Alpha

Stand: 21. Juli 2002
Version 1.0.0

KryptoKomp

Einführung in die Kryptologie

für Anfänger und Fortgeschrittene

Alexander Schick

alias

[chronyx]

Inhaltsverzeichnis

Kryptologie

Intro

Teil I - Kryptografie

symmetrische Algorithmen

asymmetrische Algorithmen

digitale Signaturen

Hashverfahren

weiteres

Teil II - Kryptoanalyse

Teil III - Kryptologie und die Gesellschaft

Teil IV - ergänzendes Material

Teil V - ausgewählte Bücher und sonstige Quellen

Extro

Vorwort

Warum ich dieses Werk geschrieben habe

Ich persönlich empfinde die Kryptologie als ein sehr interessantes Gebiet, welches für mich eine Herausforderung für den Geist darstellt. Diese Erfahrung möchte ich auch anderen Menschen zugänglich machen, die an der Thematik interessiert sind und einen Einstieg suchen. Um diesen zu erleichtern habe ich mich bemüht, die zum grundlegenden Verständnis notwendigen Kenntnisse zusammenzutragen und niederzuschreiben. Aus diesem Antrieb heraus ist dann schließlich das entstanden, was Sie gerade in den Händen halten, das ***KryptoKomp***.

Wie dieses Werk aufgebaut ist

Die Gliederung dürfte selbsterklärend sein, weshalb ich darauf keine weiteren Worte verwenden will. Viel wichtiger ist mir an dieser Stelle zu erläutern, wie das ***KryptoKomp*** in Zukunft weitergeführt werden wird:
Das ***KryptoKomp*** ist so strukturiert, dass einzelne Texte jederzeit aktualisiert werden können. Am oberen rechten Rand der Titelseite findet sich das Datum, an welchem es das letzte Mal bearbeitet wurde. Des Weiteren kann zusätzlich an der jeweiligen Versionsnummer auf dem Deckblatt leicht auf einen Blick erkannt werden, ob die aktuellste Version vorliegt. Die Nummer setzt sich aus drei Teilen zusammen - ***x.y.z*** - und wird wie folgt aufgeschlüsselt:

- z wird jedes Mal erhöht, wenn einzelne Texte überarbeitet und verändert wurden
- y wird beim Hinzufügen neuer bzw. beim Entfernen alter Texte inkrementiert
- x wird hochgezählt, falls ein neues Kapitel (also ein eigenständiger Hauptteil) dazukommt

Da ich nicht vorhabe einen Newsletter für das KryptoKomp einzuführen lohnt es sich, in mehr oder weniger regelmäßigen Abständen bei der offiziellen Quelle vorbeizuschauen. Ich denke, alle zwei bis drei Monate dürfte dabei völlig ausreichend sein.

Was ich von Ihrer Meinung halte

Sehr viel! Ich freue mich über jede einzelne E-Mail. Sei es Ihre Meinung, Anregungen, ein einfaches Hallo - egal was, nur her damit!
Mir persönlich am liebsten sind natürlich Lobgesänge, für das ***KryptoKomp*** am Besten sind hingegen Verbesserungsvorschläge, Fehlermeldungen und Ergänzungsmaterial.

Was die rechtlichen Aspekten angeht

Da ich selbst das ***KryptoKomp*** geschrieben habe liegen selbstverständlich alle Rechte erst einmal bei mir. Das schließt eine kommerzielle Nutzung durch andere ohne meine ausdrückliche Erlaubnis von vorneherein aus. Gegen was ich hingegen nichts einzuwenden habe ist die kostenlose Verteilung, sei es durch das Hochladen auf die eigene Homepage, durch das Weitergeben an einen Freund oder Freundin oder durch sonstige Verteilerwege, solange der Autor (chronyx@paramind.org) und

die offizielle Quelle (http://www.paramind.org/) genannt werden, nichts verändert wird und ich darüber in einer E-Mail informiert werde. Außerdem darf kein Copyright und sonstige Rechte verletzt werden.

Zum Thema Copyright: Ich habe mich bemüht, keine Copyrights oder sonstige Rechte zu verletzten. Sollte dies dennoch der Fall sein, dann bitte ich um schnellstmögliche Benachrichtigung, so dass der Missstand augenblicklich behoben werden kann.

Die hier vorgestellten Informationen betreffend kann ich sagen, dass ich mich bemüht habe keine Fehler zu machen, jedoch kann ich die Fehlerfreiheit nicht garantieren.

Halte ich hier wirklich das Original in den Händen?

Da ich - wie oben erwähnt - nichts dagegen habe, wenn das KryptoKomp kostenlos verteilt wird, ist es gut möglich, dass Ihre Version nicht von der offiziellen Quelle stammt. Das macht nichts, solange nichts verändert wurde. Doch wie können Sie diesbezüglich sicher sein? Können Sie das nachprüfen?
Die Antwort lautet: Ja. Ich habe das Programm ***md5sum.exe*** benutzt, um eine Liste mit Checksummen zu erstellen, anhand derer die Echtheit der Dateien nachgeprüft werden kann. Das Programm kann u.a. auf http://www.etree.org/ heruntergeladen werden; dort findet sich auch eine gute Beschreibung, wie es zu bedienen ist.
Die Liste mit den Checksummen ist bei der offiziellen Quelle zu finden und nur dort! Nur so ist die Echtheit der Liste - zumindest einigermaßen - sichergestellt. Die offizielle Quelle ist - um es noch einmal in aller Deutlichkeit zu sagen - die PARAllel MINDs Cooperation (http://www.paramind.org/). Solltet Ihr eine veränderte Version finden, dann wäre ich für eine Benachrichtigung dankbar.

Wer hat das KryptoKomp eigentlich geschrieben?

Mein richtiger Name ist Alexander Schick. Im Internet benutze ich den Spitznamen ***chronyx*** und arbeite bei der ***PARAllel MINDs Cooperation***. Fragen beantworte ich gerne ... einfach an chroynx@paramind.org schicken.

Doch nun genug der langen Vorreden! Viel Spaß beim Lesen und Lernen!

Teil I – Kryptografie

Einführung in die Kryptografie

Seitdem Menschen dazu fähig sind, Intrigen zu spinnen, Pläne zu schmieden und diese in irgendeiner Form aufzuzeichnen gibt es das Bedürfnis, geheime Informationen einem ausgewählten Personenkreis mitzuteilen, ohne dass Unbefugte vom Inhalt der Nachricht erfahren. Die Kryptografie, eine Disziplin der Kryptologie, beschäftigt sich genau mit diesem Problem und das schon seit Jahrtausenden. Von einfachen, heute fast primitiv scheinenden bis hin zu hochkomplexen, auf mathematischen und physikalischen Problemen und Sachverhalten basierenden Verfahren hat sich diese Wissenschaft entwickelt.
In diesem ersten Teil werden die bekanntesten Algorithmen und Vorgehensweisen erläutert, jedoch ohne im großen Stil auf den geschichtlichen Hintergrund einzugehen. Dies würde bei weitem den Rahmen dieser Publikation sprengen. Wer jedoch Interesse an der Historie hat, dem werden die Buchvorstellungen in Teil V weiterhelfen.
Bevor ich nun endlich anfangen, Sie in die Geheimnisse dieser faszinierenden Wissenschaft einzuführen, sind noch einige Begriffserklärungen, Darstellungsarten und Grundsätze zu klären. Erschrecken Sie nicht, wenn Sie mit einem Großteil zum jetzigen Zeitpunkt noch nichts anfangen können; hier sind zu Gunsten der besseren Nachschlagmöglichkeit eine Vielzahl von Grundsätzen, Regeln und Vereinbarungen zusammengefasst, welche Sie zum Teil erst in den fortgeschrittenen Kapiteln benötigen. Kehren Sie am besten regelmäßig zu dieser Seite zurück, sobald Sie neue Begriffe sehen oder etwas nachschauen wollen.

Kryptologie, Kryptografie, Kryptoanalyse

Das griechische Wort "kryptos" bedeutet "geheim", "logos" ist die "Lehre". Kryptologie ist demzufolge die "Lehre von etwas geheimem". Da es aber immer zwei Standpunkte bezüglich geheimen Dingen gibt - der eine will etwas geheim halten, der andere will etwas geheimes unbefugt in Erfahrung bringen - spaltet sich diese Wissenschaft in zwei Pfade auf. Die Kryptografie ("graphein", schreiben - übersetzt etwa Geheimschrift) stellt bestimmte Vorgehensweisen auf, mit denen eine Botschaft geheim gehalten und nur den berechtigten Personen zugänglich gemacht werden kann. Die Kryptoanalyse hingegen hat als Ziel, die kryptografischen Verfahren zu brechen und unberechtigt Zugang zu der geheimen Botschaft zu erlangen.
Man kann sich denken, dass im Laufe der Geschichte ein Katz und Maus-Spiel zwischen den Vertretern beider Richtungen stattgefunden hat. Die eine Seite trumpfte mit immer besseren Verfahren auf, welche von der anderen Seite immer und immer wieder gebrochen wurden, bis ... aber das werden Sie später noch lesen.
Es wäre jedoch fatal sich auf nur eine Seite festzulegen, denn auch wenn sich diese Wissenschaft in zwei Hauptrichtungen aufspalten lässt, so gab und gibt es jedoch kaum einen Wissenschaftler einer Richtung, der nicht auch die andere beherrscht. Denn um auf einem Gebiet gut zu sein muss man mit den Fähigkeiten und Kenntnissen des Gegners vertraut sein.

Grundbegriffe und Darstellungsarten

Die Nachricht, welche verschlüsselt werden soll, wird als ***Klartext*** bezeichnet. Dieser wird durch ein Verfahren (***Algorithmus***) unter Einbeziehung eines ***Schlüssels*** in den ***Geheimtext*** (auch ***Chiffre*** genannt) umgewandelt - er wird ***verschlüsselt*** bzw. ***chiffriert***. Der Empfänger des Geheimtextes wendet nun ebenfalls ein Verfahren zum ***Entschlüsseln*** (***Dechiffrieren***) unter Einbeziehung des Schlüssels an (oft ist es das gleiche Verfahren, nur rückwärts angewandt) und erhält als Ergebnis wieder den Klartext.
Bei Beispielen wird für den Klar- und Geheimtext meist noch eine entsprechende Darstellungsart verwendet, um die Unterscheidung zu erleichtern. Der ***klartext*** wird durchgängig klein geschrieben, der ***GEHEIMTEXT*** groß. Dies heißt jedoch nicht, dass dies in der "freien Wildbahn" ebenfalls so ist.
Um sich das Zusammenwirken von Algorithmus und Schlüssel besser vorstellen zu können, geht man am besten davon aus, dass der Algorithmus eine allgemeine Vorschrift ist (für alle Benutzer gleich!), welche den Umgang mit dem Klartext vorgibt - jedoch gibt es in dem Algorithmus einen Platzhalter (Variable), welche für den Schlüssel steht. Da jeder Benutzer einen individuellen Schlüssel besitzt, können mehrere Leute das gleiche Verfahren benutzen und jeder erzielt individuelle Ergebnisse (naheliegende Analogie ist das Türschloss - der Algorithmus ist gleich, die Schlüssel sind verschieden, also kann man nicht einfach in das Haus des Nachbarn marschieren).
Da die Kryptologie eng mit der Mathematik verwandt ist verwundert die ausgeprägte Verwendung der kurzen und prägnanten mathematischen Schreibweise nicht. Die Algorithmen werden durch Funktionen dargestellt, bei der Verschlüsselung E(x) (encrypt), bei der Entschlüsselung D(x) (decrypt). Der Schlüssel (key) findet sich in der Variablen k, bzw. bei Rundenschlüsseln in k_i wieder. Das Zusammenwirken zwischen Algorithmus und Schlüssel wird entweder als Funktionenschar $E_k(x)$ oder als zusätzlicher Parameter E(x, k) ausgedrückt. Jetzt fehlt nur noch der Platzhalter x - an seine Stelle tritt entweder die Variable m (message, Klartext) oder c (cipher, Geheimtext); falls die Daten blockweise bearbeitet werden sollten, wird der entsprechenden Variablen ein Index hinzugefügt, beispielsweise c_{i-1}.

Welche Arten von Sicherheiten gibt es?

Diese Frage mag sich bei erstmaligem Lesen etwas seltsam anhören, weil es entweder Sicherheit gibt oder nicht gibt. Da jedoch aber bei den in der Kryptologie verwendeten mathematischen Verfahren Pseudo-Einwegfunktionen vorkommen können, die theoretisch zwar lösbar sind, aber mit einem Zeitaufwand von einigen Millionen Jahren verbunden sind, wurden für den Begriff Sicherheit drei Unterarten definiert.
Die ***absolute Sicherheit*** ist, wie der Name schon sagt, absolut. Es gibt kein Sicherheitsloch und keine Möglichkeit, die gewünschte Information auf einem zuverlässigen Weg zu erlangen. Da der Begriff absolute Sicherheit wirklich ernst genommen wird, bedeutet das für kryptografische Algorithmen, dass sie mathematisch nachweisbar zu 100% sicher sein müssen. Dieses Kriterium erfüllen in der Praxis nur wenige, als Beispiel sei an dieser Stelle das One-Time-Pad genannt.
Neben der absoluten Sicherheit gibt es noch die ***praktische Sicherheit***, welche den Kryptoanalytikern weitaus lieber (da brechbar) ist, die folgendermaßen definiert werden kann: Wenn entweder der Aufwand zum Knacken einer Chiffre den Wert der

Nachricht übersteigt oder der Zeitaufwand zur Durchführung dieses Vorhabens zu hoch ist und die Nachricht somit nicht mehr aktuell wäre, dann bezeichnet man das Verfahren als praktisch sicher.
Bietet ein Verfahren ***keine ausreichende Sicherheit***, dann reiben sich die Herren Kryptoanalytiker die Hände und beginnen mit dem Knacken der Nachricht, da sie nun endlich realistische Aussichten auf Erfolg haben. Diese Situation tritt in unserer heutigen Zeit bei einem gewissenhaften Umgang mit den kryptografischen Werkzeugen jedoch sehr selten auf.

Konfusion und Diffusion

Diese zwei Begriffe finden sich vor allem bei den neuen Algorithmen wieder und werden als qualitative Aussagen über eine Chiffre benutzt. Je stärker und je effektiver diese Eigenschaften realisiert worden sind, desto schwerer ist normalerweise auch eine Kryptoanalyse.
Mit ***Konfusion erzeugen*** wird zum Ausdruck gebracht, dass der Zusammenhang zwischen Klar- und Geheimtext verschleiert wird. So wie auch das im Deutschen vorkommende Wort "konfus" einen verwirrten und nicht logischen Zustand ausdrückt, soll auch der Geheimtext als ein möglichst wirrer Zeichenhaufen erscheinen, der idealerweise keinen logischen Zusammenhang mit der normalen Sprache aufweisen sollte. Konfusion kann beispielsweise auf primitive Weise erzeugt werden, indem man die Geheimtextlänge durch Dummyzeichen verändert, aber auch das absichtliche Einbauen von Rechtschreibfehlern oder Dialekten wirkt auf einen nichts ahnenden Analytiker oft sehr konfus.
Mit ***Diffusion erzeugen*** meint man die Verteilung der enthaltenen Informationen über den gesamten Text. Der Begriff "Diffusion" ist ebenfalls in der Biologie und Chemie zu finden und drückt dort das Einstellen einer gleichen Konzentration in einem geschlossenen Raum aus. So verteilt sich ohne Umrühren die Milch im morgendlichen Kaffee trotzdem, auch wenn es etwas länger dauert, da die Natur nach Ausgeglichenheit strebt. Auf die Kryptologie übertragen kommt jeder Buchstabe mit einer bestimmten Häufigkeit, einer bestimmten Konzentration in jedem Text vor, was einen möglichen Angriffspunkt für eine Kryptoanalyse darstellt. So darf sich als Beispiel die relative Häufigkeit des deutschen Buchstabens "e" niemals im Geheimtext wieder finden, da einer Entschlüsselung somit nicht mehr vieles im Wege stehen würde.

Shannon und Kerckhoff

Diese beiden Herren stellten zwei wichtige Grundaussagen auf, die gerade bei modernen Verfahren Anwendung finden, um ein Grundniveau an Sicherheit zu gewährleisten.
Shannons Maxime: "Der Feind kennt das benutzte kryptografische Verfahren."
Kerckhoffs Maxime: "Die Sicherheit eines kryptografischen Verfahrens beruht alleine auf dem Schlüssel, der zum Dechiffrieren benötigt wird."
Hält man sich an die erste Maxime, dann geht man davon aus, dass der Feind genau weiß, wie Nachrichten verschlüsselt werden (der Feind kennt also den Algorithmus). Die einzige Unbekannte ist für ihn der Schlüssel. Hier kommt die zweite Maxime ins Spiel, die ein Verfahren erst als sicher einstuft, wenn der Dechiffrierschlüssel das einzige geheime Element darstellt und der Rest öffentlich bekannt sein darf.
Hält man sich an beide Maximen, so vermeidet man den Fehler, sich auf falsche Annahmen zu stützen, wie "der Feind bekommt nie den Bauplan unserer Maschinen".

Zu schnell ist es im Ernstfall passiert, dass ein Exemplar der Maschine erbeutet wird, oder dass auf subtileren Wegen Kenntnisse darüber gewonnen werden. Im Friedensfall bietet die Einhaltung der ersten Maxime zusätzlich den Vorteil, dass man Verfahren öffentlich bekannt geben kann, um sie so von einer Vielzahl unabhängiger Denker auf Schwachstellen überprüfen lassen zu können. Einzig und allein der Schlüssel darf das Element darstellen, welches geheim gehalten werden muss.

Hintergrund: symmetrische Algorithmen

Was sind symmetrische Algorithmen?

Ein Verfahren gilt als symmetrisch, wenn sowohl für die Ver- als auch für die Entschlüsselung ein und derselbe Schlüssel verwendet wird, welcher für eine sichere Anwendung absolut geheim gehalten werden muss. Keine Bedingung, aber ein häufiges Merkmal ist zudem die Ähnlichkeit der Ver- und Entschlüsselungsfunktion, welche beide nicht selten identisch sind (E(x)=D(x), E(E(m))=m) bzw. das genaue Gegenteil darstellen.

Vorteile

Symmetrische Algorithmen zählen zu den schnellsten kryptografischen Verfahren, welche selbst große Datenmengen zügig verschlüsseln können. Dabei ist der Geheimtext - falls überhaupt - nicht wesentlich größer als das Original, wodurch auch bei der verschlüsselten Speicherung riesiger Datenbestände keine weiteren Probleme auftreten. Da sie zu den am besten erforschten Verschlüsselungsalgorithmen zählen haben sie mittlerweile ein sehr hohes Sicherheitsniveau erreicht, welches auch strengsten Ansprüchen genügt und darüber hinaus durch ihre Variationsvielfalt für fast jedes Anwendungsgebiet die passende Antwort parat haben. Und zu guter Letzt ermöglichen sie durch ihre Linearität sowohl einen einfachen soft- als auch hardwareseitigen Einsatz.

Nachteile

So schön sich die Vorteile auch anhören mögen sind dennoch einige schwerwiegende Nachteile vorhanden, welche die Verwendung symmetrischer Verschlüsselungsverfahren einschränken. Will man die Verschlüsselung für kommunikationstechnische Belange einsetzen, dann muss zuerst einmal der Schlüssel auf einem sicheren Wege ausgetauscht werden. Dies kann häufig ein ernsthaftes Problem werden, da Leitungen angezapft, Boten bestochen und Briefe abgefangen werden können. Selbst wenn dieses Problem überwunden und die geheimen Schlüssel ausgetauscht wurden, so wollen diese noch verwaltet werden. Bei einer handvoll Schlüsseln ist dies nicht weiter schwierig, aber schon bei zehn (n) Kommunikationsteilnehmern sind 45 geheime Schlüssel notwendig (n(n-1)/2), Tendenz stark steigend.
Zusätzlich zu den Problemen des Schlüsselaustausches und des Schlüsselmanagements kann man mit symmetrischen Algorithmen nicht gut digitalen Signaturen realisieren, was das Einsatzgebiet weiter einschränkt.

Transposition

der Algorithmus

Bei der Transposition handelt es sich um eine Neuanordnung der Zeichen einer Nachricht in einer neuen Reihenfolge nach einer vorher festgelegten Vorschrift. Bei diesem Verfahren beschreibt der Algorithmus das prinzipielle Vertauschen von Zeichen, während der geheim zu haltende Schlüssel nähere Angaben zu dem tatsächlichen Ablauf, also wer mit wem vertauscht wird, macht.
So könnte ein denkbarer Schlüssel das Vertauschen des ersten mit dem letzten Buchstaben, dann des zweiten mit dem vorletzten Buchstaben, usw. beschreiben.
Die Entschlüsselung ist bei der Transposition besonders einfach, denn sie ist mit der Verschlüsselungsfunktion identisch, weshalb der Geheimtext einfach nochmals verschlüsselt werden muss. Es gilt also in diesem Fall: $E(m) = D(m)$, woraus $E(E(m)) = m$ folgt.

die liebe Theorie

Auf den ersten Blick erscheint dieses Verfahren der Kryptografie als unsicher und - es ist es genau genommen auch. Jedoch darf man nicht vergessen, dass es schon bei einer Nachricht mit nur zehn (n) Zeichen insgesamt 3.628.800 (n!) Möglichkeiten gibt, die Buchstaben des Textes unterschiedlich anzuordnen. In Kombination mit anderen Verfahren kann dies als Verstärkung durchaus sehr nützlich sein.

Alice und Bob

Alice und Bob sind auf einer langweiligen Party eingeladen und sitzen mit den Feiernden zusammen am Tisch. Alice will Bob eine kurze Nachricht zukommen lassen, jedoch verhindern, dass Außenstehende den Inhalt bei kurzer Betrachtung des Textes sofort erkennen können. Also schreibt sie die Nachricht "Langweilig! Gehen wir im Internet surfen?" einfach rückwärts auf eine Serviette und schiebt sie Bob zu. Dieser weiß wie er die Nachricht "nefrustenretnimiriwneheggiliewgnal" zu lesen hat und beide verabschieden sich. Den restlichen Abend verbringen sie auf der ParaMind-Homepage bevor sie vollgestopft mit interessanten und wertvollen Informationen glücklich ins Bett fallen.

weitere Varianten

Die beiden folgenden Vorgehensweisen sind die wohl bekanntesten Anwendungsbeispiele der Transposition und dürfen hier keinesfalls fehlen:

Gartenzaun-Transposition

Bei der Gartenzaun-Transposition wird (im übertragenen Sinne) jede Latte eines Zaunes von oben bis unten beschrieben (z.B. drei Buchstaben pro Latte) und der Geheimtext gebildet, indem man die beschrifteten Latten zeilenweise abschreibt (erster Buchstabe der ersten Latte, erster Buchstabe der zweiten Latte, ...).

Skytale
Die Skytale ist eine Walze mit vorgegebenem Durchmesser, um welche man einen Papierstreifen wickelt und die Nachricht zeilenweise draufschreibt. Erst wenn der Empfänger diesen Streifen nun um eine Skytale mit dem gleichen Durchmesser wickelt, kann er die Nachricht im Klartext lesen.

"Mach mal, Malcom!"

Malcom hat es sich als Kryptoanalytiker zur Aufgabe gemacht, Angriffspunkte für jedes kryptografische Verfahren zu finden, welches er unter die Finger bekommen kann. So macht er sich in seinem Arbeitszimmer zu Hause nun an seine erste Analyse...
Da dieses Verfahren meist von Menschen angewandt wird, ist es in der Praxis oft auf kurze, einfach zu merkende und/oder besonders prägnante Schlüssel beschränkt, welche er bei einem Angriff zuerst mittels einer speziell angepassten Brute-Force Attacke testet. Hat er damit keinen Erfolg, dann kann er - bevor er nicht nur logisch erscheinenden, sondern tatsächlich alle Kombinationen testet (Brute-Force) - nach sprachbedingten Regelmäßigkeiten suchen, wie beispielsweise die Abstände ausgewählter, in der Regel als Kombination auftretende Buchstaben und daraus Rückschlüsse auf den Schlüssel zu ziehen versuchen.

Substitution

Die Bezeichnung Substitution hat ihren Ursprung im lateinischen Wort "substituo", auf Deutsch "an die Stelle setzen". Bei Substitutionsalgorithmen findet folglich eine Ersetzung der Klartextbuchstaben durch Geheimtextbuchstaben statt, welche einer vorher aufgestellten Vorschrift, dem Algorithmus, folgt. Der Schlüssel gibt dabei an, welche Buchstaben mit welchen vertauscht werden und ist sowohl für die Ver- als auch für die Entschlüsselung gleich.
In der Praxis finden sich vor allem die monoalphabetische und die polyalphabetische Substitution, welche anschließend genauer erklärt werden.

monoalphabetische Substitution

der Algorithmus

Bei der monoalphabetischen Substitution wird (allgemein gesprochen) jeder Klartextbuchstaben in einen Geheimtextbuchstaben überführt bzw. durch einen solchen ersetzt. Der geheime Schlüssel gibt die korrespondierenden Klar-/ Geheimtextbuchstabenpaare an.
In der Regel wird als Schlüssel ein sogenanntes Geheimtextalphabet aufgestellt, welches für jeden Klartextbuchstaben genau einen Geheimtextbuchstaben vorgibt. Soll nun ein Text verschlüsselt werden, so wird dieser Zeichen für Zeichen durchgegangen und zu jedem Klartextbuchstaben anhand des Geheimtextalphabets der passende Geheimtextbuchstaben ausgewählt, welchen man dann niederschreibt. Auf diese Weise geht man den gesamten Klartext durch bis er vollständig chiffriert worden ist.
Tritt der spezielle Fall ein, dass es sich bei dem Geheimtextalphabets um eine einfache Verschiebung des Klartextalphabets handelt (a zu G, b zu H, usw.), dann kann

die Substitution kürzer auch als Addition ausgedrückt werden (übrigens ebenfalls auf Bitebene möglich). Dafür gelten für gewöhnlich folgende Regeln:
A = 0, B = 1, ..., Z = 25
A + A = A // A + B = B // B + B = C // B + Z = A
Zum Dechiffrieren des Geheimtextes dreht man das Verfahren bzw. genauer gesagt den Schlüssel um und verschlüsselt den Geheimtext nochmals. So wird aus der Vereinbarung a = I, b = J, usw. der Dechiffrierschlüssel I = a, J = b, usw. gebildet, mit welchem man durch eine weitere Chiffrierung den Klartext erhält.

die liebe Theorie

Bei einem Alphabet mit n Buchstaben gibt es n! mögliche Geheimtextalphabete, was beim deutschen Alphabet ohne Sonderzeichen (26 Buchstaben) in Zahlen 403.291.461.126.605.635.584.000.000 entspricht. Eine ungeheuer groß erscheinende Zahl, welche dazu führte, dass dieses Verfahren lange Zeit als sicher angesehen wurde. Da es in der Praxis einem einfachen Brute-Force Angriff auch tatsächlich widerstehen kann, wurde dieser Irrglaube zusätzlich noch bestätigt. So bräuchte ein Computer, der eine Milliarde mögliche Kombinationen pro Sekunde ausprobieren könnte, dafür im Mittel dennoch mehr als 6 Milliarden Jahre, was ungefähr dem sechsfachen Alter unserer Erde entspricht. Weiter unten werden indirektere und weitaus effektivere Ansätze beschrieben, durch welche diese riesige Zahl jedoch drastisch reduziert.

