

CrypTool

A free software program

- **for creating awareness of IT security issues**
- **for learning about and obtaining experience of cryptography**
- **for demonstrating encryption algorithms and analysis procedures**

www.cryptool.de
www.cryptool.com
www.cryptool.org



Contents

Introduction

1. What is CrypTool?
2. Why CrypTool?
3. Audience

CrypTool Overview

1. Features
2. Software package contents
3. New in release 1.3.xx

Examples of use

1. Hybrid encryption visualised
2. Digital signature visualised
3. Attack on RSA encryption with short RSA modulus
4. Analysis of encryption used in the PSION 5 PDA
5. Demonstration of weak DES keys
6. Locating key material (keyword: NSAKEY)

Contact address

Introduction

1. What is CrypTool?

- a freeware Program with graphical user interface
- a tool for applying and analysing cryptographic algorithms
- with extensible online help, understandable without deep crypto knowledge
- contains nearly all state of the art crypto algorithms
- “playful” introduction to modern and classical cryptography
- not a “hacker tool”

2. Why CrypTool?

- origin in Deutsche Bank’s IT security awareness program
- developed in co-operation with universities
- improve IT security related courses in universities and companies

3. Audience

- target group: students of computer science, commercial IT and mathematics
- also for: interested computer users and application developers
- prerequisites: secondary school mathematics or programming skills

CrypTool Overview

1. Features

Cryptography

Classical algorithms

- Caesar
- Vigenère
- Hill
- Monoalphabetic substitution
- Homophonic substitution
- Playfair
- Permutation
- Addition
- XOR
- Vernam

To facilitate performing text book examples with CrypTool

- alphabet can be configured
- treatment of white space etc. configurable

Cryptoanalysis

Attacks on classical algorithms

- ciphertext only
 - Caesar
 - Vigenère
 - Addition
 - XOR
- known plaintext
 - Hill
 - Playfair
- manual
 - mono-alphabetic substitution

Supporting analysis procedures

- entropy, floating frequency
- histogram, n-gram analysis
- auto-correlation
- ZIP compression test

CrypTool Overview

1. Features

Cryptography

Modern symmetric algorithms

- IDEA, RC2, RC4, DES, 3DES
- last round AES candidates
- AES (=Rijndael)

Asymmetric algorithms

- RSA with X.509 certificates
- RSA demonstration
 - to facilitate performing text book examples with CrypTool
 - alphabet and block length configurable

Hybrid encryption

- RSA combined with AES encryption
- visualised by an interactive data flow diagram

Cryptoanalysis

Brute force attack on symmetric algorithms

- implemented for all algorithms
- assumption:
 - entropy of plain text small
- search space limited to 20 bit

Attack on RSA encryption

- factor RSA modulus
- workable for bit lengths ≤ 250

Attack on hybrid encryption

- attack on RSA (see below) or
- attack on AES (see above)

CrypTool Overview

1. Features

Cryptography

Digital Signature

- RSA with X.509 certificates
 - signature procedure visualised by an interactive data flow diagram
- DSA with X.509 certificates
- Elliptic Curve DSA, Nyberg-Rueppel

Hash functions

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

Random generators

- SECUDE
- X^2 modulo N
- Linear Congruence Generator (LCG)
- Inverse Congruence Generator (ICG)

Cryptoanalysis

Attack on RSA Signature

- RSA module factorisation
- workable up to approx. 250 bit

No attack implemented

Random data analysis

- FIPS-PUB-140-1 test battery
- periodicity, Vitany, entropy
- histogram, n-gram analysis
- auto-correlation
- ZIP compression test

CrypTool Overview

2. Software package contents

CrypTool program

- all functions integrated in *one* program with uniform graphical user interface
- platforms: Win32 and Linux with WINE emulator
- cryptography based on Secude library (www.secude.com)
- arbitrary precision arithmetic: Miracl library (<http://indigo.ie/~mscott/>)

AES-Tool

- standalone program for AES encryption

Extensive online help (Winhelp)

- context sensitive online help for *all* program functions
- detailed usage examples for many program features

Script (PDF) with background information on

- encryption algorithms • prime numbers • digital signature
- elliptic curves • public key certification • elementary number theory

Short story “Dialogue of the Sisters” by Dr. C. Elsner



completely bilingual
English
German

CrypTool Overview

3. New in release 1.3.xx

Most important changes (details: see ReadMe-en.txt):

Release 1.3.00 published January 2002

- completely bilingual English/German
- dialog box consistency and comprehensibility improved
- Windows 9x file size limit removed
- homophonic and permutation encryption
- random generators, random data analysis (FIPS-140-1, periodicity, n-gram)
- AES-Tool: create self-decrypting files (AES)
- demonstration: number theory and RSA crypto system (further improved in 1.3.03)
- PKCS#12 export/import for PSEs

Release 1.3.03 published September 2002

- visualisation of hybrid encryption and decryption
- visualisation of signature creation and verification
- hash value calculation of large files (without loading them into memory)
- visualisation of the sensitivity of hash functions to changes in the hashed data
- short story “Dialogue of the Sisters” by Dr. C. Elsner included

