Max-Born-Gymnasium

Facharbeit im Leistungskurs Mathematik

über das Thema

Sicherheit im WLAN – Mathematische Hintergründe

von

Sebastian Wiesner

Kursleiter: Herr Riedl

Abgabetermin: _____

Punkte: _____

Punkte eingetragen am: _____

Zurückgegeben am: _____

Dem Direktorat vorgelegt am: _____

_____

Unterschrift des Kursleiters

Inhaltsverzeichnis

Abbildungsverzeichnis

1. Einführung

„Sicher ist, dass nichts sicher ist“

Karl Valentin

Dieser Spruch des bayrischen Komikers Karl Valentin gilt auch Zeitalter von DSL und Multimedia unverändert. Dennoch ist heute ein hohes Maß an Sicherheit erforderlich, zumal die Vernetzung nicht nur im globalen Maßstab, sondern auch im kleinen privaten Rahmen stetig zunimmt. Viele besitzen nicht nur einen Computer, sondern neben dem normalen Bürorechner auch noch ein Laptop oder einen Multimedia-PC fürs Wohnzimmer. Da stellt sich schnell die Frage nach dem Verbinden dieser Rechner zwecks gemeinsamem Internetzugang und Datenaustausch. Das Verlegen von Kabeln im Eigenheim oder in der Mietwohnung ist nicht jedermanns Sache und auch nicht immer möglich. Moderne Technik bietet hier eine Lösung: Netzwerke werden als sogenannte *WLANs* schnell und ohne das Verlegen von Kabeln eingerichtet. Den wenigsten Nutzern sind jedoch die Sicherheitsrisiken dieser Technik bewusst. Ein kabelgebundenes Netzwerk ist einfach zu kontrollieren, sind entsprechende Anschlusspunkte doch in der Regel gut geschützt durch die sprichwörtlichen „eigenen vier Wände“.

Anders sieht es jedoch bei drahtlosen Netzwerken aus: Aufgrund der Eigenschaften von Funkwellen kann sich grundsätzlich jeder mit dem Netzwerk verbinden. Das ist im besten Falle unerwünscht, etwa wenn der Nachbar den eignen Internetzugang mitbenutzt. Im schlimmsten Fall jedoch kann sorgloser Umgang mit der Technik sogar strafrechtliche Relevanz besitzen, etwa wenn ein Unbekannter den eigenen Internetzugang zum illegalen Herunterladen urheberrechtlicher geschützter Musik verwendet [Hei].

Unverständlicherweise wird dieser Aspekt von potentiell Betroffenen allzu oft ausgeblendet. Dabei ist die Vermeidung derartiger Risiken relativ einfach, verlangt allerdings zumindest ein grundsätzliches Verständis der entsprechenden Sicherheitstechnologien. Deren Grundlagen sind die Wissenschaften der Kryptographie und der Kryptanalyse, also der Lehre von der Verschlüsselung von Daten beziehungsweise vom Brechen von Verschlüsselungen. Da heutzutage beide Wissenschaften eng miteinander verwoben sind, spricht man oft auch von der Kryptologie als Vereinigung der beiden Gebiete.

In dieser Arbeit werden zunächst die mathematischen Hintergründe der Kryptologie erläutert. Darauf aufbauend werden dann die aktuellen Technologien zum Schutz von drahtlosen Netzwerken untersucht.

2. Theoretische Hintergründe

2.1. Nachrichten

Um ein Verständnis für die Kryptologie zu entwickeln, gilt es sich zunächst den Gegenstand der Verschlüsselung anzusehen: die **Nachricht**. Eine Nachricht ist prinzipiell jede Art der Kommunikation zwischen zwei oder mehr Personen oder Institutionen. In der Kryptographie bezeichnet man eine unverschlüsselte Nachricht als **Klartext** (*engl. plaintext*) M. In diesem Zustand versandt, ist die Nachricht für jede beliebige Person lesbar. Dies ist jedoch oftmals nicht erwünscht, wenn sie persönliche oder gar sicherheitsrelevante Informationen beinhaltet, welche nur einem bestimmten Empfängerkreis zugänglich sein sollten.