Alice und Bob

Da die Transposition zu leicht zu durchschauen ist, vereinbaren Alice und Bob ein einfach zu merkendes Geheimtextalphabet, um bei der nächsten langweiligen Party eine schnelle, aber dennoch einigermaßen sichere Kommunikationsmöglichkeit zu haben. Das Geheimtextalphabet sieht folgendermaßen aus:

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

Alice testet nun das Verfahren, indem sie folgende Botschaft verschlüsselt an Bob schickt: "Komm nachher zu mir, ich moechte Dir was schoenes zeigen!" Verschlüsselt ergibt das folgende Nachricht:
"NRPPQDFKKHUCXPLULFKPRHFKWHGLUZDVVFKRHQHVCHLJHQ".
Bob entschlüsselt die Nachricht und besucht total aufgeregt gegen Abend Alice. Dort wird er auch nicht enttäuscht, denn Alice zeigt ihm die neuesten ParaMind-Nachrichten im zweiten Level von ParaMind, welche wieder brisante Neuigkeiten enthalten.

weitere Varianten

Die effektivsten Varianten sind mittlerweile so bekannt, dass sie einen eigenen Text verdienen (polyalphabetische Substitution direkt im Anschluss und Nomenklator). Weniger bekannt hingegen ist die ***homophone Verschlüsselung***, welche das Problem der noch im Geheimtext existierenden relativen Häufigkeiten (siehe dazu auch weiter unten) bekämpft und den Angriff mittels einer Häufigkeitsanalyse vereiteln

soll. Dazu werden für jeden Buchstaben so viele Geheimtextbuchstaben (entweder durch zusätzliche Symbole oder durch mehrstellige Zahlen) deklariert und verwendet, so dass die relative Häufigkeit jedes im fertigen Geheimtext vorkommenden Buchstabens genau 1% beträgt (zur Verdeutlichung: für den Buchstaben e mit einer relativen Häufigkeit von 17,40% müssten 17 Geheimtextbuchstaben deklariert werden, während für z mit 1,13% ein einziger Geheimtextbuchstabe ausreichend wäre - Häufigkeitsangaben nach Simon Singhs Geheime Botschaften).

Als besonders bekannte Beispiele sind zusätzlich noch zwei Stück anzuführen. Zum einen ist dies die historische Caesar-Chiffre, welche der berühmte römische Feldherr und spätere Diktator zur Kommunikation mit seinen Feldherren benutzte. Hierbei handelt es sich um ein einfaches Geheimtextalphabet im oben beschriebenen Stil, bei welchem jeder Buchstabe um drei Stellen nach vorne verschoben wird (a wird zu D, b zu E, ... in manchen Publikationen sind andere Beträge zu lesen, das Prinzip bleibt aber das gleiche). Zum anderen ist das teilweise noch heute verwendete ROT-13 Verfahren zu erwähnen, bei welchem die Buchstaben des Geheimtextalphabetes um 13 Stellen verschoben werden (ROT = rotation). ROT-13 wurde/wird vor allem dazu benutzt, um die Suche nach Schlüsselwörtern für nicht eingeweihte Personen zu erschweren. So würde die Suche nach "warez" scheitern, wenn auf der entsprechenden Seite nur der Begriff "JNERM" auftaucht.

"Mach mal, Malcom!"

Malcom betrachtet sich zuerst die in Frage kommenden Schlüssel und entdeckt einen besonders schwachen, welcher den Klartext praktisch durch sich selbst ersetzen würde, oder einfacher ausgedrückt jedem Buchstaben den Betrag 0 hinzuaddiert. Ein vernünftiger Mensch würde solch einen Schlüssel natürlich nicht wählen, aber ein schlechtes Computerprogramm könnte diesen Fehler begehen.
Bei der Analyse der erzeugten Geheimtexte stößt er auf einen interessanten Sachverhalt: In der Sprache gibt es gewisse Regelmäßigkeit in Bezug auf die Häufigkeit einzelner Buchstaben und Buchstabenpaaren. Da jede Nachricht immer 1:1 übertragen wird, finden sich diese Häufigkeiten noch im Geheimtext wieder und einem statistischen Angriff sind Tür und Tor geöffnet.

polyalphabetische Substitution

der Algorithmus

Wie der Name schon vermuten lässt, handelt es sich bei der polyalphabetischen Substitution um eine Weiterentwicklung der monoalphabetischen, mit dem einen Unterschied, dass hier mehrere Geheimtextalphabete (poly - viele) verwendet werden, 7wobei jedes dieser Geheimtextalphabet eindeutig gekennzeichnet werden muss (durch eine Nummer, durch den Anfangsbuchstaben, etc.). Diese Kennzeichnung ist notwendig, da als Schlüssel nicht mehr die einzelnen Geheimtextalphabete ausgetauscht werden müssen, sondern nur noch ein einziges Schlüsselwort, welches die Reihenfolge der Geheimtextalphabete vorgibt (wurden die Alphabete durch Zahlen gekennzeichnet, so wäre ein mögliches Schlüsselwort 1324). Der Algorithmus gibt hier die Geheimtextalphabete vor (wer auf die Einhaltung von Shannons und Kerckhoffs Maxime verzichten will, der kann diese zusätzlich geheim halten) und be-

schreibt den Vorgang der Verschlüsselung, welcher in der Regel wie folgt lautet: "Wähle anhand des ersten Zeichens des Schlüssels das zu verwendende Geheimtextalphabet aus und verschlüssle damit den ersten Klartextbuchstaben. Wähle anhand des zweiten Zeichens des Schlüssels das zu verwendende Geheimtextalphabet aus und verschlüssle damit den zweiten Klartextbuchstaben. usw." Wurde das Schlüsselwort vollständig abgearbeitet, dann wird wieder mit dem ersten Zeichen begonnen. Auf diese Weise wird nun verfahren bis der gesamte Klartext verschlüsselt worden ist.

die liebe Theorie

Ein Angriff über die Häufigkeitsanalyse wird durch dieses Verfahren enorm erschwert, da je nach Anzahl der Geheimtextalphabete und der Beschaffenheit des Schlüssels die tatsächlichen Häufigkeiten nicht mehr mit den relativen übereinstimmen und ein solcher Angriff somit zum Scheitern verurteilt ist. Man nehme einmal an, dass nur zwei Geheimtextalphabete benutzt werden, welche bis auf die Tatsache, dass alle Buchstaben im zweiten um eine Stelle verschoben wurden (e entspricht im ersten Geheimtextalphabet einem F, im zweiten einem G), identisch sind. Kommt in einem Text nun das Wort "eleet" vor (bewusst falsch geschrieben im Hinblick auf den bekannten Ausdruck "31337"), so würde es mit dem Schlüssel "12" nach der Verschlüsselung folgendermaßen aussehen: "FNFGU". Für e wurde hier sowohl F als auch G verwendet, was durch einfaches Abzählen der Buchstaben für eine Häufigkeitsanalyse nicht verwertbare tatsächliche Häufigkeiten ergeben würde.
Neben der Vereitelung des erfolgversprechendsten Angriffes auf diese Art der Verschlüsselung bietet die polyalphabetische Substitution ein um einige Stellen höheres Sicherheitsniveau: Bei 26 Buchstaben (n) für welche nur zwei (m) zufällige Geheimtextalphabete erzeugt wurden, gibt es alleine schon m(n!) mögliche Kombinationen, was in Zahlen ausgedrückt etwa 800.000 Milliarden Milliarden entspricht. Für einen reinen Brute-Force Angriff sieht es sehr schlecht aus, was auch der Grund dafür sein dürfte, dass die bekannteste Vertreterin dieser Verschlüsselungsart, die Vigenère-Verschlüsselung, den Spitznamen "Le Chiffre indéchiffrable" (etwa: die nicht dechiffrierbare Chiffre) trägt.

Alice und Bob

Alice hat die Homepage der PARAllel MINDs Cooperation mittlerweile in ihr Herz geschlossen und als Startseite eingerichtet. Eines Tages entdeckt sie wieder einige neue, interessante Texte im großen InfoDocs-Archiv und möchte diese Information Bob schnellstmöglich zukommen lassen, mit welchem sie sich gerade in einem öffentlichen Chat unterhält. Also benutzt sie die vor einiger Zeit ausgemachte, neue Verschlüsselungsmethode, welche zwei Geheimtextalphabete, ein ROT-3 und ein ROT-6 permutiertes, verwendet. Sie schreibt also die Nachricht "neuetextebeiparamind" verschlüsselt in den Chat. Jeder andere Teilnehmer wird denken, dass sie einfach willenlos auf die Tastatur gedrückt hat, doch Bob kann mit Hilfe der Geheimtextalphabete und ihres vereinbarten Schlüssels "12" das Buchstabengewirr "QKXKWKAZHHHOSGUGPOQJ" entschlüsseln und kommt in den Genuss der neuen Informationen.

Hier die verwendeten Alphabete:

Klar	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
Geheim 1	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
Geheim 2	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F

weitere Varianten

Zusätzlich zu der enormen Vielfalt, die sich alleine schon durch die freie Wählbarkeit der Geheimtextalphabete und der zusätzlichen Möglichkeit unterschiedlicher Schlüssel ergibt, gibt es unzählige weitere Varianten - denkbar wäre beispielsweise eine polyalphabetische Substitution mit homophonen Geheimtextalphabeten, der Alptraum eines jeden Hobbykryptoanalytikers. Jedoch gibt es auch in dieser Klasse wieder ein besonderes, sehr bekanntes historisches Verfahren, welches nach dem Namen seines Schöpfers als Vigenère-Verschlüsselung bekannt geworden ist und auf Grund seines Bekanntheitsgrades und seiner Komplexität eine eigene Seite verdient hat (zu finden bei den ausgewählten Algorithmen auf Seite 40).

"Mach mal, Malcom!"

Zuerst überprüft Malcom den Schlüsselraum auf schwache Schlüssel, wobei ihm sofort drei schwache Schlüsselgruppen auffallen. Zur ersten Gruppe gehören alle ein Zeichen lange Schlüssel, da mit diesen eine polyalphabetische Substitution per definitionem nicht möglich ist. Zur zweiten Gruppe schwacher Schlüssel zählt er alle Exemplare beliebiger Länge, die ausschließlich bzw. übermäßig häufig die gleichen Geheimtextalphabete vorschreiben (z.B. durch den Schlüssel "AAA"), da man als Ergebnis im Extremfall wieder nur eine monoalphabetische Substitution erhält. Und schlussendlich sind Schlüssel mit Wiederholungen und Regelmäßigkeiten zu vermeiden (etwa "ABCABC"), da diese durch ihre Länge ein größeres Sicherheitsniveau vortäuschen, als dies tatsächlich der Fall ist.
Bei Betrachtung des Algorithmus selbst fällt auf, dass der gleiche Schlüssel immer und immer wieder verwendet wird. Bei kurzen Schlüsseln könnte dies zum Verhängnis werden, da auf diese Weise häufig vorkommende, kurze Wörter (wie das deutsche "die") mit dem gleichen Schlüsselteil verschlüsselt werden könnten. Im Idealfall (für den Analytiker) kann man durch diese Regelmäßigkeiten (nicht ausreichende Erzeugung von Konfusion und Diffusion) auf die Schlüssellänge schließen, den Geheimtext in seine Einzelbestandteile zerlegen und eine Erfolg versprechende Häufigkeitsanalyse starten. Dieser Angriff erscheint Malcom auch als der aussichtsreichste und er entscheidet sich, diesen in sein Sortiment wirkungsvoller Angriffsarten aufzunehmen.

Produktchiffren

der Algorithmus

Eine Produktchiffre entsteht durch Kombination von Transpositions- und Substitutionsalgorithmen, welche direkt nacheinander auf den Klartext angewandt werden. So kann ein Text beispielsweise zuerst durch eine monoalphabetische Substitution ver-

schlüsselt werden, wonach die Buchstaben der neu entstandenen Chiffre untereinander vertauscht werden bevor erneut eine Substitution ausgeführt wird.
In der Praxis werden häufig für komplizierte mechanische Verschlüsselungen, die bei näherem Betrachten eine komplexe polyalphabetische Substitution darstellen, der Ausdruck Produktchiffrierer benutzt. Dies ist prinzipiell nicht falsch, kann aber gerade bei Einsteigern zur Verwirrung führen, besonders was die Einordnung verschiedener Verfahren anbelangt.
Die so zustande kommenden Algorithmen sind in der Regel so komplex und unhandlich, dass sie erst mit der Erfindung der entsprechenden (elektro-)mechanischen Pendants größere Verbreitung fanden. Die zwei bekanntesten Beispiele sind die Rotormaschine und, darauf aufbauend, die Enigma.

die liebe Theorie

Man darf nicht den Fehler begehen und annehmen, dass sich das Wort PRODUKTchiffre aus der Mathematik ableitet, denn dann würde das implizieren, dass man die Sicherheit zweier kombinierter Verfahren miteinander multiplizieren und das man die einzelnen Verfahren untereinander vertauschen kann. Beides entspricht nicht der Wahrheit. Vielmehr stammt der Begriff aus der ursprünglichen Bedeutung, nämlich dem Ergebnis eines Prozesses, bei dem mehrere Ausgangsstoffe miteinander kombiniert werden, wobei man am Schluss ein Endprodukt erhält.
Doch in welchem Maße steigert sich nun das Sicherheitsniveau?

Beispiel: zwei monoalphabetische Substitutionen:
Werden zwei Substitutionen hintereinander angewandt, so hat dies vor allem Auswirkungen auf die effektive Schlüssellänge. Das hat folgenden Grund: Im Prinzip wird jeder Buchstabe zweimal verschlüsselt, es finden also jeweils zwei Additionen mit unterschiedlichen Beträgen statt. Die Situation, in der ein Klartextbuchstabe mit einem bereits dagewesenen Betrag abermals verschlüsselt wird tritt jedoch nicht nach dem Ende eines der Schlüssel ein, sondern erst wenn beide Schlüssel gleichzeitig ihr Ende erreicht haben. Die effektive Schlüssellänge entspricht also dem kleinsten, gemeinsamen Vielfachen der verwendeten Schlüssel. Auf diese Weise kann zum Beispiel durch drei Schlüssel mit der Länge Drei, Vier und Fünf eine effektive Schlüssellänge von 60 erreicht werden! Wird dazu noch ein Transpositonselement eingebaut, sind dem durchschnittlichen Angreifer einige Kopfschmerzen sicher. Untenstehende Tabelle verdeutlicht das Prinzip der vergrößerten Schlüssellänge:

c	o	d	e	c	o	d	e	c	o	d	e
r	e	d	r	e	d	r	e	d	r	e	d

Alice und Bob

Nach einem brisanten Bericht auf den Seiten von PM über "Schnüffler in diversen Channels" (Level 2) wird Alice (zu Recht) ein wenig paranoid. Sie überlegt sich, wie sie ihre bisherigen Verschlüsslungsmethoden verbessern könnten. Dazu surft sie ein wenig auf ihrer Lieblingsseite in der Hoffnung einen Geistesblitz zu erfahren. Und siehe da: Beim Anblick des Cybermans in Level2 mit den ausgestreckten Armen kommt ihr die Idee! Ein Schlüsselwort in die rechte, eines in die linke Hand, einmal

kombinieren und fertig ist ein leicht zu merkender, aber dennoch um einiges sicherere Schlüssel.

weitere Varianten

Die wohl bekannteste Vertreterin der Produktchiffriergeräte stellt die Enigma dar. Hierbei handelt es sich grob gesagt um eine erweiterte Rotormaschine, welche von der Wehrmacht im Zweiten Weltkrieg für die Verschlüsselung benutzt wurde, praktischerweise in schreibmaschinenähnlicher Form. Über diese Apparatur ist soviel zu sagen, dass man dies auf einer Seite nicht unterbekommen kann, weshalb bei den Links speziell zu diesem Thema weiterführende Informationen zu finden sind. Wer noch mehr darüber erfahren will findet im Angebot gut sortierter Buchhändler und in der Videothek.

"Mach mal, Malcom!"

Malcom sieht sich jetzt einem echten Problem gegenüber. Prinzipiell kann er auf seine Methoden und Erfahrungen aus den Kategorien Transposition und Substitution zurückgreifen. Er hat jedoch mit einer weitaus höheren Komplexität zu rechnen, was für ihn den Einsatz von Rechenmaschinen fast zwingend voraussetzt.

Codes

der Algorithmus

Codes sind einfach Wörter oder Zeichen, die für andere Wörter stehen. Aus diesem Grund kann man den Algorithmus auch in einem Satz beschreiben: "Ersetze alle im Klartext in Frage kommenden Wörter durch ihre Codewörter." Der Schlüssel stellt in diesem Fall ein so genanntes Codebuch dar, welches die Wort/Codewort-Paare enthält.
Zur Verdeutlichung ein bekanntest Beispiel aus dem Film „23“: Pepe wird gefragt, wann die Reise nach "Paris" losgeht, woraufhin er antwortet, dass sich erst der "Reiseleiter" melden muss. Paris war in diesem Fall das Codewort für Ost-Berlin, der Reiseleiter war der Kontaktmann des KGB.

die liebe Theorie

Mit der richtigen Verwendung von Codes helfen diese bei einer einfachen Unterhaltung, welche belauscht wird, die wahren Ziele und Absichten geheim zu halten. Passend gewählt fügen sie sich außerdem nahtlos in einen Text ein und können so einen Feind auf die falsche Fährte führen.

Alice und Bob

Da Alice und Bob in letzter Zeit zu oft das Missgeschick widerfährt, auf langweilige Parties eingeladen zu werden, vereinbaren sie Codewörter, um auf die umständliche und recht auffällige Verschlüsselung verzichten zu können. Für einen Besuch der ParaMind-Seite benutzt Alice beispielsweise "Ich muss dringend meine wichtige Medizin

nehmen (, die ich zu Hause vergessen habe). Kannst Du mich bitte nach Hause fahren?" Auf diese Weise kennt Bob sofort Alices wahre Absicht und beide haben zeitgleich eine passende und glaubwürdige Ausrede, die Feiernden zu verlassen.

weitere Varianten

Codes gibt es in den unterschiedlichsten Formen, seien das nun Bezeichnungen für irgendwelche Geheimoperationen des Militärs, das Zeichnen des Fisches im frühen Christentum oder das Kratzen am Kopf als Startsignal. Codes sind vielfältig, doch eines ist immer gleich: je unauffälliger sie sind und je logischer sie sich in einen Text oder eine Situation einfügen, desto leichter kann man damit einen potentiellen Angreifer effektiv täuschen.
Eine weiterentwickelte Anwendung der Codewörter stellt der Nomenklator dar, der auf der Algorithmen-Seite zu finden ist.

"Mach mal, Malcom!"

Die größte Schwierigkeit für Malcom ist in der Regel die Entdeckung der Tatsache, dass Codes überhaupt verwendet werden. Hat er dies jedoch erst einmal bemerkt, dann kann er entweder einfach raten, was nur bei schlechten Codes Erfolg versprechend ist, oder er beobachtet die Zusammenhänge zwischen der Kommunikation der beteiligten Parteien und den kurz darauf folgenden Geschehnissen. So kann er durch Beschattung der beteiligten Personen, durch die Analyse des Kommunikationsverkehrs oder schlicht durch das Feststellen von Veränderungen hinter die wahre Bedeutung von Codes kommen.

Block- und Stromchiffrierung

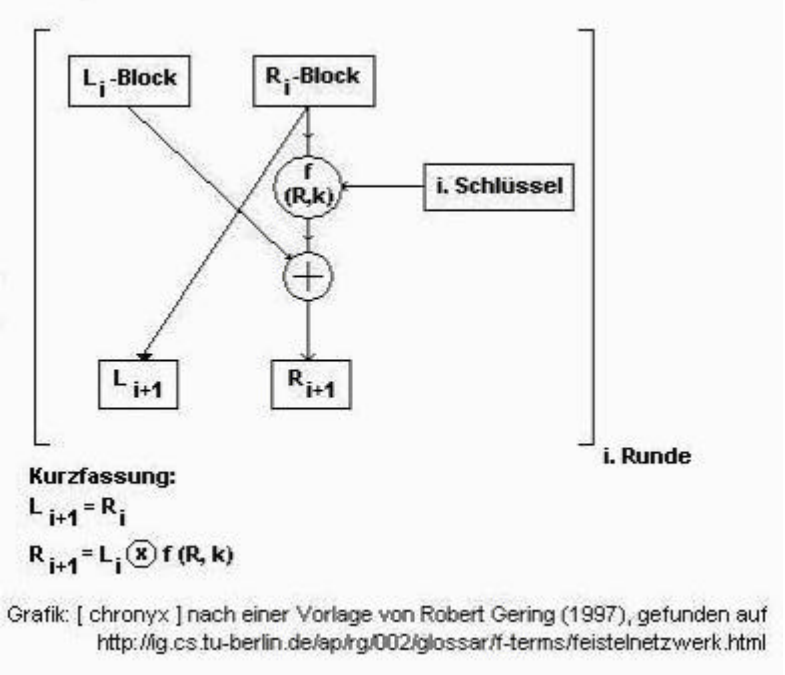
die verschiedenen Arten

Noch einmal zur Erinnerung: Ein Verfahren wird als symmetrisch bezeichnet, wenn sowohl zum Ver- als auch zum Entschlüsseln der gleiche Schlüssel verwendet wird. Der aufmerksame Leser wird feststellen, dass alle bisherigen Verschlüsselungsverfahren genau diese Eigenschaft besitzen und somit das Prädikat symmetrisch verdienen. Die Verfahren, die nun behandelt werden sollen, folgen alle (bzw. fast alle) einer gleichen Grundstruktur, welche die in der Einleitung genannten Grundsätze (Konfusion, Diffusion, Shannons und Kerckhoffs Maxime) möglichst gut erfüllen will. Um das Grundprinzip zu erläutern, gehe ich nicht auf etwaige Besonderheiten und Abweichungen konkreter Verfahren ein; diese werden auf der Algorithmenseite genauer durchleuchtet.
Die moderne symmetrische Kryptografie kennt zwei Hauptgruppen effizienter Verschlüsselungsverfahren. Dies wäre zum einen die Blockchiffrierung (ECB, CBC, CFB, OFB) und zum anderen die Stromchiffrierung, welche jeweils unterschiedliche Vor- und Nachteile besitzen und für die verschiedensten Situationen konzipiert worden sind.

Blockchiffrierung

Bisher wurde ein Klartext immer Zeichen für Zeichen in den Geheimtext umgewandelt. Viel schwieriger wäre es für einen Kryptoanalytiker, wenn der Klartext blockweise, also immer ganze Textpassagen auf einmal, verschlüsselt werden würde und genau dies wird bei der Blockverschlüsselung gemacht. Der Klartext wird zu Beginn der Verschlüsselung in einzelne, jeweils gleichlange Blöcke aufgeteilt, welche anschließend unter Einwirkung des Schlüssels einfachen mathematischen Operationen unterzogen werden (wie Additionen, Multiplikationen, XOR-Verknüpfungen und einfachen Substitutionen). Eine solche Operationenfolge wird auch als Runde bzw. Halbrunde bezeichnet.
Ist das kryptografische Verfahren so konzipiert worden, dass es mehrere Runden pro Block vorschreibt, dann spricht man auch von iterativer Blockverschlüsselung. Durch das mehrmalige Ausführen der gleichen Runden pro Block wird nach einer gewissen Anzahl (Abhängig vom Verfahren) der gern gesehene Effekt erzielt, dass jedes Klartextbit vom Schlüssel abhängig ist, was konkret bedeutet, dass die kleinste Änderung in der Originalnachricht zu einem komplett anderen Geheimtextblock führen würde. Dieser Effekt, der dazu führt, dass sich jede winzige Änderung mit jeder weiteren Runde im Geheimtextblock immer stärker fortpflanzt, nennt man passenderweise Lawinen-Effekt (auch Avalanche-Effekt).

Eine spezielle Konstruktionsweise iterativer Blockchiffren stellt das ***Feistel-Netzwerk*** dar. Dieses folgt im Großen und Ganzen den oben beschriebenen Grundsätzen, jedoch werden die Klartextblöcke nochmals in eine rechte und in eine linke Hälfte zerlegt, R und L. In jeder Runde wird ausschließlich die R-Hälfte den mathematischen Operationen unterzogen, unter Einwirkung des Schlüssels und der L-Hälfte, welche dabei unverändert bleibt. Die L-Hälfte dient dabei einem bestimmten Zweck, nämlich der XOR-Verknüpfung mit der R-Hälfte. Wenn die Runde vollständig durchlaufen und der R-Block komplett verschlüsselt wurde, dient für die nächste Runde als Eingabeblocks die vorherige unbearbeitete R-Hälfte als L-Hälfte und das Ergebnis der vorherigen Runde als R-Hälfte. Wurde nun die vorgeschriebene Anzahl Runden ausgeführt, dann wird der nächste Klartextblock herangezogen. Auf diese Art und Weise wird mit dem kompletten Klartext verfahren, bis er vollständig verschlüsselt worden ist.
Ein großer Vorteil des Feistel-Netzwerkes und damit auch ein Grund für seine große Verbreitung ist die Tatsache, dass durch die Konstruktion schon wenige Runden ausreichen, um einen Textblock komplett vom Schlüssel abhängig zu machen.



Grafik: [chronyx] nach einer Vorlage von Robert Gering (1997), gefunden auf http://ig.cs.tu-berlin.de/ap/rg/002/glossar/f-terms/feistelnetzwerk.html

Aus diesen Grundprinzipen der Blockchiffrierungen haben sich in der heutigen Zeit vier wichtige Spezialformen entwickelt, welche jeweils andere Vorteile bieten (wobei gleich im Vorfeld zu sagen ist, dass sich die letzten drei Verfahrensarten sehr ähnlich sind). Zur Verschlüsselung wird jeweils die allgemeine Verschlüsselungsfunktion E(x) verwendet - ob es sich hierbei um ganz einfache XOR-Verknüpfungen handelt, oder um hochkomplizierte Abfolgen, spielt dabei keine Rolle.

