

Verschlüsselung, Kryptographie und Kryptoanalyse verstehen

Praktische Einführung
anhand von Beispielen mit
dem Lernprogramm CrypTool

*Prof. Bernhard Esslinger
und CrypTool-Team*

April 2016



Bedeutung der Kryptographie

Einsatzbeispiele für Kryptographie

- Telefonkarten, Handys, Fernbedienungen
- Geldautomaten, Geldverkehr zwischen Banken
- Electronic cash, Online-Banking, Sichere E-Mail
- Satellitenfernsehen, PayTV
- Wegfahrsperrung im Auto
- Digital Rights Management (DRM)



Kryptographie wird schon lange nicht mehr nur von Agenten, Diplomaten und Militärs benutzt. Kryptographie ist eine moderne, mathematisch geprägte Wissenschaft.

- Der Durchbruch für den breiten Einsatz kam mit dem Internet.
- Für Firmen und Staaten ist es wichtig, dass sowohl die Anwendungen sicher sind, als auch, dass...
- **...die Nutzer (Kunden, Mitarbeiter) ein Mindestverständnis und Bewusstsein (Awareness) für IT-Sicherheit besitzen!**



- **Vertraulichkeit (*Confidentiality*)**

Lesen des eigentlichen Inhalts für Unbefugte „praktisch“ unmöglich machen

- **Authentifizierung (*Authentication*)**

Identitätsbeweis des Senders gegenüber dem Empfänger einer Nachricht

- **Integrität (*Integrity*)**

Eigenschaft, dass die Nachricht nicht verändert wurde

- **Verbindlichkeit (*Non-Repudiation*)**

Der Empfänger kann den Nachweis erbringen, dass der Sender die Nachricht mit identischem Inhalt abgeschickt hat (Leugnen zwecklos)

Was bietet CrypTool

4

Ausgewählte Beispiele mit CrypTool 1, CrypTool 2 und JCrypTool

13

Projekt / Ausblick / Kontakt

45



Was bietet CrypTool

1. Was ist CrypTool
 2. Funktionsumfang
 3. Sprachstruktur analysieren
 4. Konzepte zur Benutzerfreundlichkeit
 5. Herausforderungen für den Programmierer
-

Was ist CrypTool



Was kann CrypTool?

- Kostenloses, Open-Source Lern-Programm mit graphischer Oberfläche.
Es gibt 3 Offline-Varianten (diese werden hier genauer vorgestellt) und eine Online-Variante.
- Kryptographische Verfahren anwenden *und* analysieren
- „Spielerischer“ Einstieg in moderne und klassische Kryptographie
- Sehr umfangreiche Online-Hilfe; ohne tieferes Kryptographie-Wissen verständlich
- Enthält fast alle State-of-the-art-Funktionen der Kryptographie
- Entwickelt von derzeit rund 70, meist ehrenamtlich arbeitenden Personen weltweit

Warum CrypTool?

- Ursprung im End-User Awareness-Programm einer Großbank
- Entwickelt in Kooperation mit Hochschulen → mediendidaktischer Anspruch
- Verbesserung der Lehre an Hochschulen und der betrieblichen Ausbildung

Zielgruppe

- Kernzielgruppe: Studierende der Informatik, Wirtschaftsinformatik, Mathematik
- Aber auch: Computernutzer und Anwendungsentwickler, Mitarbeiter, Schüler, Geocacher
- Voraussetzung: Computer-Kenntnisse
- Wünschenswert: Interesse an Mathematik und Programmierung

Funktionsumfang (Auszug)

(1/3) [vergleiche <https://www.cryptool.org/de/ctp-dokumentation/ctp-funktionsumfang>]

Kryptographie

Verschlüsselungsklassiker

- Caesar (incl. ROT-13)
- Monoalphabetische Substitution (incl. Atbash)
- Vigenère
- Hill
- Homophone Substitution
- Playfair
- ADFGVX
- Byteweise Addition
- XOR / Vernam / OTP
- Solitaire
- Permutation / Transposition
(Gartenzaun, Skytale, Doppelwürfel, ...)

Optionen zum besseren Nachvollziehen von Literaturbeispielen

- Alphabet wählbar
- Behandlung von Sonderzeichen einstellbar

Kryptoanalyse

Angriffe auf klassische Verfahren

- Ciphertext-Only
 - Caesar
 - Vigenère (nach Friedman + Schrödel)
 - Byteweise Addition
 - XOR
 - Substitution
 - Playfair
- Known-Plaintext
 - Hill
 - Einstufige Permutation/Transposition
- Manuell (unterstützt)
 - Monoalphabetische Substitution
 - Playfair, ADFGVX, Solitaire

Unterstützende Analyseverfahren

- Entropie, gleitende Häufigkeit
- Histogramm, n-Gramm-Analyse
- Autokorrelation
- Perioden
- Zufallszahlenanalyse



Kryptographie

Moderne symmetrische Verschlüsselung

- IDEA, RC2, RC4, RC6, DES, 3DES, DESX
- AES-Kandidaten der letzten Auswahlrunde (Serpent, Twofish, ...)
- AES (= Rijndael)
- DESL, DESXL

Asymmetrische Verschlüsselung

- RSA mit X.509-Zertifikaten
- RSA-Demonstration
 - zum Nachvollziehen von Literaturbeispielen
 - Alphabet und Blocklänge einstellbar

Hybridverschlüsselung (RSA + AES)

- Visualisiert als interaktives Datenflussdiagramm

Kryptoanalyse

Brute-Force-Angriff auf symmetrische Algorithmen

- Für alle Algorithmen
- Annahmen:
 - Entropie des Klartextes klein,
 - Teilweise Kenntnis der Schlüssels, oder
 - Kenntnis des Klartextalphabets

Angriff auf RSA-Verschlüsselung

- Faktorisierung des RSA-Moduls*
- Gitterreduktions-basierte Angriffe

Angriff auf Hybridverschlüsselung

- Angriff auf RSA oder
- Angriff auf AES (Seitenkanalangriff)

* Vergleiche die Untersuchung von Schulz und Witten: „Zeit-Experimente zur Faktorisierung – Ein Beitrag zur Didaktik der Kryptologie“, in: LOG IN, Heft 166/167, 2010, S. 107-114, http://bscw.schule.de/pub/bscw.cgi/d864899/Schulz_Witten_Zeit-Experimente.pdf



Kryptographie

Digitale Signatur

- RSA mit X.509-Zertifikaten
 - Signatur zusätzlich visualisiert als interaktives Flussdiagramm
- DSA mit X.509-Zertifikaten
- Elliptic Curve DSA, Nyberg-Rueppel

Hashfunktionen

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

Zufallsgeneratoren

- Secude
- $x^2 \bmod n$
- Linearer Kongruenzgenerator (LCG)
- Inverser Kongruenzgenerator (ICG)

Kryptoanalyse

Angriff auf RSA-Signatur

- Faktorisierung des RSA-Moduls
- Praktikabel bis ca. 250 Bit bzw. 75 Dezimalstellen (auf Einzelplatz-PC)

Angriff auf Hashfunktion / digitale Signatur

- Generieren von Hash-Kollisionen für ASCII-Texte (Geburtstagsparadoxon). (Bei 40 Bit finden moderne PCs eine Kollision für beliebige Hashfunktionen in weniger als 5 Minuten.)

Analyse von Zufallsdaten

- FIPS-PUB-140-1 Test-Batterie
- Periode, Vitányi, Entropie
- Gleitende Häufigkeit, Histogramm
- n-Gramm-Analyse, Autokorrelation
- ZIP-Kompressionstest

Sprachstruktur analysieren

Hier ein Beispiel mit CrypTool 1 (CT1)

Anzahl Einzelzeichen, n-Gramme, Entropie

- z.B. im Menü: „Analyse“ \ „Werkzeuge zur Analyse“ \ ...

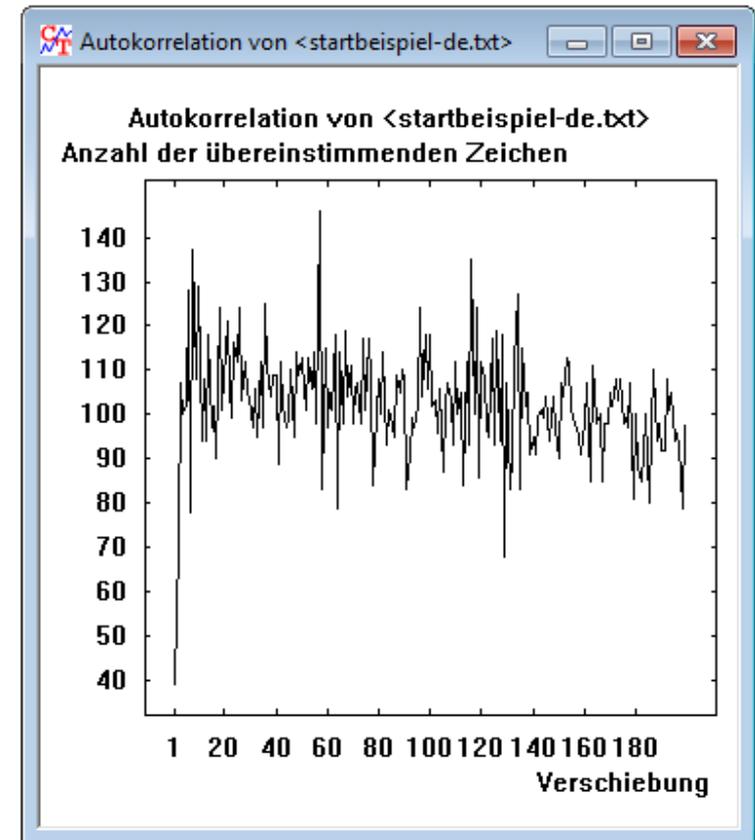
N-Gramm-Liste von startbeispiel-de.txt

Auswahl

Histogramm
 Digramm
 Trigramm
 4 -Gramm

Anzeige der
häufigsten N-Gramme
(erlaubte Werte: 1-5000).

Nr.	Zeichen...	Häufigkeit in %	Häufigkeit
1	E	18.2162	241
2	I	10.6576	141
3	N	10.4308	138
4	S	7.3318	97
5	T	6.6515	88
6	R	6.4248	85
7	A	4.6107	61
8	L	4.5351	60
9	D	4.4596	59
10	H	3.7037	49
11	O	3.3258	44
12	C	3.0990	41
13	M	2.5699	34
14	F	2.3432	31
15	U	2.1164	28
16	B	1.7385	23
17	K	1.5117	20
18	P	1.2850	17
19	G	1.2094	16
20	Y	0.9826	13
21	V	0.8314	11
22	W	0.7559	10
23	Z	0.7559	10
24	J	0.2268	3
25	X	0.2268	3



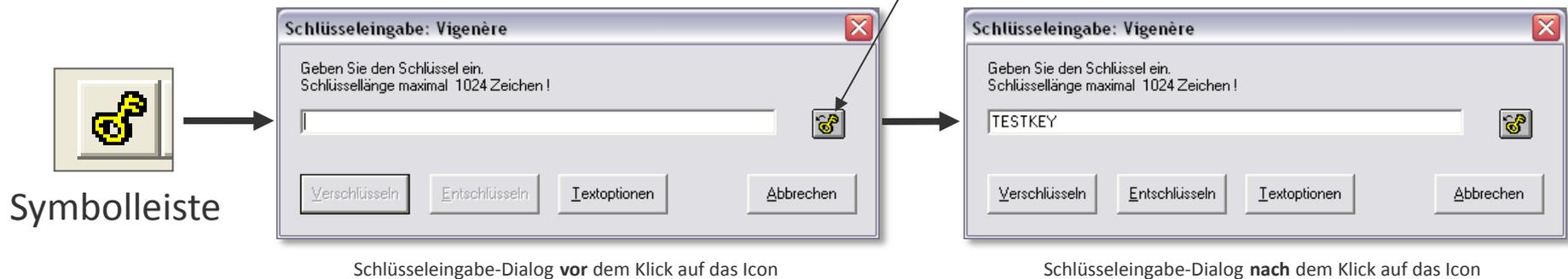
* Hier waren die Einstellungen in den Textoptionen so, dass nur Buchstaben gewählt wurden und die Groß- und Kleinbuchstaben nicht unterschieden wurden.

Kontext-sensitive Hilfe (F1)

- F1 bei einem gewählten Menüeintrag zeigt Informationen zum Verfahren
- F1 in einer Dialogbox erläutert die Bedienung des Dialogs
- Diese Hilfen und die Inhalte des übergeordneten Menüs sind in der Online-Hilfe immer gegenseitig verlinkt.

Einfügen von Schlüsseln in die Schlüsseleingabe-Maske

- Mit Strg-V (Paste) kann man immer einfügen, was in der Zwischenablage (Clipboard) enthalten ist
- Die zur Verschlüsselung benutzten Schlüssel können aus den Geheimtext-Fenstern per Icon in der Symbolleiste „entnommen“ und durch ein komplementäres Icon in der Schlüsseleingabemaske in das Schlüsselfeld eingefügt werden. Dazu wird ein CrypTool-interner Schlüssel-Speicher benutzt, der pro Verfahren zur Verfügung steht. Dies ist insbesondere bei „strukturierten“ Schlüsseln wie der homophonen Verschlüsselung hilfreich.



