by Pierre Loidreau
<pierre.loidreau/at/ensta.fr>

About the author:

Pierre arbeitet als Wissenschaftler und Lehrer bei der ENSTA (Ecole Nationale Supérieure de Techniques Avancées). Er arbeitet auf dem Gebiet Kryptosysteme und Fehlerkorrekturcodes. Er "spielt" fast jeden Tag mit Linux herum und manchmal spielt er Tennis.

Eine Einführung in die Kryptographie

Abstract:

Dieser Artikel wurde zuerst in der "Security Spezialausgabe" des Linux Magazine France veröffentlicht. Der Editor, die Autoren und die Übersetzer haben freundlicherweise LinuxFocus die Erlaubnis gegeben, alle Artikel dieser Spezialausgabe zu veröffentlichen, sobald diese auf Englisch verfügbar sind. Ein großes Dankeschön geht an alle, die bei dieser Arbeit beteiligt sind. Dieser Absatz wird in jedem Artikel dieser Serie erscheinen.

Translated to English by:
Axelle Apvrille
<axellec/at/netcourrier.com>

Warum Kryptographie –– 2500 Jahre Geschichte

Die Geschichte der Kryptographie geht zurück zu den Ursprüngen der Menschheit, als Menschen anfingen miteinander zu kommunizieren. Sie hatten das Bestreben auch geheime Nachrichten auszutauschen, auf eine sichere Art. Die erste gezielte Anwendung von technischen Methoden um Nachrichten zu verschlüsseln geht auf die Griechen zurück. Sie benutzten um 6 vor Christus einen Stock namens "scytale". Der Sender einer Nachricht würde ein Papier um diesen Stock rollen und seine Nachricht der Länge nach darauf schreiben. Danach würde er das Papier wieder vom Stock abrollen und verschicken. Der Empfänger wäre nicht in der Lage die Nachricht zu lesen, ohne die genaue Dicke des Stocks zu kennen. Die Stockdicke war also der geheime Schlüssel. Später haben römische Heere Caesar's Code benutzt, um Botschaften auszutauschen. Caesar's Code verschiebt das Alphabet um drei Buchstaben.

In den nächsten 19 Jahrhunderten wurden mehr oder weniger clevere experimentelle

Verschlüsselungsverfahren entwickelt. Ihre Sicherheit hing davon ab, wieviel Vertrauen der Benutzer in dieses Verfahren hatte. Im 19. Jahrhundert schrieb Kerchoff eine der Prinzipien der modernen Kryptographie. Es besagt, daß die Sicherheit eines kryptographischen Systems nicht in dem Verfahren der Verschlüsselung liegt, sondern von der Länge des verwendeten Schlüssels abhängt.

Von diesem Augenblick an konnte man erwarten, daß kryptographische Systeme dieser Anforderung gerecht wurden. Es fehlte jedoch noch an mathematischem Hintergrund und Werkzeugen, um die Widerstandsfähigkeit gegen Angriffe zu testen. 1948 und 1949 untermauerte Claude Shannon die moderne Kryptographie mit zwei Veröffentlichungen: "A Mathematical Theory of Communication" und "The Communication Theory of Secrecy Systems". Diese zwei Artikel fegten Hoffnungen und Vorurteile gleichermaßen hinweg. Shannon bestätigte, daß Vernam's Cipher, die einige Jahre zuvor vorgeschlagen worden war, auch als One Time Pad (einmal Schlüssel) bekannt, das einzige sichere System war, das es jemals geben kann. Leider war dieses System in der Praxis unbenutzbar.... Das ist der Grund, warum die Sicherheit eines Systems heute an dem Rechenaufwand zum Entschlüsseln gemessen wird. Man behauptet, daß ein System sicher ist, wenn kein bekannter Angriff einfacher ist als das Durchprobieren aller möglichen Schlüssel.

AES (Advanced Encryption Standard)

Kürzlich, im Oktober 2000, hat das NIST (National Institute of Standards and Technology) einen neuen Standard zur Verschlüsselung unter 15 Kandidaten ausgewählt. Dieser neue Standard soll den alten DES Algorithmus ablösen. Rijndael −− ein Phantasiename aus den Namen der Erfinder, Rijmen und Daemen, wurde als AES ausgewählt.

