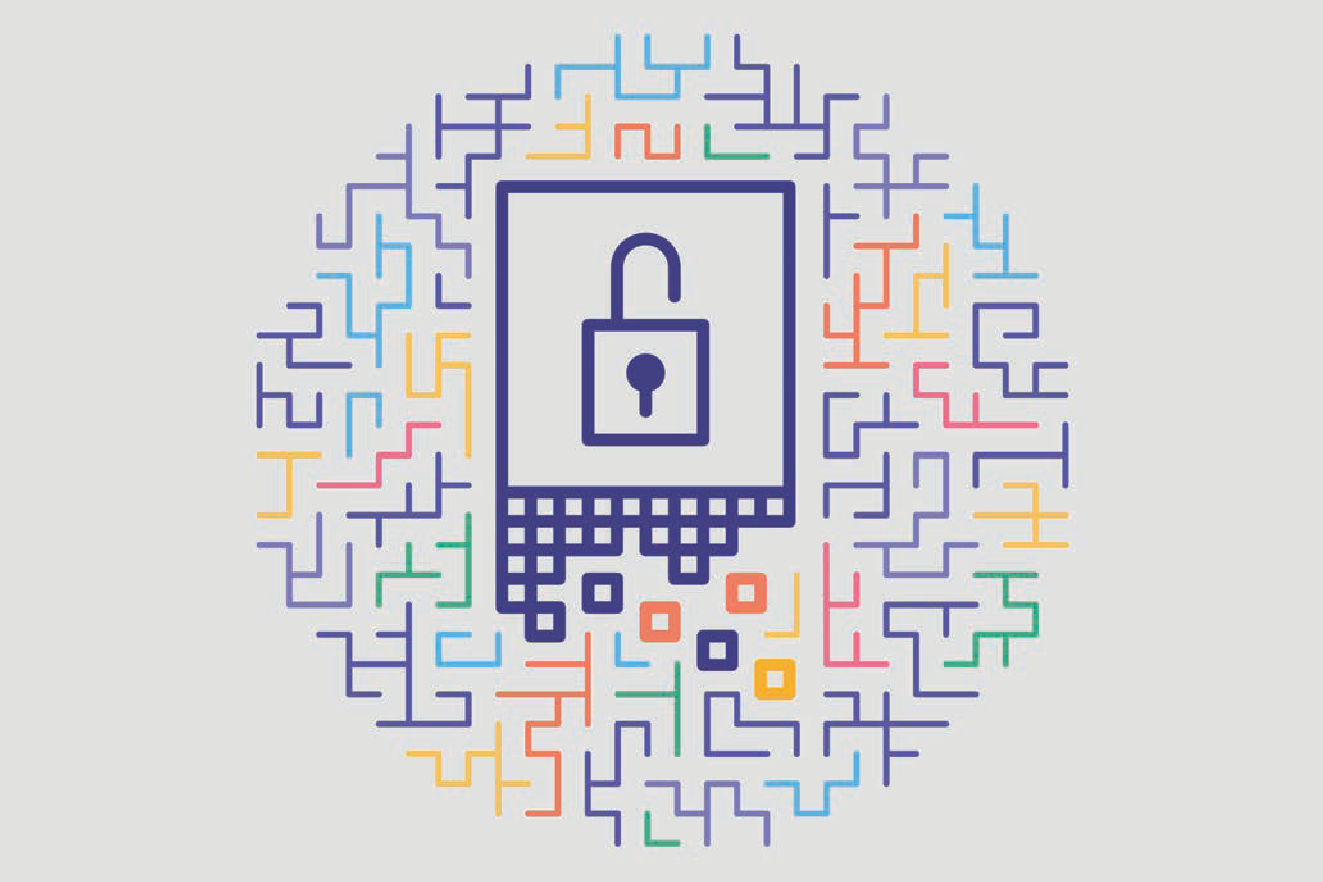
STUDIENSKRIPT



## Kryptologie

DLMCSEAITSC02



Übergeordnete Lernziele

##### Einleitung 9



Die Kryptologie als Lehre und Wissenschaft der vertraulichen Nachrichtenübermittlung umfasst die beiden Teilgebiete Kryptografie und Kryptoanalyse. Kryptografie ist die Wissenschaft der Verschlüsselung von Informationen, so dass diese nur für den vorgesehenen Empfänger entzifferbar sind. Kryptoanalyse dagegen ist die Wissenschaft von der Entschlüsselung von verschlüsseltem Text ohne Kenntnis des Schlüssels. In diesem Studienskript befassen wir uns darüber hinausgehend auch mit der geheimen Weitergabe von Informationen mit Hilfe der Steganografie, also dem Verbergen einer Nachricht durch deren Einbettung in eine andere Nachricht, so dass niemand außer dem beabsichtigten Empfänger überhaupt von der Existenz der Nachricht weiß, im Gegensatz zur Kryptografie, bei der die Nachricht unkenntlich gemacht wird.

Wir werden lernen, dass die Kryptografie Informationen schützt, indem sie Daten so mischt, dass nur zusätzliche geheime Informationen, der Schlüssel, die ursprünglichen Daten wiederherstellen können. Wenn der zur Ver- und Entschlüsselung einer Nachricht verwendete Schlüssel derselbe ist, spricht man von symmetrischer oder Ein-Schlüssel-Kryptografie. Indem wir die Rollen der Schlüssel vertauschen und den privaten Schlüssel zum Verschlüsseln und den öffentlichen Schlüssel zum Entschlüsseln verwenden, erhalten wir die Möglichkeit, eine Nachricht digital zu signieren. Bei der asymmetrischen Kryptografie liegt die Sicherheit ihrer Algorithmen für öffentliche Schlüssel in der inefﬁzienten Berechnung von mathematischen Funktionen auf ganzen Zahlen begründet. Bei symmetrischen kryptografischen Algorithmen basiert die Sicherheit darauf, dass kleine Unterschiede in der Eingabe und im Schlüssel zu großen Unterschieden in der Ausgabe führen.

Wir werden zwischen symmetrischer und asymmetrischer Kryptografie unterscheiden und deren größte Herausforderungen diskutieren. Eine Lösung bieten Public-Key-Infrastrukturen, bei denen private Schlüssel entweder von dritten, hierarchisch aufgebauten Behörden oder durch das Netz des Vertrauens (Web of Trust) zertifiziert werden, wobei das Vertrauen in einem Netz wiederum transitiv übertragen wird. Eine der bekanntesten Einsatzmöglichkeiten für Chiffrierungen sind Hashfunktionen. Eine Hashfunktion ist ein Algorithmus, der aus einer Eingabe mit variabler Größe eine Ausgabe mit fester Größe erzeugt. Die von einer kryptografischen Hashfunktion aus einer beliebig langen Zeichenkette erzeugte Ausgabekette mit fester Länge ist eine Form der eindeutigen Signatur.

Für bekannte Kryptografie-Algorithmen wie AES, DES, RSA und ECC besprechen wir deren Widerstandsfähigkeit gegen gängige Methoden der Kryptoanalyse wie Brute-Force-Angriffe und, im Falle von kryptographischen Hashes, Brute-Force-Angriffe mit Rainbow-Tables. Wir werden uns zudem die empfohlenen Schlüsselgrößen ansehen und den Aufwand berechnen, der erforderlich ist, um alle möglichen Schlüssel zu überprüfen.

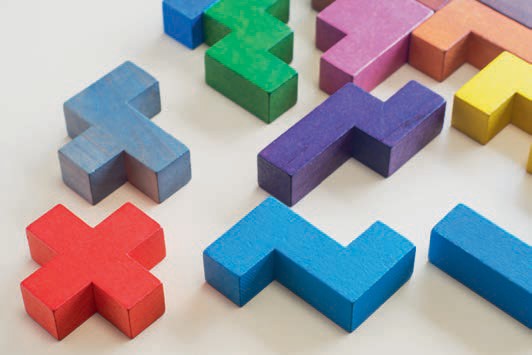
[www.iubh.de](http://www.iubh.de/)

##### 10 Einführung

Schließlich werden wir uns mit Netzwerkverbindungen wie VPN über den OSI-Stack und bekannten Netzwerkprotokollen wie TLS und seinem Vorgänger SSL zur Authentifizierung, Geheimhaltung und Authentifizierung von über TCP gesendeten Daten befassen. Auch rechtliche Aspekte wie die Datenschutz-Grundverordnung und andere Rechtsgrundlagen, die sich auf die Verwendung von Verschlüsselung auswirken, werden zur Sprache kommen.

Als Anwendungsbeispiele für die Kryptografie in verschiedenen Bereichen werden wir das Online-Banking-Protokoll FinTS, die aktuellen Fortschritte bei der Blockchain als vertrauenswürdiger Dritter in Form einer Datenbank mit Einträgen und die Möglichkeiten des Quantencomputings diskutieren, die es zu bedenken gilt. Die anonyme Datenübertragung im Internet durch das Onion-Routing im Tor-Netzwerk rundet unseren Praxisexkurs ab.

[www.iubh.de](http://www.iubh.de/)



# Lektion 1

## Grundkonzepte der Kryptologie

#### LERNZIELE

Nach der Bearbeitung dieser Lektion werden Sie wissen, …

… was Kryptologie, Kryptografie und Kryptoanalyse auszeichnet.

… welche Einsatzmöglichkeiten es für die Kryptologie hinsichtlich Vertraulichkeit, Integrität und Verfügbarkeit gibt.

… welche kryptografischen Algorithmen es bisher gegeben hat und welche Bedeutung sie bis heute haben.

… was Kerckhoffs‘ Prinzip besagt.

… wozu Hashfunktionen eingesetzt werden.

DL-E-DLMCSEAITSC02-U01

1. Grundkonzepte der Kryptologie

### Einführung

Verschlüsselung Auch als Chiffrierung bezeichnet, werden Daten so verschlüsselt, dass sie nur durch zusätzliche geheime Informationen wieder lesbar gemacht werden können. Die Umkehrung dieses Vorgangs wird als Dechiffrierung oder Entschlüsselung bezeichnet.

Schlüssel Dies ist die zusätzliche geheime Information, die für die Entschlüsselung praktisch unerlässlich ist.

Symmetrische Kryptographie Kryptographie ist symmetrisch, wenn derselbe Schlüssel zur Ver- und Entschlüsselung verwendet wird.

Asymmetrische Kryptographie Kryptographie ist asymmetrisch, wenn zum Verschlüsseln und Entschlüsseln unterschiedliche Schlüssel verwendet werden. Der Schlüssel zum Verschlüsseln ist öffentlich und der Schlüssel zum Entschlüsseln ist privat.

Kryptografie dient dem Schutz von Informationen durch **Verschlüsselung**, d. h. die Umwandlung von verständlichen Daten in unentzifferbare Daten, die nur durch eine zusätzliche geheime Information, den Schlüssel, wieder rückgängig gemacht werden kann (Entschlüsselung oder Dechiffrierung). Verschlüsselte (chiffrierte) Daten können nur durch Kenntnis des Schlüssels entschlüsselt (dechiffriert) werden. Da die ursprünglichen Daten grundsätzlich noch wiederhergestellt werden können, kann dieses Vorgehen als Konservierung bezeichnet werden. Da historisch gesehen nur schriftliche Nachrichten verschlüsselt wurden, werden die Ausgangsdaten, auch wenn es sich um eine Folge von 1en und 0en (wie in der symmetrischen Kryptografie) oder eine Zahl (wie in der asymmetrischen Kryptografie) handelt, als Klartext und die verschlüsselten Daten als Geheimtext bezeichnet.

###### Single-Key- und Public-Key-Kryptografie

Historisch gesehen war der Schlüssel zur Umkehrung der Umwandlung von verständlichen Daten in unentzifferbare Daten sowohl zur Entschlüsselung als auch zur Verschlüsselung erforderlich. In früheren Zeiten war der Schlüssel zum Ver- und Entschlüsseln immer derselbe. Symmetrische Kryptografie wurde von den Ägyptern bereits 2000 Jahre v. Chr. verwendet, im Zweiten Weltkrieg bei der Enigma-Maschine eingesetzt und findet auch heute noch Anwendung bei der Verschlüsselung eines drahtlosen Netzwerks (z. B. durch den AES-Algorithmus).

Die asymmetrische Kryptografie wurde dagegen erst in den 1970er Jahren erfunden. Hier unterscheiden sich der Schlüssel zum Verschlüsseln (der öffentliche Schlüssel) und der Schlüssel zum Entschlüsseln (der geheime oder private Schlüssel) voneinander. Tatsächlich ist nur der Schlüssel zum Entschlüsseln privat, während der Schlüssel zum Verschlüsseln öffentlich (allen bekannt) ist. Die symmetrische Verschlüsselung vermeidet das Risiko der Kompromittierung des Chiffrierschlüssels beim Austausch des Schlüssels mit dem Chiffrierer und beim Besitz des Chiffrierschlüssels (durch den Chiffrierer zusätzlich zum Dechiffrierer).

Außerdem ist es für eine digitale Signatur sinnvoll, wenn die Schlüssel ihre Rollen vertauschen, d.h. der private Schlüssel verschlüsselt und der öffentliche Schlüssel entschlüsselt. Während die verschlüsselte Nachricht nicht mehr geheim ist, kann jeder Besitzer des öffentlichen Schlüssels überprüfen, ob der private Schlüssel die ursprüngliche Nachricht verschlüsselt hat.

Heute sind solche asymmetrischen Kryptografie-Algorithmen im Internet allgegenwärtig. Beispiele hierfür sind RSA, das auf der schwierigen Faktorisierung von Primzahlen beruht, oder ECC, das auf der Schwierigkeit beruht, Punkte in endlichen Kurven zu berechnen. Diese schützen (finanzielle) Transaktionen auf sicheren Websites (die durch ein Vorhängeschloss in der Adressleiste des Browsers gekennzeichnet sind).

###### Datenformat

Bis zum Digitalzeitalter konzentrierte sich die Kryptografie auf die Umwandlung von verständlichem Text in unentzifferbaren Text. Heutzutage befasst sich die Kryptografie schwerpunktmäßig mit der Umwandlung von verarbeitbaren (digitalen) Daten in unentzifferbare (digitale) Daten. Bei den Daten kann es sich z. B. um eine digitale Datei (Text, Bild, Ton, Video usw.) handeln. Sie werden als Bitfolge ( angegeben durch eine

Grundkonzepte der Kryptologie

Folge von 0en und 1en) oder Bytefolge (angegeben durch eine Folge von hexadezimalen Paaren 00, 01,

..., FE, FF) oder als Zahl (angegeben wie gewohnt durch ihre dezimale Ausprägung 0, 1, 2, 3 ...) bezeichnet. Erinnern wir uns daran, dass jede Bitfolge 1011*…* eine Zahl *n* in Binärschreibweise n = 1 + 0 · 2 + 1 · 22 + 1 · 23 + ... ist (und umgekehrt).

In der symmetrischen Kryptografie, deren Algorithmen die Bits z. B. durch Permutation und Substitution umwandeln, wird die Betrachtungsweise als eine Bitfolge (genauer gesagt von hexadezimalen Ziffern, deren sechzehn Symbole 0-9 und A-F einer Gruppe von vier Bits entsprechen) bevorzugt. Die Betrachtungsweise als Zahl wird in der asymmetrischen Kryptografie bevorzugt, deren Algorithmen sie mit mathematischen Funktionen wie der Potenzierung und Exponentiation bearbeiten. Der Schlüssel, die zusätzliche geheime Information, kann dabei verschiedene Formen annehmen. Zwar ist die gewählte Form in erster Linie eine Frage der Zweckmäßigkeit, doch die gängigsten Ausprägungen sind eine Zahl oder eine Buchstabenfolge, z. B. ein Passwort, oder eine geheime Phrase (mit Leerzeichen).

Bei dem antiken Skytale-Algorithmus, bei dem eine Pergamentrolle um einen Stab gewickelt wird, besteht der Schlüssel aus dem Umfang (in Buchstaben) des Stabes, also einer kleinen Zahl. Heutzutage sind PIN-Codes (Persönliche Identifikationsnummer) oder Passwörter im täglichen Leben allgegenwärtig. Damit Sie sich den Code besser einprägen können, sollten Sie vollständige geheime Sätze oder Passphrasen verwenden.

Asymmetrischer Schlüssel

-----BEGIN PGP PUBLIC KEY BLOCK -----

iQI2BCABCAAgFiEErZvSGgyHY7r19sbt31JsxV5IVU0FAl7aa3sCHQAACgkQ31Js xV5IVU0E7xAAiy24cTZz2OK+OwcoB2GxDjFC/qYV8sup3aFUqDRCpXRDRee18t+e

/N/pJUzY0CurgsNMl3bRDDMuZNPw/2/jaSYUrSpDsTHwqMI4sDAvqXOB1j6bMEUm S1/ME9EJ+LMdRhtpnmqfsHkptGfnMrrRfHAdzo43Fw5j/xmFx7eb9a8DdQMkBM9m zBtfoxB9d2/ZVqk0yoFwovuUnwVnEaOx0xLtdxdHlxRUWUF6nXJ278IHr6h4wolO JEBtZv9VN8f3rbmnaWDLEplfnOIVwAB2Becch7t1sBYbxy3rzvSU9cEHIpYgWuyQ wX2qy20+vfHEGMkwlv83l2ncacXRDeHXKAtigxhqirrAyUSThWxsAOGNJMmPq6RY aCg8IvgW09rvwU6CqZ2QuBjxXN8K0EtSH+RHH4FVTTP4wtbg3UsV/aXgNCWHF9EO V+bbhaig5bhC6lS/inosVg8ooOhW6+czl2in4G4mHSRrPZ8w74jfcxBm0ubbAdSy DUrMcNGHJ8ITjGKp23WKYpcnss5vMFYGKBa2eKlOmemPTzb3SHxlzAaHC6nef+GQ

W/wHgP5KgtVG4Y+HXtu+0tJhRCqlOPGzxVaBX0b61CYzN5jurAhZi2CCZ1E+xte+

7wEZ7j7zawT0Ev9oZpT56m00OTUcjfRPDy9tQlehuNLI6XXXLLW0F+M=

=+tEF

-----END PGP PUBLIC KEY BLOCK-----

Dieser asymmetrische Schlüssel wird aufgrund seiner Länge in einer Datei von mehreren Kilobyte Größe gespeichert, die als ASCII-Armor bezeichnet wird.

### Begriffe

Das Präfix Krypto- stammt vom griechischen *kryptós*: „versteckt, verborgen, geheim“. Dementsprechend ist Kryptografie (vom griechischen *gráphein*: „schreiben“) die Kunst des verborgenen oder geheimen Schreibens, d. h. Informationen so zu verändern, dass sie für alle anderen als den vorgesehenen Empfänger unentzifferbar sind. Kryptoanalyse (vom griechischen *analýein*:„Auflösung“) oder Codeknacken ist die Kunst, die verborgene Schrift ohne den zugehörigen Schlüssel zu entschlüsseln. Zum Knacken von Chiffrierungen wird die verschlüsselte Information ohne Kenntnis des Schlüssels wiederhergestellt oder gefälscht.

Kryptologie (von griechisch *lógos*: „Wort“, „Vernunft“ , „Lehre“ oder „Bedeutung“) ist die Wissenschaft von der verborgenen, vertrauenswürdigen Nachrichtenübermittlung, die Kryptografie und Kryptoanalyse umfasst. Der allgemeinste Begriff ist Kryptologie und wird oft als Synonym für Kryptografie und auch Kryptoanalyse verwendet. Häufig synonym verwendete Adjektive sind in diesem Zusammenhang geheim, privat und vertraulich. Sie alle beschreiben Informationen, die anderen Personen nicht bekannt sind oder die ihnen nicht bekannt sein sollen. Etwas ist

* *geheim*, wenn es nur einer bestimmten Person oder Gruppe bekannt ist,
* *vertraulich*, wenn es geheim gehalten werden, d. h. nicht an andere Personen weitergegeben werden soll,
* *privat*, wenn es geheim bleiben soll, insbesondere wenn es um eine Einzelperson gegenüber dem Staat geht.

Geheimhaltung ist zwar auch heute noch wichtig, aber seit dem Aufkommen der Kryptografie mit öffentlichen Schlüsseln in den 1980er Jahren ist sie nicht mehr der einzige Zweck der Kryptologie. Damit elektronische Geräte das ersetzen können, was früher von Hand gemacht wurde, wurden digitale Signaturen und Authentifizierung eingeführt.

Steganografie (von griechisch *steganos*: „verborgen“) ist die Kunst, eine Nachricht zu verbergen, indem man sie in eine andere Nachricht einbettet (in ein Bild, eine Audio-Datei oder ein binäres Datenformat), so dass niemand außer dem vorgesehenen Empfänger von der Existenz der Nachricht überhaupt weiß. Im Gegensatz dazu wird bei der Kryptografie nicht die Existenz der Nachricht selbst verschleiert, sondern nur ihr Inhalt. Allerdings deutet eine Verschlüsselung darauf hin, dass bestimmte – vermutlich

Grundkonzepte der Kryptologie

wichtige – Informationen absichtlich verborgen werden. Da eine typische Bild- oder Tondatei Hunderte von Kilobyte groß ist, können einige wenige Bytes verändert werden, um eine geheime Botschaft zu übermitteln, ohne dass dies für das unbedarfte Auge oder Ohr erkennbar wäre. (OPTED, o. J.)

Kryptografie

Kryptografie wird bereits seit langer Zeit eingesetzt, um beispielsweise militärische Nachrichten vor dem Feind zu verbergen. Inzwischen haben (elektronische binäre) Daten schriftliche Texte ersetzt, und was früher von Boten ausgetauschte oder geheimgehaltene geschriebene Nachrichten waren, sind heute gesicherte Daten, die zwischen Computern übertragen oder auf einem Computer gespeichert werden. Asymmetrische Kryptografie ermöglicht durch digitale Signaturen, dass alle Teilnehmer an einer Kommunikation zweifelsfrei veriﬁziert werden können, was als Non-Repudiation (Nachweisbarkeit) bekannt ist. Dies eröffnet zahlreiche Möglichkeiten, insbesondere für den elektronischen Handel.

Alle kryptografischen Verfahren, die als Kryptosysteme bezeichnet werden, werden in symmetrische und asymmetrische Kryptosysteme eingeteilt. Symmetrische Kryptosysteme werden danach eingeteilt, ob sie mit Bitblöcken von fester Länge, z. B. 128 Bit, (Blockchiffre wie AES oder RSA) oder mit einzelnen Bits (Stromchiffre wie RC4) arbeiten. Stromchiffren sind zwar in der Regel einfacher, schneller und für Echtzeitübertragungen hervorragend geeignet, aber sie sind im Allgemeinen weniger sicher und werden daher seltener verwendet (ein WLAN-Netzwerk wird z. B. üblicherweise durch eine Blockchiffre wie AES gesichert).

Code

Eine Kodierung ist eine Regel zum Ersetzen eines Informationsbits, z. B. eines Buchstabens, durch ein anderes, in der Regel, um es für die Verarbeitung durch einen Computer vorzubereiten. Beispiele:

* Morsecode
* American Standard Code for Information Interchange (ASCII-Code) aus dem Jahr 1963, der auf Computern 128 Zeichen (und Operationen wie Rücktaste und Zeilenvorschub) durch 7-Bit-Zahlen darstellt. In ASCII entspricht ein kleines a der Bitfolge 1100001, ein kleines b der Bitfolge 1100010 usw., ein großes A entspricht der Bitfolge 1000001 und ein großes B wiederum der Bitfolge 1000010.
* UTF-8 (8-bit Unicode Transformation Format). Hierbei handelt es sich um eine von Ken Thompson und Rob Pike entwickelte Zeichenkodierung mit variabler Länge, die jedes universelle Zeichen des Unicode-Standards durch eine Folge von ein bis vier Bytes darstellt und zudem mit ASCII rückwärtskompatibel ist. Dieser

Standard bietet die Möglichkeit, bis zu 232 ≈ 4 Milliarden Zeichen abzubilden und enthält u.a. die Alphabete

zahlreicher Sprachen, wie Deutsch und Chinesisch, sowie bedeutungsvolle Symbole wie Emoticons.

Chiffren

Eine Chiffre ersetzt genau wie eine Kodierung eine Information (die von einem einzelnen Bit bis zu einer ganzen Zeichenfolge reichen kann) durch eine andere. Die Ersetzung erfolgt jedoch nach einer durch einen Schlüssel festgelegten Regel, so dass niemand ohne dessen Wissen die Ersetzung umkehren kann.

Heute werden Informationen für die Verarbeitung durch einen Computer kodiert und für einen sinnvollen Grad an Informationssicherheit chiffriert. Informationen, die im Internet zu finden sind, werden zunächst in das ASCII-Format umgewandelt, dann in Chiffren verschlüsselt, z. B. mit dem AES-Algorithmus (Advanced Encryption

Chiffre

Diese Regel wird verwendet, um Informationen so zu ersetzen, dass deren Umkehrung nur mit Kenntnis des Schlüssels möglich ist.

Standard), und schließlich für die Übertragung mit Fehlerkorrekturcodes erneut kodiert. Nach der Übertragung muss der Empfänger dasselbe in umgekehrter Reihenfolge tun (Bellare & Rogaway, 2000).

### IT-Sicherheit: Bedrohungen und verbreitete Angriffsmethoden

CIA

In der Informationssicherheit steht CIA für die Vertraulichkeit, Integrität und Verfügbarkeit (*Confidentiality*, *Integrity*, *Availability*) von Informationen.

Ein prägnantes Akronym, das die grundlegenden Ziele der Informationssicherheit zusammenfasst, lautet CIA. Noch umfassender sind die „Fünf Säulen der Informationssicherheit“, die Authentifizierung und Non-Repudiation ergänzen (Chen et al., 2019).

Mit Hilfe der Kryptografie können wir alle diese Ziele hervorragend realisieren. Eine gute Verschlüsselung, wie sie durch gründlich getestete Standardalgorithmen wie AES oder RSA erreicht wird, ist computertechnisch praktisch nicht zu knacken. Stattdessen werden die Schlüssel oder der Klartext in vielen Fällen vor der Verschlüsselung oder nach der Entschlüsselung gestohlen. Die Kryptografie bietet also ein hohes Maß an technischer Sicherheit, doch menschliches Versagen, z. B. aus Bequemlichkeit oder unangebrachtem Vertrauen, ist der größte Schwachpunkt der Informationssicherheit.

###### Vertraulichkeit

Vertrauliche Informationen sollen geheim gehalten und nicht an andere Personen weitergegeben werden. Dies können beispielsweise Informationen sein, die nur Ihrem Arzt oder Ihrer Bank bekannt sind. In der Rechtswissenschaft bezeichnet der Begriff Vertraulichkeit die Beziehung zwischen einem Mandanten und seinem Anwalt oder Bevollmächtigten und bezieht sich auf das Vertrauen, das der eine dem anderen entgegenbringt. Die Internationale Organisation für Normung (ISO) definiert Vertraulichkeit als die Sicherstellung, dass Informationen nur denjenigen zugänglich sind, die zum Zugriff berechtigt sind (2005). In der IT bedeutet dies, dass die auf Computern gespeicherten sensiblen Informationen nicht an unbefugte Personen, Programme oder Geräte weitergegeben werden. So muss beispielsweise verhindert werden, dass eine beliebige Person mit Zugang zu einem Netzwerk gängige Tools verwendet, um den Datenverkehr abzuhören und wertvolle private Informationen abzufangen.

###### Integrität

Bei der (Daten-)Integrität geht es um die zuverlässige, vollständige und fehlerfreie Übertragung und den Empfang oder die Speicherung von Daten. Dies bedeutet, dass die ursprünglichen Daten nicht verändert oder verfälscht wurden; insbesondere entsprechen die Daten den Erwartungen.

Wenn die Daten durch elektronische Beschädigung mittels Software oder durch physische Beschädigung des Datenträgers verändert wurden, sind die Daten unlesbar. Wenn wir eine Datei herunterladen, überprüfen wir ihre Integrität, indem wir ihren Hash berechnen und ihn mit dem veröffentlichten Hash vergleichen. Ohne diese Prüfung könnte jemand zum Beispiel einen Trojaner in ein Installationsprogramm für Microsoft Windows einbauen.

Grundkonzepte der Kryptologie

###### Verfügbarkeit

Auch wenn es nichts mit Kryptologie zu tun hat, geht es bei der Informationssicherheit um die Verfügbarkeit von Informationen gegenüber Bedrohungen wie Angriffen (z. B. DoS (Denial of Service), Störfällen (z. B. Stromausfällen) oder Naturkatastrophen (z. B. Erdbeben). Um dies zu erreichen, ist es am besten, einen Sicherheitsspielraum zu haben und Redundanz einzuplanen, insbesondere

* parallel redundante Failover-Hardware, wie z. B. einen Server oder ein Netzwerk. Dieses System ist ständig in Betrieb, so dass bei einer festgestellten Störung des Primärsystems die Verarbeitung automatisch umgeschaltet werden kann und ein
* Eindringen in das System verhindert wird, indem die Netzverkehrsmuster auf Anomalien überwacht und der Netzverkehr ggf. gesperrt wird.

###### Authentifizierung

Das Wort authentisch kommt aus dem Griechischen *authentikós* und bedeutet „echt oder unverfälscht“. Authentifizierung ist also der Nachweis von etwas (oder jemandem) als „authentisch“. So könnte es darum gehen, die Identität einer Person oder die Herkunft eines Objekts zu bestätigen. In der IT bedeutet Authentifizierung

* „Nachweis der Identität eines Benutzers oder seiner Berechtigung, auf ein Objekt zuzugreifen“ d. h. einen Computer davon zu überzeugen, dass eine Person diejenige ist, die sie nach der Identifizierung vorgibt zu sein, und
* „Nachweis, dass eine Nachricht nicht verändert oder verfälscht wurde“ (IBM, o. J.).

Eine Person weist ihre Identität durch einen Beleg nach. Es gibt verschiedene Möglichkeiten, diesen Beleg abzufragen, insbesondere auf Computern. Die gängigsten Methoden sind PIN-Nummern und Passwörter, aber auch Chipkarten oder biometrische Identifizierungsverfahren wie Fingerabdrücke oder Iris-Scans können diese Aufgabe übernehmen.

Ein häufiges Szenario ist der Man-in-the-Middle-Angriff (MITM), bei dem der Angreifer die Identität des anderen Kommunikationsteilnehmers annimmt. Zur Lösung dieses Problems werden Zertiﬁkate, digitale Signaturen von Dritten, eingesetzt. Dies kann entweder, wie beim Web of Trust in OpenPGP, durch Signaturen zwischen Personen, die sich untereinander bekannt sind, oder, wie auf sicheren Seiten in einem Webbrowser, durch eine Signatur einer vorbehaltlos vertrauenswürdigen, zentralen Zertifizierungsstelle realisiert werden.

###### Non-Repudiation

Repudiation ist ein juristischer Begriff für die Leugnung einer rechtlichen Bindung (z. B. einer Vereinbarung oder Verpflichtung). Eine Repudiation liegt vor bei einer Person, die a) sich weigert, eine rechtliche Verpflichtung zu akzeptieren oder mit ihr in Verbindung gebracht zu werden, b) sich weigert, die Gültigkeit der rechtlichen Verpflichtung anzuerkennen (z. B. seine Unterschrift), und c) sich weigert, die rechtliche Verpflichtung zu erfüllen.

Die Non-Repudiation (Nichtabstreitbarkeit) stellt sicher, dass ein Vertrag später von keiner der Parteien bestritten werden kann und die Identität des mutmaßlichen Absenders oder Empfängers einer bestimmten Nachricht feststeht. In der Informatik bedeutet dies, dass die Authentifizierung im Nachhinein nicht mehr widerlegt werden kann. Dies wird durch eine digitale Signatur erreicht. So beweist z. B. ein elektronischer Beleg, dass ein bestimmter Nutzer eine Nachricht, z. B. eine Anweisung zum Kauf eines Artikels in einer Online-Auktion, gesendet hat. Wenn in der E-Mail steht, dass Thomas sie geschickt hat, kann Thomas dies nicht abstreiten. In der heutigen globalisierten Wirtschaft, in der persönliche Vereinbarungen von Angesicht zu Angesicht oft nicht mehr möglich sind, ist die Nichtabstreitbarkeit für einen sicheren Geschäftsverkehr unerlässlich.

### Historischer Überblick

Die Geschichte der Kryptografie reicht mindestens 4000 Jahre zurück. Wir unterscheiden drei Perioden (Davies, 1997):

1. Bis zum zwanzigsten Jahrhundert waren die Methoden ganz klassisch, im Wesentlichen Stift und Papier.
2. Zu Beginn des zwanzigsten Jahrhunderts wurden sie durch efﬁzientere und ausgefeiltere Methoden auf komplexen elektromechanischen Maschinen ersetzt. Dabei handelte es sich hauptsächlich um Rotor-Chiffriermaschinen für die polyalphabetische Substitution, wie z. B. die Enigma, die von den Achsenmächten während des Zweiten Weltkriegs eingesetzt wurde.
3. Seit dem frühen zwanzigsten Jahrhundert hat die Digitalisierung, also die Ablösung analoger Geräte durch digitale Computer, immer komplexere Methoden möglich gemacht. Die meistgetesteten Algorithmen sind DES (bzw. seine dreifache Iteration 3DES) und sein Nachfolger AES für die symmetrische Kryptografie sowie RSA und sein Nachfolger ECC für die asymmetrische Kryptografie.

###### Klassische Kryptografie

Von der Antike bis zum Ersten Weltkrieg wurde Kryptografie von Hand ausgeführt und war daher in ihrer Komplexität und Länge auf höchstens ein paar Seiten beschränkt. Obwohl die Grundlagen der Kryptoanalyse bekannt waren, wurden sie wegen der fehlenden Automatisierung in der Kryptografie nicht wirklich angewandt. Daher war die Kryptoanalyse bei einer ausreichenden Menge an Geheimtext und entsprechendem Aufwand praktisch immer erfolgreich. Die Araber waren die Ersten, die sich der Kryptoanalyse bedienten. Sie verwendeten sowohl Substitutions- als auch Transpositionschiffren und kannten sowohl die Häufigkeitsverteilung der Buchstaben als auch den wahrscheinlichen Klartext bei der Kryptoanalyse.

Skytale

Eine Skytale (aus dem Griechischen für „Stock“ oder „Stab“) besteht aus einem Stab, der mit einem Pergamentband umwickelt ist, auf dem eine geheime Botschaft geschrieben steht. Das Pergamentband wird auf den Stab gewickelt und beschrieben. Die Geheimschrift ist nur lesbar, wenn das Pergament um einen Stab mit demselben Durchmesser gewickelt wird. Es handelt sich um eine Transpositionschiffre: Sie vermischt oder transponiert die Buchstaben des Klartextes (OPTED, o. J.).

Caesar-Chiffre

Die Caesar-Chiffre ist eine der einfachsten und bekanntesten Verschlüsselungen, benannt nach Julius Caesar, der sie zur Nachrichtenübermittlung mit seinen Generälen verwendete. Bei dieser Substitutionschiffre wird jeder Buchstabe des Klartextes durch einen festen anderen Buchstaben ersetzt. Jeder Buchstabe

Grundkonzepte der Kryptologie

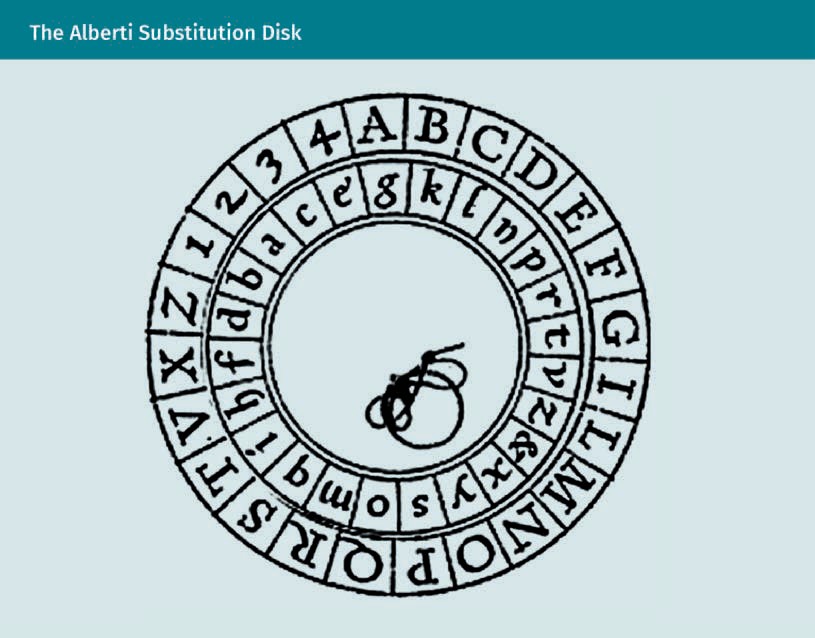
des Klartextes wird um die gleiche Anzahl von Positionen im Alphabet verschoben, d. h. jeder Buchstabe des Klartextes wird durch einen Buchstaben ersetzt, der eine bestimmte Anzahl von Positionen weiter hinten im Alphabet steht (Bauer, 2000).