Herausforderungen für unsere Programmierer



Verschiedene Funktionen parallel laufen lassen

- Beispiel: Bei der Faktorisierung laufen die verschiedenen Algorithmen in Threads

Hohe Performance

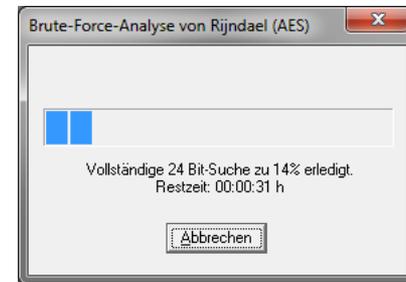
- Beim Finden von Hashkollisionen (Anwendung des Geburtstagsparadoxons) oder bei der Brute-Force-Analyse

Speicherbeschränkung beachten

- Beim Floyd-Algorithmus (Mappings für das Finden von Hashkollisionen) oder beim Quadratischen Sieb

Zeitmessung und -abschätzung

- Ausgabe der Restzeit (z.B. bei der Brute-Force-Analyse)



Wiederverwendung / Integration

- Masken zur Primzahlgenerierung
- RSA-Kryptosystem (schaltet nach erfolgreicher Attacke von der Ansicht des Public-Key-Anwenders zur Ansicht des Private-Key-Besitzers). Ein Angreifer kennt zu Beginn auch nur den Public-Key.

Automatisierung der Konsistenz der Ressourcen (Funktionen, GUI, Hilfe)

- Inklusive verschiedener Sprachen

Was bietet CrypTool

4

Ausgewählte Beispiele mit CrypTool 1, CrypTool 2 und JCrypTool

13

Projekt / Ausblick / Kontakt

45

Ausgewählte Beispiele mit CrypTool

Inhalt des Kapitels

Skytale von Sparta	Seite 15
Caesar-Verschlüsselung	Seite 19
Enigma-Verschlüsselung	Seite 25
Visualisierungen von AES (Rijndael-Chiffre)	Seite 28
Passwort-Qualitätsmesser (PQM) und Passwort-Entropie	Seite 31
Authentifizierung in einer Client-Server-Umgebung	Seite 33
Visualisierung von sicherer E-Mail per S/MIME	Seite 34
Signatur-Demo	Seite 35
Brute-Force-Analyse	Seite 40
Ein One-Time-Pad bitte nur genau einmal benutzen!	Seite 42
ECB- und CBC-Block-Modi von DES und AES als Bild	Seite 43
Lernspiele: Hier der Zahlenhai	Seite 44

Skytale von Sparta

Die Idee

Generelle Einordnung und historischer Hintergrund

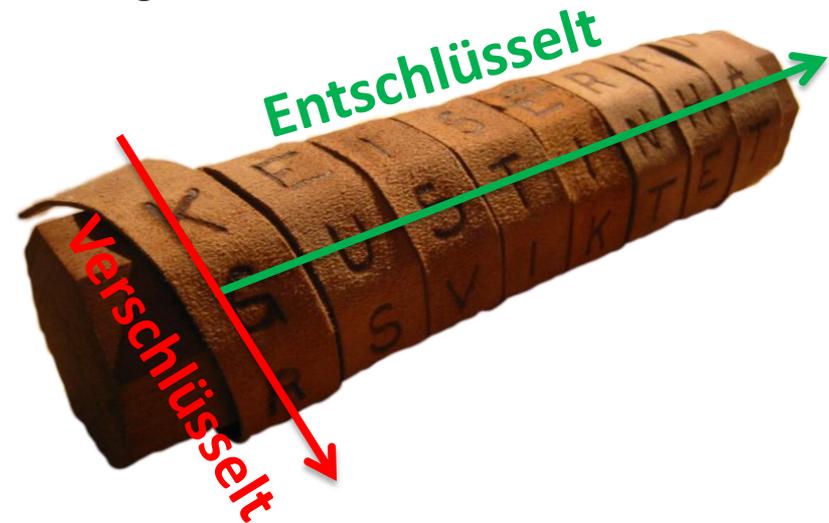
- Die Skytale von Sparta (etwa 500 v. Chr.) ist ein **Transpositions-Verfahren** – die Buchstaben des Klartextes bleiben erhalten, werden aber in ihrer Position vertauscht.
- Die Skytale ist das älteste militärisch genutzte Verschlüsselungsverfahren.
- Beschrieben vom griechischen Historiker/Schriftsteller Plutarch (45 - 125 n. Chr.)

Verschlüsselung

- Ein Band wird um einen eckigen Zylinderstab gewickelt und der Klartext wird längs der Zylinderseitenflächen auf das aufgewickelte Band geschrieben. Abgewickelt ergibt sich auf dem Band eine scheinbar willkürliche Zeichenfolge.

Entschlüsselung

- Wird das Band um einen Zylinder mit gleichem Durchmesser gewickelt, so lässt sich der Klartext wieder lesen.



Skytale von Sparta

Implementierung in CrypTool 1 (CT1)

Menü in CrypTool 1

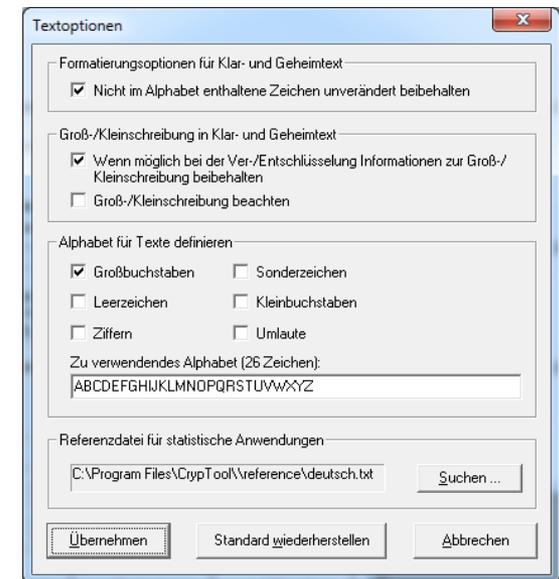
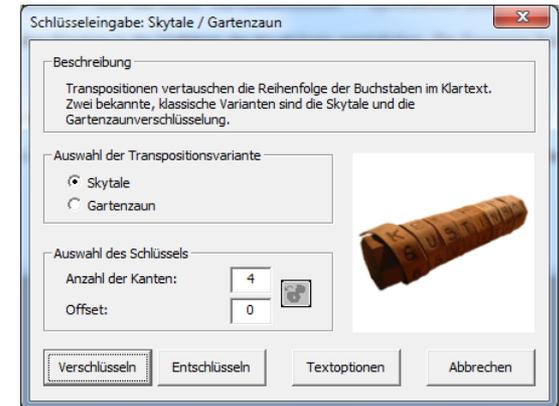
- „Ver-/Entschlüsseln“ \ „Symmetrisch (klassisch)“ \ „Skytale/Gartenzaun ...“

Auswahl des Schlüssels

- Anzahl der Kanten
- Offset

Textoptionen

- Erreichbar über das Menü: „Optionen“ \ „Textoptionen“ oder direkt bei der Schlüsseleingabe über den Button „Textoptionen“
- Einstellung der Art der Verarbeitung von Groß-/Kleinbuchstaben
- Definition des Alphabets zur Verarbeitung des Textes (z.B. welche Zeichen ver-/entschlüsselt werden)
- Auswahl der Datei mit statistischen Referenz-Kennzahlen (z.B. für die Häufigkeitsanalyse) zur Anpassung der statistischen Unterschiede verschiedener Sprachen



Skytale von Sparta

Implementierung in CrypTool 2 (CT2)

Vorlage in CrypTool 2

- Kryptographie → Klassisch → Skytale-Chiffre

The screenshot displays the CrypTool 2.1 (Nightly Build 6712.1) interface for the Skytale cipher. The main workspace shows a workflow diagram with three components: 'Texteingabe' (Text Input), 'Sk...' (Skytale cipher), and 'Textausgabe' (Text Output). The 'Texteingabe' component contains the text 'Dies ist eine geheime Botschaft.' and shows '32 Zeichen, 1 Zeile' and '0 %' progress. The 'Sk...' component is labeled 'Skytale' and shows '0 %' progress. The 'Textausgabe' component contains the ciphertext 'Dsseeei thti ti hmBsa.ei ngeeocf_' and shows '33 Zeichen, 1 Zeile' and '0 %' progress. A 'Schlüssel' (Key) component is also visible, showing '11' and '2 Dezimalziffern, 4 Bit' and '0 %' progress. The interface includes a menu bar with 'Start', 'Bearbeiten', and 'Kryptotutorien', a toolbar with 'Neu', 'Öffnen', 'Speichern', 'Drucken', 'Starten', 'Stoppen', and 'Protokoll', and a sidebar with 'Komponenten' and 'Suche'. The status bar at the bottom indicates 'Info: 19:00:00:055: Ausführungsmaschine erfolgreich gestoppt'.

Skytale von Sparta

Zusammenfassung

- Die Skytale ist ein Transpositions-Verfahren, die Reihenfolge der Zeichen im Klartext wird nur vertauscht, aber alle Zeichen bleiben unverändert erhalten (keine Substitution der Zeichen).
- Große Verbreitung im persischen Reich. Nach dem Untergang der klassischen Kulturen taucht das Verfahren erst wieder im Mittelalter auf.*
- Es werden zwei Zylinder (Holzstäbe) mit gleichem Durchmesser benötigt. Einmal zum Verschlüsseln und einmal zum Entschlüsseln der Nachricht.
- Mehrfaches Anwenden dieser Verschlüsselung kann auf eine einzelne Transposition (Zeichen des Klartextes werden umsortiert) zurückgeführt werden.

- Beispiel:

Klartext: STARTBE ISPIEL

Geheimtext: SITSAPR ITEBLE

Schlüssel: 2

Parameter: Alphabet soll nur Großbuchstaben enthalten, Nicht-Alphabetzeichen bleiben stehen.

* Quelle Wikipedia: <http://de.wikipedia.org/wiki/Skytale>

Caesar-Verschlüsselung

Die Idee

Generelle Einordnung und historischer Hintergrund

- Die Caesar-Verschlüsselung (Julius Caesar, 100 - 44 v.Chr.) ist eine einfache **Substitutions-Verschlüsselung**.

Verschlüsselung

- Zur Verschlüsselung wird jeder Buchstabe aus dem Klartextalphabet auf den entsprechenden Buchstaben des Geheimentextalphabets abgebildet. Mit dieser Abbildung werden die Zeichen des Alphabets lediglich um einen konstanten, definierten Wert verschoben. Dieser numerische Wert ist der Schlüssel.

GALLIA EST OMNIS DIVISA ...

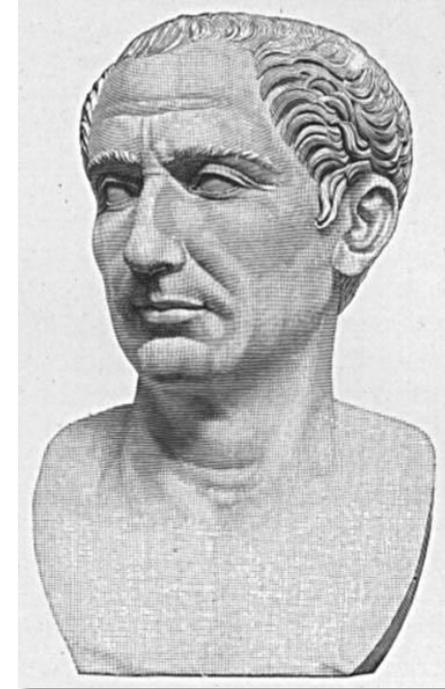
Klartextalphabet:

ABCDEFGHIJKLMN**O**PQRSTUVWXYZ

Geheimentextalphabet:

DEFGHI**J**KLMNOPQRSTUVWXYZ**A**C

JDOOLD HVW RPQLV GLYLV D ...



Entschlüsselung

- Zur Entschlüsselung wird die Abbildung invers zur Verschlüsselung ausgeführt: War der Schlüssel zum Beispiel eine Verschiebung des Alphabets um +4, kann man mit einer Verschiebung des Alphabets von -4 die Verschlüsselung wieder aufheben.