Examples of use

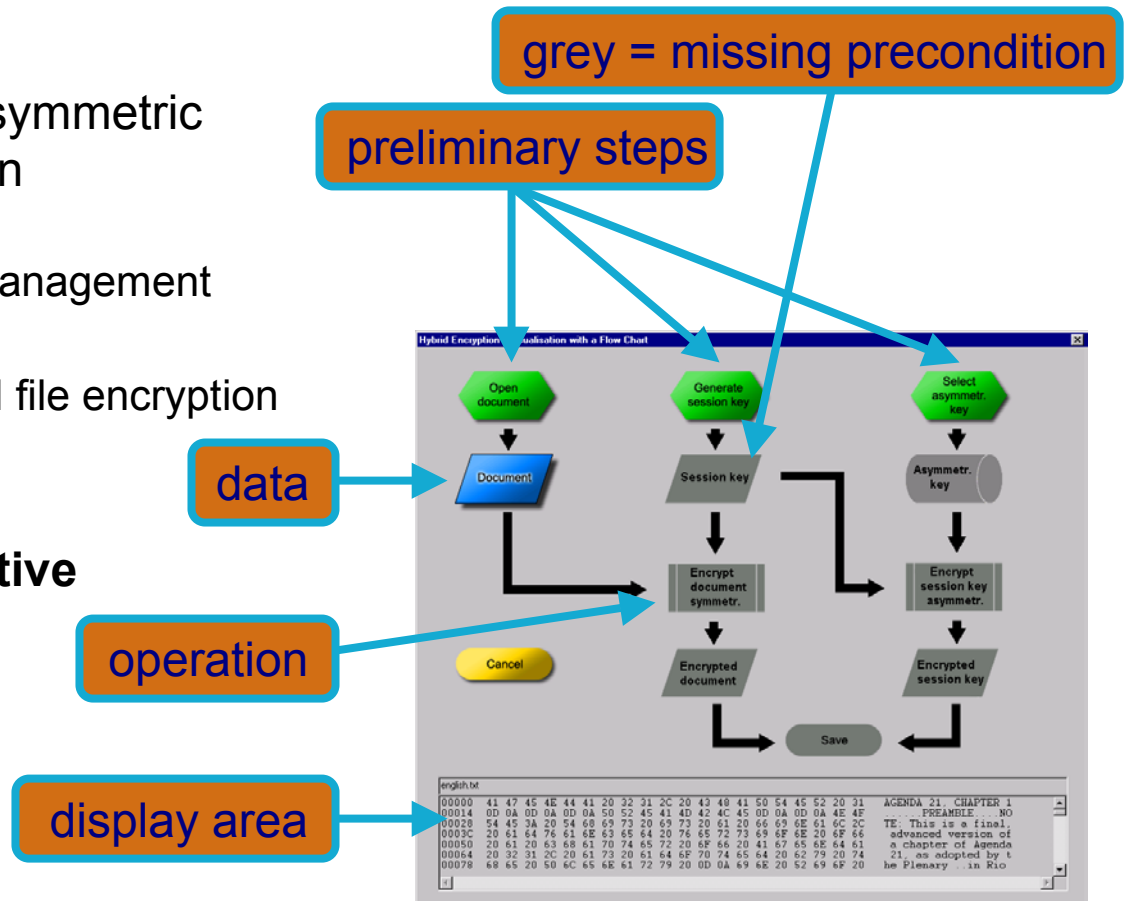
1. Hybrid encryption visualised

Hybrid encryption

- combines advantages of symmetric and asymmetric encryption
 - speed
 - simple and scalable key management
- widely used in practice
 - e-mail (S/MIME, PGP) and file encryption
 - SSL (https)

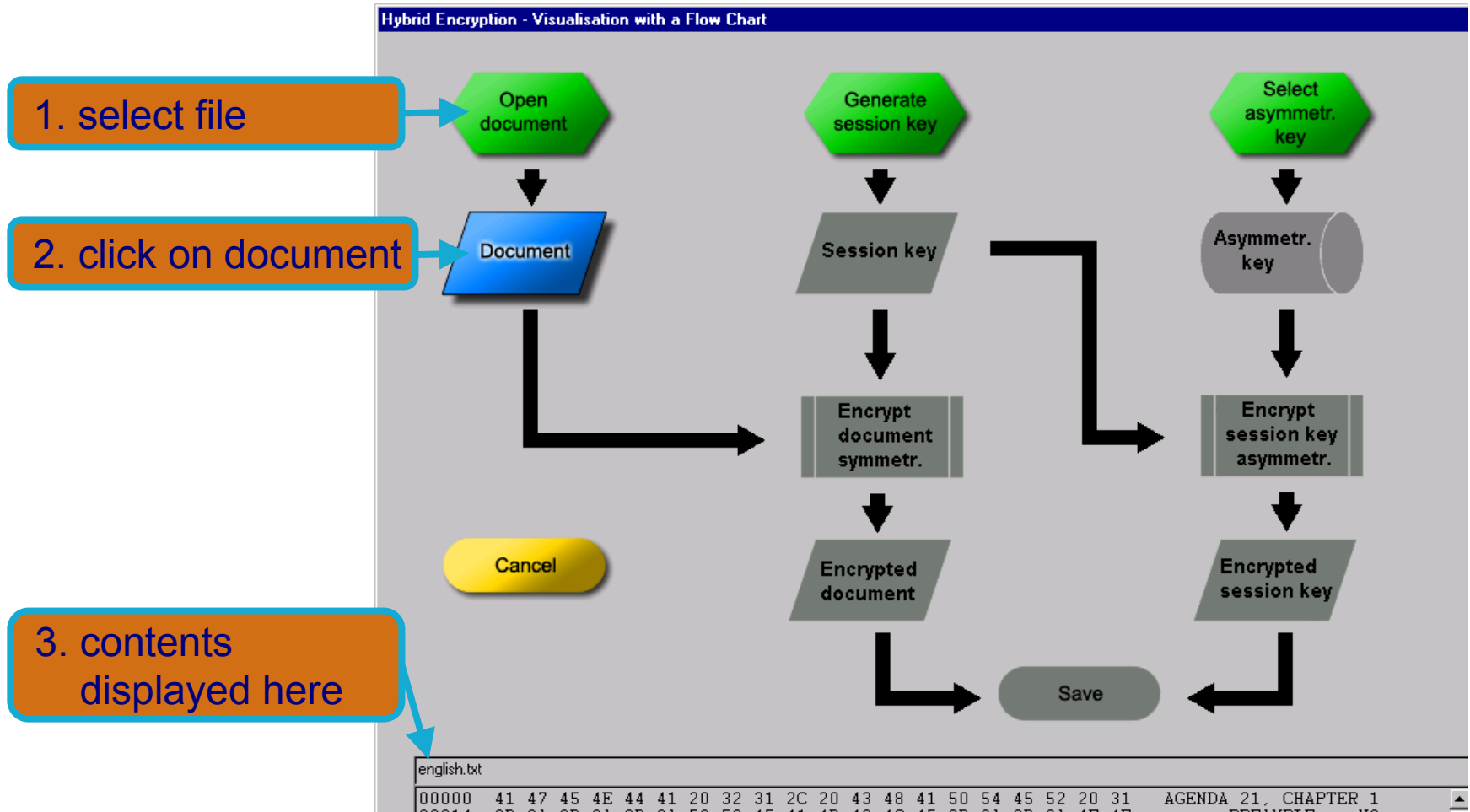
Visualisation by an interactive data flow diagram