Dies zu ermöglichen, ist das Ziel der Kryptographie. Dazu wird der Klartext mittels mathematischer Funktionen, sogenannter Algorithmen (vgl. Abschnitt 2.2), in den **Geheimtext** oder **Chriffretext** (*engl. ciphertext*) C umwandelt, der nun nicht mehr lesbar ist.

Sowohl Klartext als auch Chiffretext sind Abfolgen von Zeichen. Man bezeichnet sie daher als **Zeichenketten**. Die Menge aller Zeichen, die in diesen Zeichenketten vorkommen, ist das **Alphabet** A. Die Anzahl aller Zeichen in diesem Alphabet nennt man die Mächtigkeit $n = |A|$. Der Text „abcde“ ist zum Beispiel eine Zeichenkette über dem Alphabet $A = \{a, b, c, d, e\}$ mit der Mächtigkeit $n = 5$. Die Länge der Nachricht selbst ist für die Verschlüsselung allerdings nicht relevant.

Im Bezug auf die Technik, die das Thema dieser Arbeit bildet, sind Nachrichten jedoch keine normalen, aus Buchstaben bestehenden Texte, sondern binäre Datenpakete, welche über das drahtlose Netzwerk verschickt werden. Diese Nachrichten werden folglich über dem binären Alphabet $A = \{0, 1\}$ gebildet, welches die Mächtigkeit $n = 2$ besitzt. Im binären Alphabet werden Zeichen überlicherweise als **Bit** bezeichnet, wobei jeweils 8 Bit zu einem **Byte** zusammengefasst werden, da Computer nicht auf einzelnen Zeichen operieren können. Die einzelnen Zeichen eines Bytes werden dann als duale Zahl interpretiert. Ein Byte kann folglich Werte zwischen 0 (Dualzahl 00000000) und 255 (Dualzahl 11111111) annehmen.

2.2. Algorithmen

Um nun aus dem Klartext einen Chiffretext zu erhalten, d.h. die Nachricht zu verschlüsseln (*engl. to encrypt*), bedient man sich, wie bereits gesagt, eines **Algorithmus**. Ein zur Verschlüsselung verwendeter Algorithmus erhält die Bezeichnung **Chiffre** E. Um den Chiffretext in den Klartext zu entschlüsseln (*engl. to decrypt*), wird anschließend die Dechiffrierfunktion D verwendet. Es gilt also $C = E(M)$ und $M = D(C)$. Daraus folgt:

$$D(E(M)) = M \tag{1}$$

Diese Gleichungen sind nur lösbar, wenn D die Umkehrfunktion von E ist. Deswegen sollte zur Verschlüsselung nur eine umkehrbare Funktion verwendet werden. Andernfalls wäre die Entschlüsselung eines Chiffretextes nicht mehr durchführbar.

2.2.1. Eingeschränkte Algorithmen

Nach Gleichung 1 erhält der Algorithmus nur einen einzigen Parameter, nämlich den Klartext. Dies bedeutet, dass jede Person, die im Besitz des Algorithmus ist, Chiffretext entschlüsseln kann.

Definition 2.1 *Ein Algorithmus heißt **eingeschränkt**, wenn die Sicherheit von der Geheimhaltung des Algorithmus abhängt.*

Daraus ergeben sich drei grundsätzliche Probleme:

1. Ändert sich die Gruppe der Nutzer eines Algorithmus, müsste zur Gewährleistung der Sicherheit der Algorithmus geändert werden. Die Entwicklung eines neuen Algorithmus ist in der Praxis aber sehr aufwändig.
2. Ist in einer handelsüblichen kryptographischen Software ein Algorithmus in den Quellcode integriert, so kann der Algorithmus jederzeit aus dem Quellcode abgeleitet werden. Insofern erweckt eine solche Software lediglich den Anschein von Sicherheit, ohne dass das eigentliche Problem (die Offenlegung des Algorithmus) damit gelöst ist.
3. Da die Offenlegung eines eingeschränkten Algorithmus seiner Funktion grundsätzlich entgegensteht, kann er auch nicht durch die eine größere kryptologische Gemeinschaft hinsichtlich seiner Sicherheit überprüft werden.