Caesar-Verschlüsselung

Implementierung in CrypTool 1 (CT1)

Menü in CrypTool 1

- **Animation**
„Einzelverfahren“ \ „Visualisierung von Algorithmen“ \ „Caesar“
- **Anwendung**
„Ver-/Entschlüsseln“ \ „Symmetrisch (Klassisch)“ \ „Caesar / Rot-13“

Auswahl des Schlüssels

- Den Wert des ersten Buchstabens als 0 oder 1 definieren.
- Schlüssel als Zahl oder als Buchstaben eingeben (ergibt die Verschiebung)

Textoptionen

- Standardmäßig werden die 26 Großbuchstaben als Alphabet definiert (und Kleinbuchstaben vor der Verschlüsselung in Großbuchstaben konvertiert). Im Textoptionen-Dialog (siehe [Seite 16](#) bei [Skytale von Sparta](#)) kann man aber auch andere Zeichen in das Alphabets aufnehmen oder Klein- und Großbuchstaben unterschiedlich behandeln.

Screenshot of the 'Schlüssel eingabe: Caesar / ROT-13' dialog box in CrypTool 1. The dialog is titled 'Schlüssel eingabe: Caesar / ROT-13' and contains the following sections:

- Beschreibung:** Hier können Sie für das Caesar-Verfahren den Schlüssel eingeben. Caesar ist eine monoalphabetische Substitution, bei der die Zeichen des Klartext-Alphabets durch Shiften um einen bestimmten Wert auf das Geheime-Text-Alphabet abgebildet werden. Dieser Verschiebewert ist der Schlüssel. Sie können den Schlüssel sowohl als Zahl als auch als einzelnes Alphabet-Zeichen eingeben. Rot-13 ist ein Spezialfall, bei dem der Schlüssel fest auf den Wert der halben Länge des Klartext-Alphabets gesetzt wird. Diese Variante ist nur wählbar, wenn die Länge des Alphabets eine gerade Zahl ist.
- Variante auswählen:** Radio buttons for 'Caesar' (selected) and 'Rot-13'.
- Interpretation des ersten Alphabetzeichens:** Radio buttons for 'Wert des ersten Alphabetzeichens = 0 (z.B. "A"=0)' (selected) and 'Wert des ersten Alphabetzeichens = 1 (z.B. "A"=1)'. A small icon of a key is visible next to these options.
- Schlüsseleingabe:** Radio buttons for 'Alphabetzeichen' (with input field 'F') and 'Zahlenwerte' (with input field '5'). A small icon of a key is visible next to these options.
- Informationen zur Verschlüsselung:** 'Verschiebung um 5'. Below this, it states 'Das Alphabet (26 Zeichen) wird bei der Verschlüsselung abgebildet' and shows the mapping:
von: ABCDEFGHIJKLMNOPQRSTUVWXYZ
auf: FGHIJKLMNOPQRSTUVWXYZABCDE
- Buttons:** 'Verschlüsseln', 'Entschlüsseln', 'Textoptionen' (highlighted with a dashed border), and 'Abbrechen'.

Caesar-Verschlüsselung

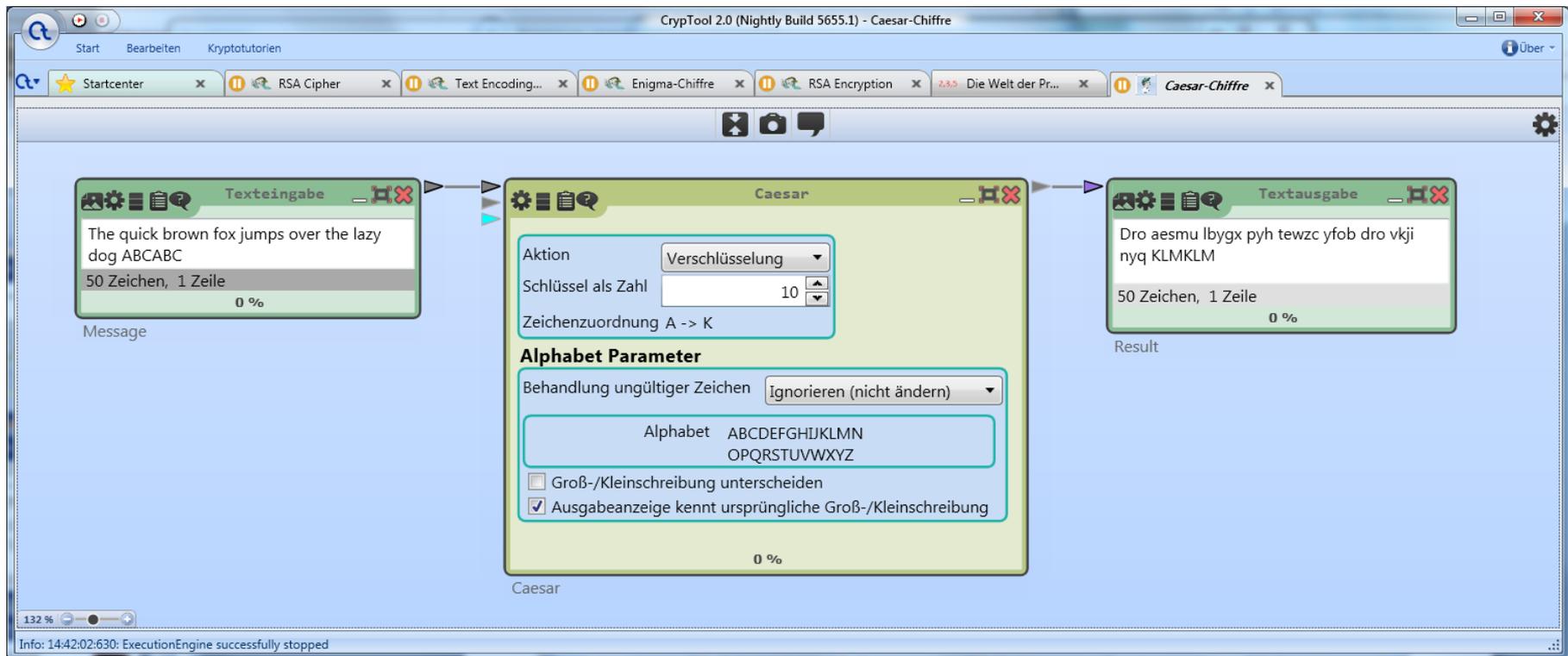
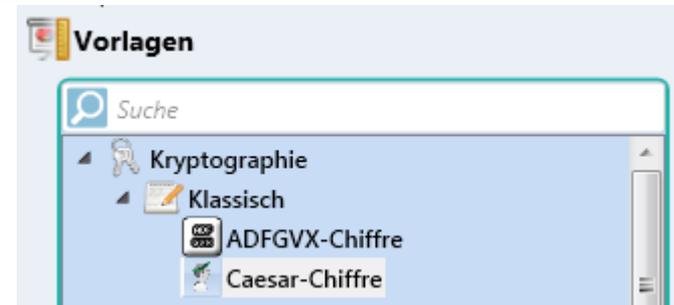
Implementierung in CrypTool 2 (CT2)

Vorlage in CrypTool 2

- Kryptographie → Klassisch → Caesar-Chiffre

Änderungen im Arbeitsbereich sofort sichtbar

- Nach jeder neuen Eingabe wird das Ergebnis sofort angezeigt.



Caesar-Verschlüsselung

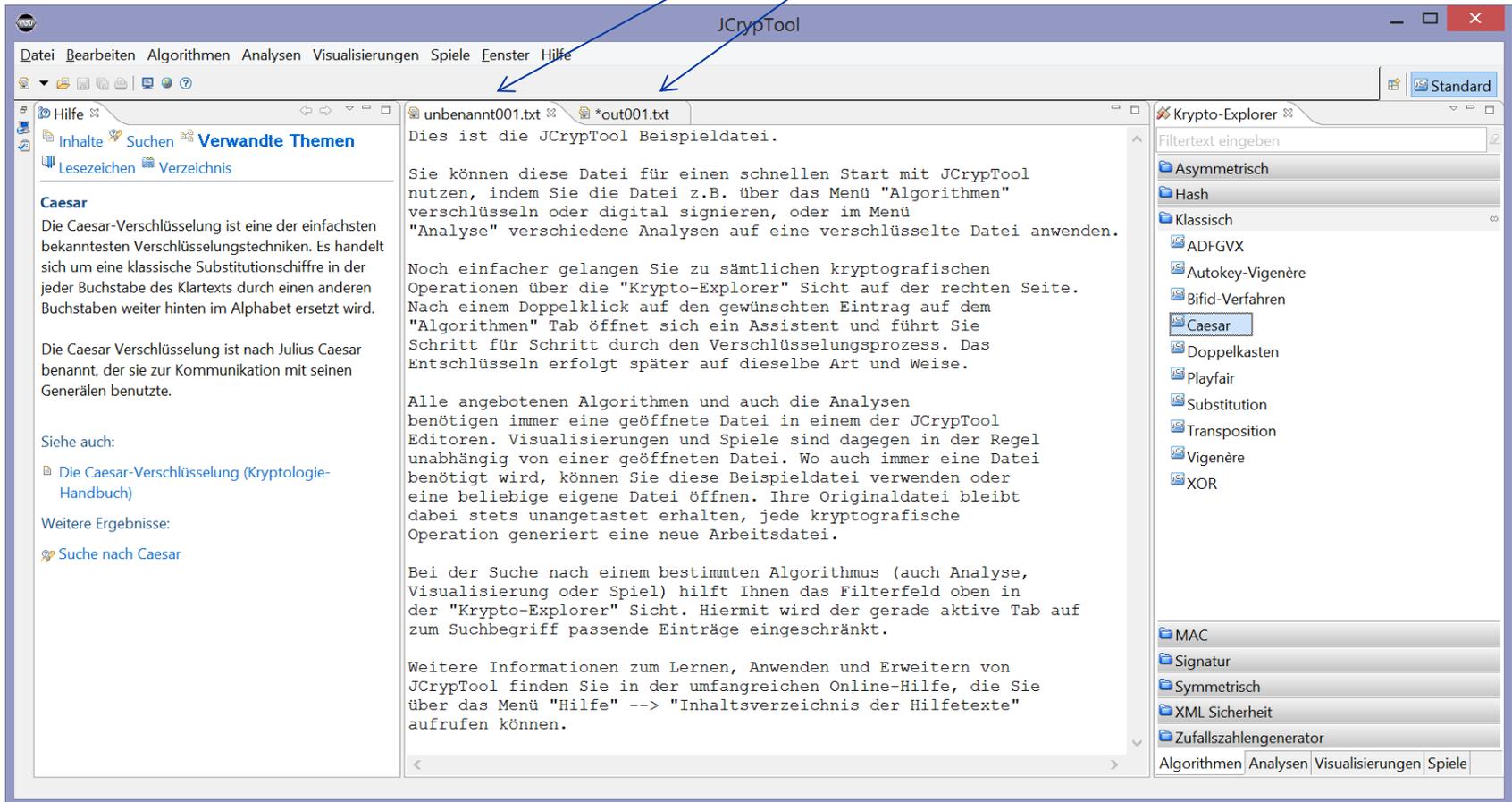
Implementierung in JCrypTool (1/2)

Menü in JCrypTool

- Anwendung
„Algorithmen“ \ „Klassisch“ \ „Caesar“

Reiter mit dem Klartext

Reiter mit dem verschlüsselten Ergebnis



Caesar-Verschlüsselung

Implementierung in JCrypTool (2/2)

Standarddialog in JCT zur Eingabe der Parameter und des Schlüssels

Caesar

Zum Verschlüsseln des gewählten Dokuments müssen Sie einen Schlüssel eingeben.
Außerdem können Sie ein Alphabet auswählen.

Operation

Verschlüsseln Entschlüsseln

Alphabet (current length: 52)

Klar-/Geheimtextalphabet: Alphabet auswählen: Lateinisches Alphabet (A-Z,a-z) Eigenes Alphabet erstellen Alphabet anzeigen

Interpretation des ersten Alphabetzeichens: Verschiebewert = 0 Verschiebewert = 1

Nichtalphabet-Zeichen im Eingabedokument vor der Verschlüsselung herausfiltern

Schlüssel

Geben Sie den Schlüssel ein. Als Alphabetbuchstabe: C oder als Zahl (Verschiebewert entlang des Alphabets): 2

Vorangehende Texttransformation

Dem Alphabet entsprechende Texttransformationen vor der Ver-/Entschlüsselung anwenden (nächste Seite)

JCT-Kommandozeile

Das Verfahren mit den in dieser Maske gewählten Parametern können Sie auch über die JCrypTool-Konsole ausführen (Fenster -> Sicht anzeigen -> Konsole). Hier ist das entsprechende Kommando:

```
caesar -E -ed -k C -a "Lateinisches Alphabet (A-Z,a-z)"
```

? < Zurück Weiter > Fertigstellen Abbrechen

Caesar-Verschlüsselung

Zusammenfassung

- Die Caesar-Verschlüsselung ist ein mono-alphabetisches Substitutions-Verfahren. Es wurde von dem römischen Schriftsteller Sueton (Gaius Suetonius Tranquillus, 70 - 122 n.Chr.) überliefert.
- Der Spezialfall ROT-13 ist ein involutorisches Verfahren.
- Diese Verschlüsselung kann mit einer Häufigkeitsanalyse angegriffen werden. Die Häufigkeitsanalyse beruht auf der ungleichen Verteilung von Buchstaben in natürlichen Sprachen, die von dieser Verschlüsselung nicht verborgen wird.
- Eine weitere Möglichkeit besteht darin, alle Schlüssel auszuprobieren. Bei einer Alphabetlänge von 26 sind zum Erraten des Schlüssels maximal 25 Versuche nötig.
- Die Sicherheit des Verfahrens basierte auf der Geheimhaltung des Verfahrens (die Schlüssel waren bei Caesar festgelegt auf den Wert 3), was dem Kerckhoffs'sche Prinzip* widerspricht.