- playful learning leads to deeper understanding



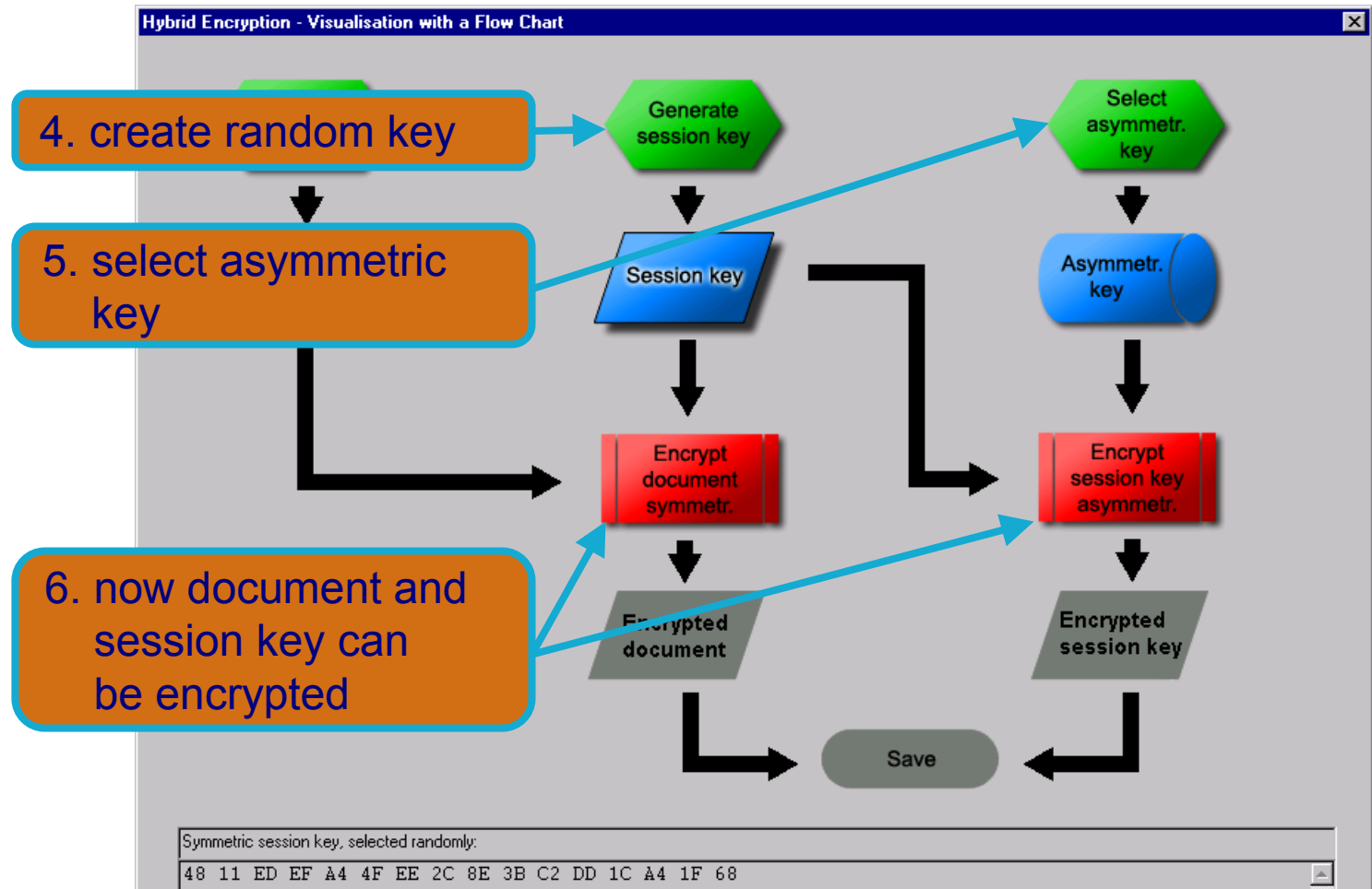
Examples of use

1. Hybrid encryption visualised: Preparation



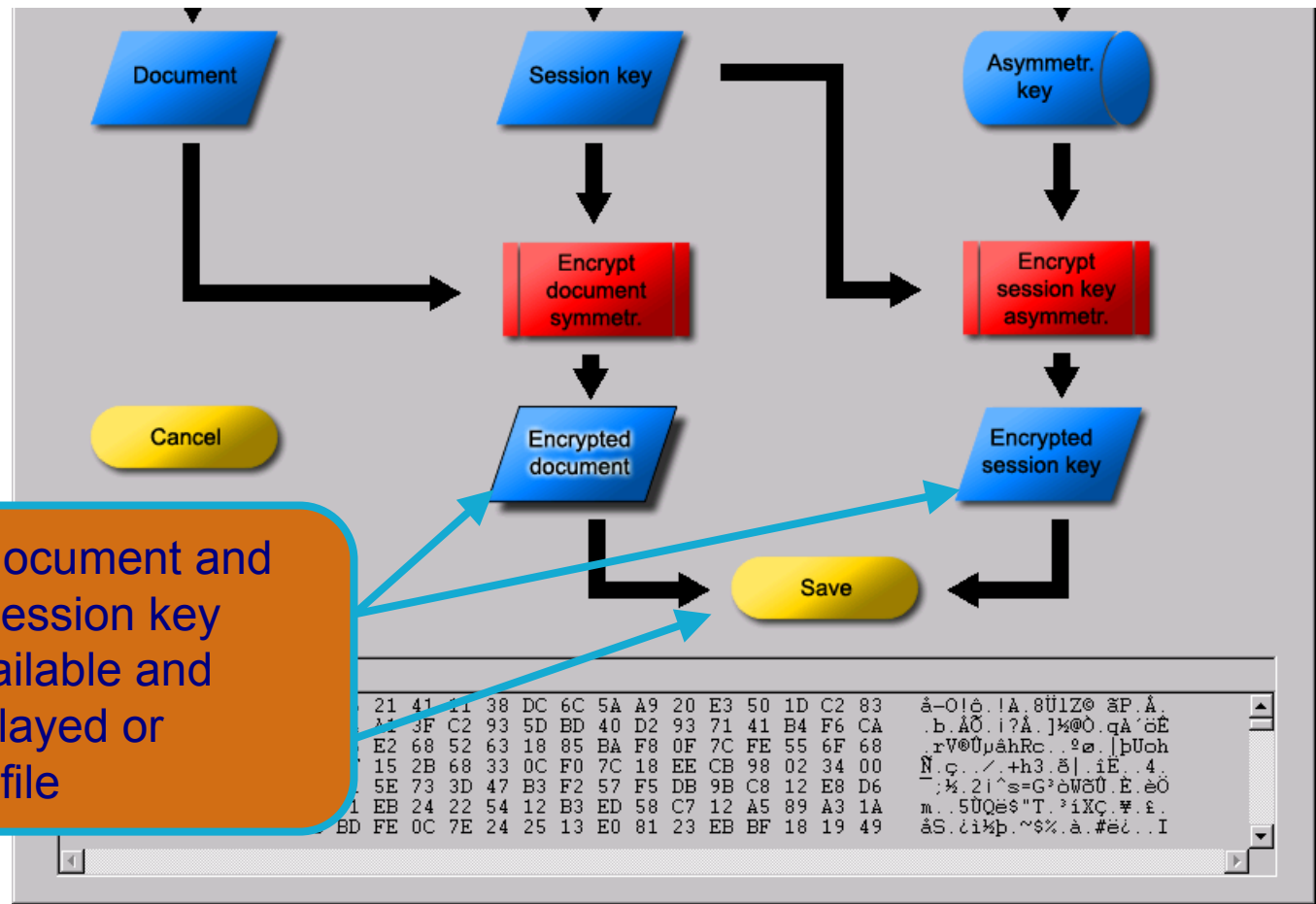
Examples of use

1. Hybrid encryption visualised: Cryptography



Examples of use

1. Hybrid encryption visualised: Result



7. encrypted document and encrypted session key are now available and can be displayed or written to a file