Eine Lösung für diese Probleme wäre die Einführung eines zusätzlichen Parameters, des **Schlüssels** in Gleichung 1.

2.2.2. Algorithmen mit Schlüssel

Fügt man der Gleichung 1 einen Schlüssel hinzu, so gilt $C = E_k(M)$ und $M = D_k(C)$. Daraus folgt:

$$D_k(E_k(M)) = M \tag{2}$$

Algorithmen, die den selben Schlüssel sowohl zur Ver- als auch zur Entschlüsselung verwenden, heißen **symmetrische Algorithmen**. Solche, die zwei verschiedene Schlüssel K_1 und K_2 zur Ver- und Entschlüsselung verwenden, nennt man **asymmetrische Algorithmen**. Für die Betrachtung von drahtlosen Netzwerken ist nur die erste Art

von Bedeutung, da die im Rahmen dieser Arbeit behandelten Techniken ausschließlich auf symmetrischen Algorithmen aufbauen.

Ein Schlüssel ist, wie auch Klar- und Chiffretext, eine Zeichenkette über einem bestimmten Alphabet, wobei dieses Alphabet nicht gleich dem Alphabet von Klar- oder Chiffretext sein muss. Bei der Verschlüsselung kabelloser Netzwerke wird der Schlüssel allerdings genau wie Klar- und Chiffretext über dem binären Alphabet gebildet.

Im Gegensatz zu den Nachrichten ist die Länge des Schlüssels wichtig für die Verschlüsselung, da die Komplexität einer Verschlüsselung mit der Länge der Schlüssel steigt. Dadurch ist für das Brechen der Verschlüsselung ein höherer Aufwand erforderlich. Heutzutage werden binäre Schlüssel für symmetrische Algorithmen üblicherweise in einer Länge zwischen 128 und 256 Bit verwendet. Die Länge bei asymmetrischen Algorithmen liegt in der Regel zwischen 1024 und 2048 Bit.

Durch die Verwendung eines Schlüssels werden die ersten zwei der in Abschnitt 2.2.1 erwähnten Probleme erfolgreich beseitigt:

1. Ändert sich die Gruppe der Nutzer, so muss nur der Schlüssel geändert werden. Dies ist wesentlich weniger aufwändig und erheblich sicherer als das Ändern des Algorithmus.
2. Die Offenlegung eines Algorithmus durch die Integration in eine Software ist nun möglich, da die Sicherheit nur vom Schlüssel abhängt.

Der notwendige Schritt zur Beseitgung des dritten Problems wird jedoch oftmals nicht getan. Viele Firmen, die kryptographische Algorithmen entwickeln, vertrauen noch immer auf das Prinzip **„security by obscurity“**. Dabei wird ein Algorithmus nicht veröffentlicht, sondern bleibt geheim.

Dies hat zum Teil katastrophale Folgen für die Sicherheit. Anerkannte Kryptologen haben keine Gelegenheit, diese Algorithmen genauestens zu überprüfen. Deswegen existieren keine verlässlichen Aussagen über die Sicherheit eines solche Algorithmus. Fehler im Design bleiben möglicherweise lange unentdeckt. Wird dieser Algorithmus dann trotzdem Teil einer weit verbreiteten Anwendung oder Geräts (z.B. eines WLAN-Routers), so gefährdet eine einzige Sicherheitslücke die Daten aller Anwender.