* Das Kerckhoffs'sche Prinzip (auch als Kerckhoffs' Maxime bezeichnet) wurde 1883 von Auguste Kerckhoffs formuliert. Es ist ein Grundsatz der modernen Kryptographie, der besagt, dass die Sicherheit eines Verschlüsselungsverfahrens auf der Geheimhaltung des Schlüssels und nicht des Verfahrens beruhen muss.

Enigma-Verschlüsselung

Die Idee (Rotormaschinen / Elektromechanische Verschlüsselungsmaschinen)

Enigma-Verschlüsselung

- Erfinder der Enigma war Arthur Scherbius (1878-1929)
- Über 200.000 Maschinen waren im 2. Weltkrieg im Einsatz.
- Der rotierende Walzensatz bewirkt, dass jedes Zeichen des Textes mit einem neuen Alphabet verschlüsselt wird (poly-alphabetische Verschlüsselung).
- Das polnische Cipher Bureau brach die Vorkriegs-Enigma schon 1932.
- Darauf aufbauend wurde die Kriegs-Enigma gebrochen, mit massivem Einsatz (etwas 7000 Personen in GB), unter Nutzung der ersten Entschlüsselungsmaschinen sowie erbeuteter Original-Maschinen, abgefangener täglicher Statusmeldungen (z.B. Wetternachrichten), etc.



Konsequenzen der erfolgreichen Kryptoanalyse

„Allgemein wird die Kompromittierung des Enigma-Codes als einer der strategischen Vorteile angesehen, der maßgeblich zum Gewinn des Krieges durch die Alliierten geführt hat. Es gibt Historiker, die vermuten, dass der Bruch der Enigma den Krieg um etliche Monate, vielleicht sogar um ein volles Jahr, verkürzt hat.“*

* vgl. auch Wikipedia: [https://de.wikipedia.org/wiki/Enigma_\(Maschine\)](https://de.wikipedia.org/wiki/Enigma_(Maschine))

Enigma-Verschlüsselung

Implementierung in CrypTool 2 – Darstellung ohne Präsentation (Visualisierung)

Vorlage in CrypTool 2

- Kryptographie → Klassisch → Enigma-Chiffre

The screenshot displays the CrypTool 2.0 interface for the Enigma cipher workflow. The main workspace shows a flow from 'Texteingabe' (Text Input) to 'Textausgabe' (Text Output). The 'Texteingabe' window contains the plaintext: `DASOB ERKOM MANDO DERWE HRMAQ TGIBT BEKAN NTXAA CHENX AACHE NXIST GERET TETXD URQGE BUEND ELTEN EINSA TZDER HILFS KRAEF TEKON NTEDE EBEDR OHUNG ABGEW ENDET UNDDI ERETT UNGDE RSTAD TGEGE NXEIN SXAQT XNULL XNULL XUHS IQERG ESTEL LTWER DENX`. The 'Textausgabe' window shows the resulting ciphertext: `LIPQHSVDWCLYXZQFXHIUVWVDJOBINZXRCEOTVNCIONTFQNSXWISXKHJ DAGDJV AKUKVMJAHSZQQJHZOIAVZOWMSCASRDNXKSRFHXCXCMPIGXUJCKISYSHETX VVOVDQLZYTJNXNUWKZRXUJFXMBDIBRVMJKRHTCUJPTFEIYNYNBEAQJCLMUOD FWMARQCFOBWN`. A third window, 'Enigma', displays the Enigma logo and has 'Präsentation abgeschaltet' (Presentation disabled) checked. The 'Parameter' panel on the right is configured for 'Enigma I / M3' with key 'RTZ'. The rotor settings are: Rotor 1 (III, seit 1930), Rotor 2 (IV, seit 1938, M3 "Heer"), Rotor 3 (I, seit 1930), and Umkehrwalze (UKW B, 2. November 1937). The ring settings are: Ring 1 (rechts) 8, Ring 2 26, Ring 3 16. The 'Steckerbrett' (steckerboard) is set to 'DRNATLHGZEMCOLL'. A small error message is visible: 'This example plaintext has been taken from: http://de.wikipedia.org/wiki/Enigma_(Maschine)'. The status bar at the bottom shows 'Info: 09:06:00:499: ExecutionEngine successfully stopped'.

Enigma-Verschlüsselung

Implementierung in CrypTool 2 – Darstellung mit Präsentation (Visualisierung): D → L

The screenshot displays the CrypTool 2.0 interface for the Enigma cipher. The main window shows the rotor configuration (Rotorlage) and the resulting index of coincidences (Ringstellung).

Rotorlage:

Ringstellung	Ringstellung	Ringstellung
16	26	8

Ringstellung:

Ringstellung	Ringstellung	Ringstellung
16	26	8

The rotor configuration is visualized as a grid of letters (A-Z) with connections between them. The rotor positions are I, R, T, A. The ring settings are 16, 26, 8.

The presentation shows the encryption of the message "DASOBERKOMMANDODERWEHRMAQTGIBTBKANTX AACHENXAACHENXISTGERETTETXDURQGEBUENDE LTENEINSATZDERHILFSKRAEFTEKONNTE DIEBED ROHUNGABGEWENDETUNDDIERETTUNGDERSTADTG". The resulting ciphertext is "L".

Info: 09:03:38:446: Enigma encryption done. The resulting index of coincidences is n. def.

Visualisierungen von AES (Rijndael-Chiffre) (1 / 3)

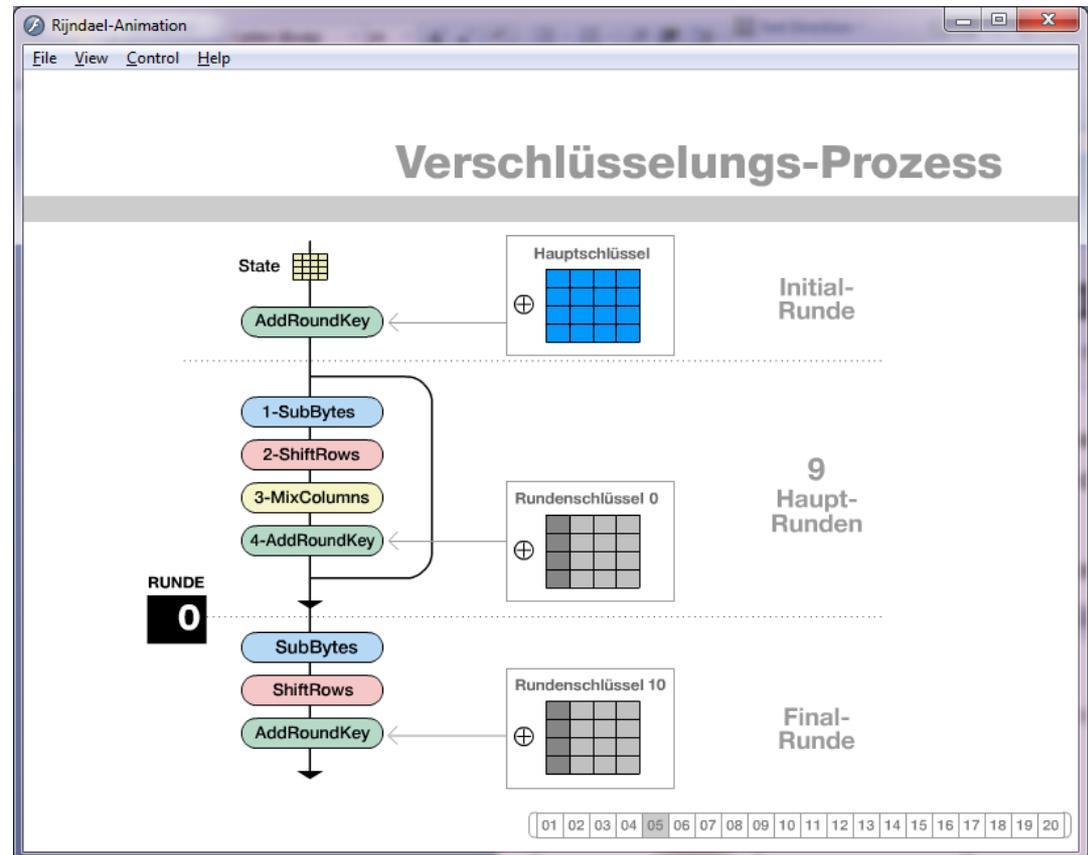
Rijndael-Animation

Rijndael-Animation

- Zeigt die Verschlüsselungsprozesse in jeder Runde (mit fixen Ausgangsdaten)

Menü in CrypTool 1

- „Einzelverfahren“ \
- „Visualisierung von Algorithmen“ \
- „AES“ \ „Rijndael-Animation“



Visualisierungen von AES (Rijndael-Chiffre) (2 / 3)

Rijndael-Inspektor

Rijndael-Inspektor

- Zum Ausprobieren mit eigenen Daten (zeigt den Inhalt der Datenmatrix in jeder Runde)

Menü in CryptTool 1

- „Einzelverfahren“ \
- „Visualisierung von Algorithmen“ \
- „AES“ \ „Rijndael-Inspector“

The screenshot shows the Rijndael-Inspector application window. At the top, there is a menu bar (File, View, Control, Help) and a title bar (Rijndael-Inspector). Below the menu bar, the application name "Rijndaelinspektor" is displayed in a red header, along with a "Testdaten laden" button and a close button. The main interface is divided into several sections:

- Control:** Radio buttons for "Verschlüsseln" (selected) and "Entschlüsseln".
- Input (Klartext):** A 4x4 matrix of hex values: 00 44 88 cc, 11 55 99 dd, 22 66 aa ee, 33 77 bb ff.
- Schlüssel:** A 4x4 matrix of hex values: 00 04 08 12, 01 05 09 13, 02 06 10 14, 03 07 11 15.
- Output:** A 4x4 matrix of hex values: b4 7f 33 a7, a4 30 e1 48, 6e a8 30 e3, 24 06 98 10.
- Process View:** A detailed view of the encryption process over three rounds. It shows the state of the data matrix at different stages: "Start der Runde", "Nach SubBytes", "Nach ShiftRows", "Nach MixColumns", and "Rundenschlüssel". The process is visualized as a sequence of operations: Input matrix + Rundenschlüssel = Resultant matrix. The Rundenschlüssel is shown as a 4x4 matrix of hex values: 00 04 08 12, 01 05 09 13, 02 06 10 14, 03 07 11 15.

Visualisierungen von AES (Rijndael-Chiffre) (3 / 3)

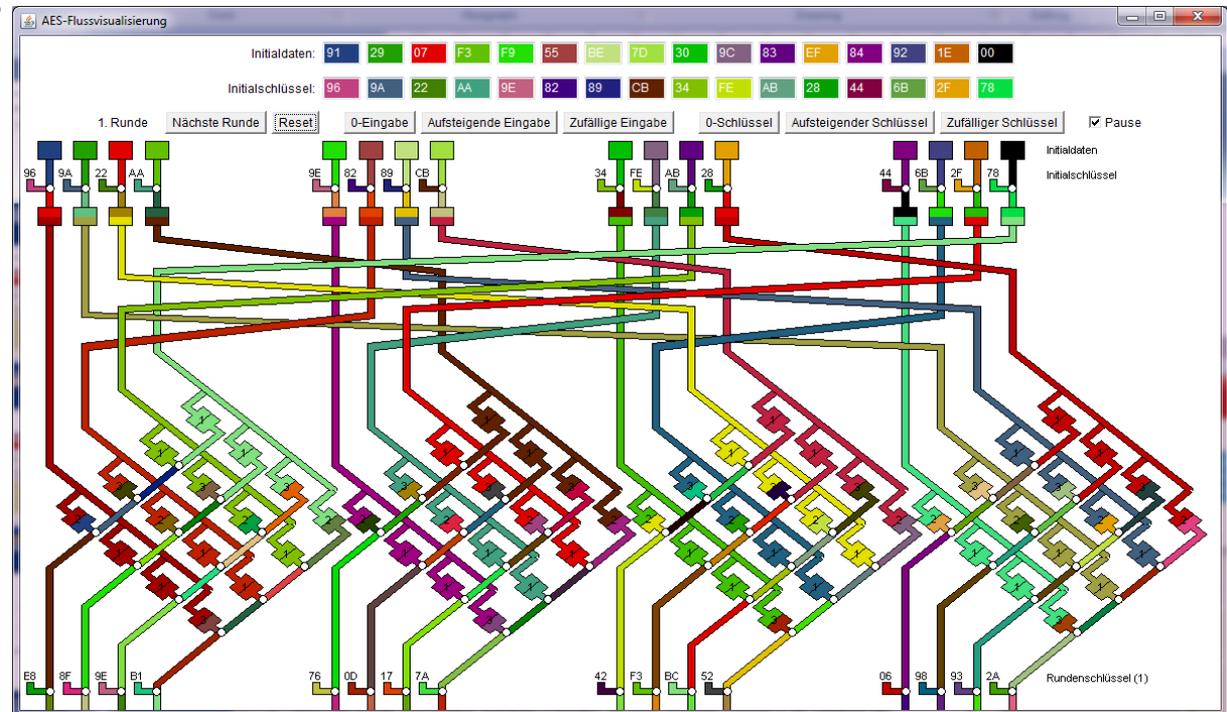
Rijndael-Flussvisualisierung

Rijndael-Flussvisualisierung

- Visualisierung der Datenveränderung je Runde durch Farbverläufe

Menü in CrypTool 1

- „Einzelverfahren“ \ „Visualisierung von Algorithmen“ \ „AES“ \ „Rijndael-Flussvisualisierung“