Dieses kryptographische System wird als "block cipher" bezeichnet, da es 128−Bit Blöcke verschlüsselt. Verschiedene Optionen erlauben die Benutzung von Schlüsseln der Länge 128, 192 oder 256 Bit. Nur zu deiner Information: Der DES war eine 64 Bit "block cipher" mit einer Schüssellänge von 56 Bit. Tripple−DES verschlüsselt 64 Bit Blöcke mit einer Schüssellänge von 112−Bit.

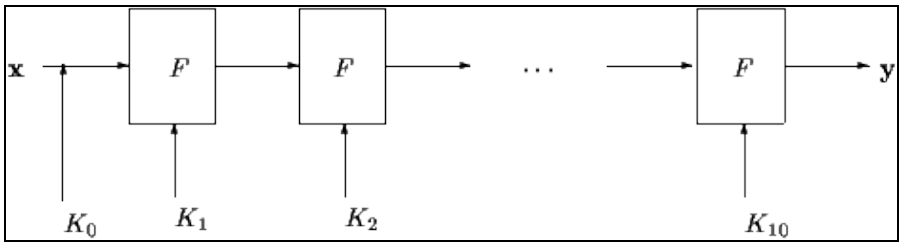

Abb. 1: AES Iterationen

Das Verfahren von AES ist in Abb. 1 beschrieben. Zunächst wird der geheime Schlüssel K0 mit der Nachricht ver−XOR−ed. Danach wir die Funktion F iterativ (wie bei allen "block ciphers") angewandt, unter Benutzung von Unterschlüsseln, die durch eine Schüsselexpansionsroutine erzeugt werden.

Für AES wird die Funktion F 10 mal angewandt.

- Abbildung 2 beschreibt wie die Funktion F zum Verschlüsseln iteriert wird. Ein 128−Bit Block wird in 16 Bytes unterteilt. Zuerst wird die Substitution S auf jedes Byte angewendet. Danach wird die Permutation P auf die 16 Bytes angewandt. Der 128−Bit Teilschüssel aus der Schlüsselexpansionsroutine wird dann Bitweise hinzuaddiert.
- Der Schlüssel Ki von Runde n°i ergibt sich aus der Schlüsselexpansion von Unterschlüssel K(i−1) aus Runde n°i−1 und K0 ist der geheime Schlüssel. Die Schüsselexpansionsroutine ist in Abbildung 3 beschrieben. Die 16 Bytes von Schlüssel K(i−1) werden in Viererblöcken bearbeitet.
  Die letzten vier Bytes werden mit der Substitution S − dieselbe Substitution, die in Funktion F benutzt wird − bearbeitet. Danach werden die 4 resultierenden Bytes zu dem Alpha−Element hinzugefügt. Dieses Element ist ein vordefiniertes Element, das von der Nummer der Iteration abhängt. Um letztlich Ki zu erhalten, werden die resultierenden 4 Bytes bitweise zu den ersten 4 Bytes von K(i−1) addiert. Danach wird das Ergebnis zu den nächsten 4 Bytes hinzugefügt u.s.w...