Bacon-Chiffre

Francis Bacons Chiffre aus dem Jahr 1605 ist eine Anordnung der Buchstaben a und b in Fünfergruppen (von denen es 25 = 32 gibt), die wiederum jeweils einen Buchstaben des englischen Alphabets darstellen (26). Heute würden wir dies als Code bezeichnen, aber damals illustrierte dies den bedeutenden Grundsatz, dass nur zwei verschiedene Zeichen ausreichen, um eine Information zu übermitteln (Tudhope, 1993).

Alberti-Scheibe

Im Jahr 1470 beschrieb Leon Battista Alberti die erste Chiffrierscheibe, die die Buchstaben des Alphabets zyklisch verschiebt. Er empfahl, die Verschiebung nach jeweils drei oder vier Wörtern zu ändern und so eine polyalphabetische Chiffre zu konzipieren, bei der die gleichen Buchstaben durch andere ersetzt werden. Mehr als vier Jahrhunderte später setzte die US-Armee im Ersten Weltkrieg das gleiche Gerät ein (de Leeuw & Bergstra, 2007).



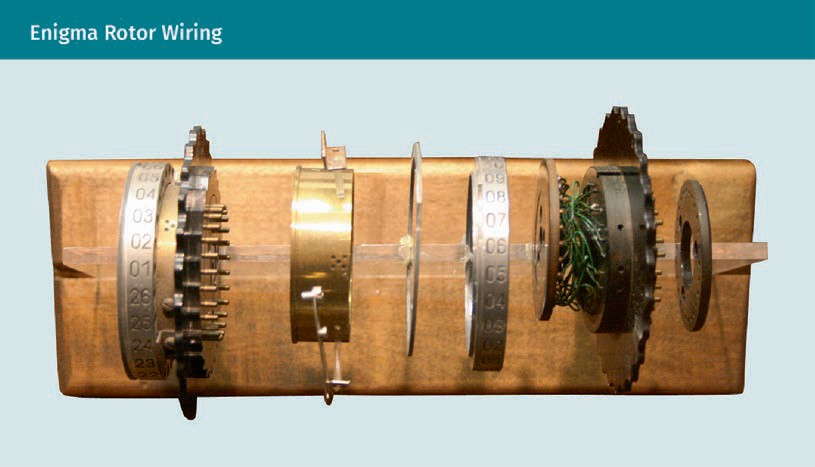
ADFGVX-Chiffre

Die berühmteste Chiffre des Ersten Weltkriegs war die deutsche Fraktionierungschiffre ADFGVX, die am 5. März 1918, also gegen Ende des Ersten Weltkriegs, eingeführt wurde. Fritz Nebel, ein Nachrichtenoffizier der Armee, entwickelte die vermeintlich unknackbare ADFGVX-Chiffre für den Einsatz

durch mobile militärische Einheiten. Sie ersetzte die 26 Buchstaben und 10 Ziffern durch eine 6 x 6-Matrix mit Paaren aus den Buchstaben A, D, F, G, V und X. Der resultierende Text wurde dann in Form eines Rechtecks geschrieben, und die Spalten wurden in der durch den Schlüssel angegebenen Reihenfolge gelesen (Dooley, 2013, Kapitel 5).

###### Elektromechanische Kryptografie mit Rotor-Chiffriermaschinen

Die ersten Erfolge bei der Mechanisierung der Kryptografie stellten sich kurz nach dem Ersten Weltkrieg ein. Nachdem sich Telegrafie und Radio in den 1920er Jahren durchgesetzt hatten, entstand ein Bedarf an Kryptografie und die Entwicklung von Rotor-Chiffriermaschinen nahm an Fahrt auf. Dieser Automatisierungsschritt ermöglichte schnellere Abläufe mit höherer Komplexität und geringeren Fehlerquoten als bei der manuellen Verschlüsselung (Dooley, 2013, Kapitel 5).



Bei der Enigma-Maschine sind die Rotoren übereinander angeordnet. Die Drehung eines Rotors bewirkt, dass sich der nächste um 1 / 26 einer vollen Umdrehung dreht. Während des Betriebs gibt es einen elektrischen Pfad durch alle Rotoren. Durch Schließen des Tastenkontakts des Klartextbuchstabens auf einer schreibmaschinenähnlichen Tastatur wird ein Strom an einen der Kontakte des Startrotors abgegeben. Der Strom fließt dann durch die Kabel der gestapelten Rotoren (abhängig von der Drehlage) und endet an einer Anzeige, wo er die Lampe des Geheimtextbuchstabens zum Leuchten bringt.

###### Digitale Kryptografie

Die weitere Entwicklung der Verschlüsselungsmaschinen führte zu noch schnelleren Rotor-Chiffriermaschinen mit elektronischer Ersetzung ihrer Rotoren, aber das Konzept blieb dasselbe: verschobene monoalphabetische Substitutionen führten zu einer polyalphabetischen Substitution. Mit dem Aufkommen des Digitalcomputers in den 1980er Jahren änderte sich an der Entwicklung nichts mehr. Allerdings sind solche buchstabenweisen Substitutionen immer noch „linear über die Buchstaben“, so dass der aus einem Klartext gewonnene Geheimtext verrät, wie man alle Buchstaben eines Klartextes von (höchstens) gleicher Länge entschlüsseln

Grundkonzepte der Kryptologie

kann. Eine buchstabenweise Substitution hat also eine geringe Verteilung (Diffusion) zur Folge und bewirkt kaum Veränderungen. Eine optimale Diffusion ist immer dann erreicht, wenn die Änderung eines einzigen Buchstabens des Klartextes die Änderung der Hälfte der Buchstaben des Geheimtextes bewirkt. Wenn der Angreifer Zugang zu den Geheimtexten vieler Klartexte hat, die er möglicherweise selbst ausgewählt hat, kann er den Schlüssel aus den Geheimtexten von zwei Klartexten ermitteln, die sich in einer einzigen Position unterscheiden.

DES

Der Computer ermöglichte demgegenüber eine wesentlich bessere Diffusion. Computer ermöglichten zudem die Kombination von Substitutionen (z. B. Caesar-Chiffre) mit Transpositionen (z. B. Skytale), wodurch eine wesentlich bessere Streuung erreicht wurde. Dies führte 1976 zur Entwicklung einer der am häufigsten verwendeten Verschlüsselungen der Geschichte, dem Data Encryption Standard (DES) (Davies, 1997).

AES

Im Januar 1997 kündigte das US National Institute of Standards and Technology (NIST) einen öffentlichen Wettbewerb für einen Nachfolger des veralteten DES an, den Advanced Encryption Standard (AES). Im Oktober 2000 wurde Rijndael, das von Joan Daemen und Vincent Rijmen entwickelt wurde, ausgewählt und damit zum AES (NIST, 2000).

Da es aufgrund der verbesserten Rechenleistung möglich war, den feststehenden 56-Bit-DES-Schlüssel durch eine vollständige Schlüsselsuche zu finden, verlangten die NIST-Spezifikationen für den AES eine höhere Schlüssellänge, sofern dies erforderlich werden sollte. Rijndael erwies sich nicht nur als immun gegen die ausgeklügeltsten bekannten Angriffe, wie z. B. differentielle Kryptoanalyse, und als elegantes und einfaches Design, sondern war auch so klein, dass es auf Chipkarten implementiert werden konnte (mit weniger als 10.000 Byte Code) und gleichzeitig ausreichend flexibel für höhere Schlüssellängen (Daemen & Rijmen, 1999; 2002).

Public-Key-Kryptografie

Seit den 1980er Jahren hat das Aufkommen der Public-Key-Kryptografie im Informationszeitalter digitale Signaturen und Authentifizierungen ermöglicht und damit Möglichkeiten geschaffen, durch elektronische Informationen das zu ersetzen, was früher durch physische Dokumente erreicht wurde. Difﬁe und Hellman (1976) waren die Ersten, die öffentlich eine asymmetrische Verschlüsselung vorschlugen. Konzeptionell beruht sie auf einer Falltürfunktion (genauer gesagt, der diskreten Exponentialfunktion). Diese invertierbare Funktion ist leicht zu berechnen, doch ihre Umkehrung ist kaum zu berechnen, wenn keine zusätzlichen Informationen vorliegen: der geheime Schlüssel (Difﬁe & Hellman, 1976).

Zum Verschlüsseln wird die Funktion angewendet. Zum Entschlüsseln wird die Umkehrung mit dem geheimen Schlüssel verwendet. Im Konzept von Difﬁe und Hellman (1976) zum Beispiel ist diese Funktion die Exponentialfunktion, allerdings in einem anderen Definitionsbereich als dem der reellen Zahlen, den wir gewohnt sind. Tatsächlich haben sie nur ein Verfahren zum Austausch eines geheimen Schlüssels über einen unsicheren Kanal geschaffen. Der kryptografische RSA-Algorithmus (Rivest et al., 1978) oder der Elgamal-Algorithmus (Elgamal, 1985) wendeten dieses Konzept erstmals an. Diese Algorithmen ermöglichten nicht nur die Chiffrierung mit einem öffentlichen Schlüssel, wodurch das Problem der geheimen Kommunikation gelöst wurde, sondern machten auch digitale Signaturen mit dem privaten Schlüssel zur Chiffrierung möglich, was wohl den kommerziellen Durchbruch einleitete. Diese Algorithmen haben immer noch Bestand, aber andere, wie ECC, gelten heute als effizienter und ebenso sicher.

### Sicherheitskriterien

Kerckhoffs‘ Prinzip Der Geheimtext muss auch dann sicher sein, wenn mit Ausnahme des Schlüssels alles über ihn allgemein bekannt

ist.

Ideale Diffusion Das Shannonsche Ideal besagt, dass sich die Hälfte der Bits im Geheimtext ändert,

wenn sich ein Bit im Klartext oder Schlüssel ändert.

Hashfunktion Ein Algorithmus, der aus einer Eingabe mit variabler Größe eine Ausgabe mit fester Größe erzeugt, wird als Hashfunktion

bezeichnet.

Prüfsummenfunktion Ein Algorithmus, der schnell und ohne Kollisionen aus einer Eingabe mit variabler Größe eine Ausgabe mit fester Größe erzeugt, wird als Prüfsummenfunktion bezeichnet.

Kerckhoffs‘ Prinzip postuliert die Unabhängigkeit der Sicherheit eines Kryptoalgorithmus von dessen Geheimhaltung. Die Kenntnis des Schlüssels gefährdet eine einzelne Verschlüsselung, während die Kenntnis des Algorithmus alle jemals durchgeführten Verschlüsselungen gefährdet. Ein öffentlich bekannter Algorithmus garantiert, dass die Schwierigkeit der Entschlüsselung nur von der Kenntnis des Schlüssels, nicht aber vom Algorithmus abhängt. Je häufiger er verwendet wird, desto wahrscheinlicher wird es, dass der Algorithmus irgendwann bekannt wird. Damit der Algorithmus brauchbar ist, muss er also sicher sein, auch wenn er öffentlich bekannt ist.

###### Shannons Kriterien

Shannons Prinzipien der Konfusion und der Diffusion geben Kriterien für eine nicht ableitbare Beziehung zwischen dem Geheimtext und dem Schlüssel in Bezug auf den Klartext vor. Während die Ausgabe der Chiffrierung, der Geheimtext, deterministisch von der Eingabe, dem Klartext, und dem Schlüssel abhängt, zielt der Algorithmus darauf ab, diese Beziehung zu verschleiern, um sie so kompliziert, verflochten und verworren wie möglich zu machen. Jeder Buchstabe der Ausgabe (Geheimtext) hängt von jedem Buchstaben der Eingabe (Klartext) und des Schlüssels ab.

Shannon (1949) formulierte es so: „Der Feind kennt das System“, was heute als Shannons Maxime bekannt ist. Das Gegenbeispiel wäre, sich auf einen potenziell schwachen, aber unbekannten Algorithmus zu verlassen: „Sicherheit durch Dunkelheit“. Es gibt zahlreiche historische Beweise für die Vergeblichkeit eines solchen Vorhabens (z. B. die ADFGVX-Chiffre).

### Hashfunktionen

Eine Hashfunktion ist ein Algorithmus, der aus einer Eingabe variabler Größe (z. B. einer Textdatei, einer Bilddatei oder einem komprimierten Archiv) eine Ausgabe fester (Byte-)Größe (in der Regel 16 bis 64 Byte) erzeugt. Aus einer Eingabe variabler Größe berechnet eine Hashfunktion eine Ausgabe fester Größe, die als Hashwert oder, je nach Verwendung, auch als (Nachrichten-)Digest, digitaler Fingerabdruck oder Prüfsumme bezeichnet wird. Der md5-Hash (Größe von 16 Bytes) des Wortes „Schlüssel“ in hexadezimaler Kodierung lautet beispielsweise 0f511d362346c2f658bb594d5b04224e.

Prüfsumme

Eine (einfache) Hashfunktion oder Prüfsummenfunktion sollte die folgenden Merkmale erfüllen:

* schnelle Berechnung eines Hashwertes für beliebige Daten und
* geringe Wahrscheinlichkeit, dass zwei (nur leicht) unterschiedliche Nachrichten denselben Hashwert ergeben.

Eine Hashfunktion sollte sich daher wie eine Zufallsfunktion verhalten, gleichzeitig aber deterministisch und schnell zu berechnen sein.

Grundkonzepte der Kryptologie

So wäre zum Beispiel die simpelste Prüfsumme die Summe der Bits in der Eingabe, die auf die feste Ausgabegröße abgeschnitten wird. Hierbei handelt es sich fast um eine Hashfunktion: sie ist schnell und und es ist in der Tat unwahrscheinlich, dass zwei verschiedene Nachrichten denselben Hashwert ergeben. Allerdings erhält man leicht zwei fast identische Nachrichten mit dem gleichen Hashwert. Winzige Änderungen könnten so unentdeckt bleiben.

Kryptografische Hashfunktion

Die Ausgabezeichenkette mit fester Länge, die eine kryptografische Hashfunktion aus einer Zeichenkette beliebiger Länge (z. B. einer wichtigen Nachricht) erzeugt, ist also eine Art nicht nachahmbarer Signatur. Eine Person, die den Hashwert kennt, kann die ursprüngliche Nachricht nicht daraus ableiten; nur die Person, die die ursprüngliche Nachricht kennt, kann nachweisen, dass der „Hashwert“ aus dieser Nachricht erzeugt wurde. Heer (2008) beschreibt es so: „Eine Hashfunktion ist schwach kollisionsresistent, wenn das Auffinden eines Urbildes rechnerisch nicht praktikabel ist“, d. h. aus einer gegebenen Ausgabe eine (noch unbekannte) entsprechende Eingabe zu finden. Starke Kollisionsresistenz beschreibt somit eine Hashfunktion, für die „das Auffinden einer Kollision rechnerisch nicht praktikabel ist“ (Heer, 2008), d. h. zwei Eingaben mit der gleichen Ausgabe zu finden. Andernfalls könnte ein Angreifer eine autorisierte Nachricht durch eine unautorisierte ersetzen.

###### Anwendungsfälle

Hashfunktionen werden für die Abfrage von Datenbankeinträgen zur Fehlererkennung und -korrektur (z. B. für Datenintegritätsprüfungen) und in der Kryptografie zur Identifizierung von Daten bei Verschleierung ihres Inhalts für Datenauthentizitätsprüfungen und Authentifizierung (z. B. zum Speichern von Passwörtern, digitalen Signaturen) verwendet.

In der Praxis sind die meisten Hashfunktionen, selbst bei Prüfsummen, kryptografisch. Sie sind zwar langsamer, aber auf den meisten Geräten immer noch schnell genug. Manchmal – z. B. bei der Speicherung von Passwörtern – sind sie absichtlich langsam, damit die Passwörter nicht durch eine erschöpfende Suche unter den wahrscheinlichen Kandidaten anhand ihrer Hashwerte schnell gefunden werden können. Die am häufigsten eingesetzte kryptografische Hashfunktion war MD5, üblicherweise mit einer Ausgabelänge von 128 Bit, erfunden von Ron Rivest vom Massachusetts Institute of Technology im Jahr 1991.

SHA

Die Secure Hash Algorithms (SHA) sind kryptografische Hashfunktionen, die von der National Security Agency (NSA) und dem NIST bereitgestellt werden. SHA-1 ist der Nachfolger von MD5 mit einer Hashgröße von 160 Bit. MD5 war eine frühere, weit verbreitete Hashfunktion, die immer mehr Sicherheitsmängeln zum Opfer fiel (sie wurde aber nicht geknackt, d. h. es ist kein rechnerisch praktikabler Weg bekannt, eine Eingabe aus einer bestimmten Hashfunktion zu erzeugen). SHA-1 wurde im Digital Signature Algorithm (DSA) verwendet, wie er im Digital Signature Standard (DSS) der Internet Engineering Task Force vorgeschrieben ist.

Zwischenzeitlich gab es die drei SHA-Algorithmen SHA-1, SHA-2 und SHA-3 (veröffentlicht 2015) mit zunehmender Sicherheit, die die Unzulänglichkeiten der jeweiligen Vorgänger abschwächten. „SHA-2“ erlaubt Hashes mit unterschiedlichen Bitgrößen. Die Bitanzahl wird an das Präﬁx „SHA“ angehängt, z. B. „SHA-224“, „SHA-256“, „SHA-384“ und „SHA-512“.

Zusammenfassung

Kryptografie schützt Informationen, indem sie Daten so mischt, dass nur zusätzliche geheime Informationen, der Schlüssel, die ursprünglichen Daten wiederherstellen können. Bis Ende der 1970er Jahre war die symmetrische oder Ein-Schlüssel-Kryptografie üblich. Es folgte die asymmetrische Kryptografie.

Die Sicherheit asymmetrischer kryptografischer Algorithmen, auch Public-Key-Algorithmen genannt, beruht auf der inefﬁzienten Berechnung mathematischer Funktionen auf den ganzen Zahlen. Die Sicherheit symmetrischer kryptografischer Algorithmen basiert darauf, dass kleine Unterschiede in der Eingabe und im Schlüssel zu großen Unterschieden in der Ausgabe führen. Eine gute Verschlüsselung, wie sie durch Standardalgorithmen wie AES oder RSA erreicht wird, ist computertechnisch praktisch nicht zu knacken.

Eine Hashfunktion ist ein Algorithmus, der aus einer Eingabe mit variabler Größe eine Ausgabe mit fester Größe erzeugt. Wir unterscheiden zwischen kryptografischen (oder Einweg-) und Prüfsummen- (oder nicht-kryptografischen) Hashfunktionen. Für kryptografische Hashfunktionen haben wir zudem die schwache und die starke Kollisionsresistenz definiert, die davon abhängen, ob es praktisch unmöglich ist, eine unbekannte Nachricht für einen bestimmten Hashwert zu finden und ob es praktisch unmöglich ist, zwei Nachrichten mit demselben Hashwert zu finden.



# Lektion 2

## Symmetrische Kryptosysteme

#### LERNZIELE

Nach der Bearbeitung dieser Lektion werden Sie wissen, …

… welche grundlegenden symmetrischen kryptografischen Algorithmen es gibt: Substitution und Transposition.

… welche die einzige kryptografisch vollkommen sichere Chiffre ist: das One-Time-Pad.

… was die modernen Algorithmen Data Encryption System und Advanced Encryption System ausmacht.

… welche vielfältigen Einsatzmöglichkeiten es für (kryptografische) Hashfunktionen gibt.

DL-E-DLMCSEAITSC02-U02

1. Symmetrische Kryptosysteme

### Einführung

Bis zum Ende der 1970er Jahre, vor der Veröffentlichung von Difﬁe und Hellman (1976) und Rivest et al. (1978), waren alle (bekannten) kryptografischen Algorithmen symmetrisch (oder Ein-Schlüssel-Verfahren), d. h. sie verwendeten denselben Schlüssel zum Ver- und Entschlüsseln. Jeder historische Algorithmus, so ausgeklügelt er auch gewesen sein mag, sei es die Caesar-Chiffre, die Skytale oder die Enigma, war symmetrisch.

Symmetrische Algorithmen verarbeiten die Eingabe als Zeichenkette (aus Bits oder Buchstaben) durch (iterierte) Substitutionen und Transpositionen. Im Gegensatz dazu beruhen asymmetrische Algorithmen auf einem rechnerisch schwierigen Problem, z. B. der Faktorisierung einer zusammengesetzten Zahl in ihre Primfaktoren, und betrachten die Eingabe als eine natürliche Zahl. Die einzige vollkommen sichere Chiffre ist das One-Time-Pad, bei dem der Schlüssel so lang wie der Klartext ist und der Geheimtext durch buchstabenweises Hinzufügen jedes Buchstabens des Schlüssels zu dem entsprechenden (d. h. an der gleichen Stelle befindlichen) Buchstaben des Klartextes gewonnen wird. Für komplexere Nachrichten, z. B. Text-, Bild- oder Videonachrichten, ist ein großer Schlüssel jedoch unpraktisch. Heute bedeutet das, dass zum Verschlüsseln einer Festplatte eine weitere Festplatte benötigt wird, auf der sich der Schlüssel befindet. Um die kürzere Schlüssellänge zu kompensieren, erzeugen moderne Algorithmen im Idealfall so viele Verflechtungen, dass sie eine nahezu perfekte Diffusion erreichen. Mit anderen Worten: Die Änderung eines einzigen Bits der Eingabe oder des Schlüssels bewirkt die Änderung von etwa der Hälfte der Ausgabebits. Moderne Algorithmen wie der Data Encryption Standard (DES) oder der Advanced Encryption Standard (AES) sind Blockchiffren mit Substitutions- und Permutationsverfahren, d. h., sie verschlüsseln jeweils ein Datenpaket durch wiederholte Transpositionen und Substitutionen.

### Substitution und Transposition

Die beiden grundlegenden Verfahren zur Verschlüsselung sind Transposition und Substitution:

* Eine Transposition verändert die Reihenfolge (d.h. transponiert oder permutiert) der Zeichen im Text, nicht aber die Zeichen selbst.
* Bei einer Substitution wird jedes Zeichen im Text durch ein anderes (eine andere Gruppe von) Zeichen ersetzt, jedoch nicht die Reihenfolge der Zeichen geändert.

Die historisch gesehen prototypischen Algorithmen für diese beiden Verfahren sind:

* Caesars Substitutionschiffre, die jeden Buchstaben im Klartext um drei Positionen vorrückt und z. B. A als D, B als E usw. verschlüsselt, sowie
* die Permutation des Klartextes durch die Skytale (Stab von Licurgo), bei der das Pergamentband um den Stab gewickelt wird und der Text horizontal darauf geschrieben wird.

Symmetrische Kryptosysteme

Wir werden sehen, dass selbst bei vielen möglichen Schlüsseln ein Algorithmus – wie z. B. der durch eine beliebige Permutation des Alphabets gegebene, der fast 280 Schlüssel hat – leicht geknackt werden kann, wenn er Regelmäßigkeiten, wie die Häufigkeit der Buchstaben, beibehält. Substitutions- und Permutationsnetze verbinden und iterieren diese beiden komplementären prototypischen Algorithmen, um dieses Ziel zu erreichen.

###### Substitutionschiffren

Bei einer Substitutionschiffre bestimmt der Schlüssel die Substitutionen des Klartextalphabets (betrachtet als eine Menge von Einheiten von Zeichen wie einzelne Buchstaben oder Buchstabenpaare) durch das Geheimtextalphabet. Wenn zum Beispiel die Einheiten des Klartextes und des Geheimtextes beide Buchstaben des lateinischen Alphabets sind, dann vertauscht eine Substitution die Buchstaben des lateinischen Alphabets. Handelt es sich bei der Substitutionschiffre um eine monoalphabetische Chiffre (z. B. die Caesar-Chiffre), so wird auf jeden Buchstaben des Klartextes unabhängig von seiner Position dieselbe Substitution angewendet. Wenn die Substitutionschiffre polyalphabetisch ist (wie die Enigma), dann variiert die Substitution mit der Position des Buchstabens im Klartext. Zum Verschlüsseln wird jede alphabetische Einheit des Klartextes durch die substituierte alphabetische Einheit ersetzt und zum Entschlüsseln entsprechend invertiert. Die monoalphabetische Substitution gilt als unsicher, weil die Häufigkeit der Zeichen, die im Klartext und im Geheimtext zu finden sind, gleich bleibt. Im Englischen zum Beispiel genügen etwa 25 Buchstaben des Geheimtextes für eine Kryptoanalyse. Das zentrale Verfahren zur Reduktion der Erhaltung der Häufigkeit von Einzelbuchstaben im Geheimtext, die so genannte polyalphabetische Substitution, besteht in der Verwendung mehrerer Geheimalphabete.

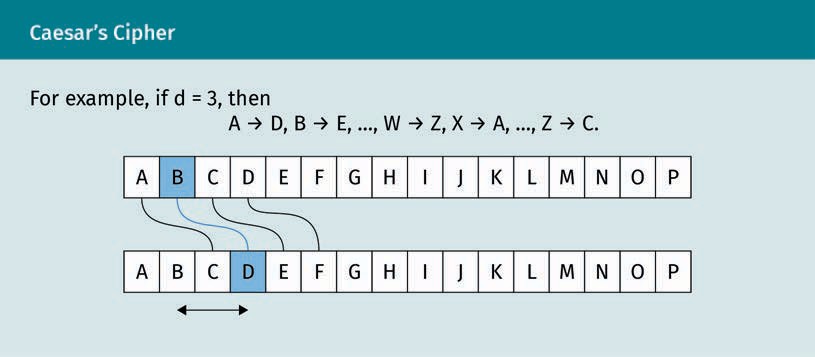
Verschiebung um einen festen Abstand

Die einfachste Substitutionsschiffre ist eine zyklische Verschiebung des Klartextalphabets, die so genannte Caesar-Chiffre. Die römischen Kaiser Caesar (100-44 v. Chr.) und Augustus (63-14 v. Chr.) bedienten sich dieser Methode (Bauer, 2016). Wir legen einen Abstand d zwischen den Buchstaben in alphabetischer Reihenfolge fest, d. h. eine Zahl zwischen 0 und 25, und verschieben dann jeden Buchstaben des (lateinischen) Alphabets um diesen Abstand d (nach vorne). Wir stellen uns dabei vor, dass Alphabet sei kreisförmig, d. h. die Buchstaben sind in einem Ring angeordnet, so dass die Verschiebung eines Buchstabens am Ende des Alphabets zu einem Buchstaben am Anfang des Alphabets führt.

Substitutionschiffre Bei dieser Chiffre wird jede alphabetische Einheit des Klartextes durch eine entsprechende alphabetische Einheit ersetzt.

Caesar-Chiffre Diese Substitutionsschiffre verschiebt die

alphabetische Position jedes Klartextbuchstabens um den gleichen Abstand.



Es gibt 26 Schlüssel (einschließlich des trivialen Schlüssels d = 0). Bei der Caesar-Chiffre wird jeder Buchstabe des Alphabets um den gleichen Abstand verschoben. Zur Entschlüsselung wird jeder Buchstabe um den negativen Abstand –d verschoben, d. h. d Positionen nach hinten.

Substitution durch Permutation der Buchstaben des Alphabets

Anstatt jeden Buchstaben durch einen um den gleichen Abstand d verschobenen Buchstaben zu ersetzen, können wir jeden Buchstaben durch einen beliebigen Buchstaben ersetzen, wie im folgenden Beispiel.

A B … Y Z

…

E Z … G A

Wir mischen die Buchstaben untereinander. Auf diese Weise erhalten wir 26 · 25 · 1 = 26! > 1026

Schlüssel (was in etwa der Anzahl von Passwörtern entspricht, die mit 80 Bit möglich sind).

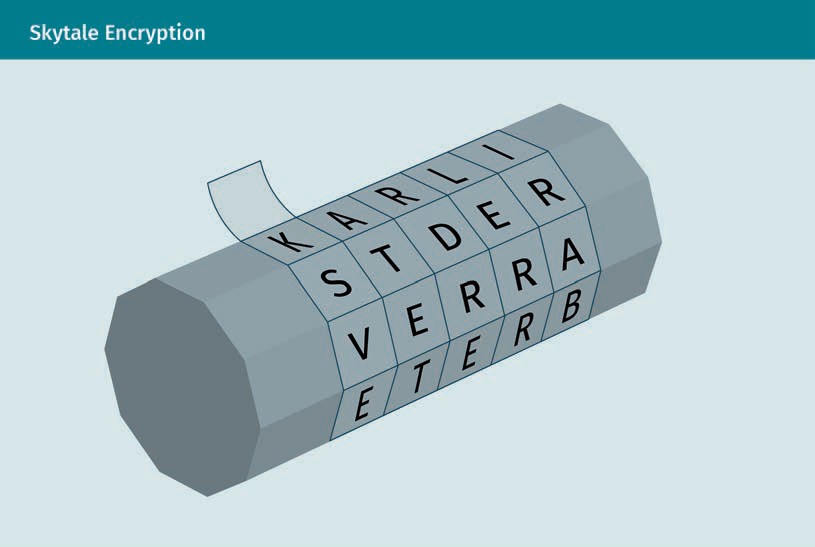
###### Transpositions- bzw. Permutationschiffren

Eine Transpositionschiffre verschlüsselt den Klartext durch Permutation seiner Einheiten (und entschlüsselt ihn durch die umgekehrte Permutation). Jede alphabetische Einheit bleibt gleich; die Verschlüsselung hängt lediglich von der Position der Einheiten ab.

Die Skytale ist ein Stab, mit dem die Spartaner wie folgt verschlüsselt haben: (1) man wickelt ein Band um einen schmalen Stab, (2) man beschriftet dieses Band horizontal, also entlang der längeren Seite und

(3) man wickelt das Band ab (Bauer, 2016). Die auf dem Band transponierten Buchstaben konnten nur mit einem Stab gleichen Umfangs (dem Schlüssel) auf die gleiche Weise entziffert werden, wie der Text verschlüsselt wurde: Man wickelte das Band um den Stab und las den Text darauf horizontal, entlang der längeren Seite.

Symmetrische Kryptosysteme



In diesem Fall ist der Schlüssel die Zahl, die sich aus der Anzahl der Buchstaben ergibt, die um den Umfang des Stabes herum angeordnet sind.

###### Sicherheit historischer Beispiele

Wenden wir nun die festgelegten Sicherheitskriterien auf die Substitutionsschiffre an.

Caesar-Chiffre

Diese einfache Substitutionschiffre verstößt gegen alle wünschenswerten Eigenschaften. Kerckhoffs‘ Prinzip besagt zum Beispiel, dass der Algorithmus öffentlich ist, sobald die Methode bekannt ist. In Anbetracht der geringen Anzahl von 25 Schlüsseln lässt sich der Geheimtext in kurzer Zeit durch einen Brute-Force-Angriff (Durchprobieren aller Möglichkeiten) knacken.

Substitution durch willkürliche Permutation der Buchstaben des Alphabets

Eine Ersetzung durch eine beliebige Permutation der Buchstaben des Alphabets hat 26 · 25 · 1 = 26! > 1026 Schlüssel, so dass ein Brute-Force-Angriff rechnerisch nicht durchführbar ist.

Dies verstößt jedoch gegen die Ziele der Diffusion und der Konfusion. Wenn der Schlüssel (Permutation des Alphabets) den Buchstaben α gegen den Buchstaben β austauscht, dann ergibt sich

* eine geringe Konfusion, da die Substitution von β im Schlüssel nur die Substitution jedes Buchstabens β im Geheimtext impliziert,
* eine geringe Diffusion, da die Substitution eines Buchstabens α im Klartext nur die Ersetzung des entsprechenden Buchstabens β im Geheimtext impliziert.

Brute-Force-Angriff Diese erschöpfende Schlüsselsuche überprüft jeden möglichen Schlüssel.

###### Produktchiffre

Produktchiffre Eine Zusammensetzung von Chiffren, bei der die Ausgabe einer Chiffre als Eingabe für

die nächste dient.

Eine Produktchiffre setzt Chiffren zusammen, d. h. bei einem zweifachen Produkt ist die Ausgabe der einen Chiffre die Eingabe der anderen. Der Geheimtext der Produktchiffre ist der Geheimtext der endgültigen Chiffre. Kombinieren wir Transpositionen nur mit Transpositionen oder Substitutionen nur mit Substitutionen, ist die erhaltene Chiffre wieder eine Transposition oder Substitution und kaum sicherer. Eine Mischung aus beiden, eine Transposition mit Substitutionen, kann die Chiffre jedoch tatsächlich sicherer machen.

Eine Fraktionierungschiffre ist eine Produktchiffre, bei der jedes Zeichen im Klartext durch eine Gruppe von Zeichen (in der Regel Paare) ersetzt wird und der erhaltene Geheimtext anschließend transponiert wird. Die bekannteste Fraktionierungschiffre war die ADFGVX-Chiffre, die von den deutschen Streitkräften während des Ersten Weltkriegs verwendet wurde (Bauer, 2016). Bei dieser Chiffre wurden die 26 Buchstaben des lateinischen Alphabets und zehn Ziffern in einer 6 × 6-Tabelle angeordnet und durch die Buchstabenpaare A, D, F, G, V und X ersetzt, die die Zeile und Spalte des Buchstabens oder der Ziffer angeben. Der resultierende Text wurde wie üblich von links nach rechts in die Zeilen einer Tabelle geschrieben und dann jede Spalte in der durch ein Schlüsselwort angegebenen Reihenfolge gelesen.

Wir werden uns nun ansehen, wie moderne Chiffren diese Idee einer Produktchiffre weiterentwickeln, um eine hohe Diffusion zu erreichen.

### Blockchiffren

Bei den klassischen Chiffren wurden in der Regel einzelne Buchstaben und manchmal auch Buchstabenpaare ersetzt. Systeme, die mit Dreiergruppen (Trigrammen) oder größeren Buchstabengruppen arbeiteten, wurden als zu umständlich angesehen und fanden keine breite Anwendung.

Stattdessen ist es sicherer, einen ganzen Block (von Buchstaben anstelle eines einzelnen Buchstabens) entsprechend dem Schlüssel zu substituieren. Das Alphabet dieses Austauschverfahrens wäre allerdings riesengroß, so dass dieses Ideal praktisch unerreichbar ist, insbesondere bei einer so beschränkten Hardware wie einer Chipkarte mit einem 8-Bit-Prozessor. Für einen Block von 16 Byte hätte diese Substitutionstabelle zum Beispiel die horrende Größe von 2128 • 16 Byte. In der modernen Kryptografie mit einem Schlüssel beträgt die Informationseinheit jedoch in der Regel 128 Bit, d. h. etwa 27 Buchstaben, während in der Kryptografie mit zwei Schlüsseln, die auf dem RSA-Algorithmus basiert, üblicherweise Einheiten von 2048 Bit, d. h. etwa 620 Buchstaben, verwendet werden.

So ersetzt AES zum Beispiel nur jedes Byte, jeden Eintrag in einem Block, eine Ersetzungstabelle von 28 = 256 Einträgen von 1 Byte (und transponiert anschließend die Einträge). Wir werden sehen, dass sich diese Verfahren so gut ergänzen, dass sie praktisch so sicher sind wie eine Substitution des gesamten Blocks.

Symmetrische Kryptosysteme

###### Block- und Stromchiffren

Bei einer Blockchiffre wird der Klartext in Blöcke gleicher Größe unterteilt und jeder Block mit einem gemeinsamen Schlüssel verschlüsselt; ein Block könnte zwar aus einem einzigen Zeichen bestehen, ist aber normalerweise größer. Bei DES beträgt die Blockgröße beispielsweise 64 Bit, bei AES 128 Bit.

Eine Stromchiffre unterteilt den Klartext in Einheiten, die normalerweise aus einem einzigen Zeichen bestehen, und verschlüsselt dann die i-te Einheit des Klartextes mit der i-ten Einheit eines Schlüsselstroms. Beispiele sind das One-Time-Pad, Rotor-Chiffriermaschinen (wie die Enigma) und Triple DES (bei dem die Ausgabe einer Verschlüsselung die Eingabe der nächsten Verschlüsselung ist). Bei einer Stromchiffre muss derselbe Abschnitt des Schlüsselstroms, der zur Verschlüsselung verwendet wurde, auch zur Entschlüsselung verwendet werden. Daher müssen die Schlüsselströme von Absender und Empfänger zu Beginn und danach ständig synchronisiert werden.

