Prof. Dr. Norbert Pohlmann, Malte Hesse

Kryptographie: Von der Geheimwissenschaft zur alltäglichen Nutzanwendung (III) – Symmetrische Verschlüsselungsverfahren

In dieser Folge unserer Reihe befassen wir uns mit den am weitesten verbreiteten Verschlüsselungsverfahren sowie ihren wichtigsten Vor- und Nachteilen.

Die bekanntesten und gebräuchlichsten symmetrischen Verschlüsselungsverfahren sind DES und AES

Die in der letzten Ausgabe vorgestellten elementaren Verfahren gehören ohne Ausnahme zur Gattung der so genannten symmetrischen Verschlüsselungen. Deren Hauptkennzeichen besteht darin, dass alle an einer Kommunikation beteiligten Instanzen den gleichen (vollständigen) Schlüssel kennen und einsetzen. Wir haben jedoch gesehen, dass diese Verfahren relativ leicht zu brechen sind und damit heute keinen ausreichenden Schutz mehr bieten. Um diesen Nachteil auszugleichen, verknüpft man im praktischen Einsatz mehrere elementare Verfahren mit verschiedenen kryptographischen Eigenschaften zu so genannten Produktverschlüsselungen. Diese zeichnen sich dadurch aus, dass sie schwerer zu entschlüsseln sind als jedes Einzelverfahren. Eine der gängigsten Methoden ist dabei die iterative (wiederholte) Verknüpfung nichtlinearer Substitutionen und Permutationen. Bekannteste Vertreter dieser Gattung sind der *Data Encryption Standard* (DES), der *Advanced Encryption Standard* (AES) und der *International Data Encryption Algorithm* (IDEA). Allerdings ist letzteres Verfahren, das 1990 gemeinsam von der Firma Systec AG und der ETH Zürich entwickelt wurde, noch bis 2011 lizenzkostenpflichtig und entsprechend gering verbreitet, weswegen eine weitere Betrachtung hier unterbleibt.

Data Encryption Standard

Beide beruhen auf der wiederholten Verknüpfung elementarer Verfahren wie Substitution und Permutation

Der DES stellt eine Block-Produkt-Verschlüsselung aus nichtlinearer Substitution und Permutation dar, die schlüsselgesteuert in einer Iterationsschleife 16 Mal durchlaufen werden. Kleine Änderungen im Klartext oder Schlüssel führen dabei zu großen Änderungen im Schlüsseltext – ein Verhalten, das alle als sicher geltenden kryptographischen Verfahren aufweisen müssen. Entwickelt wurde der Algorithmus bereits Mitte der 70er-Jahre von einem IBM-Forscherteam um Horst Feistel, Walter Tuchman und Don Coppersmith, das mit der US-Standardisierungsbehörde NBS (heute NIST) und der *National Security Agency* (NSA) zusammenarbeitete. Als lizenzkostenfreies Verfahren, das überdies von vornherein für eine Hardware-Implementierung optimiert war, konnte sich DES sehr bald auch international etablieren. Auf Kritik stieß

allerdings von Anfang an die Kooperation zwischen IBM und der NSA, die u. a. zur Reduzierung der ursprünglich geplanten Schlüssellänge von 128 auf 56 Bit führte, wodurch der Algorithmus für Brute-Force-Angriffe anfällig wurde. Heute gilt diese Schüsselläge weder theoretisch (absolut) noch praktisch (rechnerisch) als sicher, da moderne Rechner die insgesamt 2^{56} möglichen Schlüsselkombinationen in zu kurzer Zeit durchspielen und damit das Verfahren „knacken“ können. Aufgrund seiner weiten Verbreitung und der bereits erwähnten guten Eignung für die Realisierung in Hardware wurde es jedoch beständig weiterentwickelt und ist noch heute als Triple-DES-Verfahren mit einer Schlüssellänge von 168 Bit im Einsatz, bei dem der Inputblock mit unterschiedlichen Schlüsseln dreimal verschlüsselt wird.

