CRYPtOOL

KRYPTOLOGIE MIT CRYPTOOL v 1.4.30

Praktische Einführung in Kryptographie und Kryptoanalyse

Umfang, Technik und Zukunft von CrypTool

Prof. Bernhard Esslinger und CrypTool-Team, Juli 2010

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

Übersicht (I)

I. CrypTool und Kryptologie – Überblick

1. Definition und Bedeutung der Kryptologie
2. Das CrypTool-Projekt
3. Beispiele klassischer Verschlüsselungsverfahren
4. Erkenntnisse aus der Entwicklung der Kryptographie

II. Was bietet CrypTool?

1. Überblick
2. Beispiele zur Interaktion
3. Herausforderungen für Entwickler

III. Ausgewählte Beispiele

1. RSA-Verschlüsselung / Primzahltests / Hybridverschlüsselung und Digitale Zertifikate / SSL
2. Elektronische Signatur visualisiert
3. Angriff auf RSA-Verschlüsselung (Modul N zu kurz)
4. Analyse der Verschlüsselung im PSION 5
5. Schwache DES-Schlüssel
6. Auffinden von Schlüsselmaterial („NSA-Key“)
7. Angriff auf Digitale Signatur durch Suche nach Hashkollisionen
8. Authentisierung in einer Client-Server-Umgebung
9. Demonstration eines Seitenkanalangriffs (auf ein Hybridverschlüsselungsprotokoll) (…)

Übersicht (II)

III. Ausgewählte Beispiele

10. Angriffe auf RSA per Gitterreduktion
11. Zufallsanalyse mit 3-D Visualisierung
12. Secret Sharing als Anwendung des Chinesischen Restsatzverfahrens (CRT) und nach Shamir
13. Anwendung des CRT in der Astronomie (Lösung linearer Kongruenzsysteme)
14. Visualisierung von symmetrischen Verschlüsselungsverfahren mit ANIMAL
15. Visualisierungen von AES
16. Visualisierung der Enigma-Verschlüsselung
17. Visualisierung von sicherer E-Mail mit S/MIME
18. Erzeugung eines Message Authentication Code (HMAC)
19. Hash-Demo
20. Lernprogramm zur Zahlentheorie und zur asymmetrischer Verschlüsselung
21. Punktaddition auf elliptischen Kurven
22. Passwort-Qualitätsmesser und Passwort-Entropie
23. Brute-Force-Analyse
24. Skytale / Gartenzaun
25. Hill-Verschlüsselung / Hill-Analyse
26. CrypTool Online-Hilfe

IV. Projekt / Ausblick / Kontakt

Inhalt

I. **CrypTool und Kryptologie – Überblick**

II. Was bietet CrypTool?

III. Ausgewählte Beispiele

IV. Projekt / Ausblick / Kontakt

Bedeutung der Kryptographie

Einsatzbeispiele für Kryptographie

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

- Kryptographie ist schon lange nicht mehr nur auf Agenten, Diplomaten und Militärs begrenzt. 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!

Definition Kryptologie und Kryptographie

Kryptologie *(vom Griechischen kryptós, "versteckt," und lógos, "Wort") ist die Wissenschaft von sicherer (allgemein geheimer) Kommunikation. Diese Sicherheit bedingt, dass die berechtigten Teilnehmer in der Lage sind, eine Nachricht mit Hilfe eines Schlüssels in einen Geheimtext zu transferieren und zurück. Obwohl der Geheimtext für jemand ohne den geheimen Schlüssel unlesbar und unfälschbar ist, kann der berechtigte Empfänger entweder das Chiffrat entschlüsseln, um die den verborgenen Klartext wieder zu erhalten, oder verifizieren, dass die Nachricht aller Wahrscheinlichkeit nach von jemand geschickt wurde, der den richtigen Schlüssel besaß.*

Kryptographie *beschäftigte sich ursprünglich damit, für Vertraulichkeit von geschriebenen Nachrichten zu sorgen. Die kryptographischen Prinzipien werden jedoch genauso angewandt, um den Informationsfluss zwischen Computern oder Fernsehsignale zu verschlüsseln. ...Heutzutage liefert die moderne (mathematische) Wissenschaft der Kryptologie nicht nur Verfahren zur Verschlüsselung, sondern auch zur Integrität, für elektronische Signaturen, für Zufallszahlen, sicheren Schlüsselaustausch, sichere Container, elektronische Wahlen und elektronisches Geld. Damit kommen diese Verfahren in einer breiten Palette von Anwendungen des modernen Lebens zum Einsatz.*

Quelle: Britannica (www.britannica.com)

Ähnliche Definitionen finden sich auch auf Wikipedia:

- http://de.wikipedia.org/wiki/Kryptologie
- http://de.wikipedia.org/wiki/Kryptografie

Sicherheitsziele der Kryptographie

- **Vertraulichkeit *(Confidentiality)***

  Lesen des eigentlichen Inhalts für Unbefugte „praktisch“ unmöglich machen

- **Authentifizierung *(Authentication)***

  Identitätsbeweis des Senders gegenüber dem Empfänger einer Nachricht

- **Integrität *(Integrity)***

  Eigenschaft, die bedeutet, 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)

CrypTool-Projekt

- Ursprung im Awareness-Programm einer Großbank (betriebliche Ausbildung)
  → **Sensibilisierung der Mitarbeiter**
- Entwickelt in Kooperation mit Hochschulen (Verbesserung der Lehre)
  → **Mediendidaktischer Anspruch**

1998	**Projektstart** – Aufwand bisher mehr als 40 Mannjahre
2000	CrypTool als **Freeware** verfügbar für Windows
2002	CrypTool auf der **Bürger-CD des BSI** „Ins Internet – mit Sicherheit“
2003	CrypTool wird **Open-Source** – Hosting durch die Uni Darmstadt (Fr. Prof. Eckert)
2007	CrypTool in deutsch, englisch, polnisch und spanisch
2008	.NET- und Java-Versionen begonnen – Hosting durch die Uni Duisburg (Hr. Prof. Weis) und SourceForge
2010	CT1 auch in Serbisch; .NET- und Java-Release-Versionen sollen veröffentlicht werden

- **Auszeichnungen**

2004	TeleTrusT	(TTT Förderpreis)
2004	NRW	(IT-Sicherheitspreis NRW)
2004	RSA Europe	(Finalist beim European Information Security Award)
2008	"Ausgewählter Ort" bei der Standortinitiative "Deutschland – Land der Ideen"	

- **Entwickler**
  - Entwickelt von Mitarbeitern verschiedener Firmen und Universitäten, Schülern + Studenten
  - → Weitere Projekt-Mitarbeiter oder verwertbare vorhandene Sourcen sind immer herzlich willkommen (z.Zt. arbeiten ca. 50 Leute weltweit mit).

Beispiele aus der klassischen Kryptographie (1)

Älteste bekannte Verschlüsselungsverfahren

- **Tattoo auf kahlgeschorenen Kopf eines Sklaven** (verdeckt von nachgewachsenen Haaren)
- **Atbash** (um 600 v. Chr.)
  - Hebräische Geheimschrift, umgedrehtes Alphabet
- **Skytale von Sparta** (etwa 500 v. Chr.)
  - Beschrieben vom griechischen Historiker/Schriftsteller Plutarch (45 - 125 n. Chr.)
  - Zwei Zylinder (Holzstäbe) mit gleichem Durchmesser
  - Transposition (Zeichen des Klartextes werden umsortiert)

Beispiele aus der klassischen Kryptographie (2)

Caesar-Verschlüsselung (mono-alphabetische Substitution)

- **Caesar-Verschlüsselung** (Julius Cäsar, 100 - 44 v.Chr.)
- Einfache Substitutionschiffre

```
                GALLIA   EST   OMNIS   DIVISA ...

Klartextalphabet:   ABCDEFGHIJKLMNOPQRSTUVWXYZ

Geheimtextalphabet: DEFGHIJKLMNOPQRSTUVWXYZABC

                JDOOLD   HVW   RPQLV   GLYLVD ...
```

- **Angriff:** Häufigkeitsanalyse (typische Verteilung von Zeichen)

Vorführung mit CrypTool über folgende Menüs:

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

Beispiele aus der klassischen Kryptographie (3)

Vigenère-Verschlüsselung (poly-alphabetische Substitution)

- **Vigenère-Verschlüsselung** (Blaise de Vigenère, 1523-1596)
- Verschlüsselung mit einem Schlüsselwort unter Nutzung einer Schlüsseltabelle
- Beispiel
  Schlüsselwort: **`CHIFFRE`**
  Verschlüsselung: **`VIGENERE`** wird zu **`XPOJSVVG`**
- Das Klartextzeichen wird ersetzt durch das Zeichen in der Zeile des Klartextes (bspw. V) und in der Spalte des Schlüsselwortzeichens (bspw. c). Das nächste Zeichen (bspw. I) wird in der Spalte des zweiten Zeichens des Schlüsselwortes (bspw. h) abgelesen, usw.
- Sobald man beim letzten Zeichen des Schlüsselwortes angekommen ist, beginnt man wieder mit dem ersten Zeichen des Schlüsselwortes.
- **Angriff** (u. a. durch Kasiski-Test): Es können gleiche Klartextzeichenkombinationen mit jeweils der gleichen Geheimtextzeichenkombination auftreten. Der Abstand dieser Muster kann nun genutzt werden, um die Schlüsselwortlänge zu bestimmen. Eine anschließende Häufigkeitsanalyse kann dann den Schlüssel bestimmen.

Tableau carré, dit « Carré de Vigenère »

Beispiele aus der klassischen Kryptographie (4)

Weitere Verfahren der klassischen Kryptographie

- **Homophone Substitution**
- **Playfair** (erfunden 1854 von Sir Charles Wheatstone, 1802-1875)
  - veröffentlicht von Baron Lyon Playfair
  - Substitution eines Buchstabenpaares durch ein anderes anhand einer quadratischen Alphabetsanordnung
- **Übermittlung von Buchseiten**
  - Adaption des One-Time-Pads (OTP)
- **Lochschablonen** (Fleißner)
- **Permutationsverschlüsselung**
  - „Doppelwürfel“

    (Reine Transposition / sehr effektiv)

Kryptographie in der Neuzeit

Entwicklung der Kryptographie in den letzten 100 Jahren bis 1970

Klassische Verfahren

- werden teilweise heute noch eingesetzt. (nicht alles geht per Computer…)
- und deren Prinzipien **Transposition** und **Substitution** fanden Eingang beim Design moderner Algorithmen:
  Kombination der einfacheren Operationen (eine Art der Mehrfach-Verschlüsselung, cascades of ciphers) auf Bit-Ebene, Blockbildung, Runden.

Verschlüsselungsverfahren wurden

- weiter **verfeinert**,
- **mechanisiert** bzw. **computerisiert**, und
- bleiben zunächst **symmetrisch**.

Beispiel erste Hälfte 20. Jahrhundert

Elektromechanische Verschlüsselungsmaschinen (Rotormaschinen)

Enigma-Verschlüsselung (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.
- Das polnische Cipher Bureau brach die Vorkriegs-Enigma schon 1932.
- Darauf aufbauend wurde die Enigma gebrochen mit massivem Einsatz [etwas 7000 Personen in UK, mit den ersten Entschlüsselungsmaschinen sowie erbeuteten Original-Maschinen, Nutzen abgefangener täglicher Statusmeldungen (z.B. Wetternachrichten), …].
- **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.“

(http://de.wikipedia.org/wiki/Enigma_%28Maschine%29 vom 06.03.2006)

Kryptographie – Entscheidende Erkenntnisse (1)

Kerckhoffs-Prinzip (formuliert 1883)

- Trennung von Algorithmus (Verfahren) und Schlüssel
  z.B. bei Caesar:
  - Algorithmus: „Verschiebe Alphabet um eine bestimmte Anzahl Positionen zyklisch nach links“
  - Schlüssel: Diese „bestimmte Anzahl Positionen“ (bei Caesar: 3)
- Kerckhoffs-Prinzip:
  Das Geheimnis liegt im Schlüssel und nicht im Algorithmus bzw. keine „security through obscurity“.