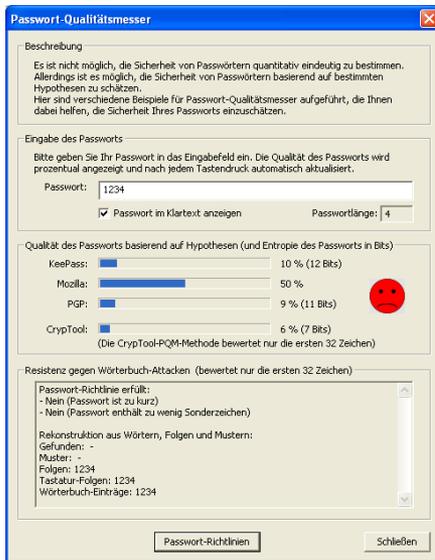


Passwort-Qualitätsmesser (PQM) und Passwort-Entropie

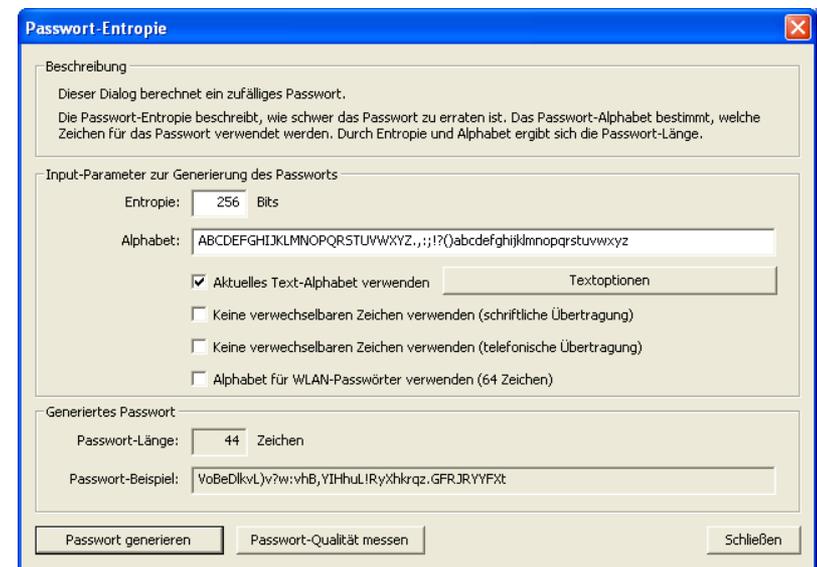
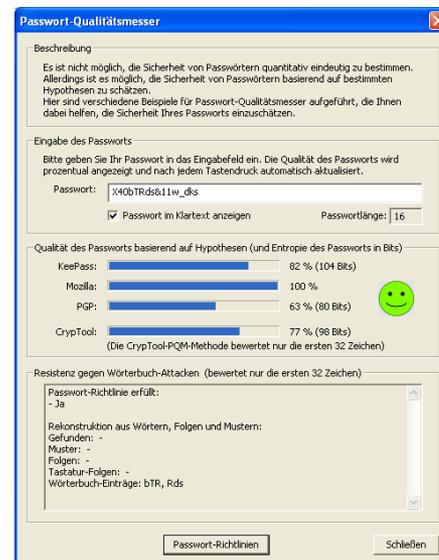
Funktionen

- Messung der Qualität von Passwörtern
- Vergleich mit PQMs aus anderen Applikationen: KeePass, Mozilla und PGP
- Experimentelle Bewertung durch CrypTool-Algorithmus
- Beispiel: Eingabe eines Passwortes im Klartext

Passwort: **1234**



Passwort: **X40bTRds&11w_dks**



Menü: „Einzelverfahren“ \ „Tools“ \ „Passwort-Qualitätsmesser“

Menü: „Einzelverfahren“ \ „Tools“ \ „Passwort-Entropie“

Passwort-Qualitätsmesser (PQM) und Passwort-Entropie

Erkenntnisse des Passwort-Qualitätsmessers

- Höhere Qualität des Passwortes durch die Verwendung von **verschiedenen Zeichenarten**: Groß-/Kleinschreibung, Zahlen und Sonderzeichen (**Passwortraum**)
- Passwortqualität hängt primär von der **Länge des Passwortes** ab!
- **Passwortentropie** als Maß der Zufälligkeit der Wahl von Zeichen aus dem Passwortraum (je zufälliger die Wahl, desto besser das Passwort)
- Passwörter sollten **nicht in einem Wörterbuch vorkommen**.

Qualität eines Passwortes aus der Angreifer-Perspektive

- Angriff auf ein Passwort (sofern beliebig viele Versuche zugelassen sind):
 1. Klassischer **Wörterbuchangriff**
 2. Wörterbuchangriff mit **weiteren Varianten** (z.B. Jahreszahlen anhängen: Sommer2007)
 3. **Brute-Force-Angriff** durch Test aller Kombinationen (ggf. mit Einschränkungen auf Zeichenarten)
- ⇒ Ein gutes Passwort sollte so gewählt werden, dass es den Angriffen #1 und #2 standhält, im Hinblick auf #3 zumindest 9 Zeichen lang ist, und Zahlen sowie Sonderzeichen beinhaltet.

Authentifizierung in einer Client-Server-Umgebung

- Interaktive Demo für verschiedene Authentifizierungs-Verfahren
- Definierte Möglichkeiten des Angreifers
- Sie können in die Rolle eines Angreifers schlüpfen.
- **Lerneffekt**
Nur die wechselseitige Authentifizierung ist sicher.

Menü in CT1: „Einzelverfahren“ \ „Protokolle“ \ „Authentisierungsverfahren im Netz“

Visualisierung von sicherer E-Mail per S/MIME

S/MIME-Visualisierung

- Control-Center: Signieren/Verschlüsseln von Nachrichten mit verschiedenen Parametern
- Animation: Von der Erstellung beim Sender bis zum Lesen beim Empfänger

The image shows two overlapping windows from a software application. The background window is titled "S/MIME Visualisierungs-Control-Center v1.0". It contains a text area for the message content, fields for recipient ("bob@web.de"), sender ("alice@wunderland.de"), and subject ("Nachricht wird signiert"). There are radio buttons for "Signieren" (selected) and "Verschlüsseln". A "Signieren starten" button is at the bottom. The foreground window is titled "S/MIME Animation" and shows a cartoon character with blonde hair and glasses standing next to a computer monitor displaying an Outlook interface. Below the character and monitor are navigation buttons: "Prolog", "E-Mail schreiben", "Kanonisieren", "Transfercodieren", "Weiterleiten", "Signieren", and "Transport". A text box at the bottom of the animation window reads: "Alice schreibt Bob die E-Mail, die Ihre digitale Signatur tragen soll. Die Signatur soll sicherstellen, dass die E-Mail tatsächlich von Alice stammt. Um zu sehen wie diese Signatur tatsächlich aussieht, werfen wir einen Blick hinter die Kulissen." Navigation buttons for the animation include "<< Szene zurück", "< Zurück", "Vor >", "Szene Vor >>", and "Schließen".

Menü in CT1: „Einzelfahren“ \ „Protokolle“ \ „Sichere E-Mail mit S/MIME...“



Problematik

- Elektronische Dokumente können à priori nicht auf den Urheber überprüft werden. Dazu braucht man ein Verifizierungsmerkmal des Autors, dies kann z.B. eine Unterschrift sein.
- Ein rein elektronisches Dokument kann abgewandelt werden, ohne dass dies bemerkt wird.

Um diese Problematik zu umgehen, kann der Autor ein elektronisches Dokument digital signieren.

Funktionsweise des Signierens

- Der Urheber generiert aus dem Dokument einen Hashwert.
- Der Hashwert wird (bei Verwendung von RSA) mit dem privaten Schlüssel des Autors verschlüsselt.
- Den verschlüsselten Hashwert und die benutzte Hashfunktion stellt der Autor mit dem Dokument zur Verfügung.
- Ein Interessent, der die Integrität des Dokuments überprüfen möchte, kann nun mit dem öffentlichen Schlüssel den Hashwert des Dokuments aus dem verschlüsselten Hashwert zurückgewinnen.
- Den Hashwert kann er selbst gegenprüfen, indem er die vom Autor benutzte Hashfunktion erneut auf das Dokument anwendet. Sind die beiden Werte identisch, kann er sich sicher sein, dass das Dokument nicht verfälscht wurde.



Im Menü unter:

„Visualisierungen“ \ „Signatur- Demo“

Der Algorithmus in der Anwendung

- In diesem Plugin gibt es die Möglichkeit, ein Dokument oder einen selbst eingegebenen Text zu signieren.
- Als Hashmethoden stehen die Funktionen MD5, SHA-1, SHA-256, SHA-384 und SHA-512 zur Verfügung.
- Anschließend kann, je nach gewählter Hashfunktion, als Signaturfunktion DSA, RSA, ECDSA oder RSA mit MFG1 benutzt werden.

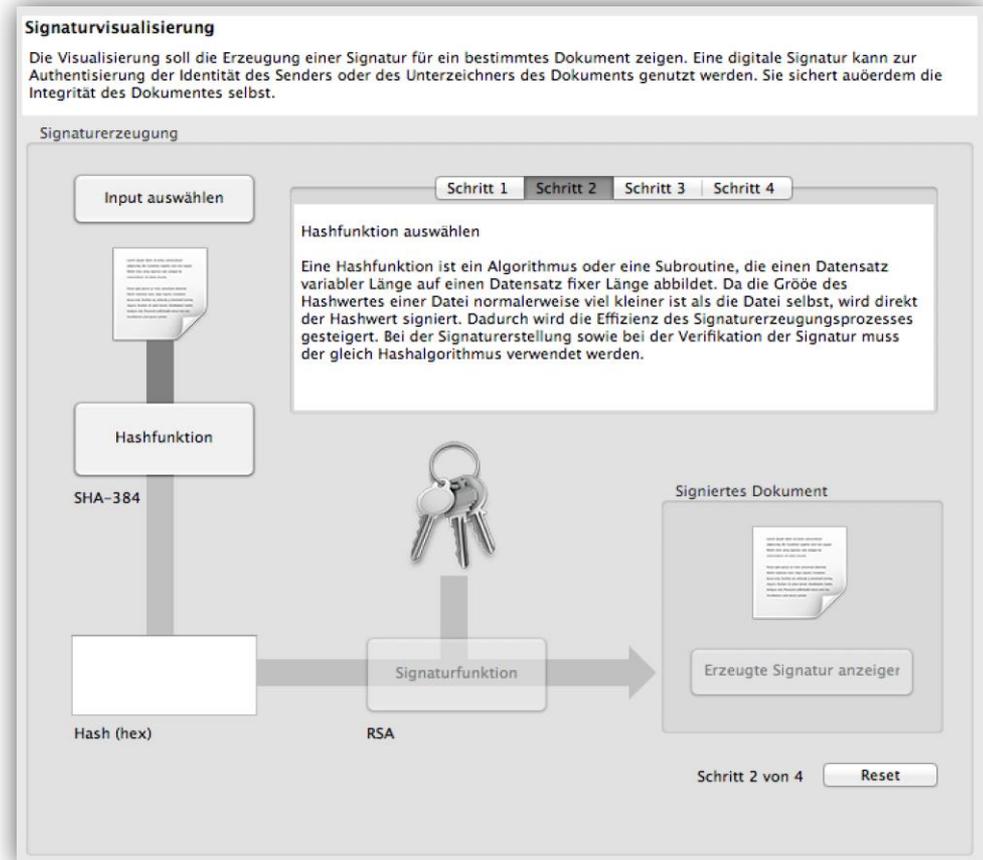
Signatur-Demo

Ein Anwendungsbeispiel 1/2

Ein Dokument zu signieren ist nicht aufwändig und geht in zwei Schritten.