Advanced Encryption Standard

Der lange gebräuchliche internationale Quasi-Standard DES wurde im Jahr 2000 durch AES abgelöst

Als weitere Folge dieser Entwicklung sah sich die NIST im September 1997 veranlasst, abermals einen Wettbewerb zur Einführung eines neuen Verschlüsselungsstandards abzuhalten. Der Algorithmus musste vor allem zwei Kriterien erfüllen, nämlich erstens – natürlich – zumindest für die nächsten zwei Jahrzehnte als rechnerisch sicher gelten und zweitens – anders als IDEA – lizenzkostenfrei sein. Weitere Anforderungen betrafen Leistungsfähigkeit, Effizienz, Flexibilität und Implementierbarkeit: Die erforderlichen Operationen sollten schnell erfolgen, Speicher und CPU der eingesetzten Rechner nicht zu sehr belasten und sich sowohl für den Einsatz in Embedded-Geräten als auch als „reine“ Software-Lösung (etwa fürs Online-Banking) eignen.

Von den ursprünglich eingereichten 21 Vorschlägen genügten 15 diesen Mindestanforderungen. Im Verlauf der weiteren Auswertung blieben schließlich fünf Verfahren für die „letzte Runde“ übrig, unter denen sich das Wettbewerbskomitee schließlich für den sog. Rijndael-Algorithmus entschied, den ein Kryptologen-Team der belgischen Universität Leuven um Vincent Rijmen vorgelegt hatte. Den Ausschlag gab dabei einerseits, dass das Verfahren von vornherein für den Einsatz mit variablen Schlüssellängen von 128, 192 und 256 Bit ausgelegt war und andererseits im Gegensatz zu den Alternativvorschlägen in Hard- wie Software-Implementierungen gleichermaßen schnell funktionierte. Praktische Einsatzbeispiele für den mit dem NIST-Entscheid in *Advanced Encryption Standard* „umgetauften“ Algorithmus sind u. a. die E-Mail-Verschlüsselung mit Hilfe von S/MIME und Übertragungsprotokolle wie IPSec und SSL für den Datenaustausch im Internet.

Funktionsweise von AES

AES unterzieht Klartextblöcke von 128, 192 oder 256 Bit Länge bis zu 14 Transformationen

Wie sein Vorgänger bzw. „Noch-Rivale“ ist auch AES ein Produktverschlüsselungsverfahren, welches in mehreren Runden die Bits transformiert. Dazu wird zunächst der Klartext in Blökke fester Bitlängen eingeteilt, die 128, 192 oder 256 Bit betragen kann. Die Anzahl der Transformationsrunden hängt dabei von der Block- und der Schlüssellänge ab und beträgt 10, 12 oder

14. Jede Runde besteht aus einer Reihe byteorientierter Transformationen, welche die Stärken vieler anderer Verschlüsselungsalgorithmen kombinieren. Im ersten Schritt werden die in einem zweidimensionalen Array abgelegten Zeichen des Klartext-Blocks der so genannten *ByteSub-Transformation* unterworfen. Es handelt sich also um eine monoalphabetische Substitution der einzelnen Bytes, die über eine Tabelle (die so genannte S-Box) festgelegt wird (vgl. Abb. 1).

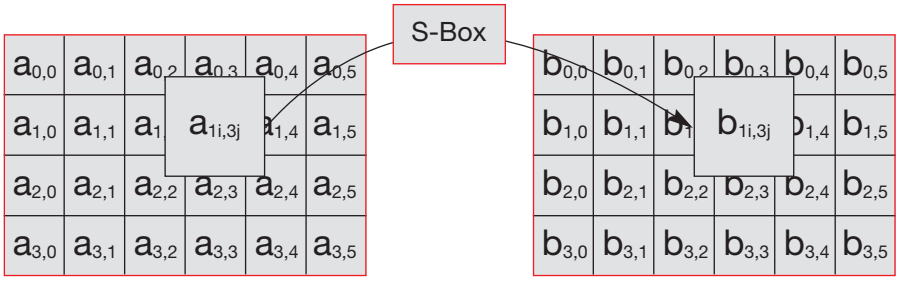

ABB. 1: Erster Schritt von AES ist die sog. ByteSub-Transformation, hier dargestellt für den Fall einer Blocklänge von 192 Bit, bei denen der Block in einem Array von 4 x 6 Bytes abgelegt ist.