###### Feistelchiffren

Eine Feistelchiffre oder ein Substitutions- und Permutationsnetzwerk (SPN) gliedert den Text (Bytefolge) in n-Byte-Blöcke (z. B. n = 16 für AES) und verschlüsselt jeden Block durch Iteration (z. B. zehnmal bei AES und fünfmal in unserem prototypischen Modell) der folgenden drei Schritte:

* + 1. Addieren (XOR) des Schlüssels,
    2. Substituieren des Alphabets (das in Teilblöcken des Blocks arbeitet, z. B. auf jedem Byte), und
    3. Permutieren aller Teilblöcke eines Blocks.

Nach der Addition (XOR) des Schlüssels, der Ersetzung des Alphabets, z. B. im AES-Algorithmus (jedes Byte, hexadezimales Buchstabenpaar, durch ein anderes), und der Permutation des Textes (aus dem aktuellen Schritt, „Zustand“ genannt), z. B. im AES, werden die Einträge in jeder Zeile (und den Spalten) permutiert.

Diese beiden einfachen Verfahren, die Substitution des Alphabets und die Permutation des Textes, ergänzen sich gut. Sie führen nach wenigen Iterationen zu starker Konfusion und Diffusion. In der ersten und letzten Runde werden die Schritte vor und nach dem Hinzufügen des Schlüssels weggelassen, da sie die kryptografische Sicherheit nicht erhöhen. Da der Algorithmus öffentlich ist (Kerckhoffs' Prinzip), ist jeder Angreifer in der Lage, alle Schritte rückgängig zu machen, die keine Kenntnis des Schlüssels erfordern. Obwohl sich eine Feistelchiffre also von klassischen Kryptosystemen zu unterscheiden scheint, handelt es sich tatsächlich um eine Produktchiffre aus Transpositionen und Substitutionen.

Stromchiffre und Blockchiffre im Vergleich

Eine Stromchiffre arbeitet mit einzelnen Zeichen (z. B. einzelnen Bytes), während eine Blockchiffre mit Gruppen von Zeichen (z. B. jeweils 16 Bytes groß) arbeitet.

Substitutions- und Permutationsnetzwerk Bei dieser Chiffre wird jeder Block nach dem Hinzufügen eines Schlüssels iterativ substituiert und permutiert.

### Data Encryption Standard (DES)

Data Encryption

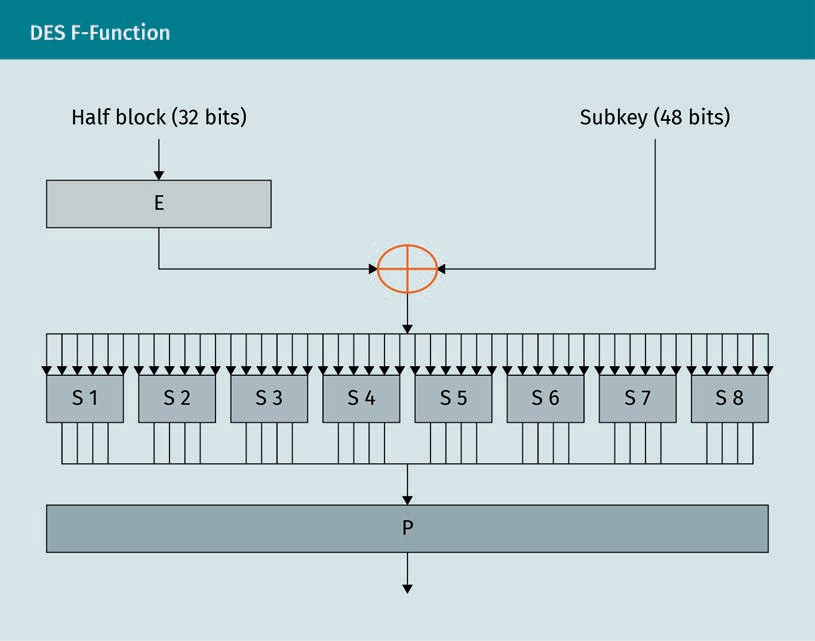
Standard Diese Blockchiffre hat eine Schlüssellänge von 56 Bit. Sie wurde von Horst Feis-

tel von IBM entwickelt.

Der Data Encryption Standard (DES) wurde 1977 zu einem öffentlichen Standard, nachdem er einen vom US National Bureau of Standards (heute National Institute of Standards and Technology, NIST) ausgeschriebenen öffentlichen Wettbewerb gewonnen hatte. Der DES umfasst 16 Iterationen, bei denen Substitution und Permutation (Transposition) durchgeführt werden. Die Blockgröße beträgt 64 Bit, und der Schlüssel besteht aus 64 Bit. Allerdings können nur 56 Bit vom Benutzer ausgewählt werden und bilden den Schlüssel: die restlichen acht sind redundante Paritätsprüfbits.

Wie der Name seines Erfinders vermuten lässt, handelt es sich um eine Feistelchiffre oder ein Substitutions- und Permutationsnetzwerk, ähnlich dem oben beschriebenen Prototyp. Der Algorithmus gruppiert den Text (eine Bytefolge) in 32-Bit-Blöcke mit Teilblöcken von vier Bit und verschlüsselt jeden Block in 16 Iterationen der folgenden drei Schritte, die als Feistelfunktion oder F-Funktion bezeichnet werden, in der angegebenen Reihenfolge

1. Addieren (XOR) des Schlüssels,
2. Substituieren jedes 4-Bit-Teilblocks des Blocks durch die S-Box (in hexadezimaler Schreibweise), und
3. Permutieren aller Teilblöcke.



Symmetrische Kryptosysteme

In jeder Runde i wird die Ausgabe der vorangegangenen Runde in die 32 am weitesten links liegenden Bits L(i) und die 32 am weitesten rechts liegenden Bits R(i) aufgeteilt. R(i) wird zu L(i + 1), während R(i + 1) die Ausgabe einer komplexen Funktion, L(i) + f(R(i)*,* K(i + 1)) ist, deren Eingabe der i + 1-te Block der Schlüsselbits, K(i + 1) und die gesamte vorangegangene Zwischenchiffre ist. Dieser Vorgang wird 16 Mal wiederholt.

Entscheidend für die Sicherheit von DES ist die nichtlineare S-Box der vom NIST spezifizierten F-Funktion f. Sie ist sowohl nichtlinear, d. h. f(A) + f(B) ≠ f(A + B), als auch eine Maximierung der Konfusion und Diffusion im Sinne der Shannon-Kriterien für eine sichere Chiffre.

###### Schlüsselgröße und die Geburt von 3DES

Die Sicherheit des DES ist wie bei jedem Algorithmus nicht größer als der Aufwand für die Suche nach 256 Schlüsseln. Was 1977 noch als undurchführbare Aufgabe für einen Computer galt, schaffte eine spezielle DES-Suchmaschine zwei Jahrzehnte später in drei Tagen. Ein Workaround, Triple DES oder 3DES genannt, wurde entwickelt, das dem DES mit einem 112-Bit-Schlüssel die doppelte Schlüsselgröße gegenüber den DES-Schlüsseln verleiht, indem einfach zwei normale DES-Schlüssel verwendet werden.

Die Verschlüsselung sieht dann so aus: V(1) (E(2) (V(1); (1) verschlüsseln mit dem ersten Schlüssel, (2) entschlüsseln mit dem zweiten Schlüssel und (3) verschlüsseln mit dem ersten Schlüssel.

Die Verschlüsselung sieht dann so aus: V(1) (E(2) (V(1) – (1) verschlüsseln mit dem ersten Schlüssel, (2) entschlüsseln mit dem zweiten Schlüssel und (3) verschlüsseln mit dem ersten Schlüssel.

Wenn die beiden Schlüssel übereinstimmen, wird dieses Kryptosystem zu einem gewöhnlichen DES mit nur einem Schlüssel. Somit ist Triple-DES rückwärtskompatibel mit Anlagen, die für (Single-)DES implementiert wurden. DES war der erste kryptografische Algorithmus, der Kerckhoffs‘ Prinzip erfüllte.

### Advanced Encryption Standard (AES)

Im Januar 1997 kündigte das NIST einen öffentlichen Wettbewerb für einen sogenannten Advanced Encryption Standard (AES) an, der den früheren symmetrischen Verschlüsselungsstandard, den Data Encryption Standard (DES), ersetzen sollte. Nachdem der 56-Bit-Schlüssel des DES mit einer Brute-Force-Attacke geknackt wurde, verlangten die NIST-Spezifikationen für den AES eine größere Schlüssellänge. Der AES-Algorithmus war der Gewinner dieses Wettbewerbs (Bauer, 2016).

###### Beurteilung der Sicherheit

Die Erfinder von AES konnten nachweisen, dass sich diese beiden Verfahren so gut ergänzen, dass sie nach mehreren Iterationen das fehlende Ersetzen des gesamten Blocks durch einen anderen nahezu ausgleichen (Daemen & Rijmen, 1999; 2002). Wie es schon bei

3DES

Dieser Begriff bezeichnet die dreifache Anwendung von DES mit der doppelten Schlüsselgröße des DES-Algorithmus.

Advanced Encryption Standard

Dieses Substitutions- und Permutationsnetzwerk mit einer (variablen) Schlüssellänge von in der Regel 128 Bit wurde im Jahr 2000 als Nachfolger von DES zu einem kryptografischen Standard.

DES der Fall war, so ist auch AES Jahrzehnte nach seiner Einführung noch immer gegen alle Angriffe der Kryptoanalyse gewappnet, aber es ist nicht zu erwarten, dass es den Entwicklungen in der Computertechnik nachgibt, wie es bei DES der Fall war, da die Schlüsselgröße anpassbar ist.

###### Anwendungsfälle

Rijndael zeichnete sich durch seine Einfachheit und seine besonders rechenökonomische Implementierung aus. Es war nicht nur ein sicheres Verschlüsselungsverfahren, sondern dank seines eleganten und einfachen Designs auch klein genug, um auf Chipkarten implementiert zu werden (weniger als 10.000 Byte Code). Bis heute gilt er als der sicherste Algorithmus. Es besteht kein Bedarf an einem weiteren symmetrischen kryptografischen Standardalgorithmus. Er wird folgerichtig überall eingesetzt. Zur Verschlüsselung eines drahtlosen Netzwerks wird beispielsweise ein Schlüssel verwendet, so dass die Verschlüsselung symmetrisch ist. Die sicherste und daher empfehlenswerteste Option ist die Verschlüsselung mit AES.

###### Algorithmus

AES ist eine Blockchiffre, d. h., sie arbeitet mit Byte-Blöcken, in der Regel mit einer Größe von 16 Byte (AES-128).

|  |  |
| --- | --- |
| SubBytes | Ersetzen jedes Bytes des Blocks gemäß einer Substitutionstabelle (S-Box) mit 28 = 256 Einträgen von 1 Byte. |
| ShiftRows | Zyklisches Verschieben der Einträge der letzten drei Zeilen |
| MixColumns | Hinzufügen der Spalten dazwischen. |

###### Blockweise Verschlüsselung

Der AES-Algorithmus gruppiert den Klartext (und die Schlüssel) in 4×B Byte-Rechtecke, wobei B die Anzahl der Spalten des Blocks ist: 4, 6 oder 8. Üblicherweise, und im Folgenden gilt das auch für uns, ist B = 4, d. h. das Rechteck ist ein Quadrat. Auf hexadezimaler Basis hat ein solches Quadrat zum Beispiel die Form

A1 13 B1 4A A3 AF 04 1E 3D 13 C1 55

B1 92 83 72

Symmetrische Kryptosysteme

###### Der Rijndael-Binärkörper

Um den AES zu verstehen, sind einige mathematische Definitionen von Gruppen und Feldern sowie Grundkenntnisse im Rechnen mit Polynomen nötig. Die nachfolgenden mathematischen Darstellungen erläutern die AES-Funktionen SubBytes, ShiftRows und MixColumns im Detail.

Gruppen und Felder

Eine Gruppe ist eine Menge G mit

* einer Operation • : G × G → G, die das Assoziativgesetz erfüllt,
* einem neutralen Element e, sodass gilt x • e = x = e • x für alle x in G), und
* einem inversen Element y für jedes Element x, sodass gilt x • y = e = y • x.

Allgemein wird die Operation mit •, das neutrale Element mit 1und das inverse Element von x

in G mit x-1 bezeichnet.

Beispiel

Die Menge der rationalen Zahlen ungleich Null \*\* mit der Multiplikationsoperation • ist eine Gruppe. Ist die Gruppe G kommutativ (die Operation erfüllt das Kommutativgesetz), so wird die Operation mit +, das neutrale Element mit Null und das inverse Element von x in G mit –x bezeichnet.

Beispiel

Die Menge der rationalen Zahlen \* mit der Additionsoperation + ist eine kommutative Gruppe. Ein „Feld“ ist eine Menge F mit einer Additions- und Multiplikationsoperation + und • so dass gilt:

* die Menge F mit + ist eine kommutative Gruppe,
* die Menge F\* = F - { 0 } mit • ist eine kommutative Gruppe, und
* das Distributivgesetz ist erfüllt.

Beispiel

Die Menge der rationalen Zahlen \* mit Addition + und Multiplikation • ist ein Feld.

Bytes als Polynome mit binären Koefﬁzienten vom Grad 7

Ein Byte, eine Folge b7 ... b1, b0 von acht Bits aus {0, 1} wird als Polynom mit binären Koeffizienten betrachtet durch

b7 . . . b1b0 b7X7 + ! + b1X+ b0

So entspricht beispielsweise die Hexadezimalzahl 0 x 57 oder die Binärzahl 0101 0111 dem Polynom x6 + x4 + x2 + x + 1.

Alle Additionen und Multiplikationen im AES finden im binären Feld IF28 mit 28 = 256 Elementen statt, einer Zahlenmenge mit Addition und Multiplikation, die dem Assoziativ-, Kommutativ- und Distributivgesetz (z. B. \*) entspricht, und wie folgt definiert ist. Sei

IF2 = 0,1

das „Feld zweier Elemente“ mit der Addition 1 + 0 = 0 + 1 = 1 und 0 + 0 = 1 + 1 = 0 (entspricht der ⊕-Addition durch XOR), und der (natürlichen) Multiplikation 1 • 0 = 0

• 1 = 0 • 0 = 0 und 1 • 1 = 1.

Sei

IF2 X = +/2+ = die Polynome über IF2,

d. h. die finiten Summen a0 + a1 X + a2 X2 + ·+ an Xn bis a0, a1, …, an bis IF2 und sei

IF28: = IF2 X / X8 + X4 + X3 + X + 1 .

Das heißt, das Ergebnis der beiden Operationen + und • in IF2[X] ist der Rest der Division durch X8 + X4 + X3 + X + 1.

Addition

Die + -Addition von zwei Polynomen ist die Addition in IF2 Koefﬁzient zu Koeffizient. Das heißt, dass die + -Addition als Bytes durch die XOR-Addition gegeben ist.

Multiplikation

Die Multiplikation • ist gegeben durch die natürliche Multiplikation, gefolgt von der Division mit Rest durch das Polynom

m x = x8 + x4 + x3 + x + 1 .

In hexadezimaler Notation ist beispielsweise 57• 83 = C1 , da

x6 + x4 + x2 + x + 1 x7 + x + 1 = x13 + x11 + x9 + x8 + x6 + x5 + x4 + x3 + 1

und

x13 + x11 + x9 + x8 + x6 + x5 + x4 + x3 + 1 = x5 + x3 + 1 x8 + x4 + x3 + x + 1 + x7

+ x6 + 1

Die Multiplikation mit dem Polynom 1 verändert nichts: Es ist das neutrale Element. Für ein beliebiges Polynom b(x) berechnet Euklids erweiterter Algorithmus die Polynome a(x) und c(x) , so dass

Symmetrische Kryptosysteme

b x a x + m x c x = 1 .

Das heißt, bei der Division mit dem verbleibenden a(x) b(x) für m(x) Rest 1. Dies bedeutet, dass

a der Kehrwert in IF 8 ist,

2

b−1 x = a x in IF28 .

Wenn wir ein Byte b in IF 8 invertieren, meinen wir Byte a = b–1.

2

###### Runden

Der AES verschlüsselt jeden Block iterativ, in Runden. R sei die Anzahl der Runden, die von B abhängt. Es gibt R = 10 Runden für B = 4 Spalten, R = 12 Runden für B

= 6 Spalten und R = 14 Runden B = 8 Spalten.

Für uns gilt R = 10. In diesen Runden werden Schlüssel erzeugt und der Klartext wird durch folgende Operationen ersetzt und transponiert:

|  |  |  |
| --- | --- | --- |
| 1. | Runde r = 0 | AddRoundKey zum Hinzufügen (durch XOR) des Schlüssels zum Klartextblock |
| 2. | Runde r = 1,  …, R – 1 | Zum Verschlüsseln sind folgende Funktionen anzuwenden   1. SubBytes zum Ersetzen jedes Bytes (= Folge von acht Bits) durch eine besser verteilte Folge von Bits 2. ShiftRows zum Permutieren der Einträge in jeder Zeile des Quadrats 3. MixColumn zum Hinzufügen der Spaltenboxen im Quadrat zwischen den Spalten 4. AddRoundKey zur Erzeugung eines Schlüssels aus dem Schlüssel der vorherigen Runde und dessen Hinzufügen (durch XOR) zum Quadrat |
| 3. | Runde r = R | Zum Verschlüsseln sind folgende Funktionen anzuwenden   1. SubBytes 2. ShiftRows 3. AddRoundKey |

Im Vergleich zu früheren Runden entfällt die Funktion MixColumn. Es zeigt sich, dass MixColumn und AddRoundKey nach einer geringfügigen Änderung von AddRoundkey die Reihenfolge ändern können, ohne das Ergebnis der beiden Operationen zu verändern. In dieser äquivalenten Reihenfolge erhöht die Operation MixColumn nicht die kryptografische Sicherheit, da die letzte Operation ohne Kenntnis des Schlüssels invertierbar ist. Somit kann die letzte MixColumn-Operation entfallen.

Das CrypTool 1 bietet in seinem Menü „Individuelle Verfahren > Visualisierung von Algorithmen > AES“ einen Eintrag „Animation“ der Runden und einen Eintrag „Inspektor“ zum Experimentieren mit den Werten von Klartext und Schlüssel. Die verwendeten Funktionen werden im Folgenden näher beschrieben.

SubBytes

SubBytes ersetzt jedes Byte des Blocks durch ein anderes Byte, das in der Substitutionstabelle der S-Box angegeben ist.

Berechnung des Werts des Eintrags, durch den die S-Box die einzelnen Bytes ersetzt:

* + 1. Berechnung des Kehrwerts B in IF 8,

2

* + 1. Berechnung von ai = b1 + bi+4 + bi+5 + bi+6 + bi+7 + ci

mit i = 0, 1, …, 7 als Index jedes Bits eines Bytes, und

* + B = b7 b6 b5 b4 b3 b2 b1 b0 ist das Eingangsbyte,
  + A = a7 a6 a5 a4 a3 a2 a1 a0 ist das Ausgangsbyte der Operation und
  + c ist das konstante Byte 01100011.

In Matrixform:

a0 b0

1 0 0 0 1 1 1 1 1

a1 1 1 0 0 0 1 1 1 b1 1

a2 1 1 1 0 0 0 1 1 b2 0

a3 1 1 1 1 0 0 0 1 b3 0

= +

a4 1 1 1 1 1 0 0 0 b4 0

a5 0 1 1 1 1 1 0 0 b5 1

a6 0 0 1 1 1 1 1 0 b6 1

0 0 0 1 1 1 1 1 0

b

a7 7

ShiftRows

ShiftRows verschiebt die Zeilennummer l des Quadrats l Positionen nach links (wobei die Zeilennummern von Null an gezählt werden, d. h. l durchläuft 0, 1, 2 und 3, und die Verschiebung erfolgt zyklisch). *Anmerkung:* Die erste Zeile wird nicht verschoben.

MixColumn

MixColumn multipliziert jede Spalte des Blocks mit einer festen Matrix:

Symmetrische Kryptosysteme

* wenn Bj (mit den Koeffizienten b0,j, b1,j, b2,j und b3,j) der Spalte j des Eingangsblocks entspricht, und
* wenn Aj (mit den Koeffizienten a0,j, a1,j, a2,j und a3,j) der Spalte j des Ausgangsblocks der Operation entspricht,

dann

a0,j

a1, j

=

a2, j

a3, j

02 03 01 01

01 02 03 01

01 01 02 03

03 01 01 02

b0,j

b1, j

.

b2, j

b3, j

Das Byte a0,j wird zum Beispiel wie folgt berechnet:

a0,j = 2 · b0, j + 3 · b1, j + b2, j + b3, j .

AddRoundKey

AddRoundKey fügt durch die XOR-Operation den Schlüssel W(r) der aktuellen Runde r zum B-Quadrat des Geheimtextes hinzu, das heißt

B ⊕ W r .

Der Schlüssel wird spaltenweise erzeugt. Wir bezeichnen sie mit W(r)0, W(r)1, W(r)2 und

W(r)3, d. h.

W r = W r 0 ∣ W r 1 ∣ W r 2 ∣ W r 3 .

Da der Schlüssel 16 Byte hat, hat jede Spalte vier Byte. Das heißt, die letzte Spalte des vorherigen Rundenschlüssels plus das Ergebnis von ScheduleCore wird auf die erste Spalte des vorherigen Rundenschlüssels (den wir mit T bezeichnen) angewendet. Hier ist ScheduleCore die Zusammensetzung von Transformationen:

* SubWord ersetzt jedes der 4 Byte von T entsprechend der S-Box von SubBytes.
* RotWord verschiebt T ein Byte nach links (zirkulär, d.h. das letzte Byte wird das erste).
* Rcon(r) addiert (durch XOR) zu T den konstanten Wert, in hexadezimaler Schreibweise [(02)r–1 00 00 00] (wobei die Potenz, d. h. das iterierte Produkt, im Rijndael-Feld IF 8 berechnet wird). Das einzige Byte, das sich ändert, ist das Erste, indem entweder der Wert 2r–1 (für r ≤ 8) oder der Wert 2r–1 in F 8 für r = 9, 10 addiert wird.

2

2

1. Der Schlüssel der ersten Runde W(0) ist durch den Anfangsschlüssel W gegeben.
2. Für r = 1, ..., R (wobei R die Gesamtzahl der Runden ist, für uns R = 10), werden die vier Spalten W(r)0, W(r)1, W(r)2 und W(r)3 des neuen Schlüssels wie folgt aus den Spalten des alten Schlüssels W(r-1) erzeugt:
   1. Die erste Spalte W(r)0 ist gegeben durch

W r 0 = W r − 1 0 ⊕ ScheduleCore W r − 1 3 ;

* 1. Die nächsten Spalten W(r)1, W(r)2 und W(r)3 sind für i = 1, 2 und 3 gegeben durch

W r i = W r i − 1 ⊕ W r − 1 i;

* 1. Das heißt, die vorherige Spalte des aktuellen Rundenschlüssels plus die aktuelle Spalte des vorherigen Rundenschlüssels.

###### Diffusion

Wir stellen fest, dass die einzige Transformation, die nicht afﬁn ist, die Inversion im Feld IF 8 in der Operation SubBytes ist. Bei der Operation SubBytes werden angewandt: die Inversion in IF 8, eine lineare Applikation, und die Übersetzung durch einen konstanten Vektor (a) ShiftRows ist eine lineare Permutation, (b) MixColumn ist eine lineare Addition, und (c) AddRoundKey ist die Translation durch den Rundenschlüssel.

2

2

Im Hinblick auf die Ziele der idealen „Diffusion“ und „Konfusion“ ist zu beachten, dass in jedem Schritt etwa die Hälfte der Bits (in SubBytes) oder Bytes (in MixColumn und ShiftRows) ersetzt und transponiert werden. Die Komplementarität der einfachen Operationen für hohe Sicherheit, d.h. ihre Erzeugung von hoher Konfusion und Diffusion nach einigen Iterationen, wird durch die Substitution des Alphabets und die Permutation von Text deutlich. Bei letzteren ist dies insbesondere bei der Permutation zwischen den Einträgen jeder Zeile und der Permutation zwischen den Spalten zu beobachten. Das Cryptool von Forma Estudio (2007) ist ein Beispiel für diese Eigenschaft.

Wir sehen, wie sich dieser kleine anfängliche Unterschied ausbreitet und nach nur vier Runden zu sehr unterschiedlichen Ergebnissen führt! Dies macht die Immunität von AES gegen differenzielle Kryptoanalyse plausibel.

Für den Fall, dass alle Schlüssel- und Klartexteinträge gleich 00 sind, verstehen wir auch die Auswirkung des Addierens der Konstante Rcon(r) zu dem Schlüssel in jeder Runde. Daher kommt die gesamte Konfusion!

### Kryptografische Hashfunktionen

Eine Hashfunktion wandelt beliebige Daten (z. B. eine Datei), d. h. eine Zeichenkette mit variabler Länge, in eine Zeichenkette mit fester Länge (in der Regel 16 oder 32 Byte) um. Sie wandelt eine große Datenmenge in eine kleine Informationsmenge um.

Symmetrische Kryptosysteme

###### Hash als ID

Eine Hashfunktion sollte auf alle möglichen Sequenzen mit fester Länge anwendbar sein und einer gleichförmigen Zufallsvariablen ähneln, d. h. die Wahrscheinlichkeit für jeden Funktionswert ist gleich groß.

Wenn die Ausgabe beispielsweise aus 256 Bits besteht, dann sollte im Idealfall jeder Wert die gleiche Wahrscheinlichkeit 2–256 haben. Das heißt, die Ausgabe identiﬁziert die Eingabe praktisch eindeutig (mit einer Kollisionswahrscheinlichkeit von idealerweise 2–256). So könnte man sich beispielsweise einen Datenhash von einer Datei als ihren Personalausweis vorstellen. Ein Hash identiﬁziert viele Daten mit wenigen Informationen.

Da die Länge der Hash-Sequenz begrenzt ist (selten ≥ 512 Bits) und die Länge der Eingabesequenz unbegrenzt ist, kommt es zu Kollisionen, d. h. zu gleichen Hashes aus verschiedenen Dateien. Der Algorithmus minimiert jedoch die Wahrscheinlichkeit von Kollisionen, indem er die Werte so gleichmäßig wie möglich verteilt – oder intuitiv gesagt: so zufällig wie möglich macht. Genauer gesagt ist jede mögliche Sequenz mit fester Länge ein Wert, und die Wahrscheinlichkeit für jeden der Werte ist die gleiche.

###### Kryptografische Hashfunktionen

Für eine kryptografische oder Einweg-Hashfunktion muss der Algorithmus resistent sein gegenüber

1. der Erstellung eines Urbildes, d.h. die Funktion ist unidirektional. Wenn eine Ausgabe gegeben ist, ist der schnellste Weg, eine Eingabe mit dieser Ausgabe zu finden, ein Brute-Force-Verfahren, d. h. die Prüfung aller möglichen Eingaben.
2. der Erstellung einer zweiten umgekehrten Abbildung, d. h. bei einer Eingabe ist der schnellste Weg, eine andere Eingabe mit demselben Ergebnis zu finden, ein Brute-Force-Verfahren.
3. der Erstellung von Kollisionen, d. h. der schnellste Weg, zwei (beliebige) Einträge mit demselben Ergebnis zu finden, ist ein Brute-Force-Verfahren.

Nach Kerckhoffs' Prinzip muss der Algorithmus zudem öffentlich sein. In der Praxis ist die wichtigste Eigenschaft die Resistenz gegen den Angriff über die Erstellung einer zweiten, umgekehrten Abbildung, und die am wenigsten wichtige ist die gegen Kollisionsangriffe. Es gibt verschiedene Algorithmen (z. B. MD4, MD5 und SHA-1), die nicht gegen Kollisionsangriffe resistent sind, aber dennoch verwendet werden. Der CRC-Algorithmus ist zum Beispiel eine (nicht kryptografische) Hashfunktion. Andere gängige kryptografische Hashfunktionen sind SHA-256 und SHA-3.

Die Ausgabe der Hashfunktion SHA-256 von IUBH lautet zum Beispiel 0a65cfb2c33b58749182d3eb965155336a52219793ff633016cb3942b2809a3f. Die Ausgabe einer Hashfunktion, aber auch Message Digest oder digitaler Fingerabdruck, je nach Eingabe, werden zum Beispiel für die Integrität und Authentifizierung von Nachrichten verwendet.

###### Gängige kryptografische Hashfunktionen und Anwendungen

Die am häufigsten verwendeten (kryptografischen) Hash-Algorithmen sind auch heute noch 16 Byte groß, wie MD4 oder MD5, oder SHA-1, das 20 Byte verwendet. Obwohl keine von ihnen Kollisionsangriffen standhält, sind sie nach wie vor beliebt. Die Einzelheiten ihrer Implementierung sind im Request for Comments (RFC) beschrieben. Ein RFC spezifiziert öffentlich die Einzelheiten eines geplanten Internetstandards oder einer neuen Version eines bestehenden Standards und wird in der Regel von Forschern an Universitäten und in Unternehmen verfasst, um Feedback einzuholen. Ein RFC wird online und in offiziellen Sitzungen der von der Internet Engineering Task Force (IETF) beauftragten Arbeitsgruppe diskutiert. So wurden beispielsweise Netzwerkstandards wie IP und Ethernet in RFCs dokumentiert.

Der Message-Digest-Algorithmus (MD4) wurde 1991 von Ron Rivest entwickelt, einem der drei Erfinder des RSA-Algorithmus. Er wird in RFC 1320 beschrieben, ist schnell, aber anfällig für Urbild-Angriffe.

MD5 wurde von RSA Data Security entwickelt. Das in RFC 132 beschriebene Verfahren ist anfällig für Kollisionen, aber nicht für die Erstellung eines zweiten Urbildes. Es wird häufig für Integritätsprüfungen, von Software mit Peer-to-Peer-Protokoll (P2P) und als Passwortspeicher verwendet.

Der Secure-Hash-Algorithmus (SHA-1) wurde vom NIST entwickelt. Er ist anfällig für Kollisionen, aber nicht für die Erstellung eines zweiten Urbildes. Die neueren Versionen, wie SHA-256 und SHA-3, sind gegenüber den alten SHA-1 oder MD4 und MD5 vorzuziehen.

Digitale Signaturen

Sind die Rollen des öffentlichen und des privaten Schlüssels vertauscht, so handelt es sich bei der Verschlüsselung um eine digitale Signatur. Während die verschlüsselte Nachricht nicht mehr geheim ist, kann jeder Besitzer des öffentlichen Schlüssels überprüfen, ob der private Schlüssel die ursprüngliche Nachricht verschlüsselt hat. Theoretisch würde jedoch das Signieren einer Datei (unter Verwendung des RSA-Algorithmus), die mit dem RSA-Algorithmus verschlüsselt wurde, diese entschlüsseln. In der Praxis wird ein kryptografischer Hash in der Regel mit dem privaten Schlüssel verschlüsselt, so wie eine Signatur ausreicht, um die signierte Datei eindeutig zu identifizieren (ihr Inhalt ist jedoch oft geheim, z. B. beim Signieren eines geheimen Schlüssels).

(H)MAC

Zur Nachrichtenauthentifizierung wird ein Message Authentication Code (MAC) verwendet. Wenn dieser MAC durch eine Hashfunktion gesichert ist, wird er als Keyed-Hash Message Authentication Code (HMAC) bezeichnet. Das American National Standards Institute (ANSI) hat einen Message Digest, einen Hash einer Nachricht, mit einem Passwort als Hash-Schlüssel unter Verwendung des DES-Algorithmus als neuen Stand der Kryptografie entwickelt. Ein HMAC wird also aus einer Nachricht und einem geheimen Schlüssel erzeugt, der in einer kryptografischen Hashfunktion wie SHA-1 oder MD5 verwendet wird. Der MAC-Wert schützt die Integrität und Authentizität einer Nachricht, indem er es den vorgesehenen Empfängern ermöglicht, Änderungen am Inhalt der Nachricht zu erkennen. Der HMAC liefert dem Benutzer außerdem sowohl eine Integritäts- als auch eine Authentizitätsprüfung (Krawczyk et al., 1997).

Symmetrische Kryptosysteme

Datenspeicherung und -integrität

Hashfunktionen (sowohl kryptografische als auch nicht kryptografische) werden für die schnelle Datenabfrage, d. h. mit konstanter Dauer, in einer Hashtabelle und in einem Hash-Baum (auch Merkle-Baum) verwendet. Sie werden außerdem dazu eingesetzt, die Integrität einer Datei angesichts versehentlicher Änderungen zu gewährleisten, d. h. um Unterschiede zwischen der Datei und einer Referenzversion zu erkennen.

Passwörter

Kryptografische Hashfunktionen werden verwendet, um Werte gleichmäßig zu verteilen (Schlüsselstreckung, engl. Key Stretching), was sie intuitiv weniger vorhersehbar macht, bekannt als Key Derivation Function (KDF), und um Passwörter zu erzeugen und zu speichern, als PBKDF (Password Based Key Derivation Function). Kryptografische Hashfunktionen werden auch verwendet, um die Integrität einer Datei vor Manipulationen zu schützen, d. h. um Unterschiede zwischen der Datei und einer Referenzversion (in der Regel die Version, bevor die Datei übertragen wird) zu erkennen. Insbesondere, um die Echtheit einer Datei zu gewährleisten und Unterschiede zwischen der Datei und einer Version zu erkennen, die unter der Kontrolle einer bestimmten Person stand.

Beachten Sie den Unterschied zwischen Authentizität und Authentifizierung. Erstere garantiert die Gleichheit der von einer Person empfangenen und gesendeten Daten, z. B. bei der digitalen Signatur, Letztere die Identität dieser Person, z. B. bei einem sicheren Website-Zugang.

Die oben aufgeführten kryptografischen Hash-Algorithmen verteilen die Werte gleichmäßig, sind aber schnell und eignen sich daher nicht für die Erstellung von Passwörtern, da sie anfällig für Brute-Force-Angriffe sind. Um diese zu verhindern, sind die PBKDF-Algorithmen (a) absichtlich langsam, wie z. B. bcrypt, (b) erfordern absichtlich viel Speicher für die Berechnung, wie z. B. der scrypt-Algorithmus, oder (c) werden nur einmal für jeden Eintrag verwendet, garantiert durch ein „Salt“, ein zusätzliches, einzigartiges, normalerweise zufälliges Argument. Ohne Salt ist der Algorithmus anfällig für Angriffe durch „Rainbow Tables“, Tabellen mit den Hashes der gängigsten Passwörter.

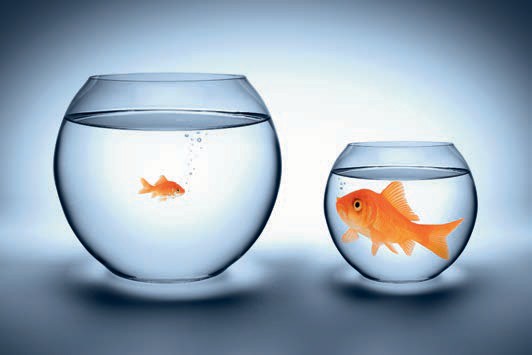
Zusammenfassung

Beim Chiffrieren gibt es zwei grundlegende Verfahren: Transposition und Substitution. Transpositionen ordnen den Text neu an, ohne die Zeichen zu verändern; Substitutionen ersetzen die Zeichen des Textes, ohne die Reihenfolge zu verändern. Feistelchiffren sind unsicher, weil sie die statistischen Daten des Klartextes erhalten. Bei modernen Chiffren werden im Gegensatz dazu Substitutions- und Permutations-Chiffren kombiniert, die als Substitutions- und Permutationsnetzwerke bezeichnet werden.

Es gibt zwar keine Methode der Kryptoanalyse, die schneller ist als eine erschöpfende Schlüsselsuche, aber moderne Chiffren wie DES (1976) oder AES (2000) erreichen in der Praxis dasselbe. Mit anderen Worten: Es ist keine kryptoanalytische Methode bekannt, die schneller als die erschöpfende Schlüsselsuche ist. Das entscheidende Kriterium ist eine hohe Diffusion im Sinne von Shannon.