2.2.3. Starke Kryptographie

Idealerweise sollte die Sicherheit einer verschlüsselten Nachricht ausschließlich von der Geheimhaltung des Schlüssels abhängen. Diese Forderung wurde bereits im 19. Jahrhundert vom holländischen Kryptographen Auguste Kerckhoffs [Wika] aufgestellt und gilt noch heute unter dem Namen Kerckhoffs' Prinzip [Wikb] als eines der grundlegenden Theoreme der Kryptographie.

Definition 2.2 *Das Prinzip von Kerckhoffs besagt, dass die Sicherheit eines Verschlüsselungsverfahrens nur von der Geheimhaltung des Schlüssels abhängen darf, nicht jedoch von der Geheimhaltung des Algorithmus. [Ert03, Seite 19]*

Dieses Prinzip führte zu einem offenen Modell bei der Entwicklung neuer Algorithmen, welches besonders beim Wettbewerb um die Entwicklung des AES (Advanced Encryption Standard) Algorithmus [NIS] zu sehen war.

Bei diesem Modell werden Algorithmen schon während ihrer Entwicklung möglichst frühzeitig offengelegt. Dadurch wird Kryptologen überall auf der Welt die Analyse dieses Algorithmus ermöglicht. Finden selbst anerkannte Kryptologen über einen längeren Zeitraum hinweg keine Schwachstellen, so ist davon auszugehen, dass ein Algorithmus im praktischen Einsatz ein großes Maß an Sicherheit bietet. Auf der anderen Seite werden mögliche Schwachstellen wiederrum frühzeitig erkannt und publiziert, so dass ein unsicherer Algorithmus idealerweise gar nicht erst von einer breiten Anwenderschaft genutzt wird.

Algorithmen, die nach diesem Modell entwickelt wurden und Kerckhoffs' Prinzip erfüllen, bezeichnet man als **starke Kryptographie**. Hier ist der Mensch als Besitzer des Schlüssels die größte Schwachstelle, auf die sich in der Praxis die meisten Angriffe konzentrieren (vgl. Abschnitt 2.5.1).

Damit wäre auch das dritte der in Abschnitt 2.2.1 aufgezeigten Probleme beseitigt. Die starke Kryptographie bietet dem Anwender nicht nur den ungehinderten Zugang zu hochwertigen und sicheren Algorithmen, sie ermöglicht ihm auch eine realistische Einschätzung des Gefahrenpotentials. Zeigt ein Algorithmus eine Schwachstelle, erfährt der Anwender dies meistens noch vor dem böswilligen Angreifer und hat somit die Chance, rechtzeitig entsprechende Maßnahmen zu ergreifen, in dem er zum Beispiel auf einen anderen öffentlich verfügbaren Algorithmus der starken Kryptographie umsteigt.

2.3. Kryptosysteme

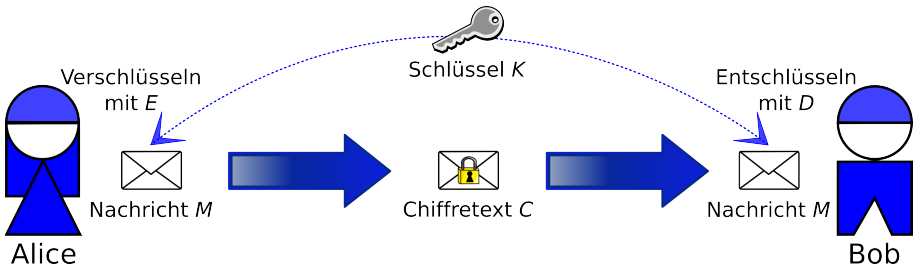

Abbildung 1: Ein Kryptosystem (Sebastian Wiesner, 2007)

Fügt man nun alle in den vorherigen Abschnitten erwähnten Elemente zusammen, so erhält man ein **Kryptosystem** (Abbildung 1[1]), bestehend aus

- dem Klartext M,
- dem Chiffretext C,
- dem Schlüssel K,
- der Chiffrierfunktion E
- und der Dechiffrierfunktion D.