One-Time-Pad – Shannon / Vernam

- Beweisbar sicher, jedoch praktisch kaum anwendbar (benutzt beim Roten Telefon*).

Shannons Konzepte: Konfusion und Diffusion

- Zusammenhang zwischen M, C und K möglichst komplex (M=Message, C=Cipher, K=Key)
- Jedes Chiffrezeichen sollte von möglichst vielen Klartextzeichen und vom gesamten Schlüssel abhängen
- „Avalanche effect“ (kleine Änderung, große Wirkung)

Trapdoor-Function (Falltür, Einweg-Funktion)

- in einer Richtung schnell, in die andere (ohne Geheim-Information) nicht
- nur mit dem Geheimnis geht auch die andere Richtung (Zugang zur Falltür)

* Siehe http://de.wikipedia.org/wiki/Hei%C3%9Fer_Draht

Beispiel für die Verletzung des Kerckhoffs-Prinzips

Geheimnis sollte nur im Schlüssel und nicht im Algorithmus liegen

- **Handy-Verschlüsselung angeblich geknackt** (07.12.1999)

  „Die beiden israelischen Kryptologen Alex Biryukov und Adi Shamir haben Medienberichten zufolge den Verschlüsselungsalgorithmus geknackt, der GSM-Handy-Telefonate auf der Funkstrecke zur Mobiltelefon-Basisstation schützt. Das Verfahren soll mit einem handelsüblichen PC auskommen, der mit 128 MByte RAM und zwei 73 GByte Festplatten ausgestattet ist. Auf diesem soll das Programm der Forscher durch eine Analyse der ersten zwei Gesprächsminuten in weniger als einer Sekunde den verwendeten Schlüssel errechnen können. Umstritten ist, ob und mit welchem Aufwand es möglich ist, die Gespräche überhaupt abzufangen, um sie anschließend zu dechiffrieren. Eines zeigen die Vorfälle um die GSM-Verschlüsselungsalgorithmen A5/1 und A5/2 aber schon jetzt deutlich: ***Der Versuch, Krypto-Verfahren geheim zu halten, dient nicht der Sicherheit.*** *Das hat anscheinend auch die GSM-Association gelernt: Ihr Sicherheitsdirektor James Moran äußerte dem Online-Magazin Wired gegenüber, dass man künftige Algorithmen von vorneherein offen legen will, um der Fachwelt eine Prüfung zu ermöglichen.“*
  [http://www.heise.de/newsticker/meldung/7183]

- **Weiteres Beispiel: Netscape Navigator** legte 1999 die Passworte für den Zugriff auf E-Mail-Server noch proprietär schwach verschlüsselt ab.vigator

Beispiel für eine One-Time-Pad-Adaption

Kleiderbügel einer Stasi-Spionin
mit verstecktem One-Time-Pad
(Aus: *Spiegel Spezial 1/1990)*

Menü:
„Ver-/Entschlüsseln“ \
„Symmetrisch (klassisch)“ \
„Vernam“

Schlüsselverteilungsproblem

Schlüsselverteilung bei symmetrischer Verschlüsselung

Wenn **2 Personen** miteinander mit einer symmetrischen Verschlüsselung kommunizieren, brauchen sie ***einen* gemeinsamen und geheimen Schlüssel**.

Wenn bei n Personen jeder mit jedem geheim kommunizieren möchte, dann braucht man $\mathbf{S_n = n * (n-1) / 2}$ Schlüssel.

Das sind bei

n = 100 Personen bereits

$\mathbf{S_{100} = 4.950}$ Schlüssel; bei

n = 1.000 Personen sind es

$\mathbf{S_{1000} = 499.500}$ Schlüssel.

(Quadratischer Anstieg:

Faktor 10 mehr Personen,

Faktor 100 mehr Schlüssel)

Entwicklung der Schlüsselzahl

Kryptographie – Entscheidende Erkenntnisse (2)

Lösung des Schlüsselverteilungsproblems durch asymmetrische Kryptographie

Asymmetrische Kryptographie

- Jahrhunderte lang glaubte man: Sender und Empfänger brauchen dasselbe Geheimnis.
- Neu: Jeder Teilnehmer hat ein Schlüsselpaar („Lösung“ des Schlüsselverteilungsproblems)

Asymmetrische Verschlüsselung

- „Jeder kann ein Vorhängeschloss einschnappen lassen oder einen Brief in einen Kasten werfen“
- MIT, 1977: Leonard Adleman, Ron Rivest, Adi Shamir (bekannt durch RSA)
- GCHQ, Cheltenham, 1973: James Ellis, Clifford Cocks (am 18.12.1997 öffentlich zugegeben)

Schlüsselverteilung

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

Sicherheit in offenen Netzen (wie dem Internet) wäre ohne asymmetrische Kryptographie extrem teuer und komplex!

Durchführung von Ver- und Entschlüsselung

Symmetrische und asymmetrische Verschlüsselung

geheim

a) Symmetrische Verschlüsselung: $\mathbf{K_E = K_D}$ (z.B. AES)

b) Asymmetrische Verschlüsselung: $\mathbf{K_E \neq K_D}$ (z.B. RSA)

öffentlich privat/geheim

Kryptographie – Entscheidende Erkenntnisse (3)

Steigende Bedeutung der Mathematik und der Informationstechnologie

- **Moderne Kryptographie** basiert stärker auf **Mathematik**
  - Trotzdem gibt es weiter symmetrische Verfahren wie den AES (bessere Performance und kürzere Schlüssellängen als die auf rein mathematischen Problemstellungen beruhenden asymmetrischen Verfahren).
- Die Sicherheit praktisch eingesetzter Verfahren hängt entscheidend vom Stand der **Mathematik** und der **Informationstechnologie** (IT) ab.
  - Berechnungskomplexität (d.h. Rechenaufwand in Abhängigkeit von der Schlüssellänge, Speicherplatzbedarf, Datenkomplexität)
    → siehe aktuell RSA: Bernstein, TWIRL-Device, RSA-768 (CrypTool-Skript, Kap. 4.11.3)
  - Sehr hohe Intensität in der aktuellen Forschung
    Faktorisierung, nicht-parallelisierbare Algorithmen (wegen Quantencomputing), besseres Verständnis von Protokoll-Schwächen und Zufallszahlengeneratoren, usw.
- Entscheidender Irrtum: *„Echte Mathematik“* hat keine Auswirkungen auf den Krieg. (G.H. Hardy, 1940)
- Hersteller entdecken **Sicherheit** als ein zentrales **Kaufkriterium.**

Demo mit CrypTool

- ***Statistische Analyse***
- ***Zweimal nacheinander ist nicht immer besser:***

  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 (Ausgabe-Key, Brute-Force-Analyse)***

Inhalt

I. CrypTool und Kryptologie – Überblick

II. Was bietet CrypTool?

III. Ausgewählte Beispiele

IV. Projekt / Ausblick / Kontakt

Was bietet CrypTool?

E-Learning

1. Was ist CrypTool?

- Kostenloses Programm mit graphischer Oberfläche
- Kryptographische Verfahren anwenden *und* analysieren
- Sehr umfangreiche Online-Hilfe; ohne tieferes Kryptographiewissen verständlich
- Enthält fast alle State-of-the-art-Kryptographiefunktionen
- „Spielerischer“ Einstieg in moderne und klassische Kryptographie
- Kein „Hackertool“

2. 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

3. Zielgruppe

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

Inhalt des Programmpakets

Deutsch, Englisch, Polnisch, Spanisch und Serbisch

CrypTool-Programm

- Alle Funktionen integriert in *einem* Programm mit einheitlicher graphischer Oberfläche
- Läuft unter Win32
- Nutzt Kryptographiefunktionen aus den Bibliotheken von Secude, cryptovision und OpenSSL
- Langzahlarithmetik per Miracl und GMP, Gitterbasenreduktion per NTL (V. Shoup)

AES-Tool

- Standalone-Programm zur AES-Verschlüsselung (selbst extrahierend)

Lernbeispiel

- Der „Zahlenhai“ fördert das Verständnis für Teiler und Primzahlen.

Umfangreiche Online-Hilfe (HTML-Help)

- Kontextsensitive Hilfe mit F1 für *alle* Programmfunktionen (auch auf Menüs)
- Ausführliche Benutzungs-Szenarien (Tutorials) für viele Programmfunktionen

Skript (.pdf-Datei) mit Hintergrundinformationen

- Verschlüsselungsverfahren • Primzahlen/Faktorisierung • Digitale Signatur
- Elliptische Kurven • Public Key-Zertifizierung • Elementare Zahlentheorie • Krypto 2020 • Sage

Zwei Kurzgeschichten mit Bezug zur Kryptographie von Dr. C. Elsner

- „Der Dialog der Schwestern“ (eine RSA-Variante als Schlüsselelement)
- „Das chinesische Labyrinth“ (zahlentheoretische Aufgaben für Marco Polo)

Authorware-Lernprogramm zur Zahlentheorie

Funktionsumfang (1)

Kryptographie

Verschlüsselungsklassiker

- Caesar (und ROT-13)
- Monoalphabetische Substitution (und Atbash)
- Vigenère
- Hill
- Homophone Substitution
- Playfair
- ADFGVX
- Byteweise Addition
- XOR
- Vernam
- Permutation (Gartenzaun, Skytale, Doppelwürfel, …)
- Solitaire

Optionen zum besseren Nachvollziehen von Literaturbeispielen

- Alphabet wählbar
- Behandlung von Leerzeichen etc. einstellbar

Kryptoanalyse

Angriffe auf klassische Verfahren

- Ciphertext-Only
  - Caesar
  - Vigenère (nach Friedman + Schrödel)
  - 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
- Base64 / UU-Encode

Funktionsumfang (2)

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)

Funktionsumfang (3)

Kryptographie

Digitale Signatur

- RSA mit X.509-Zertifikaten
  - Signatur zusätzlich visualisiert
- 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 mod 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 (Geburtstagsparadox) (bis 40 Bit in etwa 5 min)

Analyse von Zufallsdaten

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

Funktionsumfang (4)

Visualisierungen / Demos

- Caesar, Vigenère, Nihilist, DES (mit ANIMAL)
- Enigma (Flash)
- Rijndael/AES (zweimal mit Flash, einmal mit Java)
- Hybride Ver- und Entschlüsselung (AES-RSA und AES-ECC)
- Erzeugung und Verifikation von Signaturen
- Diffie-Hellman-Schlüsselaustausch
- Secret Sharing (mit CRT oder mit dem Schwellenwertschema nach Shamir)
- Challenge-Response-Verfahren (Authentisierung im Netz)
- Seitenkanalangriff
- Sichere E-Mail mit dem S/MIME-Protokoll (mit Java und Flash)
- Grafische 3-D-Darstellung von (Zufalls-)Datenströmen
- Sensibilität von Hashfunktionen bezüglich Änderungen an den Daten
- Zahlentheorie und RSA-Kryptosystem (mit Authorware)

Funktionsumfang (5)

Weitere Funktionen

- Verschiedene Funktionen zu RSA und Primzahlen
- Homophone und Permutationsverschlüsselung (Doppelwürfel)
- PKCS #12-Import/Export für PSEs (Personal Security Environment)
- Hashwerte großer Dateien berechnen, ohne sie zu laden
- Generischer Brute-Force-Angriff auf beliebige moderne symmetrische Algorithmen
- ECC-Demo (als Java-Applikation)
- Passwort-Qualitätsmesser (PQM) und Passwort-Entropie
- Und vieles mehr …