Hashfunktionen wandeln beliebige Daten mit variabler Länge in eine Ausgabe mit fester Länge um. Der ausgegebene Hashwert sollte eine gute Diffusion aufweisen: die Umkehrung eines Eingabebits führt zu Änderungen bei etwa der Hälfte der Ausgabe; er sollte für alle möglichen Sequenzen mit fester Länge anwendbar sein und einer gleichförmigen Zufallsvariablen ähneln, d. h. die Wahrscheinlichkeit für jeden

der Funktionswerte ist die gleiche. Bei einer Ausgabe mit fester Länge von 256 Bits sollte im Idealfall jeder Wert die gleiche Wahrscheinlichkeit 2-256 haben. Es handelt sich um eine kryptografische (oder Einweg-) Hashfunktion, wenn eine Umkehrung rechnerisch undurchführbar ist und wenn ähnliche Daten unterschiedliche Hashwerte ergeben. Empfohlen werden die neueren Versionen SHA-256 und SHA-3.



# Lektion 3

## Asymmetrische Kryptosysteme

#### LERNZIELE

Nach der Bearbeitung dieser Lektion kennen Sie …

… die Vorteile und Grenzen der asymmetrischen Kryptografie.

… Falltürfunktionen und modulare Arithmetik.

… die gängigsten asymmetrischen kryptografischen Algorithmen und die ihnen zugrunde liegenden Falltürfunktionen.

DL-E-DLMCSEAITSC02-U03

1. Asymmetrische Kryptosysteme

### Einführung

Das große praktische Problem der Kryptografie mit nur einem Schlüssel ist die Schlüsselverteilung, d.h. die geheime Weitergabe desselben Schlüssels an alle Kommunikationsteilnehmer, die normalerweise weit voneinander entfernt sind, bevor sie sicher kommunizieren können. Problematisch ist zudem die große Anzahl von Schlüsseln, die eine Gruppe von Kommunikationsteilnehmern für eine sichere Kommunikation benötigt.

1976 fanden Difﬁe und Hellman heraus, dass das Problem der Schlüsselverteilung durch einen Algorithmus gelöst werden kann, der (rechnerisch) folgende Bedingungen erfüllt:

* + einfache Erstellung eines passenden Schlüsselpaares für die Ver- und Entschlüsselung,
  + einfache Verschlüsselung und Entschlüsselung,
  + unmögliche Wiederherstellung eines der Schlüssel trotz Kenntnis des Algorithmus, des anderen Schlüssels und einer beliebigen Anzahl übereinstimmender Klartext- und Geheimtext-Paare, und
  + unmögliche Wiederherstellung des Klartextes für nahezu alle Schlüssel k und Nachrichten x.

Als Benutzerin dieses Algorithmus hält Alice ihren Entschlüsselungsschlüssel geheim, macht aber ihren Verschlüsselungsschlüssel öffentlich, zum Beispiel in einem öffentlichen Verzeichnis. Jeder, der sicher mit Alice kommunizieren möchte, braucht nur ihren öffentlichen Schlüssel nachzuschlagen, um ihr einen Geheimtext zu senden, den nur sie entschlüsseln kann, d. h. eine Nachricht kann verschlüsselt werden, ohne dass eine Geheimhaltung erforderlich ist. Jeder, der den entsprechenden öffentlichen Schlüssel verwendet, kann einen mit dem geheimen Schlüssel von Alice verschlüsselten Geheimtext entschlüsseln, d. h. ein Absender kann identifiziert werden, ohne dass eine Geheimhaltung erforderlich ist.

Die Sicherheit kryptografischer Algorithmen mit zwei Schlüsseln hängt von der Schwierigkeit der Berechnung eines mathematischen Problems ab, z. B. der Faktorisierung einer Zahl, die das Produkt aus zwei großen Primzahlen ist. Im Idealfall ist die Berechnung des geheimen Schlüssels gleichbedeutend mit der Lösung des schwierigen Problems, so dass der Algorithmus mindestens so sicher ist, wie das zugrunde liegende mathematische Problem schwierig ist. Dies wurde für keinen der Standardalgorithmen bewiesen, obgleich angenommen wird, dass es für jeden von ihnen gilt.

###### Arithmetische Falltüren

Um die Nützlichkeit der (modularen) Arithmetik in der Kryptografie zu erkennen, erinnern wir uns daran, dass die asymmetrische Kryptografie auf einer Falltürfunktion basiert, die leicht berechenbar sein muss, deren Umkehrung aber ohne Kenntnis eines geheimen Schlüssels praktisch unberechenbar sein muss.

Die Einfachheit der Berechnung der Funktion entspricht der Einfachheit der Verschlüsselung, während die Schwierigkeit der Berechnung der Umkehrung der Schwierigkeit der Entschlüsselung, d.h. der Umkehrung der Verschlüsselung, entspricht. RSA verwendet zum Beispiel als Verschlüsselungsfunktion die Erhöhung auf eine n-te Potenz und als Entschlüsselungsfunktion deren Umkehrung, das Wurzelziehen.

Während sowohl die Funktion selbst als auch ihre Umkehrung leicht mit der üblichen Multiplikation von Zahlen berechnet werden können, verwenden kryptografische Algorithmen (wie RSA) die modulare Arithmetik, um die Berechnung der Umkehrfunktion ohne Kenntnis des Schlüssels komplexer zu gestalten.

Asymmetrische Kryptosysteme

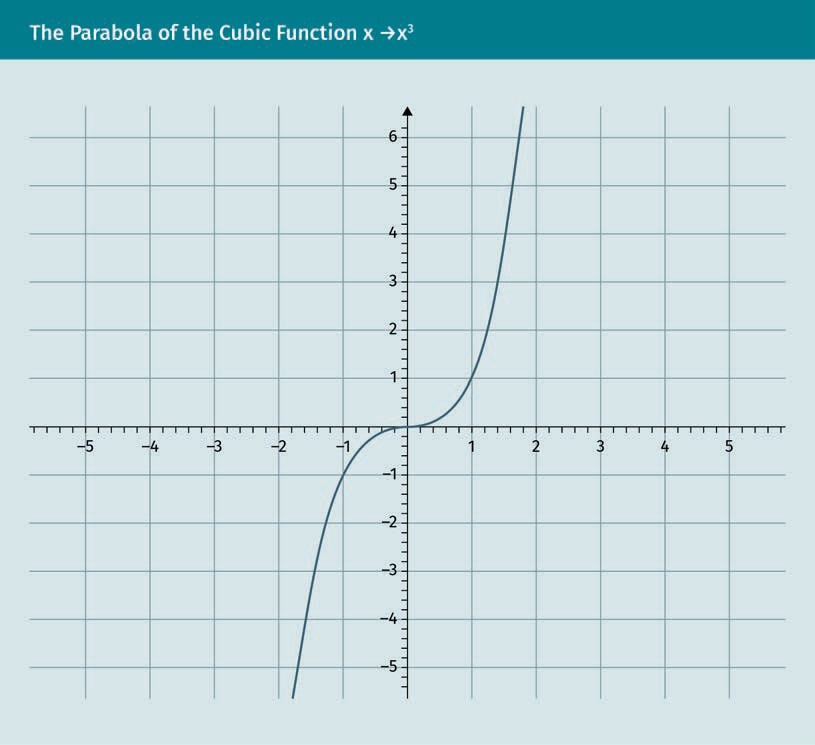
Die modulare oder zirkuläre Arithmetik kennen wir bereits aus der Arithmetik der Uhr, bei der m = 12 als gleich 0 angesehen wird. Da der Zeiger nach jeder Umdrehung wieder bei 0 beginnt zu zählen, ist es zum Beispiel drei Stunden nach 11 Uhr zwei Uhr, 11 + 3 = 14 = 12 + 2 = 2. In diesen Definitionsbereichen, die man als infinite Ringe bezeichnet, werden (die Graphen der) Funktionen unregelmäßig und praktisch unberechenbar, zumindest ohne Kenntnis einer Abkürzung (des Schlüssels).

Modulare Arithmetik als Randomisierung

Die Schwierigkeit, die Umkehrung zu berechnen, entspricht der Schwierigkeit der Entschlüsselung, d. h. der Umkehrung der Verschlüsselung. Bei einem asymmetrischen kryptografischen Algorithmus basieren die Einfachheit der Verschlüsselung einer Zahl und die Schwierigkeit der Entschlüsselung einer Zahl auf einer invertierbaren Funktion, so dass diese zwar leicht berechenbar ist, aber sich ihre Umkehrfunktion nur schwer berechnen lässt. Zum Beispiel sind die Umkehrungen der Falltürfunktionen Potenzierung x /xe (im RSA-Algorithmus) und Exponentiation x /gx (im Difﬁe-Hellman-Algorithmus) durch das Wurzelziehen x /x1/e und den Logarithmus log*g* gegeben. Wir können sie im Definitionsbereich der reellen Zahlen 0 leicht berechnen (z.B. mit der Bisektionsmethode für stetige Funktionen dank der Verknüpfung von 0), aber in ihren kryptografischen Definitionsbereichen sind sie nahezu unberechenbar. Wir wollen diese finiten Definitionsbereiche nun einführen.

Funktionen auf diskreten Mengen

Ihr Definitionsbereich ist nicht die Menge der ganzen Zahlen + (oder die der reellen Zahlen 0 , die sie einschließt), weil beide Funktionen, Exponentiation und Potenzierung, stetig über 0 sind.



Wenn der Definitionsbereich dieser Funktionen 0 wäre, könnten ihre Umkehrungen über 0 angenähert werden, zum Beispiel durch iterative Bisektion, bei welcher der Umkehrpunkt in Intervallen eingeschlossen wird, die bei jeder Iteration halbiert werden.

Endlicher Ring Dies ist eine endliche Menge, die 0 und 1 enthält und über die eine Summe erklärt ist, die dem Assoziativgesetz und dem Kommutativgesetz

gehorcht.

Endliche Ringe

Um die iterative Annäherung an die Null zu vermeiden und damit die Berechnung der Umkehrfunktion zu erschweren (neben der Erleichterung der Berechnung der eigentlichen Funktion), ist der Definitionsbereich einer Falltürfunktion ein endlicher Ring, angegeben durch

+/m+ = 0,1, . . . , m − 1

für eine natürliche Zahl m. In solch einem endlichen Ring gilt

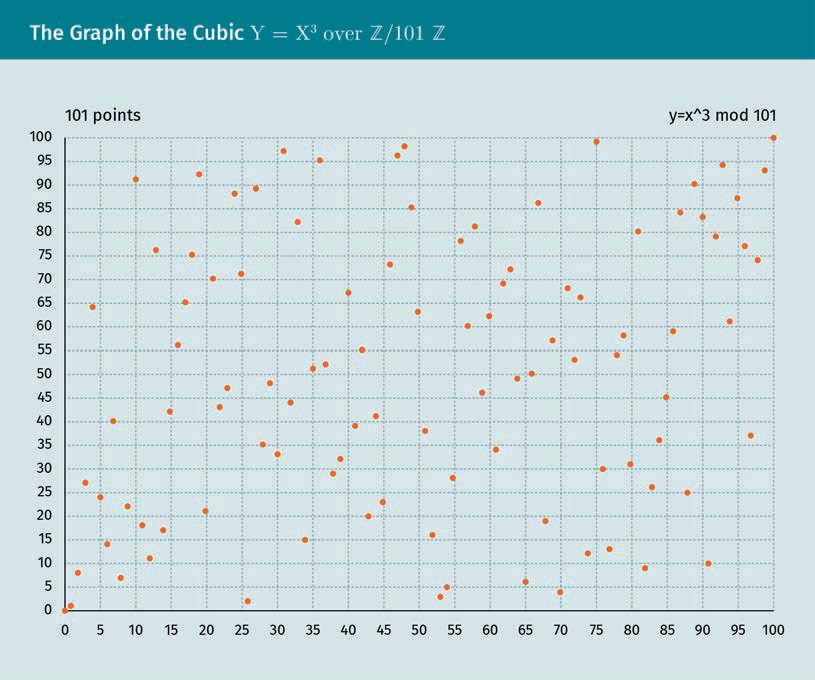
m = 1 + ! + 1 = 0

Asymmetrische Kryptosysteme

und notwendigerweise hat jede Addition (und damit jede Multiplikation und jede Potenzierung) ein Ergebnis < m. Also ist + / m + seine Addition anders als die von + (oder 0). Für m = 7 erhalten wir z. B.

22 = 2 · 2 = 4 und 32 = 3 · 3 = 7 + 2 = 0 + 2 = 2 .

Wir werden diese endlichen Ringe zunächst anhand der Beispiele + / 12 + und + / 7 + (der Ring, der durch die Zahlen der Stunden einer Uhr und der Wochentage gegeben ist) einführen, dann für jedes m.



Wenn wir uns die Graphen der Funktionen ansehen, die auf 0 so regelmäßig sind, stellen wir fest, dass die Parabel über dem endlichen Ring + /101 + zunächst genauso regelmäßig über + / 101 + ist wie über +. Ab x = 11 (wegen 112 = 121 > 100) beginnt sie sich jedoch unregelmäßig zu verhalten (mit Ausnahme der Symmetrie der Parabel auf der Mittelachse x = 50.5 aufgrund der Gleichheit ((–x)2 = x2).

Zwei ganze Zahlen a und b sind kongruent modulo m

a ≡ b mod m

wenn m | a – b, das heißt, wenn ihre Differenz a – b durch m teilbar ist.

Kongruenz

Zwei ganze Zahlen a und b sind kongruent modulo m, wenn sie nach der Division durch m den gleichen Rest übrig lassen.

Die Zahl m ist der Modul. Oder anders formuliert, a ≡ b mod m , wenn a und

b beim Dividieren durch m den gleichen Rest lassen.

### Der Difﬁe-Hellman-Schlüsselaustausch

Difﬁe-Hellman-Schlüssel-

austausch Dieses Protokoll war eine offene Vereinbarung über einen gemeinsamen geheimen Schlüssel, dessen Sicherheit auf der Unmöglichkeit beruhte, den Logarithmus modulo einer großen Zahl zu berechnen.

Das erste veröffentlichte Protokoll, in dem ein gegenseitiger geheimer Schlüssel offen vereinbart wurde, ist der Difﬁe-Hellman-Schlüsselaustausch (Difﬁe & Hellman, 1976). Dies ist noch keine Zwei-Schlüssel-Kryptografie, da beide Korrespondenten den einzigen geheimen Schlüssel kennen. Die asymmetrischen kryptografischen Algorithmen, die auf diesem Protokoll aufbauen (z. B. Elgmal und ECC), erzeugen für jede Nachricht einen eindeutigen Schlüssel.

###### Schlüsselaustauschprotokoll

Bezeichnen wir in jeder asymmetrischen Verschlüsselung eine öffentliche Zahl mit einem Großbuchstaben und eine geheime Zahl mit einem Kleinbuchstaben. Damit sich zwei Parteien, Alice und Bob, offen auf einen geheimen Schlüssel einigen können, kombinieren sie zunächst eine geeignete Primzahl p (der Modul) und eine geeignete natürliche Zahl g (die Basis).

Zunächst generiert Alice eine Hälfte des Schlüssels, indem sie (1) eine Zahl a wählt, (2) A

≡ga mod p berechnet und (3) A an Bob überträgt. Danach generiert Bob die andere Hälfte des Schlüssels, indem er (1) eine Zahl b wählt, (2) B ≡gb mod p berechnet und (3) B an Alice übermittelt. Der gemeinsame geheime Schlüssel zwischen Alice und Bob lautet

b a

b a aab ba b

c ( A mod p = g mod p mod p = g mod p = g mod p = g mod p mod p = B

mod p .

Dieses Protokoll zeigt, wie man einen gemeinsamen geheimen Schlüssel offen erstellt. Dieser Schlüssel kann dann zur Verschlüsselung der gesamten weiteren Kommunikation verwendet werden, so wie AES. Er zeigt jedoch weder, wie man eine Nachricht verschlüsselt, noch wie man sie signiert.

###### Sicherheit

Die Sicherheit des Difﬁe-Hellman-Schlüsselaustauschs beruht auf der Schwierigkeit, den Logarithmus modulo p zu berechnen. Ein Abhörender würde den geheimen Schlüssel Ab = Ba aus A und B erhalten, wenn er folgendes berechnen könnte:

a = loggA oder b = loggB mod p .

Während eine Potenz leicht berechenbar ist, erst recht in der modularen Arithmetik, ist ihre Umkehrung, der Logarithmus, also der Exponent für eine bestimmte Potenz, bei geeigneter Wahl von p und g praktisch nicht berechenbar.

Asymmetrische Kryptosysteme

###### Logarithmus

Da zu Beginn (für x < logg p) die Werte gx über + / p + gleich den Werten gx über + sind, müssen die geheimen Zahlen a und b ausreichend groß sein, d. h. > logg p. Um dies sicherzustellen, werden diese Zahlen in der Praxis künstlich erhöht, das heißt, die Nachricht wird aufgefüllt.

Der derzeit schnellste Algorithmus zur Berechnung des Logarithmus x aus gx ist eine Adaption des allgemeinen Zahlenfeldsiebs (Gordon, 1993). Die Anzahl der Operationen zur Berechnung des Logarithmus einer ganzen Zahl mit n Bits entspricht grob

exp logn1/3 .

### Der RSA-Algorithmus

Der bekannteste Public-Key-Algorithmus ist der Rivest-Shamir-Adleman-Kryptoalgorithmus (kurz (RSA-Algorithmus) (Rivest et al., 1978). Ein Benutzer wählt im Geheimen ein Paar Primzahlen p und q , die so groß sind, dass die Faktorisierung des Produkts N = p q die geschätzte Rechenleistung für die gesamte Nutzungsdauer der Chiffre übersteigt. Die Zahl N wird der Modul sein, d.h. unsere Falltürfunktion basiert auf + / N + = { 0, 1, …, N-1 }.

N ist öffentlich, p und q jedoch nicht. Wenn die Faktoren p und q von N bekannt wären, dann könnte der geheime Schlüssel leicht berechnet werden. Damit RSA sicher ist, darf die Faktorisierung rechnerisch nicht durchführbar sein, was derzeit 2048 Bit entspricht. Der Schwierigkeitsgrad der Faktorisierung verdoppelt sich ungefähr für jeweils drei zusätzliche Ziffern in N.

Der RSA-Algorithmus erzeugt einen öffentlichen Schlüssel zum Verschlüsseln und einen privaten Schlüssel zum Entschlüsseln. Im Vergleich zum Difﬁe-Hellman-Protokoll hat dieser Algorithmus den Vorteil, dass er vollständig asymmetrisch ist. Es besteht keine Notwendigkeit, einen gemeinsamen geheimen Schlüssel zu teilen (und der geheime Schlüssel wird nur an einem einzigen Ort aufbewahrt). Stattdessen hat ein einziger Kommunikationspartner Zugriff auf den geheimen Schlüssel. In diesem Fall wird die Kommunikation jedoch nur gegenüber dem Besitzer des geheimen Schlüssels verschlüsselt. Um in beide Richtungen zu verschlüsseln, erstellt entweder jeder Kommunikationspartner einen asymmetrischen RSA-Schlüssel, oder der andere Kommunikationspartner verschlüsselt und sendet einen symmetrischen Schlüssel (bekannt als hybride Verschlüsselung, da sie ein asymmetrisches mit einem symmetrischen Kryptosystem kombiniert).

###### Zahlentheorie

Mit der Eulerschen Formel seienp und q verschiedene Primzahlen. Ist a nun weder durch p

noch durch q teilbar, dann

a p − 1 q − 1 ≡ 1 mod pq .

RSA

Dieser Algorithmus verschlüsselt durch Potenzierung. Seine Sicherheit beruht auf der rechnerischen Unmöglichkeit, ein Produkt aus Primzahlen zu faktorisieren.

Wurzeln ziehen

Seien p und q verschiedene Promzahlen, N = p q und φ(N) = (p – 1)(q – 1). Für jeden Exponenten n so dass

n ≡ 1 mod 3 N

haben wir

an ≡ a mod N für jede ganze Zahl a .

Wenn m ≡1 mod φ(N), dann gilt nach der Eulerschen Formel am ≡a mod N, d. h. die Potenzierung ist die Identitätsfunktion,

m· ≡ id mod N .

Insbesondere, wenn m = E d das Produkt zweier ganzer Zahlen E und d ist, d. h.

Ed ≡ 1 mod 3 N ,

dann

a = am = aEd = aE d .

Das heißt • d = • 1/E mod N. Eine Potenz zu berechnen ist viel einfacher als eine Wurzel!

Beispiel

Wenn p = 3 und q = 11 dann

N = pq = 33 und 3 N = p − 1 q − 1 = 20 .

Wenn E = 7 und d = 3, dann n = Ed = 21 ≡1 mod 20. Für die Basis 2 prüfen wir zum Beispiel

2E = 27 = 128 = 29 + 3 · 33 ≡ 29 mod N

und

29d = 293 = − 4 3 ≡ − 64 = 2 − 2 · 33 ≡ 2 mod N .

Somit

E 29 = 2 = 29d mod N .

Asymmetrische Kryptosysteme

Euklidischer Algorithmus

Ein gemeinsamer Teiler von zwei ganzen Zahlen a und b ist eine natürliche Zahl, die diese beiden Zahlen teilt. Der größte gemeinsame Teiler von zwei ganzen Zahlen ist die größte natürliche Zahl, die diese beiden teilt. Der größte gemeinsame Teiler von 12 und 18 ist zum Beispiel 6. Bezeichnen wir mit mdc(a,b) den größten gemeinsamen Teiler von a und b,

mdc a, b = die größte natürliche Zahl, die a und b teilt.

Die wiederholte Division mit Rest ergibt einen efﬁzienten Algorithmus zur Berechnung des größten gemeinsamen Teilers, d. h. den euklidischen Algorithmus.

Erweiterter Euklidischer Algorithmus

Für die Berechnung des Exponenten der Entschlüsselungsfunktion benötigen wir mehr Informationen als den größten gemeinsamen Teiler (berechnet durch den euklidischen Algorithmus). Im erweiterten Algorithmus stellt man fest, dass in jedem Schritt des euklidischen Algorithmus der größte gemeinsame Teiler mdc(x, m) von x und m eine Linearkombination (oder Summe von Vielfachen) von x und mist, d. h.

mdc x, m = λx + µm für die ganzen Zahlen x und m .

Als Beispiel nehmen wir mdc(528, 220) = 44 und tatsächlich

44 = 5 · 748 − 7 · 528 .

###### Verschlüsselungsalgorithmus

Wir erinnern uns, dass ein Großbuchstabe für eine öffentliche Zahl steht (und umgekehrt), während ein Kleinbuchstabe für eine geheime Zahl steht. Wir sehen uns zwei Kommunikationspartner an: Alice sendet die Nachricht m über einen unsicheren Kanal heimlich an Bob.

1. Für die Erzeugung des Schlüssels wählt Bob zwei Primzahlen p und q sowie einen Exponenten E,

der teilerfremd ist zu φ(N) := (p – 1)(q – 1).

1. Zur Übermittlung des Schlüssels sendet Bob Alice das Produkt N := pq (den Modul) und den Exponenten E (den öffentlichen Schlüssel).
2. Zur Verschlüsselung berechnet Alice M = mE mod N und übermittelt M an Bob.
3. Zur Entschlüsselung berechnet Bob (mit dem erweiterten euklidischen Algorithmus) d so, dass Ed

≡1 mod (p – 1)(q – 1) (da E teilerfremd ist zu φ(N)), und berechnet Md = mEd = m mod N (mit der Eulerschen Formel).

Zusammenfassend lässt sich sagen, dass die Potenzierung y = xE mod N verschlüsselt, wobei der Exponent E der öffentliche Schlüssel ist. Dementsprechend entschlüsseln wir mit der Umkehrung, dem Ziehen der E-ten Wurzel x = y1/E mod N. Sie ist praktisch nicht berechenbar. Aber modulo N, mit der Eulerschen Formel, gibt es d, so dass

y1/E = yd mod N

für eine Zahl d, die der euklidische Algorithmus aus E sowiep und qberechnet. Der geheime Schlüssel ist also d, oder, hinreichend genau, die Kenntnis der Primfaktoren p und q von N.

###### Sicherheit

Da der Modul N, der Exponent E und die verschlüsselte Nachricht M (= mE) alle öffentlich sind, basiert die rechnerische Sicherheit von RSA allein auf der Schwierigkeit, eine Wurzel aus einer großen Zahl zu finden:

m ≡ E M = M1/E mod N .

### Elliptische Kurven

Wir schreiben IFp := + / p +. Unter allen Kurven besteht der entscheidende Vorteil der elliptischen Kurven (gegeben durch die Gleichung y2 = x3 + ax + b) darin, dass man auf ihnen Punkte hinzufügen kann: p + q + r = 0 wenn eine Linie durch p*,* q und r verläuft.

Difﬁe-Hellman über elliptischen Kurven

Ein Analogon des Difﬁe-Hellman-Protokolls, in dem die iterative Multiplikation einer Zahl modulo p durch die iterative Addition eines Punktes auf einer elliptischen Kurve ersetzt wird. Das Difﬁe-Hellman-Protokoll (über IFp) hat ein Analogon über elliptischen Kurven. Anstatt wiederholt (n-mal) die Basis g in IF \* zu *multiplizieren*, d.h. zu berechnen

Es ist von Vorteil, den Logarithmus über einer endlichen elliptischen Kurve zu verwenden (die Funktion, die für einen gegebenen Punkt G und Y den Skalar x in 5 so bestimmt, dass Y = x G) statt des Logarithmus über IFp (die Funktion, die bei gegebenen Zahlen g und y den Exponenten x so bestimmt, dass y ≡gx mod p). Abhängig von der Anzahl der Bits n von p (in Bezug auf die schnellsten derzeit bekannten Algorithmen), steigt die Zeit zur Berechnung des Logarithmus über einer elliptischen Kurve linear an und benötigt etwa n/2 Operationen, während die Zeit zur Berechnung des multiplikativen Logarithmus sublinear ansteigt und etwa n1/3 Operationen benötigt.

Zum Beispiel entspricht die Sicherheit eines 2048-Bit-Schlüssels für den multiplikativen Logarithmus ungefähr der eines 224-Bit-Schlüssels für den Logarithmus über einer elliptischen Kurve. Ein 512-Bit-Schlüssel für eine elliptische Kurve entspricht einer Länge von 15360 Bit eines RSA-Schlüssels.

###### Elliptische Kurven

Eine elliptische Kurve E über einem endlichen Körper (in dem 0 ≠2, 3) ist eine Gleichung

y2 = x3 + ax + b

Asymmetrische Kryptosysteme

für die Koeffizienten a und b so, dass die Kurve nicht singulär ist, d. h. ihre Diskriminante ist ungleich Null, 4 a3 + 27 b2 ≠ 0.

Die Gleichung y2 = x3 + ax + b ist die Weierstraß-Normalform, aber es gibt einige andere, die sich als rechnerisch efﬁzienter erwiesen haben, wie Montgomery

By2 = x3 + Ax2 + x mit B A2 − 4 ≠ 0 .

Ist die Charakteristik 2, also IFq mit q = 2n, dann lautet die Gleichung y2 + cxy + dy = x3

+ ax + b.

Nach der Wahl eines Definitionsbereichs (z. B. +, \*, 0, 6 oder IFp für eine Primzahl p) bilden die Punkte (x, y) , die diese Gleichung lösen, E(x,y) = 0, eine Kurve in der Ebene. Diese Ebene, 0 steht für die übliche kartesische Ebene, + steht für ein Gitter von Punkten und + / m + steht für das endliche Gitter von Punkten innerhalb des Quadrats der Länge m , dessen untere linke Ecke im Ursprung liegt.

Zusätzlich zu den Punkten in der Ebene gibt es auch den Punkt in der Unendlichkeit (oder Idealpunkt), der mit 0 bezeichnet wird. Die Punkte der elliptischen Kurve sind also gegeben durch

E: = alle Punkte x, y so dass E x, y = 0 ∪ 0

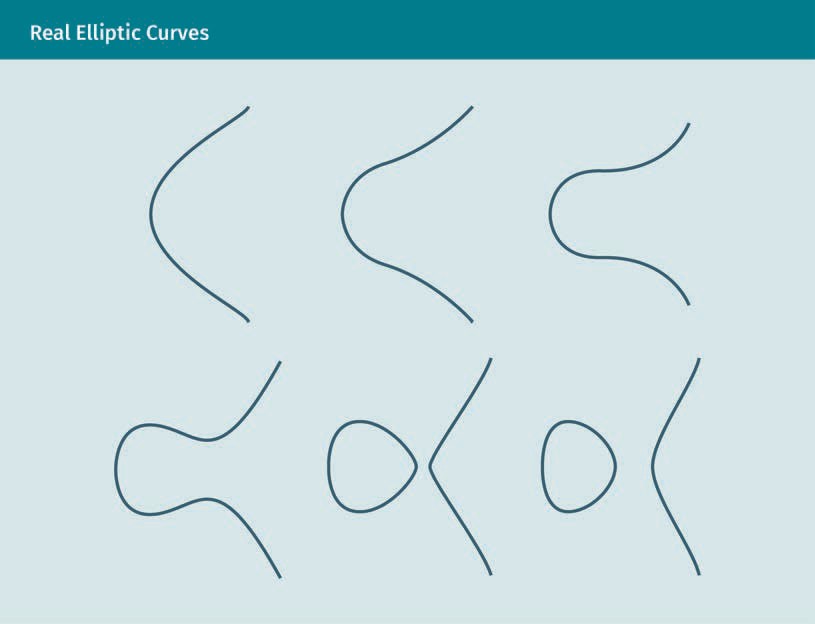
wobei der Begriff des Punktes vom Definitionsbereich abhängt. Auf einem endlichen Körper IFq ist die Anzahl der Punkte # E begrenzt durch q + 1 – t mit t ≤ 2 √{q}, also asymptotisch gleich # IF \* = q –1. Der Algorithmus von Schoof (1999) kann dies in etwa n5 Operationen für n *=* log2 q die Anzahl der binären Ziffern von q berechnen.

q

Stetige und diskrete endliche Kurven

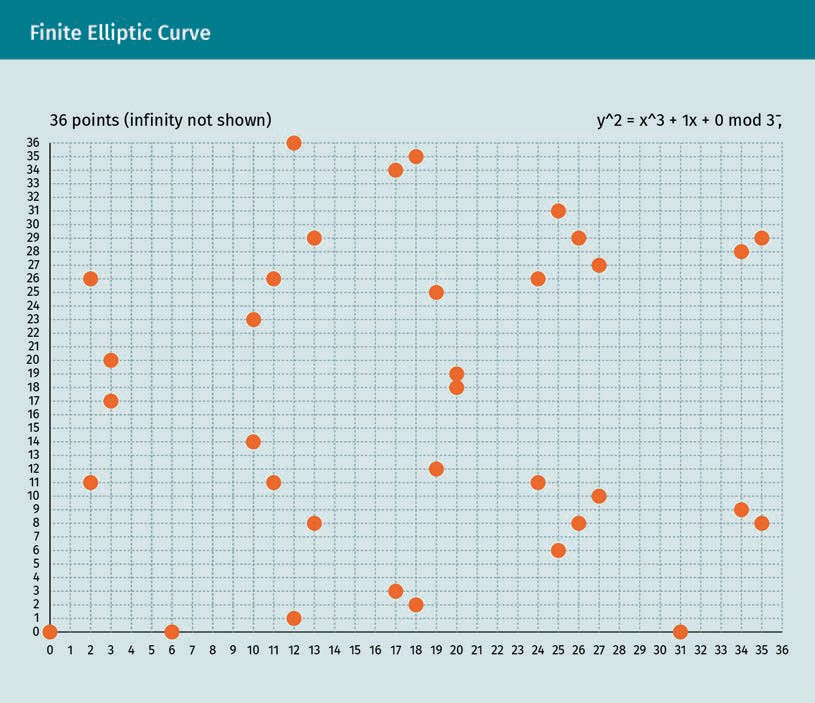
Für den Definitionsbereich 0 nehmen die Kurven in der Ebene der reellen Zahlen für verschiedene Parameter a

und b die folgenden Formen an:



Während wir auf endlichen Körpern eine diskrete Menge von Punkten erhalten (symmetrisch um die mittlere horizontale Achse).

Asymmetrische Kryptosysteme



Kurven in der Kryptografie

Damit der kryptografische Algorithmus auf dieser Kurve sicher ist, d.h. die Berechnung des Logarithmus auf ihr viel Zeit benötigt, gibt es Einschränkungen bei der Wahl vonq = pn und der elliptischen Kurve (bei ihren definierenden Koeffzienten a und b). Als Richtwert gilt zum Beispiel, dass q ≥ 2224 , um bei einer Schlüssellänge von 2048 Bit genauso sicher zu sein wie RSA. Eine sichere Wahl ist zum Beispiel Curve25519, die durch

y2 = x3 + 486662x2 + x

über IFq mit q = p2 und p = 2255 – 19 (daher der Name) gegeben ist. Die Anzahl ihrer Punkte beträgt # E = 2252 + 277423177773585193790883648493. Diese Kurve wurde als neutrale Alternative zu den empfohlenen Kurven des NIST populär, denen man bald misstraute.

###### Schlüsselaustausch mit elliptischen Kurven

ECC verwendet den Difﬁe-Hellman-Schlüsselaustausch, um einen geheimen Schlüssel zu erstellen, ihn in einen kryptografischen Hash umzuwandeln und ihn zur Verschlüsselung der Kommunikation durch einen symmetrischen kryptografischen Algorithmus zu verwenden.

Die Verschlüsselung durch ECC ist durch das ECIES (Elliptic Curve Integrated Encryption Scheme), ein hybrides Verfahren (asymmetrische Kryptografie mit symmetrischer Kryptografie), standardisiert.

Sobald der gemeinsame geheime c-Schlüssel (ein Punkt auf der endlichen elliptischen Kurve) vereinbart ist, leiten Alice und Bob einen Schlüssel für eine symmetrische Chiffre wie AES oder 3DES ab. Die Ableitungsfunktion, die eine geheime Information in das entsprechende Format umwandelt, wird Schlüsselableitungsfunktion (Key Deviation Function KDF) genannt. Eine solche standardisierte Funktion ist ANSI-X9.63-KDF mit der Option SHA-1. Das TLS-Protokoll verwendet zum Beispiel die x-Koordinate des Punktes c, verknüpft sie mit Zahlen, die sich auf die Verbindung beziehen, und berechnet einen kryptografischen Hash dieser verknüpften Zahl.

Wir wollen nun das Difﬁe-Hellman-Protokoll von der Multiplikation auf einem endlichen Körper auf die Addition auf einer endlichen elliptischen Kurve übertragen. Bezeichne G einen Punkt auf der Kurve, und

xG = G + ! + G

die x-fache iterierte Addition über die endliche elliptische Kurve (anstelle von g und gx = g ·g für einen endlichen Körper).

Aufbau

Die kritische kryptografische Zahl ist die Ordnung n des Basispunktes G , die groß genug sein muss.

Hier ein Beispiel für einen Basispunkt. Die elliptische Kurve Curve25519 mit

y2 = x3 + 486662x2 + x

über IFq with q = p2 mit p = 2255 – 19, verwendet als Basispunkt G = (xG, yG) eindeutig bestimmt durch

Schritte

1. Im ECDH-Protokoll (Elliptic Curve Difﬁe-Hellman) können Alice und Bob offen einen geheimen Schlüssel erstellen, indem sie eine Potenz q aus einer geeigneten Primzahl p , eine geeignete elliptische Kurve E über IFq und einen geeigneten Punkt G in E kombinieren.
2. Um eine Hälfte des Schlüssels zu generieren, wählt Alice eine Zahl a, berechnet A = a G und

übermittelt A an Bob.

1. Um eine weitere Hälfte des Schlüssels zu generieren, wählt Bob eine Zahl b, berechnet B ≡b G und sendet B an Alice.

Der gemeinsame geheime Schlüssel zwischen Alice und Bob lautet

c: = bA = baG = abG = aB .

Damit beide denselben Schlüssel c berechnen können, muss die Addition das Assoziativ- und Kommutativgesetz erfüllen, d.h. es ist unerlässlich, dass E eine Gruppe ist.

Das ECDHE-Protokoll, bei dem das zusätzliche E für „ephemeral“ (kurzlebig, flüchtig) steht, verwendet den gleichen Schlüsselaustausch wie das ECDH-Protokoll, verwirft aber die Schlüssel (die notwendigerweise durch permanente Schlüssel signiert werden, um die Identität zu bezeugen) direkt nach der Sitzung (Corbellini, 2015b).

