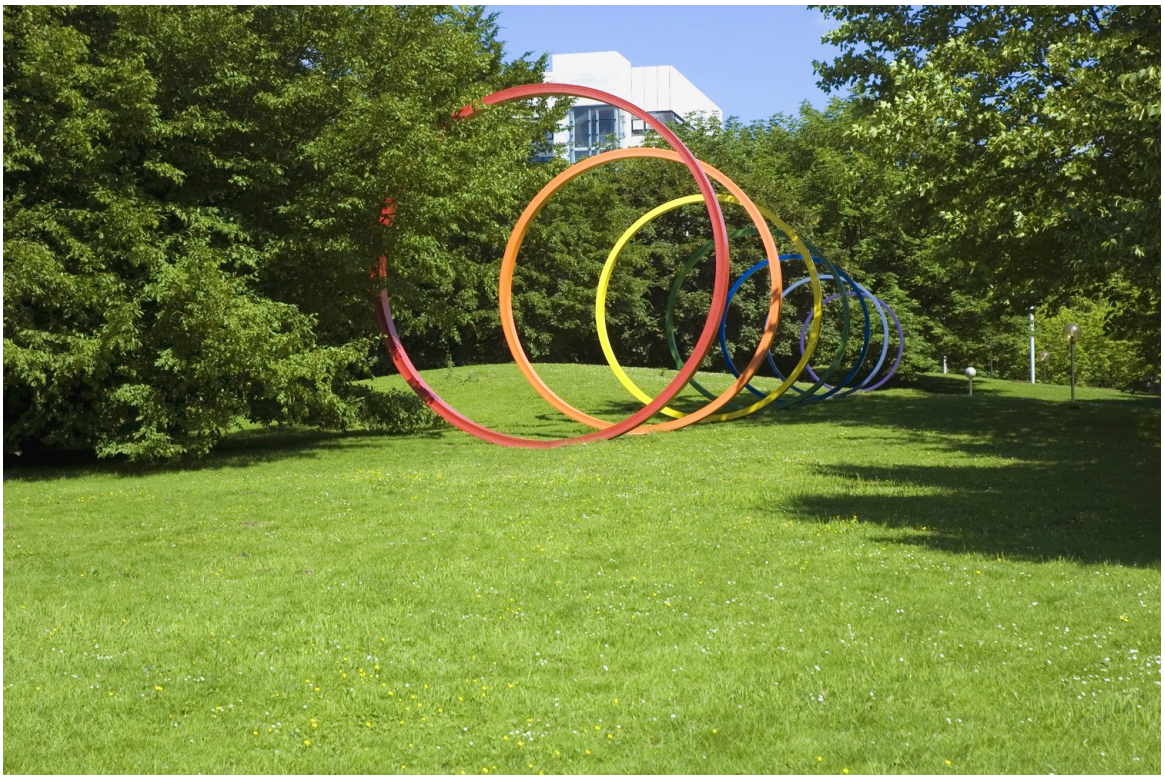


Kryptologie im Informatikunterricht der Sekundarstufe I

Arno Pasternak
Fritz-Steinhoff-Gesamtschule Hagen
TU Dortmund, Fakultät für Informatik

1. Schweizer Tag für Informatikunterricht ** 15. Januar 2010

Inhaltsverzeichnis

Vorkenntnisse

Alternativen

- Technisch orientierte Einheit und/oder
- Webseitenerstellung und/oder
- Gesellschaftlich orientierte Einheit

Alternative Vorkenntnisse

Technisch orientierte Einheit

- Ampelsteuerung
- Kopplung zweier Computer
- Datenübertragung zwischen diesen Computern
- Datenübertragung zwischen mehreren Computern
- Netzstruktur mit *offenen* Daten

Einbettung in den Informatik-Unterricht der Sek I
Die Unterrichtseinheit Kryptologie
Folgende Unterrichtseinheiten
Schlussbemerkungen
Literatur
Voraussetzungen
Ziele

Alternative Vorkenntnisse

Webseitenerstellung

- HTML-Seiten erstellen
- Transfer dieser Seiten auf einen Web-Server
- Client-Server-Struktur
- Kommunikation ist eine textbasierte Kommunikation
- Terminal-, ssh- und/oder Telnet-Sitzungen
- Gefährdung des Abhörens von ftp- und/oder Telnet-Sitzungen im Netz
- verschlüsselte Protokolle wie ssl

Alternative Vorkenntnisse

Gesellschaftlich orientierte Einheit

- Gefahren im Netz
- Kriminalität im Netz
- Online Einkaufen
- Chatten, ...