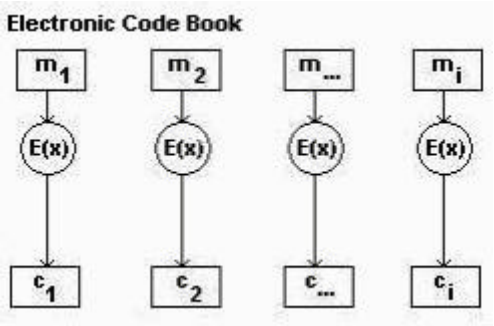
Die vier Modi sind:

- Electronic Code Book (ECB)
- Cipher Block Chaining (CBC)
- Cipher Feedback (CFB)
- Output Feedback (OFB)

Electronic Code Book (ECB):

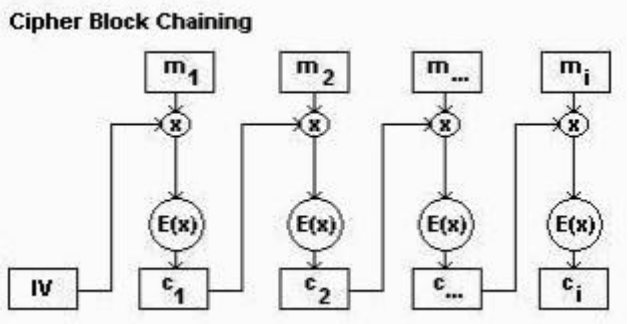
Seinen Namen hat der Electronic Code Book-Modus auf Grund der Tatsache, dass jedem Klartextblock genau ein einzigartiger Geheimtextblock zugeordnet werden kann, welche theoretisch alle in einem Buch niedergeschrieben werden könnten (die Betonung liegt hier auf theoretisch, da schon bei einem 64 Bit langen Block 2^{64} mögliche 64 Bit lange Texte existieren, welche insgesamt etwa 39,6 x 10^{27} Gigabyte Speicherplatz benötigen würden). Will man einen Block verschlüsseln, so müsste man diesen in dem Buch suchen und den dazu passenden Geheimtext niederschreiben.
Im Großen und Ganzen folgt der ECB-Modus den oben beschriebenen Prinzipien der Blockchiffrierung. Das besonders charakteristische ist die Tatsache, dass jeder Klartextblock einzeln behandelt wird, er also von den anderen Blöcken der Nachricht abgeschottet ist und keinerlei Wechselwirkung mit diesen hat, auch wenn jeder dieser einzelnen Blöcke die gleichen mathematischen Operationen bzw. Operationenfolgen durchläuft.
Bei diesem Modus hat man den entscheidenden Vorteil auf einzelne verschlüsselte Elemente zugreifen zu können, ohne die anderen berücksichtigen zu müssen. Dies ist besonders bei verschlüsselten, dynamischen Datenbanken eine unbedingt notwendige Eigenschaft, da es so möglich ist, gezielt Datenblöcke zu verändern, ohne gleich

die komplette Datenbank entschlüsseln zu müssen. Doch da liegt auch die Schwäche diese Modus, denn so wird beispielsweise der "Block Replay Angriff" ermöglicht (mehrmaliges Senden einzelner Datenblöcke, z.B. Überweisungen). Auch ist die Gefahr von Codebuch-Analysen gegeben, wobei diese bei einer großen Blocklänge und dem Vorhandensein des Lawinen-Effektes praktisch unmöglich werden.
Mathematisch kann man den ECB-Modus folgendermaßen darstellen: $c_i = E(m_i, k)$



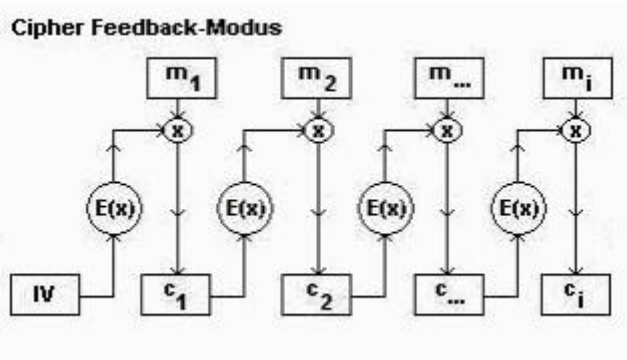
Cipher Block Chaining (CBC):

Der Cipher Block Chaining-Modus merzt die Schwächen (aber gleichzeitig auch die Vorteile) des ECB-Modus aus: Hier wird jeder Klartextblock vor seiner Verschlüsselung mit dem vorhergehenden Geheimtextblock verknüpft (z.B. durch XOR). Beim ersten Block wird für diesen Zweck ein so genannter Initialisierungsvektor (IV) benutzt. Mathematisch dargestellt sieht das Verfahren so aus: $c_i = E(m_i \times c_{i-1}, k)$. Wenn das Avalanche-Kriterium erfüllt ist, dann pflanzt sich in diesem Modus jede noch so kleine Änderung im Klartext nicht nur lawinenartig durch den entsprechenden Block fort, sondern durch alle nachfolgenden Blöcke auch. Ein Block Replay-Angriff scheidet nun auch aus, da man die ganze Nachricht senden müsste und sich keine Blocks gezielt herauspicken kann.
Es ist zwar noch möglich einen einzelnen Block einsehen zu können (vorausgesetzt man hat auf den vorhergehenden Block Zugriff), jedoch kann man diesen nicht mehr verändern ohne die gesamten nachfolgenden Blöcke auch verändern zu müssen - ein mögliches Einsatzgebiet stellen somit statische Datenbanken dar.



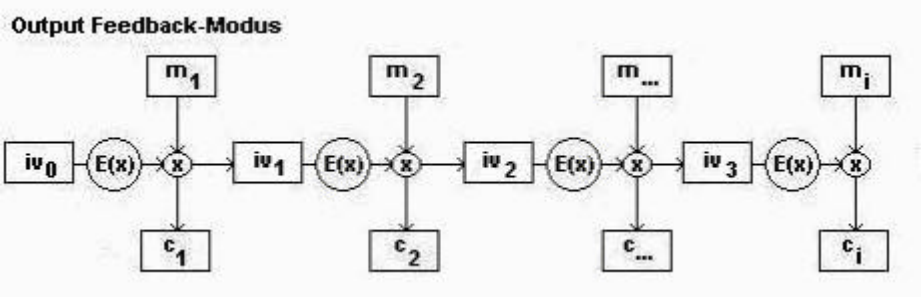
Cipher Feedback (CFB):

Beim Cipher Feedback-Modus wird der vorhergehende Geheimtextblock verschlüsselt (beim ersten Block ist auch hier wieder ein Initialisierungsvektor IV nötig) und das Ergebnis mit dem Klartextblock verknüpft. Kurz und prägnant sieht das so aus: $c_i = E(c_{i-1}) \times m_i$



Output Feedback (OFB):

Beim Output Feedback-Modus wird am Anfang wieder ein Initialisierungsvektor (IV) generiert, welcher verschlüsselt wird. Dieses Ergebnis wird nun mit dem ersten Klartextblock verknüpft und als Ergebnis erhält man den ersten Geheimtextblock - bis jetzt ist dieses Verfahren mit dem CFB-Modus identisch. Beim nächsten Klartextblock wird jedoch nicht der vorhergegangene Chiffrenblock herangezogen, sondern der verschlüsselte Initialisierungsvektor (E(iv, k) - man verzeihe mir die Tatsache, dass ich für eine Variable zwei Buchstaben verwende) wird nochmals verschlüsselt - E(E(iv, k), k) - und anschließend zur Verknüpfung herangezogen. Beim OFB-Modus ist der Block, welcher zur Verknüpfung mit dem Klartext verwendet wird, sowohl von dem Klartext selbst als auch vom erzeugten Geheimtext vollständig unabhängig. In Formeln kann man dies abschließend mit folgenden zwei Gleichungen ausdrücken: $c_i = m_i \times iv_i$ und $iv_i = E(iv_{i-1}, k)$



Stromchiffrierung

Die Stromchiffrierung findet seltener bei der Verschlüsselung gespeicherter Daten Verwendung, sondern vielmehr bei der Verschlüsselung eines kontinuierlichen Datenstroms, wie es beispielsweise bei einem Echtzeitchat der Fall ist. Es wird hier nicht gewartet bis eine große Menge Datenmaterial vorhanden ist welche man blockweise verschlüsseln kann, sondern die Daten werden in viel kleineren Größenordnungen (in der Regel Bits) sofort verschlüsselt und versendet.
Um eine sichere Verschlüsselung mit diesen kleinen Datenmengen zu ermöglichen, kann ein vom Schlüssel abhängiger Pseudo-Zufallszahlen-Generator verwendet werden, der ebenfalls einen kontinuierlichen Strom, in diesem Fall aber einen Schlüsselstrom, erzeugt (der Algorithmus ist bei diesem Fall offensichtlich an das One-Time-Pad angelehnt). Beim Pseudo-Zufallszahlen-Generator handelt es sich, wie der Name schon sagt, nicht um einen echten Zufallszahlen-Generator, da die von ihm erzeugten Zahlen von einer gewissen Information abhängig sind: dem Schlüssel. Dies ist

auch notwendig und logisch, denn sowohl der Sender als auch der Empfänger müssen die gleichen Zufallszahlen zur Verfügung haben. Der erzeugte Schlüsselstrom wird nun mit dem Datenstrom verknüpft (je nach Verfahren bzw. Einstellung unterschiedlich) und an den Empfänger weitergeleitet.
Die Art der Verknüpfung kann (unabhängig davon, ob ein Schlüsselstromgenerator verwendet wird oder statische Schlüssel) wiederum einem der vier oben vorgestellten Betriebsmodi folgen, wobei eventuell kleinere Änderungen notwendig sein könnten, um sie auf die Stromchiffren übertragen zu können.

Alice und Bob

Alice und Bob sind seit einiger Zeit Stammbesucher der PARAllel MINDs Cooperation und von deren Zielen und Idealen überzeugt. Aus diesem Grund haben beide beschlossen, ParaMind ihre Unterstützung anzubieten und als Agents aktiv mitzuarbeiten. Da ihre Arbeiten jedoch sensible Informationen enthalten, welche sie nur mit einer starken Verschlüsselung durch das Internet schicken wollen, entscheiden sie sich für ein symmetrisches Blockverfahren. Und wer weiß, vielleicht sind beide ja Hauptbestandteil der nächsten PM-News …

weitere Varianten

Wie eingangs schon erwähnt sind obige Algorithmen die allgemeine Theorie; als modifizierte Variante wird nachfolgend bei den bekannten Algorithmen ein bekanntes Verfahren vorgestellt:

- Data Encryption Standard (DES), ECB-Modus

„Mach mal, Malcom!“

Malcom sieht sich jetzt seiner bisher schwierigsten Aufgabe gegenüber, da diese Verschlüsselungsarten alle sehr sicher sind und mehrfach geprüft wurden. Da es sich aber, wie gesagt, nur um Grundprinzipien handelt, kann er noch keine konkreten und Erfolg versprechende Analysemethoden aufstellen, sondern muss sich vorerst auf die Theorie beschränken. Konkrete Beispiele gibt es dann, falls verfügbar, bei den Verfahren selbst.
Eine Schlüsselraumanalyse scheidet vorerst aus, muss jedoch dringend vorgenommen werden, da schwache Schlüssel ein enormes Sicherheitsrisiko darstellen und nicht selten sind. Ein Brute-Force Angriff ist jederzeit möglich, aber für sich alleine genommen wenig aussichtsreich. Eine wie in dem Text über Häufigkeitsanalysen erläuterte Vorgehensweise wird wohl scheitern, da die Prinzipien von Konfusion und Diffusion normalerweise eingehalten werden (ein Versuch schadet dennoch nicht). Bleibt noch die Analyse des konkreten Verfahrens, was am meisten Erfolgschancen haben dürfte. Selbst wenn man nur auf kleinere Fehler stößt, können diese eventuell einen Brute-Force Angriff erheblich erleichtern, was den Kryptoanalytiker einen (Riesen-)Schritt näher an sein Ziel bringen würde.

Quantenkryptografie

der Algorithmus

Die Quantenkryptografie beruht auf einem Gebiet der Physik, welches man mit gutem Gewissen zu einem der schwersten zählen kann. Ein berühmtes Zitat wird im Zusammenhang mit der Quantenphysik und dem Verständnis derselben gerne angeführt: "Wenn einer sagt, die Quantenphysik bereite ihm kein Kopfzerbrechen, dann hat er sie nicht verstanden." Da die Quantenphysik ein so schwer zu begreifendes Thema ist, wird an dieser Stelle erst gar nicht der Versuch unternommen, dieses ausführlich darzustellen. Wer jedoch näher an der Materie interessiert ist findet in auf der PM-Homepage weiterführende Informationen.
Da man für das Verständnis dieses Dokuments dennoch einige grundlegende Informationen zu diesem spannenden Thema benötigt, werden diese hier zusätzlich kurz erläutert, ohne zu sehr ins Detail zu gehen. Bei der Quantenphysik werden in der Regel Photonen betrachtet, die alle eine bestimmte Polarisation, einen bestimmten Spin (Drehung) haben. Wir beschränken uns hier auf die Annahme, dass es vier verschiedene Spins gibt: senkrecht, waagerecht, diagonal nach rechts, diagonal nach links und verwenden zur Darstellung die Symbole |, -, /, \. Es ist zwar möglich, Photonen mit einer bestimmten Polarisation zu erzeugen, diese aber einwandfrei zu messen ist nicht ohne weiteres machbar.
Doch zuerst ist einmal die Frage zu klären, wie man die Polarisation überhaupt misst. Man benötigt dazu einen Filter, welcher nur eine bestimmte Polarisation durchlässt - für senkrecht polarisierte Photonen folglich einen senkrechten Schlitz. Wird ein Photon darauf abgefeuert und erzeugt auf der anderen Seite einen messbaren Einschlag, dann war der Filter wohl der richtige und man kennt nun die Polarisation. Diese Sache hat aber noch einen Haken: Gehen wir von /-polarisierten Photonen aus, die auf einen |-Filter treffen. Normalerweise dürfte auf der anderen Seite des Filters logischerweise nichts zu messen sein, da die Ausrichtungen nicht übereinstimmen, doch da die Quantenphysik nicht immer viel mit Logik zu tun hat, passieren dennoch ca. 50% der erzeugten Photonen den Filter und sind auf der anderen Seite messbar. Daraus kann man schließen, dass sich in etwa die Hälfte der Photonen dem Filter anpassen und ihre Polarisation folglich ändern.
Wenn man vier Photonen gleicher Polarisation auf einen ungeeigneten Filter schießt, kann man auf der anderen Seite zwei Einschläge messen; die Hälfte der als Treffer ausgewerteten Messergebnisse sind somit falsch.
Diesen Umstand der nicht einwandfreien Messbarkeit kann man nun für die Übermittlung einer geheimen Nachricht, genauer gesagt für die unkomplizierte und absolut abhörsichere Vereinbarung eines 100%ig sicheren One-Time-Pads nutzen. Dazu benutzt man die Photonen um einzelne Bits darzustellen. Die senkrechten und die diagonal-rechts polarisierten Photonen stellen die binäre Eins dar, die waagerechten und die diagonal-links polarisierten die Null. Um die Polarisation zu messen verwendet man zwei verschiedene Schemata, ein rektilineares (+, Kombination aus senkrecht und waagerecht) und ein diagonales Schema (X, Kombination aus beiden diagonalen Polarisationen). Der Sender, Alice, sendet nun einzelne Photonen mit einer zufällig gewählten Polarisation durch die entsprechenden Filter. Der Empfänger, Bob, wählt ebenfalls zufällig seine Filter für die Messung aus und merkt sich sowohl den verwendeten Filter als auch das Messergebnis (1 oder 0). Nun teilen sich beide mit, wel-

che Filter sie benutzt haben und vergleichen diese miteinander. Da dies nicht die Messwerte (sprich 1 oder 0) offen legt, kann dies über jeden beliebigen, also auch über einen unsicheren Kommunikationskanal erfolgen. Beim Vergleichen der verwendeten Messfilter streichen sie nun alle Ergebnisse weg, bei denen nicht die gleichen Filter verwendet worden sind, da bei diesen (wir erinnern uns an den oberen Abschnitt) Messfehler aufgetreten sein könnten. Bei den verbleibenden Messwerten können sie sich jedoch absolut sicher sein, dass diese korrekt sind, da die gleichen Filter verwendet worden sind. Das bedeutet, dass beide nun eine beliebig lange und zufällige Reihe von Einsen und Nullen zur Verfügung haben, welche sie als One-Time-Pad verwenden können.
Zusätzlich ist diese Art der Schlüsselvereinbarung gegen passive Angriffe geschützt, was in der Kryptografie ein Novum darstellt. Würde jemand die Kommunikation zu belauschen versuchen, dann würde er zwangsläufig einige Polarisationen verändern, was Alice und Bob auffallen würde.

die liebe Theorie

Wie oben schon gesagt will ich an dieser Stelle nicht näher auf die Quantentheorie eingehen, da dies ein eigenständiges Thema ist. Auch zum Thema One-Time-Pads ist an dieser Stelle nichts mehr zu sagen, da dieses in dem gleichnamigen Text auf der Algorithmenseite weiter hinten in dieser Publikation beschrieben wird.

Alice und Bob

Alice möchte Bob eine absolut vertrauliche und hochbrisante Nachricht schicken, für welche ihr die bisher verwendeten Verfahren zu unsicher sind. Also entscheiden sich beide für die Quantenkryptografie. Über Glasfaserkabel schickt Alice an Bob folgende Photonen, welche Bob mit dem jeweils angegeben Filterschema misst:

Alice sendet	1	1	1	0	0	0	1	1	0
Alice's Schema:	+	X	X	+	X	+	X	+	X
Polarisation	|	/	/	-	\	-	/	|	\
Bob's Filter	+	X	+	+	X	X	X	+	X
Bob misst	1	1	1	0	0	1	1	1	0
richtiger Filter	J	J	N	J	J	N	J	J	J

Somit erhalten beide folgende Bitfolge, welche sie für ein One-Time-Pad benutzen können: 1100110 - sollte die Nachricht länger sein, dann kann man dieses Spiel theoretisch beliebig lange fortführen.
Malcom hat nun ein großes Problem, für das es mit ziemlich hoher Wahrscheinlichkeit keine Lösung geben wird. Wenn er sich nun zwischen beide Partner klemmen würde und versucht, die Kommunikation zu belauschen, dann würde er damit bewiesenermaßen zwangsläufig die Polarisationen verändern, was Alice und Bob auf jeden Fall auffallen würde. Es bleibt ihm nur noch die Möglichkeit einen Angriff auf das One-Time-Pad zu starten, was in etwa genauso erfolgreich sein wird …

weitere Varianten

Die Quantenphysik bietet noch weitere Einsatzmöglichkeiten, z.B. kann man mit ihr eine Art Wasserzeichen erzeugen (die Grundlage für diese Idee stammt von Stephen Wiesner, der das Quantengeld erfunden hat und ist in Simon Singhs Buch "Geheime Botschaften" genauer beschrieben). Da dieses Gebiet der Physik noch nicht vollständig erforscht worden ist, kann man in der Zukunft bestimmt mit weiteren überraschenden Anwendungen rechnen.

„Mach mal, Malcom!“

Malcom hat nun Pech gehabt. Er hat keine Möglichkeit dieses kryptografische Verfahren zu brechen und muss sich nun geschlagen geben. Oder gibt es doch eine Möglichkeit ...?

Hintergrund: asymmetrische Algorithmen

Was sind asymmetrische Algorithmen?

Ein Verfahren wird als asymmetrisch bezeichnet, wenn es sowohl für die Ver- als auch für die Entschlüsselung einen anderen Schlüssel verwendet. Diese recht modernen Algorithmen beruhen meist auf mathematischen Problemen welche erlauben, dass ein Schlüssel (public key, k_e) öffentlich frei verfügbar sein darf, während der andere (private key, k_d) geheim gehalten werden muss. Ersterer dient bei der so genannten Public-Key-Verschlüsselung ausschließlich zum Ver-, letzterer ausschließlich zum Entschlüsseln.

Vorteile

Die Möglichkeit der freien Verfügbarkeit des Chiffrierschlüssels löst mit einem Schlag das Problem des sicheren Schlüsselaustausches, welches jahrhundertelang für Kopfzerbrechen gesorgt hatte. Kommunikationspartner müssen ihre Chiffrierschlüssel nicht mehr mühselig und unter großem Risiko auf einem möglichst sicheren Weg austauschen, sondern können sich diese über jeden beliebigen Weg zukommen lassen. Doch nicht nur das Schlüsselaustauschproblem wurde gelöst; gleichzeitig wird das Schlüsselmanagement im Vergleich zu symmetrischen Verfahren deutlich erleichtert: Man braucht nicht mehr, wie es bei der symmetrischen Verschlüsselung nötig war, für jedes "Kommunikationspärchen" einen geheimen Schlüssel (n(n-1)/2), sondern für die gesamte Kommunikation nur die zwei eigenen plus jeweils einen weiteren öffentlichen Schlüssel pro zusätzlichen Kommunikationsteilnehmer (2+n-1).
Neben den uralten Schlüsselproblemen hat sich mit den asymmetrischen Verfahren ein weiteres wichtiges Feld eröffnet, nämlich das der digitalen Signaturen, welche in einem gesonderten Abschnitt erläutert werden.

Nachteile

Doch bei all den Vorteilen kommen auch die Nachteile nicht zu kurz, welche den ausschließlichen Gebrauch von asymmetrischen Verfahren verwehren. So sind asymmetrische Algorithmen vergleichsweise langsam, was sich besonders bei großen Datenmengen negativ auswirkt. Zusätzlich ist der erzeugte Chiffrentext deutlich größer als das Original - bei einigen Bytes macht dies nichts aus, bei mehreren Gigabytes hingegen sind die Auswirkungen gravierend.
Auch darf nicht vergessen werden, dass asymmetrische Chiffrieralgorithmen in der Regel auf mathematischen Problemen basieren, für welche es noch keine Lösung gibt und aus diesem Grund als ausreichend sicher gelten. Sollte jedoch eines Tages ein Lösungsweg entdeckt werden, so werden auf einen Schlag alle entsprechenden Verschlüsselungen zunichte gemacht.
Bezüglich der hardwareseitigen Implementierung von kryptografischen Elementen schneiden asymmetrische Algorithmen ebenfalls schlecht ab, da einfache XOR-Verknüpfungen im direkten Vergleich mit Potenzen leichter zu realisieren sind und darum den symmetrischen Verfahren der Vortritt gelassen wird.

Public-Key Kryptografie

der Algorithmus

Wie schon im vorhergehenden Text beschrieben wurde, gibt es bei asymmetrischen Verfahren zwei verschiedene Schlüssel, einem zum Ver- und einem zum Entschlüsseln. Dies funktioniert, da der private Schlüssel Informationen enthält, mit welchen man die mit dem öffentlichen Schlüssel durchgeführten Operationen rückgängig machen kann. Jetzt bleibt nur noch die Frage zu klären, mit welchen Mitteln man diese Eigenschaft in kryptografische Verfahren einbauen kann.
Eine der wohl geläufigsten und bekanntesten Funktion ist aus der Restklassenarithmetik entliehen und vielen Programmierern mehr als geläufig. Es handelt sich dabei um die modulo-Funktion, auch als mod(x) bekannt. Diese Funktion führt eine ganzzahlige Division durch, liefert als Ergebnis aber nicht den Quotienten, sondern den Rest (wie in der Grundschule üblich). Einige kleine Rechenbeispiele zur Verdeutlichung in der Pascal-Schreibweise:
7 mod 2 = 1 | 18 mod 4 = 2 | 12 mod 3 = 0 | 89 mod 11 = 1

Diese Aufgaben sind leicht zu lösen, aber in der Regel unmöglich umzukehren, wenn nicht alle Ausgangszahlen bekannt sind. So könnte beim ersten Beispiel als Ausgangszahl auch die 1, die 3, die 5, die 9, usw. gestanden haben - das Ergebnis wäre immer die 1. Alleine mit dieser Funktion lässt sich noch kein Public-Key-Algorithmus realisieren, aber das Grundprinzip wird sehr schön deutlich. Wie ein entsprechender Algorithmus aussehen könnte, wird auf der Algorithmenseite erklärt.

die liebe Theorie

Je nach verwendetem Verfahren gibt es eine andere Theorie, auf welcher der entsprechende Algorithmus basiert. Daher wird erst bei den konkreten Beispielen auf die theoretischen Grundlagen eingegangen.
Allen Algorithmen gemein sind jedoch die Tatsachen, dass asymmetrische Verfahren im direkten Vergleich zu symmetrischen einen größeren Schlüssel benötigen, um ein vergleichbares Maß an Sicherheit gewährleisten zu können.

Alice und Bob

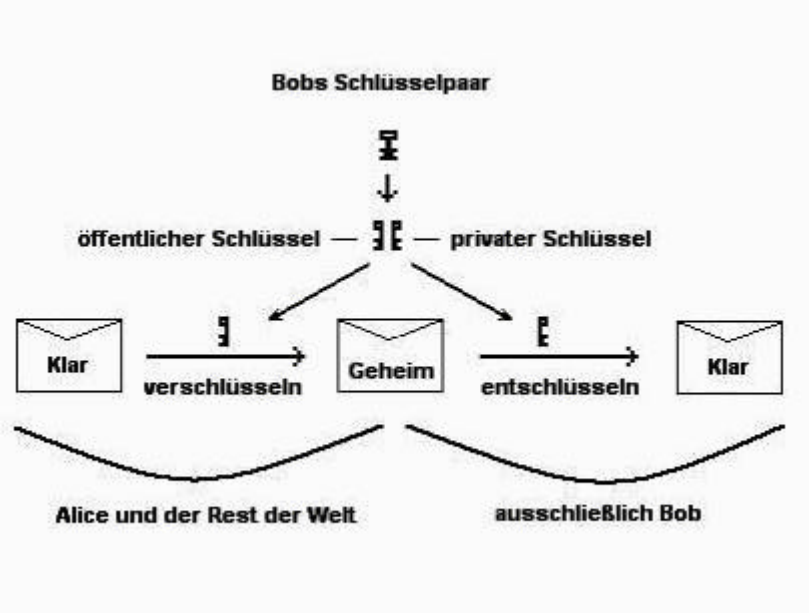
Nachdem Alice und Bob das Grundprinzip asymmetrischer Verschlüsselung durchschaut haben, wenden sie sich dem Schlüsselaustauschproblem zu. Eigentlich dürfen sie überhaupt nicht von einem Problem sprechen, da es wirklich einfacher denn je ist, die zur sicheren Kommunikation notwendigen Schlüssel auszutauschen:

einmalig auszuführende Aktion
1) Bob generiert ein Schlüsselpaar (öffentlicher und privater Schlüssel, erledigt in der Regel ein Programm)
2) und veröffentlicht seinen öffentlichen Schlüssel (z.B. im Internet als einfache ASCII-Datei), behält den privaten jedoch geheim.
3) Alice besorgt sich auf einem beliebigen Wege (z.B. über das In-

ternet) Bobs öffentlichen Schlüssel und fügt diesen ihrem Schlüsselbund hinzu - dieser ist so lange gültig, bis Bob sein Schlüsselpaar wechselt.
für jede Kommunikation auszuführende Aktion
Alice schreibt zuerst ganz normal ihre Nachricht an Bob
und verschlüsselt diese anschließend mit Bobs öffentlichem Schlüssel.
Nun kann diese nur noch von Bob entschlüsselt werden - noch nicht einmal die Senderin ist in der Lage der Nachricht noch etwas hinzuzufügen geschweige denn zu verändern.
Bob empfängt den Geheimtext und entschlüsselt ihn mit seinem privaten Schlüssel.

Alice und Bob verwenden für diese Art der Kommunikation das bekannte Programm PGP. Außerdem laden sie sich die öffentlichen Schlüssel der PMler aus dem Internet, um eine sichere Kommunikation auch mit diesen zu gewährleisten.

Schematisch dargestellt sieht der Vorgang des eigentlichen Nachrichtenaustausches folgendermaßen aus:



weitere Varianten

Bei den bekannten Algorithmen ist für die asymmetrischen Algorithmen ebenfalls ein (bzw. zwei) konkretes Beispiel zu finden:

- Diffie-Hellman (sichere Schlüsselvereinbarung)
- RSA (Faktorisierung großer Primzahlprodukte)

„Mach mal, Malcom!“

Malcom hat nun neue Kryptografieverfahren, an welchen er sich die Zähne ausbeißen kann. Da die verwendete Mathematik allgemein zugängliches Gedankengut ist, kann

er sich schnell in die Materie einlesen und merkt sehr bald, dass die momentane Sachlage einer Ironie nicht unähnlich ist: Er hat theoretisch alle Informationen zur Verfügung, die er zum Knacken der Chiffre benötigt, jedoch bräuchte er zum Berechnen der notwendigen Zahlen länger, als ein normaler Mensch sich ausmalen kann. Wie er sich vielleicht doch dem ersehnten Klartext nähern kann oder warum er angeblich alle benötigten Informationen bereits zur Verfügung hat, wird bei den entsprechenden Verfahren genauer erläutert, da man hier - wie bei den meisten allgemeinen Erklärung eines kryptografischen Algorithmus - keine konkreten Ansätze zum Brechen desselben liefern kann.
Jedoch gibt es einige allgemeine Regeln und Ansätze, welche man hier durchaus erwähnen sollte: Den Schlüssel betreffend muss natürlich sichergestellt sein, dass dieser für seinen Zweck ausreichend lang ist. Malcom wird am Anfang sicherlich einen eingeschränkten und damit schnell erledigten Brute-Force Angriff starten, um die relevanten kurzen Schlüssel zu überprüfen. Danach wird er sich wohl auf den speziell verwendeten Algorithmus stützen müssen, wobei ein Chosen Ciphertext Angriff am aussichtsreichsten sein dürfte.

Hintergrund: digitale Signaturen

Was sind digitale Signaturen?