Erster Schritt

- Wähle das zu signierende Dokument über „Input auswählen“.
- Es erscheint ein Dialog mit „Datei öffnen“ oder „Direkte Eingabe“, um einen beliebigen Text direkt einzugeben.
- Anschließend muss eine „Hashfunktion“ ausgewählt werden.
- Der Hashwert wird dann erzeugt und unten angezeigt.



Signatur-Demo

Ein Anwendungsbeispiel 2/2

Zweiter Schritt

- Über „Signaturfunktion“ kann ein Verschlüsselungsalgorithmus ausgewählt werden, mit dem der Hashwert verschlüsselt wird.
- Wir wählen als Signaturmethode „ECDSA“ aus. Darunter muss dann noch aus dem JCT-Keystore ein Schlüssel für den Signierer (hier „Alice Whitehead“) ausgewählt werden.
- Durch Klick auf „Fertigstellen“ wird die Signatur erzeugt und kann anschließend über „Erzeugte Signatur anzeigen“ betrachtet und abgespeichert werden.

Erzeugte Signatur anzeigen

Besitzer der Signatur: -

Verwendeter Schlüssel/Kurve: ANSI X9.62 prime256v1 (256 bits)

Signaturmethode: SHA384withECDSA

Signatur

Adresse	Hex	Ascii
00000	30 44 02 20 75 E0 76 4C 20 EB 02 A0 E6 2F	0D uävL ë æ/
0000E	94 5C 74 73 AC 8D 5F F9 5B 9C B2 91 1F 34	\ts~_ù]²4
0001C	00 6E 62 D6 D7 69 89 E9 02 20 47 19 8E 2C	nbÖxi¹é G,
0002A	D0 12 32 B2 C4 CA EA 67 94 95 F1 96 39 4B	D2²ÄËëgñ9K
00038	DE F9 88 83 16 C4 25 57 C9 0A EF FC D3 8F	püA%WÉ

Länge der Signatur: 70 Bits

Darstellungsmöglichkeiten der Signatur

Hex-Dump (Hex und Ascii) Oktal Dezimal Hex

Signierte Nachricht

Adresse	Hex	Ascii
00000	55 6E 64 20 77 65 6E 6E 20 73 69 65 20 6E	Und wenn sie n
0000E	69 63 68 74 20 67 65 73 74 6F 72 62 65 6E	icht gestorben
0001C	20 73 69 6E 64 2C 20 73 6F 20 6C 65 62 65	sind, so lebe
0002A	6E 20 73 69 65 20 6E 6F 63 68 20 68 65 75	n sie noch heu
00038	74 65 2E 20 0A 55 6E 64 20 77 65 6E 6E 20	te.
00046	69 68 72 20 50 72 6F 66 69 6C 20 76 65 72	ihr Profil ver

Länge der signierte Nachricht: 143 Bits

Um das signierte Dokument und die erzeugte Signatur anzuzeigen, klicken Sie auf "Hex Editor öffnen". Dort können Sie die Signatur speichern, falls Sie diese verifizieren

Hex Editor öffnen Schließen



Fazit

- Die Integrität von elektronischen Dokumenten kann mit Hilfe einer Signatur überprüft werden.
- Krypto-Algorithmen helfen auch hier, um den Autor zu verifizieren.
- Wird ein Dokument verfälscht, so ändert sich damit auch dessen Hashwert.
- Da der Hashwert bei der Verifikation für jeden zugänglich sein muss, wird dieser verschlüsselt. Entschlüsseln kann man ihn nur mit dem „richtigen“ öffentlichen Schlüssel, also dem des angeblichen Signierers. Somit kann der Hashwert, obwohl er öffentlich verfügbar ist, nicht nachträglich geändert werden, ohne dass es bemerkt würde.

Brute-Force-Analyse

Implementierung in CrypTool 1

Brute-Force-Analyse

- Hier Brute-Force-Analyse unter der Annahme, dass ein Teil des Schlüssels bekannt ist.* Computer können allein entscheiden, ob die Entschlüsselung ein sinnvoller Klartext ist!

Beispiel: Analyse mit DES (ECB)

- Versuch, mittels Brute-Force den vollständigen Schlüssel zu finden, um damit dann den Geheimtext zu entschlüsseln (Annahme: Der Klartext ist ein Block aus 8 ASCII-Zeichen)

Schlüssel (Hex)

```
68ac78dd40bbefd*  
0123456789ab****  
98765432106****  
0000000000****  
0000000000****  
abacadaba*****  
dddddddddd*****
```

Geheimtext (Hex)

```
66b9354452d29eb5  
1f0dd05d8ed51583  
bcf9ebd1979ead6a  
8cf42d40e004a1d4  
0ed33fed7f46c585  
d6d8641bc4fb2478  
a2e66d852e175f5c
```

* Wenn bei einem modernen symmetrischen Verfahren wie AES der gesamte Schlüssel zufällig und unbekannt ist, kann man den Geheimtext mit heute bekannten Mitteln nicht knacken. Laut den Dokumenten des Whistleblowers Edward Snowden speichert die NSA aber den gesamten Kommunikationsverkehr über vergangene Jahre, so dass sie es auch entschlüsseln kann, wenn sie später mal in den Besitz des Schlüssels kommt. Vergleiche die Notwendigkeit von Perfect Forward Secrecy bei SSL/TLS.

Brute-Force-Analyse

Implementierung in CrypTool 1

1. Eingabe des verschlüsselten Textes

- Hex-codierte Nachrichten können in CT 1 über das Menü: „Ansicht“ \ „Als HexDump anzeigen“ eingegeben werden.

2. Verwendung der Brute-Force-Analyse

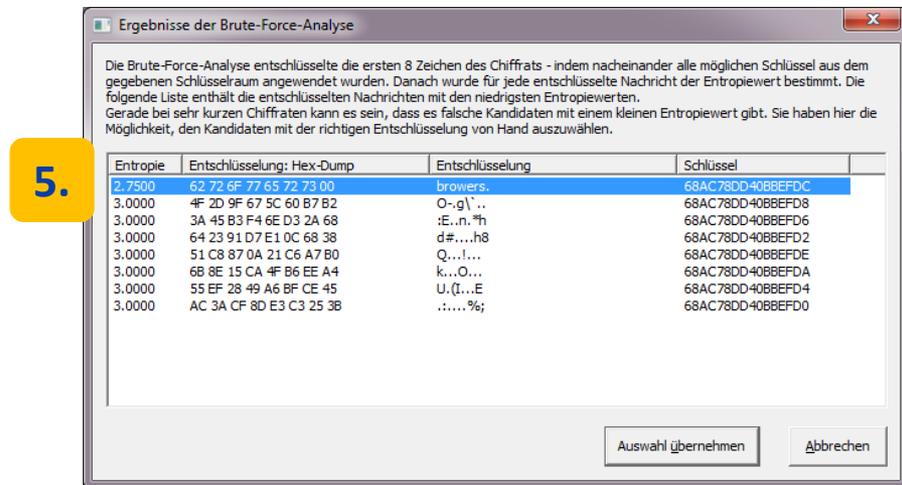
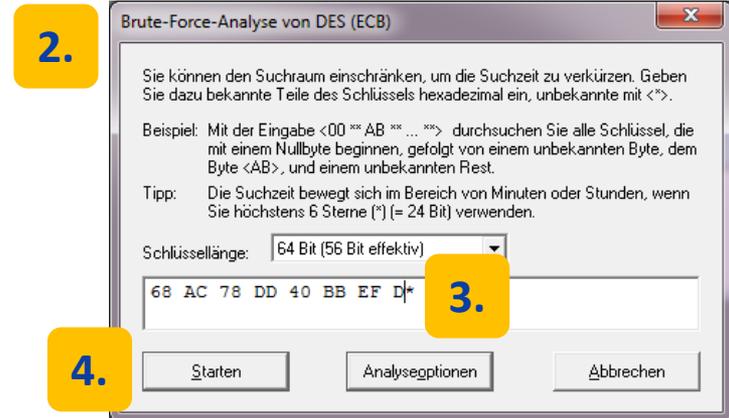
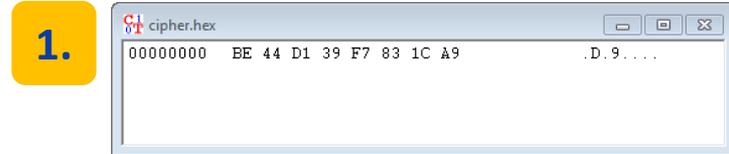
3. Eingabe des teilweise bekannten Schlüssels

4. Start der Brute-Force-Analyse

5. Analyse der Ergebnisse: Kleine Entropie deutet auf eine korrekte Entschlüsselung hin (im Beispiel hat der richtige Kandidat die kleinste Entropie, was aber nicht immer der Fall sein muss).

Menü in CrypTool 1

- „Analyse“ \ „Symmetrische Verfahren (modern)“ \ „DES (ECB)“



Ein One-Time-Pad bitte nur genau einmal benutzen!

Verschlüsselung nicht richtig parametrisiert (Demo in CrypTool 2)

The screenshot shows the CrypTool 2.0 interface with a workflow for demonstrating a One-Time-Pad (OTP) misuse. The workflow starts with two data sources, 'Data A' and 'Data B', which are converted into 'Zeichenketten' (strings) and then 'String-Dekodi' (string-decode) blocks. These strings are then processed by 'String-Kodier' (string-encode) blocks. The resulting strings are XORed with a 'One Time Pad' using 'XOR' blocks. The outputs are 'Encrypted A' and 'Encrypted B', which are then converted back to 'Bitmap' images. A final 'XOR' block takes 'Encrypted A' and 'Encrypted B' as input, resulting in 'Encrypted A XOR Encrypted B', which is a clear image of the letter 'A'. A text box in the bottom left explains the mathematical principle behind this attack.

Berechnung beendet (Zum Anhalten des Workspaces bitte Stop drücken, oder neue Daten eingeben, um eine Neuberechnung durchzuführen).

This template shows why the **One Time Pad** should always be used only once.

Plaintext A and plaintext B are both encrypted using the same one time pad:

$$\text{Encrypted_A} = \text{Plaintext_A XOR One Time Pad}$$
$$\text{Encrypted_B} = \text{Plaintext_B XOR One Time Pad}$$

Both ciphertexts analysed independently do not offer any information if the attacker has not the one time pad. But if the attacker combines Encrypted_A and Encrypted_B using the XOR-function, he can reveal a huge amount of information.

This can easily be explained by looking at the equations:

$$\text{Encrypted_A XOR Encrypted_B} = (\text{Plaintext_A XOR One Time Pad}) \text{ XOR } (\text{Plaintext_B XOR One Time Pad}) = \text{Plaintext_A XOR Plaintext_B}$$

Thus using a one time pad twice an attacker may remove the one time pad completely from both ciphertexts and gain information about both plaintexts.

So using a one time pad more than once is a bad idea :-)

Info: 13:28:07:707: Generated 1 random integers.

ECB- und CBC-Block-Modi von DES und AES als Bild

Verschlüsselung nicht richtig parametrisiert (Chaining) (Demo in CrypTool 2)

CrypTool 2.0 (Nightly Build 5655.1) - Chaining Modes.cwm

Start Bearbeiten Kryptotutorien

Startcenter RSA Cipher Text Encoding... Enigma-Chiffre RSA Encryption 2.5.5 Die Welt der Pr... Block-Modi vo...

Data Smiley Zeichenketten String-Dekodi Plaintext smiley Bildausgabe String-Dekodi String-Kodiert String-Kodiert Zeichenketten String-Dekodi String-Kodiert Zeichenketten String-Dekodi String-Kodiert Zeichenketten String-Dekodi

DES ECB DES CBC

Random Key

Encrypted Smiley ECB

Encrypted Smiley CBC

This template shows the difference between the chaining modes **Electronic Code Book (ECB)** and **Cipher Block Chaining (CBC)**.

The plaintext (the smiley picture) is encrypted using the DES algorithm. Once ECB is used to chain the cipher blocks and once CBC is used.

By comparing both results one can easily see, that even if a picture is encrypted, its content may remain recognizable due a wrong chosen chaining mode. ECB does not chain the cipher blocks while CBC does.

The result using ECB is, that every plaintext block, for instance all white and all black blocks, are encrypted to the same cipher block. So the structure of the picture remains visible.

The result using CBC is, that every plaintext block, even if they were the same, are encrypted to a different cipher block. This happens, because all cipher blocks were connected using the CBC mode. Thus, no structure remains everything is hidden.