Asymmetrische Kryptosysteme

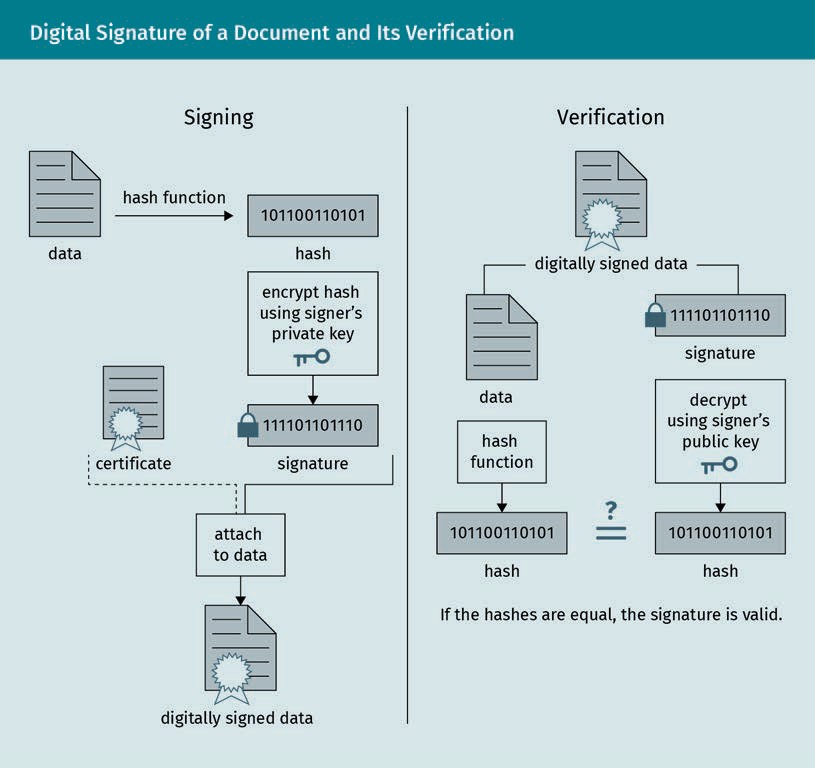
### Signaturen

Die Public-Key-Kryptografie verwendet zwei Schlüssel: einen öffentlichen Schlüssel und einen privaten Schlüssel. Normalerweise wird der öffentliche Schlüssel zum Verschlüsseln verwendet, während der private Schlüssel zum Entschlüsseln verwendet wird. So kann ein Text von der verschlüsselnden Person (Alice) nur an eine Person, die entschlüsselnde Person (Bob), übertragen werden. Die Rollen des öffentlichen und des privaten Schlüssels können vertauscht werden. Der private Schlüssel wird dann zum Verschlüsseln verwendet, während der öffentliche Schlüssel zum Entschlüsseln verwendet wird. Auf diese Weise kann der Verschlüsselnde allen Entschlüsselnden (denjenigen, die den öffentlichen Schlüssel besitzen) beweisen, dass er im Besitz des privaten Schlüssels ist: der digitalen Signatur.

Die Theorie hinter der Verschlüsselung mit dem öffentlichen Schlüssel (digitale Nachrichten) oder dem privaten Schlüssel (digitale Signatur) ist fast identisch. Lediglich die Rollen der Argumente der Falltürfunktion sind vertauscht. Beim RSA-Algorithmus zum Beispiel ist dieser Austausch von Variablen tatsächlich schon alles, was passiert. In der Praxis verschlüsseln der private Schlüssel und ein kryptografischer Hash in der Regel jedoch die Auffüllungen des Klartextes, um Schwachstellen zu vermeiden, die den Schlüssel offenbaren, wenn der Text zu kurz ist. Während bei der Verschlüsselung mit öffentlichen Schlüsseln die Funktion, mit welcher der Text (die Auffüllung) zunächst umgewandelt wird, leicht umkehrbar ist, ist die Funktion, mit welcher der Text zunächst umgewandelt wird (der Hash), bei der Verschlüsselung mit privaten Schlüsseln kaum umkehrbar.

Digitale Signatur Auf diese Verschlüsselung einer Nachricht mit dem privaten Schlüssel folgt die Entschlüsselung mit dem öffentlichen Schlüssel, um zu überprüfen, ob die ursprüngliche Nachricht

mit dem privaten Schlüssel verschlüsselt wurde.



###### RSA-Signaturalgorithmus

Beim RSA-Signaturalgorithmus besteht der einzige Unterschied darin, dass die Exponenten E and d

ihre Rollen tauschen, d. h. die signierte Nachricht ist M = md (statt mE).

Das Signieren und Dechiffrieren sind beide durch • d für den privaten Schlüssel d gegeben. Das Signieren eines verschlüsselten Dokuments (für den öffentlichen Schlüssel E , der d entspricht) ist also gleichbedeutend mit dessen Entschlüsselung! In der Praxis werden verschiedene Schlüsselpaare zum Verschlüsseln/Entschlüsseln, zum Signieren/Verifizieren verwendet und ein kryptografischer Hash h(d) des Dokuments d wird signiert, d. h. eine kleine Zahl, die das Dokument identiﬁziert.

Wir stellen fest, dass er anstelle der Originalnachricht einen kryptografischen Hash (z. B. mit dem Algorithmus MD5) der Originalnachricht signiert, und zwar mit zusätzlichen Informationen wie dem Namen des Signierenden und den Algorithmen, die zur Verschlüsselung und Berechnung des Hash verwendet wurden.

###### Digital Signature Algorithm (DSA)

Elgamal (1985) war der erste, der zeigte, wie man einen Verschlüsselungs- und Signaturalgorithmus auf der Grundlage des Difﬁe-Hellman-Protokolls entwickelt. Zwar wird der Verschlüsselungsalgorithmus nur selten verwendet (das Standard-Kryptografie-Befehlszeilentool GnuPG bietet ihn als erste Alternative zu RSA an), aber der Signaturalgorithmus steckt hinter dem des Digital Signature Algorithm (DSA), der im Digital Signature Standard (DSS) der US-Regierung verwendet wird, der 1994 vom NIST herausgegeben wurde. Elliptic Curve DSA (ECDSA) ist eine Variante des DSA, die Punkte auf endlichen (elliptischen) Kurven anstelle von ganzen Zahlen verwendet.

### Schlüsselaustausch und Public-Key-Infrastrukturen

Im Vergleich zur symmetrischen Kryptografie entfällt bei der asymmetrischen Verschlüsselung das Risiko der Kompromittierung des Schlüssels zur Entschlüsselung, das mit dem Austausch des Schlüssels mit dem Chiffrierenden verbunden ist. Diese sichere Kommunikation mit jedermann über einen unsicheren Kanal ist ein großer Vorteil gegenüber der symmetrischen Kryptografie. Wir wollen uns zunächst die klassischen Methoden zum Austausch eines symmetrischen Schlüssels ins Gedächtnis rufen, bevor wir das asymmetrische Gegenstück betrachten. Die asymmetrische Kryptografie machte es zwar möglich, einen geheimen Schlüssel offen auszutauschen, aber diese Annehmlichkeit verschleiert die Identität des Schlüsselbesitzers und macht sie anfällig für einen Man-in-the-Middle-Angriff, den die Public-Key-Infrastruktur durch die Verwendung von Zertifikaten (digitale Signaturen öffentlicher Schlüssel durch Dritte) unterbindet.

###### Symmetrische Kryptosysteme

Ein symmetrischer Schlüssel muss heimlich weitergegeben werden. Mögliche Methoden sind

Asymmetrische Kryptosysteme

* Ableitung von einem Basisschlüssel unter Verwendung einer Schlüsselableitungsfunktion (Key Derivation Function, KDF), einer kryptografischen Hashfunktion, die einen geheimen Schlüssel aus geheimen – und möglicherweise anderen öffentlichen – Informationen ableitet, z.B. einer eindeutigen Zahl,
* Erstellung eines Schlüssels aus Schlüsselteilen, die sich im Besitz verschiedener Personen befinden, z. B. als Analogie zum One-Time-Pad. Wenn *s* die geheime (binäre) Zahl ist, dann ist s = s1 ⊕s2 ⊕…⊕sn für die Teilgeheimnisse s1, s2, … Die Rekonstruktion von s ist nur möglich, wenn alle s1, s2, …

kombiniert werden.

* Übermittlung über einen anderen Kanal, z. B. persönlich, einen versiegelten Brief, per Telefon oder durch Quantenverschränkung, bei der Quantenpartikel so miteinander verbunden sind, dass sich der Zustand des einen Teilchens unabhängig von der räumlichen Trennung sofort auf das andere überträgt, obwohl sich Informationen nicht schneller als mit Lichtgeschwindigkeit bewegen können.

###### Man-In-The-Middle-Angriff (MITM)

Der große Vorteil der asymmetrischen Kommunikation besteht darin, dass die Geheimhaltung von der Authentifizierung getrennt werden kann, d. h. für einen erstellten Geheimtext wird nur der jeweilige öffentliche Schlüssel zur Entschlüsselung benötigt. Hinzu kommt, dass es keinerlei Informationen über den Dechiffrierenden gibt. Für die Kommunikation zwischen mehreren Parteien, z.B. Alice, Bob und Eve, müssen die öffentlichen Schlüssel authentifiziert werden, d.h. die Dechiffrierenden Bob und Eve müssen im Verzeichnis der Chiffrierenden Alice authentifiziert werden. Andernfalls könnte Eve Alice vorgaukeln, dass sie mit Bob kommuniziert, indem sie ihren öffentlichen Schlüssel durch den von Bob ersetzt.

Bei einem MITM platziert sich der Angreifer zwischen den Kommunikationspartnern und nimmt gegenüber jedem von ihnen die Identität des anderen an, um Nachrichten abzufangen. Ein Beispiel:

1. Bob sendet seinen öffentlichen Schlüssel an Alice. Eve fängt ihn ab und sendet Alice ihren eigenen öffentlichen Schlüssel, der Bob als Eigentümer angibt. Wenn Alice eine Nachricht an Bob sendet, dann verwendet sie, ohne es zu wissen, den öffentlichen Schlüssel von Eve!
2. Alice verschlüsselt eine Nachricht mit dem öffentlichen Schlüssel von Eve und sendet sie an Bob.
3. Eve fängt die Nachricht ab und entschlüsselt sie mit ihrem privaten Schlüssel. Sie kann die Nachricht lesen und sie verändern.
4. Eve verschlüsselt die Nachricht mit Bobs öffentlichem Schlüssel.
5. Bob entschlüsselt die Nachricht mit seinem privaten Schlüssel und schöpft keinen Verdacht.

Sowohl Alice als auch Bob sind davon überzeugt, dass sie den öffentlichen Schlüssel des jeweils anderen benutzt haben, aber in Wirklichkeit benutzen sie den von Eve!

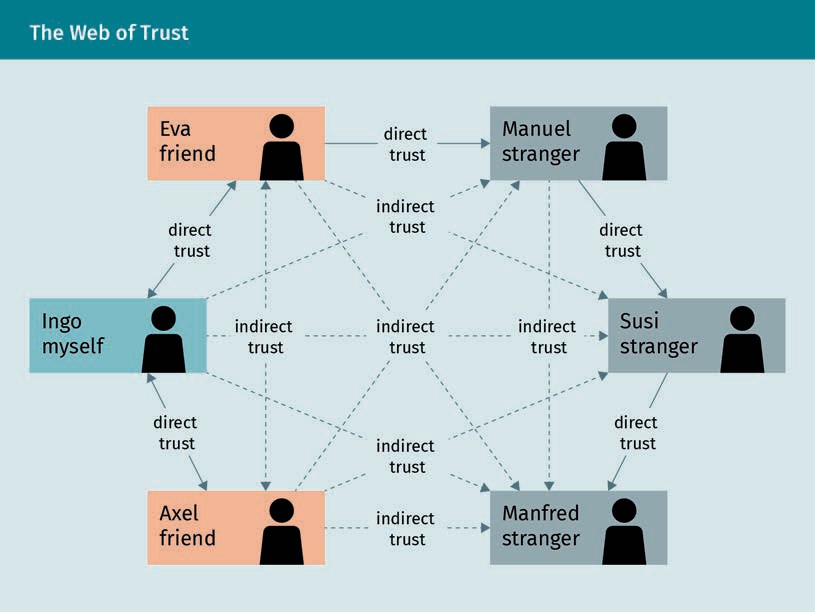
###### PKI

Eine Public-Key-Infrastruktur (PKI) eines Netzwerks stellt das Vertrauen zwischen den räumlich getrennten Benutzern her, indem sie diese zunächst authentifiziert und dann ihre öffentlichen Schlüssel durch Signieren (digitale Zertifikate) autorisiert und verteilt. In Institutionen und Unternehmen wird eine PKI oft als „Vertrauenshierarchie“ von Zertifizierungsstellen implementiert, während sie in weniger organisierten Gemeinschaften dezentralisiert sein kann und das Vertrauen von den Benutzern selbst aufgebaut wird. Eine PKI umfasst

* + (digitale) Zertifikate. Es handelt sich dabei um öffentliche Schlüssel, die signiert sind, um ihre Benutzer zu authentifizieren. Neben dem Namen und dem Schlüssel enthalten sie weitere persönliche Daten, wie z. B. eine E-Mail-Adresse und in der Regel ein Ablaufdatum.
  + Zertifikatssperrliste (Certiﬁcate Revocation List, CRL). Dies ist eine Liste von Zertifikaten, die vor Ablauf ihrer Gültigkeit widerrufen wurden, weil der Schlüssel kompromittiert wurde oder der Schlüsselinhaber aufgrund seines Verschwindens nicht mehr vertrauenswürdig ist.
  + Verzeichnisdienst. Dies ist eine durchsuchbare Datenbank mit den emittierten Zertifikaten. Beispiele hierfür sind eine Vertrauenshierarchie, ein LDAP-Server (Lightweight Directory Access Protocol, ein Standard, der von großen Unternehmen zur Verwaltung des Zugriffs von Benutzern auf Geräte, Drucker, Server und Anwendungsdaten verwendet wird) und das Web of Trust, ein Server, der eine Datenbank hostet, die über ein Webformular durchsucht werden kann.

###### Philosophie der Lösungen

Dritte, d.h. andere Identitäten mit privaten Schlüsseln, die mit ihren digitalen Signaturen bestätigen, bestätigen die Identität des Schlüsselinhabers. Es stellt sich jedoch erneut das Problem der Identität des öffentlichen Schlüssels: Wie können wir die Identitäten der Besitzer privater Schlüssel sicherstellen? Es gibt zwei Lösungen: hierarchische Autoritäten und das Web of Trust.



Während beim Web of Trust die durch Vertrauen aufgebauten Verbindungen einen Graphen bilden, bilden sie beim Ansatz der hierarchischen Autoritäten einen Baum.

Asymmetrische Kryptosysteme

Hierarchische Autoritäten

Beim Ansatz der hierarchischen Autoritäten werden die Besitzer privater Schlüssel durch hierarchische Ebenen unterschieden. Auf der höchsten Ebene befinden sich die Stammautoritäten, denen wir bedingungslos vertrauen. VeriSign, GeoTrust und Commode sind zum Beispiel wichtige US-Zertifizierungsunternehmen. Ein Blick in den Ordner /etc/ssl/certs in der Linux-Distribution openSUSE zeigt eine deutsche (TeleSec der Deutschen Telekom AG, dem ehemaligen staatlichen Telekommunikationsunternehmen), drei spanische (Firmaprofesional, ACCVRAIZ1-Agencia de Tecnología y Certiﬁcación Electrónica und ACC RAIZ FNMT-Fábrica Nacional de Moneda y Timbre) und viele amerikanische Autoritäten.

Web of Trust

Im Web of Trust sind die Besitzer privater Schlüssel nicht voneinander zu unterscheiden. Das Fehlen von Stammautoritäten, d. h. von Entitäten, denen man bedingungslos vertraut, wird durch das Vertrauen kompensiert, das ursprünglich dadurch aufgebaut wurde, dass man den öffentlichen Schlüssel persönlich erhalten hat (z. B. bei einer Schlüsselsignatur-Party) oder dass man den Schlüssel über einen anderen Kanal erhalten hat (Website, E-Mail usw.) und die Prüfsumme wieder über einen anderen Kanal (Telefon, SMS, Instant Messenger usw.) übermittelt hat. Vertrauen wird im aufgebauten Netzwerk transitiv weitergegeben.

###### Standardisierung von Philosophien im Internet

Im Internet wurde das System des Vertrauens durch hierarchische Autoritäten durch das Schema X.509 standardisiert, das hauptsächlich zur Verschlüsselung der Kommunikation zwischen einem Benutzer und einer (kommerziellen) Website (aber auch zwischen Benutzern in Unternehmensumgebungen, wie z. B. die S/MIME-E-Mail-Verschlüsselung) verwendet wird, sowie durch das OpenPGP-Schema (wie es durch das GnuPG-Programm implementiert wurde), dessen Hauptanwendung die Verschlüsselung von E-Mails ist. Dieses Schema lehnt jede Hierarchie radikal ab. Der Benutzer kann einen öffentlichen Schlüssel mit einer E-Mail-Adresse auf einem Public-Key-Server veröffentlichen, ohne auch nur zu bestätigen, dass er Zugriff auf dieses E-Mail-Konto hat.

###### DANE

Die Internet Engineering Task Force (IETF) hat (in RFC 63941: DANE Use Cases und RFC 66982: DANE Protocol) das DANE-Protokoll vorgeschlagen, das darauf abzielt, die Protokolle TLS, DTLS, SMTP und S/MIME mit DNSSEC kryptografisch abzusichern. Mit DNSSEC kann ein DNS-Resolver eine DNS-Auflösung authentifizieren, d. . feststellen, ob sie mit derjenigen auf dem maßgeblichen DNS-Server identisch ist, indem er seine Signatur (des verbindlichen DNS-Servers) überprüft. Anstatt sich wie bei diesen Protokollen vollständig auf Zertifizierungsstellen (CAs) zu verlassen, können die Inhaber von Definitionsbereichen die CAs einschränken, die die Zertifikate der Domain validieren, und können selbst Zertifikate ausgeben, ohne auf CAs zu verweisen.

Bei Verwendung von CAs gibt es keine Einschränkung bei den ausgestellten Zertifikaten. Wenn ein Angreifer die Kontrolle über eine einzige der vielen CAs erlangt, denen der Client vertraut, kann er gefälschte Zertifikate für jede Domain emittieren. DANE ermöglicht es Clients, die DNS-Server zu fragen, welche Zertifikate vertrauenswürdig sind, so dass der Domain-Inhaber den Geltungsbereich einer CA einschränken kann. Wenn dem Benutzer ein Domain-Name-Zertifikat übergeben wird (als Teil des anfänglichen TLS-Handshakes), kann der Client das Zertifikat mit einem TLSA-Ressourcendatensatz (TLSA RR) abgleichen, der im DNS für den vom maßgeblichen DNS-Server authentifizierten Dienstnamen veröffentlicht wurde.

Web of Trust Inhaber von privaten Schlüsseln

bestätigen die Identität der anderen, indem sie sich unter Gleichen gegenseitig Vertrauen schenken.

Schlüsselsignatur-Partys Das sind Treffen, bei denen die Teilnehmer ihre öffentlichen Schlüssel austauschen und gegenseitig signieren.

Der am weitesten verbreitete Standard für eine PKI ist die Hierarchie der X.509-Zertifizierungsstellen.

X.509 wurde erstmals 1998 veröffentlicht und wird von der Standardisierungsabteilung der Internationalen Fernmeldeunion (ITU-T) entwickelt. X.509 legt insbesondere ein Standardformat für elektronische Zertifikate und einen Algorithmus für die Validierung des Zertifizierungspfads fest. Das wichtigste Profil, PKIX-Zertifikat und CRL-Profil (PKIX), wurde von der IETF als Teil von RFC 3280, derzeit RFC 5280, entwickelt. Es wird von allen gängigen Webbrowsern wie Chrome und Firefox unterstützt, die über eine Liste vertrauenswürdiger X.509-Zertifizierungsstellen verfügen.

Im Einzelnen enthält die TLSA RR den Eintrag Certiﬁcate Usage, dessen Wert die Stelle einschränkt, die das Zertifikat für den Benutzer validieren darf. Je niedriger die Ebene, desto restriktiver ist sie.

* + Ebene 0: PKIX-TA (Zertifizierungsstellenzwang). Das Vertrauen des Clients beruht auf einer PKIX-Stelle.
  + Ebene 1: PKIX-EE (Servicezertifikatszwang). Das Vertrauen des Clients beruht auf einem PKIX-Zertifikat.
  + Ebene 2: DANE-TA (Trust-Anchor-Zusicherung). Das Vertrauen des Clients beruht auf einer Stelle, bei der es sich im Gegensatz zu PKIX-TA nicht um eine PKIX-Zertifizierungsstelle handeln muss.
  + Ebene 3: DANE-EE (Von Domain ausgestelltes Zertifikat). Das Vertrauen des Clients beruht auf einem Zertifikat, bei dem es sich, anders als bei PKIX-EE, nicht um ein PKIX-Zertifikat handeln muss.

Die DANE-Prüfung dient zur Bestätigung von Zertifikaten, die von öffentlichen Zertifizierungsstellen ausgestellt wurden. Mit den DANE-Werten (zwei und drei) hat der Domaininhaber die Möglichkeit, seine eigenen, selbst signierten Zertifikate für seine TLS-gesicherten Dienste zu erstellen, ohne eine dem Client bekannte Zertifizierungsstelle einbeziehen zu müssen. Durch die Wahl zwischen „Trust Anchor“ (TA) und „End Entity“ (EE) kann der Domaininhaber selbst entscheiden, ob er die DANE-Sicherheit an einem Zertifizierungsstellen- oder Server-Zertiﬁkat verankern möchte.

###### Hybride Kryptosysteme

Hybride Verschlüsselung Ein Zwei-Schlüssel-Algorithmus wird verwendet, um die Kommunikationsteilnehmer zu authentifizieren, indem die Nachrichten digital signiert werden, oder um einen Schlüssel für die Ein-Schlüssel-Kryptografie auszutauschen, und so eine efﬁziente Kommunikation zu

ermöglichen.

Zwischen zwei Parteien ist es üblich, einen Hash, z. B. einer Nachricht, als Verschlüsselungsschlüssel zu erstellen. MITM wird durch die Authentifizierung der entsprechenden öffentlichen Schlüssel bei einer Zertifizierungsstelle verhindert. Da kryptografische Algorithmen mit einem Schlüssel um einen beträchtlichen Faktor efﬁzienter sind als kryptografische Algorithmen mit zwei Schlüsseln, wird die Zwei-Schlüssel-Verschlüsselung hauptsächlich für die hybride Verschlüsselung verwendet; hierbei wird der Zwei-Schlüssel-Algorithmus verwendet, um die Kommunikationspartner zu authentifizieren, indem die Nachrichten digital signiert werden, oder um einen Schlüssel für die Ein-Schlüssel-Kryptografie auszutauschen, um eine efﬁziente sichere Kommunikation zu ermöglichen.

Im TLS-Protokoll (Transport Layer Security; früher SSL), das sichere Websites im World Wide Web verschlüsselt, verwendet ein kryptografisches Paket wie TLS\_RSA\_WITH\_3DES\_EDE\_CBC\_SHA (Identifizierungscode 0x00 0x0a) beispielsweise RSA zur Authentifizierung und zum Austausch der Schlüssel, 3DES im CBC-Modus zur Verschlüsselung der Verbindung und SHA als kryptografischen Hash.

Asymmetrische Kryptosysteme

Zusammenfassung

Die symmetrische Kryptografie leidet unter dem Problem der Schlüsselverteilung. Die asymmetrische Kryptografie löst dieses Problem scheinbar auf Anhieb, indem sie die Verwendung unterschiedlicher Schlüssel zum Ver- und Entschlüsseln ermöglicht. Allerdings muss dazu die Identität des Schlüsselinhabers bestätigt werden. Dies kann persönlich oder durch Dritte geschehen, z. B. durch Identitäten mit privaten Schlüsseln, die mit ihren digitalen Signaturen die Eigentümerschaft bestätigen. Es stellt sich jedoch erneut das Problem der Identität des privaten Schlüssels: Wie können wir die Identitäten dieser Besitzer privater Schlüssel sicherstellen? Dafür gibt es zwei Lösungen. Bei dem Ansatz über hierarchische Autoritäten werden die Besitzer privater Schlüssel durch hierarchische Ebenen unterschieden. Auf der höchsten Ebene befinden sich die Stammautoritäten, denen wir bedingungslos vertrauen. Im Web of Trust wird das Vertrauen von einem zum anderen übertragen.

Asymmetrische Kryptografie beruht auf einer Falltürfunktion, die einfach sein muss, deren Umkehrung aber ohne Kenntnis einer Abkürzung – des Schlüssels – praktisch nicht möglich sein darf.

Die Schwierigkeit, die Umkehrung zu berechnen, entspricht der Schwierigkeit der Entschlüsselung, d. h. der Umkehrung der Verschlüsselung. Die Berechnung der Umkehrfunktion wird durch modulare (oder zirkuläre) Arithmetik erschwert.

Die derzeit am weitesten verbreitete Kryptografie verwendet elliptische Kurven. Das Difﬁe-Hellman-Protokoll (über IFp) hat ein Analogon über elliptischen Kurven. Der Vorteil der Verwendung elliptischer Kurven sind kürzere Schlüssel. Kleine Schlüssel für ECC erreichen das gleiche Maß an Sicherheit wie große Schlüssel für RSA oder DH. So entspricht beispielsweise die Sicherheit eines 224-Bit-Schlüssels bei ECC der eines 2048-Bit-Schlüssels bei RSA oder DH. Dieser Faktor bei der Reduzierung der Schlüsselgröße entspricht einem ähnlichen Faktor bei der Reduzierung des Rechenaufwands.



# Lektion 4

## Authentifizierung

#### LERNZIELE

Nach der Bearbeitung dieser Lektion wissen Sie …

… über die Benutzerauthentifizierung durch Passwort, persönliche Identifikationsnummer (PIN), Chipkarte und biometrische Identifikatoren Bescheid.

… wie die Authentifizierung über eine Entfernung am sichersten erreicht wird.

… wie der geheime Schlüssel durch Challenge- und Response-Protokolle niemals preisgegeben wird.

… wie Informationen durch Zero-Knowledge-Beweise nicht durchsickern.

… wie Kerberos zwischen Benutzern und Servern vermittelt, ohne dass einer der beiden sein Passwort dem anderen preisgibt.

DL-E-DLMCSEAITSC02-U04

1. Authentifizierung

### Einführung

Authentifizierung ist die Identifizierung einer Person oder von Daten oder die Bestätigung, dass der Benutzer derjenige ist, der er vorgibt zu sein, z. B. bei der Anmeldung an einem Server durch Eingabe eines Benutzernamens und eines Passworts. Die Authentifizierung prüft zudem, ob eine Nachricht, z. B. eine per E-Mail an eine Bank gesendete Anweisung, authentisch ist, d. h. ob sie zwischen dem Zeitpunkt des Versands und dem Zeitpunkt ihres Eintreffens unverändert geblieben ist.

Diese Lektion behandelt ausschließlich die erste Art der Authentifizierung, die Identifizierung einer Person, des Benutzers, die im Internet besonders wichtig ist, da die Person sehr weit entfernt ist. In diesem Sinne zeigt die Identiﬁkation einem Computer oder einem Netzwerk, wer der Benutzer ist, in der Regel durch seinen Benutzer- (oder Konto-) Namen. Darauf folgt die Authentifizierung, die einen Computer oder ein Netzwerk davon überzeugt, dass eine Person diejenige ist, die sie vorgibt zu sein.

Zur Authentifizierung kann der Benutzer Informationen, die nur er kennt, als Identitätsnachweis verwenden. Der Benutzer kann sein Passwort oder eine persönliche Identifikationsnummer (PIN) angeben oder ein Objekt vorweisen, wie z. B. ein Software-Zertifikat mit öffentlichen oder privaten Schlüsseln, ein Hardware-Zertifikat, wie z. B. ein Token oder eine Chipkarte, die einen Mikroprozessor enthält, der einen Schlüssel speichert und kryptografische Algorithmen ausführt, oder ein mobiles Endgerät oder ein E-Mail-Konto, das einen Code empfängt. Es gibt auch biometrische Identifikatoren, wie Fingerabdrücke, Gesichtserkennung oder Iris-Scanner.

Die Authentifizierung ist nicht zu verwechseln mit der Autorisierung, der abschließenden Bestätigung der Authentifizierung, mit der die Berechtigung des Benutzers zum Zugriff auf bestimmte Inhalte festgelegt wird.

Das Authentifizierungsprotokoll kann ein einfaches Protokoll, ein Zwei-Faktor-Protokoll, einseitig oder wechselseitig sein. Wenn ein einfacher Beweis ausreicht, wie zum Beispiel eine PIN oder ein Passwort, ist es einfach. Es ist ein Zwei-Faktor-Protokoll, wenn mehr als ein Beweis erforderlich ist, zum Beispiel eine PIN und eine Chipkarte. Es handelt sich um eine einseitige Authentifizierung, wenn die Authentifizierung nur in eine Richtung erfolgt, z. B. wenn sich Partei A gegenüber Partei B authentifiziert, oder um eine wechselseitige Authentifizierung, wenn die Authentifizierung in beide Richtungen erfolgt, z. B. wenn sich Partei B auch gegenüber Partei A authentifiziert. Die meisten Betriebssysteme (wie Linux) und Anwendungen speichern einen Hash der Authentifizierungsdaten und nicht die Authentifizierungsdaten selbst. Bei der Authentifizierung muss der Hash der eingegebenen Authentifizierungsdaten mit dem gespeicherten Hash übereinstimmen, damit sie verifiziert werden können. Selbst wenn ein Angreifer die Hashes kennt, ist es praktisch unmöglich, festzustellen, welche Authentifizierungsdaten zu einem bestimmten Hash passen.

### Passwörter

Passwörter werden in den Systemdaten der autorisierten Benutzer gespeichert. Dies ist die gängigste Methode zur Authentifizierung: kostenlos, praktisch und privat. Es sollte leicht zu merken, aber schwer zu erraten sein.

Authentifizierung

###### Vorteile

Die Authentifizierung durch etwas, das man weiß (z. B. ein Passwort), hat gegenüber der Authentifizierung durch etwas, das man ist (biometrische Daten wie ein Fingerabdruck), den Vorteil, dass keine komplizierte Hardware erforderlich ist, dass sie sicher gespeichert wird und dass sie nicht gefälscht werden kann.

Die Authentifizierung durch etwas, das man weiß (z. B. ein Passwort), hat gegenüber der Authentifizierung durch etwas, das man hat (z. B. eine Chipkarte), den Vorteil, dass man es nicht mit sich herumtragen muss, dass es transparent gespeichert ist und dass es nicht verloren gehen, gestohlen oder erpresst werden kann.

###### Nachteile

Je aussagekräftiger ein Passwort ist, desto leichter ist es zu erraten (z. B. ein Wort in der Sprache des Benutzers), doch je weniger ein Muster vorhanden ist, desto schwieriger ist es, es sich zu merken. Ein Kompromiss ist eine Passphrase, d. h. ein vollständiger Satz anstelle eines einzelnen Wortes. Dieser ist zwar länger, aber sein Inhalt ist aussagekräftiger und daher leichter zu merken als eine unstrukturierte Zeichenfolge. Zur Verkürzung wird der erste Buchstabe eines jeden Wortes genommen. So kann zum Beispiel aus „Es ist besser, eine Kerze anzuzünden, als die Dunkelheit zu verfluchen“ „EibeKaadDzF“ werden.

Eine weitere Möglichkeit ist die Verwendung eines Passwortmanagers. Dieses Programm speichert alle Passwörter in einer Datei, die mit einem Master-Passwort verschlüsselt wird. Dank dieses Programms brauchen wir uns die Passwörter nicht mehr zu merken und können sehr komplex sein. Dieser Komfort ist nicht ohne Risiko: Das Master-Passwort ist immer noch ein Passwort, das bei Nutzung für ein weniger wichtiges Konto die Sicherheit aller anderen Konten gefährden kann, einschließlich der wichtigsten Konten.

###### Angriffe

Mögliche Angriffe während der Eingabe des Passworts einer anderen Person umfassen:

* Spionage bei der Eingabe des Passworts. Abhilfemaßnahmen sind lästig, z. B. das Abdecken der getippten Buchstaben oder der Tastatur oder die Verwendung einer Bildschirmtastatur.
* Keylogging (Abfangen der Tastatuseingaben)
* Login-Spooﬁng, bei dem das Konto eines Benutzers ein Login-Eingabeformular fälscht, so dass die eingegebenen Login-Daten des nächsten Benutzers gespeichert werden, eine Fehlermeldung angezeigt wird und der erste Benutzer ausgeloggt wird.
* Einen Benutzer per E-Mail oder am Telefon als Systemadministrator nach dem Passwort fragen.
* Einen Systemadministrator nach dem Passwort fragen, indem man sich als ein angeblicher Benutzer ausgibt, der sein Passwort vergessen hat.
* Einen Benutzer auffordern, sein Passwort über ein angebliches Eingabeformular zu ändern.

Angriffe auf das gespeicherte Passwort einer anderen Person nutzen vor allem die folgende Schwäche aus: Da man sich Passwörter merken muss, tendieren sie dazu, einem Muster zu folgen. Sie können zum Beispiel aus (Geburts-)Datumsangaben und bestimmten Namen aufgebaut sein. Übliche Vorgehensweisen sind die Umkehrung der Schreibweise oder die Änderung der Großschreibung von

Passwort

Eine geheime Zeichenfolge, die mit einer Benutzeridentität verknüpft ist und Zugang zu einem System (z. B. einem Computer) gewährt, wird als Passwort bezeichnet.

* + Benutzernamen,
  + Vor- oder Nachnamen,
  + Namen von Ehepartnern, Kindern, Freunden oder Haustieren,
  + Telefonnummern oder Adressen

Diese wahrscheinlicheren Kandidaten können zuerst ausprobiert werden. Alternativ kann der Angreifer auch bereits bekannt gewordenen Passwörter verwenden.

### Challenge-Response- und Zero-Knowledge-Protokolle

Challenge-Response-

Protokoll Dieses Protokoll stellt eine Aufgabe, die nur von einem Benutzer gelöst werden kann, der über die

Authentifizierungsdaten verfügt.

Zero-Knowledge-Pro-

tokoll Dieses Protokoll (ursprünglich vorgestellt von Goldwasser, Micali und Rackoff) weist die Kenntnis eines Geheimnisses nach, gibt aber keine anderen Informationen

preis.

Nonce In einem kryptografischen Protokoll (z. B. zur Authentifizierung) steht dies für eine (zufällig erzeugte) Zahl, die nur ein einziges Mal verwendet wird.

Challenge-Response-Protokolle bestehen in der Regel aus einer von zwei Aufgaben. Erstens kann es sich um eine kryptografische Hashfunktion oder eine Verschlüsselungsfunktion eines symmetrischen kryptografischen Algorithmus handeln, dessen geheimer Schlüssel vom Beweiser und vom Verifizierer gemeinsam genutzt wird. Der Verifizierer generiert eine Zufallszahl, und der Beweiser antwortet mit dem Ergebnis der Anwendung der Hashfunktion auf diese Zahl. Alternativ kann es sich um einen digitalen Signaturalgorithmus handeln, bei dem der Beweiser mit seinem privaten Schlüssel eine vom Verifizierer generierte Nachricht unterzeichnet, die der Verifizierer mit dem öffentlichen Schlüssel überprüft.

Ein Zero-Knowledge-Protokoll geht – theoretisch – noch weiter, da es zeigt, wie ein Beweiser einem Verifizierer die Kenntnis eines Geheimnisses beweisen kann, ohne dass andere Informationen (als dieser Wissensnachweis) preisgegeben werden (Goldwasser et al., 1985).

###### Challenge-Response

Ein Challenge-Response-Protokoll stellt eine Aufgabe, die nur von einem Benutzer mit zusätzlichen Authentifizierungsdaten gelöst werden kann. Ein Beispiel für eine Aufgabe wäre ein (zufällig) generierter Wert, der mit dem Passwort für den Verschlüsselungscode verschlüsselt wird. Als Beispielantwort wird ein ähnlich verschlüsselter Wert angegeben, der von dem Originalwert abhängt und damit beweist, dass der Benutzer den Originalwert entschlüsseln konnte. Dieses beispielhafte Protokoll ist auf Chipkarten sehr verbreitet.

Dieser (zufällig) generierte Wert ist eine Nonce und verhindert einen Replay-Angriff, bei dem die ausgetauschten Daten aufgezeichnet und später erneut gesendet werden.