Für ein solches Kryptosystem werden zwei grundsätzliche Anforderungen definiert [Wä04, Seite 3-4]:

Die Geheimhaltungsforderung Dem Kryptoanalytiker darf es nicht möglich sein, den Klartext aus dem Chiffretext zu bestimmen. Dies leuchtet ein, andernfalls wäre eine Verschlüsselung ja sinnlos. Allerdings darf es dem Kryptoanalytiker auch nicht möglich sein, die Dechiffrierfunktion D_k aus dem Chiffretext zu bestimmmen, selbst wenn ihm sogar der Klartext $M = D_k(C)$ bekannt ist. Durch diese Forderung wird gewährleistet, dass dem Kryptoanalytiker das Knacken der Verschlüsselung nicht gelingt.

Die Authentizitätsanforderung Einem Kryptoanalytiker darf es nicht möglich sein, die Chiffrierfunktion E_k aus dem Chiffretext zu bestimmen, selbst wenn der Klartext M bekannt ist. Zudem darf er keinen alternativen Chiffretext C' finden, für den $D_k(C')$ einen gültigen Klartext über Klartext-Alphabet A_M ergibt[2]. Dadurch wird verhindert, dass der Kryptoanalytiker gefälschte Klartexte in das Kryptosystem einschleusen kann, in dem er gezielt Chiffretexte erzeugt.

Man muss dabei beachten, dass die Forderung, weder E_k noch D_k aus C herleiten zu können, der Idee der starken Kryptographie nicht widerspricht, auch wenn dies zuerst so scheinen mag. Die Anforderungen verlangen, dass der Kryptoanalytiker allein aus dem Chiffretext C keine weiteren Informationen über das Kryptosystem ableiten kann. Dem setzt die starke Kryptographie die Annahme entgegen, dass dem Kryptoanalytiker im Bezug auf die verwendeten Algorithmen bereits alles bekannt ist. Ein starkes Kryptosystem sollte trotz Veröffentlichung der Algorithmen sicher sein. Die starke

[1] Die Namen „Alice“ und „Bob“ für die Kommunikationspartner sind Teil der kryptologischen Fachsprache.

[2] Bei numerischen Daten ist dies durchaus möglich. Der böswillige Angreifer Mallory (ebenfalls ein kryptologischer Fachausdurck) erzeugt einen zufälligen Chiffretext C' und schleust diesen in das System ein. Es ist nun äußerst unwahrscheinlich, dass Bob beim Entschlüsseln von C' einen korrekten Satz erhält. Die Manipulation würde also sofort auffallen, wenn Alice und Bob über normale, (zumindest meistens) grammatikalisch korrekte Sätze kommunizieren. Anders sieht dies jedoch aus, wenn Alice und Bob beispielsweise Telefonnummern (also Zahlenfolgen) austauschen. Entschlüsselt Bob unter diesen Bedingungen C', so erhält er in jedem Fall eine Abfolge von Zahlen. Er kann also nicht sofort entscheiden, ob die Nachricht korrekt oder manipuliert war. Dazu sind weitere Nachforschungen nötig. Die Authentizitätsforderung ist also nicht erfüllt. Zur Vermeidung derartiger Probleme verwendet man Prüfsummen (vgl. Abschnitt 3).

Kryptographie geht also noch einen Schritt weiter als die theoretischen Anforderungen.

2.4. Protokolle

Kommunizieren zwei Menschen nur gelegentlich über verschlüsselte Nachrichten miteinander, so reicht es ggf. aus, die einzelnen Teile des Kryptosystems sowie die Art des Schlüsselaustausches und der Nachrichtenübermittlung informell zu vereinbaren. Dies ist bei der Kommunikation zwischen Computern oder größeren Gruppen nicht mehr möglich.