Sprachstruktur analysieren

Anzahl Einzelzeichen, n-Gramme, Entropie

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

Nr.	Zeichen...	Häufigkeit in %	Häufigkeit
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 der Interaktivität (1)

Demo per CrypTool

Vigenère-Analyse

Das Ergebnis der Vigenère-Analyse kann manuell nachbearbeitet werden (gefundene Schlüssellänge ändern)

1. Datei „Startbeispiel-de.txt“ mit **TEST** verschlüsseln
   - *„Ver-/Entschlüsseln“ \ „Symmetrisch (klassisch)“ \ „Vigenère…“*
   - Eingabe TEST ⇨ *„Verschlüsseln“*

   Analyse der Verschlüsselung
   - *„Analyse“ / „Symmetrische Verschlüsselung (klassisch)“ \ „Ciphertext-Only“ \ „Vigenère“*
   - Schlüssellänge 4, Ermittelter Schlüssel TEST ✓

2. Datei „Startbeispiel-de.txt“ mit **TESTETE** verschlüsseln
   - *„Ver-/Entschlüsseln“ \ „Symmetrisch (klassisch)“ \ „Vigenère…“*
   - Eingabe TESTETE ⇨ *„Verschlüsseln“*

   Analyse der Verschlüsselung
   - *„Analyse“ \ „Symmetrische Verschlüsselung (klassisch)“ \ „Ciphertext-Only“ \ „Vigenère“*
   - Schlüssellänge 14 – nicht korrekt ✗
   - Schlüssellänge wird angepasst (automatisch – könnte aber auch manuell angepasst werden)
   - Ermittelter Schlüssel TESTETE ✓

Demonstration der Interaktivität (2)

Automatisierte Primzahlzerlegung

Demo per CrypTool

Primzahlzerlegung mit Hilfe von Faktorisierungsverfahren

- Verschiedene Verfahren werden in mehreren Threads parallel ausgeführt
- Alle Verfahren haben bestimmte Vor- und Nachteile
  (z.B. erkennen bestimmte Verfahren nur kleine Faktoren)

Faktorisierungs-Beispiel 1

48-stellige Dezimalzahl

316775895367314538931177095642205088158145887517
=
3 * 1129 * 6353 * 1159777 * 22383173213963 * 567102977853788110597

Faktorisierungs-Beispiel 2

75-stellige Dezimalzahl

2^250 - 1
=
3 * 11 * 31 * 251 * 601 * 1801 * 4051 * 229668251 * 269089806001 * 4710883168879506001 * 5519485418336288303251

Menü: „Einzelverfahren“ \ „RSA-Kryptosystem“ \ „Faktorisieren einer Zahl“

Konzepte zur Benutzerfreundlichkeit

1. Kontextsensitive 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.

2. Einfügen von Schlüsseln in die Schlüsseleingabe-Maske

- Mit Strg-V (Paste) kann man immer einfügen, was im Clipboard steht.
- Schon benutzte Schlüssel können aus 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 Speicher benutzt, der pro Verfahren zur Verfügung steht (nützlich bei „besonderen“ Schlüsseln wie der homophonen Verschlüsselung).

Ikonen-Leiste

Herausforderungen für den Programmierer

1. **Verschiedene Funktionen parallel laufen lassen**
   - Bei der Faktorisierung laufen die verschiedenen Algorithmen in Threads.
2. **Hohe Performance**
   - Bei der Anwendung des Geburtstagsparadoxons zum Finden von Hashkollisionen oder bei der Brute-Force-Analyse.
3. **Speicherbeschränkung beachten**
   - Beim Floyd-Algorithmus (Mappings für das Finden von Hashkollisionen) oder beim Quadratischen Sieb.
4. **Zeitmessung und -abschätzung**
   - Ausgabe der Restzeit (z.B. bei Brute-Force-Analyse)

5. **Wiederverwendung / Integration**
   - Masken zur Primzahlgenerierung
   - RSA-Kryptosystem (schaltet nach erfolgreicher Attacke von der Ansicht des Public-Key-Anwenders zur Ansicht des Private-Key-Besitzers)
6. **Automatisierung der Konsistenz der Funktionen, der GUI und der Online-Hilfe** (inklusive verschiedener Sprachen und den unterstützten Windows-BS: XP, Vista und Win7)

Inhalt

I. CrypTool und Kryptologie – Überblick

II. Was bietet CrypTool?

III. Ausgewählte Beispiele

IV. Projekt / Ausblick / Kontakt

CrypTool-Anwendungsbeispiele

Übersicht der Beispiele

1. Verschlüsselung mit RSA / Primzahltests / Hybridverschlüsselung und Digitale Zertifikate / SSL
2. Elektronische Signatur visualisiert
3. Angriff auf RSA-Verschlüsselung (Modul N zu kurz)
4. Analyse der Verschlüsselung im PSION 5
5. Schwache DES-Schlüssel
6. Auffinden von Schlüsselmaterial („NSA-Key“)
7. Angriff auf Digitale Signatur durch Suche nach Hashkollisionen
8. Authentisierung in einer Client-Server-Umgebung
9. Demonstration eines Seitenkanalangriffs (auf ein Hybridverschlüsselungsprotokoll)
10. Angriffe auf RSA mittels Gitterreduktion
11. Zufallsanalyse mit 3-D-Visualisierung
12. Secret Sharing als Anwendung des Chinesischen Restsatzverfahrens (CRT) und nach Shamir
13. Anwendung des CRT in der Astronomie (Lösung linearer Kongruenzsysteme)
14. Visualisierung von symmetrischen Verschlüsselungsverfahren mit ANIMAL
15. Visualisierungen von AES
16. Visualisierung der Enigma-Verschlüsselung
17. Visualisierung von sicherer E-Mail mit S/MIME
18. Erzeugung eines Message Authentication Code (HMAC)
19. Hash-Demo
20. Lernprogramm zur Zahlentheorie und zur asymmetrischen Verschlüsselung
21. Punktaddition auf elliptischen Kurven
22. Passwort-Qualitätsmesser und Passwort-Entropie
23. Brute-Force-Analyse
24. Skytale / Gartenzaun
25. Hill-Verschlüsselung / Hill-Analyse
26. CrypTool Online-Hilfe

Anwendungsbeispiele (1)

Verschlüsselung mit RSA

- **Grundlage für z.B. SSL-Protokoll (Zugriff auf gesicherte Web-Seiten)**
- **Asymmetrische Verschlüsselung mit RSA**
  - Jeder Benutzer hat ein Schlüsselpaar – einen öffentlichen und einen privaten.
  - Sender verschlüsselt mit dem öffentlichen Schlüssel (*public key*) des Empfängers.
  - Empfänger entschlüsselt mit seinem privaten Schlüssel (*private key*).
- **Einsatz i. d. R. in Kombination mit symmetrischen Verfahren (Hybridverschlüsselung)**

Anwendungsbeispiele (1)

Verschlüsselung mit RSA – Mathematischer Hintergrund / Verfahren

- Öffentlicher Schlüssel (public key): (n, e) [oft wird der Modulus n auch groß N geschrieben]
- Privater Schlüssel (private key): (d)

wobei

p, q sind große zufällig gewählte Primzahlen mit $n = p*q$;

d wird unter den NB $ggT[\varphi(n),e] = 1$; $e*d \equiv 1 \bmod \varphi(n)$; bestimmt.

Ver- und Entschlüsselungs-Operation: $(m^e)^d \equiv m \bmod n$

- n ist der Modulus (seine Länge ist die „Schlüssellänge“ beim RSA-Verfahren).
- ggT = größter gemeinsamer Teiler.
- $\varphi(n)$ ist die Eulersche Phi-Funktion.

Vorgehen

- Transformation von Nachrichten in binäre Repräsentation
- Nachricht $m = m_1,\ldots,m_k$ blockweise verschlüsseln, wobei für alle m_j gilt: $0 \leq m_j < n$; also maximale Blockgröße r so, dass gilt: $2^r \leq n$ $(2^r-1 < n)$

Anwendungsbeispiele (1)

Primzahltests – Für RSA werden große Primzahlen benötigt

- Schnelle probabilistische Tests
- Deterministische Tests

Die bekannten Primzahltest-Verfahren können für große Zahlen viel schneller testen, ob die zu untersuchende Zahl prim ist, als die bekannten Faktorisierungsverfahren eine Zahl ähnlicher Größenordnung in ihre Primfaktoren zerlegen können.

Für den AKS-Test wurde die GMP-Bibliothek (GNU Multiple Precision Arithmetic Library) in CrypTool integriert.

Menü: „Einzelverfahren“ \ „RSA-Kryptosystem“ \ „Primzahltest“

Bemerkung: 2^255 - 1 = 7 * 31 * 103 * 151 * 2143 * 11119 * 106591 * 131071 * 949111 * 9520972806333758431 * 57024515776397755458386431 51

Anwendungsbeispiele (1)

Hybridverschlüsselung und Digitale Zertifikate

- Hybridverschlüsselung – **Kombination aus asymmetrischer und symmetrischer Verschlüsselung**
  1. Generierung eines zufälligen symmetrischen Sitzungs-Schlüssels (Session Key)
  2. Der Session Key wird – geschützt mit dem asymmetrischen Schlüssel – übertragen.
  3. Die Nachricht wird – geschützt mit dem Session Key – übertragen.
- Problem: **Man-in-the-middle-Angriffe: Gehört der öffentliche Schlüssel (Public Key) des Empfänger auch wirklich dem Empfänger?**
- Lösung: Digitale Zertifikate – **Eine zentrale Instanz (z.B. GlobalSign, Telesec, VeriSign, Deutsche Bank PKI), der alle Benutzer trauen, garantiert die Authentizität des Zertifikates und des darin enthaltenen öffentlichen Schlüssels (analog zu einem vom Staat ausgestellten Personalausweis).**
- Hybridverschlüsselung auf Basis von digitalen **Zertifikaten ist die Grundlage für sichere elektronische Kommunikation**
  - Internet Shopping und Online Banking
  - Sichere E-Mail

Anwendungsbeispiele (1)

Gesicherte Online-Verbindung mit SSL und Zertifikaten

D.h. die Verbindung ist (zumindest einseitig) authentisiert und der übertragene Inhalt wird stark verschlüsselt.

Anwendungsbeispiele (1)

Attribute / Felder von Zertifikaten

Grundlegende Attribute / Felder

- Aussteller (z.B. VeriSign)
- Antragsteller
- Gültigkeitszeitraum
- Seriennummer
- Zertifikatsart / Version (X.509v3)
- Signaturalgorithmus
- Öffentlicher Schlüssel (und Verfahren)

Öffentlicher Schlüssel

Anwendungsbeispiele (1)

Aufbau einer gesicherten SSL-Verbindung (Server Authentication)

Client **Server**

1. SSL Verbindungsaufbau →

← Sende Server-Zertifikat **2.**

3. Überprüfung des Server-Zertifikats (mit Hilfe der gespeicherten Root-Zertifikate)

4. Ermittle öffentlichen Schlüssel des Server-Zertifikat

5. Generiere einen zufälligen symmetrischen Schlüssel (Session Key)

6. Sende Session Key
(verschlüsselt mit öffentlichem Schlüssel des Servers) →

Empfange Session Key **7.**
(Entschlüsselung durch privaten Schlüssel des Servers)

SSL-gesichert (128 Bit)

↔ ***Verschlüsselte Kommunikation auf Basis des vereinbarten Session Keys***

Anwendungsbeispiele (1)

Aufbau einer gesicherten SSL-Verbindung (Server Authentication)

Allgemein

- Das Beispiel skizziert den typischen Aufbau einer SSL-Verbindung zur Übertragung von sensitiven Informationen (z.B. Internet-Shopping).
- Beim Aufbau der SSL-Verbindung authentisiert sich lediglich der Server durch ein digitales Zertifikat (die Authentisierung des Benutzer erfolgt in der Regel durch die Eingabe von Benutzername und Passwort nach dem Aufbau der SSL-Verbindung).
- SSL bietet auch die Möglichkeit einer zweiseitigen Authentisierung auf Basis digitaler Zertifikate.