Eine Signatur ist ein Merkmal, anhand dessen man eine Person oder ein Objekt erkennen kann, optimalerweise mit einer möglichst hohen Genauigkeit und einer möglichst kleinen Irrtumswahrscheinlichkeit. Ist die Fehlerquote klein genug, so dass die Signatur als praktisch sicher (siehe dazu Einführung in die Kryptografie) eingestuft werden kann, dann steht einer Benutzung zu Identifikationszwecken nichts mehr im Wege. Bestes Beispiel aus unserem Leben ist der Fingerabdruck: Jeder Mensch (so sagt man zumindest) hat einen einmaligen Fingerabdruck, also kann man mit Kenntnis von diesem sehr genau sagen, ob ein gefundener Abdruck von dieser bestimmten Person stammt oder nicht.
Eindeutigkeit ist eine wünschenswerte Eigenschaft einer digitalen Signatur. Eine weitere ist die leichte und unkomplizierte Übertragbarkeit auf ein beliebiges anderes Objekt. Bekanntestes Beispiel ist hierfür die altbewährte Unterschrift: Sie ist nicht nur von Mensch zu Mensch verschieden, sie kann auch beliebig oft auf andere Objekte, z.B. auf die letzte Zeile eines Briefes, übertragen werden. Dadurch bekommt das Objekt die persönliche Note des Unterzeichnenden aufgedrückt, wodurch dieses als "von ihm kommend" gebranntmarkt wird. Somit kann der Signaturträger zum einen für den Inhalt des Unterschriebenen (falls es sich um einen Vertrag handelt) verbindlich verantwortlich gemacht werden, zum anderen kann man dem Inhalt das gleiche Maß an Vertrauen entgegen bringen wie dem Unterzeichner gegenüber selbst, da er mit seiner Unterschrift persönlich dafür einsteht.
Zusammenfassend kann gesagt werden, dass eine digitale Signatur vor allem zwei Eigenschaften aufweisen muss: Eindeutigkeit und Übertragbarkeit, so dass sie sowohl zur Identifikation als auch zur Authentifizierung verwendet werden kann.
Von digitalen Signaturen spricht man, wenn diese in digitaler Form vorliegen und in der digitalen Welt benutzt werden, beispielsweise im Internet als digitale Unterschrift unter einem digitalen Kaufvertrag.

Vorteile

Digitale Signaturen erlauben die gleichen Vorteile in der Welt der Bits und Bytes, wie sie in der Welt außerhalb der pulsierenden Leitungen schon lange vorhanden sind. Verbindliche Geschäftsabschlüsse sind genauso gut denkbar wie die eindeutige Kennzeichnung selbst angefertigter, digitaler Arbeiten - aber auch die sichere Authentifizierung einer Person im elektronischen Schriftverkehr ist nun machbar. Dazu kommt noch die Tatsache, dass eine digitale Signatur bei weitem schwerer zu fälschen ist, als es eine krakelige Unterschrift je sein kann, da diese eindeutig sein muss und nicht wie bei manchen Zeitgenossen jedes Mal wieder ein wenig anders aussieht. Kurz, digitale Signaturen haben das Potential, die Welt ein Stückchen sicherer und vertrauenswürdiger zu machen.

Nachteile

Wie auch normale Unterschriften kann die digitale Variante ebenfalls gefälscht werden, was bei richtiger Verwendung aber weitaus schwerer ist. Viel gefährlicher und

realistischer ist der Diebstahl der notwendigen Informationen, mit welchen man die Signatur erzeugen kann. Doch nicht nur diese Situation wäre fatal, alleine schon der Verlust der Signatur wäre ein großer Schaden - wie macht man Personen klar, mit welchen nie ein persönlicher Kontakt zustande kommen wird, dass man wirklich derjenige ist, für den man sich ausgibt, wenn die digitale Kennzeichnung den Bach heruntergegangen ist?
Neben diesen Standardschwachstellen gibt es noch ein organisatorisches Hindernis, welches aber überwindbar ist: Es muss eine vertrauenswürdige Infrastruktur geschaffen werden, über welche man sicher die zur Verifikation der Signatur notwendigen Daten austauschen kann.

Funktionsweise digitaler Signaturen

der Algorithmus

Um digitale Signaturen realisieren zu können, steht man im Wesentlichen vor folgendem Problem: Wie kann meine Signatur von jeder beliebigen Person eindeutig als die meine erkannt werden, ohne vertrauliche Informationen öffentlich preiszugeben? Die Anforderungen an das Signaturverfahren ähneln sehr stark der asymmetrischen Kryptografie. Und in der Tat können Werkzeuge der asymmetrischen Kryptografie eingesetzt werden; so ist es möglich eine Nachricht mit dem eigenen, privaten Schlüssel zu verschlüsseln (sonst wird er zum Entschlüsseln benutzt), welche ausschließlich mit dem dazugehörenden öffentlichen Schlüssel entschlüsselt werden kann. Da per definitionem nur der rechtmäßige Besitzer Zugang zu dem privaten Schlüssel hat, kann auch nur er die Nachricht signiert haben - die Nachricht muss also zwangsläufig von ihm persönlich stammen.
Da bei digitalen Signaturen die Mathematik nicht zu kurz kommt, gibt es auch hier die Möglichkeit, den Vorgang des Signierens bzw. Verifizierens in Formeln auszudrücken. Ersteres wir oft als S(x), letzteres als V(x) bezeichnet. Die unsignierte Nachricht ist m, die signierte s. Der öffentliche Schlüssel ist, wie gehabt, k_e, der private k_d. Mit $S(m, k_d)$ wird einer Nachricht also die entsprechende Signatur verpasst (=s), welche mit $V(s, k_e)$ verifiziert werden kann.

die liebe Theorie

Da bei Signaturverfahren ebenfalls hauptsächlich spezielle, mathematische Probleme Anwendung finden, steckt eine jeweils andere Theorie dahinter. Auf der Algorithmenseite werden für den interessierten Leser ausgewählte Verfahren näher durchleuchtet.

Alice und Bob

Alice und Bob unterstützen nun schon seit einer ganzen Weile aktiv die Leute von ParaMind und können immer wieder mit ihren interessanten Texten und sensiblen Nachrichten die Besucher der Homepage begeistern und schockieren. Eines Tages haben sie beide in einem IRC-Channel etwas verdächtiges bemerkt und wollen dies schnellstmöglich melden, doch wie können sie beweisen, dass wirklich sie der Sender dieser Information sind und nicht eine dritter Person, welche mit Falschmeldungen den guten Ruf der ParaNews im zweiten Level von ParaMind sabotieren wollen? Sie

entscheiden sich für die Verwendung einer digitalen Signatur (welche übrigens von dem bekannten Programm PGP ebenfalls unterstützt werden). Dabei gehen sie wie folgt vor (es wird Bobs Schlüsselpaar benutzt):

einmalig auszuführende Aktion
1) Bob generiert ein Schlüsselpaar (öffentlicher und privater Schlüssel, erledigt in der Regel ein Programm)
2) und veröffentlicht seinen öffentlichen Schlüssel (z.B. im Internet als einfache ASCII-Datei), behält den privaten jedoch geheim.
3) Ein PM-Mitglied besorgt sich auf einem beliebigen Wege (z.B. über das Internet) Bobs öffentlichen Schlüssel und fügt diesen dem eigenen Schlüsselbund hinzu - dieser ist so lange gültig, bis Bob sein Schlüsselpaar wechselt.
für jede Kommunikation auszuführende Aktion
Bob und Alice schreiben ihre sensible Nachricht auf
und verschlüsseln (bzw. signieren) diese anschließend mit Bobs privatem Schlüssel.
Nun kann diese von jeder beliebigen Person mit Bobs öffentlichem Schlüssel entschlüsselt (bzw. als von Bob kommend verifiziert) werden.

weitere Varianten

Da es nicht nötig ist, die ganze Nachricht zu signieren und somit viel Zeit und Speicherplatz zu verschwenden, wird meist nur ein kleiner Teil signiert. Dieser Teil kann beispielsweise ein Hashwert ("Fingerabdruck") der Nachricht sein, welcher aus wenigen Zeichen besteht - im nächsten Text wird erklärt, was Hashfunktionen sind.

„Mach mal, Malcom!“

Wie auch bei der Public-Key Kryptografie hat Malcom hier schlechte Chancen, die Signatur zu brechen und somit nachahmen zu können. Bei einem schwachen Schlüssel könnte er einen Brute-Force Angriff versuchen; kennt er das verwendete Verfahren wäre ein Chosen Ciphertext Angriff "relativ" erfolgversprechend. Doch seine Chancen stehen bei einem direkten Angriff mehr schlecht denn recht.

Hintergrund: Hashverfahren

Was sind Hashfunktionen?

Durch Hashfunktionen (engl. hash - Gehacktes) wird der eingegebene Text durch den dititalen Fleischwolf gedreht, so dass man seine ursprüngliche Form nicht mehr erkennen kann. In der Praxis sollten Hashfunktionen hauptsächlich zwei Eigenschaften besitzen: Erstens sollten sie Einwegfunktionen sein, welche leicht in die eine Richtung zu rechnen sind (durch den Fleischwolf durch), aber unmöglich in die andere Richtung umkehrbar sind (Gehacktes durch den Fleischwolf zurück, so dass man wieder das ursprüngliche Tier hat). Zweitens darf es nicht passieren, dass für zwei unterschiedlichen Nachrichten der gleiche Hashwert erzeugt wird - man spricht hier von Kollisionsresistenz.

Vorteile

Durch die Eigenschaften der Unumkehrbarkeit und der Kollisionsresistenz lassen sich eindeutige Abdrücke von Texten erzeugen - vorausgesetzt, der erzeugte Hashwert ist mindestens gleichlang. In der Praxis wird jedoch meist ein Algorithmus verwendet, welcher eine kürzere Zeichenkette erzeugt, ein kleinerer Abdruck von einem größeren Objekt - die Analogie zum Fingerabdruck ist unverkennbar und tatsächlich trifft man diese Bezeichnung oft an. Dieser kleinere Abdruck kann nun signiert werden - die Datenmenge ist viel kleiner und somit auch der benötigte Zeitaufwand.
Ein weiterer Vorteil findet sich bei der Speicherung von Passwörtern: Speichert man die Originale, dann können diese gestohlen und unrechtmäßig verwendet werden. Speichert man jedoch nur deren Hashwert und vergleicht diesen mit dem Hashwert des eingegebenen Passwortes, so ist eine zuverlässige Anmeldung gewährleistet, während gleichzeitig ein indirekter Schutz gegen Diebstahl besteht (gestohlenes Hackfleisch ersetzt halt auch kein Haustier).

Nachteile

Ein Nachteil ist mit der wohl geläufigsten Anwendungsform von Hashfunktionen, dem Hashen vor dem Signieren, gekoppelt, denn dabei sollte die Länge des Hashwertes möglichst kürzer als der Originaltext sein, da sonst übermäßig viel Datenmaterial anfallen würde. Dadurch geht jedoch die Eigenschaft der Kollisionsresistenz verloren, was ein potentielles Sicherheitsrisiko darstellt. Dieses kann durch die richtige Wahl von Algorithmus und maximaler Länge stark reduziert werden, so dass eine praktische Sicherheit erreicht werden kann.

Der Algorithmus

Hashfunktionen werden in der Kryptologie, wie oben schon angedeutet, vor allem beim Hashen von Passwörtern und zum Erstellen eines digitalen Fingerabdrucks verwendet. Bei beiden ist eine kurze Zeichenkette als Ergebnis wünschenswert. Bei Passwörtern ist dies meist kein Problem, bei längeren Nachrichten hingegen schon. Hier greift man zu Algorithmen, welche die Nachricht blockweise hashen und miteinander verknüpfen. Eine einfach Anwendung würde beispielsweise einen 64 Bit gro-

ßen Block hashen, dann den nächsten 64 Bit großen Block. Die Ergebnisse werden miteinander XOR-verknüpft, bevor dieser nun ebenfalls 64 Bit große Ergebnisblock mit dem nächsten gehashten 64 Bit Block wieder verknüpft wird, usw. Als Ergebnis erhält man einen 64 Bit großen Block, den Fingerabdruck der Nachricht.
Weitere Beispiele für Algorithmen kann man bei den Blockchiffren finden, wobei sich dort speziell der CBC- und CFB-Modus anbieten würde (natürlich speziell abgewandelt).

Hybridverfahren

der Algorithmus

Wie der Name schon vermuten lässt, sind Hybridverfahren Mischverfahren, die Vorteile bewährter Algorithmen miteinander kombinieren, um dadurch ein Maximum an Sicherheit zu erreichen.
Das beste Beispiel für diese Verfahren ist das weltweit bekannte Programm PGP (Pretty Good Privacy). Dieses arbeitet mit der Standardeinstellung, dass bei der Verschlüsselung sowohl ein asymmetrisches Verfahren (RSA) als auch ein symmetrisches Blockchiffrierverfahren (IDEA) Anwendung findet. Bei jeder neuen Nachricht wird ein vom Programm zufällig erzeugter und nur für dieses eine Mal verwendeter Schlüssel, auch Sitzungsschlüssel bzw. "session key" genannt, für das symmetrische Verfahren generiert, mit welchem die eigentliche Nachricht verschlüsselt wird. Der "Einmal-Schlüssel" wird anschließend mit dem asymmetrischen Verfahren verschlüsselt und der Nachricht beigefügt. Nun werden die gesamten Daten an den Empfänger geschickt, welcher zuerst den Sitzungsschlüssel entschlüsselt und mit diesem dann den Rest der Nachricht in einen brauchbaren Zustand zurückverwandelt.

die liebe Theorie

Je nach der Art der kombinierten Algorithmen und der Kombination selbst ergeben sich unterschiedliche Vorteile für die Sicherheit. Wenn man es schafft, geschickt zwei unterschiedliche Algorithmen so zu kombinieren, dass man nur die Stärken erhält, während die Schwächen in der Kombination untergehen, dann hat man das Ziel der Hybridverfahren erreicht.
Bei Betrachtung des Sicherheitsniveaus von Hybridverfahren darf man jedoch nicht davon ausgehen, dass die ursprünglichen Sicherheitsstufen zusammenaddiert werden. Bei obigem Beispiel könnte sich ein Angreifer um die asymmetrische Verschlüsselung nicht kümmern und sich gleich an das symmetrische Verfahren wagen, wobei die Sicherheit des zweiten Verfahrens die gleiche bleibt, wie bei jeder vergleichbaren symmetrischen Verschlüsselung auch.
Jetzt stellt sich nur noch die Frage, warum man diesen Aufwand betreibt, wenn sich ein Angreifer gleich um den zweiten Algorithmus kümmern könnte. Beim Beispiel PGP besitzten beide Verfahrensarten ein hohes Eigenmaß an Sicherheit. Der Grund für die Kombination liegt darin, dass das symmetrische Verfahren schneller als das asymmetrische ist, weshalb für den Großteil der Nachricht auch dieses verwendet wird. Jedoch muss ein symmetrischer Schlüssel, mit welchem eine schnelle Verschlüsselung möglich wird, auch erst einmal auf einem sicheren Weg ausgetauscht werden, was sich am besten durch ein asymmetrisches Verfahren realisieren lässt. Nebenbei erhält man auf diese Weise noch den zusätzlichen Vorteil, dass ein potentieller Angreifer weniger Chiffrenmaterial erhält, welches er analysieren könnte, da zum einen für jede neue Kommunikation ein neuer symmetrischer Schlüssel generiert wird und zum anderen mit dem asymmetrischen Schlüssel nur geringe Datenmengen bearbeitet werden. Auf diese Weise kombiniert man die Sicherheit und den unkomplizierten Schlüsselaustausch des asymmetrischen mit der Schnelligkeit des symmetrischen Verfahrens und bekommt ein Paradebeispiel eines gelungenen Hybridverfahrens.

weitere Varianten

Je nach Einfallsreichtum und Kombinationsfähigkeit der Entwickler gibt es die unterschiedlichsten Hybridverfahren. PGP wurde auf Grund seines extrem großen Verbreitungsgrades hier als Beispiel kurz beschrieben. Da es meines Wissens nach kein vergleichbares Konkurrenzprodukt gibt, wird hier auf eine Beschreibung weiterer Verfahren verzichtet, da es - wie oben schon gesagt - theoretisch unendlich viele Kombinationsmöglichkeiten gibt. Weiterführende Informationen sind wie üblich bei den Links zu finden.

"Mach mal, Malcom!"

Für Malcom ist das sinnvollste, wenn er sich das schwächste und erfolgversprechendste Glied aus der Algorithmenkette vornimmt, vorausgesetzt er durchschaut überhaupt, dass es sich um ein Hybridverfahren handelt. Beim Beispiel PGP wäre es auf lange Sicht am sinnvollsten, wenn er sich auf den RSA-Algorithmus konzentriert, da er auf diese Weise nach einer erfolgreichen Kryptoanalyse (was sehr unwahrscheinlich ist) sofort an den symmetrischen Schlüssel kommen kann und ihn nicht jedesmal aufs Neue finden muss.

Zusatz: Steganografie

Was ist Steganografie?

Die Steganografie ist in diesem Sinne keine Disziplin der Kryptografie, da man mit ihr weder eine Nachricht verschlüsselt noch sonst wie an dem Verschlüsselungsprozess beteiligt ist. Bei dieser Disziplin geht es vielmehr um das Verbergen einer Nachricht, so dass der Gegner eine Kommunikation überhaupt nicht erst bemerkt. In Kombination mit kryptografischen Elementen kann man damit die Sicherheit der eigenen Kommunikation erheblich erhöhen, vorausgesetzt beide Verfahren werden korrekt angewandt.

Welche steganografischen Verfahren werden heute benutzt?

In unserer heutigen modernen Zeit sind einige der Verfahren, die auch schon früher funktionierten, natürlich ebenfalls noch zu gebrauchen. Jedoch haben sich mit der Digitalisierung weitere, fast noch raffiniertere Einsatzmöglichkeiten eröffnet. So ist es möglich, ganze Dateien in Bildern und Audiodateien zu verstecken, ohne dass diese einen nennenswerten und vom menschlichen Auge bzw. Ohr festzustellenden Qualitätsverlust erleiden. Dies könnte man stark vereinfacht realisieren, indem man die niederwertigsten für die Farbe zuständigen Bits der Trägerdatei dazu benutzt, um die Nachricht bitweise zu verstecken.
Doch nicht nur zum Verstecken von Dateien und Nachrichten hat sich dieses Verfahren bewährt, man kann es auch benutzten, um Wasserzeichen in einer Datei zu verstecken und somit eindeutig den rechtmäßigen Besitzer feststellen zu können.
Wer genauere Informationen zum Thema Steganografie und den heute verwendeten Programmen möchte, der findet bei den Links in Teil V einige interessante Quellen.

Welche Anwendungen der Steganografie gab es früher?

Zum Schluss des ersten Teils werden jetzt einige in der Regel früher benutzte Verfahren vorgestellt, welche in Form kleiner anekdotenähnlicher Geschichten die doch zum Teil recht anspruchvolle kryptografische Kost der vorangegangenen Seiten abrunden sollen.
Da gab es den Sklaven, dem man eine Schädelrasur spendiert hatte. Anschließend wurde selbiger Schädel mit der Nachricht tätowiert und besagter Sklave durfte seine Haare wieder wachsen lassen. Sobald diese die Nachricht ordnungsgemäß verborgen haben, wurde der Sklave dem Emfänger überstellt, welcher diesem den Schädel schon wieder zur Glatze machte, um so die Nachricht lesen zu können.
Weit unkomplizierter und schneller war das Eingravieren der zu verbergenden Nachricht auf einer Platte, welche anschließend mit Wachs überzogen wurde und die Nachricht unter sich begrub. Zusätzlicher Vorteil: Man benötigt keinen Sklaven.
Eine zum Teil schönere und besonders für die künstlerisch Begabten geeignetere Methode war da schon das Erstellen eines Gemäldes, in welches man auf möglichst unscheinbare Art und Weise die Nachricht einbaute. So kam es schon einmal vor, dass man buchstäblich zwischen den Grashalmen nach Anhaltspunkten suchen musste, aber auch die gepflegten Barthaare eines unbescholtenen Mannes sollten gerüchteweise schon missbraucht worden sein - belustigend, wenn die Kombination des Safes, welcher von einem Bild verdeckt wird, mitten im portraitierten Gesicht steht und die Einbrecher dies nicht ahnen.
Wer gerne liest (z.B. über Kryptologie), der kann sich natürlich auch ein inhaltlich besonders wertvolles Exemplar nehmen und unter ganz bestimmten Buchstaben mit einer Nadel einen Einstich machen. Wenn jemand nur oberflächlich über die Seiten blättert oder ohne etwas zu ahnen das Buch liest, dem wird dies normalerweise nicht auffallen. Apropos auffallen: Wem ist die auf dieser Seite versteckte Nachricht aufgefallen? Keinem? Nun, dann scheint dieses Prinzip auch in moderneren Zeiten noch zu funktionieren …

modernes Beispiel

Und zum Schluss noch ein modernes Beispiel: In nachfolgendem Bild (dieses im Übrigen nicht allzu ernst nehmen) wurde mit dem Programm ***Camouflage 1.2.1*** (weitere Informationen bei den Links) eine kleine Textdatei versteckt:

Eine funktionsfähige Bilddatei wird normalerweise parallel mit der PDF-Datei zum Download angeboten. Ach ja, dass Passwort lautet übrigens ***stegano*** …

Ergänzung: bekannte Algorithmen

Auf dieser Seite sind einige bekannte und interessante Algorithmen der verschiedenen kryptografischen Kategorien zu finden:

symmetrische Algorithmen

- Nomenklator
- Vigenère-Verschlüsselung
- One-Time-Pad
- Data Encryption Standard (DES)

asymmetrischen Algorithmen

- Diffie-Hellmann (sichere Schlüsselvereinbarung)
- RSA (Rivest, Shamir, Adleman)

digitale Signaturen

- RSA-Signaturverfahren

Hashverfahren

- MD5

Nomenklator

der Algorithmus

Ein Nomenklator ist eine Kombination einer konventionellen Verschlüsselung (in der Regel ein Substitutionsverfahren) und einem Codebuch, weshalb er zu den Hybridverfahren zu zählen ist. Da theoretisch alle kryptografischen Algorithmen in Frage kommen können, kann man nur schwer einen allgemeingültigen Algorithmus aufstellen. Eines ist aber immer gleich: Zuerst werden alle im Text vorkommenden relevanten Wörter durch die entsprechenden Codes ersetzt, bevor der Text mitsamt den Codewörtern durch das entsprechende Kryptografieverfahren verschlüsselt wird.

die liebe Theorie

Die Verwendung eines Nomenklators ändert absolut nichts an der eigentlichen Sicherheit des gewählten Kryptografieverfahrens. Die Verschlüsselung kann mit oder ohne Codewörter auf die gleiche Weise mit der gleichen Schwierigkeit und dem gleichen Aufwand gebrochen werden. Jedoch setzt der Nomenklator genau bei diesem Fall (der Entschlüsselung des Geheimtextes durch den Feind) an: Der Angreifer hält zwar den Klartext in den Händen, kann daraus jedoch nur einen beschränkten Nutzen ziehen. Die wichtigen Informationen verbergen sich immer noch hinter den Codes mit welchen er nichts anfangen kann.
Die Verwendung eines Nomenklators kommt also in erster Linie als zusätzliche Absicherung in Frage, durch welche die wichtigsten Informationen trotz einer erfolgreichen Kryptoanalyse geheim gehalten werden können.

Vigenère-Verschlüsselung

der Algorithmus

Die Vigenère-Verschlüsselung bzw. -Verschiebung trägt den Namen ihres Schöpfers Blaise de Vigenère. Sie zählt zu den bekanntesten polyalphabetischen Substitutionen, was auf ihre unkomplizierte und dennoch effektive Anwendbarkeit zurückzuführen ist. Um einen Text zu verschlüsseln werden genauso viele Geheimtextalphabete aufgestellt wie Buchstaben im Alphabet vorkommen. Der Einfachheit halber handelt es sich dabei jeweils um eine ROT-1 Variante des vorhergehenden Alphabets (siehe dazu auch untenstehende Tabelle).

deutsches Vigenère-Quadrat																										
Geheim: A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Geheim: B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
Geheim: C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
Geheim: D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
Geheim: E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
Geheim: F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
Geheim: G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
Geheim: H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
Geheim: I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
Geheim: J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
Geheim: K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
Geheim: L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
Geheim: M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
Geheim: N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
Geheim: O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
Geheim: P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Geheim: Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
Geheim: R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
Geheim: S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
Geheim: T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
Geheim: U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
Geheim: V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
Geheim: W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
Geheim: X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Geheim: Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Geheim: Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Dieses Quadrat kann als fester Bestandteil des Algorithmus angesehen werden und ist somit frei zugänglich. Zum eigentlichen Verschlüsseln ist nur ein Schlüsselwort nötig, welches die zu verwendenden Geheimtextalphabete angibt.
Zum besseren Verständnis wird nachfolgend der Text "Kauf Dir eine Axt im Baumarkt" anhand des Schlüsselwortes "hacken" verschlüsselt:

Klar	k	a	u	F	d	i	r	e	i	n	e	a	x	t	i	m	b	a	u	m	a	r	k	t
Schlüssel	h	a	c	K	e	n	h	a	c	k	e	n	h	a	c	k	e	n	h	a	c	k	e	n
Geheim	R	A	W	P	H	V	Y	E	K	X	I	N	E	T	K	W	F	N	B	M	C	B	O	G

weitere Varianten

Um das Verfahren noch sicherer zu machen ohne seine Benutzung zu verkomplizieren, kann man auf eine andere Anwendungsart zurückgreifen. Dabei vereinbart man wie gehabt ein Schlüsselwort und schreibt dieses unter den Klartext (siehe oben). Wenn man das Wort nun aber ausgeschrieben hat und damit noch nicht den ganzen Klartext abdecken konnte (was die Regel sein dürfte), fügt man statt dem Schlüsselwort selbst die noch nicht benutzten Buchstaben des Alphabets dazu. Beim Schlüsselwort "hacken" würde das folgendermaßen aussehen: "hackenbdfgijlmopqrstuvwxyz". Dadurch wird erst bei dem 27. Klartextbuchstaben wieder der gleiche Schlüsselbuchstaben verwendet, wodurch die effektive Schlüssellänge statt sechs nun 26 beträgt - und das ohne Mehraufwand. Aber Vorsicht: Falls man sich für diese Variante entscheidet, dann sollten anders permutierte Geheimtextalphabete verwendet werden!

One-Time-Pad

der Algorithmus

Das One-Time-Pad ist der Alptraum eines jeden Kryptoanalytikers, da es auf elegante und dennoch leicht verständliche Weise größtmögliche Sicherheit mit einem einfachen Aufbau vereint.
Der Algorithmus schreibt eine simple Verknüpfung des Schlüssels mit dem Klartext vor, z.B. durch Addition (vgl. monoalphabetische Substitution) oder XOR-Verknüpfung, mit der Bedingung, dass der Schlüssel exakt die gleiche Länge wie der Klartext selbst haben muss und komplett zufällig generiert wurde. Das war auch schon die komplette Algorithmenbeschreibung und fertig ist das bewiesenermaßen sicherste Kryptografieverfahren, dass es jemals geben wird.

die liebe Theorie

Im obigen Abschnitt wurde gesagt, das One-Time-Pad sei das sicherste Verfahren, dass es jemals geben wird. Das bedeutet streng genommen ein Sicherheitsniveau von 100% - und genau so ist es auch. Da der zufällig erzeugte Schlüssel genauso lang wie der Klartext selbst ist gibt es keinen einzigen analytischen Angriffspunkt. Je nach verwendetem Schlüssel lässt sich in jeden beliebigen Geheimtext jede beliebige Nachricht gleicher Länge hineininterpretieren.
Hierzu ein Beispiel: Wir wählen als Chiffrieralgorithmus die Addition (vgl. monoalphabetische Substitution), weshalb man zum Dechiffrieren die Subtraktion verwenden muss, wobei gilt:

A = 0, B = 1, ...
0 - 1 = 25 (A - B = Z)

Nachfolgende Tabelle zeigt nun zwei gleiche Geheimtexte, welche mit zwei unterschiedlichen Schlüsseln zwei vollkommen sinnvolle, aber unterschiedliche Klartexte ergeben. Zwecks leichterer Nachvollziehbarkeit werden die entsprechenden Zahlenwerte bei den Buchstaben mitangegeben:

GEHEIM	A (0)	K (10)	R (17)	E (4)	N (13)	V (21)	W (22)	O (14)	A (0)	K (10)	R (17)	E (4)	N (13)	V (21)	W (22)	O (14)
Schlüssel	Z (25)	C (2)	D (3)	I (8)	N (13)	Q (16)	R (17)	K (10)	I (8)	I (8)	K (10)	Q (16)	D (3)	H (7)	S (18)	G (6)
klar	b (1)	i (8)	o (14)	w (22)	a (0)	f (5)	f (5)	e (4)	s (18)	c (2)	h (7)	o (14)	k (10)	o (14)	e (4)	i (8)

Jetzt bleibt nur noch die Frage zu klären, warum das weltweit sicherste Verfahren nicht überall eingesetzt wird. Die Antwort liegt auf der Hand: Der Schlüssel ist einfach zu groß und zu unhandlich.

weitere Varianten

Das wichtigste ist die Tatsache, dass der zufällig generierte Schlüssel und der Klartext exakt die gleiche Länge haben. Auf welche Art und Weise man beide Teile miteinander verknüpft, spielt dabei keine übergeordnete Rolle, solange dadurch kein kryptoanalytischer Angriff ermöglicht wird. Weiteren Varianten steht also nichts im Wege.
Es gibt auch Ansätze welche versuchen die Sicherheit des One-Time-Pads zu erreichen ohne mit einem riesigen Schlüssel hantieren zu müssen. Dabei wird ein Pseudo-Zufallszahlengenerator benutzt, welcher einen kontinuierlichen Schlüsselstrom erzeugt, der zur Verschlüsselung benutzt wird. Die Sicherheit kommt an die 100% zwar nicht heran, aber mit einem qualitativ hochwertigen Generator lassen sich beachtliche Ergebnisse erzielen.