Ziele

Verschlüsselungsverfahren

- Verschlüsselung in Computernetzen ist sinnvoll und notwendig
- verschiedene Verfahren kennenlernen
- Auswahl eines geeigneten Verfahrens
- Historie der Kryptologie

Ziele

Der Computer als Verschlüsselungsautomat

- Der Computer ist die *technische Realisierung des universellen Automatens*
- Der Computer ist ein wunderbarer Ver- und Entschlüsselungsautomat
- Schüler sollen/müssen Mechanismen eines Automaten kennenlernen
- daraus folgt: Automatismus/Algorithmus *programmieren*
- Bei uns in der FSG: Programmieren bisher in COMAL

Programmiersprache

COMAL

- 70-Jahre: B. Christensen in Dänemark entwickelt didaktische Schulsprache COMAL für die Sekundarstufe I
- leider keine Weiterentwicklung

Tcl/Tk

- 80-Jahre: John K. Ousterhout entwickelt Skriptsprache Tcl/Tk mit text- und grafikbasierter Ein- und Ausgabe
- (relativ) einfach zu erlernen, obwohl nicht didaktisch geprägt
- etwas ungewöhnliche Syntax (für erfahrene Programmierer)
- auf "allen" Systemen kostenlos zu erhalten
- grosse Entwicklergemeinde

Jede andere Sprache ist natürlich prinzipiell auch verwendbar!

Die Schülerinnen und Schüler verschlüsseln einen Satz wie:

„Bald beginnt die Fussball-WM“

Prinzipielle Verfahren der Verschlüsselung

Die Verfahren lassen sich in die Klassen

- Ersetzungsverfahren,
  bei denen die Zeichen des Klartextes durch andere Zeichen derselben oder einer anderen Zeichenmenge ersetzt werden
- Tauschverfahren,
  bei denen die Reihenfolge der Zeichen des Klartextes verändert wird
- eine Kombination dieser beiden Prinzipien

einteilen.

Mögliche Probleme bei den Verfahren der Schüler

- Die Verschlüsselung ok, aber keine eindeutige Entschlüsselung möglich.
- Es werden teilweise die tollsten Zeichensätze benutzt bzw. erfunden.
- Die Schülerinnen und Schüler erkennen schnell: Beschränkung auf Buchstaben und/oder Ziffern bzw. Zahlen als Zeichen ist sinnvoll.

Das Caesar – Verfahren

Caesar-Verschlüsselung

Ein Buchstabe im Klartext wird durch den Buchstaben ersetzt, der im Alphabet drei Stellen hinter ihm steht.

```
Klartext:    Hein Bloed
Codetext:    Khlq Eorhg
```

Tcl – Programmtext: Verschlüsselung nach *Caesar*

```
proc caesar_verschluesseln {} {
  global satz

  set stelle 0
  set verschluesselter_satz ""
  set laenge [string length $satz]

  while {$stelle < $laenge} {
    set zeichen [string index $satz $stelle]
    scan $zeichen %c asciizahl

    set asciizahl [expr $asciizahl + 3]
    set zeichen [format %c $asciizahl]
    set verschluesselter_satz $verschluesselter_satz$zeichen

    set stelle [expr $stelle +1]
    }
  set satz $verschluesselter_satz
  }
```


Tcl – Programmtext: Entschlüsselung nach *Caesar*

```
proc caesar_entschluesseln {} {
  global satz

  set stelle 0
  set entschluesselter_satz ""
  set laenge [string length $satz]

  while {$stelle < $laenge} {
    set zeichen [string index $satz $stelle]
    scan $zeichen %c asciizahl

    set asciizahl [expr $asciizahl - 3]
    set zeichen [format %c $asciizahl]
    set entschluesselter_satz $entschluesselter_satz$zeichen

    set stelle [expr $stelle +1]
    }
  set satz $entschluesselter_satz
  }
```

Tauglichkeit des Caesar – Verfahrens

Gesucht ist (ohne Hilfe des Computers) der Klartext des Satzes

T]S[XRW/Xbc/^bcTa]=

Telefonjoker gefällig?