Anmerkungen zur SSL-Verbindung

- ad (1): SSL Verbindungsaufbau – hierbei wird u.a. ausgehandelt welche Eigenschaften der Session Key besitzen soll (z.B. Bit-Länge) und welcher Algorithmus für die symmetrische Verschlüsselung verwendet werden soll (z.B. 3DES, AES).
- ad (2): Sofern Zwischenzertifikate notwendig sind (bei mehrstufigen Zertifikatshierarchien), werden diese ebenfalls übertragen.
- ad (3): In diesem Schritt werden die im Browser installierten Root-Zertifikate verwendet, um das empfangene Server-Zertifikat zu validieren.
- ad (5): Der Session Key basiert auf den unter (1) ausgehandelten Eigenschaften.

Anwendungsbeispiele (2)

Elektronische Signatur visualisiert

Elektronische Signatur

- Wird immer wichtiger durch
  - Gleichstellung mit manueller Unterschrift
  - Zunehmenden Einsatz in Wirtschaft, durch den Staat und privat
- Wer weiß, wie sie funktioniert?

Visualisierung in CrypTool

- Interaktives Datenflussdiagramm
- Ähnlich wie die Visualisierung der Hybridverschlüsselung

Menü: „Digitale Signaturen/PKI“ \
„Signaturdemo (Signaturerzeugung)“

Anwendungsbeispiele (2)

Elektronische Signatur visualisiert: a) Vorbereitung

1. Hashfunktion wählen

2. Schlüssel und Zertifikat bereitstellen (hier nicht abgebildet)

Anwendungsbeispiele (2)

Elektronische Signatur visualisiert: b) Kryptographie

3. Hashwert berechnen
4. Hashwert mit dem privaten Schlüssel verschlüsseln (unterschreiben)
5. Signatur generieren

Anwendungsbeispiele (2)

Elektronische Signatur visualisiert: c) Ergebnis

6. Das signierte Dokument kann nun gespeichert werden.

Die Reihenfolge der Schritte lässt sich variieren, soweit die Abhängigkeiten der Daten dies zulassen.

Anwendungsbeispiele (3)

Angriff auf zu kurzen RSA-Modul N

Aufgabe aus Song Y. Yan, *Number Theory for Computing*, Springer, 2000

- Öffentlicher Schlüssel
- RSA-Modul **N = 63978486879527143858831415041** (95 Bit, 29 Dezimalstellen)
- Öffentlicher Exponent **e = 17579**
- Verschlüsselter Text (Blocklänge = 8):

  C_1 = 45411667895024938209259253423,
  C_2 = 16597091621432020076311552201,
  C_3 = 46468979279750354732637631044,
  C_4 = 32870167545903741339819671379
- Der Text soll entschlüsselt werden.

Für die eigentliche Kryptoanalyse (das Finden des privaten Schlüssels) ist der Geheimtext nicht notwendig!

Lösung mit CrypTool **(ausführlich in den Szenarien der Online-Hilfe beschrieben)**

- Öffentliche Parameter in RSA-Kryptosystem (Menü „Einzelverfahren“) eintragen
- Funktion „RSA-Modul faktorisieren“ liefert die Primfaktoren p und q mit pq = N
- Daraus wird der geheime Schlüssel $d = e^{-1} \bmod (p-1)(q-1)$ abgeleitet
- Entschlüsseln des Textes mit Hilfe von d: $M_i = C_i^d \bmod N$

Angriff mit CrypTool ist für RSA-Module bis ca. 250 Bit praktikabel

Danach könnte man für jemand anderen elektronisch unterschreiben!

Anwendungsbeispiele (3)

Kurzer RSA-Modul: Öffentliche Parameter eingeben

Menü: „Einzelverfahren“ \ „RSA-Kryptosystem” \ „RSA-Demo…”

1. Öffentliche RSA-Parameter N und e eingeben

2. Faktorisieren

Anwendungsbeispiele (3)

Kurzer RSA-Modul: RSA-Modul faktorisieren

3. Faktorisierung ergibt p und q

Anwendungsbeispiele (3)

Kurzer RSA-Modul: Geheimen Schlüssel d bestimmen

RSA-Demo

RSA mit privatem und öffentlichem Schlüssel -- oder nur mit öffentlichem Schlüssel

Wählen Sie 2 Primzahlen p und q. Die Zahl N = pq ist der öffentliche RSA-Modul, und phi(N) = (p-1)(q-1) ist die Eulersche Phi-Funktion. Der öffentliche Schlüssel e ist teilerfremd zu phi(N). Daraus wird der geheime Schlüssel d = e^(-1) (mod phi(N)) berech

Zur Verschlüsselung von Daten oder zur Verifikation einer Signatur genügt es, dass Sie die öffentlichen RSA-Parameter angeben: den RSA-Modul N und den öffentlichen Schlüssel e.

Primzahleingabe

Primzahl p: 145295143558111

Primzahl q: 440334654777631

Primzahlen generieren...

RSA-Parameter

RSA-Modul N: 63978486879527143858831415041 (öffentlich)

phi(N) = (p-1)(q-1): 63978486879526558229033079300 (geheim)

Öffentlicher Schlüssel e: 17579

Geheimer Schlüssel d: 10663687727232084624328285019

Parameter aktualisieren

RSA-Verschlüsselung mit e / Entschlüsselung mit d

Eingabe als: Text, Zahlen

Optionen für Alphabet und Zahlensystem...

Wechsel in die Ansicht des Besitzers des geheimen Schlüssels.

4. p und q wurden automatisch eingetragen, und der geheime Schlüssel d wurde berechnet.

5. Optionen einstellen

Anwendungsbeispiele (3)

Kurzer RSA-Modul: Optionen einstellen

6. Alphabet wählen

7. Kodierung wählen

8. Blocklänge wählen

Anwendungsbeispiele (3)

Kurzer RSA-Modul: Text entschlüsseln

RSA-Parameter

RSA-Modul N: 63978486879527143858831415041 (öffentlich)

phi(N) = (p-1)(q-1): 63978486879526558229033079300 (geheim)

Öffentlicher Schlüssel e: 17579

Geheimer Schlüssel d: 10663687727232084624328285019

Parameter aktualisieren

RSA-Verschlüsselung mit e / Entschlüsselung mit d

Eingabe als ○ Text ◉ Zahlen

Optionen für Alphabet und Zahlensystem...

Chiffretext in Zahlendarstellung zur Basis 10 .

67645339468798682306085915 # 40145547044051327868820472463 # 30821430257177284710956359014

Entschlüsselung in den Klartext m[i] = c[i]^d (mod N)

00000000000000920190020080500 # 00000000000000514040015060020 # 00000000000000080500231518

Ausgabetext aus der Entschlüsselung (in Blöcken der Länge 8; das Symbol '#' dient nur als Trennzeichen).

ITS THE # END OF T # HE WORLD # AS WE K # NOW IT A # ND I FEE # L FINE

Klartext

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

Verschlüsseln | Entschlüsseln | Schließen

9. Geheimtext eingeben

10. Entschlüsseln

Anwendungsbeispiele (4)

Analyse der Verschlüsselung im PSION 5

Praktische Durchführung der Kryptoanalyse

Angriff auf die Verschlüsselungsoption der Textverarbeitungsapplikation im PSION 5 PDA

Gegeben: eine auf dem PSION verschlüsselte Datei

Voraussetzung

- verschlüsselter deutscher oder englischer Text
- je nach Verfahren und Schlüssellänge 100 Byte bis einige kB Text

Vorgehen

- Voranalyse
  - Entropie
  - gleitende Häufigkeit
  - Kompressionstest

  } *wahrscheinlich klassische Verschlüsselung*
- Autokorrelation
- automatische Analyse mit verschiedenen klassischen Verfahren durchprobieren

Anwendungsbeispiele (4)

PSION-PDA: Entropie bestimmen, Kompressionstest

Komprimierbarkeit: deutlicher Indikator für schwache Kryptographie (Größe wurde um 21% reduziert)

Die Entropie gibt keinen konkreten Hinweis auf ein bestimmtes Verschlüsselungs-Verfahren.

Anwendungsbeispiele (4)

PSION-PDA: Autokorrelation bestimmen

Ausgeprägtes Kamm-Muster: typisch für Vigenère, XOR und binäre Addition

* Diese verschlüsselte Datei wird mit CrypTool ausgeliefert (siehe CrypTool\examples\psion-enc.hex)

Anwendungsbeispiele (4)

PSION-PDA: Automatische Analyse

Automatische Analyse mit

- **Vigenère: kein Erfolg**
- **XOR: kein Erfolg**
- **Binärer Addition**
  - CrypTool ermittelt die Schlüssellänge mittels Autokorrelation: 32 Byte
  - Das erwartete häufigste Zeichen kann der Benutzer wählen: „e“ = 0x65 (ASCII-Code)
  - Analyse ermittelt den (unter der Verteilungsannahme) wahrscheinlichsten Schlüssel

Anwendungsbeispiele (4)

PSION-PDA: Ergebnis der automatischen Analyse

Ergebnis der automatischen Analyse unter der Annahme „binäre Addition“

- Ergebnis gut, aber nicht perfekt: 24 von 32 Schlüsselbytes richtig.
- Die Schlüssellänge 32 wurde korrekt bestimmt.

Automatische ADD-Analyse von <psion-enc.hex>, Schlüssel: <12 86 5B EF 14 98 87 2C 39 3E 43...

```
00000  65 72 67 AA 73 65 74 7A 20 28 55 53 74 47 29 06   ergªsetz (UStG).
00010  06 06 8A 72 73 74 65 72 65 41 62 B8 A8 68 6E AE   ...rstereAb¸¨hn®
00020  74 74 06 98 74 65 75 65 72 67 65 67 65 6E 73 74   tt..teuergegenst
00030  61 6E A9 20 75 6E 64 20 8C 65 6C B9 BA 6E 67 B8   an© und .el¹ºng¸
00040  62 65 72 AA 69 63 68 06 06 A7 20 31 2E 06 28 31   berªich..§ 1..(1
00050  29 20 89 65 72 20 55 6D B8 61 74 BF B8 74 65 BA   ) .er Um¸at¿¸teº
00060  65 72 20 BA 6E 74 65 72 6C 69 65 67 65 6E 20 64   er ºnterliegen d
00070  69 65 65 66 6F 6C 67 65 B3 64 65 B3 65 55 6D B8   ieefolge³de³eUm¸
00080  E4 74 7A AA 3A 06 31 2E 20 64 69 65 20 4C 69 65   ätzª:.1. die Lie
00090  66 65 B7 75 6E 67 65 6E 65 75 6E A9 65 73 6F B3   fe·ungeneun©eso³
000A0  73 74 69 AC 65 6E 20 4C 65 69 73 74 75 6E 67 65   sti¬en Leistunge
000B0  6E 2C 65 64 69 65 20 65 AE 6E 20 9A B3 74 65 B7   n,edie e®n .³te·
000C0  6E 65 68 B2 65 72 20 69 6D 20 49 6E 6C 61 6E 64   neh²er im Inland
000D0  20 67 AA 67 65 6E 20 45 B3 74 67 AA B1 74 20 AE    gªgen E³tgª±t ®
000E0  6D 20 52 A6 68 6D 65 6E 20 73 65 69 6E 65 73 20   m R¦hmen seines
000F0  55 6E B9 65 72 6E 65 68 B2 65 6E B8 65 61 75 B8   Un¹erneh²en¸eau¸
```

- Das eingegebene Passwort war nicht 32 Byte lang.
  ⇒ PSION Word leitet aus dem Passwort den eigentlichen Schlüssel ab.
- Nacharbeiten von Hand liefert den entschlüsselten Text (nicht abgebildet).