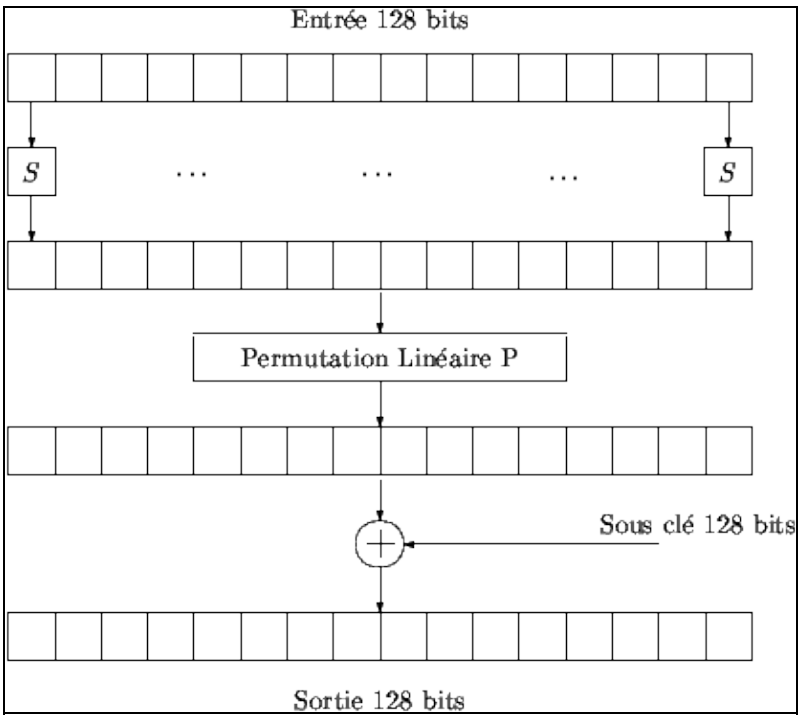

Abb 2: Funktion *F*

Nun wollen wir noch schnell sehen, wie die Substitutionen gebaut werden und wofür die Konstanten α^i gebraucht werden. Technisch und aus Gründen der einfachen Berechnung, wird ein Byte als ein Element aus einer Menge von 256 Elementen betrachtet, genannt Finite Field. In einem Finite Field existieren einfache Operationen wie Addition, Multiplikation und Inverses−Element. Die Substitution S ist die Inverse eines solchen Finite Field.
Die Substitution S ist eine sehr einfache Operation und kann deshalb leicht implementiert werden. Element α^i ist einfach die i−te Potenz im Finite Field. Das macht die AES sehr effizient.

Da AES nur mit einfachen Bit Operationen arbeitet hat es zwei Vorteile:

- Selbst reine Softwareimplementationen von AES sind schnell. Z.B bietet eine C++ Implementation auf einem Pentium 200Mhz eine Geschwindigkeit von 70Mbits/s.

- AES is resistent gegen lineare und differentielle Kryptoanalyse, da AES nicht wie DES von der Wahl der S−Box abhängt, von der immer vermutet wurde, daß sie eine Hintertür für den NSA war.

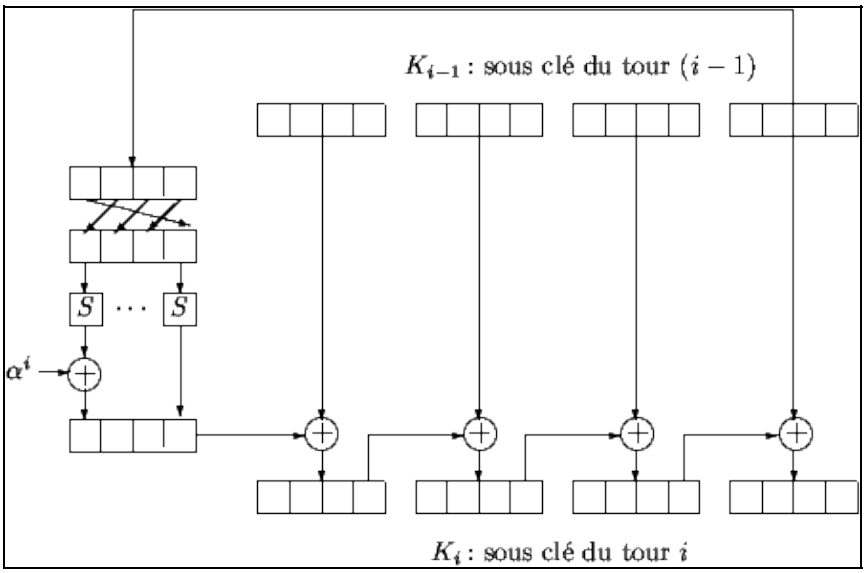

Abb. 3: Schlüsselexpansion

Public Key Cryptography

Im Jahr 1976 haben Diffie und Hellman einen Artikel mit dem Titel "New Directions in Cryptography" veröffentlicht, der zum Renner in der Kryptographiegemeinde wurde. Dieser Artikel stellte ein neues Konzept vor: Public Key Cryptography. Die bis zu dieser Zeit bekannten symmetrischen Algorithmen mit geheimen Schlüssel wurden nicht mehr den Bedürfnissen der modernen Kommunikation gerecht.