„Mach mal, Malcom!“

Malcom wird kein einziges analytisches Werkzeug finden, welches er zum Knacken dieses Codes benutzen könnte. Er müsste in diesem Fall also auf Spionage oder auf seine zwischenmenschlichen Fähigkeiten (Social Engineering, Folter, etc.) zurückgreifen.

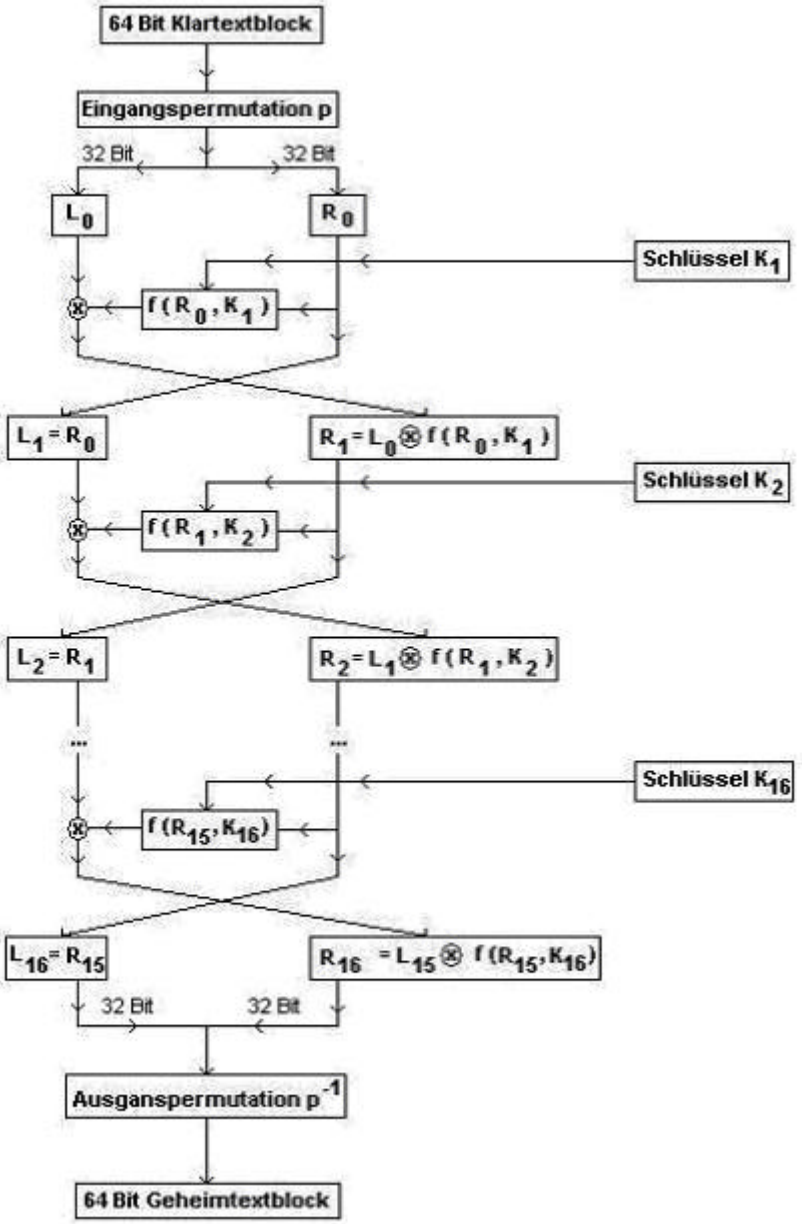
Data Encryption Standard (DES)

der Algorithmus

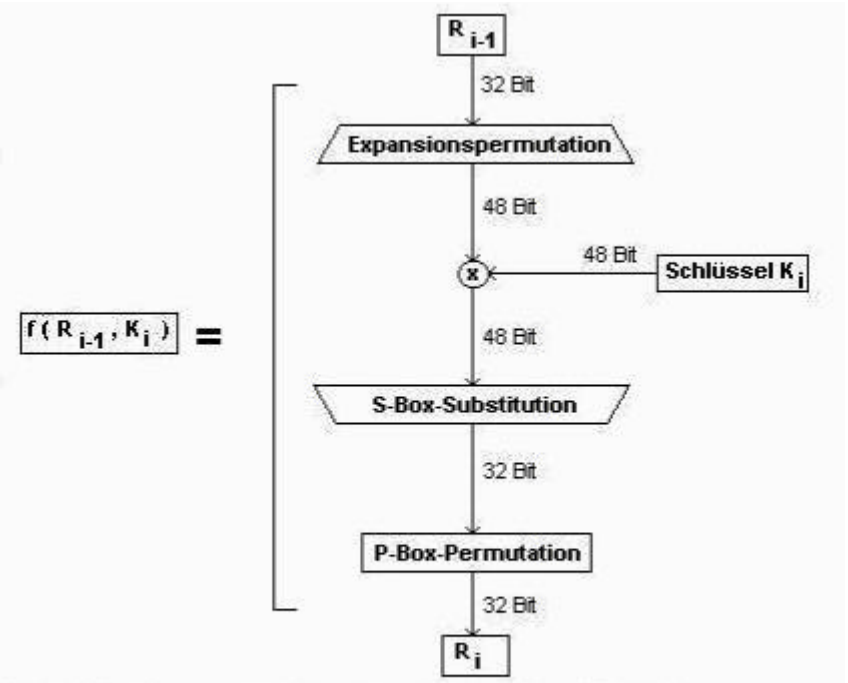
grundlegender Aufbau:
Der Data Encryption Standard (DES, auch bekannt als Data Encryption Algorithm, DEA) ist ein auf Feistel-Netzwerken basierendes, symmetrisches Blockverschlüsselungsverfahren, welches Textblöcke mit einer Länge von 64 Bit mit einem ebenfalls 64 Bit (davon 8 Paritätbits) langen Schlüssel chiffriert.
Am Anfang der Verschlüsselung wird der Text in 64 Bit lange Blöcke zerlegt. Jeder dieser Blöcke wird einer Eingangspermutation p(x) unterzogen, welche die Reihenfolge der einzelnen Bit auf eine fest vorgegebene Weise verändert. Danach wird der 64 Bit lange Block in eine rechte (R) und linke (L) Hälfte zerlegt, welche beide 32 Bit lang sind. Nun beginnt der eigentliche Verschlüsselungsvorgang, welcher aus 16 identischen Runden besteht:

Der R-Block durchläuft eine Expansionspermutation, wodurch die Länge auf 48 Bit vergrößert wird. Diese 48 Bit werden anschließend mit den 48 Bit des Teilschlüssels XOR-verknüpft. Das Ergebnis wird in acht sechs Bit lange Teile zerlegt, die als Eingabe für die acht S-(Substitutions-)Boxen dienen. Die S-Boxen geben von den jeweils sechs Eingabebits jedoch nur vier Bit aus, womit die Originallänge von 32 (8x4) Bit wieder hergestellt wird. Als vorletzter Schritt wird der 32-Bit-Block einer P-Box-Permutation unterzogen bevor am Ende dieser Runde der R-Block mit dem bis jetzt nicht benutzten L-Block XOR-verknüpft wird.
Als Eingabeblöcke der nächsten Runde dient nun als R-Block das Ergebnis des bisherigen Vorgangs, als L-Block der R-Block der vorhergehenden Runde.
Nachdem nacheinander alle 16 Runden durchlaufen wurden, wird der 64 Bit-Block noch einer Abschlusspermutation $p^{-1}(x)$ unterzogen. Das Ergebnis ist nun ein komplett verschlüsselter 64 Bit langer Geheimtextblock.

ein Block, 16 Runden, übersichtlich:
(oben beschriebenes grob dargestellt)



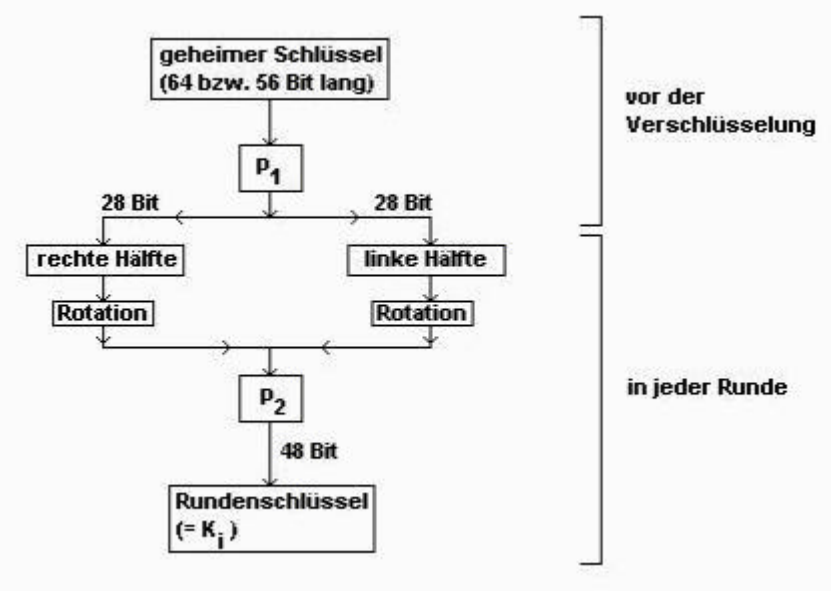
ein Halbblock, eine Runde, detailliert:
(eine beliebige der 16 Runden ausführlich dargestellt)



Grafik: [chronyx] nach einer Vorlage von Robert Gering (1997), gefunden auf http://ig.cs.tu-berlin.de/ap/rg/002/glossar/d-terms/des.html

Schlüssel:
Die ganze Zeit wurde nicht mit einem Schlüssel, sondern anscheinend mit mehreren gearbeitet. Was hat es damit auf sich? Der geheime, effektive 56 Bit (mit Paritätbits 64 Bit) lange Schlüssel wird selbst verschiedenen Operationen unterworfen, bevor er, oder genauer gesagt ein Teil von ihm, zur Verschlüsselung eingesetzt wird.
Die 56 Schlüsselbit werden zuerst durch die Permutationsfunktion $p_1(x)$ permutiert und dann in zwei Hälften zu je 28 Bit zerlegt. In jeder Runde werden diese Bit um ein bzw. zwei Bit rotiert, so dass am Ende der 16 Runden die Ausgangssituation wieder gegeben ist (und der Schlüssel nicht mehr neu eingelesen und permutiert werden muss). Um aus diesen zwei 28 Bit großen Teilen jedoch einen 48 Bit langen Rundenschlüssel zu erhalten, müssen diese eine weitere Permutationsfunktion ($p_2(x)$) durchlaufen, welche als Ergebnis den endgültigen Rundenschlüssel liefert. Wie in obigen Schemata erkennbar werden nun diese 48 Rundenschlüsselbit mit den 48 Bit der rechten Hälfte des aktuellen Blocks XOR-verknüpft.

Rundenschlüsselerzeugung:
(Schema der Rundenschlüsselerzeugung)



S-Boxen:
Bei den S-Boxen handelt es sich im Grunde genommen um nichts weiter als eine statische Tabelle deren Inhalt mit den 48 Bit verknüpft wird. Das Besondere daran ist, dass der für die Verknüpfung verwendete Tabellenteil unter Einfluss der Eingabebitfolge ausgewählt wird. Diese Tatsache (nicht-Linearität) ist im Grunde für die Sicherheit des gesamten Verfahrens verantwortlich. Ein weiterer Effekt der S-Boxen ist die Reduzierung auf 32 Bit, womit wieder die Ausgangslänge (vgl. Zustand vor der Expansionspermutation) hergestellt wird.
Zum Schluss noch eine Anmerkung: Die S-Boxen sind der Grund für das Misstrauen, welches diesem Algorithmus lange Zeit entgegengebracht wurde. Bei der Entwicklung von DES hatte die NSA ihre Hände mit im Spiel, vor allem bei den S-Boxen, in welchen man eine geheime Hintertür vermutete. Einige Zeit später wurde dann eine neue kryptoanalytische Angriffsform entwickelt, die differentielle Kryptoanalyse. Natürlich wurde dieser Angriff auch auf DES gestartet und man machte dabei eine überraschende Entdeckung: DES hielt extrem gut stand und zwar aus einem Grund - die S-Boxen sind ein nahezu perfektes Schutzschild. Es sind keine besseren Tabelleninhalte bekannt, welche das Verfahren sicherer machen würden. Das bedeutet, dass der NSA dieser Angriff schon damals bekannt war, sie diesen jedoch geheim hielt ... ob es wohl schon wirksame Angriffe gegen RSA und Co gibt?

Diffie-Hellmann

der Algorithmus

grundlegender Aufbau:
Der Diffie-Hellmann-Algorithmus ist genau genommen kein Chiffrieralgorithmus, sondern er dient zum sicheren Austausch von Schlüsseln (z.B. für DES) über unsichere

Verbindungen, einem jahrtausendealten Problem. Dabei bedient sich dieser Algorithmus dem diskreten Logarithmus Problem.
Um einen sicheren Schlüssel erstellen zu können, wählen die beiden Kommunikationspartner (Alice und Bob, A und B) zusammen eine große Primzahl p und eine beliebige weitere Zahl z. Beide Partner wählen anschließend jeder für sich eine große, geheime Zahl (a und b), berechnen jeweils $a_2 = z^a \bmod p$ bzw. $b_2 = z^b \bmod p$ und schicken das Ergebnis an den Partner. Dieser berechnet nun $k = b_2^a \bmod p$ bzw. $k = a_2^b \bmod p$. Da $k = z^{ab} \bmod p$ befinden sich nun beide im Besitz der gleichen Zahl k (dem Schlüssel), ohne dass diese jemals übermittelt werden musste. Ein eventueller Lauscher, Malcom, hat keine Chance auf diese Zahl k zu kommen, da ihm die beiden geheimen Zahlen a und b fehlen.
Dieser sichere Schlüsselaustausch hat jedoch auch einen Nachteil: Beide Partner müssen gleichzeitig anwesend (z.B. online) sein oder diese Vereinbarung stückchenweise (z.B. per E-Mail) vollziehen, wodurch eine schnelle und gleichzeitig sichere Kommunikation mit einem unbekannten Partner in der Regel nicht möglich ist. Will nun Bob an Alice nachts um drei Uhr eine geheime Nachricht schicken, so muss er erst warten bis sie aufgestanden ist und ans Telefon, Computer, Briefkasten, etc. gehen kann.

Alice und Bob

Schritt	Alice	Bob	gemeinsam	Beispiel
1	-	-	große Primzahl p große Zahl z	p = 23 z = 49
2	wählt beliebige Zahl a berechnet $a_2 = z^a \bmod p$	wählt beliebige Zahl b berechnet $b_2 = z^b \bmod p$	-	a = 20 $a_2 = 49^{20} \bmod 23 = 18$ b = 30 $b_2 = 49^{30} \bmod 23 = 6$
3	schickt $a_2 = 18$ an Bob	schickt $b_2 = 6$ an Alice	-	s.o.
4	berechnet $k = b_2^a \bmod p$	berechnet $k = a_2^b \bmod p$	-	$k = 6^{20} \bmod 23 = 16$ $k = 18^{30} \bmod 23 = 16$
5	-	-	beide haben nun ein gemeinsames k, welches sie als geheimen Schlüssel verwenden können	**k = 16**

Variable	Bedeutung der Variable
p	große Primzahl, gemeinsam gewählt
z	große, beliebige Zahl, gemeinsam gewählt

a	geheime, beliebig gewählte Zahl von A
a_2	Ergebnis von A mit a, welches an B geschickt wird
b	geheime, beliebig gewählte Zahl von B
b_2	Ergebnis von B mit b, welches an A geschickt wird
k	auf sichere Weise über unsichere Kommunikationswege ausgemachter Schlüssel für die weitere Verwendung (z.B. mit IDEA)

RSA (Rivest, Shamir, Adleman)

der Algorithmus

grundlegender Aufbau:

Der RSA-Algorithmus basiert auf dem mathematischen Problem der Faktorisierung großer Primzahlprodukte. Konkret heißt dies, dass man das Ergebnis, welches man durch die Multiplikation zweier großer Primzahlen (mit groß meint man Größenordnungen von mehreren hundert Stellen) erhält, sehr schwer wieder in die Ausgangsprimzahlen zerlegen kann. Grob gesagt kann man dies nur durch Ausprobieren aller möglicher Primzahlkombinationen erreichen, was bei diesen Größenordnungen so lange dauern würde, dass man diesen Angriff als praktisch unmöglich einstufen und das Verfahren als praktisch sicher ansehen kann.

Vor der eigentlichen Verschlüsselung müssen von Seiten des Empfängers einige Schritte unternommen werden, welche in der Praxis von Programmen übernommen werden. Zuerst wählt man zwei große Primzahlen p und q, welche beide ungefähr gleich viele Stellen haben und bildet sowohl deren Produkt n als auch die Zahl f (n) = (p-1)(q-1). Des weiteren sucht man sich zwei natürliche Zahlen e und d, welche die Bedingung ed mod f (n) = 1 bzw. ed = k(p-1)(q-1)+1 erfüllen, wobei k eine beliebige natürliche Zahl sein kann. Für die Überprüfung dieser Eigenschaft verwendet man am besten den erweiterten euklidischen Algorithmus. Hat man sich nun alle notwendigen Zahlen ausgesucht und alle Bedingungen können erfüllt werden, dann ist man - was die Schlüsselgenerierung angeht - auch schon am Ziel. e zusammen mit n stellt den öffentlichen Schlüssel dar, d ist der geheime. Die Zahlen p, q und f (n) müssen auf jeden Fall ebenfalls geheim gehalten werden, oder man vernichtet sie am besten gleich, da sie nicht mehr benötigt werden.

Will man eine Nachricht m nun verschlüsseln, so setzt man die Zahlen in die Gleichung $E(m) = m^e \bmod n = c$ ein. Dabei muss man jedoch beachten, dass m nicht größer als n sein darf. Ist dies dennoch der Fall, dann kann man die Nachricht in einzelne Blöcke zerlegen, welche diese Bedingung erfüllen. Zum Entschlüsseln der Chiffre c verwendet der Besitzer des geheimen Schlüssels folgende Gleichung: $D(c) = c^d \bmod n$ und fertig ist eines der erfolgreichsten Public-Key-Systeme.

Alice und Bob

Das folgende Beispiel bedient sich zu Demonstrationszwecken relativ kleinen Zahlen, welche in der Praxis keine ausreichende Sicherheit bieten:
Alice wählt zuerst zwei Primzahlen, p=47 und q=59 und bildet deren Produkt n=2773. Daraus berechnet sie für die Zahl f (n)=(p-1)(q-1)=2668. Als erste natürliche Zahl wählt sie e=17 (hat mit f (n) keinen gemeinsamen Teiler). Des weiteren legt sie für d=157 fest, womit auch die Bedingung ed mod f (n) = 1 erfüllt wäre. Somit besteht der öffentliche Schlüssel aus e=17 und n=2773, der geheime aus d=157.
Will Bob nun die Nachricht "krypto", in Zahlen 111825162015, an Alice senden, so zerlegt er sie zuerst in kleinere Teile für die gilt: $m_i < n$ und erhält m_1=1118, m_2=2516 und m_3=2015. Diese setzt er nun nacheinander in die Gleichung $c_i=E(m_i)=m_i^{17}$ mod 2773. Als Chiffre erhält er daraufhin 512.2666.774, welche er an Alice sendet.
Diese kann die Nachricht nun entschlüsseln, indem sie die einzelnen Chiffreblöcke in die Formel $m_i=D(c_i)=c_i^{157}$ mod 2773 einsetzt. Einfach ausprobieren - es funktioniert tatsächlich!

RSA-Signaturverfahren

der Algorithmus

Wer den RSA-Verschlüsselungsalgorithmus kennt, beherrscht schon mindestens ein Verfahren zur Erzeugung digitaler Signaturen, denn speziell bei diesem Algorithmus ist es möglich ihn einfach umzudrehen bzw. zwei Parameter miteinander zu vertauschen. Möchte Alice an Bob eine signierte Nachricht senden, dann verschlüsselt sie die Nachricht zuerst nach durch den RSA-Algorithmus, jedoch ohne Bobs öffentlichen Schlüssel zu benutzen. Diesen ersetzt sie nämlich durch ihren geheim gehaltenen privaten Schlüssel. Die Nachricht kann nun von jeder beliebigen Person "entschlüsselt" werden, die Alice öffentlichen Schlüssel kennt - also praktisch jeder.
Schlussfolgerung: Wenn Bob die Nachricht mit dem öffentlichen Schlüssel von Alice "dechiffrieren" kann, dann muss diese mit ihrem privaten Schlüssel chiffriert worden sein. Da aber ausschließlich Alice den privaten Schlüssel kennt, muss zwangsläufig die Nachricht von ihr stammen.

MD5

der Algorithmus

grundlegender Aufbau:

Eines der Designkriterien für den Algorithmus MD5 war die sichere "Zusammenstauchung" eines größeren Textes auf 128 Bit, wobei das Aufspüren von Kollisionen möglichst schwer sein sollte. Um dies zu realisieren wurde von Professor Rivest folgender Algorithmus kreiert, welcher jetzt grob dargestellt werden soll. Wer eine genauere Beschreibung mitsamt den Quellcodes benötigt, der findet bei den Links weitere Ressourcen.
Nun aber zurück zum MD5-Algorithmus. Er besteht im Wesentlichen aus fünf Schritten:

Im ersten Schritt wird die Größe der Originalnachricht durch Hinzufügen weiterer Bit so angepasst, dass die Länge in Bit + 64 ohne Rest durch 512 teilbar ist. Die Nachricht wird immer aufgefüllt, selbst wenn sie im Originalzustand die Bedingung erfüllt hätte.
Der angepassten Nachricht wird im zweiten Schritt nun ein weiterer 64 Bit großer, von der Originalnachricht abhängiger Block angehängt, so dass die gesamte Nachricht als ein Vielfaches von 16 32 Bit langen Blöcken angesehen werden kann.
Anschließend werden im dritten Schritt die vier MD5-Buffer mit jeweils 32 Bit großen Blöcken gefüllt.
Es sind vier verschiedenen Funktionen vorhanden, welche jeweils drei 32 Bit Blöcke aufnehmen und als Ergebnis einen 32 Bit Block ausspucken. Diese werden nun im vierten Schritt verwendet.
Zum Schluss werden im fünften Schritt die vier Ergebnisse der Funktionen ausgegeben, wodurch - wie oben schon erwähnt - ein 128 Bit großer Fingerabdruck der Nachricht vorliegt.

Teil II – Kryptoanalyse

Einführung in die Kryptoanalyse

"Seitdem Menschen dazu fähig sind, Intrigen zu spinnen, Pläne zu schmieden und diese in irgendeiner Form aufzuzeichnen gibt es das Bedürfnis, geheime Informationen einem ausgewählten Personenkreis mitzuteilen, ohne dass Unbefugte vom Inhalt der Nachricht erfahren. Die Kryptografie, eine Disziplin der Kryptologie, beschäftigt sich genau mit diesem Problem und das schon seit Jahrtausenden. Von einfachen, heute fast primitiv scheinenden bis hin zu hochkomplexen, auf mathematischen und physikalischen Problemen und Sachverhalten basierenden Verfahren hat sich diese Wissenschaft entwickelt." (erster Absatz der Einführung in die Kryptografie)
Seitdem Menschen dazu fähig sind ihre Kommunikation über Intrigen und geheime Pläne zu verschlüsseln gibt es das Bedürfnis, verschlüsselte Informationen einem nicht befugten Personenkreis zugänglich zu machen ohne die Erlaubnis der Befugten zu besitzen. Die Kryptoanalyse, eine Disziplin der Kryptologie, beschäftigt sich genau mit diesem Problem und das schon seit Jahrtausenden. Von einfachen, heute fast primitiv scheinenden bis hin zu hochkomplexen, auf mathematischen Grundlagen basierenden Verfahren hat sich diese Wissenschaft entwickelt.
Nachdem auf den vorangegangenen Seiten die Möglichkeiten und Werkzeuge der Kryptografen eingehend erläutert wurden wechseln wir nun auf die Gegenseite, in Malcoms Arbeitsraum und schauen ihm ein wenig über die Schultern wie er sich an den geheimen Chiffren zu schaffen macht, in der Hoffnung den Klartext oder am besten gleich den geheimen Schlüssel zu erlangen.
Doch vorher gilt es noch einige Ansichtsweisen und Grundbegriffe der Kryptoanalyse zu klären; Grundbegriffe der Kryptografie (Einführung in die Kryptografie) werden als bekannt vorausgesetzt.

aktiver und passiver Angriff:
In der Kryptoanalyse sind zwei grobe Klassifizierungen eines Angriffs bekannt, nämlich den aktiven und den passiven Angriff. Bei ersterem versucht der Angreifer den Nachrichtenverkehr seiner Gegner aktiv zu beeinflussen, indem er Nachrichten fälscht, manipuliert, stört und vernichtet. Meist hat er destruktive Absichten und will seinem Gegner lieber direkten Schaden zufügen als umfassende Kenntnisse über dessen Kommunikation zu erlangen, um ihm später auf subtilere Weise zu verwunden. Je nach Situation ist diese Art des Vorgehens auch sinnvoll und effektiv, in diesem Werk widmen wir uns jedoch mehr der verborgeneren Art des Angriffs.
Bei einem passiven Angriff will der Angreifer normalerweise nicht, dass sein Gegner von ihm und seinem Vorhaben weiß, da dies fast immer Änderungen des geheimen Schlüssels und Verschärfungen des Sicherheitssystems zur Folge hat. Aus diesem Grund beschränkt er sich oft auf das Belauschen des Nachrichtenverkehrs und auf die zur Verschlüsselung benötigten, öffentlich zugänglichen Ressourcen (bei Public-Key-Kryptografie wäre dies z.B. der öffentliche Schlüssel). Mit seinen recherchierten Informationen versucht er nun die Originalnachricht und/oder den geheimen Schlüs-

sel aus den erhaltenen Chiffren wiederherzustellen und somit Einblicke in die Kommunikation seines Gegners zu bekommen.