$\Longrightarrow$ Wie lautet denn nun der Klartext?

So ohne ist das also nicht! (für Schüler und/oder ....)

Tauglichkeit des Caesar – Verfahrens

per Automat (sofort!??)

ENDLICH IST OSTERN.

Das Caesar – Verfahren im 'SMS-System'

```
...
set pfad "server"
puts "CAESAR-Verschluesselungssystem"
puts "Eigene Adresse (z.B. 312 für R312) eingeben:"
gets stdin adresse
set satz ""
set wahl -1

while {$wahl != 0} {
    puts "=================================="
    puts "        Client: R$adresse"
    puts "=================================="
    ...
    puts "Informationen ausgeben:    i"
    puts ""
    puts "Satz eingeben:             1"
    ...
    puts "Satz speichern:            4"
    puts "Satz verschluesseln:       5"
    puts "Satz entschluesseln:       6"
    ...
    puts "Ende:                      0"
    puts ""
    puts ""
    puts "Eingabe:"
    gets stdin wahl
```


Das Caesar-Verfahren im 'SMS-System'

```
...
 if {$wahl == "i"} then {info_ausgeben}

 if {$wahl == 1} then {satz_eingabe}
 if {$wahl == 2} then {satz_ausgabe}
 if {$wahl == 3} then {satz_laden}
 if {$wahl == 4} then {satz_speichern}
 if {$wahl == 5} then {satz_verschluesseln}
 if {$wahl == 6} then {satz_entschluesseln}
 if {$wahl == 7} then {SMS_verzeichnis_ausgeben}
 if {$wahl == 8} then {versatz_eingeben}
 if {$wahl == 9} then {SMS_loeschen}
}
```

Screenshot des 'SMS-Systems'

```
------------------------------------------
Der aktuelle Satz lautet:  Veni, vidi, vici!
------------------------------------------

=========================
      Client: R312
=========================
           Menue
=========================

Informationen ausgeben:   i

Satz eingeben:            1
Satz ausgeben:            2
Satz laden   :            3
Satz speichern:           4
Satz verschluesseln:      5
Satz entschluesseln:      6
SMS Verzeichnis ausgeben: 7
Versatz eingeben:         8
SMS loeschen:             9
Ende:                     0


Eingabe:
```


Das Caesar – Verfahren im Einsatz

Wir haben hier einen Kryptologie – Workshop, also ...

Das probieren wir jetzt aus!

Das Vigenère – Verfahren

Vigenère – Verschlüsselung

Dieses Verfahren wird *Blaise de Vigenère* (1523-1596) zugeschrieben:
Statt einer regelmässigen Verschiebung wie bei Caesar wird die Ersetzung in Abhängigkeit von einem Codewort durchgeführt.

Verschlüsselung mit dem Codewort: *abc*

```
Klartext:    Hein Bloed
Codewort:    abcabcabca
Codetext:    Iglo Emqhe
```