Stattdessen müssen die Abläufe in einer formalen Vorschrift, dem **Protokoll**, festgehalten werden. Diese formale Vorschrift definiert, wie sich die an der Kommunikation beteiligten Personen oder Computer verhalten müssen. Es legt die Verfahren für den Beginn, die Durchführung und das Ende der Kommunikation innerhalb des jeweiligen Netzwerkes fest sowie das Verhalten bei unerwarteten Situationen oder sicherheitsrelevanten Ereignissen.

2.5. Kryptoanalyse

In der Kryptoanalyse beschäftigt man sich mit dem Brechen oder Knacken von Kryptosystemen, welches sich nach dem Grad des Erfolgs einteilen lässt [Ert03, Seite 23]:

- Der Kryptoanalytiker kann nur einzelne Nachrichten lesen:

  Informationsdeduktion Der Kryptoanalytiker erlangt nur einige bruchstückhafte Informationen über den Klartext oder den Schlüssel.

  Lokale Deduktion Der Kryptoanalytiker hat aus einem einzigen Chiffretext C den Klartext M rekonstruiert, allerdings ohne wesentliche Informationen über das Kryptosystem zu erlangen.

- Der Kryptoanalytiker kann jeden beliebigen mit dem Schlüssel K erzeugten Chiffretext C entschlüsseln:

  Globale Deduktion Der Kryptoanalytiker findet eine alternative Funktion D', so dass gilt $M = D'(E_k(C))$. In anderen Worten, die Funktion D' kann für einen bestimmten Schlüssel K gleichwertig zu D_k den Klartext aus dem Chiffretext berechnen, allerdings ohne die Notwendigkeit, den Schlüssel K zu kennen.

  Vollständiges Aufbrechen Der Kryptoanalytiker gelangt an den Schlüssel K, welcher jeden Chiffretext C mittels $M = D_k(C)$ entschlüsseln kann.

2.5.1. Angriffe

Das Brechen von Verschlüsselungen kann auf verschiedenen Wegen erfolgen [Ert03, Seite 23]:

Ciphertext-Only-Angriff Dem Kryptoanalytiker steht nur der Chiffretext zur Verfügung.

Known-Plaintext-Angriff Der Kryptoanalytiker kennt zusätzlich noch den zum Chiffretext passenden Klartext. Dies ist in der Praxis keine Seltenheit, da insbesondere technische Protokolle häufig ein bestimmtes Datenformat festlegen. Auch bei zwischenmenschlicher Kommunikation sind oftmals bestimmte Floskeln gebräuchlich. Viele förmliche Schreiben beginnen beispielsweise mit der höflichen Anrede „Sehr geehrte Damen und Herren“.

Chosen-Plaintext-Angriff Der Kryptoanalytiker hat die Möglichkeit, beliebigen Klartext in das Kryptosystem einzuschleusen und den resultierenden Chiffretext abzufangen. Dies ist eine gebräuchliche Angriffsart bei vielen asymmetrischen Algorithmen, da hier in der Regel ein Schlüssel öffentlich verfügbar ist.

Chosen-Ciphertext-Angriff Der Kryptoanalytiker kann beliebigen Chiffretext vorgeben und den dazugehörigen Klartext abfangen. Kann der Kryptoanalytiker diesen Angriff durchführen, so ist das Kryptosystem bereits unsicher, da der Kryptoanalytiker jeden beliebigen Chiffretext entschlüsseln kann. Das Ziel bei diesem Angriff ist lediglich, den verwendeten Schlüssel zu erhalten. Dieser Angriff wird eher in der theoretischen Kryptanalyse verwendet, um neu entwickelte Algorithmen zu testen.