Examples of use

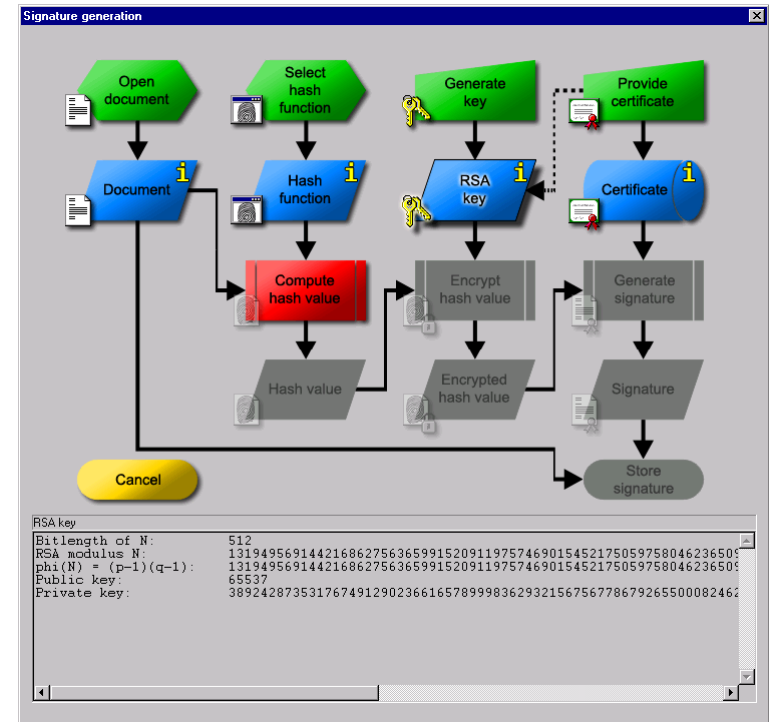
2. Digital signature visualised

Digital signature

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

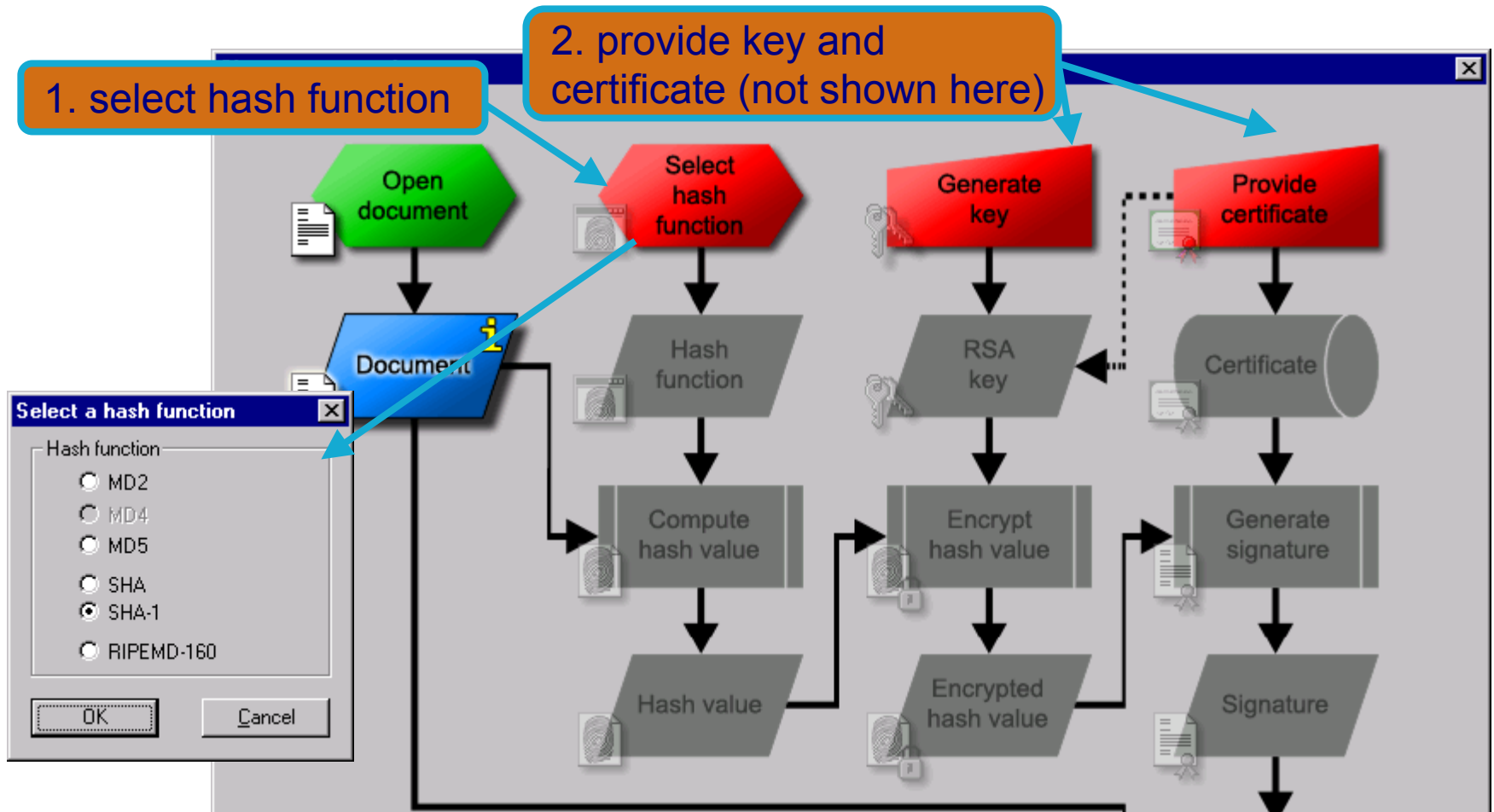
Visualisation in CrypTool

- interactive data flow diagram
- similar to the visualisation of hybrid encryption