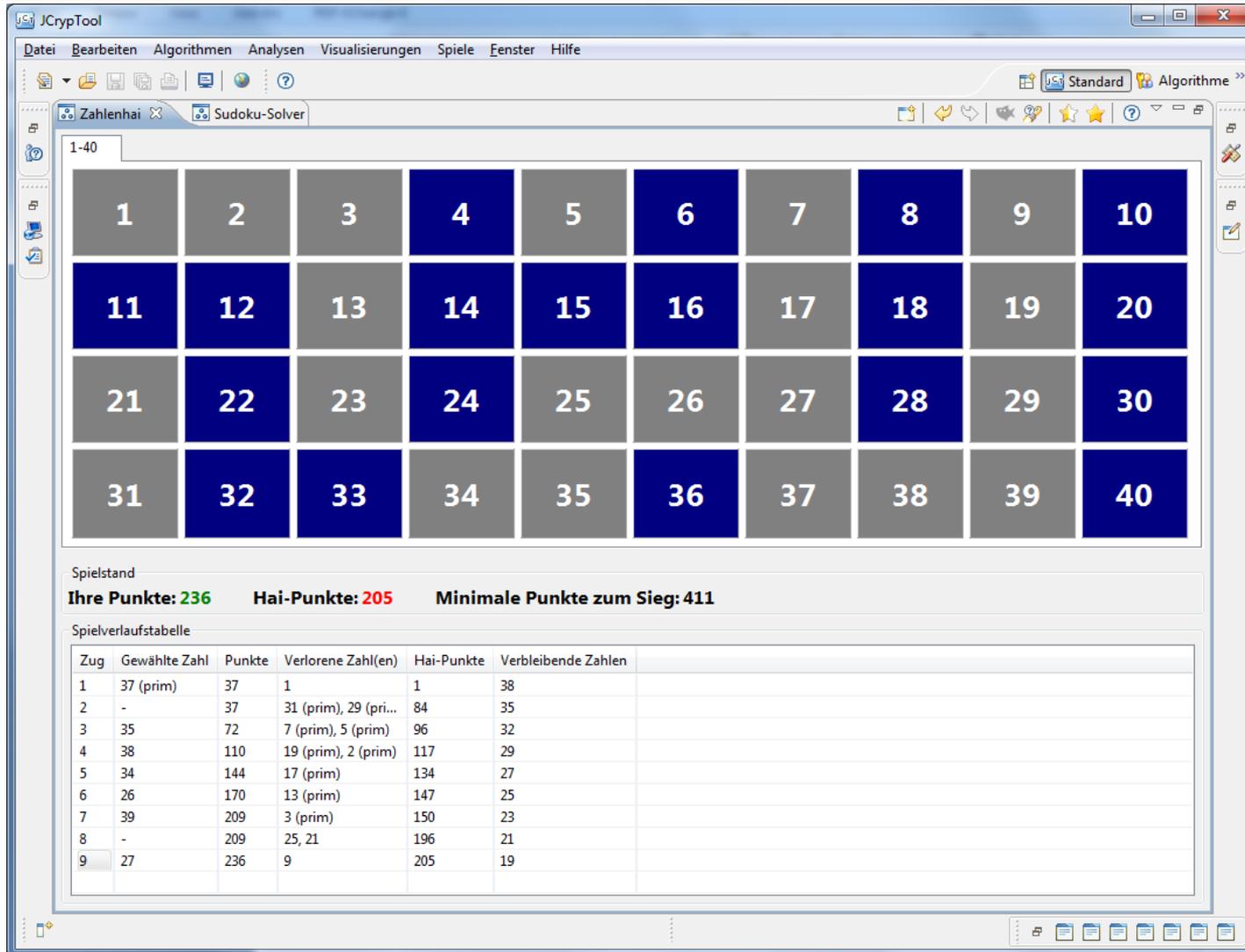
For more information on chaining modes have a look at: http://en.wikipedia.org/wiki/Block_cipher_mode_of_operation

65 %

Info: 14:54:01:618: ExecutionEngine successfully stopped

Lernspiele: Hier der Zahlenhai

Idee war ein didaktisches Mittel zu Primzahlen in Klasse 5 und 6 (Demo in JCT)



Spielstand
Ihre Punkte: 236 **Hai-Punkte: 205** **Minimale Punkte zum Sieg: 411**

Spielverlaufstabelle

Zug	Gewählte Zahl	Punkte	Verlorene Zahl(en)	Hai-Punkte	Verbleibende Zahlen
1	37 (prim)	37	1	1	38
2	-	37	31 (prim), 29 (prim)	84	35
3	35	72	7 (prim), 5 (prim)	96	32
4	38	110	19 (prim), 2 (prim)	117	29
5	34	144	17 (prim)	134	27
6	26	170	13 (prim)	147	25
7	39	209	3 (prim)	150	23
8	-	209	25, 21	196	21
9	27	236	9	205	19

Was bietet CrypTool

4

Ausgewählte Beispiele mit CrypTool 1, CrypTool 2 und JCrypTool

13

Projekt / Ausblick / Kontakt

45



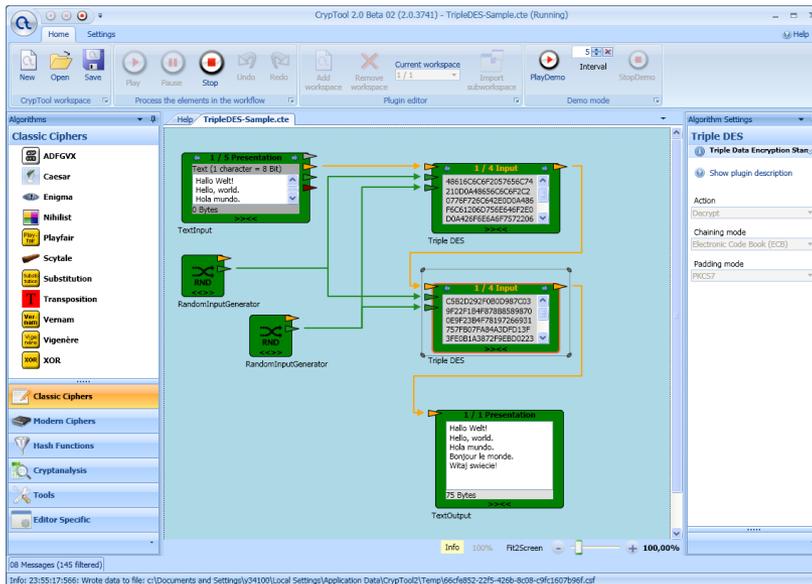
Wir freuen uns über jede weitere Mitarbeit

- Feedback, Kritik, Anregungen und Ideen
- Einbau weiterer Algorithmen, Protokolle, Analysen (Konsistenz und Vollständigkeit)
- Mithilfe bei der Entwicklung (Programmierung, Layout, Übersetzung, Test, Webseiten-Erweiterung)
 - im bisherigen C/C++ Projekt CrypTool 1 (CT1), und
 - in den **aktuellen Projekten** (bevorzugt):
 - C#-Projekt: „CrypTool 2“ = CT2 (<http://www.cryptool.org/de/ct2-machmit-de>)
 - Java-Projekt: „JCrypTool“ = JCT (<http://www.cryptool.org/de/jct-machmit-de>)
 - Browser-Projekt: „CrypTool-Online“ = CTO (<http://www.cryptool-online.org>)
- Insbesondere Lehrstühle, die CrypTool zur Ausbildung verwenden, sind herzlich eingeladen, zur Weiterentwicklung beizutragen.
- Beispiele offener Aufgaben finden sich auf den entsprechenden Entwickler-Seiten (nur englisch):
 - CT2: Siehe die Vorschlagsliste möglicher Aufgaben: <https://www.cryptool.org/trac/CrypTool2/wiki/StudentTasksProposals>
 - JCT: Siehe die Liste Projektideen: <http://sourceforge.net/apps/mediawiki/jcryptool/index.php?title=CurrentDevelopment>
- Bei signifikanten Beiträgen können die Autoren gerne namentlich erwähnt werden (in der Hilfe, im Readme, im About-Dialog und auf der Webseite).
- Derzeit wird das CrypTool-Programmpaket knapp 10.000 mal pro Monat von der CrypTool-Webseite herunter geladen (davon macht die englische Version etwas mehr als 50 % aus).

Aktuelle Entwicklung (1)

Aktiv weiter entwickelt werden die zwei Versionen CT2 und JCT

1. JCT: JCrypTool in Java
 - Plattform-unabhängiger Nachfolger von CT1: Läuft auf Windows, MacOS und Linux
 - Download von: <http://www.cryptool.org/de/jct-downloads-de/jct-downloads-weekly-de>
 - Release Candidate 7 ist verfügbar seit Dez. 2013 (seitdem gibt es wöchentlich einen neuen Weekly Build).
2. CT2: CrypTool mit C#
 - Direkter Nachfolger von CT1: Erlaubt visuelle Programmierung, ...
 - Download von: <http://www.cryptool.org/de/ct2-download-de>
 - Release CT 2.0 ist verfügbar seit Aug. 2014 (Updates für CT 2.1 gibt es seitdem täglich als Nightly Builds).

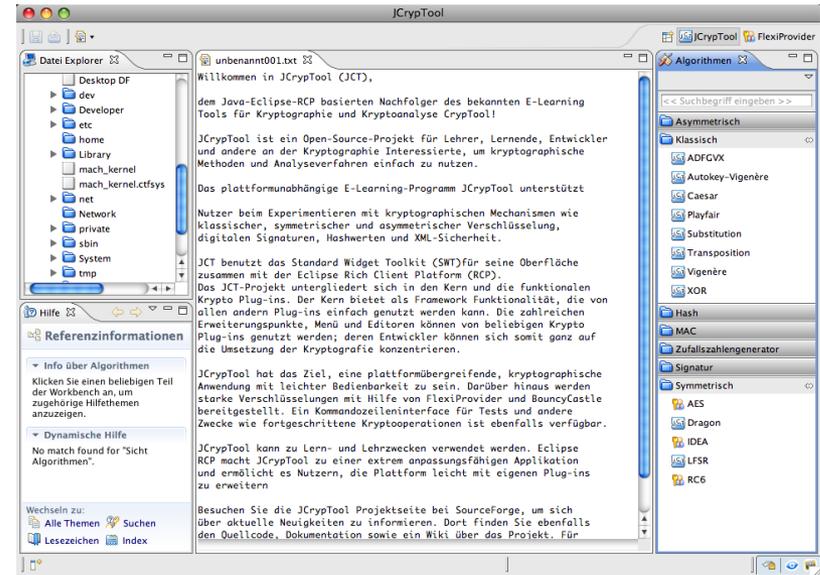
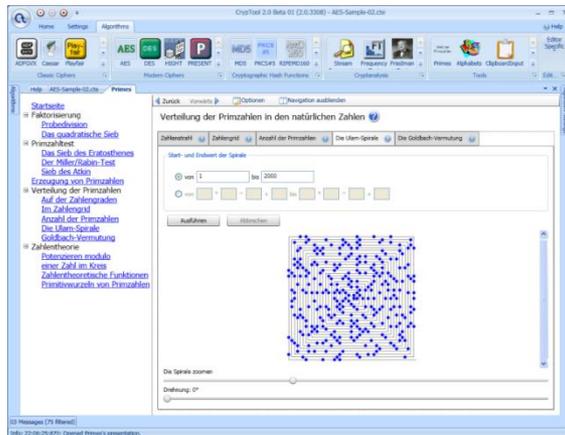
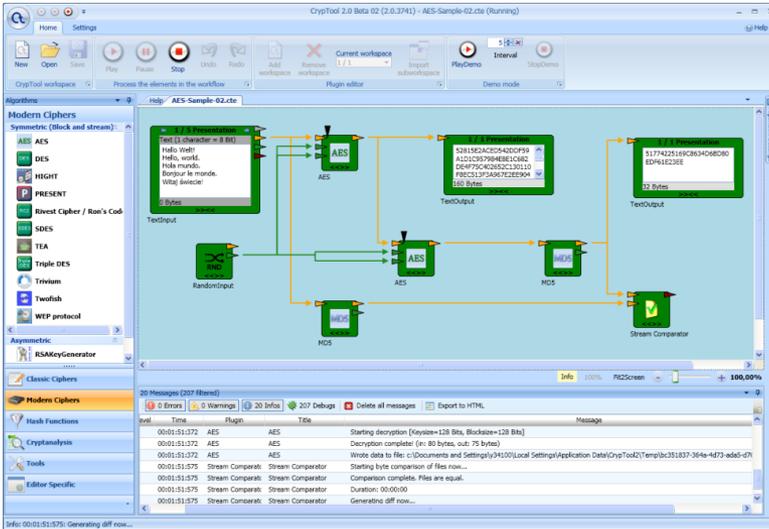


CrypTool 2 (CT2)



JCrypTool (JCT)

Aktuelle Entwicklung (2)



CrypTool 2 (CT2)

JCrypTool (JCT)



Angebot / Vorteile

- Man kann auf einem umfassenden Set aus Algorithmen, inkludierten Bibliotheken und Oberflächenelementen aufsetzen (Re-Use)
- Kostenlose Schulung, wie man in die CrypTool-Programmierung einsteigt
- Vorteil: Der eigene Code aus Seminar-, Diplom-, Bachelor-, Master- und Doktorarbeiten „verschwindet“ nicht, sondern wird weitergepflegt. Hilfe + Feedback beim Erstellen.
- Open-Source: Download aller Sourcen und Binaries der Release- und Beta-Versionen

Eingesetzte Entwicklungsumgebungen für:

- **CrypTool 1.4.31 (CT1):**
Visual C++ .NET in Microsoft Visual Studio 2015 Community Edition (kostenlos)
C++, Perl, .NET >= 4.0, und das Subversion Source-Code-Management
- **CrypTool 2.0 (CT2):**
Visual Studio 2015 Community Edition (kostenlos)
C#, WPF, und das Subversion Source-Code-Management
- **JCrypTool (JCT):**
Eclipse 4.4, RCP, SWT (kostenlos)
Java 1.8, und das GitHub Source-Code-Management

Mitglieder in der Familie der CryptTool-Webseiten:

- **CrypTool-Portal und CT1**
- **CT2-Entwicklerseite**
- **JCT-Entwicklerseite**
- **CrypTool-Online + CrypTool-Mobil**
(Einstieg in Kryptologie und erstes Ausprobieren direkt im Browser, und mit dem Smartphone)
- **CryptoPortal** für Lehrer
- **MysteryTwister C3 (MTC3)** ist ein internationaler Krypto-Wettbewerb.