Brute-Force-Angriff D wird so lange mit jedem möglichen Schlüssel auf den Chiffretext angewandt, bis ein sinnvolles Resultat erzielt wird. Dabei ist zu beachten, dass die Differenzierung zwischen sinnvollen und sinnlosen Resultaten mitunter schwierig ist. Diese Tatsache kann man sich mit folgendem Beispiel vergegenwärtigen: Ein Kryptoanalytiker versucht eine Botschaft zu entschlüsseln, welche in chinesischer Sprache abgefasst ist, obwohl er dieser Sprache nicht mächtig ist. Es wird ihm aufgrund seiner mangelnden Sprachkenntnisse nicht möglich sein, zwischen einen sinnvollen chinesischen Text und Datenmüll zu unterscheiden. Eine zweite Schwierigkeit ist die bereits erwähnte Länge heute üblicher Schlüssel. Ein Schüssel mit einer Länge von 128 Bit kann $2^{128} \approx 3 \cdot 10^{38}$ mögliche Werte annehmen. Man muss im Mittel etwa die Hälfte aller Schlüssel ausprobieren, bevor man auf den richtigen stößt. Auf einem 2,4 Ghz Prozessor würde dies selbst unter der unrealistischen Annahme, dass der Prozessor die gesamte Entschlüsselung in einem einzigen Rechenschritt vornehmen kann, mehr als $2 \cdot 10^{20}$ Jahre dauern. Das Universum selbst ist dagegen gerade einmal ungefähr $1,37 \cdot 10^{10}$ Jahre alt.

Gerade bei starker Kryptographie führt jedoch in der Regel keiner dieser Angriffe zum Erfolg. Deswegen werden zum Brechen derartiger Verschlüsselungen eher unwissenschaftliche Methoden angewandt, wie z.B. Bestechung, Gewalt oder das sogenannte Social Engineering, bei dem man sich das Vertrauen des Opfers erschleicht, um es direkt oder indirekt zur Preisgabe des Geheimnisses zu bewegen. Bekannt geworden ist das Social Engineering durch die Phishing-Mails [Wikd], in denen Anwender unter irgendwelchen Vorwänden dazu aufgefordert wurden, PINs und TANs für Online Banking in gefälschten Webseiten einzugegeben.

2.6. Sicherheit von Kryptosystemen

In den vorherigen Abschnitten dieser Arbeit wurde bereits von der Sicherheit von Algorithmen und Kryptosystem gesprochen, ohne näher zu erläutern, was man eigentlich darunter versteht.

In der Praxis bezeichnet man ein Kryptosystem als sicher, wenn es Angriffen, wie sie in Abschnitt 2.5.1 beschrieben werden, widerstehen kann.

Die folgende Definition differenziert hier weiter [Ert03, Seite 24]:

Definition 2.3 *Ein Kryptosystem gilt als* ***sicher*** *oder* ***berechnungssicher****, wenn*

- *die zum Knacken erforderlichen Investitionen den Wert der verschlüsselten Daten übersteigen oder*
- *die zum Knacken erforderliche Zeitspanne größer ist als die der erforderlichen Geheimhaltung, oder*
- *wenn der Chiffretext die zum Knacken erforderliche Länge unterschreitet.*

In der Kryptographie gibt es noch einen weiteren Grad der Sicherheit [Wä04, Seite 6]:

Definition 2.4 *Sei $p(M)$ die Wahrscheinlichkeit für das Vorkommen von M und $p(C)$ die Wahrscheinlichkeit für das Vorkommen von C. $p(M|C)$ sei ferner die Wahrscheinlichkeit dafür, dass M der zum Chiffretext C gehörende Klartext ist, unter der Bedingung, dass C abgefangen wurde. Gilt nun für alle möglichen C und M*

$$p(M|C) = p(M)$$

so ist das Kryptosystem ***absolut sicher****.*

Dies bedeutet, dass der mit $D_k(M)$ erzeugte Chiffretext vom Klartext M statistisch unabhängig ist. Der Kryptoanalytiker gewinnt also aus dem Chiffretext keine Informationen über den Klartext. Diese Art der Sicherheit wird auch als **perfekte Geheimhaltung** bezeichnet.