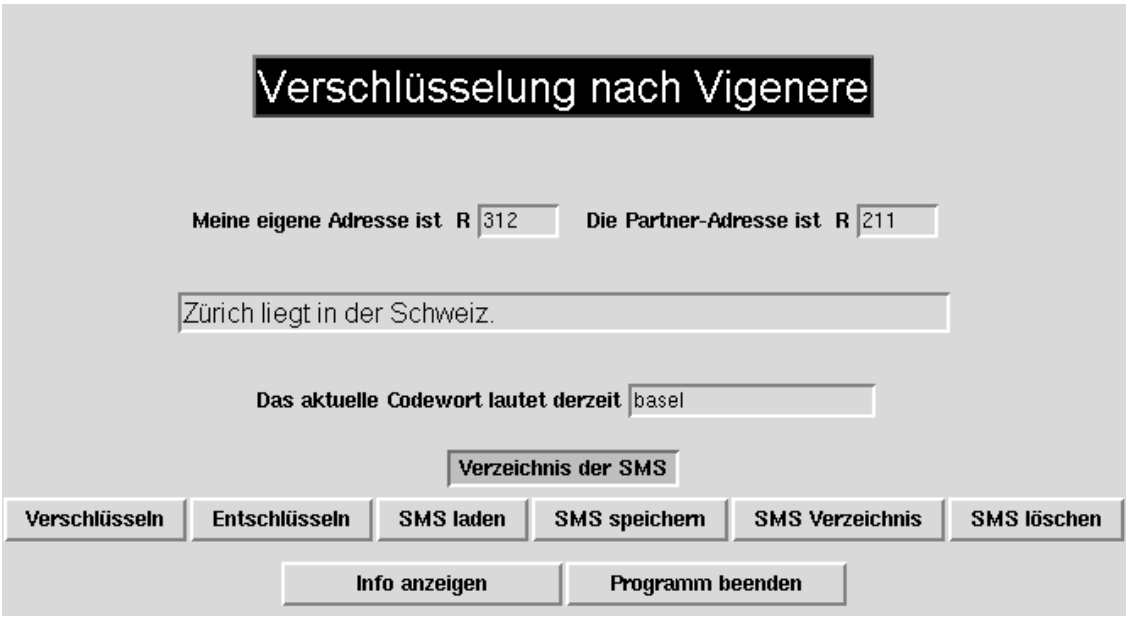

Tcl – Programmtext: Verschlüsselung nach *Vigenère*

```
proc vigenere_verschluesseln {} {
  ....
  while {$stelle < $laenge} {
    set zeichen [string index $satz $stelle]
    scan $zeichen %c asciizahl
    set verschluesselungszeichen \
        [string index $codewort $codewortstelle]
    scan $verschluesselungszeichen %c verschluesselungszahl
    set verschluesselungszahl \
        [expr $verschluesselungszahl -64]

    set asciizahl [expr $asciizahl + $verschluesselungszahl]
    set zeichen [format %c $asciizahl]
    set verschluesselter_satz $verschluesselter_satz$zeichen

    set stelle [expr $stelle + 1]
    set codewortstelle [expr $codewortstelle +1]
    if {$codewortstelle == $codewortlaenge} {
        set codewortstelle 0
        }
    }
  set satz $verschluesselter_satz
  }
```

Screenshot des 'SMS-Systems'



Das Vigenère-Verfahren im 'SMS-System'

```
# **    Aufbau der grafischen Oberfläche (Ausschnitt)   **
....
frame .kopfrahmen -pady 30
....

label .adressrahmen.adresslabel \
      -text "Meine eigene Adresse ist  R"
entry .adressrahmen.adresse -width 6 \
      -relief sunken -textvariable adresse
....
button .menuerahmen.verschluesseln \
       -text Verschlüsseln \
       -command {set satz \
        [vigenere_verschluesseln $satz $codewort]}
....
pack .kopfrahmen
pack .kopfrahmen.kopflabel
....
# **    Ende der grafischen Oberfläche    **
```

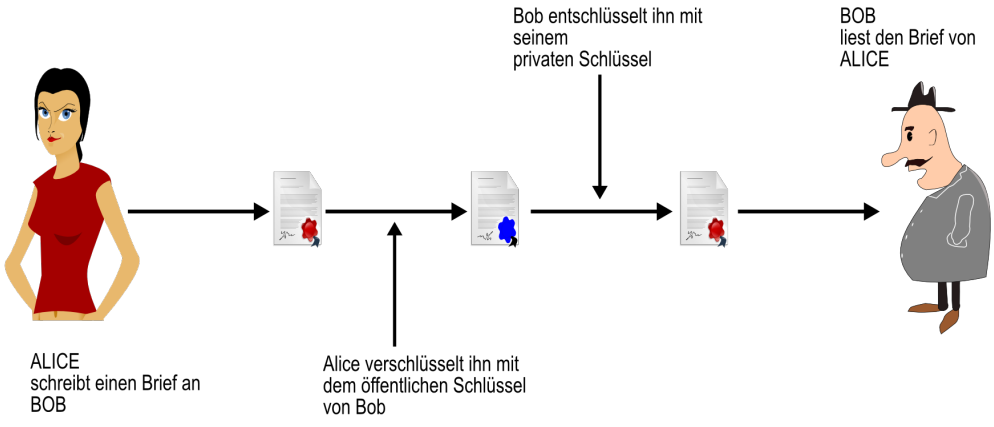
Das Public-Key – Verfahren

Public-Key – Verschlüsselung

Die Schwächen des Vigenère-Verfahrens für die *Massen*-Kommunikation werden durch das Public-Key-Verfahren ausgeglichen:

Idee: Zwei Schlüssel sind nötig

Das Public-Key – Verfahren



Das Public-Key – Verfahren

Verschlüsselung mit dem *öffentlichen Schlüssel*: 123

```
Klartext:     Hein Bloed
public Key:   1231231231
(intern:      5285285285)
Codetext:     Nijt Crsfj
```

Entschlüsselung mit dem *privaten Schlüssel*: 641

```
Klartext:     Nijt Crsfj
private Key:  6416416416
Codetext:     Hein Bloed
```


Tcl – Programmtext: *Public-Key* – Verschlüsselung (Teil I)

```
proc public_key_verschluesseln {} {
  global satz
  global schluessel

  set internercode 528
  set stelle 0
  set codestelle 0
  set internercodestelle 0
  set verschluesselter_satz ""

  set laenge [string length $satz]
  set codelaenge [string length $schluessel]
  set internercodelaenge [string length $internercode]

  while {$stelle < $laenge} {
    set zeichen [string index $satz $stelle]
    ....
    ....
    if {$internercodestelle == $internercodelaenge} {
        set internercodestelle 0
        }
    }
  set satz $verschluesselter_satz
  }
```

Tcl – Programmtext: *Public-Key* – Verschlüsselung (Teil II)

```
  while {$stelle < $laenge} {
    set zeichen [string index $satz $stelle]
    scan $zeichen %c asciizahl

    set codezahlzeichen [string index $schluessel $codestelle]
    scan $codezahlzeichen %c codezahl
    set codezahl [expr $codezahl -48]
    set internercodezahlzeichen \
        [string index $internercode $internercodestelle]
    scan $internercodezahlzeichen %c internercodezahl
    set internercodezahl [expr $internercodezahl -48]

    set summencodezahl [expr $codezahl + $internercodezahl]
    if {$summencodezahl > 9} {
       set summencodezahl [expr $summencodezahl - 10]
       }
    set asciizahl [expr $asciizahl + $summencodezahl]

    set zeichen [format %c $asciizahl]
    set verschluesselter_satz $verschluesselter_satz$zeichen

    set stelle [expr $stelle + 1]
    set codestelle [expr $codestelle +1]
    set internercodestelle [expr $internercodestelle +1]
    if {$codestelle == $codelaenge} {
        set codestelle 0
        }
    if {$internercodestelle == $internercodelaenge} {
        set internercodestelle 0
        }
    }
  set satz $verschluesselter_satz
  }
```

Tcl – Programmtext: *Public-Key* – *Ent*schlüsselung

```
proc public_key_entschluesseln {} {
  global satz
  global schluessel

  set stelle 0
  set codestelle 0
  set entschluesselter_satz ""

  set laenge [string length $satz]
  set codelaenge [string length $schluessel]

  while {$stelle < $laenge} {
    set zeichen [string index $satz $stelle]
    scan $zeichen %c asciizahl

    set codezahlzeichen [string index $schluessel $codestelle]
    scan $codezahlzeichen %c codezahl
    set codezahl [expr $codezahl -48]

    set asciizahl [expr $asciizahl - $codezahl]

    set zeichen [format %c $asciizahl]
    set entschluesselter_satz $entschluesselter_satz$zeichen

    set stelle [expr $stelle + 1]
    set codestelle [expr $codestelle +1]
    if {$codestelle == $codelaenge} {
        set codestelle 0
        }
    }
  set satz $entschluesselter_satz
  }
```

Das Public-Key – Verfahren

Was erkennen die Schüler?