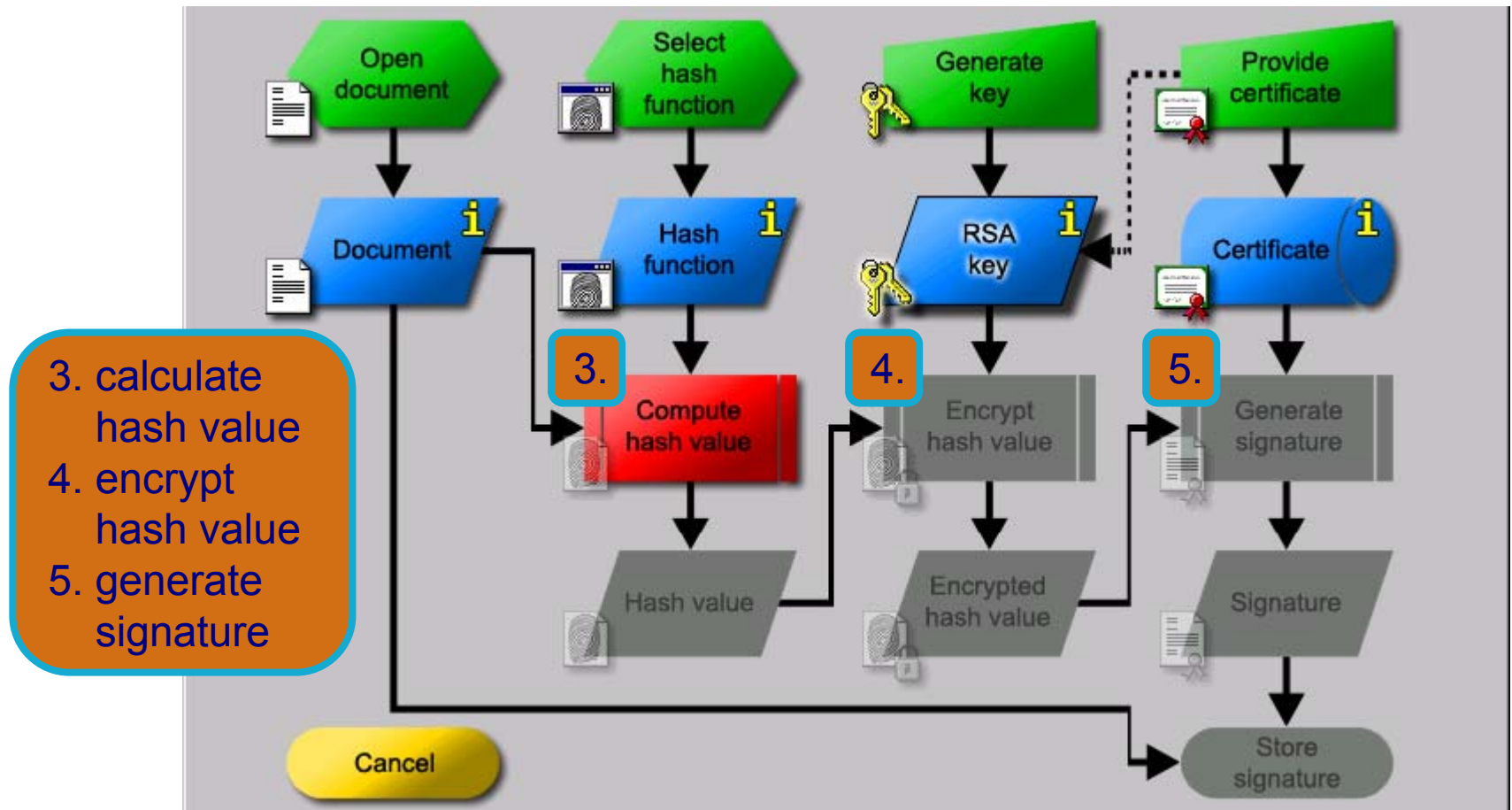
Examples of use

2. Digital signature visualised: Preparation



Examples of use

2. Digital signature visualised: Cryptography

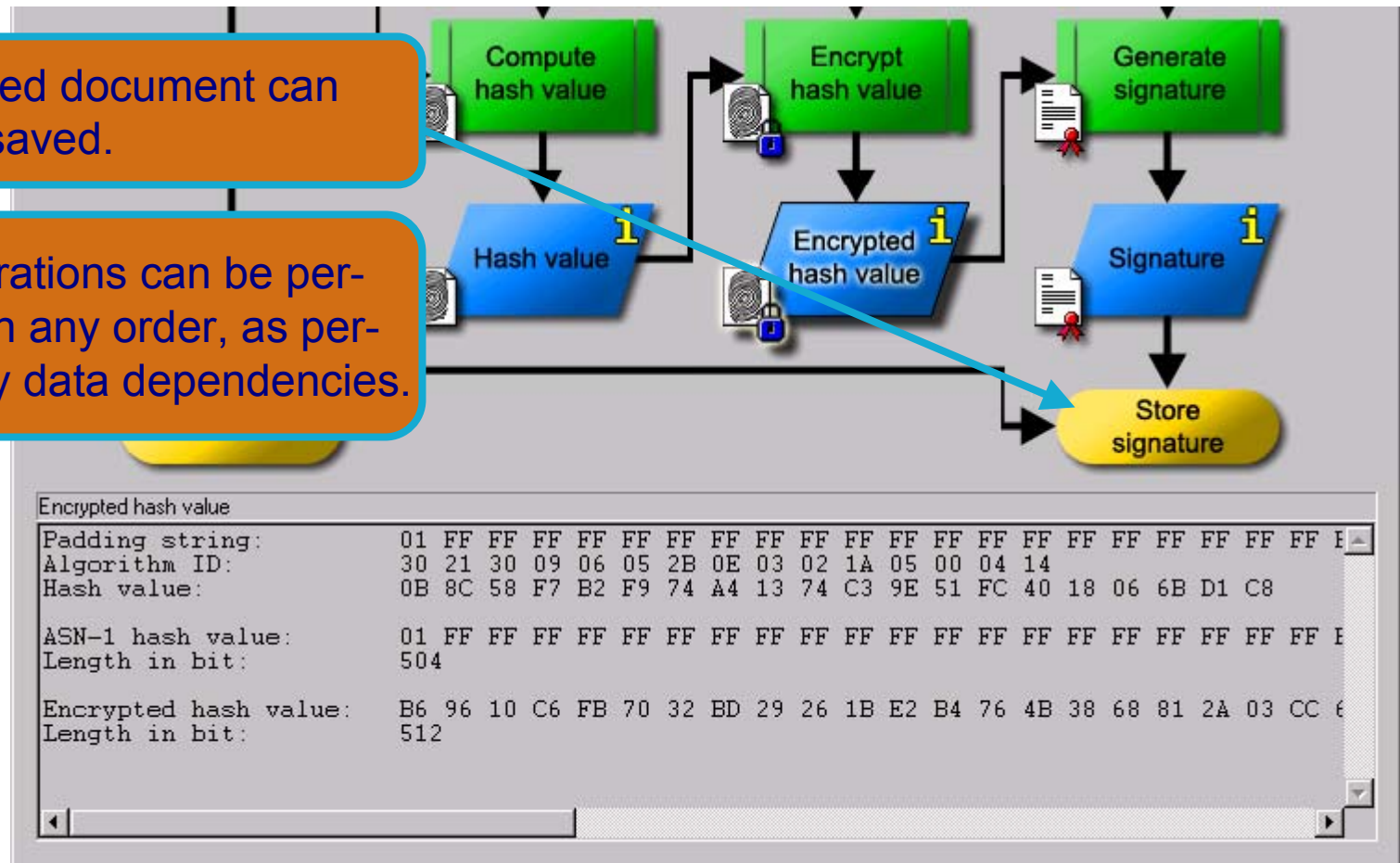


Examples of use

2. Digital signature visualised: Result

The signed document can now be saved.

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



Examples of use

3. Attack on RSA encryption with short RSA modulus

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

- public key
 - RSA modulus (95bit) $N = 63978486879527143858831415041$
 - public exponent $e = 17579$
- cipher text (block length = 14):
 - $C_1 = 45411667895024938209259253423,$
 - $C_2 = 16597091621432020076311552201,$
 - $C_3 = 46468979279750354732637631044,$
 - $C_4 = 32870167545903741339819671379$
- 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 “factorise the RSA modulus” yields prime factors p and q where $pq = N$
- based on that information 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 is workable for RSA moduli up to 250 bit

Examples of use:

3. Short RSA modulus: enter public RSA parameters

The RSA Cryptosystem

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

☐ Choose two prime numbers p and q . The number $N = pq$ is the public RSA modulus and $\phi(N) = (p-1)(q-1)$ is the Euler number. Public key e is coprime to $\phi(N)$. The private key $d = e^{-1} \pmod{\phi(N)}$ is calculated from this.

☒ For the purpose of data encryption or certificate checking it is sufficient to enter the public RSA parameters: the RSA modulus N and the public key e .

Factorisation attack

You may try to factorise the public RSA modulus N into its primes p and q .

Factorise RSA modulus...

RSA parameters

RSA modulus N (public)

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

Public key e

Private key d

☐ RSA encryption using e / decryption using d

2. factorise

1. enter RSA parameters N and e



Examples of use:

3. Short RSA modulus: factorise RSA modulus

Factorisation of a Number

Algorithms for factorisation:

- ☒ Brute-force method
- ☒ Brent algorithm
- ☒ Pollard method
- ☒ Williams method
- ☒ Lenstra algorithm
- ☒ Quadratic sieve method

Input:

Enter the number to be factorised:

63978486879527143858831415041

Factorisation:

The factorisation is represented in the format $\langle z_1^{a_1} * z_2^{a_2} * \dots * z_n^{a_n} \rangle$. Composite numbers are appearing in red color.

Last factorisation through: Pollard

Time required: 0,981 seconds.