Bei CRAM-MD5 oder DIGEST-MD5 zum Beispiel ist die Aufgabe der (iterierte) Hash des Passworts. Die Antwort wird nach dem Zufallsprinzip generiert und besteht aus dem Hashwert des Passworts und einem Wert, der vom Originalwert abhängt. Ein mögliches Szenario könnte sein:

1. Ein Server sendet eine eindeutige Aufgabe an den Client.
2. Der Client berechnet die Antwort als Hash (Aufgabe + Geheimnis) und sendet sie an den Server.
3. Der Server berechnet den erwarteten Wert der Antwort und veriﬁziert, dass er mit der Antwort des Clients übereinstimmt.

Authentifizierung

Solche verschlüsselten oder gehashten Informationen geben das Passwort selbst nicht preis, können aber genug Informationen liefern, um es mit einem Wörterbuchangriff (oder einer Rainbow-Table) zu ermitteln, d. h. durch das Ausprobieren vieler Werte. In jeden Austausch geht eine zufällig generierte Information ein, ein Salt, wie z. B. die aktuelle Zeit.

Nonce, Salt und IV (Initialisierungsvektor) sind Zahlen, die (in der Regel) nach dem Zufallsprinzip generiert, offengelegt und einmalig in einem kryptografischen Prozess verwendet werden, um dessen Sicherheit zu verbessern, indem sie ihn einzigartig machen. Eine Nonce ist eine Zahl, die ein einziges Mal in einem kryptografischen Protokoll verwendet wird, um den Austausch eindeutig zu machen. Ein Salt wird als zusätzliche Eingabe für eine Hashfunktion verwendet, um die Eingabe einzigartig zu machen (so dass dieselbe ursprüngliche Eingabe, die gehasht wird, eine andere Ausgabe ergibt). Ein Initialisierungsvektor ist eine Zahl, die als zusätzliche Eingabe für eine Verschlüsselung (einer Blockchiffre) verwendet wird, um deren Eingabe einzigartig zu machen, so dass dieselbe ursprüngliche Eingabe, die mit demselben Schlüssel verschlüsselt wird, eine unterschiedliche Ausgabe ergibt.

Challenge-Response-Protokolle, wie sie im Folgenden vorgestellt werden, werden z. B. in objektrelationalen Datenbanken wie PostgreSQL oder E-Mail-Clients wie Mozilla Thunderbird verwendet.

###### Digest-MD5

Digest-MD5 war ein gängiges Challenge-Response-Protokoll, das die in RFC2831 spezifizierte MD5-Hashfunktion verwendete. Es basierte auf der HTTP-Digest-Authentifizierung (wie in RFC2617 spezifiziert) und wurde von Melnikov (2011) obsolet gemacht.

###### Challenge-Response Authentication Mechanism (CRAM)

CRAM-MD5 ist ein Challenge-Response-Protokoll, das auf HMAC-MD5 basiert und zur Authentifizierung von Nachrichten die MD5-Hashfunktion verwendet (Krawczyk et al., 1997). Der RFC-Entwurf von Zeilenga (2008) empfiehlt, es durch CRAM zu ersetzen. Die CRAM-Schritte sind (1) der Server sendet dem Client eine Nonce, (2) der Client soll mit HMAC ( Geheimnis, Nonce) antworten, und (3) der Server berechnet erneut HMAC ( Geheimnis, Nonce) und prüft, ob es mit der Antwort des Clients übereinstimmt, um sich davon zu überzeugen, dass der Client das Geheimnis kannte.

Zu den Schwächen von CRAM gehört, dass keine gegenseitige Authentifizierung stattfindet, d.h. die Identität des Servers wird nicht veriﬁziert, und dass die verwendete Hashfunktion MD5 schnell zu berechnen ist und somit Wörterbuchangriffe erleichtert. Stattdessen ist das Key-Stretching, d. h. die Verwendung einer Hashfunktion, die absichtlich rechenintensiv ist, vorzuziehen. Die letzte Schwachstelle ist die Passwortspeicherung, bei der manche Implementierungen das unverschlüsselte Passwort des Benutzers speichern, während andere (wie Dovecot) einen vorläufigen Hashwert des Passworts speichern. Dies verhindert zwar die Speicherung des unverschlüsselten Passworts, aber für die Authentifizierung mit CRAM-MD5 ist die Kenntnis des Hashwerts gleichbedeutend mit der Kenntnis des Passworts selbst.

Salt

Eine (zufällig) generierte Zahl, die zur Eingabe einer Hashfunktion hinzugefügt wird, um die Ausgabe einzigartig zu machen, insbesondere wenn es sich bei der zusätzlichen Eingabe um geheime Informationen wie ein Passwort handelt.

###### Salted Challenge-Response Authentication Mechanism (SCRAM)

Salted Challenge-Response Authentication Mechanism (SCRAM) ist ein Challenge-Response-Protokoll zur gegenseitigen Authentifizierung (Menon-Sen et al., 2010), das CRAM-MD5 (Zeilenga, 2008) ablöst.

Während bei CRAM das Passwort des Clients als Hash auf dem Server gespeichert wird, reicht nun die Kenntnis des Hashes (anstelle des Passworts) aus, um den Client bei weiteren Authentifizierungen zu imitieren. Die Verantwortung wurde somit einfach vom Schutz des Passworts auf den entsprechenden Hash verlagert. SCRAM verhindert dies, indem es zusätzlich zu den auf dem Server gespeicherten Authentifizierungsinformationen (StoredKey), die ursprünglich aus dem Passwort des Clients (ClientKey) abgeleitet wurden, weitere Informationen verlangt.

Die Vorteile von SCRAM im Vergleich zu älteren Challenge-Response-Protokollen sind, dass die auf dem Server gespeicherten Authentifizierungsinformationen nicht ausreichen, um den Client zu imitieren (Menon-Sen et al., 2010). Insbesondere wird (a) ein Wörterbuchangriff (Rainbow-Tables) nach einem Leck in der Authentifizierungsdatenbank durch das Salt verhindert, (b) der Server kann sich gegenüber anderen Servern nicht als der Client ausgeben, da er nur teilweise Authentifizierungsinformationen speichert, und (c) die Wiederverwendung von Passwörtern nach einem Datenleck wird durch die Bindung des Hashwertes an einen einzigen Server verhindert. Nur die mit Salt versehene und gehashte Version eines Passworts wird bei der Anmeldung verwendet, und das Salt auf dem Server ist unveränderlich.

SCRAM unterstützt auch die gegenseitige Authentifizierung (durch den Client und den Server).

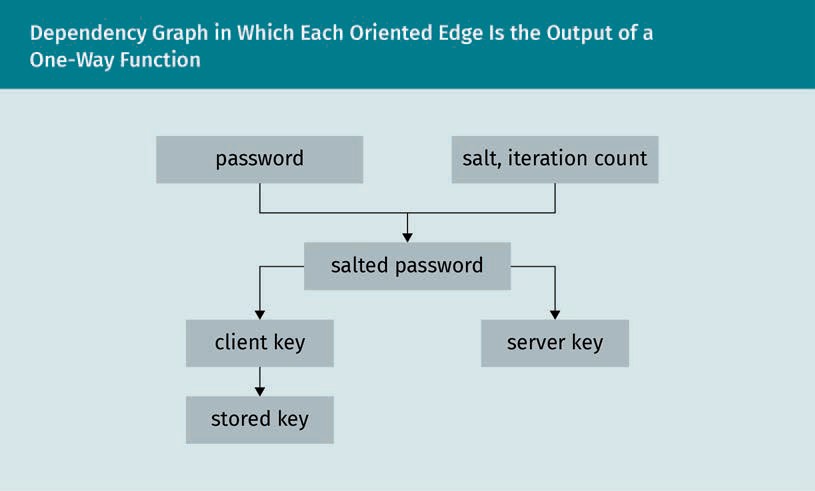
Erstellung, Übertragung und Speicherung von Schlüsseln

Wenn der Client ein Passwort erstellt, speichert der Server die abgeleiteten Schlüssel StoredKey und ServerKey zusammen mit den Parametern, die für ihre Ableitung verwendet wurden, wie folgt.

* 1. Der Client berechnet SaltedPassword, indem er die Passwort-Hashfunktion PBKDF2 (standardmäßig ist es PBKDF2, aber mittlerweile empfiehlt sich bcrypt) mit IterationCount viele Male auf die Eingabe anwendet. Die Eingabe besteht aus dem Passwort und dem Salt, d. h. SaltedPassword := PBKDF2( Passwort, Salt, IterationCount)
  2. Der Client berechnet ClientKey bzw. ServerKey durch Anwendung der HMAC-Funktion auf SaltedPassword mit den öffentlichen konstanten Strings „Client Key“ bzw. „Server Key“, d. h. ServerKey := HMAC(SaltedPassword, „Server Key“) und ClientKey := HMAC(SaltedPassword, „Client Key“)
  3. Der Client berechnet StoredKey durch Hashing von ClientKey, d. h. StoredKey := H(ClientKey) und sendet ServerKey und StoredKey an den Server (aber nicht ClientKey).

Der Server speichert StoredKey, ServerKey, Salt und IterationCount, um später Beweise von Clients zu prüfen und Beweise an Clients auszugeben. ClientKey wird zuerst verwendet, um den Client gegenüber dem Server zu authentifizieren, und ServerKey wird später vom Server verwendet, um sich gegenüber dem Client zu authentifizieren.

Authentifizierung



Der Server speichert nur den öffentlichen Teil der Wurzel (Salt und IterationCount) und die Blätter (StoredKey und ServerKey) dieses Baums, d. h. das Passwort wird nie an den Server gesendet. Es werden nur die folgenden Daten gesendet: Salt, Iterationsanzahl, ServerKey und StoredKey, d. h. HMAC(SaltedPassword, „Server Key“) und H(HMAC(SaltedPassword, „Client Key“)).

Nach einem Einbruch in die Datenbank, d. h. nachdem ein Angreifer einen ServerKey gestohlen hat, muss das Passwort eines Kunden nicht ersetzt werden, sondern nur das Salt und die Iterationsanzahl müssen geändert und ClientKey und ServerKey ersetzt werden.

Client-Authentifizierung gegenüber dem Server

Zur Authentifizierung durch den Server sendet der Client einen Authentifikator (der seinen Client-Namen und eine Client-Nonce enthält) an den Server. Der Server sendet dem Client ein Salt (salt), eine Iterationsanzahl (ic) und eine Server-Nonce (server-nonce). Daher kennen sowohl der Client als auch der Server AuthMessage := client-name, client-nonce, salt, ic, server-nonce. Der Client beweist, dass er den StoredKey kennt, indem er Folgendes berechnet:

1. ClientSignature := HMAC(StoredKey, AuthMessage) ClientProof := ClientKey ⊕ ClientSignature

Der Server stellt den ClientKey wieder her, indem er die ClientSignature berechnet (indem er den StoredKey aus dem Speicher und die AuthMessage aus diesem Austausch kennt) und den ClientKey' aus dem One-Time-Pad ClientProof „entschlüsselt“, indem er ClientKey' = ClientProof ⊕ ClientSignature berechnet.

Er berechnet StoredKey' durch H(ClientKey) und überprüft, ob der berechnete StoredKey' mit dem gespeicherten StoredKey übereinstimmt. Ist dies der Fall, wird der Client erfolgreich authentifiziert.

Wenn nur die ClientSignature gesendet würde, könnte sich ein Angreifer, der den StoredKey kennt, als der Client ausgeben. Stattdessen verlangt ClientProof zusätzlich, dass der Client den ClientKey kennt. Daher sollte der auf dem Server berechnete Wert von ClientKey' nach der Verifizierung sofort und unwiderruflich gelöscht werden.

Vorbehalt

Wenn ein Angreifer den StoredKey vom Server sowie die AuthMessage und den ClientProof aus einem Authentifizierungsaustausch kennt, kann er die ClientSignature und damit den ClientKey berechnen und sich gegenüber dem Server als Client ausgeben.

###### Zero-Knowledge-Beweise

Ein Zero-Knowledge-Protokoll zeigt, wie ein Beweiser die Kenntnis eines Geheimnisses gegenüber einem Verifizierer beweisen kann, so dass keine anderen Informationen als die Gültigkeit des Beweises (gegenüber jedermann, einschließlich dem Verifizierer) offengelegt werden.

Beweis ist hier probabilistisch zu verstehen, d. h. die Wahrscheinlichkeit, dass die Behauptung des Beweisers wahr ist, ist über jeden vernünftigen Zweifel erhaben. Da die Beweise unabhängig voneinander sind, kann die Wahrscheinlichkeit durch eine Erhöhung ihrer Anzahl beliebig gesteigert werden. Die Unmöglichkeit, aus diesem Austausch Informationen über das Geheimnis zu gewinnen, hängt mit der rechnerischen Schwierigkeit zusammen, ein mathematisches Problem zu lösen.

Diese Eigenschaft hat zwei Vorteile. Erstens kann der Veriﬁzierende keine Informationen erhalten, wenn er sich nicht an das Protokoll hält. Jeder Beweis ist unabhängig von den anderen. Zweitens kann sich der Verifizierer gegenüber einem Dritten nicht als der Beweiser ausgeben. Eine Aufzeichnung des Beweises hilft nicht dabei, einen Dritten zu überzeugen, denn die Sequenz hätte im Voraus gegenseitig abgesprochen werden können.

Vergleich mit klassischen Protokollen

Während die Behauptung, das Geheimnis zu kennen, allein nicht überzeugend ist, kompromittiert das Durchsickern von Informationen bei klassischen Protokollen, bei denen ein Beweiser (B) die Kenntnis eines Geheimnisses gegenüber dem Verifizierer (V) beweist, immer noch das Geheimnis. Wenn B sein Passwort an V übermittelt, dann erhält V und jeder, der diese Übertragung abgehört hat, alle Daten, um sich ab diesem Moment als B auszugeben. Bei einem Challenge-Response-Protokoll wird für jede Anwendung des Protokolls eine neue Aufgabe verwendet. Bei jeder Anwendung können V und ein Mithörender neue Informationen über das Geheimnis sammeln und es schließlich knacken. Wenn es sich bei der Aufgabe beispielsweise um die Verschlüsselung eines Klartextes handelt und der Angreifer diesen Klartext wählen kann, dann handelt es sich um einen Angriff mit gewähltem Klartext.

Öffentlicher Schlüssel

Bei der Abwägung der Vorteile und Nachteile zwischen einem Zero-Knowledge-Beweis und einer digitalen Signatur mit einem privaten Schlüssel (veriﬁziert durch den entsprechenden öffentlichen Schlüssel) stellen wir fest, dass dass wie bei jedem Challenge-Response-Protokoll jede Signatur, d. h. die Verschlüsselung eines Dokuments mit dem privaten Schlüssel des Benutzers, Informationen preisgibt. Im Extremfall, wenn der Angreifer diesen Klartext wählen kann, handelt es sich um einen Angriff mit gewähltem Klartext. Allerdings erfordern Zero-Knowledge-Protokolle

Authentifizierung

weniger Rechenaufwand als Public-Key-Protokolle. Da digitale Signaturen praktisch sicher sind und viele Geräte (z. B. Chipkarten) nur über eine geringe Rechenleistung verfügen, werden digitale Signaturen immer häufiger verwendet.

Die Sicherheit der meisten Zero-Knowledge-Protokolle und Public-Key-Protokolle hängt von der unbewiesenen Annahme ab, dass die Kryptoanalyse rechnerisch so schwierig ist wie ein mathematisches Problem (wie das Quadratische-Reste-Problem, die Zerlegung einer ganzen Zahl in ihre Primfaktoren und der diskrete Logarithmus).

Ali Babas Höhle

Ali Babas Höhle veranschaulicht die Prinzipien hinter einem Zero-Knowledge-Beweis. In einer kreisförmigen Höhle gibt es eine vom Eingang aus unsichtbare Tür, die den Durchgang versperrt, wenn sie nicht mit einem Passwort (z. B. „Sesam“) geöffnet wird. Damit B gegenüber V beweisen kann, dass er das Passwort kennt, ohne es preiszugeben, betritt C die Höhle unbeobachtet von V, bis er links oder rechts vor der Tür steht. V verlangt nun von B, dass er von links oder rechts kommend zum Eingang zurückkehrt. Da die Wahrscheinlichkeit, dass B die Höhle auf der gleichen Seite betreten hat, von der er – durch V aufgefordert – auch wieder zum Eingang zurückkehren soll, 1/2 beträgt und alle Beweise unabhängig sind, beträgt die Wahrscheinlichkeit, dass C das Passwort nicht kennt, nach 10 Beweisen 2–10, also weniger als ein Tausendstel. Während V überzeugt ist, dass B das Passwort kennt, kann er niemanden sonst überzeugen. Selbst wenn V die Sequenz aufgezeichnet hat, könnte sie im Voraus gegenseitig abgesprochen worden sein.

Schnorr-Identifikation

Schnorr stellte ein Zero-Knowledge-Protokoll vor, das so einfach ist, dass es auf Chipkarten ausgeführt werden kann (Schnorr, 1991). Hierbei wird die Kenntnis von diskreten Logarithmen bewiesen, d. h. der Beweiser B kennt eine ganze Zahl x, und der Verifizierer V kennt gx mod p, wobei p eine Primzahl ist. Damit B beweisen kann, dass er x kennt, ohne es preiszugeben:

1. wählt B eine beliebige ganze Zahl a und sendet ga an V.
2. V wirft eine Münze und sendet das Ergebnis c in {0, 1} an B.
3. B sendet a an V, immer wenn c = 0 ist, oder a + x an V, immer wenn c = 1 ist.

Dies ist ein Zero-Knowledge-Protokoll, denn

* wenn c = 0 ist, dann wird nichts über x preisgegeben (sondern nur über a),
* wenn c = 1 ist, dann erfährt der Veriﬁzierer a + x mod p, aber wiederum, solange nichts über a offenbart wird (wobei wir mit der Schwierigkeit der Berechnung von log mod p rechnen), wird auch nichts über x offenbart.

Wenn c = 0, dann wird keine Kenntnis von x benötigt. Wenn jedoch c = 1 ist, kann V durch ga+x = ga gx – wobei beide Werte auf der rechten Seite bekannt sind – verifizieren, ob B a + x kennt. Da die Wahrscheinlichkeit, dass c = 1 ist 1/2 beträgt und alle Beweise unabhängig sind, beträgt nach, sagen wir, 10 Beweisen die Wahrscheinlichkeit, dass V x nicht kennt, 2–10 < 1/1000.

### Biometrische Authentifizierung

Die biometrische Authentifizierung identiﬁziert den Benutzer eines Computers entweder durch physische oder verhaltensbezogene menschliche Merkmale. Zu den physischen Merkmalen gehören Fingerabdrücke (die schon lange in diesem Kontext zum Einsatz kommen), Gesichtsmerkmale wie die relative Position der Augen, der Nase, der Lippen, des Kiefers und Augenmerkmale wie das Muster der Blutgefäße in der Iris. Zu den Verhaltensmerkmalen gehören Aspekte wie die Art und Weise des Tippens, z.B. die Geschwindigkeit der Tastenanschläge oder das Auftreten von Tippfehlern (besonders nützlich zur Ergänzung eines Login-Dialogs); die Art der Handschrift, entweder statisch, wenn ein Bild verwendet wird, oder dynamisch, wenn die Zeichen auf einem Tablet anhand von (zeitlichen) Funktionen wie x- und y-Koordinaten, Druck und Neigung ausgewertet werden; und Stimmeigenschaften (die Sprechererkennung ist besonders nützlich, um die Identität von Kunden am Telefon zu überprüfen). Jedes gesprochene Wort wird in seine Formanten, die dominanten Frequenzen, zerlegt, und dann werden physiologische und verhaltensbezogene Merkmale identifiziert, welche die physiologischen Merkmale (die die Form des Stimmtrakts der Person beschreiben, d.h. der Nase, des Mundes, des Kiefers, der Zunge) oder die verhaltensbezogenen Merkmale (die die Bewegung der Nase, des Mundes, des Kiefers, der Zunge beschreiben, die den Akzent, den Ton, die Tonhöhe oder das Tempo verändern) beschreiben.

Die Authentifizierung durch das, was man ist (z. B. ein Fingerabdruck) und was man weiß (z. B. ein Passwort), hat den Vorteil, dass man, selbst wenn man es weiß, einen gewissen Aufwand betreiben muss, um es zu wiederholen, und dass man es nicht vergessen kann. Gegenüber der Authentifizierung mit dem, was man hat (z. B. ein Token), hat dies den Vorteil, dass man nichts mit sich herumtragen muss und nichts verloren oder gestohlen werden kann.

Diese Art der Authentifizierung erfordert ausgeklügelte Hardware, die offen liegt und somit imitierbar ist. Der Chaos Computer Club (CCC) und seine Mitstreiter haben zum Beispiel wiederholt gezeigt, wie einfach es ist, Fingerabdrücke zu fälschen, zum Beispiel mit Holzleim und sprühbarem Graphen. Die vielen erfolgreichen Fälschungsangriffe auf biometrische Identifizierungen, wie z.B. Fingerabdrücke und Foto-IDs, bringen uns zu dem Schluss, sie als Ergänzung und nicht als Ersatz für ein Passwort, eine Chipkarte, einen zweiten Faktor oder einfach als Ersatz für einen Benutzer zu behandeln.

### Authentifizierung in einem verteilten System

Wenn in einem verteilten System die Identität des Systems, gegenüber dem die Authentifizierung erfolgt, garantiert ist und niemand die Kommunikation abhören kann, dann könnte das Geheimnis selbst für die Authentifizierung sicher übermittelt werden. Um sicherzustellen, dass niemand außer dem beabsichtigten Empfänger das Geheimnis für die Authentifizierung sieht, darf das Geheimnis selbst niemals gesendet werden. Stattdessen überzeugen sich der Benutzer und das System gegenseitig davon, dass sie das gemeinsame Geheimnis (normalerweise ein Passwort) kennen. Die Identifizierungsdaten selbst werden also nie gesendet, sondern nur der Nachweis, dass der Benutzer Zugang zu ihnen hat. In der Praxis ist bei passwortbasierten Systemen der beliebteste Ansatz das Challenge-Response-System.

Auch wenn das Geheimnis selbst nie gesendet wird, ist es empfehlenswert, die gesamte Kommunikation zu verschlüsseln, um die Authentifizierung zu gewährleisten, z. B. durch Verschlüsselung mit einem öffentlichen Schlüssel. Andere Ansätze umfassen die Preisgabe eines Teilgeheimnisses, z.B. werden in einem Einwegsystem die

Authentifizierung

Identifikationsdaten nur einmal verwendet (TANs beim Online-Banking). Wenn die Identifizierungsdaten jedoch abgehört werden können und ihre Verwendung zur Authentifizierung und damit zur Ungültigmachung verhindert werden kann (z. B. durch Einloggen in eine gefälschte Kopie der Website der Bank), dann können sie später verwendet werden. Ein anderer Ansatz besteht darin, zusätzliche geheime Informationen über einen zweiten Kanal zu versenden, z. B. durch den Versand einer SMS im mobilen TAN-System (mTAN).

###### FIDO 2

Die FIDO (Fast IDentity Online) Alliance wurde im Februar 2013 gegründet, um in Zusammenarbeit mit vielen Unternehmen wie Google oder Microsoft offene und lizenzfreie Industriestandards für die Authentifizierung im Internet zu entwickeln. Am 9. Dezember 2014 wurde der erste FIDO-Standard veröffentlicht, der den Standard U2F (Universal Second Factor) für Hardware und Software zur Zwei-Faktor-Authentifizierung und den Standard UAF (Universal Authentication Framework) für das zugehörige Netzwerkprotokoll zur passwortlosen Authentifizierung spezifiziert. Diese Standards zielen darauf ab, die Authentifizierung im Internet zu erleichtern, indem sie das Eigentum eines Benutzers (was er hat) – wie z. B. Sicherheitstokens –, oder Eigenschaften (was er ist) – wie z. B. einen Fingerabdruck –, anstelle von Wissen (was er weiß) – wie z. B. Passwörter oder persönliche Identifizierungsnummern – akzeptieren. So muss sich ein Benutzer nicht mehr zahlreiche sichere Passwörter merken, was allerdings auch einige Nachteile mit sich bringt. Im Vergleich zu früheren Methoden der Zwei-Faktor-Authentifizierung, wie z. B. SMS-Bestätigungscodes, setzt FIDO2 voraus, dass sich der Schlüssel, z. B. das Smartphone, physisch in der Nähe des Computers befindet.

FIDO2 (2020) setzt sich zusammen aus dem

1. W3C Web Authentication Standard (WebAuthn), der den Zugriff auf das Internet mit biometrischen Informationen, mobilen Endgeräten oder FIDO-Sicherheitsschlüsseln ermöglicht. WebAuthn wird von verschiedenen Betriebssystemen, wie Windows 10 und Android, aber auch von Browsern, wie Google Chrome, Mozilla Firefox, Microsoft Edge oder Apple Safari unterstützt.
2. Client to Authenticator Protocol (CTAP) der FIDO Alliance, das auf U2F basiert. Mit der Veröffentlichung von FIDO2 wurde U2F in CTAP1 umbenannt. CTAP wird unter anderem für die Authentifizierung in Desktop-Anwendungen und Webdiensten verwendet.

Persönliche Daten und private Schlüssel befinden sich immer und ausschließlich in den Händen des Benutzers und werden nicht auf öffentlichen Servern gespeichert. Um ein Konto über FIDO2 zu registrieren, senden die Server eine Anfrage, und der FIDO2-Schlüssel generiert einen öffentlichen und einen geheimen Schlüssel aus einem geheimen Anfangsschlüssel und der Serveradresse. Dieser öffentliche Schlüssel wird an den Server übertragen, der ihn speichert und den FIDO2-Schlüssel in Zukunft eindeutig identifizieren kann.

Auf diese Weise kann sich der FIDO2-Schlüssel bei jedem Server mit einem individuellen Schlüssel identifizieren, ohne dass der Server Informationen über die Schlüsselpaare für denselben FIDO2-Schlüssel auf anderen Servern erhält. Der FIDO2-Schlüssel erzeugt für jeden Dienst ein eigenes Schlüsselpaar. Anhand der Domain der anderen Partei, z. B. Ebay oder Google, können sie nicht feststellen, welche ihrer Benutzer denselben FIDO2-Schlüssel haben. In der Praxis ist dies ein Vorteil (auf der Serverseite!) der Authentifizierung durch ein Passwort, wenn ein Benutzer häufig ähnliche Passwörter auf verschiedenen Servern verwendet.

Schlüssel

Ein FIDO2-Schlüssel (oder Authentifikator oder Token) ist die Vorrichtung, mit der man sich bei einem Dienst authentifizieren kann. Dabei kann es sich entweder um ein externes Gerät handeln, das über USB, NFC oder Bluetooth mit dem PC oder Smartphone des Benutzers verbunden wird, wie z.B. ein Sicherheitstoken zum Einstecken in einen USB-Anschluss, eine Chipkarte zum Einstecken in einen Kartenleser, ein NFC-Token, Smartphones mit Bluetooth-Schnittstelle IEEE 802.15.1 und Bluetooth-Token (Bluetooth V4.0 Low Energy 2,45 GHz). Es kann aber auch ein interner Authentifikator sein, d.h. eine Software, die den Kryptochip des PCs, Smartphones oder Tablets für FIDO2 nutzt, unterstützt von Windows 10, Android 7 und höher. Zum Schutz vor Missbrauch des FIDO2-Schlüssels kann dieser zusätzlich biometrisch oder mit einem Passwort/einer PIN gesichert werden. Wenn der Stick verloren geht, steht entweder ein registrierter Backup-Schlüssel zur Verfügung oder man muss sich über die Handynummer in Kombination mit einer E-Mail-Adresse o.ä. erneut identifizieren.

Adaption

Ein FIDO2-Schlüssel kann entweder anstelle eines Passworts oder zusätzlich zu diesem als zweiter Faktor verwendet werden. Je nachdem, wie ein Dienst FIDO2 implementiert hat, reicht der Schlüssel für die Anmeldung (Ein-Faktor-Authentifizierung) oder die Eingabe eines zusätzlichen Passworts (Zwei-Faktor-Authentifizierung) aus. Die Ein-Faktor-Authentifizierung mit FIDO2 ist bereits für Microsoft.com, Outlook.com, Ofﬁce 365 und OneDrive im Edge-Browser verfügbar. Die Zwei-Faktor-Authentifizierung mit FIDO2 funktioniert zum Beispiel mit Google, GitHub, Dropbox und Twitter.

###### Kerberos

Kerberos ist nach Cerberus benannt, der dreiköpfigen Bestie, die das Adamantinische Eingangstor zum Hades bewacht. Cerberus überprüft die Identität von toten Seelen, und Kerberos überprüft die Identität von Benutzern über ein Netzwerk (Neuman & Ts'o, 1994). Für das Problem der Schlüsselverteilung, d. h. der geheimen Weitergabe desselben Schlüssels an alle Kommunikationspartner, und für die vielen Schlüssel, die eine Gruppe von Kommunikationspartnern benötigt, um sicher miteinander zu kommunizieren, gibt es eine Lösung. Alle Kommunikationspartner müssen einer zentralen Stelle bedingungslos vertrauen. Außerdem muss jeder Benutzer in der Lage sein, für jede Kommunikation einen sicheren Sitzungsschlüssel zu erhalten, so dass jeder Kommunikationspartner nur einen Schlüssel schützen muss, während die Verantwortung für den Schutz aller Schlüssel unter den Kommunikationspartnern auf die zentrale Stelle verlagert wird.

Komponenten zur Authentifizierung und Ticketausgabe

Kerberos ist ein Netzwerkprotokoll (wie in RFC 1510-The Kerberos Network Authentication Service V5 spezifiziert), das es Benutzern ermöglicht, sich gegenseitig über ein unsicheres Netzwerk durch eine vertrauenswürdige dritte Partei, das Key Distribution Center (KDC), zu authentifizieren. Sobald ein Benutzer authentifiziert ist (Kerberos), wird er durch Zugriffsprotokolle wie LDAP (Lightweight Directory Access Protocol) autorisiert (Neuman & Ts'o, 1994).

Das KDC speichert die symmetrischen Schlüssel aller registrierten Benutzer (Client oder Server), um sie als externer Vermittler zu authentifizieren. Aufgrund seiner entscheidenden Funktion ist es eine bewährte Praxis, ein zweites KDC als Ausweichlösung zu haben.

Kerberos gruppiert Benutzer in Clients und (Dienst-)Server, die Dienste für die Clients hosten. Das Protokoll authentifiziert einen Client nur einmal, so dass alle Dienstserver ihm für den Rest seiner Sitzung vertrauen. Dies wird durch zwei Faktoren erreicht:

Authentifizierung

1. ein Ticket, ein einmalig verwendbarer Berechtigungsnachweis, der vom KDC ausgegeben wird, um einen Client gegenüber einem Server zu authentifizieren, von dem er einen Dienst anfordert, und der mit dem Schlüssel des Servers verschlüsselt ist. Es enthält die ID des Servers und des Clients, die Netzwerkadresse des Clients, einen Zeitstempel, eine Lebensdauer und einen Sitzungsschlüssel, der mit dem Schlüssel des Clients verschlüsselt ist.
2. einen Authentifikator, einen Berechtigungsnachweis, der mit dem zwischen dem Client und dem Server geteilten Sitzungsschlüssel verschlüsselt ist und der das Ticket begleitet, um den Client zu authentifizieren. Sie enthält die ID des Clients, die Netzwerkadresse des Clients und einen Zeitstempel.

Das KDC ist in einen Authentifizierungsserver (AS) und einen Ticket-Granting-Server (TGS) aufgeteilt. Der AS authentifiziert jeden Benutzer im Netzwerk. Er speichert einen symmetrischen Schlüssel für jeden Benutzer, sei es ein Client oder ein Dienstserver, der nur ihm selbst, dem AS und dem Benutzer bekannt ist. Nachdem der AS einen Kunden mit dem Schlüssel des Kunden authentifiziert hat, sendet er ihm ein Ticket-Granting-Ticket (TGT) und einen mit dem Schlüssel des Clients verschlüsselten Sitzungsschlüssel. Der TGS erzeugt nach der Authentifizierung einen Sitzungsschlüssel als Teil eines Tickets zwischen zwei Benutzern des Netzwerks. Nachdem ein Client sein TGT gesendet hat, vom TGS authentifiziert wurde und einen Dienst eines Dienstservers anfordert, sendet der TGS ein Ticket und zwei Kopien eines Sitzungsschlüssels. Der eine wird mit dem TGT-Sitzungsschlüssel des Benutzers verschlüsselt, der andere mit dem Dienstserver-Schlüssel, um den Client gegenüber dem Dienstserver zu authentifizieren und ihre Kommunikation zu sichern.

Authentifizierungsprotokoll

Damit ein Benutzer (oder Client) sicher mit einem anderen Benutzer (Dienstserver oder Anwendungsserver) über das KDC kommunizieren kann, deﬁniert das Kerberos-Protokoll zehn Nachrichten, darunter (Neuman & Ts'o, 1994)

|  |  |
| --- | --- |
| Code | Bedeutung |
| KRB\_AS\_REQ | Kerberos Authentication Service Request |
| KRB\_AS\_REP | Kerberos Authentication Service Reply |
| KRB\_TGS\_REQ | Kerberos Ticket-Granting Service Request |
| KRB\_TGS\_REP | Kerberos Ticket-Granting Service Reply |
| KRB\_AP\_REQ | Kerberos Application Request |
| KRB\_AP\_REP | Kerberos Application Reply |

Die Authentifizierung und anschließende Ticketvergabe zwischen einem Benutzer und einem Server wird durch einen Kerberos-Server vermittelt (Pixis, 2019):

1. Zum Authentifizieren des Dienstservers führt der Client folgende Schritte durch:
   1. Er authentifiziert den Authentifizierungsserver (AS) (KRB\_AS\_REQ) unter Verwendung eines dauerhaften gemeinsamen Geheimnisses (Schlüssel des Clients), und
   2. er erhält vom Authentifizierungsserver ein kurzzeitiges gemeinsames Geheimnis (Sitzungsschlüssel) und ein (Ticket-Granting) Ticket (KRB\_AS\_REP).
2. Zum Authentifizieren des Dienstservers über den Authentifizierungsserver führt der Client folgende Schritte durch:
   1. Er authentifiziert den Dienstserver mithilfe des TGT, und
   2. er fordert (KRB\_TGS\_REQ) ein Ticket vom TGS (KRB\_TGS\_REP) an, das einen Sitzungsschlüssel zwischen dem Client und dem Dienstserver enthält.
3. Der TGS führt folgende Schritte zum Erstellen des Tickets durch:
   1. Er erzeugt einen Client-to-Server-Sitzungsschlüssel;
   2. er verschlüsselt den Sitzungsschlüssel mithilfe des Client-Schlüssels;
   3. er verschlüsselt mithilfe des Dienstserver-Schlüssels das Ticket zwischen Client und Server, das die ID des Clients, die Netzwerkadresse des Clients, einen Zeitstempel, eine Lebensdauer und den Sitzungsschlüssel enthält; und
   4. er sendet die Ergebnisse der beiden Verschlüsselungen an den Client.
4. Der Client führt folgende Schritte durch, um sich direkt beim Dienstserver zu authentifizieren:
   1. Er entschlüsselt den Client-to-Server-Sitzungsschlüssel mit seinem eigenen Schlüssel, und
   2. er sendet folgendes an den Dienstserver (KRB\_AP\_REQ und KRB\_AP\_REP)
      * das Client-to-Server-Ticket, verschlüsselt mit dem Schlüssel des Dienstservers, und
      * einen Authentifikator, der die ID des Clients und einen Zeitstempel N enthält, verschlüsselt mit dem Client-to-Server-Sitzungsschlüssel.
5. Der Dienstserver
   1. ruft den Client-Server-Sitzungsschlüssel ab, indem er das Client-Server-Ticket mit seinem eigenen Schlüssel entschlüsselt,
   2. entschlüsselt mithilfe des Sitzungsschlüssels den Authentifikator und überprüft ihn. Wenn die Überprüfung erfolgreich ist, kann der Server dem Client vertrauen.
   3. sendet den um 1 erhöhten Zeitstempel verschlüsselt über den Sitzungsschlüssel an den Client.

Mit dem Sitzungsschlüssel kann der Client den Zeitstempel entschlüsseln, um ihn zu überprüfen. Wenn dies gelingt, ist das Vertrauen hergestellt und die Dienstanfragen an den Server können beginnen. Der Server stellt dem Client natürlich die gewünschten Dienste zur Verfügung (Pixis, 2019).