Häufigkeitsanalyse

Eine der größten Hürden bei der Realisierung einer sicheren Kommunikation ist die menschliche Sprache. Sie besitzt sehr viele Regeln und Regelmäßigkeiten wie Buchstabenhäufigkeit, existente und nichtexistente Bi- und Trigramme usw. die bei einer Verschlüsselung zum Verhängnis werden können, da sie einem Kryptoanalytiker als Anhaltspunkte dienen. Ihren größten Nutzen erweist dieser statistische Angriff bei mono- und polyalphabetischen Substitutionen, da dort die reellen Häufigkeiten zumindest teilweise noch vorzufinden sind. Bei moderneren Verfahren hingegen werden ausgefeiltere Taktiken benötigt, welche bei den grundlegenden modernen Angriffsarten zu finden sind.
Dieser vorliegende Text bietet übrigens nur eine Hilfestellung bei der Dechiffrierung relativ einfacher Verschlüsselungen, soll jedoch nicht als Anleitung zum Knacken aller hier vorkommenden Algorithmen angesehen werden. Solch eine vollständige Anleitung zu schreiben würde den Rahmen dieses Werkes sprengen, da es zu viele kleine Variationen gibt, welche berücksichtigt werden müssten. Wer jedoch Spaß an der Sache findet, der kann sich in seiner Freizeit an diversen in Büchern, dem Internet und bei uns vorgestellten Chiffren versuchen und über Praxiserfahrung seine Technik verfeinern, denn auch bei der Kryptoanalytik ist der Spruch "learning by doing" nicht fehl am Platz.

einfache Häufigkeitsanalyse (gegen monoalphabetische Substitution)
Die einfache Häufigkeitsanalyse stützt sich auf die Normalverteilung der Buchstaben in der Sprache. Jeder Buchstabe kommt in jedem Text mit einer bestimmten Häufigkeit vor; werden viele Texte auf die Buchstabenhäufigkeiten hin untersucht und am Schluss der Durchschnitt dieser Häufigkeiten gebildet, dann ist das Ergebnis die Normalverteilung der Buchstaben, eine repräsentative Aussage über die Wahrscheinlichkeit, wie oft ein Buchstabe in einem durchschnittlichen Text vorkommt. Auf der Seite Malcoms Munitionskammer sind einige wichtige Häufigkeitstabellen zu finden, welche für einen Kryptoanalytiker hilfreiche Werkzeuge sein können.
Wird für jeden Klartextbuchstaben genau ein Geheimtextbuchstaben verwendet (liegt also eine monoalphabetische Substitution vor), so verbergen diese zwar die richtigen Buchstaben, deren Häufigkeiten bleiben aber die gleichen. Diesen Umstand nutzt der Kryptoanalytiker und stellt die relativen Häufigkeiten der Geheimtextbuchstaben fest, welche er mit der Normalverteilung vergleicht - je länger der Geheimtext dabei ist, desto ähnlicher sind sie sich. Zusätzlich kann er noch Buchstabenpaare und -tripel heranziehen, welche ebenfalls zur Analyse hilfreich sind.
Da in der Praxis die tatsächliche Häufigkeit mit der Normalverteilung selten übereinstimmt, kommt man mit einem sturen Vergleichen und Austauschen jedoch oft nicht zum gewünschten Ergebnis. Man muss systematischer an die Sache herangehen: So könnte man mit den häufigsten Buchstaben beginnen und die dazugehörenden Bi- und Trigramme und deren Häufigkeit gleich mit untersuchen. Wenn schon einige Buchstaben entschlüsselt wurden, ist ein Blick auf das bisherige Ergebnis ebenfalls zu empfehlen, da man oft Wörter aus dem Kontext heraus erraten und somit auf weitere Buchstaben kommen kann. Durch diese Vorgehensweise kommt man Stück für

Stück dem Ziel näher bis man den Klartext rekonstruiert und den geheimen Schlüssel entdeckt hat.

erweiterte Häufigkeitsanalyse (gegen polyalphabetische Substitution)
Wenn bei einer normalen Häufigkeitsanalyse nur wirres Kauderwelsch als Ergebnis zu Tage kommt, so handelt es sich bei der vorliegenden Verschlüsselung vermutlich um eine polyalphabetische Substitution, bei der ein und derselbe Geheimtextbuchstabe für mehrere Klartextbuchstaben stehen kann und somit eine normale Häufigkeitsanalyse zum Scheitern verurteilt ist. Doch auch dieses kryptografische Chiffrierverfahren ist durch eine Variation der vorhergehenden Attacke knackbar.
Der Kryptoanalytiker muss sich vor Augen halten, dass die Buchstabenhäufigkeit nicht verändert worden ist; sie ist nur über den gesamten Text verteilt, um genau zu sein auf mehrere monoalphabetische Chiffren, nämlich auf genauso viele wie die Anzahl der Buchstaben des Schlüsselwortes beträgt (bei einem vierstelligen Schlüsselwort wurden vier verschiedene Geheimtextalphabete benutzt, weshalb man von vier unabhängigen monoalphabetischen Substitutionen ausgehen kann). Wenn er es also fertig bringt, den Geheimtext in seine Einzelteile zu zerlegen, so hat eine einfache Häufigkeitsanalyse nach obigem Beispiel bei ausreichender Textlänge wieder eine reelle Erfolgschance. Er muss jetzt "nur" noch die Länge des Schlüsselwortes herausfinden.
Um dies zu bewerkstelligen kann er sich auf die Annahme stützen, dass häufig vorkommende Buchstabenpaare (wie das deutsche "ei") und kurze Wörter in einem langen Text mehrfach auf die gleiche Weise verschlüsselt wurden, es also Wiederholungen im Geheimtext gibt. Diese Wiederholungen muss der Analytiker ausfindig machen und gleichzeitig die Länge der Wortzwischenräume zwischen gleichen Buchstabenpaaren notieren. Diese Beträge zerlegt er anschließend in ihre Teiler und listet sie in einer Tabelle auf. Im Idealfall sticht nach getaner Arbeit genau eine Zahl (ein Teiler) hervor, welche mit der Länge des Schlüsselwortes identisch ist.
Die dahinterstehende Idee ist recht einfach: Wenn man ein sechsstelliges Schlüsselwort hat, so ist es in einem ausreichend langen Text sehr wahrscheinlich, dass ein häufig vorkommendes kurzes Wort wie das deutsche "die" mehrmals im Text verschlüsselt wurde und durchschnittlich bei jedem sechsten "die" die gleichen Geheimtextalphabete benutzt worden sind. Zählt man nun die Abstände zwischen gleichen Wiederholungen, so könnte man beispielsweise die Beträge 18 und 24 erhalten. Bei anderen Wiederholungen wären 6 und 42 denkbar. Alle Zahlen haben einen Teiler gemeinsam: die Sechs.

Praxisbeispiele
Bis jetzt wurde ein allgemeiner Leitfaden für die Praxisanwendung gegeben, welcher darauf wartet auch angewandt zu werden. Auf der Seite Malcoms Lehrjahre sind einige Übungsaufgaben zu finden, an denen jeder sich versuchen kann. Vorher sollte aber die Ressourcenseite Malcoms Munitionskammer angesteuert werden, da die dort zu findenden Hilfsmittel das Leben um einiges erleichtern können.

Analyse

Auch wenn die im vorhergehenden Text beschriebene Häufigkeitsanalyse, wie der Name selbst schon zum Ausdruck bringt, analytische Elemente besitzt, so liegt der

Schwerpunkt deutlich auf der statistischen Arbeit. In dem vorliegenden Text über analytische Angriffsmöglichkeiten steht eine andere Arbeitsweise im Vordergrund. Es wird nicht hauptsächlich der Geheimtext zur Analyse herangezogen, sondern vielmehr der verwendeten Algorithmus (der zu diesem Zweck logischerweise bekannt sein muss). Dieser wird nach Fehlern und Schwachstellen analysiert, welche einen Angriff ermöglichen. Aber auch Fehler, die eine andere Angriffsart erleichtern, sind heiß begehrt. Entdeckt man z.B. eine Ungereimtheit im Algorithmus des Schlüsselgenerators durch welche im Vorfeld eine große Anzahl Schlüssel ausgeschlossen werden kann, so könnte der Fall eintreten, dass eine noch vor kurzem aussichtslose Brute-Force Attacke in den Bereich des machbaren kommt (kann man einen Schlüsselraum von 2^{64} um den Faktor 2^8 verringern, so hat man nur noch mit 1/256 der ursprünglichen Schlüsselkombinationen zu kämpfen, was bei diesen Größenordnungen Welten ausmachen kann).
Weitere denkbare Fehler sind schwache Schlüssel, das Vorhandensein sprachbedingter Regelmäßigkeiten im Geheimtext (sprich: keine ausreichende Erzeugung von Diffusion und Konfusion), eine eingebaute Hintertür, usw.

ein kleines Beispiel

Als Beispielalgorithmus einer möglichen Analyse soll eine polyalphabetische Substitution, genauer gesagt eine Vigenère-Verschlüsselung dienen (leicht abgewandeltes Beispiel aus polyalphabetische Substitution).
Zuerst überprüft Malcom den Schlüsselraum auf schwache Schlüssel, wobei ihm sofort drei schwache Schlüsselgruppen auffallen. Zur ersten Gruppe gehören alle ein Zeichen lange Schlüssel, da mit diesen eine polyalphabetische Substitution per definitionem nicht möglich ist. Zur zweiten Gruppe schwacher Schlüssel zählt er alle Exemplare beliebiger Länge, die ausschließlich bzw. übermäßig häufig die gleichen Geheimtextalphabete vorschreiben (z.B. durch den Schlüssel "AAA"), da man als Ergebnis im Extremfall wieder nur eine monoalphabetische Substitution erhält. Und schlussendlich sind Schlüssel mit Wiederholungen und Regelmäßigkeiten zu vermeiden (etwa "ABCABC"), da diese durch ihre scheinbare Länge ein größeres Sicherheitsniveau vorgaukeln, als dies tatsächlich der Fall ist.
Bei Betrachtung des Algorithmus selbst fällt auf, dass der gleiche Schlüssel immer und immer wieder verwendet wird. Bei kurzen Schlüsseln könnte dies zum Verhängnis werden, da auf diese Weise häufig vorkommende, kurze Wörter (wie das deutsche "die") mit dem gleichen Schlüsselteil verschlüsselt werden könnten. Im Idealfall (für den Analytiker) kann man durch diese Regelmäßigkeiten auf die Schlüssellänge schließen, den Geheimtext in seine Einzelbestandteile zerlegen und eine Erfolg versprechende Häufigkeitsanalyse starten. Ein weiterer Schwachpunkt ist die den Geheimtextalphabeten innewohnende Regelmäßigkeit, welche es ermöglicht mit nur einem sicher bekannten Geheim-/Klartextbuchstaben und der Position des verwendeten Buchstabens im n Zeichen langen Schlüsselwortes, 1/n des gesamten Geheimtextes zu rekonstruieren - dies ist natürlich nur möglich, wenn man sich genau an die beschriebene Vigenère-Verschlüsselung hält, ohne Modifikationen vorzunehmen.

weitere bekannte Analysen

Wer sich in die Materie vertiefen will und auch vor viel Mathematik nicht zurückschreckt, der findet in Teil V bei den Links weiterführende Informationsmöglichkeiten.

grundlegende moderne Angriffsarten

sechs allgemeine Fälle (mit sinkendem Schwierigkeitsgrad):

- Known Ciphertext Attacke
- Known Plaintext Attacke
- Chosen Plaintext Attacke
- Adaptive Chosen Plaintext Attacke
- Chosen Ciphertext Attacke
- Adaptive Chosen Ciphertext Attacke

Known Ciphertext Attacke

Bei einer Known Ciphertext Attacke steht dem Angreifer ausschließlich eine möglichst große Menge Geheimtextmaterial zur Verfügen ohne auch nur die geringste Vermutung des dahinter verborgenen Klartextes zu haben, weswegen diese Attacke auch unter dem Namen "Ciphertext Only Attack" bekannt ist. Je mehr Geheimtext der Angreifer durch Abhören, Diebstahl oder ähnliche Vorgehensweisen erlangen kann, desto erfolgreicher ist ein statistischer Angriff. Diese Angriffsform ist in der Regel in den meisten Fällen möglich, da der Geheimtext verhältnismäßig leicht zu beschaffen ist, jedoch ist der eigentliche Angriff einer der schwersten und aussichtslosesten, da der Angreifer so gut wie keine Anhaltspunkte besitzt.

Known Plaintext Attacke

Bei der Know Plaintext Attacke besitzt der Angreifer wie im obigen Beispiel eine bestimmte Menge Geheimtext jedoch inklusive der korrespondierenden Klartexte. Auf diese Weise kann er versuchen mit analytischen Methoden Zusammenhänge zwischen beiden aufzudecken und eventuell sogar in den Besitz des geheimen Schlüssels gelangen. Im Übrigen kann dieser Angriff auch schon durchgeführt werden, wenn der Angreifer gewisse Klartextstücke nur vermutet, wie z.B. die Datumsangabe in Briefköpfen.
Der bekannteste Angriff dieser Kategorie ist die lineare Kryptoanalyse.

Chosen Plaintext Attacke

Bei einer Chosen Plaintext Attacke besitzt der Angreifer die Möglichkeit, selbstgewählte Klartexte mit dem geheimen Schlüssel zu verschlüsseln, ohne jedoch Kenntnisse von diesem zu haben. Auf diese Weise kann er für die weitere Analyse wertvolle Textpaare erzeugen, indem er besonders charakteristische Texte verwendet, z.B. nur binäre Nullen oder Einsen und so - je nach Situation - eventuell sogar den geheimen Schlüssel erlangen.
Eine weitere, ebenfalls sehr große Gefahr dieses Szenarios ist die Fähigkeit des Angreifers, verschlüsselte Texte, welchen in der Regel ein hohes Maß an Vertrauen entgegengebracht wird, in Umlauf zu bringen, um dem Gegner mit gefälschten Informationen Schaden zuzufügen.
Bekanntestes Beispiel dieser Kategorie ist die differentielle Kryptoanalyse.

Adaptive Chosen Plaintext Attacke

Die Adaptive Chosen Plaintext Attacke ist im Grunde mit der Chosen Plaintext Attacke identisch, nur besitzt der Angreifer über einen gewissen Zeitraum hinweg die Möglichkeit, zusammenhängende Klar-/Geheimtextpaare zu erstellen und somit seine bisher gewonnenen, analytischen Erkenntnisse bei der nächsten Auswahl mit einfließen zu lassen, indem er die Klartext entsprechend anpasst (engl. adaptive - anpassungsfähig).

Chosen Ciphertext Attacke

Bei der Chosen Ciphertext Attacke besitzt der Angreifer sowohl einige Geheimtexte als auch die Fähigkeit, diese zu entschlüsseln (beispielsweise indem er auf eine automatische Dechiffrierapparatur Zugriff hat, was der Fall ist, wenn man die Passwörter für die Programme speichern lässt), ohne jedoch den geheimen Schlüssel zu kennen. Dieser Angriff entfaltet vor allem bei Public-Key-Systemen seine volle Wirkung und findet dort verstärkt Anwendung.

Adaptive Chosen Ciphertext Attacke

Bei der Adaptive Chosen Ciphertext Attacke befindet sich der Angreifer in exakt dem gleichen Szenario wie in obigem Beispiel, jedoch befindet er sich im Besitz bzw. hat dauerhaften Zugriff auf eine Dechiffriervorrichtung, durch welche er in die Lage versetzt wird seine bisherigen Ergebnisse in die Auswahl neuer Geheimtexte einfließen zu lassen. Angestrebtes Ziel ist wie (fast) immer den geheimen Schlüssel zu entdecken.

Brute-Force Attacke

Brute-Force stellt eine der wohl bekanntesten Angriffsformen dar, was wohl mit der einfachen Handhabung dieser Technik zu tun hat. Die englische Bezeichnung "brute force" bedeutet wörtlich übersetzt etwa "rohe Gewalt" und das zu Recht, denn bei dieser Angriffstechnik werden einfach alle möglichen Schlüssel ausprobiert bis der passende gefunden worden ist. Da jedoch die meisten weit verbreiteten Chiffrieralgorithmen einen enorm großen Schlüsselraum haben, kann man mit gutem Gewissen sagen, dass eine Brute-Force Attacke in der Regel zum Scheitern verurteilt ist.
Dazu eine kleine Rechenübung zum besseren Verständnis: Wenn man von 128 Bit Schlüssellänge ausgeht, dann enthält der Schlüsselraum genau 2^{128} verschiedene Schlüssel, was in Zahlen in etwa 340 Milliarden Milliarden Milliarden Milliarden entspricht. Selbst wenn man einen Computer besitzt, der eine Milliarde Schlüssel pro Sekunde testen kann und man eine Milliarde dieser Computer verbindet, dann müsste man diesen Rechnerverbund im Mittel trotzdem ungefähr 5391 Milliarden Jahre lang laufen lassen.
Prinzipiell ist dieser Angriff aber jederzeit und bei jedem bekannten kryptografischen System anwendbar und kann zu Weilen auch äußerst effektiv sein. Einige Situationen und Variationen werden nun vorgestellt:

Dictionary Attacke

Wieso sollte man sich lange mit komplizierten Algorithmen und zeitintensiven Verfahren herumschlagen, wenn eines der größten Sicherheitsrisikos, menschliche Nachlässigkeit, oft die besten Erfolgsaussichten bietet? Diese Frage haben sich schon viele Leute gestellt und als Ergebnis wurde die Dictionary Attacke (übersetzt etwa Wörterbuchattacke) entwickelt, eine spezielle Form des reinen Brute-Force Angriffs. Bei ihr stützt man sich auf die Vermutung, dass als Passwörter zwecks besserer Einprägsamkeit oft real existierende Wörter und Namen benutzt werden (das Klischee des Name der Freundin-Passwortes ist nicht aus der Luft gegriffen und auch heute noch aktuell). Statt nun Millionen möglicher unsinniger Zeichenkombinationen zu testen, verwendet der Angreifer z.B. ein mehrere tausend Einträge langes deutsches Wörterbuch, ein Namensverzeichnis, usw. - diverse vorgefertigte Listen sind im Internet haufenweise zu finden. Der Vorteil liegt auf der Hand: Der Zeitaufwand für einen solchen Angriff ist relativ kurz, während die Erfolgschancen überproportional gut sind. Aus diesem Grund kann es durchaus sinnvoll sein, solch eine Attacke vor einem vollständigen Brute-Force Angriff zu starten (dies ist z.B. der Fall bei Login-Passwörtern einer größeren Firma, die von den Angestellten selbst gewählt werden können; bei automatisch generierten Passwörter für eine AES-Verschlüsselung scheidet die Dictionary Attacke jedoch aus).

eingegrenzter Brute-Force Angriff

Ein eingegrenzter Brute-Force Angriff benötigt ein wenig statistische und/oder analytische Vorarbeit. Zuerst heißt es nämlich, den verwendeten Chiffrieralgorithmus und - falls vorhanden - den Schlüsselgenerator genauestens zu untersuchen, um auf diese Weise Fehler zu finden, welche den Schlüsselraum eingrenzen. Wurden tatsächlich solche Fehler gefunden und man kann auf Grund dieser Fakten bestimmte Schlüssel(gruppen) ausschließen, dann könnte sich unter den neuen Umständen ein Brute-Force Angriff, welcher vor der Analyse als hoffnungsloser Versuch abgestempelt wurde, plötzlich doch als sinnvoll erweisen.
Ein kleines Beispiel: Würde ein reiner Brute-Force Angriff drei Jahre dauern, dann käme ein solcher wohl nicht in Frage. Kann man aber die Anzahl möglicher Schlüssel um den Faktor 1024 verringern, dann bräuchte man für einen eingegrenzten Angriff etwas mehr als einen Tag - ein Brute-Force Angriff käme somit wieder in Frage.

sonstige Angriffsmöglichkeiten

In der Kryptoanalytik ist es wie in der Liebe: Jedes Mittel, welches zum Erfolg führt, ist erlaubt. Ähnlich wie die enorme Variationsvielfalt der Kryptografie gibt es auch in der Kryptoanalyse eine Vielzahl denkbarer Ansätze und Verfahren, welche allein der Menge wegen nicht alle Erwähnung finden können. Einige wurden jedoch herausgepickt, da sie entweder bekannt, raffiniert oder beides zusammen sind.
Auswahl einiger spezieller Fälle:

- Differentielle Kryptoanalyse
- Lineare Kryptoanalyse
- Man In The Middle Attacke
- Timing Attacke

Differentielle Kryptoanalyse

Die differentielle Kryptoanalyse zählt zur Klasse der known plaintext bzw. (adaptive) chosen plaintext Angriffe. Sie richtet sich ausschließlich gegen rundenbasierte Blockchiffren, welche auf konventionellen Chiffriertechniken, wie z.B. einfachen Verknüpfungen, basieren. Zusätzlich zu möglichst vielen Klartexten mitsamt den dazugehörigen Geheimtexten muss der Chiffrieralgorithmus bekannt sein. Die Analyse funktioniert folgendermaßen: Zuerst werden zwei Klartexte ausgewählt und die Unterschiede zwischen beiden festgestellt. Anschließend werden die zwei korrespondierenden Geheimtexte betrachtet und ebenfalls auf Unterschiede hin untersucht. Anhand der Ergebnisse (wie sich die Unterschiede zwischen den Klartexten in den Geheimtexten fortpflanzen) und der Funktionsweise des Algorithmus wird nun versucht Rückschlüsse auf den geheimen Schlüssel zu ziehen. Im Idealfall kann der Kryptoanalytiker weitere Textpaare selbst herstellen und diese eigenhändig seinen speziellen Bedürfnissen anpassen (adaptive).
Eine kleine Anmerkung am Rande: Der jahrelang gültige Data Encryption Standard (DES) wurde teilweise von der National Security Agency (NSA), einem amerikanischen Geheimdienst, entwickelt, welche sich vor allem an den S-(Substitutions-)Boxen zu schaffen machte. Aus diesem Grund vermuteten viele Leute eine von dieser Organisation eingebaute Hintertür, welche ihr eine schnelle Entschlüsselung ermöglichen sollte und die Sicherheit dieses Verfahrens herabsetzt. Fakt ist anscheinend aber, dass die NSA die S-Boxen optimal gegen eine differentielle Kryptoanalyse abgesichert hat, welche zu dieser Zeit jedoch noch überhaupt nicht bekannt war. Ein für die Kryptologie typisches und wichtiges Beispiel, wie manchmal mit Entdeckungen umgegangen wird, um sich selbst Vorteile zu verschaffen, aber auch gleichzeitig ein Beispiel, dass die NSA nicht immer nur zu ihrem eigenen Vorteil handelt.

Lineare Kryptoanalyse

Die lineare Kryptoanalyse zählt zur Klasse der known plaintext Angriffe und wird normalerweise eingesetzt, wenn der verwendete Algorithmus nicht bekannt ist. Der Kryptoanalytiker versucht durch das Beobachten der linearen Zusammenhänge zwischen dem ihm unbekannten Schlüssel, dem Klar- und dem Geheimtext Rückschlüsse auf das verwendete Verfahren und eventuell sogar auf den geheimen Schlüssel ziehen zu können.

Man In The Middle Attacke

Der Man-In-The-Middle Angriff zählt zur Klasse der aktiven Angriffe (der erste in diesem Werk), bei welchem sich in die Kommunikation zweier Personen, Alice und Bob, eine dritte Person, Malcom, einklinkt, die Nachrichten abfängt, eventuell seinen Bedürfnissen entsprechend manipuliert und anschließend an den rechtmäßigen Empfänger weiterleitet. Dieser Angriff ist jedoch nur möglich, wenn beide Kommunikationspartner keine Möglichkeit haben, den anderen eindeutig zu identifizieren. Dies wäre beispielsweise bei Brieffreunden der Fall, aber auch bei den moderneren Kommunikationsmöglichkeiten sind einige brisante Situationen denkbar.
Dazu ein Beispiel: Alice und Bob schicken sich ihre öffentlichen Schlüssel, welche Malcom aber abfängt und gegen zwei ihm bekannte austauscht, während er die Originale behält. Alice und Bob wiegen sich bei ihrer Kommunikation nun in Sicherheit, während Malcom jedoch jede weitere Nachricht ebenfalls abfängt, mit seinem einge-

schmuggelten Schlüssel entschlüsselt, eventuell manipuliert, mit dem richtigen Schlüssel wieder verschlüsselt und an den rechtmäßigen Empfänger weiterleitet. Dieser merkt, wenn der Angriff sauber ausgeführt wird, von dem ganzen Vorgang nichts und verlässt sich weiterhin auf den (zu Recht) guten Ruf asymmetrischer Verschlüsselung, während Malcom diese geschickt und heimlich umgeht.

Timing Attacke

Die Timing Attacke zählt wieder zur Klasse der passiven Angriffe und versucht durch das Messen der Zeit bei kryptografischen Operationen Rückschlüsse auf den Schlüssel zu ziehen. Unterschiedliche Chiffrier- und Dechiffrierzeiten sind beispielsweise bei Verfahren mit Tabellenzugriff denkbar, bei Potenzierungen, usw.
Bekannt wurde diese noch recht neue Angriffsform vor allem nach dem erfolgreichen Knacken von Produkten der Netscape-Reihe.