Factorisation result:

145295143558111 * 440334654777631

Details

Close

3. factorisation yields p and q

CrypTool

The RSA modulus N has been successfully factorised into the primes p and q!
You can now perform the RSA operation with the secret key d.
For this purpose just click the button Decrypt.

OK

Examples of use:

3. Short RSA modulus: determine private key d

The RSA Cryptosystem

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

- ☒ Choose two prime numbers p and q . The number $N = pq$ is the public RSA modulus and $\phi(N) = (p-1)(q-1)$ is the Euler number. Public key e is coprime to $\phi(N)$. The private key $d = e^{-1} \pmod{\phi(N)}$ is calculated from this.
- ☐ For the purpose of data encryption or certificate checking it will do with the published RSA parameter: the RSA modulus N and the public key e .

Prime number entry

Prime number p : 145295143558111

Prime number q : 440334654777631

Generate prime numbers

RSA parameters

RSA modulus N : 63978486879527143858831415041 (public)

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

Public key e : 17579

Private key d : 10663687727232084624328285019

Update parameters

RSA encryption using e / decryption using d

Input as ☒ text ☐ numbers

Options for alphabet and number system...

Enter either as text or as numbers in the format number[1] # ... # number[n] (numbers of base 1240752)

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

5. adjust options



Examples of use:

3. Short RSA modulus: adjust options

Options for RSA Encryption

Text options

☐ All 256 ASCII characters

☒ Specify alphabet: Number of characters: 27

ABCDEFGHIJKLMNOPQRSTUVWXYZ

Block length

The number of characters that are encrypted with each RSA operation.
The maximum size is subject to the length of the RSA modul N in bit, the number of characters in the alphabet and the method used for the coding.

Block length in characters: (Maximum block length 2 characters)

Number system

The numbers for RSA encryption and decryption will be represented in the following number system

☒ Decimal ☐ Binary ☐ Octal ☐ Hexadecimal

OK Cancel

6. select alphabet

7. select coding method

8. select block length



Examples of use:

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	

[Update parameters](#)

RSA encryption using e / decryption using d

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

Ciphertext coded in numbers of base 10

7091621432020076311552201 # 46468979279750354732637631044 # 32870167545903741339819671279

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

00000000000001401202118011200 # 00000000000001421130205181900 # 000000000000011805001301

Output text from the decryption (into segments of size 8; the symbol '#' is used as separator).

NATURAL # NUMBERS # ARE MADE # BY GOD

Plaintext

NATURAL NUMBERS ARE MADE BY GOD

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

9. enter cipher text

10. decrypt

Examples of use

4. Analysis of encryption used in the PSION 5 PDA

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



Starting point: an encrypted file on the PSION

Requirements

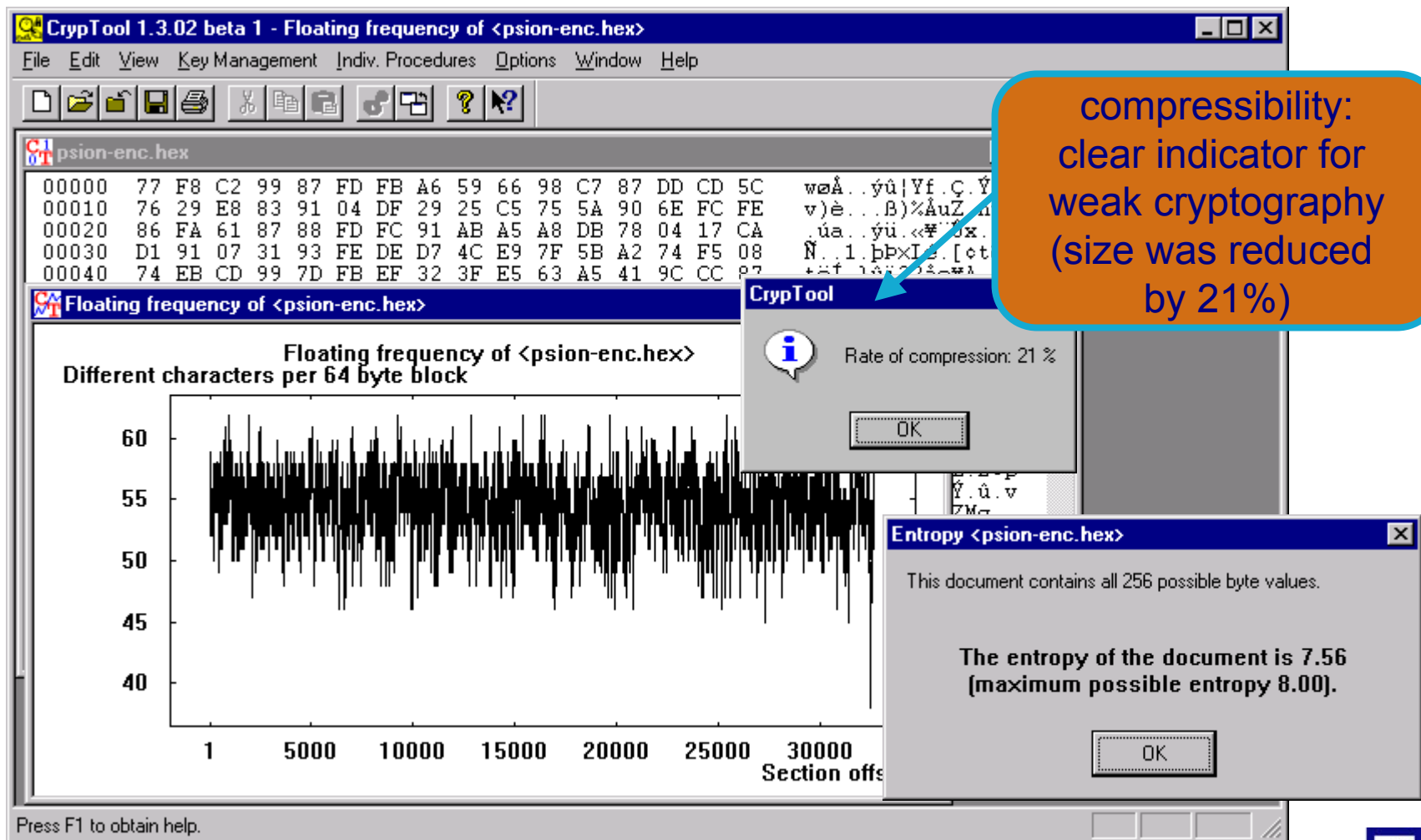
- encrypted English or German text
- depending on method and key length, 100 bytes up to several kB of text

Procedure

- pre-analysis
 - entropy
 - floating entropy
 - compression test
 - auto-correlation
 - try out automatic analysis with classical methods
- } ⇒ **probably classical encryption algorithm**

