the CrypTool application you will find predominantly mathematically oriented information on using cryptographic procedures. The main chapters have been written by various authors and are therefore independent from one another. At the end of each chapter you will find bibliographical references and web links.  
• The first chapter explains the principles of symmetric and asymmetric encryption and describes shortly the

script-en.pdf - Adobe Reader  
File Edit View Document Tools Window Help

Bookmarks

- Overview
- Contents
- Preface to the 10th Edition of the CrypTool Script
- Introduction -- How do the Script and the Program Play together?
- 1 Encryption Procedures
  - 1.1 Symmetric encryption
  - 1.2 Asymmetric encryption
  - 1.3 Hybrid procedures
  - 1.4 Further details
  - Bibliography
  - Web links
- 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 Crypto 2020 --- Perspectives for long-term cryptographic security
- A Appendix
  - A.1 CrypTool Menus
  - A.2 Authors of the CrypTool Script
  - A.3 Bibliography of Movies and Fictional Literature with Relation to Cryptography, Books for Kids with Collections of Simple Ciphers
  - A.4 Learning Tool for Elementary Number Theory
  - A.5 Using Sage with this Script
  - GNU Free Documentation License
  - List of Figures
  - List of Tables
  - Index

The CrypTool Script:  
Cryptography, Mathematics and More  
Background reading  
for the free eLearning program CrypTool  
(10th edition – distributed with CrypTool version 1.4.30)

Copyright (c) Prof. Bernhard Esslinger (co-author and editor)  
and the CrypTool Development Team, 1998-2009  
Frankfurt am Main, Germany  
[www.cryptool.org](http://www.cryptool.org)  
July 9, 2009

CrypTool - Educational Tool for Cryptography and Cryptanalysis - Download - Mozilla Firefox  
Datei Bearbeiten Ansicht Chronik Lesezeichen Extras Hilfe  
CrypTool - Educa... SourceForge.net: Pr... CrypTool 2 - Archite... CrypTool-Online / Lin... Cryptoportal für Leh... SourceForge.net: jcr...  
http://www.cryptool.com/index.php/en/download-topmenu-63.html

**CRYpTOOL**  
About Features Screenshots Documentation Download Latest stable version: 1.4.21 Download  
Stable Beta 1.4.30 download & test now

**Download**

**CrypTool 1.4.30 - Beta 03**  
This is a stable beta of the next release of CrypTool 1.x.  
Please download and give us feedback:  
• [CrypTool 1.4.30 Beta 03 - English version](#)  
• [CrypTool 1.4.30 Beta 03 - German version](#)