Den zweiten Schritt stellt die *ShiftRow-Transformation* dar, eine Permutation, bei der alle Zeilen des Arrays außer der ersten abhängig von der Zeilen- und Blocklänge um maximal vier Spalten nach links verschoben werden. Überlaufende Zellen werden von rechts fortgesetzt (vgl. Abb. 2).

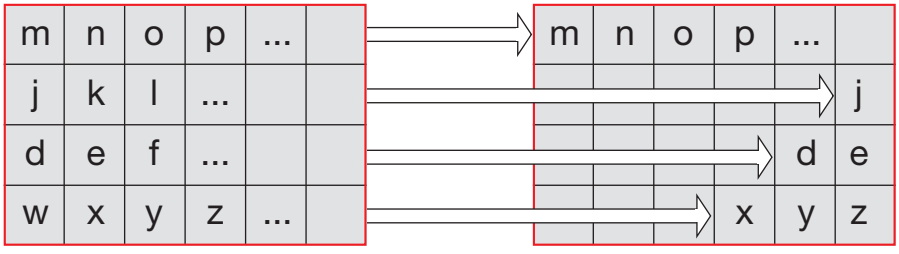

ABB. 2: Im zweiten AES-Schritt wird das Ergebnis der ByteSub- einer ShiftRow-Transformation unterzogen.

Der dritte Schritt ist die *MixColum-Transformation,* bei der jede Spalte des Arrays mit einem festen Polynom multipliziert wird (vgl. Abb. 3). Die MixColumn-Transformation wird in der letzen Runde von AES überschlagen und es folgt dann direkt der nächste Schritt.

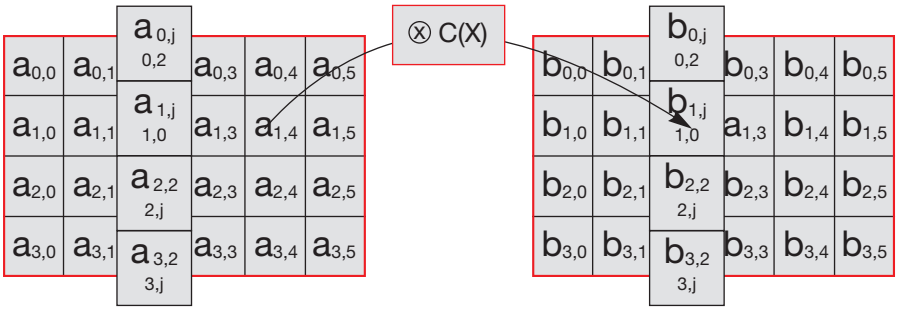

ABB. 3: Im dritten Schritt, der MixColumn-Transformation, multipliziert AES jede Spalte des Arrays mit einem zuvor festgelegten Wert.

In der abschließenden *AddRoundKey-Transformation* wird der aus dem geheimen Schlüssel ermittelte Rundenschlüssel mit dem Array durch ein bitweises XOR verknüpft (vgl. Abb. 4).

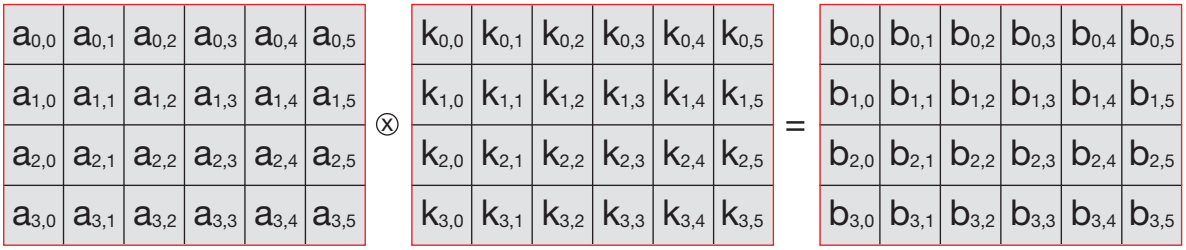

ABB. 4: Die abschließende AddRoundKey-Transformation fügt den vorigen Schritten den Rundenschlüssel hinzu.