Anwendungsbeispiele (4)

PSION-PDA: Bestimmung der restlichen Schlüsselbytes

Schlüssel während der automatischen Analyse in die Zwischenablage kopieren

Im Hexdump der automatischen Analyse:

- Falsche Bytepositionen bestimmen, z.B. 0xAA an Position 3
- Korrespondierende korrekte Bytes erraten und notieren: „e“ = 0x65

Im Hexdump der verschlüsselten Ausgangsdatei:

- Ausgangsbytes an der ermittelten Bytepositionen bestimmen: 0x99
- Mit CALC.EXE korrekte Schlüsselbytes errechnen: 0x99 - 0x65 = 0x34

Schlüssel aus der Zwischenablage:

- Korrigieren 12865B**34**1498872C393E43741396A45670235E111E907AB7C0841A2E8068E50
- Verschlüsseltes Ausgangsdokument mittels binärer Addition entschlüsseln
- Nun sind die Bytepositionen 3, 3+32, 3+2*32, ... ok

Anwendungsbeispiele (5)

„Schwache“ DES-Schlüssel – Implementierung bestätigt die Angaben der Literatur [vgl. HAC]

CrypTool 1.4.30 - weak-DES.txt

Datei Bearbeiten Ansicht Ver-/Entschlüsseln Digitale Signaturen/PKI Einzelverfahren Analyse Optionen Fenster Hilfe

- Symmetrisch (klassisch)
- Symmetrisch (modern)
  - IDEA...
  - RC2...
  - RC4...
  - DES (ECB)...
  - DES (CBC)...
  - Triple DES (ECB)...
  - Triple DES (CBC)...
  - Rijndael (AES)... Shift + Strg + R
  - Weitere Algorithmen
  - AES (selbstextrahierend)...
- Asymmetrisch
- Hybrid

weak-DES.txt

DES weak key demo

Schlüsseleingabe: DES (ECB)

Geben Sie den Schlüssel mit Hexadezimalzeichen (0..9, A..F) ein.

Schlüssellänge: 64 Bit (56 Bit effektiv)

01 01 01 01 01 01 01 01

Verschlüsseln Entschlüsseln Abbrechen

2× verschlüsseln mit …
ergibt wieder den Klartext

weak-DES.txt

DES weak key demo

DES (ECB)-Verschlüsselung von <weak-DES.txt>, Schlüssel <01 01 01 01 01 01 01 01>

```
00000000  0B 76 4F 0A 1A 2E 70 A8 2D 3B A3 4B F0 76 6E 2B E0 2B A4 26 58 2B 63 31   .vO...p.-;.K.vn+.+.&X+c1
```

DES (ECB)-Verschlüsselung von <DES (ECB)-Verschlüsselung von <weak-DES.txt>, Schlüssel <01 01 01 01 ...>, Schlüssel <...>

```
00000000  44 45 53 20 77 65 61 6B 20 6B 65 79 20 64 65 6D 6F 00 00 00 00 00 00 00   DES weak key demo.....
```

Anwendungsbeispiele (6)

Auffinden von Schlüsselmaterial

Die Funktion „Gleitende Häufigkeit“ eignet sich zum Auffinden von Schlüsselmaterial und verschlüsselten Bereichen in Dateien.

Hintergrund

- Diese Daten sind „zufälliger“ als Text oder Programmcode.
- Sie sind als Peak in der „gleitenden Häufigkeit“ zu erkennen.
- Beispiel: der „NSA-Key“ in advapi32.dll (Windows NT)

Anwendungsbeispiele (6)

Vergleich der gleitenden Häufigkeit anderer Dateien

Anwendungsbeispiele (7)

Angriff auf digitale Signatur

Angriff:

Finde zwei Nachrichten mit dem gleichen Hashwert !

Menü: „Analyse“ \ „Hashverfahren“ \ „Angriff auf den Hashwert einer digitalen Signatur“

Anwendungsbeispiele (7)

Angriff auf digitale Signatur: Idee (1)

Angriff auf die digitale Signatur eines ASCII-Textes durch Suche nach Hashkollisionen

Idee

- ASCII-Text kann mittels **nicht-druckbarer** Zeichen modifiziert werden, ohne den lesbaren Inhalt zu verändern
- Modifiziere parallel zwei Texte, bis eine Hashkollision erreicht wird
- Ausnutzung des Geburtstagsparadoxons (Geburtstagsangriff)
- Generischer Angriff auf beliebige Hashfunktion
- In CrypTool implementiert im Rahmen der Bachelor-Arbeit *„Methoden und Werkzeuge für Angriffe auf die digitale Signatur“*, 2003.
  - Angriff ist gut parallelisierbar (nicht implementiert)

Konzepte

- Mappings
- Modifizierter Floyd-Algorithmus (konstanter Speicherbedarf)

Anwendungsbeispiele (7)

Angriff auf digitale Signatur: Idee (2)

1. **Modifikation:**
   Ausgehend von der Nachricht M werden N verschiedene Nachrichten $M_1, \ldots, M_N$ – „inhaltlich" gleich mit der Ausgangsnachricht – erzeugt.
2. **Suche:**
   Gesucht werden *modifizierte* Nachrichten M_i^H und M_j^S mit gleichem Hashwert.
3. **Angriff:**
   Die Signaturen zweier solcher Dokumente M_i^H und M_j^S sind identisch.

Für Hashwerte der Bitlänge n sagt das Geburtstagsparadoxon:

- Kollisionssuche zwischen M^H und $M_1^S, \ldots, M_N^S$: $N \approx 2^n$
- Kollisionssuche zwischen $M_1^H, \ldots, M_N^H$ und $M_1^S, \ldots, M_N^S$: $N \approx 2^{n/2}$

Erwartete Anzahl der zu erzeugenden Nachrichten, um eine Kollision zu erhalten.

Hashkollisionssuche (1)

Mapping durch Textmodifikation

- grün / rot: Pfade aus einem Baum in den Zyklus, die zu einer für den Angreifer nützlichen (grün) / nicht nutzbaren Kollision (rot) führen.
- quadratisch / rund: Hashwert hat gerade / ungerade Parität
- schwarz: alle Knoten im Zyklus

Hashkollisionssuche (2)

Floyd-Algorithmus: Treffen im Zyklus

Beispiel: Funktionsgraph mit 32 Knoten

Schritt 1: Finden des Treffpunktes im Zyklus:

- Zwei Folgen mit gleichem Start [16]: die eine Folge hat Schrittweite 1, die andere Schrittweite 2.
- Es gilt für alle Zykluslängen (aufgrund der Graphentheorie):
  - die Folgen enden immer in einem Zyklus.
  - beide Folgen treffen sich in einem Knoten im Zyklus (hier 0).

Hashkollisionssuche (3)

Eintritt in den Zyklus (Erweiterung von Floyd): Finde Eintrittspunkt

Schritt 2: Finden des Eintrittspunktes von Folge 1 in den Zyklus [25]:

- Folge 1 noch mal vom Startwert loslaufen lassen; dritte Folge mit Schrittweite 1 ab Treffpunkt im Zyklus (hier 0) loslaufen lassen.
- Es gilt: Die Folgen treffen sich im Eintrittspunkt (hier 25) der Folge
- Die Vorgänger (hier 17 und 2) liefern die Hashkollision.

Hashkollisionssuche (4)

Geburtstagsangriff auf die digitale Signatur

Auseinandersetzung mit dem Floyd-Algorithmus

- Visuelle & interaktive Darstellung des Floyd- Algorithmus („Wanderung im Mapping“ in einen Zyklus hinein).*
- Adaption des Floyd- Algorithmus für den Signaturangriff.

* Der Floyd-Algorithmus ist implementiert. Die Visualisierung von Floyd ist noch nicht in CrypTool integriert.

Anwendungsbeispiele (7)

Angriff auf digitale Signatur

Ein Beispiel für ein **„gutartiges“ Mapping** (fast alle Knoten darin sind grün gefärbt). In diesem Graphen gehören die meisten Knoten zu einem großen Baum, der in den Zyklus mit einem geraden Hashwert gelangt und wo der Eintrittspunkt-Vorgänger im Zyklus ungerade ist.
D.h. der Angreifer findet für fast jeden zufälligen Startpunkt eine brauchbare Kollision.

Anwendungsbeispiele (7)

Angriff auf digitale Signatur: Durchführung

Menü: „Analyse“ \ „Hashverfahren“ \
„Angriff auf den Hashwert einer digitalen Signatur“

Anwendungsbeispiele (7)

Angriff auf digitale Signatur: Ergebnisse

Die ersten 32 Bit des Hashwertes sind gleich.

Praktische Resultate

- 72 Bit *Teilkollisionen* (Übereinstimmung der ersten 72 Bit-Stellen der Hashwerte) konnten im Zeitraum von wenigen Tagen auf einem einzigen PC gefunden werden.
- Signaturverfahren mit Hashverfahren bis zu 128 Bit Länge sind heute mit massiv parallelen Verfahren angreifbar!
- Es sollten Hashwerte mit mindestens 160 Bit verwendet werden.

Zusätzlich zur interaktiven Bedienung

Automatisierte Offline-Funktion in CrypTool: Durchspielen und Loggen der Ergebnisse für ganze Sets von Parameterkonfigurationen. Möglich durch entsprechenden Aufruf von CrypTool über die Eingabeaufforderung.

Anwendungsbeispiele (8)

Authentifizierung in einer Client-Server-Umgebung

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

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

Anwendungsbeispiele (9)

Demonstration eines Seitenkanalangriffes (auf ein Hybridverschlüsselungsprotokoll)

Menü: „Analyse“ \ „Asymmetrische Verfahren“ \ „Seitenkanalangriff auf Textbook-RSA“

Anwendungsbeispiele (9)

Idee zu diesem Seitenkanalangriff

Ulrich Kühn: *„Side-channel attacks on textbook RSA and ElGamal encryption“*, 2003

Voraussetzungen [CCA-Angriff (Chosen-ciphertext attack) auf Dechiffrierorakel]

- RSA-Verschlüsselung: $C = M^e \pmod N$ und Entschlüsselung: $M = C^d \bmod N$.
- 128-Bit Sessionkeys (in M) werden „Textbuch-RSA“ verschlüsselt (Null-Padding).
- Der Server kennt den geheimen Schlüssel d und
  - benutzt nach der Entschlüsselung nur die 128 niederwertigsten Bit (keine Überprüfung der Null-Padding-Bit) (d.h. er erkennt nicht, wenn dort was anderes als Nullen stehen).
  - liefert eine Fehlermeldung, wenn bei der Entschlüsselung ein „falscher“ Session Key bestimmt wird (entschlüsselter Text kann nicht vom Server interpretiert werden). Im anderen Fall kommt keine Meldung.

Angriffsidee: Approximation von Z auf 129 Bitstellen aus der Gleichung $N = M * Z$ per $M = \lfloor\lfloor N/Z \rfloor\rfloor$

$C = M^e \pmod N$

Für Z werden die Bitstellen sukzessive ermittelt: Pro Schritt erhält man 1 Bit mehr. Der Angreifer modifiziert C nach C’ (siehe unten). Abhängig davon, ob es beim Server (Empfänger) zu einem Bit-Überlauf bei der Berechnung von M’ kommt, schickt er eine Fehlermeldung oder nicht. Basierend auf dieser Information erhält der Angreifer ein Bit für Z.

$C' = M'^e = M^e \cdot (1+Z \cdot 2^{128})^e \pmod N$

Anwendungsbeispiele (10)

Mathematik: Angriffe auf RSA per Gitterreduktion

- Veranschaulicht, wie die Parameter des RSA-Verfahrens beschaffen sein müssen, damit sie den aktuellen, auf Gitterreduktion beruhenden Angriffen aus der Literatur standhalten.
- **Drei Varianten**, die *nicht* standhalten
  1. Der geheime Exponent d ist im Verhältnis zu N zu klein.
  2. Einer der Faktoren von N ist teilweise bekannt.
  3. Ein Teil des Klartextes ist bekannt.
- Diese Annahmen sind realistisch.