Start

CRYPTOOL-ONLINE

Über Chiffren Kodierungen Kryptoanalyse Highlights CrypTool-Hauptseite

Start

Was ist CryptTool-Online?

Chiffren Wie funktionieren klassische Chiffrierverfahren? 	Kryptoanalyse Wie kann man den Klartext auch ohne den Schlüssel herausfinden? 
Kodierungen Wo werden Kodierungen eingesetzt und wie funktionieren sie? 	Highlights Weitere interessante Themen, z.B. "Was sind sichere Passwörter?" 

Verschlüsseln direkt im Browser

CrypTool-Online bietet einen spannenden Einblick in die Welt der **Kryptologie**. Es werden eine Vielzahl von Chiffrierverfahren sowie Kodierungen und Analysetools auf einfache Weise, meist anhand eines Beispiels, vorgestellt. Der Schwerpunkt liegt dabei auf einer **verständlichen Erläuterung**, die Interesse an der Kryptographie und der Kryptoanalyse fördern soll. Daher kann mit jedem vorgestellten Verfahren auch direkt auf dieser Website experimentiert werden.

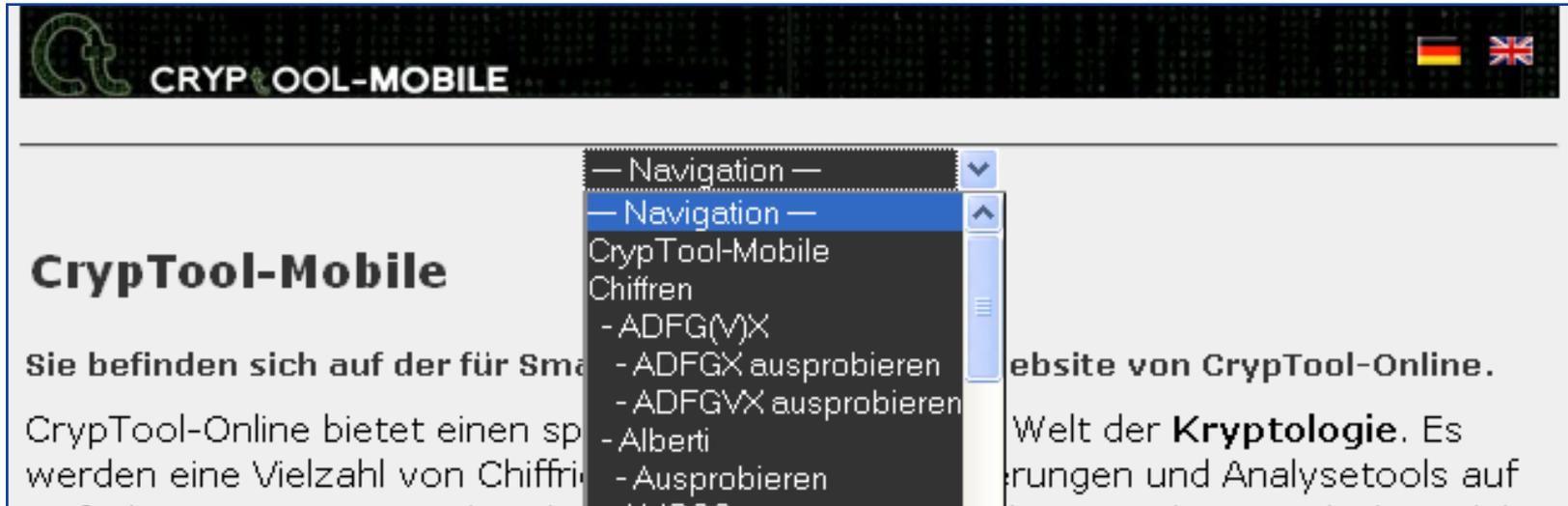
So können Sie in kurzer Zeit die Funktionsweise von historisch bedeutsamen Kryptographieverfahren (wie zum Beispiel das der Enigma, welches den Verlauf des zweiten Weltkriegs entscheidend beeinflusst hat) erlernen und mit den hier angebotenen Tools **selber Texte verschlüsseln**. Sie können aber auch bereits verschlüsselte Texte entschlüsseln und analysieren, um Schwachstellen einer Chiffrierung aufzufindig zu machen.

Bei CryptTool-Online handelt es sich um die Onlinevariante des freien E-Learning-Programms **CrypTool**. Während CryptTool-Online hauptsächlich zum Kennenlernen und Testen der Funktionsweise gedacht ist, können mit der Downloadvariante von CrypTool auch längere Texte bearbeitet und leistungstärkere Analysen durchgeführt werden.

- **Chiffrierverfahren** (unter anderen: ADFGVX, Alberti, Bifid, Caesar, Enigma, Four-Square, Freimaurer, Navajo, Nihilisten, Playfair, Vigenère)
- **Kodierverfahren** (ASCII, Bacon, Base64, Code39, Huffman, Morse [auch zum Raten und Anhören])
- **Analysetools** (unter anderen: Autokorrelation, Häufigkeitsanalyse, n-Gramm-Analyse)
- **Highlights** (unter anderen: AES, Passwort-Generator, Passwort-Check, Matrix-Bildschirmschoner)

Links Kontakt Impressum Sitemap

Copyright © 1998 - 2013 PROJECT / Contributors



Einstieg in Kryptologie und erstes Ausprobieren mit dem Smartphone.

Basierend auf neuester Web-Technologie (AngularJS 2, Bootstrap, JavaScript, Joomla) entwickelt das Projekt gerade mit Studenten und Web-Experten die neue Version, die bis Ende 2016 live gehen soll.



CRYPTOPORTAL für Lehrer

Über Unterrichtsmaterial Linksammlung Registrierung CrypTool Einloggen

Filterkriterien

Land:

Schultyp:

Autor:

Material enthält folgenden Text:

Unterrichtsmaterial

Passend zu den Filterkriterien: 13 Unterrichtsmaterialien
Letzte Aktivität: 18.01.2010

[1] **3 praktische Unterrichtsbeispiele zur Verschlüsselung mit Excel und CrypTool** 0 Kommentare

Autor: Anonym
Land: alle deutschsprachigen Länder
Schultyp: Gymnasien

Materialien zum Workshop "Kryptographie im Unterricht" von Rainer Gürth. Die 3 Beispiele (ROT13, Vigenère und RSA) werden in einem PDF-Arbeitsblatt erläutert und als Excel-Programm angeboten, um sie [...]

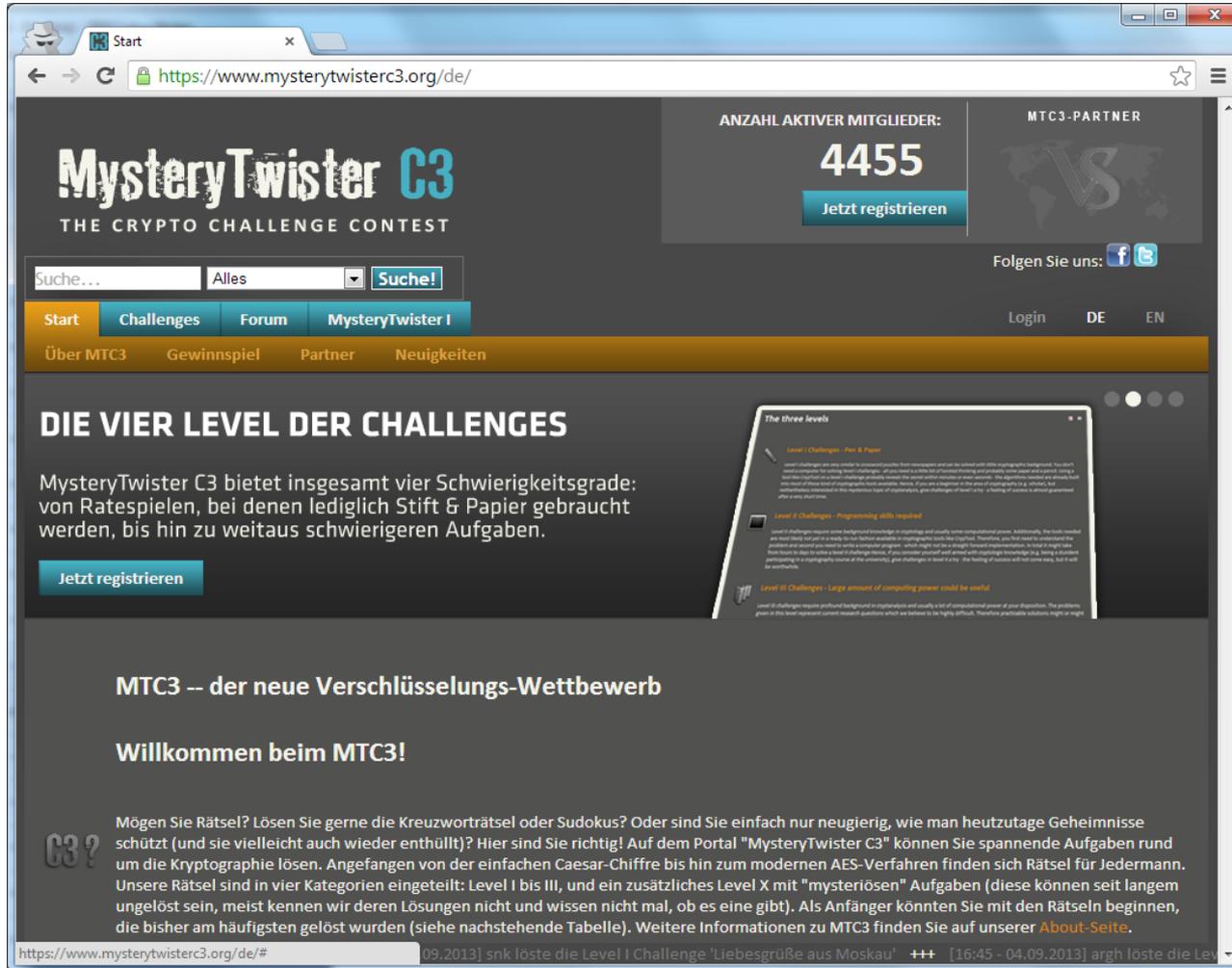
[material.zip](#) 1245 mal heruntergeladen

[2] **Asymmetrische Kryptologie: Schritt-für-Schritt-Anleitung mit CrypTool (01/2010)** 0 Kommentare

Autor: B E
Land: alle deutschsprachigen Länder
Schultyp: alle Schultypen

Diese Anleitung mit über 60 Folien und vielen Screenshots aus CrypTool-1 beschreibt, wie man asymmetrische Krypto-Verfahren zum Verschlüsseln und Signieren verwendet. Die Anleitung kann als komplette [...]

[Asymmetrische Kryptologie am Beispiel RSA entdecken_v1.1.pdf](#) 395 mal heruntergeladen



Ein Wettbewerb
für Anfänger und
Fortgeschrittene
Über 7000
Mitglieder
Über 200 Krypto-
Rätsel
Ein moderiertes
Forum
Eine Hall-of-
Fame
Ein enges
Rennen an der
Spitze
Etwas für Dich?

MysteryTwister C3 (MTC3) ist ein internationaler Krypto-Wettbewerb.



DAS E-Learning-Programm für Kryptologie

- Seit über 15 Jahren ein lebendiges Open-Source-Projekt
- Mehr als 1.000.000 Downloads
- Weltweiter Einsatz in Schulen und Universitäten sowie Firmen und Behörden
- Umfangreiche Online-Hilfe und Dokumentation
- Frei verfügbar, kostenlos, mehrsprachig

Weitere Aktivitäten im CrypTool-Projekt:

- Internationaler Krypto-Wettbewerb MTC3 mit Rätseln für alle
- Schülerkrypto-Kurse



Prof. Bernhard Esslinger
(Leiter des CrypTool-Projekts, Universität Siegen)

bernhard.esslinger@uni-siegen.de

www.cryptool.org