Die in den einzelnen Runden benutzten Rundenschlüssel werden aus dem originalen Schlüssel durch eine so genannte Expansions-Funktion berechnet. Diese berechnet die Rundenschlüssel mit XOR, zyklischen Shifts und einem Tabellen-Lookup. Dabei wird z. B. ein Puffer der Länge

(Blocklänge in Bit) * (Anzahl der Runden + 1)

gefüllt, dem dann die jeweiligen Rundenschlüssel entnommen werden. Die ersten n Bit (n = verwendete Schlüssellänge) des Puffers entsprechen dem Schlüssel in unverfälschter Form, alle anderen jeweils n Bit entstehen aus den vorherigen durch zyklische Permutation und eine Substitution, die der Byte-Sub-Transformation ähnelt. Vor Beginn der ersten Runde erfolgt eine initiale AddRoundKey-Transformation, die den Klartext mit dem ersten Rundenschlüssel verknüpft. Die Entschlüsselung erfolgt analog zur Verschlüsselung.

Insgesamt kombiniert der AES-Algorithmus also eine Reihe von Elementarverschlüsselungen in besonders geschickter Anordnung. Die relative Einfachheit macht das Verfahren besonders schnell und flexibel, so dass es zurzeit keine echte Alternative gibt.

Verwaltung von Schlüsseln (Key Management)

Höhere Anforderungen als die Verschlüsselungsoperationen stellen Erzeugung, Speicherung, Wechsel und Verteilung der benötigten Schlüssel

Komplexere Anforderungen als die Verschlüsselungsoperationen selbst stellt die Verwaltung der benötigten Schlüssel – komplexer deshalb, weil an dieser Stelle des Verfahrens der „Unsicherheitsfaktor Mensch“ ins Spiel kommt: Erzeugung, Speicherung, Wechsel und Verteilung von Schlüsseln sind meist entweder Sache des Anwenders oder werden zumindest von diesem angestoßen. In jedem Stadium dieses Zyklus lauern bestimmte Gefahren, die wir an dieser Stelle lediglich kurz abhandeln, um uns in der nächsten Folge mit der Hauptschwierigkeit auseinanderzusetzen.

Bei der *Erzeugung* besteht das Risiko, dass der Anwender einen zu einfachen Schlüssel wie zum Beispiel den eigenen Vornamen verwendet, den auch ungeübte Angreifer leicht erraten können.
Aus diesem Grund sollten die Schlüssel immer mit Hilfe von echten Zufallszahlengeneratoren berechnet werden. Weiter sind Aspekte wie Streuung, Periodizität und Gleichverteilung zu beachten.

Auch die *Verwahrung* ist nicht ganz trivial: Schlüssel sind als Zufallszahlen aufgrund ihrer Entropie-Eigenschaft leicht im Speicherbereich eines Computers aufzufinden. Daher sind sie selbst zusätzlich durch eine Verschlüsselung zu schützen oder, besser noch, in einer Sicherheitsumgebung zu verwahren. Dabei haben sich etwa SmartCards und USB-Token bewährt, die auch unter dem Oberbegriff *Personal Security Environment* zusammengefasst werden. Der geheime Schlüssel wird auf ihnen gespeichert und verlässt zu keiner Zeit die sichere Umgebung. Zugriff hat nur der legitime Besitzer, der sich gewöhnlich zusätzlich durch PIN-Eingabe authentifiziert. Die SmartCard enthält neben den Schlüsseln auch die Algorithmen, die für die Verschlüsselungsoperationen benötigt werden.