Menü: „Analyse“ \
„Asymmetrische Verfahren“ \
„Gitterbasierte Angriffe auf RSA“ \ …

Anwendungsbeispiele (11)

Zufallsanalyse mit 3-D-Visualisierung

3-D Visualisierung zur Analyse von Zufallszahlen

Beispiel 1

- Öffnen einer beliebigen Datei (z.B. Bericht in Word oder PowerPoint-Präsentation)
- Es empfiehlt sich eine zumindest 100 KB große Datei zu wählen
- 3-D-Analyse über das Menü: „Analyse“ \ „Zufallsanalyse“ \ „3-D-Visualisierung“
- Ergebnis: **Strukturen sind offensichtlich erkennbar**

Beispiel 2

- Generierung von Zufallszahlen: „Einzelverfahren“ \ „Tools“ \ „Zufallsdaten erzeugen“
- Hierbei sollte man zumindest 100.000 Bytes an Zufallsdaten erzeugen
- 3-D-Analyse über das Menü: „Analyse“ \ „Zufallsanalyse“ \ „3-D Visualisierung“
- Ergebnis: **Gleichverteilung (keine Strukturen erkennbar)**

Anwendungsbeispiele (12)

Secret Sharing mittels CRT – Implementierung des Chinesischen Restsatzverfahrens

Secret Sharing Beispiel (1)

- **Problemstellung**
  - 5 Personen erhalten jeweils einen Schlüssel
  - Um Zugriff zu erlangen, müssen mindestens 3 der 5 Personen anwesend sein
- **„Optionen“** ermöglicht weitere Details des Verfahrens einzustellen.

- **„Rechenweg“** zeigt die Schritte zur Generierung der Schlüssel.

Menü: „Einzelverfahren“ \
„Anwendungen des Chinesischen Restsatzverfahrens“ \
„Secret Sharing mittels CRT“

Anwendungsbeispiele (12)

Secret Sharing mittels Schwellenwertschema von Shamir

Secret Sharing Beispiel (2)

- **Problemstellung**
  - Ein geheimer Wert soll unter n Personen aufgeteilt werden.
  - t von n Personen sind notwendig um den geheimen Wert wiederherzustellen.
  - (t, n) Schwellenwertschema
- **Menü:** „Einzelverfahren“ \ „Secret Sharing Demo (nach Shamir)“
  1. Angabe des Geheimnisses K, sowie Anzahl der Teilnehmer n und Schwellenwert t
  2. Polynom generieren
  3. Parameter übernehmen
- Mittels **„Rekonstruktion“** kann das eingegebene Geheimnis wiederhergestellt werden.

Anwendungsbeispiele (13)

Anwendung des CRT in der Astronomie (Lösung linearer Kongruenzsysteme)

Problemstellung aus der Astronomie

- Wie lange dauert es, bis sich eine gegebene Anzahl Planeten (mit unterschiedlichen Umlaufgeschwindigkeiten) auf einem Bahnradiusvektor s treffen.
- Ergebnis ist ein System simultaner Kongruenzen, das sich mit Hilfe des Chinesischen Restsatzes (CRT) lösen lässt.
- In dieser Demo können bis zu 9 Kongruenzen aufgestellt und mittels CRT gelöst werden.

Menü: „Einzelverfahren“ \ „Anwendungen des Chinesischen Restsatzverfahrens“ \ „Astronomie und Planetenbewegung“

Anwendungsbeispiele (14)

Visualisierung von symmetrischen Verschlüsselungsverfahren mit ANIMAL (1)

Animierte Darstellung verschiedener symmetrischer Verfahren

- Caesar
- Vigenère
- Nihilist
- DES

CrypTool

- Menü: „Einzelverfahren“ \ „Visualisierung von Algorithmen“ \ …
- Steuerung der Animation über integrierte Steuerelemente

Animationsgeschwindigkeit

Skalierung der Darstellung

Steuerung der Animationsschritte (Vor, Zurück, Pause, etc.)

Direkte Anwahl eines Animationsschrittes

Anwendungsbeispiele (14)

Visualisierung von symmetrischen Verschlüsselungsverfahren mit ANIMAL (2)

Visualisierung der DES-Verschlüsselung

Nach der Permutation des Eingabeblocks mit Hilfe des Initialisierungsvektors IV wird der Schlüssel K mit Hilfe von PC1 und PC2 permutiert.

Die Kernfunktion *f* des DES verknüpft die rechte Blockhälfte R_{i-1} mit dem Teilschlüssel K_i.

Anwendungsbeispiele (15)

Visualisierungen von AES (Rijndael-Chiffre) – mit Flash

Rijndael-Animation (die Rijndael-Chiffre war Gewinner der AES-Ausschreibung)

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

Rijndael-Inspector

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

Menü: „Einzelverfahren“ \ „Visualisierung von Algorithmen“ \ „AES“ \ „Rijndael-Animation“ bzw. „Rijndael-Inspector“

Anwendungsbeispiele (15)

Flussvisualisierungen von AES (Rijndael-Chiffre) – mit Java

Rijndael-Flussvisualisierung

- Visualisierung der Datenveränderung je Runde durch Farbverläufe

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

Anwendungsbeispiele (16)

Visualisierung der Enigma-Verschlüsselung

Rotorstellung verändern

Auswahl der Rotoren

Setzen der Stecker

Anzeige der Einstellungen

Eingabe des zu Klartext

Ausgabe des verschlüsselten Textes

Zufälliges Setzen und Zurücksetzen der Enigma

Klicken für weitere HTML-Hilfe

Anwendungsbeispiele (17)

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

Menü: „Einzelverfahren“ \ „Protokolle“ \ „Sichere E-Mail mit S/MIME…“

Anwendungsbeispiele (18)

Erzeugung eines Keyed-Hash Message Authentication Code (HMAC)

Keyed-Hash Message Authentication Code (HMAC)

- Gewährleistet
  - Integritätsschutz der Nachricht
  - Authentizität der Nachricht
- Basis: Ein gemeinsamer Schlüssel für Sender und Empfänger
- Alternativ: Digitale Signatur

Berechnung eines HMAC in CrypTool

1. Auswahl der Hashfunktion
2. Auswahl der HMAC-Variante
3. Angabe eines Schlüssels (je nach HMAC-Variante auch zwei Schlüssel)
4. Erzeugung des HMAC (automatisch)

Menü: „Einzelverfahren“ \ „Hashverfahren“ \ „Generieren von HMACs“

Anwendungsbeispiele (19)

Hash-Demo

Sensitivität von Hashfunktionen bei Änderungen des Originaltextes

1. Auswahl der Hashfunktion
2. Zusätzliches Einfügen von Zeichen im Text

Beispiel

Die Eingabe eines zusätzliches Leerzeichens hinter „CrypTool“ in der Originaldatei bewirkt eine 45,6%-ige Änderung der Bits des resultierenden Hashwertes.

Eine gute Hashfunktion sollte auf jede noch so kleine Änderung der Originaldatei möglichst sensitiv reagieren – *„Avalanche effect“* (kleine Änderung, große Wirkung).

Menü: „Einzelverfahren“ \ „Hashverfahren“ \ „Hash-Demo“

Anwendungsbeispiele (20)

Lernprogramm zur Zahlentheorie und zur asymmetrischen Verschlüsselung

- **Zahlentheorie**
  Tutorial plus graphische Elemente und Tools zum Ausprobieren
- **Themen**
  1. Ganze Zahlen
  2. Restklassen
  3. Primzahlerzeugung
  4. Asymmetrische Verschlüsselung
  5. Faktorisierung
  6. Diskrete Logarithmen

Menü: „Einzelverfahren“ \ „Zahlentheorie interaktiv“ \ „Lernprogramm zur Zahlentheorie“

Anwendungsbeispiele (21)

Punktaddition auf elliptischen Kurven

- Visualisierung der Punktaddition auf elliptischen Kurven
- Grundlage der Elliptischen Kurven Kryptographie (ECC)

Beispiel 1

- Punkt P auf der Kurve markieren
- Punkt Q auf der Kurve markieren
- Schaltfläche „P+Q“: Die Gerade durch P und Q schneidet die Kurve im Punkt -R.
- Spiegelung an der X-Achse ergibt R

Beispiel 2

- Punkt P auf der Kurve markieren
- Schaltfläche „2*P“: Die Tangente an P schneidet die Kurve in -R.
- Spiegelung an der X-Achse ergibt R

Menü: „Einzelverfahren“ \ „Zahlentheorie interaktiv“ \ „Punktaddition auf Elliptischen Kurven“

Anwendungsbeispiele (22)

Passwort-Qualitätsmesser (PQM) und Passwort-Entropie (1)

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“

Anwendungsbeispiele (22)

Passwort-Qualitätsmesser (PQM) und Passwort-Entropie (2)

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 Angreiferperspektive

- Angriff auf ein Passwort (sofern beliebig viele Versuche zugelassen sind):

1. Klassischer **Wörterbuchangriff**
2. Wörterbuchangriff mit **weiteren Varianten** (z.B. 4-stellige Zahlen: 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 8 Zeichen lang ist und Zahlen sowie Sonderzeichen beinhaltet.

Anwendungsbeispiele (23)

Brute-Force-Analyse (1)

Brute-Force-Analyse

Optimierte Brute-Force-Analyse unter der Annahme, dass ein Teil des Schlüssels bekannt ist.

Beispiel: Analyse mit DES (ECB)

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

Schlüssel (Hex)	Verschlüsselter Text (Hex)
68ac78dd40bbefd*	66b9354452d29eb5
0123456789ab****	1f0dd05d8ed51583
98765432106*****	bcf9ebd1979ead6a
0000000000******	8cf42d40e004a1d4
000000000000****	0ed33fed7f46c585
abacadaba*******	d6d8641bc4fb2478
dddddddddd******	a2e66d852e175f5c

Anwendungsbeispiele (23)

Brute-Force-Analyse (2)

1. Eingabe des verschlüsselten Textes
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 mögliche Entschlüsselung. Allerdings hat bei diesem Beispiel aufgrund des kurzen Textes der richtige Kandidat nicht die kleinste Entropie.

Menü: „Ansicht“ \ „Als HexDump anzeigen“

Menü: „Analyse“ \ „Symmetrische Verfahren (modern)“ \ „DES (ECB)“

Anwendungsbeispiele (24)

Skytale / Gartenzaun

Skytale und Gartenzaun

- Transpositionsverfahren
  (Vertauschen der Reihenfolge der Zeichen im Klartext)

Auswahl des Schlüssels

- Anzahl der Kanten (Skytale)
- Anzahl der Zeilen (Gartenzaun)
- Offset

Textoptionen

- Übergeordnete Optionen (Menü: „Optionen“ \ „Textoptionen“)
- Formatierungsoptionen
- Verarbeitung von Groß-/Kleinbuchstaben
- Alphabet für die Verarbeitung des Textes
  (welche Zeichen sollen ver-/entschlüsselt werden)

Menü: „Ver-/Entschlüsseln“ \ „Symmetrisch (klassisch)“ \ „Skytale/Gartenzaun …“

Anwendungsbeispiele (25)

Hill-Verschlüsselung / Hill-Analyse (1)

Hill-Verschlüsselung

- Polygraphische Substitutionsverschlüsselung
- Basiert auf linearer Algebra

Schlüssel

- Alphabetzeichen (siehe Textoptionen) oder Zahlenwerte
- Eingabe oder zufällige Schlüssel erzeugen
- Wahl der Multiplikationsart
- Dimension der Matrix
- Hill-Optionen

Menü:
„Ver-/Entschlüsseln“ \
„Symmetrisch (klassisch)“ \
„Hill …“

Anwendungsbeispiele (25)

Hill-Verschlüsselung / Hill-Analyse (2)

Hill-Verschlüsselung

- Beispiel-Text mit dem Schlüssel `LVMH`

Hill-Analyse (Known Plaintext)

1. Langer Klar-/Geheimtext

- Auswahl des Klartext (startbeispiel-de.txt)
- Auswahl des Geheimtext (Hill-Verschlüsselung von <startbeispiel-de.txt>)
- „Schlüssel suchen“

2. Verkürzter Klar-/Geheimtext

- Alles löschen im Klartext bis auf den Anfang („CrypTool“)
- Reduzieren des Geheimtext auf „PnhdJovl“
- „Schlüssel suchen“ findet korrekten Schlüssel!