Das neue an Public Key Cryptography war die Idee der Trapdoor One−Way Functions. Diese Funktionen lassen sich einfach in einer Richtung berechnen, aber es gibt keine gute Möglichkeit, die Umkehrfunktion zu berechnen. Jedoch haben sie eine Hintertür und wenn man diese Hintertür kennt, kann man auch die Umkehrfunktion leicht aufstellen. Mit anderen Worten: Alle können Nachrichten verschlüsseln, aber nur der, der die Hintertür kennt, kann die Nachricht wieder entschlüsseln. Das war die Geburtsstunde von Alice und Bob. Alice und Bob sind zwei Personen, die geheime Nachrichten austauschen wollen und erfolgreich Eindringlinge schlagen. Eindringlinge, die ihrer Kommunikation lauschen und Daten verändern.

Das schönste Beispiel der Public Key Cryptography (und sicher das Einfachste) wurde zwei Jahre später, 1978, präsentiert. Es wurde erfunden von Rivest, Shamir und Adleman und ist daher als RSA bekannt. Es basiert auf der mathematischen Schwierigkeit eine ganze Zahl in Primfaktoren zu zerlegen. Der geheime Schlüssel besteht aus drei Zahlen *(p,q,d)* wobei p und q zwei Primzahlen mit ungefähr gleicher Größe sind und d eine relative Primzahl zu p−1 and q−1 und die unten stehende Gleichung erfüllen muß. Der öffentliche Schlüssel besteht aus dem Paar *(n,e)*, wobei n=pq, und e der inverse Modulus von (p−1)(q−1) ist.

$$ed = 1 \bmod (p-1)(q-1).$$

Nehmen wir an, Alice möchte einen Text verschlüsselt mit Bobs öffentlichem Schlüssel verschicken (n,e). Dazu transformiert sie die Nachricht in eine Zahl m kleiner n und dann berechnet sie:

$$c = m^e \bmod n,$$

und schickt c zu Bob. Bob berechnet auf seiner Seite mit dem geheimen Schüssel (p,q,d):

$$c^d \bmod n = m^{ed} \bmod n = m.$$

Bei RSA ist die Trapdoor One−Way Function die Funktion, die eine ganze Zahl x<n, mit dem Wert $x^e \bmod n$ assoziiert.

Seit RSA wurden viele andere public key cryptosystems entwickelt. Eine der beliebtesten Alternativen zu RSA ist ein Kryptosystem, das auf diskreten Logarithmen basiert.

Moderne Kryptographie

Public key cryptography ist sehr interessant, weil sie einfach zu benutzen ist und viele Sicherheitsprobleme löst:

- *Individuen identifizieren:* Wenn Alice mit Bob Nachrichten austauscht, möchte sie sicher sein, daß es wirklich Bob ist und nicht jemand der vorgibt, Bob zu sein. Dazu benutzt sie ein Identifikationsprotokoll, das im allgemeinen auf RSA oder diskreten Logarithmen basiert.
- *Dokumentenautentifizierung:* Eine autorisierte Stelle autentifiziert ein Dokument mit einer *digitalen Signatur*. Das Signieren besteht im Anhängen von einigen Bits, die aus dem Inhalt des Dokuments und des Schlüssels der autorisierten Stelle berechnet werden. Das geschieht mit einem Hash Algorithmus wie MD5 oder SHA. Jede Person, die Zugang zu dem Dokument hat, kann überprüfen, daß es wirklich von der autorisierten Stelle unterschrieben wurde. Dazu wird ein Signaturschema benutzt. Eines der bekanntesten ist ElGamal −− wieder basierend auf diskreten Logarithmen.

Daneben bietet public key cryptography, genau wie Kryptographie mit geheimen Schüssel, eine sichere und geheime Kommunikation.

Nehmen wir an, dass Alice geheim mit Bob kommunizieren möchte. Alice holt sich dazu aus einem öffentlichen Verzeichnis den öffentlichen Schüssel von Bob und verschlüsselt damit ihre Nachricht. Bob erhält die Nachricht und kann sie mit seinem geheimen Schlüssel entschlüsseln. Beide Schlüssel, öffentlicher Schüssel und geheimer Schlüssel, haben sehr unterschiedliche Rollen. Deshalb nennt man das auch asymmetrische Kryptographie. Bei Systemen mit geheimen Schlüssel wird derselbe Schlüssel zum Ver− und Entschlüsseln benutzt (symmetrisch).