### Chipkarten

Eine Chipkarte ist wie eine Kreditkarte geformt, enthält aber einen Mikroprozessor, der Informationen sicher speichert und verarbeitet. Im Gegensatz dazu speichert eine Magnetstreifenkarte nur wenige Informationen (etwa <100 Byte) und kann diese nicht verarbeiten. Die Verarbeitung von Informationen auf einer Chipkarte ist durch Sicherheitsalgorithmen gesichert; für den Zugriff auf die Daten ist eine erfolgreiche Autorisierung erforderlich (Borst et al., 2001).

###### Komponenten

Der Großteil der Verarbeitung in Chipkarten ist kryptografischen Verfahren gewidmet, insbesondere der Verschlüsselung zwischen On-Chip-Komponenten. Für die Verwendung benötigt eine Chipkarte eine externe Stromversorgung und ein Avvclock-Signal, das durch den Kontakt mit einem Chipkartenleser bereitgestellt wird (den die Authentifizierung normalerweise auch beschreiben kann).

Authentifizierung

Das Betriebssystem der meisten Chipkarten implementiert einen Standardsatz von Steuerbefehlen, wie sie in ISO 7816 oder CEN 726 standardisiert sind. Die Komponenten einer Chipkarte sind

1. Random-Access Memory (RAM), der Daten liest und schreibt, aber Informationen nur vorübergehend speichert, solange Strom vorhanden ist. Normalerweise hat eine Chipkarte 1 kByte RAM.
2. Read-Only Memory (ROM) speichert Informationen dauerhaft. Das Betriebssystem und die Verschlüsselungsalgorithmen werden gespeichert. Normalerweise hat eine Chipkarte 32 kByte ROM. Zur Verbesserung der Sicherheit ist der ROM in tieferen Schichten des Halbleiters verborgen.
3. EEPROM (Electrically Erasable Programmable Read Only Memory) speichert Informationen dauerhaft, ist aber langsam und kann nur für eine begrenzte Anzahl von Schreib-/Lesevorgängen verwendet werden (etwa 100 000 Mal). Normalerweise hat eine Chipkarte 8-128 kByte EEPROM. Zur Verbesserung der Sicherheit ist der EEPROM mit einer Metallschicht ummantelt.
4. Der Prozessor war früher ein 8-Bit-Mikrocontroller, aber es werden zunehmend leistungsfähigere 16- und 32-Bit-Chips verwendet. Um die Geschwindigkeit der Verschlüsselungsberechnungen zu erhöhen, wird häufig ein Koprozessor eingesetzt.

Es gibt einen einzigen Eingangs-/Ausgangsanschluss, der durch kleine Datenpakete namens APDUs (Application Protocol Data Units) gesteuert wird. Denn Daten fließen nur mit etwa 9600 Bits pro Sekunde und im Halbduplex-Verfahren, d. h. die Daten können entweder vom Leser zur Karte oder von der Karte zum Leser fließen, aber nicht bidirektional. Eine Chipkarte kann langsam ausgelesen werden, während sich sowohl das Lesegerät als auch die Chipkarte gegenseitig durch ein Challenge-Response-Protokoll mit einer symmetrischen Schlüsselverschlüsselung authentifizieren. Zur Verschlüsselung generiert die Karte eine Zufallszahl, gibt sie an das Lesegerät weiter, das die Zahl wiederum mit einem gemeinsamen Schlüssel verschlüsselt und an die Chipkarte zurückgibt. Schließlich vergleicht die Karte das Verschlüsselungsergebnis mit ihrer eigenen Verschlüsselung. Nach der gegenseitigen Authentifizierung wird jede ausgetauschte Nachricht durch einen Nachrichtenauthentifizierungscode (HMAC) verifiziert, der aus der Nachricht, dem Verschlüsselungscode und einer Zufallszahl berechnet wird (Borst et al., 2001).

###### SIM-Karten

Die Universal Integrated Circuit Card (UICC) oder das Universal Subscriber-Identity Module (USIM) ist für mobile Endgeräte in GSM- und UMTS-Netzen sehr beliebt. Es handelt sich um eine Chipkarte mit einer Größe von einigen hundert kB, die die Sicherheit und Integrität der persönlichen Daten gewährleistet. UICCs Subscriber Identiﬁcation Module (SIM) ist eine Anwendung, welche die Authentifizierungsinformationen speichert. Die UICC speichert nicht nur den Identifizierungsschlüssel des Mobiltelefonteilnehmers und die Teilnehmerinformationen, sondern auch dessen Einstellungen, Textnachrichten, Kontakte und zuletzt gewählte Nummern. Im Vergleich zu Kreditkarten ist ein größerer EEPROM verfügbar.

###### Vorteile

Kryptografische Schlüssel müssen nicht gespeichert werden und können beliebig komplex sein, aber kryptografische Schlüssel können nicht über die Entfernung abgerufen werden. Das Verfahren ist weniger anspruchsvoll, verwendet weniger teure Hardware und die kryptografischen Schlüssel werden sicher auf einem Gerät gespeichert. So werden Fälschungen mit Fingerabdrücken oder Venenscannern unterbunden. Auf einer Chipkarte gespeicherte Schlüssel

sind viel schwieriger zu lesen und hinterlassen weniger Spuren. Sie geben zum Beispiel nie den Schlüssel preis, sondern liefern nur die Informationen, nach denen gefragt wurde, und sie sind immun gegen Keylogger, die Tastenanschläge aufzeichnen.

### Identität und Anonymität

Anonymität stammt vom griechischen Wort für „namenlos“ (OPTED o. J.). Umgangssprachlich bedeutet es, dass die Identität einer Person unbekannt ist. Hier bedeutet es, dass ein Element (z. B. eine Person oder ein Computer) innerhalb einer Menge (z. B. einer Gruppe oder eines Netzwerks) nicht identiﬁzierbar ist (innerhalb dieser Menge). Der Schutz der eigenen Identität vor der Preisgabe ist nicht nur für jemanden notwendig, der gegen das Gesetz verstößt, z. B. bei dem Versuch, einen Computer in einem Netzwerk zu missbrauchen, sondern auch als Vorsichtsmaßnahme gegen einen möglichen Missbrauch. Entsprechende Gegenmaßnahmen sind verfügbar:

* + ein Proxy-Server zwischen dem Benutzer und dem Internet, der neben anderen Aufgaben wie dem Zwischenspeichern häufig genutzter Daten und der Beschränkung des Zugriffs auf bestimmte Benutzer auch IP-Adressen (die einen Computer in einem Netzwerk identifizieren) verbergen kann,
  + das Tor-Projekt, das den gesamten Datenverkehr des Benutzers verschlüsselt und durch viele Zwischenstationen leitet, die sich gegenseitig nicht kennen, und
  + Virtuelle private Netzwerke, wie OpenVPN oder IPsec, die den gesamten Datenverkehr des Benutzers verschlüsseln und an einen zentralen Server weiterleiten.

Diese Gegenmaßnahmen sind jedoch mit Unannehmlichkeiten verbunden, wie z. B. einer aufwändigen Einrichtung und einer langsameren Datenübertragung. Eine weniger beeinträchtigende praktische Maßnahme ist die Anwendung von Best Practices zum Schutz der eigenen Privatsphäre im World Wide Web. So können wir beispielsweise Browser-Add-ons verwenden, die Tracker herausfiltern, z.B. aus Cookies, Referrern (die URL der vorherigen Webseite, von der aus ein Link verfolgt wurde) und Anfragen an zentrale Content Delivery Networks (CDNs), wie uBlock Origin, Privacy Badger, Don't track me Google und Decentraleyes.

###### Identitätsdiebstahl

Mit dem Begriff Identität, vom lateinischen *identitas*, (*idem*, gleich und *entitas*, Wesen), bezeichnen wir die Merkmale, durch die jemand oder etwas eindeutig erkennbar ist, d.h. die keine andere Person oder Sache besitzt (OPTED, o.J.). Identitätsdiebstahl ist die Aneignung der Identität einer anderen Person, in der Regel des Namens einer anderen Person oder anderer persönlicher Informationen wie PINs, Sozialversicherungsnummern, Führerscheine oder Bankdaten, um einen Betrug zu begehen, z. B. um ein Bank- oder Kreditkartenkonto für Betrugszwecke zu eröffnen. Mit dem Aufkommen digitalisierter Datensätze und der Anonymität im Internet ist Identitätsdiebstahl zunehmend zum Problem geworden.

Beim SIM-Karten-Tausch zum Beispiel verschafft sich der Angreifer die Handynummer eines Opfers, um vorübergehend dessen Online-Identität anzunehmen. Der Angreifer verschafft sich zunächst persönliche Daten des Opfers, in der Regel den Namen, die Handynummer und die Postanschrift. Sie nutzen dann die Tatsache aus, dass Mobilfunkbetreiber ihren Kunden in der Regel eine neue SIM-Karte anbieten, z. B. nachdem das Telefon verloren gegangen ist, und übernehmen die bisherige Telefonnummer.

Authentifizierung

Der Angreifer gibt sich nun als echter Kunde aus und telefoniert mit dem Kundendienstzentrum. Tatsächlich reicht es aus, die Handynummer zu kennen, um das Passwort eines Instagram-Kontos zurückzusetzen.

Zusammenfassung

Bei der Identifikation wird einem Computer mitgeteilt, wer der Benutzer ist, in der Regel durch den Namen des Benutzers (oder des Kontos). Darauf folgt die Authentifizierung, die Verifizierung der Identität des Benutzers, d. h. die Überzeugung eines Computers, dass eine Person diejenige ist, die sie vorgibt zu sein. Um sich zu authentifizieren, weist der Benutzer seine Identität durch Informationen nach, die nur er kennt (z. B. ein Passwort), die nur er hat (Hardware wie eine Chipkarte) und die nur ihn beschreiben (z. B. biometrische Identifikationsmerkmale).

Jede Methode hat ihre jeweiligen Vor- und Nachteile. Insbesondere Passwörter, die am häufigsten verwendete Methode, müssen leicht zu merken sein, was sie wiederum in der Praxis schwächt. Der FIDO2-Standard zielt darauf ab, sie durch Hardware und biometrische Authentifizierung zu ersetzen (oder zumindest zu ergänzen).

Anstatt das Geheimnis selbst bei der Authentifizierung preiszugeben, ist es sicherer, nur dessen Kenntnis zu beweisen. In einem Challenge-Response-Authentifizierungsprotokoll erfordert die erfolgreiche Beantwortung der Aufgabe die Kenntnis der Aufgabe, wie z. B. die Ver- und Entschlüsselung von Zufallsdaten mit dem geheimen Schlüssel. Bei einem Zero-Knowledge-Protokoll können im Gegensatz zu einem Challenge-Response-Protokoll keinerlei Informationen über den geheimen Schlüssel gewonnen werden, sofern die Berechnung einer mathematischen Funktion als nicht möglich angesehen wird.

Das Kerberos-Protokoll vermittelt zwischen Benutzern und Servern über einen zentralen Server, der die symmetrischen Schlüssel aller Parteien speichert. Anstatt dass die Parteien gegenseitig die Kenntnis ihrer symmetrischen Schlüssel beweisen, wird für jeden Kommunikationspartner ein Sitzungsschlüssel von begrenzter Gültigkeit erstellt.



# Lektion 5

## Kryptoanalyse – Verschlüsselungen brechen

#### LERNZIELE

Nach der Bearbeitung dieser Lektion wissen Sie …

… gegen welche Szenarien wir uns hinsichtlich der Kryptoanalyse schützen müssen.

… wie Sie die Dauer eines Brute-Force-Angriffs abschätzen und ihn beschleunigen können.

… welche Rolle die differentielle Kryptoanalyse und die Frequenzanalyse beim Brechen von Chiffren spielen.

DL-E-DLMCSEAITSC02-U05

1. Kryptoanalyse – Verschlüsselungen brechen

### Einführung

Kryptoanalyse Das Wiederherstellen oder Erzeugen von verschlüsselten Informationen ohne Kenntnis des Schlüssels wird als Kryptoanalyse

bezeichnet.

Kryptoanalyse ist die Kunst, verborgene Schriften zu entschlüsseln, d. h. Chiffren zu brechen. Das Präﬁx „Krypto“ kommt aus dem Griechischen für „verborgen“ und „Analyse“ stammt aus dem Griechischen für „auflösen“ (OPTED, o. J.). Die Geschichte liefert zahlreiche Erfolgsgeschichten der Kryptoanalyse, unter anderem die Entschlüsselung der deutschen Rotor-Chiffriermaschine Enigma durch polnische und britische Streitkräfte im Zweiten Weltkrieg. Wir werden einige bewährte Prinzipien der Kryptoanalyse vorstellen. In der Praxis waren die Intuition des Kryptoanalytikers und seine Fähigkeit, subtile Muster im Geheimtext zu erkennen, von größter Bedeutung, aber diese sind nur schwer zu vermitteln. Heute jedoch basiert die Kryptoanalyse auf der Mathematik und wird durch den efﬁzienten Einsatz enormer Rechenleistung angewandt (Dooley, 2013, Kapitel 5).

### Frequenzanalyse

Die Kryptoanalyse von Ein-Schlüssel-Kryptosystemen beruht auf Mustern im Klartext, die sich auf den Geheimtext übertragen. Bei einer monoalphabetischen Substitutionsschiffre zum Beispiel sind die Häufigkeiten des Auftretens der Buchstaben im Klartextalphabet dieselben wie im Geheimtextalphabet. Dies lässt sich kryptoanalytisch nutzen, indem man erkennt, dass die Chiffre eine monoalphabetische Substitutionsschiffre ist und die wahrscheinlichsten Kandidaten für die Klartextbuchstaben ermittelt.

###### Lateinisches Alphabet

Eine Substitution durch eine beliebige Permutation der Buchstaben des Alphabets, wie z. B.

A B … Y Z

…

E Z … G A

besitzt

26 · 25!1 = 26! > 1026

Schlüssel, weshalb ein Brute-Force-Angriff rechnerisch undurchführbar ist. Allerdings verstößt diese Methode gegen die Ziele der Diffusion und der Konfusion. Wenn der Schlüssel (also die gegebene Permutation des Alphabets) den Buchstaben α gegen den Buchstaben β austauscht, dann ergibt sich eine mangelhafte Konfusion. Das liegt daran, dass die Substitution von β im Schlüssel nur die Substitution jedes Buchstabens β im Geheimtext impliziert. Ebenso impliziert die Substitution eines Buchstabens α im Klartext nur die Substitution des entsprechenden Buchstabens β im Geheimtext. Der Algorithmus ermöglicht statistische Angriffe auf die Häufigkeit von Buchstaben, Bigrammen (Buchstabenpaaren) und Trigrammen (Dreiergruppen von Buchstaben). Im Englischen zum Beispiel ist der häufigste Buchstabe „e“, das häufigste Bigramm ist „th“ und das häufigste Trigramm ist „the“.

Kryptoanalyse – Verschlüsselungen brechen

Wir sollten also für den Anfang folgende Substitutionen durchführen:

* der häufigste Buchstabe des Geheimtextes wir mit dem häufigsten Buchstaben der englischen Sprache (e) ersetzt,
* das häufigste Bigramm des Geheimtextes wird mit dem häufigsten Bigramm der englischen Sprache (th) ersetzt,
* das häufigste Trigramm des Geheimtextes wird mit dem häufigsten Trigramm der englischen Sprache (the) ersetzt.

Je mehr Geheimtext uns vorliegt, desto wahrscheinlicher ist es, dass diese Substitution mit dem Geheimtext übereinstimmt.

Wenn wir diese Häufigkeiten zum Beispiel auf den Geheimtext ACB ACBGA ACSBDQT anwenden, erhalten wir THE THE\*\* TH\*E\*\*\* für die noch zu entschlüsselnden Buchstaben, die mit \* gekennzeichnet sind. Durch die Beschränkungen des englischen Vokabulars und der Satzstruktur ergibt sich THE THEFT THREAPS.

###### Homophone

Ein Ansatz zum Verbergen der Buchstabenhäufigkeiten ist die Verwendung mehrerer Homophone im Verhältnis zur Häufigkeit des Buchstabens im Klartext, z. B. doppelt so viele Symbole für E wie für S, so dass jedes verschlüsselte Zeichen im Durchschnitt gleich oft im Geheimtext vorkommt. Der deutsche Mathematiker und Astronom Carl Friedrich Gauß nahm an, dass seine Einführung von Homophonen eine solche Chiffre unknackbar machte. Andere Häufigkeiten im Klartext überstehen die Verschlüsselung jedoch (teilweise), wie z. B. Digramme. „TH“ kommt am häufigsten vor – etwa 20 Mal so häufig wie „HT“ – und so weiter, was die Kryptoanalyse bei einem ausreichend langen Geheimtext immer noch erheblich erleichtert (Jagetiya & Krishna, 2020).

### Brute-Force-Angriffe

In der Praxis hängt die Sicherheit eines Kryptosystems in erster Linie von seiner Widerstandsfähigkeit gegenüber den efﬁzientesten (bekannten) Methoden der Kryptoanalyse und dem Rechenaufwand ab, der zur Überprüfung aller Schlüssel erforderlich ist, einem Brute-Force-Angriff. Wenn wir genügend Zeit und Rechenkapazität haben, wird der richtige Schlüssel irgendwann gefunden. Wenn nur der Geheimtext verfügbar ist, würde ein Brute-Force-Angriff den Geheimtext Block für Block und Schlüssel für Schlüssel entschlüsseln, um einen sinnvollen Text zu erhalten. Im Vergleich zur Ein-Schlüssel-Kryptografie, die sich auf rechnerisch schwierige mathematische Probleme verlässt (d.h. die Laufzeit wächst exponentiell mit der Bitlänge der Eingabe), ist die Kryptoanalyse der Zwei-Schlüssel-Kryptografie die der Computermathematik. Die Aufgabe besteht darin, einen Algorithmus zu finden, der die Berechnungszeit verkürzt, idealerweise einen mit polynomialer Laufzeit in der Anzahl der Eingabebits.

In der Praxis können wir jedoch für die Primfaktorzerlegung, die in RSA verwendet wird, oder den diskreten Logarithmus im Difﬁe-Hellman-Schlüsselaustausch Algorithmen finden, deren Laufzeit langsamer als exponentiell mit der Anzahl der Eingabebits (subexponentiell) wächst, aber keinen mit polynomialer Laufzeit. Das Mooresche Gesetz besagte ursprünglich, dass sich die verfügbare Rechenleistung jedes Jahr ungefähr verdoppelt. Seit seiner Einführung beträgt der Geschwindigkeitszuwachs

Homophone Unterschiedliche Chiffrierzeichen für denselben Klartextbuchstaben werden als Homophone bezeichnet.

bei der Primfaktorzerlegung etwa sechs zusätzliche Binärziffern pro Jahr. Da sich also der Rechenaufwand für jeweils drei zusätzliche Ziffern verdoppelt, vervierfacht sich der Geschwindigkeitszuwachs pro Jahr. Um die Sicherheit eines Schlüssels über das Jahr 2020 hinaus zu gewährleisten, sind derzeit mindestens 2048 Binärziffern erforderlich.

###### Schlüsselgrößen im Vergleich

Diese Tabelle zeigt einen Vergleich der Schlüsselgrößen in Bits mit einem vergleichbaren Sicherheitsniveau zwischen einem symmetrischen Algorithmus wie dem AES, einem asymmetrischen Algorithmus mit elliptischen Kurven und einem asymmetrischen Algorithmus wie Difﬁe-Hellman oder RSA (Lenstra & Verheul, 2001).

|  |  |  |
| --- | --- | --- |
| Symmetrischer Schlüssel | Asymmetrisch-elliptischer Schlüssel | Klassischer asymmetrischer Schlüssel |
| 80 | 160 | 1024 |
| 112 | 224 | 2048 |
| 128 | 256 | 3072 |
| 192 | 384 | 7680 |
| 256 | 512 | 15360 |

Die Grundlage für die Werte der Tabelle bilden die schnellsten bekannten Algorithmen zur Lösung des kryptografischen Problems. Gegeben sei ein Eingabeschlüssel mit n Bits, dann gilt:

* Bei einem symmetrischen Algorithmus wie AES besteht der schnellste Algorithmus darin, alle möglichen Schlüssel auszuprobieren, was einer Komplexität (= Anzahl der Operationen) von O(2n) entspricht.
* Für den Logarithmus über einer endlichen elliptischen Kurve ist der schnellste derzeitige Algorithmus der generische Babystep-Giantstep-Algorithmus (oder, etwas schneller, der ρ-Algorithmus von Pollard), dessen Komplexität ungefähr O(2√n) beträgt.
* Für klassische asymmetrische Algorithmen, entweder RSA oder Difﬁe-Hellman, auf einem endlichen Körper ist der schnellste Algorithmus das "allgemeine Zahlenfeldsieb", dessen Komplexität für ein großes n O(2√3n) ist.

In der Praxis beschleunigen die kleineren ECC-Schlüssel die kryptografischen Verfahren im Vergleich zu RSA und Difﬁe-Hellman um einen Faktor > 5 (zusätzlich zur Erleichterung des Austauschs zwischen Personen und zur Bandbreiteneinsparung). Allerdings hat ECC im Vergleich zu RSA auch Nachteile, z. B. hängt sein Signaturalgorithmus von einem Pseudo-Zufallszahlengenerator ab, der bei schlechter Programmierung den privaten Schlüssel preisgibt.

Kryptoanalyse – Verschlüsselungen brechen

### Rainbow-Tables

Rainbow-Tables sind eine Methode zum Brechen von Passwörtern, bei der ein Passwort-Hash mit vorberechneten Hashes der wahrscheinlichsten Passwörter verglichen wird, d.h. ein Kompromiss zwischen mehr Speicher und weniger Rechenaufwand.

Eine Rainbow-Table ist eine Tabelle mit kryptografischen Hashes der gebräuchlichsten Passwörter. Auf diese Weise werden häufigere Passwörter früher aufgedeckt. Die Generierung der Tabelle richtet sich nach der kryptografischen Hashfunktion und dem verwendeten Zeichensatz, der Länge des Passworts, der Anzahl der Tabelleneinträge und so weiter. Gängige kryptografische Hash-Algorithmen wie MD4/5 und SHA sind schnell. Sie sind daher für die Erstellung von Passwörtern ungeeignet, da sie anfällig für Brute-Force-Angriffe sind. So ist beispielsweise MD5 als kryptografische Hashfunktion auf Schnelligkeit ausgelegt und eignet sich daher gut für einen Rainbow-Table-Angriff, während Hashfunktionen wie PBKDF1, PBKDF2, bcrypt, scrypt und das neuere Argon2 so konzipiert wurden, dass sie diese Art von Angriff verhindern, indem sie absichtlich langsam (bcrypt) und/oder absichtlich speicherhungrig (scrypt) sind. Ein Rainbow-Table-Angriff kann jedoch am effektivsten verhindert werden, indem die verwendete Hashfunktion für jedes Passwort einzigartig gemacht wird. Dies wird durch die Verwendung eines Salt erreicht.

###### Lookup-Tabelle

Eine Lookup-Tabelle (LUT) ist eine Datenstruktur, die dazu dient, Laufzeitberechnungen durch einfache Array-Indizierungsoperationen zu ersetzen. In der LUT werden alle Einträge vorberechnet und anschließend lediglich nachgeschlagen, um Laufzeit zu sparen, da Array-Indizierungsoperationen im Allgemeinen schneller sind als Berechnungen (Kumar et al., 2013).

###### Meet-In-The-Middle-Angriff auf DES

Die Schlüsselgröße des symmetrischen Industriestandard-Kryptoalgorithmus DES betrug lediglich 56 Bit, so wenig, dass er schon kurz nach seiner Einführung Brute-Force-Angriffen zum Opfer fiel. Daher wurde angenommen, dass eine doppelte Verschlüsselung für zwei verschiedene Schlüssel die Schlüsselgröße effektiv auf 112 Bit verdoppeln würde. Der Meet-in-the-Middle-Angriff von Difﬁe und Hellman tauscht jedoch Speicher gegen Zeit, um den Schlüssel in nur 2n+1 Verschlüsselungen (unter Verwendung von etwa 2n gespeicherten Schlüsseln) statt der erwarteten 22n Verschlüsselungen zu finden (van Oorschot & Wiener, 1996).

Nehmen wir an, der Angreifer kennt einen Klartext P und seinen Geheimtext C, d. h.

C = EK: EK′ P ,

wobei E die Verschlüsselung mit K’ bzw. K’’ bezeichnet. Der Angreifer (1) berechnet EK(P) für alle möglichen Schlüssel K und speichert die Ergebnisse im Speicher, (2) entschlüsselt den Geheimtext, indem er DK(C) für jeden K berechnet, und (3) sucht nach Übereinstimmungen zwischen diesen beiden Mengen, deren Schlüssel wahrscheinlich mit denen übereinstimmen, die zur Verschlüsselung von P zu C verwendet wurden.

Daher war eine dreifache Verschlüsselung, 3DES, notwendig, um die Schlüsselgröße effektiv zu verdoppeln und die Entschlüsselung für die zukünftige Computerleistung abzusichern.

### Bekannte/ausgewählte Klartexte/Geheimtexte

Die asymmetrische Kryptografie verwendet mathematische Methoden – genauer gesagt modulare Arithmetik – zur Verschlüsselung. Die Sicherheit, die Schwierigkeit der Entschlüsselung, der asymmetrischen Kryptografie basiert auf mathematischen Berechnungsproblemen, die seit Jahrhunderten als schwierig gelten. Die symmetrische Kryptografie (z. B. Hashfunktionen) verwendet kunstvollere Verschlüsselungsmethoden, die darauf abzielen, die Diffusion und Konfusion zu maximieren, hauptsächlich durch iterative Substitution und Permutation. Die Sicherheit der symmetrischen Kryptografie beruht einfach darauf, dass sie über Jahre hinweg ständigen Angriffen standhält, d.h. sie ist aus praktischer Sicht zufriedenstellend, weniger jedoch im Hinblick auf das Ringen um ewige Wahrheiten.

###### Perfekte Sicherheit

Perfekte Sicherheit Dies ist die Wahrscheinlichkeit, dass ein Klartext und ein Schlüssel, die zu einem bestimmten Geheimtext führen, für alle Klartexte und alle Schlüssel gleich

sind.

Klartexte kommen in der Regel nicht mit der gleichen Wahrscheinlichkeit vor. Dies hängt zum Beispiel von der Sprache, dem Jargon oder dem verwendeten Protokoll ab. Eine Chiffre ist vollkommen sicher, wenn keiner ihrer Geheimtexte etwas über den entsprechenden Klartext verrät. Außerdem ist die Wahrscheinlichkeit für jeden Klartext (stochastisch) unabhängig von jedem Geheimtext.

Bezeichnen wir mit p einen Klartext und mit P(p) seine Wahrscheinlichkeit. In Formeln ausgedrückt, gilt für jeden Klartext p und jeden Geheimtext c, dass P(p|c) = P(p). In der Praxis bedeutet dies, dass ein Angreifer, der einen Geheimtext c abfängt, keinen Vorteil hat, d. h. die Wahrscheinlichkeit, dass er den Klartext kennt, ist die gleiche als würde er c nicht kennen.

Shannon (1949) bewies das folgende Theorem über die Bedingungen, unter denen eine Chiffre vollkommen sicher ist: Gegeben eine endliche Anzahl von Schlüsseln und Klartexten mit positiven Wahrscheinlichkeiten, d. h. P(p) > 0 für jeden Klartext p, ist die Chiffre perfekt sicher, wenn gilt:

* die Wahrscheinlichkeitsverteilung ist gleichmäßig, d. h. alle Wahrscheinlichkeiten sind gleich, und
* für jeden Klartext p und jeden Geheimtext c gibt es einen eindeutigen Schlüssel k , um c aus p zu erhalten.

Statistische Abweichungen neigen dazu, das Kryptosystem zu schwächen. Insbesondere ist es wichtig, einen zuverlässigen Zufallszahlengenerator für die Schlüssel zu verwenden.

###### One-Time-Pad

Das einzige vollkommen sichere Kryptosystem ist das One-Time-Pad. Ein solches vollkommen sicheres Kryptosystem ist jedoch unpraktisch. Für Echtzeitanwendungen, wie z. B. im Internet, wird es kaum verwendet. Das One-Time-Pad addiert (durch die XOR-Operation) jedes Bit des Klartextes t mit dem (positionsmäßig) entsprechenden Bit eines Schlüssels c , der die gleiche Länge hat, und wird nach Gebrauch verworfen, d.h. er wird nicht zur Verschlüsselung anderer Klartexte verwendet.

Kryptoanalyse – Verschlüsselungen brechen

Der Geheimtext T = (T1, T2, …) ist somit

T = t ⊕ c = t1 ⊕ c1, t2 ⊕ c2, . . . .

Diese Chiffre ist so sicher wie theoretisch möglich!

Wenn der Klartext aus einem einzigen Block t besteht, dann ist diese einfache (XOR)-Addition eines Schlüssels, des One-Time-Pads, ein sicherer Algorithmus. Es ist jedoch oft unpraktisch oder sogar fast unmöglich, einen Schlüssel zu verwenden, der so lang ist wie der Klartext, z. B. um die Kommunikation über ein Netzwerk zu verschlüsseln. Wir müssen hierbei vorher wissen, wie viele Daten übertragen werden sollen.

Es ist eine schlechte Idee ( wenngleich naheliegend), denselben Schlüssel für zwei verschiedene Blöcke zu verwenden. Wenn der Klartext zum Beispiel aus zwei Blöcken b’ und b’’ besteht, dann ergibt sich bei diesem Algorithmus die Summe (XOR)

One-Time-Pad

Bei einem One-Time-Pad ist der Schlüssel genauso lang wie der Klartext. Die Schlüssel werden Buchstabe für Buchstabe (oder Bit für Bit) addiert, um den Geheimtext zu erhalten.

b′ ⊕ c ⊕ b: ⊕ c = b′ ⊕ b:

der beiden Chiffrierblöcke b’ ⊕ c und b’’ ⊕ c ist gleich der Summe b’ ⊕ b’’ der beiden klaren Blöcke. Das liegt daran, dass die Addition XOR per Definition selbstinvers ist, d. h. x ⊕ x = 0 unabhängig davon, ob die binäre Ziffer x = 0 oder x = 1 ist!

Es kann vorkommen, dass die Chiffrierung des ersten Blocks durch ein One-Time-Pad dem zweiten Block entspricht. Leider ist der zweite Block kein guter Schlüssel, denn er ist nicht zufällig, sondern sein Inhalt ähnelt in der Regel dem des ersten Blocks, d. h. der Schlüssel ist vorhersehbar.

###### Erprobte Sicherheit

Da perfekte Sicherheit nicht realisierbar ist, wird Sicherheit nur durch die Widerstandsfähigkeit gegen bekannte Angriffe oder durch die Reduzierung der rechnerischen Schwierigkeit auf die eines als schwierig anerkannten (rechnerischen, mathematischen) Problems nachgewiesen: erprobte Sicherheit. Es gibt zwar erprobte sichere symmetrische Kryptosysteme, aber die efﬁzientesten und am weitesten verbreiteten Algorithmen, wie z.B. AES, erweisen sich nur gegen bekannte Angriffe, wie z. B. die der differentiellen oder linearen Kryptoanalyse, als resistent.

Formal erprobte Sicherheit

Die mathematischen Probleme, auf denen die Schwierigkeit der Entschlüsselung bei asymmetrischen kryptografischen Algorithmen beruht, sind alle NP-komplett, d. h. ihre Lösungen sind in polynomialer Laufzeit (in der Bitlänge der Eingabe) veriﬁzierbar und alle anderen derartigen Probleme können darauf reduziert werden. Jeder kryptografische Algorithmus (verschlüsselt oder) entschlüsselt also eine Nachricht mit dem Schlüssel in polynomialer Laufzeit in der Bitlänge des Schlüssels.

Im Gegensatz dazu benötigen alle bekannten Algorithmen zur Dechiffrierung ohne den Schlüssel exponentielle Laufzeit in Abhängigkeit von der Bitlänge des Schlüssels. Gemäß der P-NP-Vermutung gibt es keinen Algorithmus, der eine polynomiale Laufzeit (in der Bitlänge des Schlüssels) benötigt. Da die Vermutung noch nicht bewiesen oder widerlegt wurde, könnte es theoretisch polynomiale Algorithmen geben, die ohne den Schlüssel in polynomialer Laufzeit entschlüsseln. Nach jahrzehntelangen ununterbrochenen Bemühungen der Kryptoanalytiker weltweit gilt dies jedoch als unwahrscheinlich.

Das erste Beispiel für ein solches nachweislich sicheres Kryptosystem war die semantische Sicherheit von Goldwasser et al. (1982), die den Schwierigkeitsgrad der Entschlüsselung auf die Berechnung des quadratischen Restes reduziert. Gegeben x und N ein Produkt aus zwei Primzahlen, ist es schwierig zu bestimmen, ob x quadratisch modulo N (d. h., ob es y so gibt, dass x = y2 mod N oder nicht), wenn, und nur wenn, das Jacobi-Symbol für x +1 ist und die Primfaktoren von N unbekannt sind.

Das Goldwasser-Micali-Kryptosystem besteht aus:

1. Einem Algorithmus zur Schlüsselerzeugung, der den privaten Schlüssel als zwei Primzahlen p und q erzeugt, und den öffentlichen Schlüssel N = p q und eine Zahl x , die weder modulo p noch modulo q quadratisch ist (so dass das Jacobi-Symbol von x für N +1 und für ihre beiden Faktoren p und q –1 ist). Wenn zum Beispiel p, q ≡3 mod 4, dann genügt x = N–1 .
2. Einem probabilistischen Verschlüsselungsalgorithmus. Wenn m = (m1, m2, …) die Bits des Klartextes

sind, dann werden die Zahlen y1, y2, … , die nicht durch p und q teilbar sind, erzeugt, und die chiffrierte Nachricht ist M = (M1, M2, …) mit M1 = y 2 xm1, M1 = y 2 xm1, …

1

1

1. Einem deterministischen Entschlüsselungsalgorithmus. Wenn M = (M1, M2, …) die verschlüsselte Nachricht ist, dann ist m1 = 0 nur genau dann, M1 quadratisch modulo N ist, ... was durch die Kenntnis der beiden Faktoren p und q von N schnell festgestellt werden kann.

Paradox

Theoretische Sicherheit bleibt eine unzulängliche Idealisierung der Realität. Ajtai und Dwork (1997) haben zum Beispiel ein theoretisch sicheres Kryptosystem vorgestellt und bewiesen, das nur ein Jahr später gebrochen wurde. „Erprobt“ bedeutet nicht „wahr“. Ein erprobtes sicheres System ist nicht notwendigerweise wirklich sicher, da der Nachweis in einem formalen Modell erbracht wird, das bestimmte Verfahrensprinzipien, Angreifer, ein bestimmtes Sicherheitsziel und die Schwierigkeit des Problems, auf das der Nachweis reduziert wird, voraussetzt.

Zum Beispiel unterscheidet sich das implementierte Kryptosystem vom formalen Kryptosystem. Ein Teilziel ist für den Angreifer bereits ausreichend. Wenn das Sicherheitsziel zum Beispiel darin besteht, dass der Angreifer nicht den gesamten Klartext aus dem Geheimtext ableitet, dann kann es bereits ausreichen, dass er einen Teil des Klartextes ableitet. Außerdem könnte der Nachweis falsch sein! Trotz dieser Ungewissheit ist ein Nachweis der Sicherheit ein nützliches ( theoretisch zwar notwendiges, aber praktisch unzureichendes) Kriterium für die Sicherheit eines Kryptosystems.

###### Angriffsszenarien

Was bedeutet Sicherheit? Das Kriterium, dass der Angreifer den Klartext nicht aus dem Geheimtext ableiten kann, ist unzureichend, da er andere nützliche (Teil-)Informationen über den Klartext gewinnen könnte. Aber selbst die Unmöglichkeit, nützliche Informationen über den Klartext abzuleiten, ist unter bestimmten Umständen unzureichend. Wenn die Public-Key-Verschlüsselung deterministisch ist (d. h. wenn dieselbe Eingabe immer dieselbe Ausgabe liefert, wie es bei der RSA-Verschlüsselung gemäß Lehrbuch der Fall ist) und der Angreifer die Anzahl der möglichen Klartexte einschränken kann (er weiß z. B., dass der Geheimtext „ja“ oder „nein“ lautet), dann kann er alle diese möglichen Klartexte mit dem öffentlichen Schlüssel verschlüsseln und die

Kryptoanalyse – Verschlüsselungen brechen

Geheimtexte mit den verschlüsselten Texten vergleichen. Bei einem asymmetrischen Algorithmus sollten wir davon ausgehen, dass der Angreifer den öffentlichen Schlüssel kennt. Sie können also jeden beliebigen Klartext verschlüsseln und ihn mit dem Geheimtext vergleichen, d. h. sie können einen Chosen-Plaintext-Angriff (CPA) durchführen.