Welche Textlänge im Klar-/Geheimtext-Paar ist notwendig, um den richtigen Schlüssel zu finden?

Menü: „Analyse“ \ „Symmetrische Verschlüsselung (klassisch)“ \ „Known Plaintext“ \ „Hill...“

Anwendungsbeispiele (26)

CrypTool Online-Hilfe (1)

Menü: „Hilfe“ \ „Startseite“

Hilfe zu CrypTool 1.4.30

Einblenden Zurück Vorwärts Abbrechen Aktualisieren Startseite Drucken Optionen

Verschlüsselung

Mit Hilfe eines Verschlüsselungsschlüssels wird ein Dokument verschlüsselt und ist dann nicht
das Dokument wieder lesen zu können, muss es entschlüsselt werden.

In CrypTool sind verschiedene **Verschlüsselungsverfahren** vorhanden; sie sind über das Men
/Entschlüsseln erreichbar.

Verschlüsselungsverfahren

Ein Verschlüsselungsverfahren wird benötigt, um vertrauliche Informationen über unsichere Infor
zum Beispiel ein Netzwerk, zu übertragen. Die Information wird vor der Übertragung vom Sende
und nach der Übertragung vom Empfänger entschlüsselt.

Man spricht von einem symmetrischen Verschlüsselungsverfahren, wenn der Schlüssel des Se
des Empfängers identisch sind. Verschlüsselungsverfahren mit unterschiedlichen Schlüsseln v
Empfänger nennt man asymmetrisch.

Bei modernen symmetrischen Verschlüsselungsverfahren unterscheidet man zwischen **Blockc**
Stromchiffren:

- Blockchiffren verschlüsseln Blöcke fester Länge (z. B. 64 oder 128 bit).
  In CrypTool verfügbare Blockchiffren sind IDEA, RC2, DES (ECB), DES (CBC), Triple DE
  Triple DES (CBC), Rijndael (AES), MARS, RC6, Serpent, Twofish, DESX, DESL und DES
- Stromchiffren verschlüsseln die Nachricht bitweise.
  Aus dieser Kategorie ist RC4 verfügbar.

Eine Übersicht über alle Verschlüsselungsverfahren in CrypTool befindet sich auf der Hilfeseite z
/Entschlüsseln.

Weitere Informationen zu Verschlüsselungsverfahren befinden sich im Skript, z.B. im Kapitel
"Verschlüsselungsverfahren".

Hilfe zu CrypTool 1.4.30

Einblenden Zurück Vorwärts Abbrechen Aktualisieren Startseite Drucken Optionen

CBC-Modus

CBC steht für **C**ipher **B**lock **C**haining.

Bei diesem Modus fließt das Ergebnis der Verschlüsselung früherer Blöcke in die Verschlüsselung des aktuellen Blockes mit ein. Jeder Block des verschlüsselten Textes hängt also nicht nur vom zugehörigen Klartextblock, sondern auch von allen vorherigen Klartextblöcken ab. Für den ersten Klartextblock wird zusätzlich ein Initialisierungsblock benötigt.

Die Verschlüsselung läuft wie folgt:

1. Der erste Klartextblock wird verschlüsselt.
2. Der resultierende Block verschlüsselten Textes wird mit dem nächsten Klartextblock mittels exklusivem Oder verknüpft und erst anschließend verschlüsselt.
3. Punkt 2 wird so lange fortgesetzt, bis das Ende des Klartextes erreicht ist.

Die Entschlüsselung läuft analog ab:

1. Der erste Block des verschlüsselten Textes wird entschlüsselt.
2. Der nächste Block des verschlüsselten Textes wird zunächst entschlüsselt anschließend mit dem vorhergehenden Block des verschlüsselten Textes mittels exklusivem Oder verknüpft.
3. Punkt 2 wird so lange fortgesetzt, bis das Ende des verschlüsselten Textes erreicht ist.

Dieses Verfahren hat allerdings immer noch den Nachteil, dass zwei Nachrichten bis zum ersten Unterschied den gleichen verschlüsselten Text ergeben. Insbesondere liefern also identische Nachrichten den gleichen verschlüsselten Text. Um das zu verhindern, verwendet man sogenannte Initialisierungsvektoren. Der Initialisierungsvektor ist ein Zufallswert, der unverschlüsselt mit der Nachricht übertragen werden kann. Vor der Verschlüsselung (bzw. Entschlüsselung) des ersten Blockes wird dieser Block mit dem Initialisierungsvektor mittels exklusivem Oder verknüpft. Anschließend läuft das Verfahren so ab wie oben beschrieben.

Im ECB-Modus wird jeder Block unabhängig von den anderen Blöcken verschlüsselt.

Anwendungsbeispiele (26)

CrypTool Online-Hilfe (2)

Hilfe zu CrypTool

Ausblenden | Zurück | Vorwärts | Abbrechen | Aktualisieren | Startseite | Drucken | Optionen

Inhalt | Index | Suchen

Zu suchendes Schlüsselwort:

gitter

- Gitterreduktion
- Gleichverteilung
- Gleitende Häufigkeit
- GMP
- GnuPG
- GPL/GNU
- Grafikoptionen
- Haftungsausschlussklausel
- Hashfunktion
- Hashwert einer Datei
- Hexadezimale Ein-/Ausgaben
- Hilfe
- Hill-Verschlüsselungsverfahren
- Hin- und Rücktransformation
- Histogramm
- Historie
- Homophone Substitution
- Hybrid
- Hybridentschlüsselung
- Hybridverschlüsselung
- Hypothese H0
- IDEA-Verschlüsselungsverfahren
- Importieren (Zertifikat) / PKCS #12
- Initialisierungsdatei
- Interoperabilität
- Inverse Permutation
- involutorisch
- ISIS / MTT
- Kartenspiel
- Kaskade
- Kaskadierende Chiffre
- Klartext
- Known-Plaintext-Angriff
- Komprimieren
- Konfigurationsdatei
- Kongruenzgenerator
- Kontakt
- Korrelation
- Kryptoanalyse
- Krypto-Bibliotheken
- Kryptographie
- Kryptologie

Anzeigen

Menü Gitterbasierte Angriffe auf RSA (Menü Einzelverfahren \ RSA-Kryptosystem)

Das Menü **Gitterbasierte Angriffe auf RSA** enthält folgende Befehle:

Faktorisieren mit teilweise bekanntem p	Angriff auf RSA mit Gitterreduktionsverfahren, sofern ein Teil eines der beiden Primfaktoren von N bekannt ist.
Angriff auf stereotype Nachrichten	Angriff auf RSA mit Gitterreduktionsverfahren, sofern ein Teil des ursprünglichen Klartextes einer abgefangenen Nachricht bekannt und e klein ist.
Angriff auf kleine geheime Schlüssel	Angriff auf RSA mit Gitterreduktionsverfahren, sofern d im Verhältnis zu N zu klein gewählt ist.

Allen hier vorgestellten gitterbasierten Angriffen liegt ein ähnlicher Ansatz zu Grunde: Zunächst wird das Problem, RSA zu brechen in die Suche nach Nullstellen eines Polynoms modulo einer ganzen Zahl (meist N) umgeformt. Eine solche Nullstelle zu finden ist ein schwieriges Problem.

Um die Nullstellen zu berechnen werden zu dem ursprünglichen Polynom noch eine Reihe weiterer Polynome aufgestellt, von denen bekannt ist, dass sie die selbe Nullstelle besitzen. Aus den Koeffizienten dieser Polynome wird eine Gitterbasis aufgestellt. Diese wird mit dann z.B. mit dem LLL-Algorithmus reduziert, um einen kurzen Vektor zu finden.

Aus dem gefundenen kurzen Vektor wird wieder ein Polynom aufgestellt. Man kann zeigen, dass wenn der gefundene Vektor kurz genug ist, das aufgestellte Polynom die gesuchte Nullstelle nicht nur modulo N sondern über allen Zahlen besitzt.

Beispiel:

Das Polynom $q_1(x) = 3x+1$ hat eine Nullstelle x_0 **modulo** 7. Es sei bekannt, dass das Polynom $q_2(x) = 4x-1$ die gleiche Nullstelle x_0 **modulo** 7 besitzt. Aus den Polynomen werden nun die Vektoren $b_1=[3\ 1]$ und $b_2=[4\ -1]$ aufgestellt. Alle ganzzahligen Linearkombinationen dieser Vektoren stellen Punkte in einem Gitter dar. Die Abbildung links zeigt einen Ausschnitt dieses Gitters. Jeder Gitterpunkt kann wiederum als Polynom interpretiert werden, das ebenfalls die gesuchte Nullstelle besitzt. Ein kurzer Vektor des Gitters ist $b_3=[1\ -2]$ aus dem das Polynom $h(x) = x-2$ gebildet wird. Dieses Polynom hat in $x_0=2$ eine Nullstelle sowohl über den ganzen Zahlen als auch über den ganzen Zahlen **modulo** 7. Man erkennt, dass $x_0=2$ auch Nullstelle der Polynome $q_1(x)$ und $q_2(x)$ **modulo** 7 ist. ($3x_0+1=7$, 7 **modulo** 7 = 0)

Bemerkung:

Anwendungsbeispiele (26)

CrypTool Online-Hilfe (3)

Hilfe zu CrypTool 1.4.20 (DE)

Ausblenden | Zurück | Vorwärts | Abbrechen | Aktualisieren | Startseite | Drucken | Optionen

Inhalt | Index | Suchen

Zu suchendes Schlüsselwort:

base

- Base64-Codierung
- BC
- Bearbeiten
- Beenden
- Beispiele/Szenarien
- Binäres exklusives Oder
- Bitlänge
- Blöcke
- Brute-Force-Angriff
- Bücher
- Byteweise Addition
- Caesar-Verschlüsselungsverfahren
- CBC-Modus
- Challenge
- Challenge-Response-Demo
- CHI^2-Verteilung
- Chiffrat
- Chiffretext
- Chinesischer Restsatz (CRT)
- Chosen-Plaintext-Angriff
- Ciphertext
- Ciphertext-Only-Angriff
- Codierungen
- Copyright
- CrypTool
- cv cryptovision
- Datei
- Default-Settings
- Demonstrationen/Szenarien
- DESL-Verschlüsselungsverfahren
- DES-Verschlüsselungsverfahren
- Dialog (Übersicht aller Dialoge)
- Dialog der Schwestern
- Diffie-Hellman-Schlüsselaustauschv
- Digitale Signaturen/PKI
- Digitales Zertifikat
- Digramm
- DIN
- Dokument
- Dokument-Eigenschaften
- Drucken
- DSA
- ECB-Modus
- ECSP-DSA-Signatur
- ECSP-NR-Signatur
- Einordnung/Überblick/Übersicht

Anzeigen

Vergleich von Base64- und UU-Codierung

Die Codierungen bei Base64 und UUencode sind sehr ähnlich, was nachstehendes Schaubild zeigt:

Durch die ähnliche Art der Codierung weisen die Codierungen auch gemeinsame Vor- und Nachteile auf:

Vorteile	Nachteile

• Beliebige Binärdaten können mit einem 6-bit Zeichensatz dargestellt werden:
  ○ Keine Probleme mit 7-bit Zeichensatzbeschränkungen.
  ○ Keine Probleme mit Zeilenlängenbeschränkungen oder besonderen Steuerzeichen.
• Vergrößerung nur um ca. 33 % (anstelle von z.B. Kodierung in Hexadezimalwerte = Vergrößerung um 100 %).
	• Keine Unterstützung für die Aufteilung von großen Dateien.
	• Vergrößerung der Dateien um ca. 33 % (im Vergleich zur Originaldatei).
	Nur UUencode:
	• Keine EBCDIC Unterstützung.
	• Keine definierten Standards.