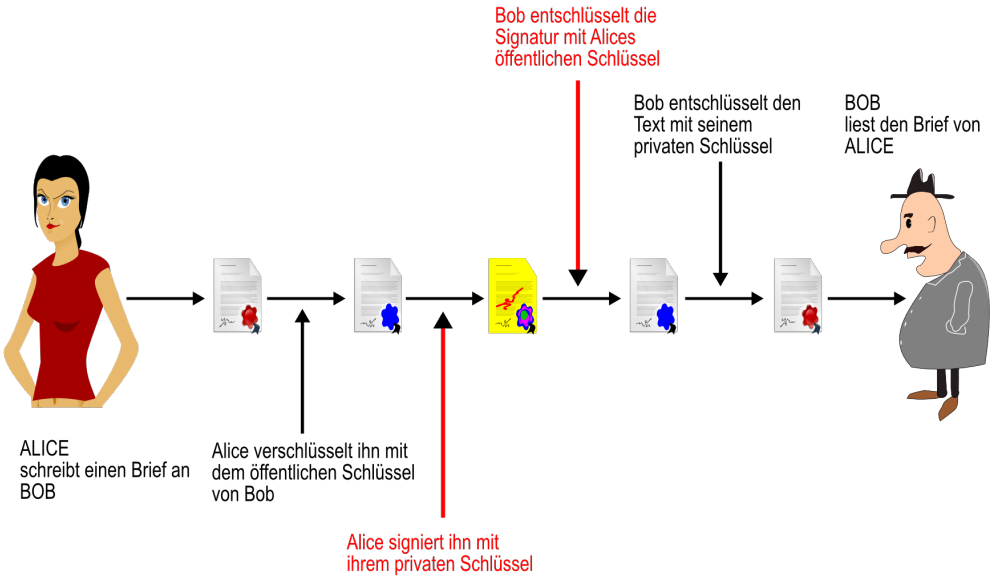
- Es gibt Verfahren, bei denen zwei Schlüssel notwendig sind: der public key und der private key.
- Die Berechnungsvorschrift ist bekannt:
  Daher ist es notwendig, dass die Berechnung der Schlüssel sehr aufwändig ist.
- Ein Problem stellt die Erstellung der Schlüsselpaare dar:
  Es ergibt sich die Notwendigkeit nach Schlüsselerzeugungsstellen oder es muss „Zufallsprinzip“ - Systeme geben.

Signieren von Daten

Die *Authenzität* von Nachrichten ist wichtig:
woher weiss ich, dass mein Gegenüber auch der ist, als der er sich auszugibt.
Naheliegend ist nun, das der Klartext zweimal verschlüsselt wird:
Zuerst wird der private Schlüssel des Absenders verwendet, dann der öffentliche Schlüssel des Empfängers.
Mit diesem Verfahren ist der Text vom Absender also signiert worden.

PS: Das von uns verwendete Verfahren ist nicht symmetrisch!

Signieren von Daten



Folgende Unterrichtseinheiten

- E-Mail
- Betriebssyteme und Netze
- Online – Banking

Schlussbemerkungen

- Am Ende der Sek I muss eine Schülerin bzw. ein Schüler ein Verständnis über Verschlüsselung besitzen.
- Das gilt für alle Schüler!
- Dies gehört in einer vernetzten Welt zur Allgemeinbildung dazu.
- Ich bezweifle, dass die dazu notwendigen Kenntnisse und Kompetenzen in einem der heutigen üblichen Pflichtfächer vermittelt werden können.
- daraus folgt:

INFORMATIK

muss in der Sek I und auch in der Sek II

PFLICHTFACH

werden!

Abrosimov, Leonid I. ; Deutschmann, Jörg ; Horn, Werner ; Reif, Holger ; Reschke, Dietrich ; Schiller, Jochen ; Seitz, Jochen ; Krüger, Gerhard (Hrsg.) ; Reschke, Dietrich (Hrsg.):
Lehr- und Übungsbuch Telematik.
Leipzig : Fachbuchverlag, 2000. –
ISBN 3–446–21053–9

Batzer, Peter:
Die ENIGMA.
In: *LOG IN*
16 (1996), Nr. 5/6, S. 44–51

Baumann, Rüdeger:
Digitale Unterschrift, Sichere Rechtsgeschäfte im Internet (Teil 1).
In: *LOG IN*
19 (1999), Nr. 2, S. 46–49

Baumann, Rüdeger:
Digitale Unterschrift, Sichere Rechtsgeschäfte im Internet (Teil 2).
In: *LOG IN*
19 (1999), Nr. 3/4, S. 82–88

Bauer, Friedrich L.:
Entzifferte Geheimnisse : Methoden und Maximen der Kryptologie; mit ... 26 Tabellen.
3., überarb. und erw. Aufl.
Berlin : Springer, 2000. –
ISBN 3–540–67931–6

Behrens, Rolf:
Zahlentheorie und Datenschutz.