Im Allgemeinen werden die Angriffsszenarien danach eingeteilt, wie viel der Kryptoanalytiker über den Geheimtext weiß. Kennen sie nur den Geheimtext oder gibt es wahrscheinliche oder bekannte Paare von Geheimtexten und Klartexten? Gibt es auch einen ausgewählten Klartext oder Geheimtext? Um zum Beispiel eine monoalphabetische Chiffre zu brechen, genügt dank der Frequenzanalyse in der Regel der Geheimtext allein. Oft kennt oder errät der Kryptoanalytiker jedoch einen Teil des Klartextes, z. B. die Präambel eines Briefes (z. B. eine förmliche Begrüßung) oder ein Computerdateiformat (z. B. einen Identifikator). Und schließlich können sie entweder den Absender auffordern, einen von ihnen gewählten Klartext zu verschlüsseln oder den Empfänger, einen von ihnen gewählten Geheimtext zu entschlüsseln.

Angriff nur mit Geheimtext

Der Angreifer verfügt über die Geheimtexte mehrerer Nachrichten, die mit dem gleichen Algorithmus verschlüsselt wurden. Ihre Aufgabe ist es, so viel Klartext wie möglich wiederherzustellen oder, besser noch, die verwendeten Algorithmen und Schlüssel zu finden.

Angriff mit mutmaßlichem Klartext

Der Angreifer ist im Besitz des Geheimtextes und vermutet, dass der Klartext bestimmte Wörter oder sogar ganze Sätze enthält. Ihre Aufgabe ist es, so viele Klartexte wie möglich wiederherzustellen oder, besser noch, die Algorithmen und Schlüssel zu finden, die verwendet wurden. Enigma zum Beispiel, die kryptografische elektromechanische Rotor-Chiffriermaschine, die von den Achsenmächten im Zweiten Weltkrieg eingesetzt wurde, wurde durch die Wiederholungen der Nachrichten, die sie verschlüsselte, gebrochen: Der Wetterbericht wurde täglich gesendet und als solcher am Anfang jeder Nachricht angekündigt.

Angriff mit bekanntem Klartext

Der Angreifer verfügt über einen Geheimtext und den entsprechenden Klartext. Ihre Aufgabe ist es, den verwendeten (Algorithmus und) Schlüssel wiederherzustellen; so fällt z. B. die lineare Kryptoanalyse in dieses Szenario. Ein Angriff aus dem Jahr 2006 auf das WEP-Protokoll (Wired Equivalent Privacy) zur WLAN-Verschlüsselung nutzte beispielsweise die Vorhersagbarkeit von Teilen der verschlüsselten Nachrichten aus, nämlich die Header des 802.11-Protokolls.

Angriff mit ausgewähltem (oder adaptivem) Klartext (CPA)

Der Angreifer besitzt die Geheimtexte der Klartexte, die er frei wählen kann. Der Angreifer kann den Klartext in Abhängigkeit von dem nach jeder Dechiffrierung erhaltenen Text frei anpassen und die daraus resultierenden Änderungen im Geheimtext analysieren. Ihre Aufgabe ist es, den verwendeten Algorithmus und Schlüssel wiederherzustellen. Die differentielle Kryptoanalyse fällt in dieses Szenario.

Dies ist das minimale Angriffsszenario, auf das man bei asymmetrischer Kryptographie vorbereitet sein muss! Da der Verschlüsselungsschlüssel öffentlich ist, kann der Angreifer Nachrichten nach Belieben verschlüsseln. Wenn der Angreifer also die Anzahl der möglichen Klartexte reduzieren kann, z. B. weil er weiß, dass diese entweder „Ja“ oder „Nein“ lauten, dann kann er alle möglichen Klartexte mit dem öffentlichen Schlüssel verschlüsseln und mit dem abgefangenen Geheimtext vergleichen. Auch der klassische RSA-Algorithmus ist beispielsweise anfällig für diesen Angriff. Um sich gegen diesen CPA-Angriff zu schützen, muss daher jede Implementierung dieses Algorithmus den Klartext vor der Verschlüsselung mit Zufallsdaten auffüllen.

Angriff mit ausgewähltem (oder adaptivem) Geheimtext

Der Angreifer hat einen Geheimtext c und die Klartexte der Geheimtexte (außer von c), die er frei wählen kann. Der Angreifer kann den Klartext in Abhängigkeit von dem nach jeder Dechiffrierung erhaltenen Text frei anpassen und die daraus resultierenden Änderungen im Geheimtext analysieren. Ihre Aufgabe ist es, den verwendeten Algorithmus und Schlüssel wiederherzustellen. Der Angreifer muss zum Beispiel eine Blackbox einer Chiffriermaschine analysieren, d.h. deren innere Funktionsweise ist unbekannt.

Nur wenige praktische Angriffe fallen in dieses Szenario, aber es ist wichtig für Nachweise der Sicherheit. Wenn die Widerstandsfähigkeit gegen die Angriffe dieses Szenarios nachgewiesen werden kann, dann ist die Widerstandsfähigkeit gegen jeden realistischen Angriff auf den gewählten Geheimtext gegeben.

###### Semantische Sicherheit

Wenn ein asymmetrischer Algorithmus verwendet wird, dann sollte davon ausgegangen werden, dass der Angreifer den öffentlichen Schlüssel kennt. Sie können also jeden beliebigen Klartext verschlüsseln und ihn mit dem Geheimtext vergleichen, d. h. sie können einen Chosen-Plaintext-Angriff (CPA) durchführen.

IND-CPA-sicher „Nicht unterscheidbar bei einem Chosen-Plaintext-Angriff“ bedeutet, dass kein Angreifer eine Wahrscheinlichkeit von mehr als der Hälfte hat, zwei Geheimtexte zu unterscheiden.

Eine Chiffre ist sicher gegen IND-CPA, wenn kein Angreifer unterscheiden kann, welcher von zwei zuvor ausgewählten Klartexten dem danach erhaltenen Geheimtext entspricht. Konkret ist das Kryptosystem unter CPA nicht unterscheidbar, wenn jeder Angreifer in probabilistischer Polynomialzeit nur einen unbedeutenden „Vorteil“ gegenüber dem zufälligen Erraten hat (Abdalla et al., 2016).

Die vier Schritte des IND-CPA-Spiels mit polynomialer Laufzeitbeschränkung (in der Bitlänge des Schlüssels k) auf die Berechnungen des Angreifers (durchgeführt bei der Erstellung der beiden Klartexte, Schritt zwei, und bei der Wahl des Klartextes, der dem Geheimtext entspricht, Schritt vier).

1. Es wird ein Schlüsselpaar erstellt, ein geheimer und ein öffentlicher Schlüssel, beide mit k Bits. Der Angreifer erhält den öffentlichen Schlüssel.
2. Der Angreifer berechnet zwei Klartexte M0 und M1 von gleicher Größe.
3. Die Chiffriermaschine wählt zufällig ein Bit b ∈ {0, 1}, verschlüsselt Mb und gibt den Geheimtext an den Angreifer weiter.
4. Der Angreifer wählt ein Bit b’ ∈ {0, 1}.

Der Angreifer, der das Bit b’ im vierten Schritt zufällig wählt, liegt mit einer Wahrscheinlichkeit von ½ richtig, wobei die Wahrscheinlichkeit P(b = b’) – 1/2 nicht signiﬁkant ist, wenn der Angreifer einen nicht signiﬁkanten „Vorteil“ hat. Dies ist der Fall, wenn sie mit einer Wahrscheinlichkeit ≥1 / 2 + =(k) gewinnen, wobei = eine nicht signifikante Funktion in k ist, d. h., für jede (von Null verschiedene) Polynomfunktion p gibt es k0, so dass =(k) < 1/p(k) für alle k > k0. Eine nicht signifikante Differenz sollte eingeräumt werden, da der Angreifer seine Erfolgswahrscheinlichkeit leicht über 1/2 erhöht, indem er einen geheimen Schlüssel errät und versucht, den Geheimtext damit zu entschlüsseln. Obwohl dieses Spiel für ein asymmetrisches Kryptosystem formuliert ist, kann es an den symmetrischen Fall angepasst werden, indem die Public-Key-Chiffre durch ein kryptografisches Orakel ersetzt wird, eine Blackbox-Funktion, die den geheimen Schlüssel aufbewahrt und beliebige Klartexte auf Anfrage des Angreifers verschlüsselt.

Kryptoanalyse – Verschlüsselungen brechen

### Seitenkanalangriffe

Physikalische Komponenten, die von Seitenkanalangriffen verwendet werden, sind z. B. Messungen der Rechenzeit, des Stromverbrauchs und der elektromagnetischen oder akustischen Emissionen. Wir werden uns auf Rechenzeitangriffe (Timing Attacks) konzentrieren. Diese können aus der Ferne durchgeführt werden. Allerdings leiden die Messungen oft unter Rauschen, d. h. unter zufälligen Störungen durch Quellen wie Netzwerklatenz, Festplattenzugriffszeiten und Korrektur von Übertragungsfehlern. Für die meisten Rechenzeitangriffe muss der Angreifer die Implementierung kennen. Diese Angriffe können jedoch auch zum Reverse Engineering verwendet werden.

Wir werden uns auf das Beispiel eines Rechenzeitangriffs konzentrieren, der die Zeit für die Berechnung der Potenzen ganzer Zahlen misst. Dazu müssen wir zunächst verstehen, wie ganzzahlige Potenzen berechnet werden.

###### Exponentiation durch Quadratur

Die Exponentiation durch Quadrierung (auch bekannt als Square-and-Multiply-Algorithmus oder binäre Exponentiation) ist ein Algorithmus, der zur schnellen Berechnung großer ganzzahliger Potenzen einer Zahl durch binäre Expansion des Exponenten verwendet wird. Dies ist besonders nützlich in der modularen Arithmetik. Um bn zu berechnen, sind statt b multipliziert mit b n Mal, nur 2 • log2 n Multiplikationen nötig.

Bei einer nichtnegativen ganzen Zahl als Basis b und einem Exponenten e expandieren wir zum Berechnen von be mod M den Exponenten binär, d. h.

e = e0 + e12 + e222 + ! + es2s mit e0, e1, . . . , es ∈ 0,1 ,

und berechnen

Seitenkanalangriff Dieser Angriff nutzt Informationen über die physische Implementierung einer Chiffriermaschine.

Rechenzeitangriff

Ein Angriff, der die Laufzeit kryptografischer Verfahren misst und sie mit geschätzten Verfahren vergleicht.

1 2 22 2s

b , b ,b , . . . , b

mod M .

Aufgrund von b^{2n+1} = b^{(2n)2} = (b^{2n})2, d. h. jede Potenz ist das Quadrat der vorhergehenden (und höchstens M), ist jede Potenz, eine nach der anderen, leicht berechenbar und ergibt

e e + e 2 + e 22 + ! + e 2s

e 2 e1

22 e2

2s e

b = b 0 1 2

s = b 0 b b

! b s .

Im Nachhinein zählen nur Potenzen mit e0, e1, …, es gleich 1; die anderen können weggelassen werden.

Zum Berechnen von 35 mod 7 expandieren wir z. B.

5 = 1 + 0 · 21 + 1 · 22

und berechnen

31 = 3, 32 = 9 ≡ 2, 322 = 32 2 ≡ 22 = 4 mod 7

woraus sich folgendes ergibt

35 = 31 + 22 = 31 · 322 = 3 · 4 ≡ 5 mod 7 .

###### Anfällige Algorithmen

Für die Ausführung der binären Exponentiation sehen wir uns den Exponenten an, insbesondere die Anzahl der Bits, die gleich 1 sind. Seine Laufzeit hängt linear von dieser Zahl ab. Diese Zahl allein reicht zwar nicht aus, um einen Schlüssel zu finden, aber eine statistische Korrelationsanalyse der Exponentiationen mit verschiedenen Basen hilft bei der Ableitung des Exponenten.

Kryptoalgorithmen, die mit Exponentiation modulo einer großen Primzahl verschlüsseln sind für diesen Angriff anfällig. Dazu gehören RSA, Difﬁe-Hellman und Elgamal. Bei RSA zum Beispiel ist die Nachricht die Basis b und der Schlüssel der Exponent e. Brumley und Boneh (2005) demonstrierten einen netzwerkbasierten Rechenzeitangriff auf SSL-fähige Webserver unter Verwendung von RSA, der den privaten Schlüssel innerhalb eines Tages erfolgreich wiederherstellte. Dies führte zum weit verbreiteten Einsatz von Verschleierungstechniken, um Korrelationen zwischen Schlüssel und Verschlüsselungszeit zu verbergen.

###### Beispiel für die Exponentiation (wie in Difﬁe-Hellman verwendet)

Kochis (1996) hat im folgenden Algorithmus zur Berechnung der modularen Exponentiation aufgedeckt, d. h. zur Berechnung von R(y) = yx mod n für n öffentlich und y bekannt, aber x geheim, einen Fehler aufgedeckt. Der Angreifer kann durch die Berechnung von R(y) für mehrere Werte von y und bei Kenntnis von n, y sowie der Rechenzeit x wie folgt ableiten.

Seiw die Bitlänge von x und s\_0 = 1. Für k von 0 bis w – 1: wenn das k-te Bit von x den Wert 1 hat, dann sei R\_k = (s\_k \* y) mod n; sonst sei R\_k = s\_k. Sei s\_{k+1} = R\_k^k mod n

Ende (der For-Schleife) Rückgabe von (R\_{w-1})

Je nach dem Wert des k-ten Bits von x wird entweder (sk × y) mod n oder nichts

berechnet. Daher wird die Ausführungszeit des Algorithmus für verschiedene Werte von y schließlich den Wert des k-ten Bits ergeben.

Kryptoanalyse – Verschlüsselungen brechen

### Moderne kryptoanalytische Algorithmen

Um die Gründe für die Auswahl der einzelnen Schritte eines Blockchiffrieralgorithmus wie AES zu verstehen, muss man wissen, welche Angriffe er abwehrt. Ein leistungsfähiger moderner kryptoanalytischer Algorithmus ist die differenzielle Kryptoanalyse, die auf Blockchiffren anwendbar ist. Es setzt einen CPA voraus; der Angreifer sendet Paare von (leicht) unterschiedlichen Klartexten, deren Geheimtexte er erhält. Dann untersucht er, wie sich Unterschiede in der Eingabe (auf Differenzpfaden) durch das Netzwerk von Verschlüsselungsumwandlungen zu Unterschieden in der Ausgabe fortpflanzen. Daemen und Rijmen (1999) bewiesen die Widerstandsfähigkeit von AES gegen breite Pfade.

###### Prototypische Feistelchiffre

Wir wollen diese Technik anhand des einfachen Modells einer Feistelchiffre (Heys, 2002) demonstrieren, die den Klartext in Blöcke von 16 Bits aufteilt und jeden Block in vier Blöcke von vier Bits unterteilt.

Für jede Runde gibt es einen entsprechenden (unabhängigen) Schlüssel. In jeder der ersten drei Runden gilt:

1. Addiere den Rundenschlüssel C zum Block B / B ⊕ C.
2. Ersetze die Bits der 4 Teilblöcke entsprechend der Tabelle (in hexadezimaler Notation)

|  |  |
| --- | --- |
|  | |
| 2. | 0 |
| 3. | 1 |
| 4. | 2 |
| 5. | 3 |
| 6. | 4 |
| 7. | 5 |
| 8. | 6 |
| 9. | 7 |
| 10. | 8 |

|  |  |
| --- | --- |
| 11. | 9 |
| 12. | A |
| 13. | B |
| 14. | C |
| 15. | D |
| 16. | E |
| 17. | F |
| 18. | E |
| 19. | 4 |
| 20. | D |
| 21. | 1 |
| 22. | 2 |
| 23. | F |
| 24. | B |
| 25. | 8 |
| 26. | 3 |
| 27. | A |
| 28. | 6 |
| 29. | C |
| 30. | 5 |

Kryptoanalyse – Verschlüsselungen brechen

|  |  |
| --- | --- |
| 31. | 9 |
| 32. | 0 |
| 33. | 7 |

1. Vertausche Bit i aus dem Teilblock j mit j aus dem Teilblock i.
2. Addiere in der vorletzten vierten Runde den Rundenschlüssel zu dem Block B /B ⊕ C und ersetze jeden der vier Teilblöcke mit vier Bits durch die Tabelle.
3. Addiere in der letzten fünften Runde den Rundenschlüssel zum Block, B /B ⊕ C.

In der vierten Runde wird der letzte Schritt, die Permutation, ausgelassen, da er nur den letzten Schlüssel der fünften Runde permutieren würde. Aus kryptografischer Sicht ist das überflüssig.

In der letzten Runde werden die letzten beiden Schritte (Substitution und Permutation) ausgelassen, da der Algorithmus öffentlich ist (nach Kerckhoffs' Prinzip) und von jedem Dechiffrierer ohne Kenntnis des Schlüssels rückgängig gemacht werden kann. Aus kryptografischer Sicht sind sie überflüssig.

Die Substitutionstabelle stammt aus dem DES-Algorithmus und wird gemeinhin als Substitutionsbox (S-Box) bezeichnet.

###### Differenzielle Kryptoanalyse

Der Traum eines Kryptoanalytikers ist es, zu erfahren, ob ein Teil des gewählten Schlüssels korrekt ist, d. h. ob er mit dem entsprechenden Teil des Schlüssels übereinstimmt, der zur Verschlüsselung des Textes verwendet wurde. Bei der Heys-Chiffre hat der Schlüssel zum Beispiel eine Länge von 16 Bit. Wenn der Kryptoanalytiker herausfindet, dass die Hälfte (acht Bits) des versuchten Schlüssels mit der entsprechenden Hälfte des richtigen Schlüssels übereinstimmt, muss er nur alle möglichen Kombinationen dieser acht Bits und der restlichen acht Bits testen. Damit reduziert sich die Anzahl der zu testenden Kombinationen von 216 = 65536 auf 2 • 256 = 512.

Kriterium für die Entschlüsselung

Bei einem Brute-Force-Angriff entschlüsselt der Kryptoanalytiker den verschlüsselten Text mit jedem nur möglichen Schlüssel. Um nun festzustellen, ob der versuchte Schlüssel korrekt ist, prüft er, ob der Inhalt verständlich ist. Dazu könnten beispielsweise die Häufigkeiten der Buchstaben, Paare und Dreiergruppen des entschlüsselten Textes gezählt und mit den Häufigkeiten der vermuteten Sprache des Klartextes verglichen werden. Wenn sie nahe beieinander liegen, ist der Klartext wahrscheinlich verständlich und der versuchte Schlüssel wurde vom Verschlüsseler verwendet.

Wenn die Chiffre eine einzige Runde hat, dann ist dieses Kriterium anwendbar. Wenn die Chiffre jedoch zwei oder mehr Runden hat und der Dechiffrierer die letzte Runde des Entschlüsselungsalgorithmus mit einem bestimmten Schlüssel ausführt, dann ist dieses Kriterium nicht mehr anwendbar, da der erhaltene Text die Ausgabe des Verschlüsselungsalgorithmus (mit demselben Schlüssel) aus der vorletzten

Runde ist. Stattdessen ist das Kriterium der differenziellen Kryptoanalyse, den richtigen Schlüssel gefunden zu haben, probabilistisch. Der getestete Schlüssel ist wahrscheinlich korrekt, wenn für eine bestimmte „eingehende“ Differenz ∆X und eine bestimmte „ausgehende“ Differenz ∆Y, Klartextpaare mit Differenz ∆X mit einer bestimmten Wahrscheinlichkeit in Geheimtextpaaren mit Differenz ∆Y resultieren.

Damit die differenzielle Kryptoanalyse anwendbar ist, muss der Kryptoanalytiker in der Lage sein, eine beliebige Anzahl von frei gewählten Klartexten mit demselben Schlüssel zu verschlüsseln und die verschlüsselten Texte zu untersuchen.

Eine Differenzial ist ein Paar D = (∆X, ∆Y) aus Eingangs- bzw. Ausgangsdifferenzen ∆X

bzw. ∆Y. Die differenzielle Kryptoanalyse nutzt die hohe Wahrscheinlichkeit einer Differenz

∆X := X’ ⊕ X’’ zwischen zwei Klartexten X’ und X’’, die sich zu einer Differenz ∆Y = Y’ ⊕ Y’’ zwischen zwei Geheimtexten Y’ und Y’’ fortpflanzen (für X’ und X’’) (in der vorletzten Runde). Hier ist X’ ⊕ X’’ die bitweise XOR-Addition, d. h. ∆X zeigt alle Bits an, bei denen sich X’ und X’’ unterscheiden). Das Paar D = (∆X, ∆Y) ist das Differenzial. Die Differenz von X’ und X’’ ist ∆X := X’ ⊕ X’’*.*

Damit die differenzielle Kryptoanalyse efﬁzient ist, muss es ein Differenzial D mit hoher Wahrscheinlichkeit pD geben. Daher ist unter allen eingehenden Paaren mit der Differenz ∆X die Wahrscheinlichkeit eines ausgehenden Paares mit der Differenz ∆Y (in der vorletzten Runde) pD. Genauer gesagt, verschlüsselt der Chiffrierer eine statistisch signifikante Anzahl von Paaren (> 1/pD) von Klartexten mit der Differenz ∆X , um die Anzahl der verschlüsselten Paare von Geheimtexten mit der Differenz ∆Y zu zählen.

Häufigkeit der Differenzen für eine Substitutionstabelle

Eine afﬁne Transformation A ist die Zusammensetzung aus einer linearen Anwendung, d. h.A(x ⊕ y) = A(x) ⊕ A(y) für alle x und yund einer Translation, d. h. A(x) = x ⊕ x0 für ein vorgegebenes x0.

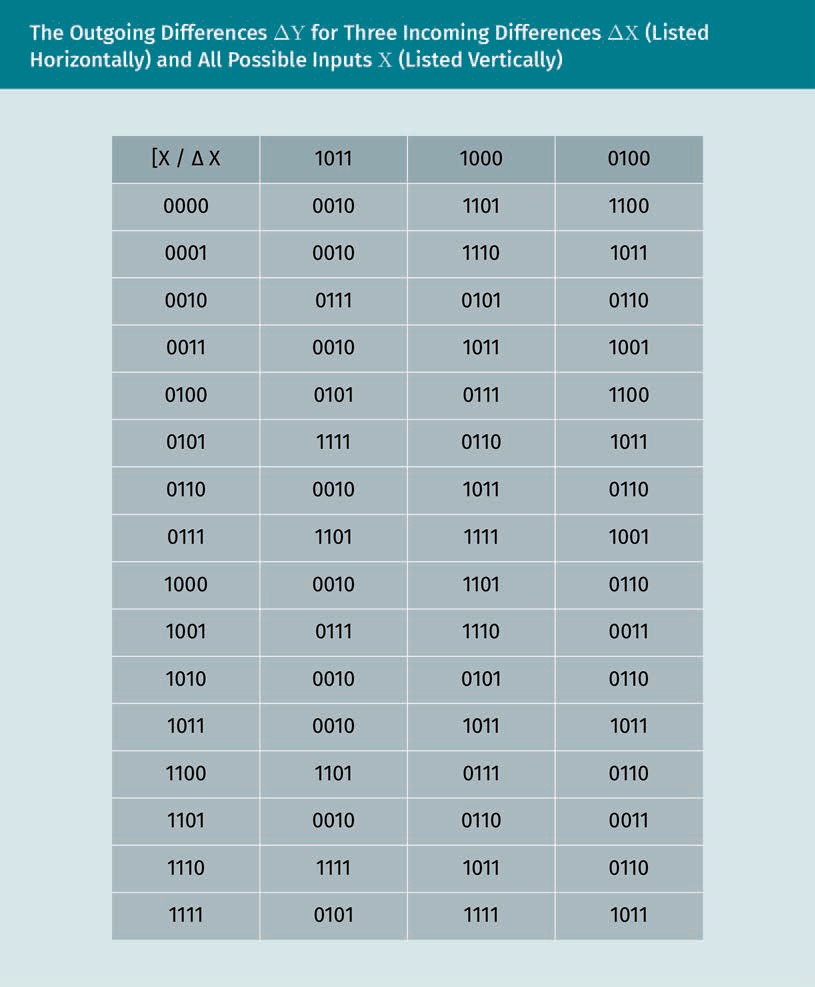
Für eine afﬁne Transformation A ist die ausgehende Differenz ∆Y unabhängig von dem eingehenden Paar X’ und X’ (sondern nur abhängig von ∆X). Die Transformation A

* sofern sie linear ist, ist stets, d. h. für jedes eingehende Paar mit der Differenz ∆X, die ausgehende Differenz ∆Y = A(∆X), und
* sofern es eine Translation ist, dann gilt immer ∆Y = ∆X.

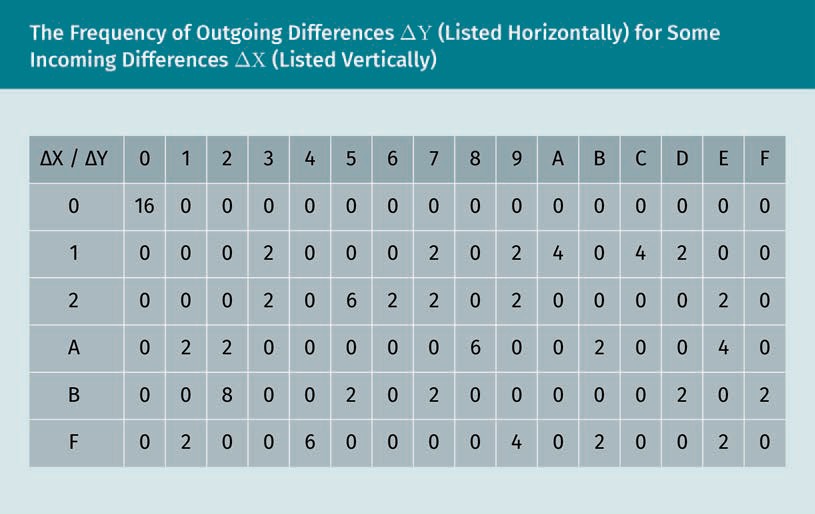
Was die erste und zweite Funktion jeder Runde einer Feistelchiffre betrifft, so ist die Addition des Schlüssels eine Translation, und die Permutation ist linear.

Daher ist die ausgehende Differenz unabhängig von dem eingehenden Paar. Die ausgehende Differenz ∆Y der Substitution wird jedoch nicht allein durch die eingehende Differenz ∆X bestimmt, sondern sie hängt von X’ und X’’ ab! Wir prüfen die Substitutionstabelle, um ein Differenzial D mit hoher Wahrscheinlichkeit pD zu ﬁnden, d. h. um eine eingehende Differenz ∆X mit einer großen Anzahl von Paaren X’ und X’’ zu ﬁnden, die eine ausgehende Differenz ∆Y ergeben. Bei ∆X gibt es 24 = 16 mögliche Eingaben X’ (die X’’ = X’ ⊕ ∆X bestimmen), und wir zählen die Häufigkeiten der 24 = 16 möglichen ausgehenden Differenzen ∆Y = 0, 1, …, F.

Kryptoanalyse – Verschlüsselungen brechen



Zählen wir für jede eingehende Differenz ∆X wie oft jede ausgehende Differenz ∆Y unter allen eingehenden Paaren X’ und X’’ auftaucht, so dass X’ ⊕ X’’ = ∆X.



Die Einträge für jede Zeile summieren sich zu 16, der Anzahl aller möglichen Paare für eine bestimmte Differenz. Die erste Zeile bestätigt, dass zwei gleiche Eingaben zu zwei gleichen Ausgaben führen. Die höchste Zahl ist 8 und wird erreicht für ∆X = B und ∆Y = 2. Außerdem taucht die Zahl 6 fünfmal auf. Wir werden unsere Differenziale unter denjenigen mit diesen hohen Häufigkeiten auswählen.

Beispiel

In der Häufigkeitstabelle

* einer Translation, wie z. B. dem Hinzufügen des geheimen Schlüssels, sind alle Felder null, außer denen in der ersten Spalte, die den Wert 16 haben.
* sind für eine lineare Operation wie eine Permutation von Bits in jeder Zeile alle Einträge null, außer einem mit dem Wert 16.

Differenzpfade

Ein Differenzpfad ist ein Tupel von Differenzen, so dass jeder Eintrag∆Ui die Eingabe der S-Box der i-ten Verschlüsselungsrunde ist. Gegeben eine Feistelchiffre und einen Differenzpfad (∆U1, ∆U2,

…), und gegeben die ausgehende Differenz ∆Vi der S-Box der Runde i, ist die eingehende Differenz der nächsten Runde ∆Ui+1 das Ergebnis der Anwendung der Permutation auf ∆Vi. (Die Addition des Schlüssels ändert als Translation nichts an der Differenz.)

Wir wollen den wahrscheinlichsten Differenzpfad D in der Heys-Chiffre ﬁnden

D = ∆U1, ∆U2, ∆U3, ∆U4

Kryptoanalyse – Verschlüsselungen brechen

oder zumindest einen Pfad, bei dem jedes Differenzial (∆Ui, ∆Vi) zu den wahrscheinlichsten gehört. Jedes Differenzial besteht aus vier Teildifferenzialen, die den vier Teilblöcken von vier Bits entsprechen, die einen Block von 16 Bit bilden. Um einen solchen wahrscheinlichen Differenzpfad zu ﬁnden, maximieren Sie in jeder Runde die Häufigkeit jedes Teildifferenzials, d. h. die Anzahl, wie oft die S-Box die eingehende Differenz (des Teildifferenzials) in die ausgehende Differenz transformiert. Minimieren Sie insbesondere die Anzahl der (aktiven) Teildifferenzen, die ungleich Null sind.

Ein Beispiel für einen solchen Pfad D sehen wir hier: Die Differenz in der ersten Runde sei

∆U1 = 0000 1011 0000 0000 ,

die durch S-Box 2 ersetzt wird durch

∆V1 = 0000 0010 0000 .

Durch die anschließende Permutation erhalten wir als Differenz beim Eintritt in die zweite Runde

∆U2 = 0000 0000 0100 0000

die durch S-Box 3 ersetzt wird durch

∆V2 = 0000 0000 0110 0000 .

Da die Anzahl der Bits 2 und 3 von Null verschieden sind, erhalten wir durch die anschließende Permutation als eingehende Differenz der dritten Runde, dass bei zwei aktiven Teildifferenzialen

∆U3 = 0000 0010 0000

die durch S-Box 2 und 3 ersetzt wird durch

∆V3 = 0000 0101 0000 .

Durch die anschließende Permutation ist die Eingabe der vierten Runde schließlich

∆U4 = 0000 0110 0000 0110 .

Bezeichnen wir mit Si,j die Substitution des Teilblocks j durch die S-Box in der i-ten Runde. Auf unserem Differenzpfad tragen wir für jede Runde i = 1,2,3 und für jedes Teildifferenzial j = 1,2,3,4 abweichend von Null die Wahrscheinlichkeit der Substitution Si,j ein, die die eingehende Differenz X (in hexadezimaler Notation) in eine ausgehende Differenz Y umwandelt:



Wenn wir annehmen, dass die Differenziale einer Runde unabhängig von den Differenzialen der vorherigen Runde sind (was eine vernachlässigbar unzutreffende Vereinfachung ist), dann ist die Wahrscheinlichkeit pD der verketteten Substitutionen, die

∆U1 = 0000 1011 0000 0000

in

∆U4 = 0000 0110 0000 0110 .

transformiert, das Produkt aus den Wahrscheinlichkeiten der einzelnen Substitutionen,

pD = 8/16 · 6/16 · 6/16 · 6/16 = 27/1024 .

Um den Schlüssel zu ﬁnden, muss der Kryptoanalytiker für jede mögliche Kombination vonK5,5, …, K5,8 und K5,13, …, K5,16 ein (ganzzahliges) Vielfaches m von 1/pD ≈ 38 Paaren von Klartexten U’1 und U’’1 mit Differenz ∆U1

* 1. Das Paar U’1 und U’’1 verschlüsseln,
  2. Die Chiffre bis zur S-Box-Eingabe in der vierten Runde mit dem Rundenschlüssel umkehren

K5 = 0000 K5,5, . . . , K5,8 0000 K5,13, . . . , K5,16

um das Paar υ’4 und υ’’4 mit Differenz υ4 zu erhalten, und

* 1. Die Differenz ∆υ4 mit ∆U4 vergleichen. Wenn sie übereinstimmen, dann erhöhen wir den Zähler n

um 1.

Wenn für eine Kombination von Teilblöcken K5,5, …, K5,8 und K5,13, …, K5,16 die Zählung n/m

≈ pD ergibt, d. h. das Verhältnis zwischen der Anzahl n der übereinstimmenden Paare und der Anzahl m der gesamten Paare liegt nah bei der Wahrscheinlichkeit pD, dann sind diese Teilblöcke wahrscheinlich die Teilblöcke 2 und 4 des von der Chiffre verwendeten Rundenschlüssels 5.

Kryptoanalyse – Verschlüsselungen brechen

Die Schlussfolgerung, dass wir die richtigen Teilblöcke gefunden haben, basiert auf den folgenden Hypothesen: (a) dass die Differenziale einer Runde unabhängig von den Differenzialen der vorherigen Runde sind und

(b) dass eine Wahrscheinlichkeit für übereinstimmende Paare, die nahe an der berechneten liegt, auf den richtigen Schlüssel hinweist.

Beide haben keine strenge mathematische Grundlage, sondern sind nur plausibel, weil jede Runde versucht, so viel wie möglich zu streuen, d. h. den Wert jedes Ausgangsbits praktisch unabhängig von allen Eingangsbits zu machen. Es ist unwahrscheinlich, dass es einen Schlüssel gibt, der anders ist, aber die gleiche Wahrscheinlichkeit reproduziert.

Zusammenfassung

Die Kryptoanalyse, die Kunst, Verschlüsselungen zu brechen, basiert derzeit auf der Mathematik und wird durch den efﬁzienten Einsatz enormer Rechenleistung angewandt.

Die Sicherheit eines Kryptosystems gegen Brute-Force-Angriffe und die empfohlenen Schlüsselgrößen hängen in erster Linie von seiner Widerstandsfähigkeit gegen die efﬁzientesten (bekannten) Methoden der Kryptoanalyse (unter Verwendung einer Hintertür) und dem Rechenaufwand ab, der erforderlich ist, um alle Schlüssel zu überprüfen (unter Verwendung der Vordertür), indem die entschlüsselte Ausgabe auf wahrscheinliche Muster eines Klartextes geprüft wird.

Wenn die geheimen Informationen als kryptografische Hashes gespeichert sind, verwendet ein Brute-Force-Angriff eine Rainbow-Table. Dies ist vielversprechend gegen schnell berechnete Hashfunktionen wie MD4/5 und kann verhindert werden, indem die verwendete Hashfunktion für jedes Passwort einzigartig gemacht und ein Salt hinzugefügt wird.

Das einzige vollkommen sichere Kryptosystem ist das One-Time-Pad, bei dem ein Schlüssel in der Größe des Klartextes zum Klartext hinzugefügt wird. Die asymmetrische Kryptografie verwendet mathematische Methoden, insbesondere die modulare Arithmetik, für die Verschlüsselung. Die symmetrische Kryptografie verwendet kunstvollere Methoden der Chiffrierung, die die Diffusion und Konfusion durch wiederholte Substitution und Permutation maximieren. Die Sicherheit symmetrischer kryptografischer Algorithmen beruht schlicht und einfach auf ihrer Widerstandsfähigkeit gegen jahrelang andauernde Angriffe.

Bei einem Seitenkanalangriff werden Informationen aus der physischen Implementierung einer Chiffriermaschine verwendet, wie z. B. Messungen der Rechenzeit, des Stromverbrauchs, der elektromagnetischen Emissionen oder der Schallemissionen. Ein Rechenzeitangriff misst die Laufzeiten von kryptografischen Verfahren und vergleicht sie mit den geschätzten Laufzeiten, z. B. nutzt man aus, dass die Laufzeit der Berechnung einer Potenz von der Anzahl der von Null verschiedenen Bits ihres Exponenten abhängt (Bei RSA und Difﬁe-Hellman ist dies der Schlüssel.).

Die differenzielle Kryptoanalyse wird auf Blockchiffren unter der Annahme eines Angriffs mit ausgewählten Klartexten angewendet. Der Angreifer sendet Paare von unterschiedlichen Klartexten, deren Geheimtexte er erhält. Dann untersucht er, wie sich Unterschiede in der Eingabe (auf Differenzpfaden) durch das Netzwerk von Verschlüsselungsumwandlungen zu Unterschieden in der Ausgabe fortpflanzen.