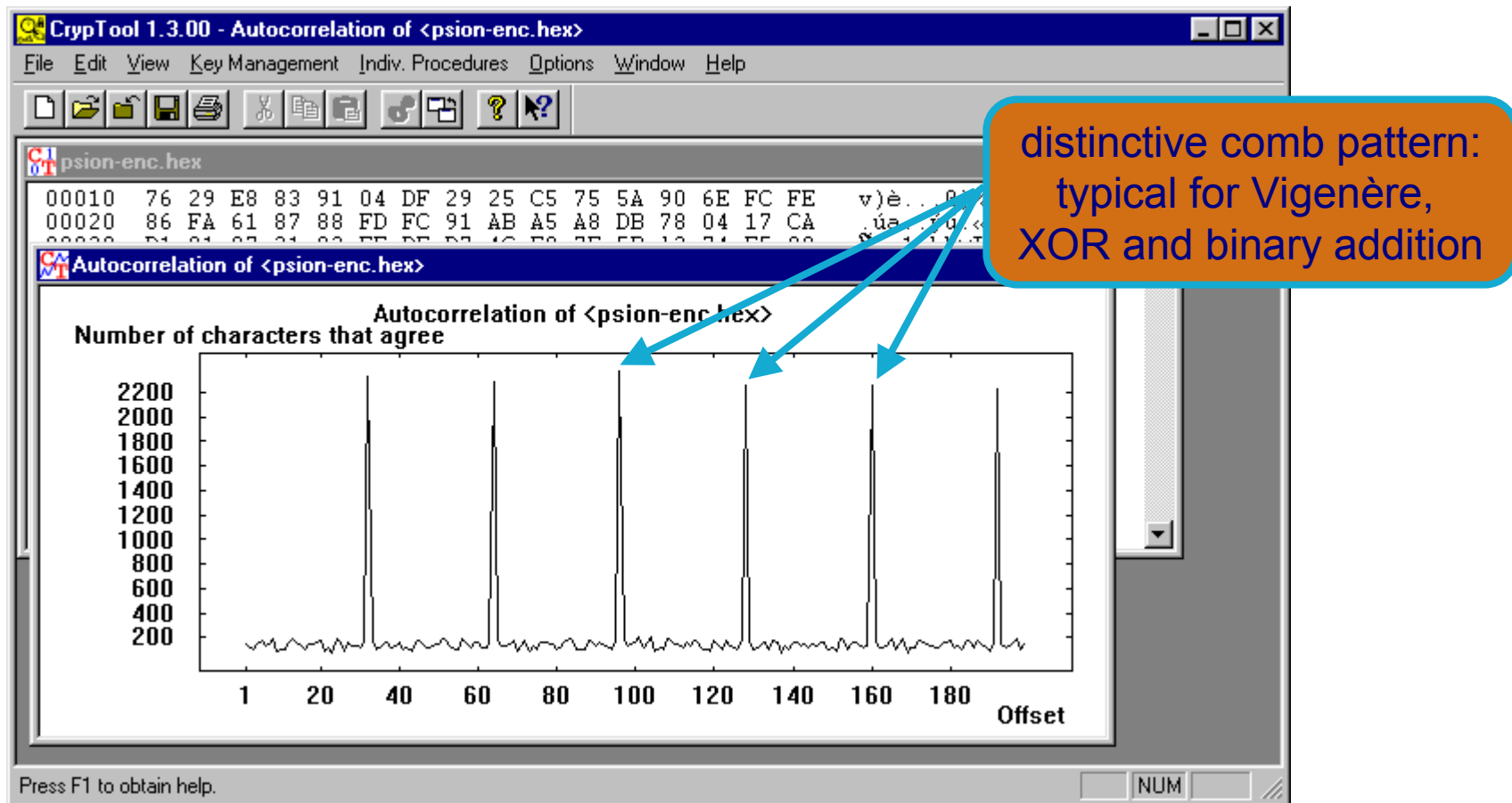
Examples of use

4. PSION PDA: determine entropy, compression test



Examples of use

4. PSION PDA: determine auto-correlation



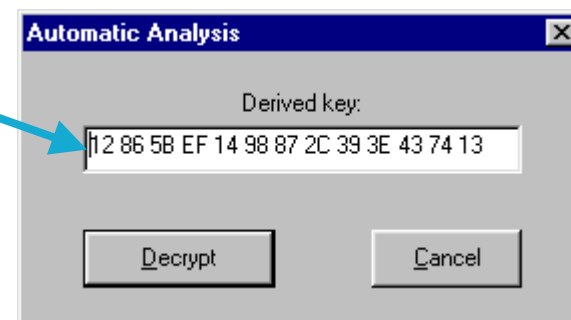
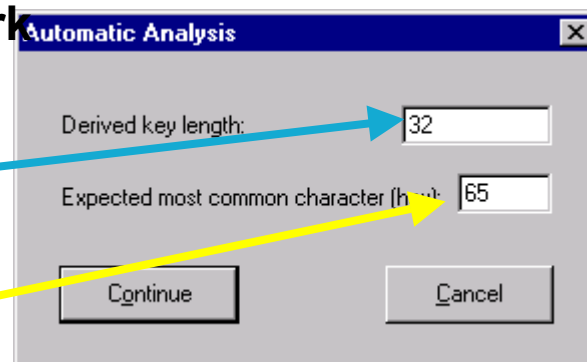
Examples of use

4. PSION PDA: automatic analysis

Automatic analysis using XOR: does not work

Automatic analysis using 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)
- Results: good, but not perfect



Examples of use

4. PSION PDA: results of automatic analysis

Results of automatic analysis with assumption “binary addition”:

- results good, but not perfect: 24 out of 32 key bytes correct.
- the key length was correctly determined.
- 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)

The screenshot shows a window titled "Automatic Addition Analysis of <psion-enc.hex>, key: <12 86 5B EF 14 98 87 2C 39 3E 43 74 13 ...". The window is divided into two main sections. The left section displays a hex dump of the encrypted data, with addresses from 00000 to 000F0 on the left and hex values in pairs on the right. The right section displays the decrypted text, which is a German sentence: "erg³setz (UStG).
...rstereAb, 'hn@
tt...teuergegenst
an@ und .el¹ng,
ber³ich..\$ 1..(1
) .er Um,at²,te²
er ²nterliegen d
ieefolge³de³eUm,
ätz³..1. die Lie
fe..ungeneun@eso³
stiren Leistunge
n,edie e@n .³te.
neh²er im Inland
g³gen E³tg³tt @
m R|hmen seines
Un¹erneh²en.eau.

Examples of use

4. PSION PDA: determining the remaining key bytes

Copy key to clipboard during automatic analysis

In automatic analysis hexdump,

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

In encrypted initial file hexdump,

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

Correct key from the clipboard

- 12865B341498872C393E43741396A45670235E111E907AB7C0841...