Malcoms Lehrjahre

Auf dieser Seite sind einige Links zu Übungsaufgaben (in der HTML-Version) zum Thema Kryptoanalyse zu finden, an welchen sich der interessierte Leser versuchen kann. Ich freue mich über jede E-Mail an chronyx@paramind.org, welche die jeweils richtige Lösung enthält!

monoalphabetische Substitution

Aufgabe 01/01

LUPOXGGEBUEZEXQWUBVHZVEXQWXUPXVILVAQXVIBJWXILaXGIXFVLXDOPWSQGT
XBXGPXBVXPXBVQVIXUTSBJPWXVJXEQGWPWLJXPXBVEXPZVIXGPKGLXDOWBJXPTX
PWJXEXVYZUUXYLGIXPJXGXIXPQVIIXGLQTGXJQVJBVOZEEBVJXVAXBVXVIXEBUEZYL
GPXOGGXBDOQVIPXOGLEPZVIXGUBDOQVIPXBWXGHZGPXDOSBJNLOGXVKUZXWSU
BDOHXGPDOYQVIXVQVIQVXGYLGWXWSQGQXDAJXAXOGWYLGOLWWXFLVBFLQXVU
LVIVBDOWLQTJXOZXGWPBDOQXEXGBOVSQHXGYQVIXGVIBXGXBDOWQXFXGIBXXG
HZVPXBVXVTLOGWXVFBWJXEGLDOWOLWWXYLGXVFBWWUXGYXBUXSQXBVXGUXJX
VIXBFLQXVULVIJXYZGIXVQVILUUJXFXBVJULQEWXFLVYLPBFFXGIBXLUWXVUXQWXL
QDOGXIXVFZDOWXVILPPIXGEQXOUHZVEXQWXUPXVIHZUUXGPWZUUXVPXBBVIXVX
VPBDOIBXPDOLXWSXOLXQTWXVQVIYXVVILPVZDOVBDOWTQXGPXBVXVGQTJXVQXJ
WXILVVPWLQVWXFLVQXEXGIBXQVJXEGZDOXVXUXEXVPAGLTWIBXSXBWEUBXEVBD
OWPWXOXVLEXGLQTOXGGVEXQWUBVPDOBXVPBXYXVBJYBGAQVJLQPSQQXEXVFB
WVXQVSBJYLGXGVBDOWLVIXGPLUPFBWTQXVTSBJLUPXGVXQVQVIVXQVSBJYLGPLJ
WXVIBXUXQWXXGPLXOXVZDOJQWLQPLEXGQVHXGLXVIXGWYLXGXSQWGXTTXVIXG
JXYXPXVFLVDOXPDOQXWWXUWXVIXVAZKTQVIFXBVWXVILPPXBSQHBXUIXPJQWXV
XPPXBXBVTLDOQVEBUUBJILPPNXFLVILVPDOXBVXVIXYBJXNQJXVIQVIZEXVIGXBVVZ
DOLVJXEUBDOQVXGPDOZXKTUBDOXGXBDOWQXFXGEXPBWSXVPZUUWXILTQXGYB
GIXGEXSLOUXVFQXPPXVPLJWXVPBXXPBPWVBDOWVLWQXGUBDOQVIYBGIXBVPDO
UXDOWXPXVIXVXOFXV

Aufgabe 02/01

ACASTCQMBKIMRMBIMCJMQDMCSVBQALPJAQQLQJRAQQDLMGHIMMCFBGVLQJLQI
MCFLGVIMJMQDTJMQVTDDBIFVADMQMBQMQKMBGVIMQFGVCBIILQJFMKDFIPLMC
MBQMQRAKJKAMLPMCFBQJBVCMPLFFFWLCMQQBGVIKMBGVIZLKMFMQJTGVQBGVIR
MBIOTNSBWPMKHCMLZIMMBQMYLMKKMJMQWPAJLQJALPJMCQAFFMQMCJMFAVMC
RAFMCFLGVIMBGVVADMJBMZMBGVMQCBGVIBSSMKMFMQFASIMMCZLFBGVPCTJTBF
IZLNDMCSSBWPMKSMKALPMQRAFNASMCJTCIRTVKSMFMVMQVADMQADMCMCHANA
LPJMNFMKDMQRMSZLCLMGHLQJBFIDMCSADSMSAQSMQACASTCQZTMSMCIMMCVAI
IMFMKDFIJMQRLQFGVZLJMNVTGVFBIZZLSMVMQJMQQMCVTPPIMJTCIMIRAFZLFMV
MQJAFBVQALFFMBQMCCAIKTFBSHMBIVMCALFPLMVCMQHTMQQIMJTGVJBMZMBIJC
AMQSIMWKTMIZKBGVFWCAQSMCOTCAQLQJCAQQIMZLNSBWPMKLMDMCJBMSCTFIL
PMQVBQALPAKFMCJAQQALPJMNVTGVFBIZFAFFDKBGHIMMCFBGVLNADMCJBMFTQQ
MFGVBMQOMCJLQHMKILQJJBMRMKIOMCFGVRTNNMQLQJMQICLMGHIMCRAQJIMFB
GVOTQQTCJMQCBQSFLNRBMJMCQAGVQTCJMQZLCLMGHLQJMCFAVQBGVIFAKFJBMP
MCQMQDMCSMQLCJTCIRTFBMBQSAQZRMBIMCPMCQMKASMQFAVMCRBMJMCLNMB
QMQSCTFFMQOTSMKOBMKKMBGVIMBQMQAJKMCJMCBQRMBIMQHCMBFMQKAQSFA
NZLCMCJMQBMJMCFGVRMDIM

Aufgabe 03 / 01

OVOOVQCYEGSYQGSLEFQWFCPTTAZYSGMSQWSHHFQGSCZSLJXLSLPLFEGSTVAZXN
SLBFAZSXWSLTAZCFPSYQWQXAZVQWSHLFTAZWFZVQSVCSQWSQGLFYHNSPFQESQ
TSVVQWSQSLTSVGWSLELXTTSLVGGNSEFQQTXCFQESJSLTYQRSQESBSTSQBFLBVSV
HPCYEMXEWVSWYQRCSBSCGFQVZHJXLYSNSLYQWCFYGNLFYTGSWSLBVQWVQTSV
QSQXZLSQSLRXQQGSQVAZGTTSZSQFCTWVSPYQRSCQWSQTGSLQSYQWPSLQMYTS
VQSLLSAZGSQBXTVAZWFTESNVLESWSTTYSWSQTSLTGLSARGSESBFCGVESTAZFGGS
QWVSTVAZJXHZVHHSCFNZXNSQHYSWSJSLTYAZGSSLWVSMSVGSQYQWFNTAZQVGG
SVZLSLPFZLGQFAZMYLSAZQSQFNSLTSVQESWFSAZGQVTBFLTAZCFSPLVEYQWJSLBV
LLGMYSLTGBFLSQTVSXZQSFQMYZFCGSQSQGTSGMCVAZTAZQSCCESLVGGSQYQWW
FQQZFGGSSLVHHXLESQELFYSQSVQSQNCSVAZSQEXCWSQSQTAZVHHSLESTSZSQYQ
WTVSBFLSQMYWSLTGVCCSQTGFWGYQWWSHELXZFYTFYPWSHNSLEESRXHHSQYQ
WRFYHBFLSQTVSVQTSVQSHTAZYGMFQESCFQEGWFBFLWSLESPCYSESCGSTAZFGGS
QBVSWSLYHZVQBSEESMXESQYQWWVSHSQTAZSQJSLEVQESQJXLFQETGFNSLEFQW
FCPZFGGSVZHNSLYZVESQWMYESTOLXAZSQYQWSLBFLVQSVQSHBVQRSCSVQESTAZ
CFPSQHYSWSFNSLYQLYZVEBFLTVAZYQWSYGCVAZWSTRXHHSQTYQWESZSQTNSBY
TTGYQWWFTTHFSQQSLTOLFAZSQYQWEFQWFCPNSPSZCSEFNYQWWFQQLVGGSQT
VSBSVGSLLVGGSQWYLAZWVSQFAZGWFTBFLDSGMGWVSMBSVGSQSVQWVSWLVGG
SQFAZGTSVGSLVQWSQTGSVQESNCVARGZFGGSWVSTSFNTAZSYCVAZSSLVQQSLYQE
CVSTTVZQEFQMBFAZBSLWSQYQWSLSLTAZFYSLGSYQWWFTESLFSYTAZWSTBVQWS
TBFLHVGSVQSHHFCSLPYSCCGJXQWLXZSQWSQTGVHHSQ

polyalphabetische Substitution

Aufgabe 01 / 02

GVULBUEJNHOZHRHACIGWWLZGIEMZPVCBGYAOAIVOIXKVTHLQAEELLCATMLZNUG
GYQJRYCJSZREQSQHIKJPAOOIAOIXKVTLBCIBCIPJEIEAMXHEKRMIAIEOQZAIRIIMCFF
LAWAJKJAWIFUIWJKJVVZZBFVZJMIUILJMUKGMWMCVDBNAEIQCMMFCQDKVRUKUO
XHRVZGNCFQRQZSTTHKFEIEYIXKZPNKMYGVVLZCFFLZYOJFUAYNJUWWJFZPNLISYK

QIXKZPNLQDVINLMOGGTIDLJGTCGAKQYMIEORSWDTJHHSZMRKSAAABGUNDNXGYN
DRVYHTGSWVWOJPYGNXMSXRZONMYCJSVTKCJSNHJEZLCUSKJUKIGKWDNETJZQKC
FPMQBIGUWWJFZPNQNDEMLZIECMLGGOXILZFICCULVTKCJSGIEWEUVICDVUWEIOP
UZMFDPTAUEMTWYSDRUVWSKCAVZEUGKNDLVUUMOBZQZVZTJVHBIUBGYWMAEILJ
JOBQZQNCYKJPOEEOVLZLCQCMMCCQJSKAICTQIDGCAKCEJRIFKGGROZZABKUOKIEI
WQIGWNVWYIEIWQIGFHKMVTYRSCBIEUWWMTJRVZOSTCUVZRGQWLMEZRYWSYH
WHVOEEEVUKUKGYZVIEDVEWOFMZZZMRKSMMRVOVBZATELANRVRLIOEITMKOOLV
LZMSREPAZCLTPBTSVTPIGSJJLTGSJOHZOCRTKANMJUTBKSDWYNVTKCJSNNZHMMM
SFEPIGHRERQIGJUOANLJWWMMCFOWCOEIUDQOCYUFVALFQKQIGKEWQKTVNUMO
TICJMMTKTHKFIEIAZJJRPZCHTJWUQSVZTPQQIIWZOZNVTHBJRNCWEZBXTHJWEIYL
PJIJYPVBAKGZ

Aufgabe 02 / 02

AR^[K_eS_eeYkcW@S_Rfa][_ZNVWXRbnKZbO_UepYOObV`MXSKRejYV`^R^=UcRO
_eeMUS\8bpVe]Q_Q]OVUOT^\Xkc$DUeTdbK_[\1cgZa_^XRTSR^`IYb`RbYUeSXdYiJ
{RK[^Ju]Z^RjlSZ\NRckJV`=aQXZUc\PXJIYZ_RcjK]VS[d\X]SQb^^K_x`RbjZVQUaU?O
_bO_dlKcSX\T\XRSR[\`IYS7NcjTRVWR^`TUWO9Q^KgS\`Uk`eeO_T\T{1RVV]XV\S[
U`TV[`RbkXVbLNb\TKSSaQlLhOXQ[eGTYO[jlQE\XR^%/_RO_@iGiW]QQiLUOX[UiYeP
OVF\XUOMUd\O_@SPXkKcRK`;eGTYO[T\XGS\`S_RfS]`Uc[_UO_\X[SSXwQ\N_ZSPX\
O_SW5QlYUc\PXj[TV_[WjHVTOU\%*RR__S_]Z`NWU[Kc>\VfXZaS\`_e]VW^Rb_O_W
R_UGXZdKacgNRS\RWXXR\^VUiZ{aYYQeMVaSRc`IYO_ST\XcWMUd`MV\=RYkKUS]4
UjKehO`QlLYOOYd%*ZS]X\`TXbS[T\XEVO\b`KkeK_cZN`SXb^[MfbtOb`TXbKOUiJV\
X\S_KZ\STUjIYeO_g`KXSXQUGX`PVR]\SZb]VS_rD]aVb[JZSOVW\TV>\VfXZd^RNUiK_
WMUdbU_aO^e\TeUO`S_[Vbdaz[GdWO]b`TkWZVUcRg]XQUd9eOKaf\X]S^gdnKcRO
[[fK_\^ReeJhWO\R\TV`aNU_TeYK[^X[TVNRbJZRO^RYeK8SPNXiHVRObd\T}EOVd\X
YWXUQkJZS@Rb^G_UO[X\OeUOgU`MexNNcjLRa^WU[Kd[O[cZN]WMUUJ_dbOZ[fSa
`YZYkZZS\OQiOdb_[TjK]P]aWlZgS\`d\I\bO5YeZV`^bUiK_[SaT\XKSSaUeZUSMXdnKc
RO[|;GUc\PXn[V`NR^GXZdKa`\Xd]XR^eOTV^[eiJf`MUT\TDbKNdXTX`OVVYGcx]\^[
Kc\KbS_Jf`MU[iO^WXR\cK<`KRVkK{eOYS_KdWMUJlMR\Qge[K_aO[c`H]SX6^]Uc[Ka
YfTV\`RbjIYOPSUeQ`SX[d\T}/L`S_RZS]`UeJ\]O[^\T8SQ[UiJV`=PXc[Va]R_O_bO_\\
Mf\Qge[OVaOZD_K^OOV^\K_baNV]TV\NR7\MV\P_Q^KdbOY\\T+EO_RiO_U^RYeK_
9\V]`TVZVR^[Gkct`U`TV\=PXc[Va]R\q[YWXaUiRVUO[/

Aufgabe 03 / 02

yQcU]Zv3@y36a4U^@K6ReZNJ)bjcOOpUdfaK6[`^YZ3[_XVKv3@aXJk@ja>)r}9<tY8
U\X^K-^;[WS9^YWKZ-_^\\U8_[[VR>ebBOX*eUS_T+zTM]Y7YST`U6Q\XOS-
]9Z^K6^U`KR7cUT\K*VYfSK2dRQaG)Xb`RG8|D/:[)RU^XO1]dPKH)YTUO'9VWMLKNe
]0KZ)^f[XK-^UY/T(UYZOO2jUXXKuUWYOT8Uja^K-
\UZ_T(TYQ]K%]QZNK6U^1XJ)gYQNK6[_^\K/dja]G1]UZd[*eUSOTP9@zLK6^YYWZ,Y
U^LK-TYQ+[*WQNOp(YU_OK-^jQVT)^4M^K2cUSWK2dUP_X'Xc:OZ>jeXOO8U^x,K-
^QQRK6Ub.OZ6QST^[2WcORK-^d@-
6%\\QXU)dYSOTceVSKH)^eQLK6^UTWK2jeWYK2^UZtG&UbPSK7Yc`X[6Y^WVK-
^UZ8K8jgQ\Q)^f[\Z)Y\TKL8|2QSJ)^1a]S%ccQXJ-
UTM]/2dU^XK8Y^faO7SXQXG2WUZYS1U^TKZN[QZXY)\R_^K-
^UUXL%SXQ\<3bWMXM;YUPSKwURQ\Z6QWaXM:_^0KZ)^zf_K-
^U^ZX3WbMWS8USTXO7SXQX.)bQa]L3bTQ\[2WgQ\J)^|@-
69^T5:T%UXQ\H)dbMMN8Ud0SKceVSKH)TU_>)rYc`OYNUYZOT:UbNST(e^S]U6YUZ

^O)bdQX*%dUZ^X%^cROX>eU^WU)W\UMN)^|@-
6&Ud^KI,dU`NO)4Q`OT%\cQST)^C`\U1f_Z,_8UcaXJ2YST^G0cRQ]Z-
]]`O(0_UOUKP4YQd[9URQ\Z6QWQXJ)^5UXN)YdQX])bTQXG0cCQQS)^dQLK>UYOR
T)d|.OO(YU_OS:UbNST(e^S]U6YUZ^O)bdQX9)bfUMKNQST^K8D3<K[*JebOX0QU_]
O+[UU^pfQdQXL0ec_UU2db[VR)e^P/X6UYORH%b[QSZPD3<SY8Y^POXnQWQOT8c
`^OI,U^PO*%dUZaO)TU^RK6je_^K0\UZL`;|^Q_G2jeRYX(UbZt])^^_SK:UbXYX)^z
NOY'XQQNO+d_POX%ecPOS>UY`VO'XUZ<G,]UZQK*Q\XOT7Y^Pv;1TYQ]`9f_XVH6
Y^SOTNgUU]Y8D3<TK(U]f_[)RU^^X%WUZOTdidQOO2UCQ[[)^jZ_S1Ubf_rfQc@-
6(Uc1WV*QUZQK6c]a]Y(U]-LY)^TQ\K-
^U.OY8QU`SM9^Wrj'e;pw+I/^_cVK(WUYOT8wVO2^U^RG0RUUXK7RU_^O1]dQX
@)YdMLY'X^U^Z>ebqMQ7SXUMQ)^|5]Z(YU_XO'XdPOXhQ\Xt])bTQXJ-
U4M^K2UbZO[8WU_MN-S[`v*)bEYQG2W]U^H)cSTKK(YW`OTfQdQX]-
bTP_X'XTQX@9cQ`dK-
^U^:X9UV__S1UWQRG2TXMLZP5^`NK'[dPKYv3@POYg]`RKK2WU^]K-
^RQ]I,QUPSM8Uc?OM1U^`tJ%^^cSX(TYQ]K7CUSWK2dUUXL%SXbOX;_bROTP4U^
+H7U^POX;YbPNG2^TUOY)cCQQS)^dZO[:Ub_OT(U^vLO7TYQOT8c`^OI,U^PO()cd
MOZ-WeZQK-^d^SL*d|0SKceVSKH)TU_36-
cdQ]p)Y^QX=)Wv^Y[8UwbYScRcQXJ)bzP_X'XTM]4)djcOX/XYZaK+zjaW+1`VMOT+U
bf_L-
^TQXrk@UUX\)bRUXJ9^W_VU7Uc<\U8_[[VRP4Q_LK(Ue`OJ(Qc0KZ)^`MUK8UfQ\R3
bUZQK,U^WYK2^UZYJ)bfQ\Y4QU`OZ%^[[WS)^|0SK)^d_ZX)SXQXJ)^4M^K2cUSW
K2dUWYK2^UZKR7_YZPG0cSTOXtUYTOT*_\SOH)Y]1WV*QUZQK6Q^`\K*VUZv*-
UBUMN8YW_^K0\eZQJ)bUUX`)\^QX9)W]QXZ)]e_]O2TYQ]K16QXVp:_^POT,_UTOX)
^CORO'XdQXJ)c??3qo_TQVR7eUNOX2_]YOT;UbPOTPC_SOY)XUZOX+QUZdK2cYOR:
e@eZN/rRUUOO2U]F_Y%]]QXY4YUXST-
XbQX,9^[`SU2U^x>G8cQQMN0YST]Z)\\`36.U^Q=I,\eQ]Y)\[[WV3^UZ^K(QbvK[*T
YQ]O'X]U^:e@QaPH%eUZVG)cc`v.-
UbbOX)Y^UQK2TYQTK;UYXSM)^@^YZ3[_XVK|efQ\R%Uc_SM/UY``U2D3<_T(UVRO
Q8YfQAK+VYZN[2WTQ]/rTe^MNk^d^Kq9^T5XZ)b^Q^N-^gQQrfQc@-
6OCUSWK2dgU\J:_]5:K-^WQUG4`cQVZ9^T_OZ>dTQW*%dUZ^K-
\cQST)^UUQK2U^5:qjUQPOX:_bMXru_RMVJ(Qc0KZ)^`MUK8cUUXK2f_^QK7UXQXK
25]\PG)^WQ\K6bUUMN8XQ`t]-
bTPOX)^d_ZX)SXQXJ)4Q`OT8UYXWO8TUY>)r{CQQS)^dbYT(UbQXZ7`bQMN)^TaO
H)bWQYX(^U`OTrb_`YQ3_MN-SX``K6gU^^K8|

"Mix it, baby." (hardcore)

Aufgabe 01 / 03

*#A[(%Ts/Dx!!!+X]SgWa(vC2Q_Gt/|vC2cTG9^Øw|fSZgWz'vC2cTG9^|s/dVygUAZS+
Ss/V{~?V!PaaJ9^0s0t1dK)#()1t1RQ'(E8`P]PgW1({Ø+s/C#}?VØPaWPx.,8`gO`gUAZ)
-
_]q%)#|8`UOOG9^(xC2cTG9^{w!`{q%T){8`dVQgWm){/Xcq%$Ø?Vc_Rq%!*/wØt1
bR$)?TC1cTG9^}s~Ts/QyA[(%Ts/Yt/|&0|s-gVZ'vC24[F9^,s&Syq%`~-
8`dVQTxA[t!t1XKz#-NC2OZF9^-
z!bSq%,z,8`[WSJ)I?TC1.ZF9^_#Øt1`C,A[(%Ts/N|!Ø(It1aJt/?V&ds/Yt.?V$_^P{9^x!
Øt12QwA[v&fWPGwA[(%Ts-
gV'!y%ds/H'*&8`dVQgW}x&(^S`UCAY8_/]PgWa(vC2QMNØ~{8`dVQgW'!y%ds/%t4

E8`P]PgW/ØwC2RMT~)|'0t1TG9^zs)[SPgUAZa&VVao9^X!Øt1aJxA[w3T]UPzA[s+Ss/
V{~?V*_aZK#!?V4TaQgW/ØwC2TUT(/?VØPhygUAZS+Ss/($}?V0PWPm9^dw1t1aJx-
|8`QSq%tA[x&b[MOx)-
8`X]q%)#|8`]WPU)A[##t1aJxAY8_gOaG'.E8`P]PgW'|(C2WagW}!*&SSq%)#|8`gOa
G'.?V#b^YgW/ØwC2fMVx-
,BC0s.!#}?Vc_Rq%!z{wC2cTG9^}{/]OYG#/E8`P]PgW}!*&SSPgW/ØwC2fMVx-
,8`gVUE{A[+!bSq#9].!ØTaq%)#|8`UW_Ot(|!1t1RT$(?V1WSq%,z-
w/cs/Y{$zzC2fQTxA[s}_eQgW/ØwC0s.H|-
&s*T]a{9^x!Øt1UV9^0s0t1`QCAY8_/]PgWa(vC2QMNØ~{8`dVQgWØ!&*P[QP)A[Z!P
eQPCA[S+Ss/V{~?V!fSZK#!?V|^Rq%)#|8^t0YQ')!!$t1dG'~?V1WSq%(~z#+Ss/Ft4G
8^t0,PwA[Y,Ss/Ut${@C29QV9^-z!t1dC)~+'C2dZFx-
?V1WSq%{~x*!^s/DxA[y|dVQTx}?TC1c[Ix/Øw/t1bP)*?V,^Sq%%'xu!zs/C#}?V)Tcq
%)#|8`SafgW'x!Øt1MR%~x&Wt1MPwA[{1t/q$,z,8`c^ygUAZS+Ss/($}?V~PZXGwA[(
%Ts/F'4?V)P]PgW_x&1W*q%t){8`dVQgW!x(%TaUPzA[(,VSaJx-
?TC1^RgW/ØwC2fMVx-
,8`ROXNx}?V%Ts/4xz,NC2OZF9^_#Øt1`C,A[(%Pcq%|/?V4Pbq%z*(vKt/q$T){8`5^
PgW.x{Øzs/-x/?V1WSq%xz+(%t1NT|)~8`U^_V{A[y/Pb`m9^-
z!t1TG'{?TC1hUGØ}!!$t1`Gx}E8`P]PgW/ØwC2T_W|/?V1bSQgW4!w)SWZI9^}&2Xcq
%tØ-w/t1TK(AY8_ZWZFAA[+%_bQgW.|wØt1UU9^!!C2WaUx'}@C2d]Q#A[(%Ts/Gt-
-zWt1MPwA[{1t1dC(A[',|s-gVZ'vC2cTG9^|s/dVq%u-
()$Wcq%y*+(%t1STt.,@C2OZF9^Øw/Qs/[|~%v&^Uq%(~|vC0s.Cy/|&C2VUU9^${
+Syq%t){8`dVQgW/+w!t1fKx'{{+Vs/H'0!(It1dJ$.|8`cSQF9^0s0t1UP9[?U&dbQNyG?
V|UcQT9^Ø{0t1WK#}S8`P]PgWa(vC2bMY9^-
z|ds/K)A[+|cs/I$*{BC0s.!#}?V1WSq%x1|!&^Uq%t){8`dVQgW((&+X]SgW2|&!t1aJx
A[(%XaPgW}x-Kt/q$T){8`5^PgW.x{Øzs/-
x/?V1WS_G9^ywC2ZUI{/,8`X]q%)#|8`UW_Ot(|!1t1[H9^-
z!t/q${~x*!^s/V$A[v&fWPG9^-z!t1PC.A[x/_[q%)#|8`^WSJ)V?V|^Rq%Ø~-
8`dVQO9^ywC2T[T9[?U0XUZUAA[s+Ss/H$-
?V0TO`Q#.E8`P]PgWØ(&C2RM[(G?V|^Rq%.~x&0*s-gVZ'vC2ZQV9^-
z!]s/DxA[x,bs/N|!Ø(0t1UP9^-
z!t1RK'(xØ!^cq%$Ø?V1WSq%{~x*!^s/V$AY8_VWcG9^%{$Wcq%*+(!C2cTG9^|s/d
V'gWz'vC2WagW2x'C2b[o9[?U]^Rq%Z*{8`]OPG9^-+,t1STxz-
8`[WSJ).T8`dVQgW!+w|dS_gW'!y%ds/V$A[&2[Sq%)#|8^t0PC.G?V|^Rq%)#|8`[S`
Ux-?V)XUTV9^-
#C2abNxA[(%Ts/P|!Ø(Wt1TG9^&sØTs/V{~?V0dO_U9[?U|[b[o9[?U]^Rq%Z*{8`cSa
gW/Øw*t1UP9^-
z!t1RK'(xØ!^cq%$Ø?V1WSq%{~x*!^s/V$A[y&fSq%Ø$~z1t/q$*+(!C2cTG9^|s/dVw
gUAZS+Ss/V$A[&2[Sq%$1|&C2cTG9^{s6t1MPwA[#3Taq%)#|8`^WSJ)G?V|^Rq%)
*?VØXeUFxA[(%Ts-
gV'!y%ds/H'*&8`dVQgW}x&(^S`UOA[s+Ss/($}?V0Pfq%)#x(C2WagW2x'C2U[QwI?T
C1.ZF9^-z!t1QXx)!!$t1MPwA[(%Ts/O$-'{+Vs/Yx-
|8`dVQgWØ()/dVq%wz2BC0s.!#}?Vc_Rq%(z!vIt17G)A[(%Ts/Yt/|&0t1NT|)~8`U^_V
{A[s}e]PC#/%-
C2cTG9^&#3X]SgUAZu/TOaW'~?V1WOagW#x(%t1XKy~E8`P]PgWØ(+)t1aJt/?V*Ph
q%y'28`PP[XxA[(%Ts/Gt--
zC0s.K#A[(%Ts/Q%~'8`UW_Ot(|!1t1[H9^Øw|fSZo9[?U]^Rq%Z*{8`RaQC)~{8`Va
QC)A[+%PZQUAA[s+Ss/G+~+-
C2ZUX|)~8`RaQC)0+wC2cTC)AY8_]^cG)#E8`gVUE{A[(%Ts/Yt/|&0t1NT$0~z1t1RQ'
/Ø8`PPbPwz'()iyq%tØ-

w/t1aJx$+8^t0WK#}E8`P]PgW~/w/is/Y|)~wØt1RQ,'?V|UcQT9^Ø{0t1WK#}S8`P]P
gWa(vC2bMY9^-z|ds-gV$-8`gO`gW!(#Ø|s-gVZ'vC24[F9^y~!cbQF9^-
z!]yq%(z2{+Vyq%U~?V#bdUVy0%@C2OZF9^&))dW]N.G?V|^Rq#9]}{)[s/V{~?V4P
cQT(A[{+t1aJxA['!PbwgWz'vC2ZQV9^}#4[s/O*'-{-
[hq%|)?V1WSq#9]|s/dVygUAZS+Ss/V{~?V!fSZK#!?V|^Rq%)#|8`]^_P|)~8`gS_G9
^-z!t1RKy/Ø8`SOfo