Damit ein eingesetztes Kryptoverfahren auch sicher bleibt, ist zudem von Zeit zu Zeit ein *Schlüsselwechsel* erforderlich. Wie häufig dieser erfolgt, hängt, wie wir bereits in der ersten Folge gesehen haben, vom Einsatzzweck bzw. der Anwendung und der Umgebung ab – von täglich bis einmal im Jahr ist alles denkbar. Festgelegt wird dieser Zeitraum in einer eigenen Policy, die im laufenden Management- und Sicherheitsprozess zu erarbeiten und regelmäßig an die Bedürfnisse des Anwenders anzupassen ist.

Das schwierigste und daher wichtigste Problem jedoch besteht in der Verteilung des oder der Schlüssel an mögliche Kommunikationspartner. Sind Anwendungen mit höchsten Sicherheitsanforderungen betroffen, wird dafür selbst heute oft noch ein vertrauenswürdiger Bote eingesetzt, der den auf einem physikalischen

Medium (Papier oder Datenträger) fixierten Schlüssel vom Ort der Erzeugung zu den Einsatzorten bringt (vgl. Abb. 5). Speziell in sehr großen Umgebungen erfordert dies jedoch einen kaum vertretbaren Geld- und Zeitaufwand. Zudem ist selbst bei dieser Methode (und erst recht bei allen anderen) weder theoretisch noch praktisch auszuschließen, dass der Schlüssel bei der Übermittlung in die Hände Unbefugter gelangt – wodurch automatisch das gesamte Verfahren ausgehebelt wird. Daher hat man für diese Zwecke unterschiedliche Key-Management-Protokolle entwickelt, die sich ihrerseits wiederum eigener, asymmetrischer Verschlüsselungsverfahren bedienen. Wie diese funktionieren, zeigen wir in der nächsten Ausgabe.

Für das Schlüsselmanagement wurden daher eigene Protokolle entwickelt, die sich asymmetrischer Verschlüsselungsverfahren bedienen

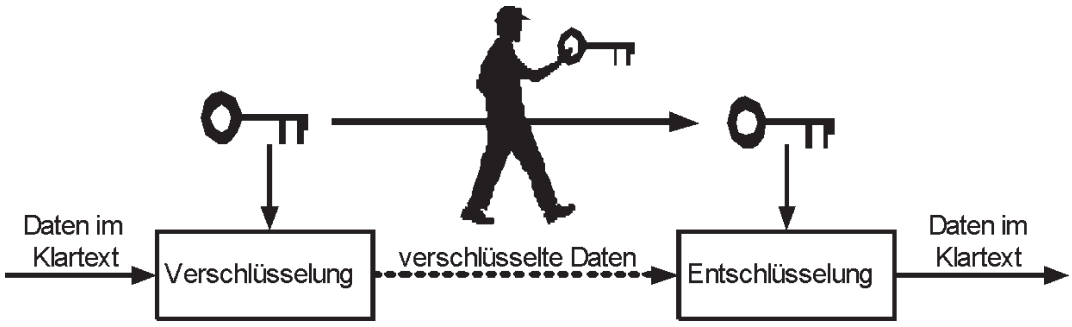

ABB. 5: Die größte Sicherheitslücke aller Kryptoverfahren besteht bei der Übermittlung der Schlüssel an die Anwender: Selbst wenn für diesen Schritt – wie hier schematisch dargestellt – ein vertrauenswürdiger Bote gewählt wird, lassen sich Verlust oder Diebstahl nie ganz ausschließen.

Zu den Autoren:
Prof. Dr. Norbert Pohlmann ist Geschäftsführender Direktor des Instituts für Internet-Sicherheit der Fachhochschule Gelsenkirchen. E-Mail-Kontakt: norbert.pohlmann@informatik.fh-gelsenkirchen.de

Malte Hesse ist Mitarbeiter am Institut für Internet-Sicherheit der Fachhochschule Gelsenkirchen. E-Mail-Kontakt: hesse@internet-sicherheit.de