Decrypt encrypted initial document using binary addition

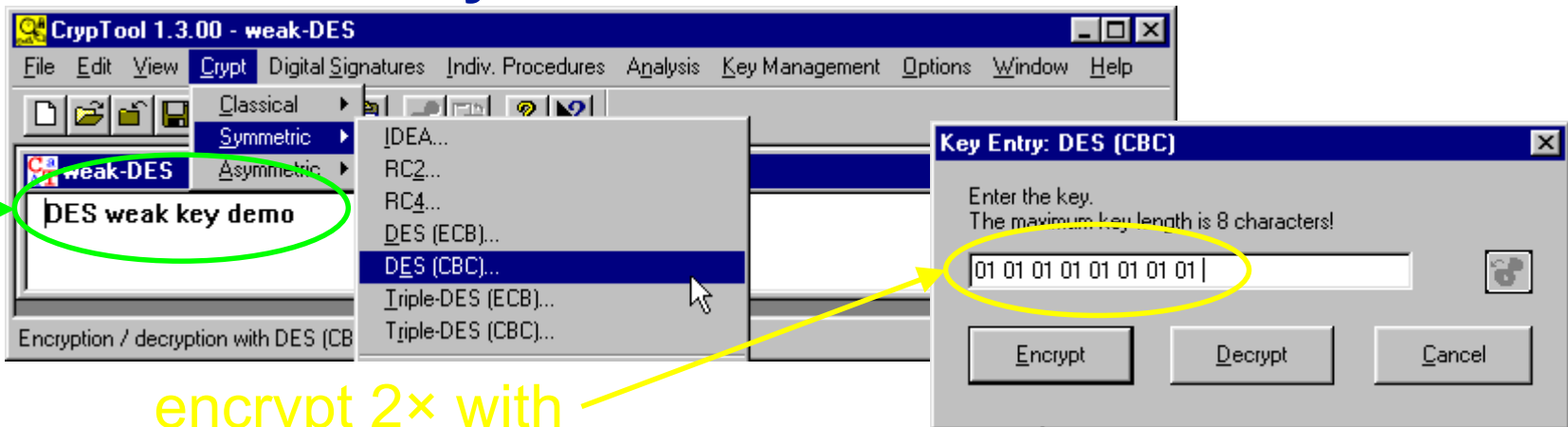
- bytes at position 3, 3+32, 3+2*32, ... are now correct

```
ADD decryption of <...>, key <...>
00000  65 72 67 65 73 65 74 7A 20 28 55 53 74 47 29 06  ergesetz (UStG).
00010  06 06 8A 72 73 74 65 72 65 41 62 B8 A8 68 6E AE  ...rstereAb,`hn@
00020  74 74 06 53 74 65 75 65 72 67 65 67 65 6E 73 74  tt.Steuerggegenst
```

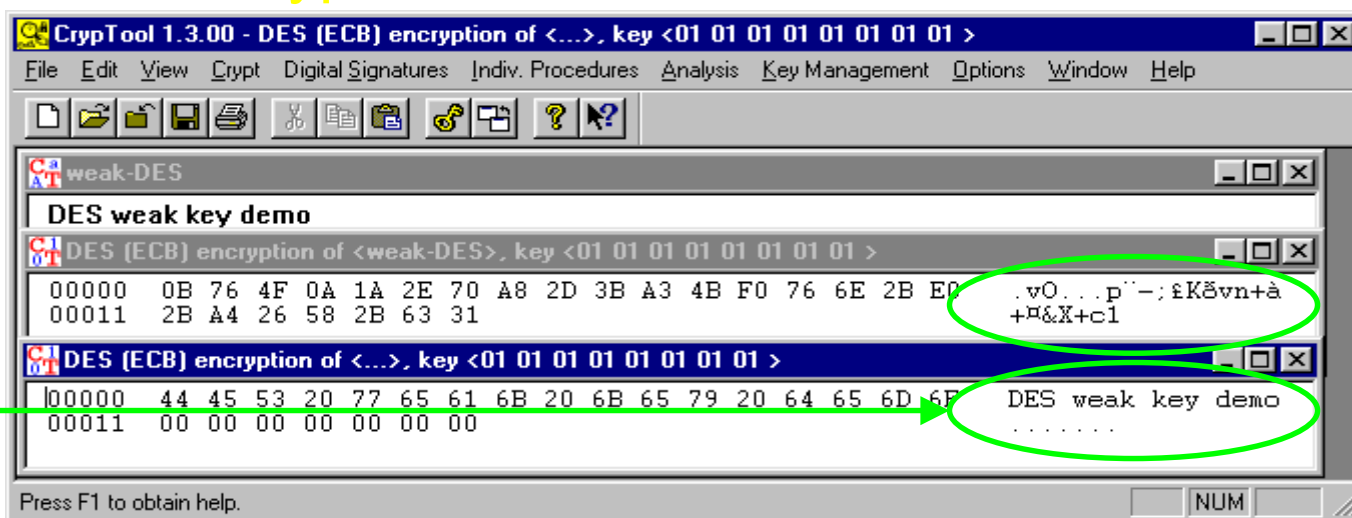


Examples of use

5. Weak DES keys



encrypt 2× with



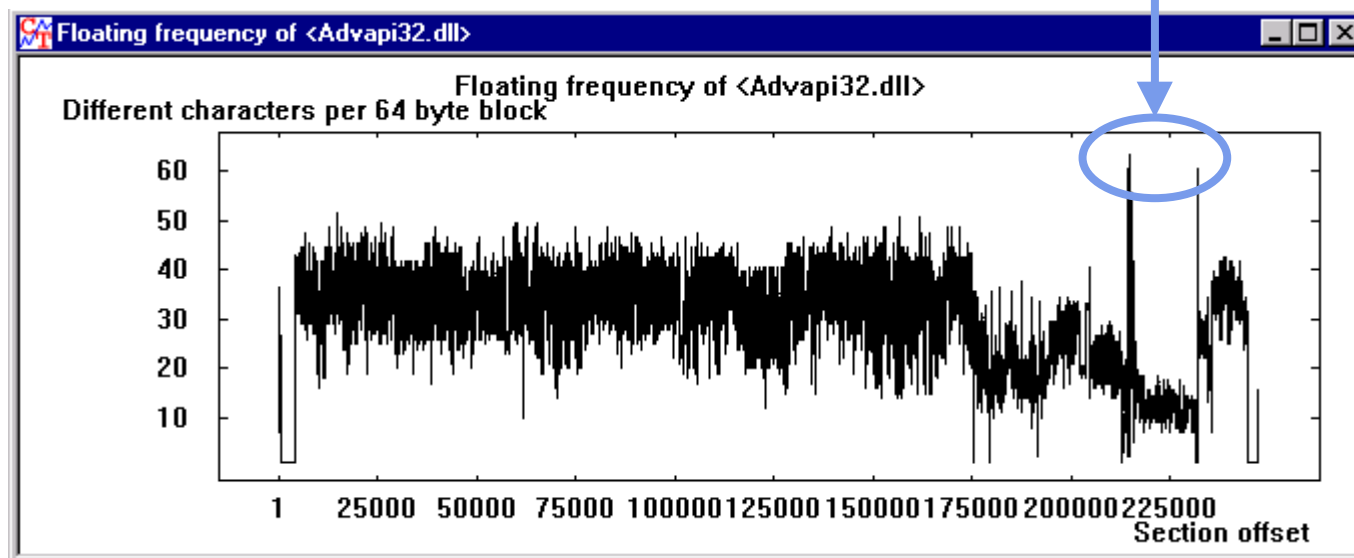
Examples of use

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 recognised as peaks in the “floating frequency”
- example: the “NSAKEY” in advapi32.dll



Contact address

Prof. Dr. Claudia Eckert

University Darmstadt

Faculty of Computer Science

IT Security

Wilhelminenstr. 7

64283 Darmstadt, Germany

claudia.eckert@sit.fraunhofer.de

Bernhard Esslinger

- University Siegen

Faculty of Economics

- Deutsche Bank AG

Head of Information Security

bernhard.esslinger@db.com

besslinger@web.de

Jörg Cornelius Schneider

Deutsche Bank AG

joerg-cornelius.schneider@db.com

js@joergschneider.com

www.cryptool.de

www.cryptool.org

www.cryptool.com