Inhalt

I. CrypTool und Kryptologie – Überblick

II. Was bietet CrypTool?

III. Ausgewählte Beispiele

IV. Projekt / Ausblick / Kontakt

Weiterentwicklung (1)

Beispiele, was nach CrypTool Release 1.4.30 geplant ist (vgl. Readme-Datei)

CT1	FIPS-Test , der Pakete von Längen ungleich 2500 Bytes untersucht; …
JCT	Tri-partite Schlüsselvereinbarung
JCT	Visualisierung der Interoperabilität von S/MIME- und OpenPGP-Formaten
JCT	Entropie-Untersuchungen
JCT	Fleißner-Schablone, Autokey-Vigenère, interaktive Kryptoanalyse klassischer Verfahren
JCT	Analyse von Transpositionschiffren mit dem ACO-Algorithmus
JCT	Visualisierung von Zero-Knowledge-Beweisen
JCT	Visualisierung eines Quantum-Schlüsselaustausch-Protokolls (BB84)
JCT	Action-History mit der Möglichkeit, Produktchiffren (Kaskaden) zu bauen und abzuspielen
CT2	Umfangreiche Visualisierungen zum Thema Primzahlen
CT2	Verschlüsselung und automatisierte Kryptoanalyse der Enigma, und evtl. der Sigaba
CT2	Cube-Attack (I. Dinur und A. Shamir: „Cube Attacks on Tweakable Black Box Polynomials“, 2008)
CT2	Demo von Bleichenbachers Angriff auf RSA-Signaturen
CT2	Demo virtueller Kreditkartennummern als Ansatz gegen Kreditkartenmissbrauch
CT2	WEP-Verschlüsselung und WEP-Analyse
CT2	Massenmustersuche
CT2	Verteilte Kryptoanalyse
CT2	Demo SOA-Security (SOAP-Nachrichten nach WS-Security zwischen den Beteiligten)
CT2	Framework zum Bau und zur Analyse von LFSR-Stromchiffren
CT2	Grafik-Design-orientierter Modus für Einsteiger plus Expertenmodus
CT2/JCT	Erstellung einer Kommandozeilenversion für Batch-Steuerung
CT2/JCT	Moderne Pure-Plugin-Architektur mit Nachladen von Plugins
Alle	Weitere Parametrisierung / Flexibilisierung der vorhandenen Verfahren
Ideen	Visualisierung des SSL-Protokolls // Demo zur Visuellen Kryptographie

CT1 = CrypTool 1.x

Neue Versionen:

CT2 = CrypTool 2.0

JCT = JCrypTool

(beide auf den nächsten Folien eingeführt)

Weiterentwicklung (2)

In Arbeit: die zwei Nachfolger-Versionen von CT v1 (vgl. Readme-Datei)

1. JCT: Portierung und Neudesign von CrypTool in Java / SWT / Eclipse 3.6 / RCP
   - siehe: http://jcryptool.sourceforge.net
   - Release Candidate 2 ist verfügbar seit Mai 2010.
2. CT2: Portierung und Neudesign von CrypTool mit C# / WPF / VS2010 / .NET 4.0
   - Direkter Nachfolger des aktuellen Releases: erlaubt visuelle Programmierung, …
   - Download von: http://cryptool2.vs.uni-due.de/index.php?page=14&lm=1&ql=4
   - Beta 3 ist verfügbar seit Juli 2010 (seit Juli 2008 werden Betas permanent geupdated).

CrypTool 2 (CT2)

JCrypTool (JCT)

Weiterentwicklung (3)

CrypTool 2 (CT2)

JCrypTool (JCT)

CrypTool als Framework für eigene Arbeiten

Angebot

- 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- und Doktorarbeiten „verschwindet“ nicht, sondern wird weitergepflegt.

Aktuelle Entwicklungsumgebung für *CT1*: **Microsoft Visual Studio C++ , Perl, Subversion Source-Code-Management**

- CrypTool 1.4.30: Visual C++ .NET (= VC++ 9.0)(= Visual Studio 2008 Standard)
- Beschreibung für Entwickler: siehe readme-source.txt
- Download: Sourcen und Binaries der Release-Versionen
  Interessierte und Entwickler erhalten auch die Sourcen der aktuellen Betas.

Entwicklungsumgebungen von *CT2* und *JCT*

- CT2 – C#-Version: .NET mit Visual Studio 2010 Express Edition (kostenlos) und WPF
- JCT – Java-Version: mit Eclipse 3.6, RCP, SWT (kostenlos)

CrypTool – Bitte um Mitwirkung

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 und
- In den neuen Projekten (bevorzugt):
  - C#-Projekt: „CrypTool 2.0“ = CT2
  - Java-Projekt: „JCrypTool“ = JCT
- 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: See the list http://cryptool2.vs.uni-due.de/, Volunteer, Open Tasks
  - JCT: See the wiki http://sourceforge.net/apps/mediawiki/jcryptool/index.php?title=CurrentDevelopment
- Signifikante Beiträge können namentlich erwähnt werden (in der Hilfe, Readme, About-Dialog und auf der Webseite).
- Derzeit wird das gesamte CT v1-Programmpaket etwa 6.000 mal pro Monat von der CrypTool-Webseite herunter geladen (davon etwas mehr als 50 % die englische Version).
  Die Betas der zwei Nachfolger-Versionen werden schon über 1.000 mal pro Monat heruntergeladen.

CrypTool – Fazit

DAS E-Learning-Programm für Kryptologie

Seit über 10 Jahren ein erfolgreiches Open-Source-Projekt

Mehr als 300.000 Downloads

Weltweiter Einsatz in Schulen und Universitäten sowie Firmen und Behörden

Umfangreiche Online-Hilfe und Dokumentation

Frei verfügbar

Mehrsprachig

Kontaktadresse

Prof. Bernhard Esslinger

Universität Siegen
Institut für Wirtschaftsinformatik

Deutsche Bank AG
Direktor, IT-Security Manager

esslinger@fb5.uni-siegen.de

www.cryptool.org

www.cryptool.com

www.cryptool.de

www.cryptool.es

www.cryptool.pl

Weitere Kontaktadressen: siehe Readme im CrypTool-Programmpaket

Weitere Lektüre

Als Einstieg in die Kryptologie und mehr

- **Klaus Schmeh: *„Codeknacker gegen Codemacher. Die faszinierende Geschichte der Verschlüsselung“*, 2. Auflage, 2007, W3L**
- **Simon Singh: *„Geheime Botschaften“*, 2000, Hanser**
- **Udo Ulfkotte: *„Wirtschaftsspionage“*, 2001, Goldmann**
- **Johannes Buchmann: *„Einführung in die Kryptographie“*, 4. Auflage, 2008, Springer**
- **Paar / Pelzl: *„Understanding Cryptography – A Textbook for Students and Practitioners“*, 1. Auflage, 2009, Springer**
- **Klaus Schmeh: *„Kryptografie – Verfahren, Protokolle, Infrastrukturen“*, 4. Auflage, 2009, Dpunkt**
- **Swoboda / Pramateftakis / Spitz: *„Kryptographie und IT-Sicherheit -- Grundlagen und Anwendungen“*, 2008, Vieweg + Teubner**
- **Beutelspacher / Schwenk / Wolfenstetter: *„Moderne Verfahren der Kryptographie“*, 5. Auflage, 2004, Vieweg**
- **[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 [Englisch]**
- **Vielfältige Krypto-Literatur – siehe Links auf der CrypTool-Webseite sowie Quellenangaben in der Online-Hilfe von CrypTool (z.B. von Wätjen, Salomaa, Brands, Schneier, Shoup, Stamp/Low, …)**
- **Bedeutung der Kryptographie in dem breiteren Rahmen von IT-Sicherheit, Risiko-Management und organisatorischen Kontrollen**
    - Siehe z.B. Kenneth C. Laudon / Jane P. Laudon / Detlef Schoder: „Wirtschaftsinformatik“, 2. Auflage, 2009, Pearson, Kapitel 15 „IT-Sicherheit“
    - Wikipedia: http://de.wikipedia.org/wiki/Risikomanagement
    - CrypTool-Seite: http://cryptool.com/index.php/de/cryptool-for-awareness-aboutmenu-74.html

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

CRYPtOOL

Über | Funktionen | Screenshots | Dokumentation | Download | Cryptoportal für Lehrer

Aktuelle stabile Version: 1.4.21 Download

Über

- Was ist CrypTool?
- CrypTool in der Lehre
- CrypTool für Awareness
- Präsenz in Printmedien
- Auszeichnungen
- Mitwirkende
- Partnerprojekte
- Kontakt

Ausgewählter Ort 2008 in:

Deutschland
Land der Ideen

"CrypTool has been produced in an extremely professional and innovative way. It is helpful as an educational tool for the novice and provides

Was ist CrypTool?

Das Programm CrypTool ist ein freies E-Learning-Programm für Windows, mit dem kryptographische Verfahren angewendet und analysiert werden können. Diese Software wird weltweit eingesetzt. Dabei unterstützt sie eine moderne Lehre an Schulen und Hochschulen sowie die Sensibilisierung von Firmenangehörigen.

Die aktuelle Version bietet **unter anderem**:

- Zahlreiche klassische und moderne kryptographische Algorithmen (Ver- und Entschlüsselung, Schlüsselerzeugung, sichere Passworte, Authentisieren, sichere Protokolle, ...)
- Visualisierung einiger Verfahren (z.B. Caesar, Enigma, RSA, Diffie-Hellman, Digitale Signaturen, AES)
- Kryptoanalyse gegen ausgewählte Algorithmen (z.B. Vigenère, RSA, AES)
- Kryptoanalytische Messverfahren (z.B. Entropie, N-Gramme, Autokorrelation)
- Unterstützende Verfahren (z.B. Primzahltest, Faktorisierung, Base64-Kodierung)
- Lernprogramm zur Zahlentheorie
- Umfangreiche Online-Hilfe
- Begleitendes Skript mit weiterführenden Informationen über Kryptologie

Ursprünglich für IT-Sicherheits-Schulungen im Unternehmen entwickelt, hat sich CrypTool inzwischen zu einem bedeutenden Open-Source-Projekt im Bereich Kryptologie entwickelt.

Seit dem Frühjahr 2008 wird durch das CrypTool-Projekt auch das ***Cryptoportal für Lehrer*** betrieben. Dieses Portal wurde auf Anregung mehrerer Lehrer ins Leben gerufen. Hier soll insbesondere Schul-Lehrern eine Plattform geboten werden, auf der sie Unterrichtsmaterialien und Links rund um das Thema Kryptologie zur Verfügung stellen können.

Derzeit arbeitet das CrypTool-Team an zwei Zukunftsprojekten für die aktuelle, in C++ geschriebene CrypTool-Version 1.4.x. Beide Nachfolgeprojekte nutzen modernste Standards der Software-Entwicklung, sind aber noch im

www.cryptool-online.org

CrypTool-Online ist das neueste Mitglied in der Familie der CrypTool-Webseiten:

- **CrypTool-Seite** (CT1)
- **CT2-Entwicklerseite**
- **JCT-Entwicklerseite**
- **CryptoPortal** für Lehrer
- **CrypTool-Online** (Einstieg und erstes Ausprobieren direkt im Browser).

www.cryptoportal.org

Downloads: Software und CrypTool-Skript