Public key cryptography bietet einen weiteren riesigen Vorteil. Wenn n Leute miteinander kommunizieren wollen, dann braucht man bei geheimen Schlüsseln für jedes möglich Paar einen Schlüssel. Also n(n−1) Schlüssel. Bei Public key cryptography braucht man nur n Schlüssel. Ein riesiger Vorteil, wenn mehr als tausend Leute miteinander reden möchten. Weiterhin ist es schwierig, mit symmetrischer Kryptographie ein neues Mitglied in die Gruppe einzuführen. Man müßte dazu n Schlüssel erzeugen und sie auf einem sicheren Weg an alle verteilen. Im Gegensatz dazu braucht man bei public key cryptography lediglich einen neuen Schlüssel im Schlüsselverzeichnis zu veröffentlichen.

Öffentlicher Schlüssel oder geheimer Schlüssel: Die richtige Wahl treffen.

Im vorherigen Abschnitt haben wir gesehen, daß public key cryptography viele Probleme löst die Kryptographie mit geheimem Schlüssel nicht lösen kann. Da mag man sich fragen, warum AES, ein symmetrisches Verfahren (geheimer Schlüssel), entwickelt wurde. Es gibt zwei wesentliche Gründe:

- Zunächst einen praktischen Grund. Public key cryptosystems sind sehr langsam. Eine Software Implementation von AES ist tausend mal schneller als RSA und RSA ist fast unmöglich in Hardware zu implementieren. Eine AES Hardwareimplementation ist möglich und liefert höchste Übertragungsgeschwindigkeit.
- Die innere Struktur von public key cryptosystems führt zu anderen Sicherheitsproblemen.

  Public key cryptosystems brauchen viel längere Schlüssel im Vergleich zu symmetrischen Verfahren, um gleiche Sicherheit zu gewährleisten. Bei einem symmetrischen Verfahren ist die Sicherheit so gut wie der Aufwand, alle möglichen Schlüsselkombinationen zu testen. Bei einer Schlüssellänge von 128 Bit muß man also 2^{128} Kombinationen testen.

  Bei public key cryptosystems ist das nicht so. Ein RSA mit 512 Bit ist weit weniger sicher als AES mit 128 Bit. Die einzige Möglichkeit, die Sicherheit eines public key cryptosystems zu berechnen, ist den Aufwand für den momentan besten bekannten Angriff zu berechnen. Eine Gruppe von Forschern hat es kürzlich geschafft, eine 512 Bit Zahl zu faktorisieren. Der allgemeine Hinweis ist jetzt, daß man 1024 Bit RSA benutzen soll.

Zum reinen Verschlüsseln sind also symmetrische Verfahren besser. Zimmermann hat ein interessantes Verfahren ausgearbeitet, daß sowohl asymmetrische als auch symmetrische Verfahren benutzt: PGP. Wenn Alice und Bob kommunizieren wollen, dann sieht das so aus:

- Alice und Bob handeln einen geheimen Schlüssel mit einem Schüsselaustauschprotokol aus. Dieses Protokoll benutzt public key cryptography. Z.B mit dem Diffie–Hellman Algorithmus.
- Danach kommunizieren sie mit einem symmetrischen Verfahren, z.B mit dem IDEA Algorithmus.

Nach der Kommunikation wird der geheime Schlüssel (session key) einfach vernichtet. Normalerweise ist der leichter angreifbare Teil der Kommunikation das Schlüsselaustauschprotokoll mit public key cryptography.

Bibliografie

Geschichte der Kryptographie:

- S. Singh : *Histoire des codes secrets*. Jean–Claude Lattès, 1999.
- D. Kahn : *The Codebreakers: the story of secret writing*. MacMillan publishing, 1996.

Zu AES :

- http://csrc.nist.gov/encryption/aes/rijndael/
- http://www.esat.kuleuven.ac.be/rijmen/rijndael/

Kryptographie im Allgemeinen :

- Artikel von Anne Canteaut und Fran Lévy–dit–Véhel : http://www–rocq.inria.fr/canteaut/crypto_moderne.pdf

• B. Schneier : *Applied Cryptography*. John Wiley and Sons, 1996.

Translation information:
fr −−> −− : Pierre Loidreau <pierre.loidreau/at/ensta.fr>
fr −−> en: Axelle Apvrille <axellec/at/netcourrier.com>
en −−> de: Guido Socher <guido(at)linuxfocus.org>

2005−01−11, generated by lfparser_pdf version 2.51