Einbettung in den Informatik-Unterricht der Sek I
Die Unterrichtseinheit Kryptologie
Folgende Unterrichtseinheiten
Schlussbemerkungen
Literatur

Stuttgart : Landesstelle für Erziehung und Unterricht, 1984
(Materialien zur Lehrerfortbildung, Reihe Mathematik)

Christensen, Børge R:
Strukturierte Programmierung mit COMAL 80, 2., verbesserte Auflage.
München : Oldenbourgh-Verlag, 1985

Heine, Werner:
Die Hacker.
Reinbeck bei Hamburg : Rowohlt Taschenbuch Verlag, 1985

Harrison, Mark ; McLennan, Michael:
Effektiv Tcl/Tk programmieren.
Bonn [u.a.] : Addison-Wesley, 1998
(Professionelle Programmierung). –
ISBN 3–8273–1409–7

Hromkovič, Juraj:
Sieben Wunder der Informatik : eine Reise an die Grenze des Machbaren mit Aufgaben und Lösungen.
1. Aufl.
Wiesbaden : Teubner Verlag, 2006
(Lehrbuch Informatik). –
ISBN 3–8351–0078–5 ; 978–3–8351–0078–7

Hromkovič, Juraj:
Lehrbuch Informatik : Vorkurs Programmieren, Geschichte und Begriffsbildung, Automatenentwurf.
1. Aufl.
Wiesbaden : Vieweg und Teubner Verlag, 2008

technische universität dortmund
Arno Pasternak
Kryptologie im Informatikunterricht der Sekundarstufe I (38/38)

(Studium). –
ISBN 978–3–8348–0620–8

Kippenhahn, Rudolf:
Verschlüsselte Botschaften, 4. Auflage.
Hamburg : Nikol-Verlag, 2006

Köhler, Klaus:
Sicherheit durch Kryptographie?
In: *FIFF Kommunikation*
20 (2003), Nr. 2, S. 39–46

Koubek, Jochen:
Sicherheit von Online-Bezahldiensten.
In: *LOG IN*
(2006), Nr. 140, S. 25–29

Ousterhout, John K.:
Tcl und Tk : Entwicklung grafischer Benutzerschnittstellen für das X-Window-System.
1. Aufl.
Bonn : Addison-Wesley Professional Computing, 1995
(Professional computing). –
ISBN 3–89319–793–1

Pasternak, Arno:
Was wir vergessen haben.
(2005). –
http://pasternak.in.hagen.de/if/koenigstein/vergessen.htm

Rüger, Hubert:

Einbettung in den Informatik-Unterricht der Sek I
Die Unterrichtseinheit Kryptologie
Folgende Unterrichtseinheiten
Schlussbemerkungen
Literatur

Information Security.
In: *FIFF Kommunikation*
20 (2003), Nr. 2, S. 37–38

Schwill, Andreas:
Fundamentale Ideen der Informatik.
In: *Zentralblatt. für Didaktik der Mathematik*
(1993), S. 20–31

Seiffert, Monika:
Verschlüsselungsmethoden (Teil 2).
In: *LOG IN*
14 (1994), Nr. 3, S. 33–40

Strathern, Paul:
Turning & der Computer.
Frankfurt a.M. : Fischer Taschenbuchverlag, 1997

Schulz, Ralph-Hardo ; Witten, Helmut:
RSA & Co. in der Schule (Neue Folge – Teil 1).
In: *LOG IN*
(2006), Nr. 140, S. 45–54

Schulz, Ralph-Hardo ; Witten, Helmut:
RSA & Co. in der Schule (Neue Folge – Teil 2).
In: *LOG IN*
(2006), Nr. 143, S. 50–58

Tranquillus, Gaius S.:

technische universität dortmund
Arno Pasternak
Kryptologie im Informatikunterricht der Sekundarstufe I (38/38)

Divus Julius
(Vitae Caesarum)

Webster, Tim ; Francis, Alex (Hrsg.):
Tcl/tk für Dummies : gegen den täglichen Frust mit Tcl/Tk.
Bonn : mitp, 2000. –
ISBN 3–8266–2872–1

Wilkens, Ulrike:
Das allmähliche Verschwinden der informationstechnischen Grundbildung.
Aachen : Shaker-Verlag, 2000. –
ISBN 3–8265–7125–8