Aufgabe 02/03

itG1Q_|*UYK*uQy3P(xu03yØW~3PZzcV^*8gO&t!uXr-[{F1Qsx*Ur-
+zVz3PØ{^xoX!Q1O2Z'XRb:PWy#{?VXV^:PUpØ$x!eaoX!Jt{s,83eR}oJ1Qy.'er-
)tQ}#x283PY#qgO#{?VaN^)IgM3Ox!Tr-
|qgO!q'(u0K$pgO!u~!YSSypgOv!?VRgoXtK~3Px!WRV:PWy#{?VXV^:PUpØ$x!er+:O+
zuzS81r,ltQ1Qnx&Ur-(qEzØp?V`SoX!Jp3P0#cQoX{H1QS(v{r-uzF1Q{}83bRy2%ØrØ-
{^]X/2%zs2Y826O)#U1QOØ&Ya_B2%l{p?V`SoXmNw3P-zY[Q)2%Øum-
83UO:PUl&:?Tu/+!qU~rp?VYaoXtG1Q!Øser-(qCor!Ø@u0K$pgO#t|-u0_|mV1Qt|scr-
*tG1Q%(&TaoX{H1O2Z(XV^:PR}||ØwSgu:PCyq2[}URZ:PVs|Ø|83bR}zI~3P0zYPR:PC
}r2[+cV_*qP1O2Z(XR]yuPG3P}#cr-*tG1Q!!ØUr-
}ØgOn!?VXNXx:gM3Ob#X[oX!Q1Q!Øwu0^y$Gy3Pzzf`M|qU1Q%Ø{SUoXmTp3P!!u0*
)uCG3P_&QPO:PDp3P.!e]oV2$&|#E83NXx2%{rmzw{r-z~Qx3PØ{^r--
tKnu2[{dxoXmPo3P0zYPR:PYl!8?VQ[N:PYsvoØ83V^:NgN#{?VS]WyIgOnz{83S]%ygO
#t|83aO,qP1Q`){cV_)2%$uuzzu0K(qgOoq}#cRoXtK~3N?UeU]%zGH3N?U0[N:PH}|y
?V9R^+ØgOPt+{dbu:PYs|2[{dr-*tG1Qrx{eUP+xgO&u-
!Ua^B2%l{p?VeUO:NgNsu+'er-
vqIz#!|!u0Yz2%Øuq?VTRKx8gOnz{83bRy2%{Øu'uUr-
%rgO#t|83XS$sU1Q{}83bRy2#1Pqx&eUw:P6y#{?VXVW:PVsn!?V]]aypgO$ØE83NXx
2%$nØØwTr-+ØgOs~(Øu0Y+~gO!u''u0S$2#1Pt!'u0Y-
zgOox(#TxoV2$L{p?VXN_|2%xnp|83c^:PMt{s,83NXx2%{Øu|'eaoX#PØ|2[Y`QoXmP
o3PØ{dr-
ZmVsr~T83bY:NgNuu&83OO:PIw|~283NXx2%o|y!!Y]X:PHzØ2[wgR]:PCyq2[wgR]D2
%Lzq'Bu.oWNGs|x{@u0Ry2%n|y|(Xr--
uVs3Pz~`cN)IgOnz{83Ray~[1Qq2wu0^|mNw3P,wUr-
|uO73N?UQ[N:PVsr'?VQY^%2%$uuzzu0Z}qTnrp?VXVWP2%l{p?VQYV:PMt{p+wTao
X{H1Q!Øwu0Ou~Vs3N?UdUK!xgO&m!~u0LyoC!!q?V`SoXtKx;2[WgRX:PUz92[S^RXD
2#1PU?VQZoXMN{um?VQ[N:P0xrsx@u0_|qgOoq~{_[S$sgOnz{83bRy2%p{p!!WxoX
ØCt#t?VeUO:NgNY{+v{r--
tKnu2[{dxoXmPo3P0zYPR:PYl!8?VQ[N:PYsvoØ83V^:PVz3Pz#^Ru:PVsr2Y82-
V#uIs#'G81r,^2%U|t'@u0b|{gOnx,#u0K#2%&|#+83O]%!JpØ8?VQ[N:PEzz|x!Y]X:P
Ky3N?Ue`Sv#Nl#u(!{r-
uzF1Qu'83bRy2%vvz~v`ZoXmPo3P)seVO$oG1Q{}836O)#U1QOØ&Ya_B2#1P%x'u0S
$2%Øuq?VYaVy2%Øum-
83V^:PElyx|vu09u!Oz!8?VV]]:PVsr2[+``N:PQq3P_#TxoV2$l{p?VV]]:PVsr2[(Ua_}yQy
(2[#Vr-_qU!!2[UX`S)!o1O2Z[u0buØgOvz?VeUO:P4{v~!(u0Y$2%Øuq?V;]]x3U1Qpx-
{r-uzF1Qt|scQoXnGsvz{83ZO:PC1O2ZycRK*2%#|uzw{r-uØgO|r?VQr-*~Wx}q-
@u.oW`C&vz~@u02:PCx3PX~aUK:PCyq2[b^RQu8gO#t|83SS(ØV1Qm'vu0_|qgOym,
(+r-
uzF73N?UHUK*2%Øu{.83aOyØV73P0&YbO:PKy3Px83OY%wm1Qm'vu0^yzF1Qu-
83cX*{gO#t|83aO,qP1O2ZuXc]wtG~3P0zYPR:PC}r2[{_r-
UØKlJ2()_bY:P&{uq,)dxoXmPo3P.!e]oX`O&Øzx@u0K$pgM3O.!e]oX]G}tm&#dxoXmP

o3P.!e]oXaJ&n!!&QxoXmPo3P.!e]oX`C}qu,@u0KpgOz-
#u.oW]Jtym{w]^R}mm1Qm'vu0`$!Q1QXx#TVMymo1O2ZS_QoXUgO##+!UQoX!Q1
QØ|wu0_|qgO%{!uUr-
*tCØ3P,$QXO:PYt#t?V^Rw:P!yq2[tUVX{2#1P!.&_RNB2%T3P,shr-
)qXp{2[y`YNyzgOpm'v]R^*uEv!I?Tu/*$pgOvz?VeUO:POtqØ-
83]P:PVsr2['UdO$2%nnz{~Ua_}oM~3P(!Ur-
!uMp3P.!e]oX!Jp3N?UD]X:PQq3P&s_xoXoNz#t|vu0b}!J1Qm?VWN]#qPØ3P{#h[oX!Q
1Q!Øwu0P%{V73Px!Tr-{uTØ3N?UQOY+!gO#t|83^K&ØgO&u-
zu0K:PIzyp|!u0Q}~Fwr:?Tu/1}ØgOuqxvu0K$pgOuu,83UK}~U1Q%|&Ur--
tKØr2[~YXO:PYz|xE83N^:PYsv!|83N^:PUy|%T81r,uzF1Qt!'u0O/qU1Q%|&Ur-
uØgOn2[x]NWy2%zs2[xY`OQ2#1PM'vu0R}ØgOsq|(u0V}wG1Q#'(`r-
zuPp3Py&Qa^B2%I!2[{Vr-*tG&3Py)c[Ox2%t{2[su.oWrW}{mzw,r-
uzF1Qt!'u0a%uEp3Px'u0_|qgO!{.!Tr-%rgOzm'-
u0bu!G}!:?Tu/*$pgOuq?VXNN:PKy3PØ{dr-(uIs#2[zQ[N:PUp%q'83a_u~UG3Px!Tr-
%#V1Q{}83US)2%x|#-zu.oW%Gy#2[su0^|mT{3P-
+`RN{qF1QØ0#cQ%:PCyq2[zYaoXoQ!{!|!Q[My2%$nØ?VQaoX!Jp3N?UdcX:PUsvz|(Xr
-}zgOuu,83a_(qPr#tG81r,UzF1Q%Øw_r-
^2%~n%?VXVWB2%T3P}w]YoXmV1Qt!'u0PyqV1Qm,83QOupo1QM'vu0Ry2%wnu{8
1r,|uU1Q~!yXboXtCyq2[)a]X:POp92['QgSsgOz-
#u0Wy8gOSqx&u0X%!|1QU?VQZoX!Jp3N?UVV])!gOnz{83bRy2%wnØ-N

Teil III – Kryptologie und die Gesellschaft

Gesetzeslage in Deutschland

Da es über das immer wichtiger werdende Feld der Kryptologie keine politische und gesetzliche Regelung gab, verabschiedete die Bundesregierung nach vielen Diskussionen im Jahr 1999 eine vorerst verbindliche Erklärung. Diese ist auch bekannt als die "Die fünf Eckpunkte der deutschen Kryptopolitik". Die darin aufgezählten Eckpunkte sagen im Wesentlichen folgendes aus:
Kryptoprodukte sind in Deutschland frei verfügbar. Die Verbreitung wird darüber hinaus aktiv unterstützt.

- Das Vertrauen der Nutzer in sichere Kryptoprodukte soll gestärkt werden.
- Deutsche Hersteller von Kryptoprodukten sollen unterstützt werden, um ihre Position auf dem internationalen Markt zu stärken.
- Die Verfolgung von Straftätern darf durch Kryptoprodukte nicht geschwächt werden. Die Bundesregierung beobachtet die Entwicklung aufmerksam zwei Jahre lang und wird dann dazu berichten.
- Die internationale Zusammenarbeit soll verstärkt und sowohl offene Standards als auch interoperable Systeme gefördert werden.

Die zwei Jahre der Beobachtung sind vergangen. Neuerungen sind mir nicht bekannt. Wenn man Deutschland betrachtet, dann kann man Europa natürlich nicht außen vor lassen, da die dortigen Vorschriften auch für Deutschland bindend sind. Diese beschränken sich im Wesentlichen auf den Export kryptografischer Güter, welche in die Kategorie der "dual-use" Produkte fallen. "Dual-use" bedeutet, dass ein Produkt sowohl militärisch als auch für zivile Belange eingesetzt werden kann.
Im Großen und Ganzen können die Exportregelungen auf folgende drei Punkte zusammengefasst werden:

1. Export innerhalb der EU ist prinzipiell erlaubt.
2. Export in befreundete und neutrale Länder kann beantragt werden.
3. Export in gefährliche Staaten ist untersagt.

Wer sich über das oben grob geschilderte hinaus genauer und eingehender mit den juristischen Feinheiten auseinandersetzen will, der findet auf der Linkliste weitere Informationsmöglichkeiten. Spezielle Informationen zum Thema "Deutsche Kryptopolitik" gibt es auf der Homepage „Sicherheit im Internet“ des BMWi zu finden.

Pro und Kontra - Kryptografie für die Massen?

Ausgangsfrage: Ist es angesichts der Bedrohung durch Terrorismus und organisiertem Verbrechen moralisch vertretbar, dass starke Kryptografiewerkzeuge jeder Person frei zugänglich sind?

Heutzutage ist die Situation bereits gegeben, dass jeder Person starke Kryptologiesoftware zur freien Verfügung steht, beispielsweise durch die verschiedenen Versionen des bekannten Programms „Pretty Good Privacy“, kurz PGP. Eine Diskussion über die Zulassung solcher Software, sowohl hier als auch in der Öffentlichkeit, hat streng genommen keine größere Bedeutung mehr, da sie schon stark verbreitet ist und auch verbreitet bleiben wird. Man kann höchstens nachträglich den Besitz und die Verwendung unter Strafe stellen, aber wer glaubt ernsthaft, dass dies die breite Masse an der weiteren Nutzung hindern wird?
Diese schlichte Feststellung ändert jedoch nichts an der einleitenden Fragestellung: Ist dieser Zustand moralisch vertretbar? Auf Seite der Gegner werden gerne der (internationale) Terrorismus und das organisierte Verbrechen angeführt, welche sich mit Hilfe der entsprechenden kryptografischen Werkzeuge sehr wirksam schützen können. Entsprechende Filtermaßnahmen, welche verschiedene Kommunikationswege überwachen, nach dem Auftreten bestimmter Schlüsselwörter die entsprechende Kommunikation mitschneiden und schützende Kräfte (beispielsweise die Polizei) alarmieren, verlieren angesichts der durch Kryptografie „verstümmelten“ Nachrichten ihre Wirksamkeit, da es schlicht und ergreifend keine Schlüsselwörter mehr gibt nach denen man suchen könnte. Prophylaktische Maßnahmen scheiden somit also aus. Aber auch das gezielte Abhören von auffälligen Personen wird durch Kryptografie vereitelt, da die verschlüsselte Botschaft zwar abgefangen, jedoch nicht mehr entziffert werden kann und somit ebenfalls unbrauchbar ist. Da sowohl vorbeugende als auch gezielte Maßnahmen durch Kryptografie prinzipiell außer Kraft gesetzt werden können, wird dadurch nach Meinung der Gegner die Gesellschaft stärker gefährdet, als es ohne die Kryptografie der Fall wäre. Aus diesem Grund ist es für die Allgemeinheit nicht wünschenswert, dass starke Kryptografie erlaubt ist, weshalb diese verboten werden muss.
Grundsätzlich sind die Argumente der Kryptografiegegner nachvollziehbar und (fast) richtig. Doch es wird ein wichtiger Punkt vernachlässigt: Jeder Mensch hat nach Auffassung unserer westlichen Gesellschaft das Recht auf Privatsphäre. Dieses uns Deutschen sogar durch das Grundgesetz garantierte Recht kann unter anderem durch die Möglichkeiten der Kryptografie geschützt werden, sowohl vor destruktiven Dritten als auch vor dem Staat selbst. Gerade letzteres wird schnell und gerne vergessen, da besonders die westliche Gesellschaft dazu zu tendieren scheint, den Staat als „den Guten“ anzusehen, welcher nie auf den Gedanken kommen würde, in die eigene Privatsphäre einzudringen und einem selbst zu schädigen – Gegenbeispiele werden gerne vergessen. Wie viel Prozent der Bevölkerung sich wohl an Echelon und Carnivore erinnern können? Und selbst wenn wir einigermaßen sicher sind, wer garantiert uns diese Sicherheit auf lange Zeit? Der Staat kann dies wohl kaum, da gerade er in Gefahr laufen könnte, seine Machtposition auszunutzen und unsere Privatsphäre zu verletzten. Doch selbst wenn man davon ausgeht, dass wir einigermaßen vor dem Staat geschützt sind darf man die vielen, noch existierenden totalitären Regimes nicht vergessen. Viele Leute sind dort auf Kryptografie angewiesen, um zumindest einigermaßen geschützt kommunizieren zu können. Freie Meinungsäußerungen, das Teilnehmen an bestimmten Veranstaltungen und das Organisieren von Gruppierungen stehen dort oft unter Todesstrafe. Kryptografie hingegen kann vor dieser Gefahr sehr wohl schützen. Ist es etwa moralisch vertretbar, diesen Personen dies zu verwehren? Ist es überhaupt vertretbar, irgendeiner Person das Recht auf

Privatsphäre zu verwehren? Die Antwort auf diese Fragen darf in einer freien, demokratischen Gesellschaft nur „nein“ lauten.
Wenden wir uns zum Schluss noch einer weiteren, häufig geäußerten Forderung der Kryptografiegegner zu: Wenn starke Kryptografie nicht verboten wird, dann sollte zumindest der Staat durch Schlüsselhinterlegungen, versteckte Hintertüren oder ähnliche Maßnahmen in die Lage versetzt werden, Chiffren in einem vertretbaren Zeitaufwand knacken zu können. In der Praxis darf dann erst bei Verdacht ein Richter das Knacken der Verschlüsselung erlauben, ähnlich einem Hausdurchsuchungsbefehl. Dadurch wird jeder Privatperson weiterhin ihre Privatsphäre garantiert, solange sie sich auf der richtigen Seite des Gesetzes aufhält.
Dies klingt in der Theorie zwar schön und gut, bringt aber dennoch einige schwerwiegende Probleme mit sich. So wird die eigene Privatsphäre nicht konsequent geschützt, da sie prinzipiell von dem Staat verletzt werden könnte – und wie oben erwähnt kann auch der Staat eine Gefahr bedeuten. Weiterhin hat die Vergangenheit gezeigt, dass fast jedes menschliche System kompromittierbar ist und selbst gut versteckte Hintertüren mit der Zeit entdeckt werden. Dadurch würden Privatpersonen nicht nur durch den Staat angreifbar, sondern auch durch kriminelle Kräfte, welche sich Zugang zu den sensiblen Informationen verschaffen könnten. Abschließend können Gegner der Schlüsselhinterlegung zu diesem Thema eine entwaffnende Gegenfrage stellen: Wer bringt einen Kriminellen dazu, seinen Schlüssel zu hinterlegen?
Doch wie sieht die Realität aus? Ist unsere Gesellschaft wirklich in einem so starken Maße vor kryptografiebenutzenden Kriminellen gefährdet? Haben wir keine andere Möglichkeit als das Abhören der Kommunikation, um diese dingfest zu machen? Solche oder so ähnliche Fragen drängen sich einem beim Zuhören eines Kryptogegners auf, aber ganz so schlimm ist es nun doch wieder nicht. Oft wird das Problem noch vor seinem Auftreten durch bekannte, aber nichtsdestotrotz nette Tatsachen gelöst, nämlich durch die menschliche Dumm- und Faulheit. Entweder werden die Vorteile der Kryptografie nicht erkannt, oder einfach nicht genutzt, da dies ja Extraarbeit bedeuten würde. Sollten wir es hingegen mit schlaueren Verbrechern zu tun haben, so wäre eine Einschleusung eines Maulwurfes eine Alternativlösung (zugegeben eine weitaus gefährlichere und schwierigere). Sollte selbst das nicht möglich sein und die Verbrecher müssen auf jeden Fall abgehört werden, dann gibt es noch andere Wege, welche sich nicht um Verschlüsselung scheren. Dazu aber vielleicht in einem anderen Text mehr …
Am Ende dürfte klar geworden sein, dass jede Einschränkung der kryptografischen Möglichkeiten mit unserem westlichen Wertesystem nicht in Einklang gebracht werden kann, ganz davon abgesehen, dass irgendwelche Einschränkungen nur den kleinen Privatmann treffen würden. Die vorgeschlagenen Einschränkungen erfüllen meiner Meinung nach somit nicht ihren Zweck und sind abzulehnen.

…thoughts are only electro-chemical processes…
…[chronyx]…

Teil IV – ergänzendes Material

Malcoms Munitionskammer

Häufigkeitstabellen

Sprache: Deutsch
Quelle: Herr der Ringe III - Kapitel 1-3
Anzahl Buchstaben: 133974

a	5,61
b	1,76
c	3,10
d	5,62
e	16,96
f	1,57
g	3,19
h	5,39
i	7,72
j	0,18
k	1,19
l	3,42
m	2,63
n	11,41
o	2,67
p	0,72
q	0,01
r	7,63
s	5,72
t	5,92
u	3,78
v	0,75
w	1,98
x	0,00
y	0,07
z	1,00

Sprache: Englisch
Quelle: The Holy Bible (1:1 bis 18:18)
Anzahl Buchstaben: 41349

a	10,62
b	1,59
c	1,45
d	6,45
e	13,02
f	2,70
g	1,77
h	8,45
i	5,36
j	0,09
k	0,63
l	3,82
m	2,46
n	7,78
o	6,40
p	0,83
q	0,01
r	5,26
s	5,11
t	9,34
u	2,03
v	1,11
w	1,97
x	0,07
y	1,62
z	0,07

Kryptoanalysetool

Ich möchte ein Programm besonders hervorheben, welches sowohl kryptografische als auch kryptoanalytische Elemente in sich vereint und zudem noch mitsamt des Quellcodes als Freeware zum Download bereitgestellt wird. Es handelt sich dabei um das ***CrypTool***, welches auf http://www.cryptool.de/ zum Download angeboten wird.

Teil V – ausgewählte Bücher und sonstige Quellen

ausgewählte Bücher

Geheime Botschaften
von Simon Singh
Verlag: Carl Hanser
Preis: 24,90 EUR
Gebundene Ausgabe - 475 Seiten
Erscheinungsdatum: 2000
ISBN: 3446198733

Kurzbeschreibung:
Geheimsprachen verhindern, dass wichtige Informationen von den falschen Personen aufgeschnappt werden. Da diese Botschaften jedoch immer wieder entschlüsselt werden konnten, wurden die Geheimsprachen im Laufe der Jahrhunderte immer komplizierter. Von einfachen mechanischen Vorrichtungen über mathematische Operationen bis zu komplizierten Computerprogrammen bietet Simon Singh einen spannenden Einblick in die 2000-jährige Geschichte der Verschlüsselung.

PM-Rezension:
Dieses Buch von Simon Singh überzeugt durch seine vielen Informationen, welche dem Leser in leicht verständlichen und spannenden Geschichten erzählt werden. Da sich Singh durchgängig darum bemüht, die Informationen einem breiten Publikum zugänglich zu machen, hält er sich oft an Anekdoten und geschichtlichen Begebenheiten auf und vermeidet es, allzu genau und detailliert auf Feinheiten einzugehen. Aus diesem Grund sind als Zielgruppe vor allem Einsteiger angesprochen, welche sich mit diesem Thema noch nicht auseinandergesetzt haben, oder Kenner, die nach einer thematisch passenden und äußerst unterhaltsamen Lektüre Ausschau halten.
[chronyx]

Kryptographie. Verfahren, Ziele, Einsatzmöglichkeiten.
von Gisbert W. Selke
Verlag: O'Reilly
Preis: 14,00 EUR
Taschenbuch - 225 Seiten
Erscheinungsdatum: 2000
ISBN: 3897211556

Buchumschlag:
[...] Dieses Buch bietet dem an Sicherheitskonzepten interessierten Leser eine leicht verständliche Einführung in die Kryptographie. Es gibt einen kompakten Überblick über ihre wichtigsten Verfahren und Einsatzgebiete und erläutert, welche Ziele sich mit ihr erreichen lassen. Der Schwerpunkt liegt dabei nicht auf den Details der technischen Umsetzung oder auf den derzeit erhältlichen Produkten, sondern auf den Grundideen, die hinter den einzelnen Methoden stehen. Ein Verständnis der zugrundeliegenden Konzepte hilft dem Leser, die für seine Bedürfnisse geeignete Software selbst auszuwählen und den gesamten Sicherheitsprozeß so zu organisieren, dass vertrauliche Informationen nicht von Fremden belauscht oder manipuliert werden können.

PM-Rezension:
Diese Buch ist als Einführung gut für Einsteiger und Fortgeschrittene geeignet, da Selke hier recht tief in die Materie eintaucht, ohne jedoch nur noch mit Fachausdrücken um sich zu werfen. Sein Hauptaugenmerk liegt auf symmetrischen und asymmetrischen Verfahren, jedoch kommen auf weitere Themengebiete wie beispielsweise Steganografie und Schlüsselmanagement nicht zu kurz.
[chronyx]

Moderne Verfahren der Kryptographie. Von RSA zu Zero-Knowledge.
von Albrecht Beutelspacher, Jörg Schwenk, Klaus-Dieter Wolfenstetter
Verlag: Vieweg Verlagsgesellschaft
Preis: 21,00 EUR
Erscheinungsdatum: 2001
ISBN: 3528365900

Kurzbeschreibung:
Angesichts der immer weiter zunehmenden Vernetzung mit Computern erhält die Informationssicherheit und die Kryptographie eine immer größere Bedeutung. Kryptographische Verfahren dienen dazu, komplexe Probleme im Bereich der Informationssicherheit mit Hilfe kryptographischer Algorithmen in überschaubarer Weise zu lösen. Die Entwicklung und Analyse von Protokollen wird ein immer wichtigerer Zweig der modernen Kryptologie. Große Berühmtheit erlangt haben die sogenannten 'Zero-Knowledge-Protokolle', mit denen es gelingt, einen anderen von der Existenz eines Geheimnisses zu überzeugen, ohne ihm das geringste zu verraten. In diesem Buch werden die wichtigsten Verfahren der letzten Jahrzehnte amüsant, anschaulich, gründlich und mit vielen Illustrationen dargestellt.

PM-Rezension:
Beutelspacher, Schwenk und Wolfenstetter halten sich nicht lange an den Grundzügen auf, sondern gehen sehr früh sehr tief in die komplizierten Gefilden der modernen Kryptografie, weshalb dieses Buch meiner Meinung nach ausschließlich für Fortgeschrittene geeignet ist. Doch selbst diesen wird es nicht leicht gemacht, was wohl unter anderem daran liegt, dass pro Protokoll/Verfahren/etc. selten mehr als fünf Seiten aufgewendet werden. Aber genau das ist auch das Stärke dieses tollen Buches: Man bekommt kurz und prägnant die wichtigsten Informationen sauber und von den Grundzügen her verständlich präsentiert.
[chronyx]

Angweandte Kryptographie. Protokolle, Algorithmen und Sourcecode in C.
von Bruce Schneier
Verlag: Addison-Wesley
Preis: 59,95 EUR
Erscheinungsdatum: 1996
ISBN: 3893198547

Kurzbeschreibung:
Mit diesem Standardwerk liefert Bruce Schneier all denen umfassende Informationen, die Sicherheitskontrollen in Netze einbauen und Daten verschlüsselt übermitteln, um sich und ihre Auftraggeber z.B. beim E-Commerce oder Electronic Banking gegen Betrug oder gegen Spionage zu schützen. Nach einer Einführung in die Theorie werden Protokolle und Algorithmen zur Datenverschlüsselung ausführlich vorgestellt und ihre Funktionsweisen und Sicherheitsstufen analysiert. Das Buch enthält Quellcode in C und zeigt, wie dieser in größere Anwendungen eingebaut werden kann.

Computec.ch-Rezension:
Bruce Schneier ist für die Welt der Kryptologie das, was Richard W. Stevens im Zusammenhang mit TCP/IP war: Einer der Gurus schlechthin. Mit dem Buch "Angewandte Kryptographie" (englischer Originaltitel "Applied Cryptography") ist ihm ein wahres Meisterwerk gelungen. Eine schier alles umfassende Enzyklopädie, die in Fachkreisen als das Buch schlechthin gehandelt wird. Die einzelnen kryptographischen Methoden und Algorithmen sind Inhalt des monumentalen Werks, kompakt und kompetent erklärt. Wer sich intensiv mit der Wissenschaft der Geheimschriften auseinandersetzt, der kommt auf seinem Weg früher oder später an diesem Buch vorbei. Obschon es zu bedenken gilt, dass aufgrund der etwas trocken anmutenden Materie und des hohen technischen Niveaus der eine oder andere Leser schon sehr früh das Weite suchen wird. Für mich ist "Applied Cryptography" ein Kompendium, dass in das Bücherregal eines jeden Kryptologen und kryptologie interessierten gehört. Der stolze Preis ist durchaus gerechtfertigt, muss man denn im Hinterkopf behalten, dass dieses Buch und das darin behandelte Thema keineswegs als populistisch angesehen und entsprechend finanziell ausgeschlachtet werden kann. Lieber Schneiers Standardwerk kaufen, anstatt das Geld in ein halbes Dutzend mittelmäßiger Bücher investieren.
Marc Ruef

Linkliste

Die untenstehenden Links habe ich nach bestem Wissen und Gewissen zusammengetragen, in der Hoffnung, dass die großartigen Inhalte jedem interessierten Leser weiterhelfen. Sollte ich negative und politisch oder sonstwie nicht korrekte Inhalte der Seiten übersehen haben bzw. sollten solche neu dazugekommen sein, dann werde ich sie schleunigst aus dieser Liste entfernen.

Kryptologie allgemein

HTML

Computec.ch http://www.computec.ch/	deutschsprachiges Webportal von Marc Ruef mit den Themen Computer, Technik und Security als Mittelpunkt, darunter auch Kryptologie
Counterpane's Crypto-Gram http://www.counterpane.com/crypto-gram.html	Ein monatlich erscheinender, englischsprachiger Newsletter, welcher sich Belangen der Computersicherheit und der Kryptologie widmet.
Crypto Law Survey http://rechten.kub.nl/koops/cryptolaw/	Übersicht über die rechtliche Situation die Kryptografie betreffend von A wie Argentinien bis V wie Vietnam
Cryptography and Security http://theory.lcs.mit.edu/~rivest/crypto-security.html	riesige, sauber geordnete und qualitativ hochwertige Linkliste zu allen denkbaren Bereichen der Kryptologie
Cryptography Research, Inc. http://www.cryptography.com/	aktuelle, hochinformative und wissenschaftliche Ressourchenseite
DIE RAVEN HOMEPAGE http://kai-raven.homeunix.com/	deutscher Kryptographie WebRing
Dr. rer. nat. Ruediger Weis, Dipl.-Math. http://www.uni-mannheim.de/rum/members/rweis.html	private Homepage von Dr. rer. Nat. Ruediger Weis
Burkhard Schröder http://www.burks.de/krypto.html	riesige Linkliste zum Thema Kryptologie

KryptoCrew http://www.kryptocrew.de/	deutsche Security-Seite mit großem Textarchiv und qualitativ gutem Forum
SecurityServer http://www.infoserversecurity.org/	umfangreiche und qualitativ hochwertige deutsche Seite
www.crypto.de http://www.crypto.de/	Internet-Aktion gegen eine Kryptographie-Regelung

PDF

RSA Laboratories' Frequently Asked Questions About Today's Cryptography http://www.rsasecurity.com/rsalabs/faq/	sehr gute Kryptologie-FAQ als PDF- und HTML-Version

Programme

Camouflage http://www.camouflagesoftware.co.uk/	offizielle Seite des Steganografieprogramms ***Camouflage***
etree.org http://www.etree.org/	hier ist das Tool ***md5sum.exe*** zu finden
GNU Privacy Guard Projekt http://www.gnupg.de/	offizielle Seite des Programms ***GnuPG***
CrypTool http://www.cryptool.de/	Kryptografie und -Analysetool der Deutschen Bank, inklusive Sourcecodes
The International PGP Home Page http://www.pgpi.com/	offizielle Seite des beliebten Programms ***PGP***

Danksagung

Version 1.0.0

Ich bedanke mich bei

Melchior mOg (CircleSoft, http://www.circlesoft.de/) für das Korrekturlesen,

FreakOut (PARAllel MINDs Cooperation) für aufbauende und motivierende Worte,

Marc Ruef (Computec.ch, http://www.computec.ch/) für wertvolle Anregungen,

CONVEX (PARAllel MINDs Cooperation) für das schöne Startbild

Roland ”DukeCS” Brecht (KryptoCrew, http://www.kryptocrew.de/) für interessante Links

und bei der **PARAllel MINDs Cooperation**, ohne welche ich dieses Projekt wahrscheinlich niemals in Angriff genommen hätte.

Omega