2.6.1. Das One-Time-Pad

Es ist nur eine einzige Chiffre bekannt, welche Definition 2.4 erfüllt. Bei diesem als One-Time-Pad [Wikc] bekannten Algorithmus wird der Klartext M mit einem einmaligen, zufällig erzeugten Schlüssel K der gleichen Länge wie der Klartext verknüpft [Ert03, Sch96]: $M \oplus K$.

Diese Art der Verknüpfung nennt man XOR-Verknüpfung (*exclusive or, exklusives Oder*). Im binären Alphabet entspricht sie einer Addition zweier Zeichen, gefolgt von einer modularen Division durch 2: $(z_1 + z_1)$ *mod* 2. Es wird also die Summe der beiden Zeichen durch zwei geteilt, wobei der Restwert dieser Division als Ergebnis genutzt wird. Diese Verknüpfung wird für jedes einzelne Zeichen ausgeführt. Chiffren, die zeichenweise arbeiten, nennt man **Stromchiffren**[3]. Sie eignen sich hervorragend für Netzwerke, in denen Daten möglichst schnell und verzögerungsfrei übertragen werden sollen. Der Schlüssel wird dabei auch als **Schlüsselstrom** bezeichnet, da er eine kontinuierliche, unendliche Abfolge von Schlüsselzeichen darstellt (Abbildung 2).

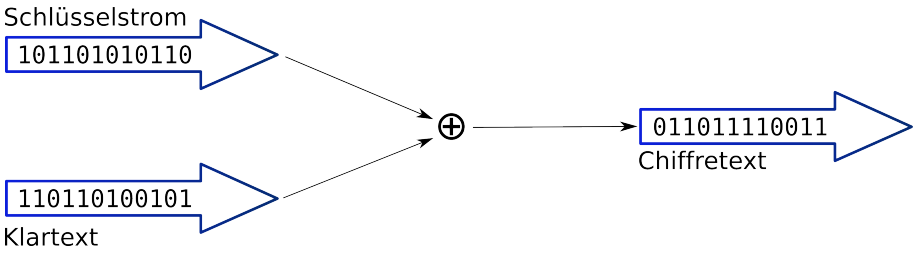

Abbildung 2: Die XOR-Verknüpfung (Sebastian Wiesner, 2007)

Problematisch beim Einsatz eines One-Time-Pads ist jedoch die Voraussetzung, dass der Schlüssel *exakt* die gleiche Länge haben muss wie der Klartext. Das bedeutet, dass bei Übertragung einer ein GB großen Datei der Schlüsselstrom ebenfalls ein GB groß sein muss. Dies ist in der Praxis kaum umzusetzen.

Aus diesem Grund gehen heutige Stromchiffren einen anderen Weg. Der Schlüsselstrom wird von einem sogenannten **Generator** erzeugt, welcher einen kontinuierlichen Strom von **Pseudozufallszahlen** bereitstellt. Diesen Strom erzeugt der Generator mit Hilfe einer besonderen Eingabe, des sogenannten **Seeds**. Dieser Seed hat eine feste Länge von in der Regel nicht mehr als 256 Zeichen. Bei gleichen Seeds erzeugt der Generator auch den gleichen Schlüsselstrom, er arbeitet also **deterministisch**. Deswegen stellt der Seed bei heutigen Stromchiffren den eigentlichen Schlüssel dar, welcher natürlich geheim gehalten werden muss.

Der Name „Pseudozufallszahl“ sagt es bereits: Mit Generatoren erzeugte Schlüsselströme sind nicht wirklich zufällig. Kennt ein Angreifer den Seed, so kann er jedes beliebige

[3] Im Gegensatz dazu arbeiten **Blockchiffren** wie DES oder AES immer auf einem Block einer bestimmten, festgelegten Menge an Zeichen.

Zeichen des Schlüsselstroms vorhersagen. Bei schlechten (sprich unsicheren) Generatoren reicht unter Umständen sogar die Kenntnis einer kleinen Abfolge von Zeichen aus dem Schlüsselstrom.

Das ist jedoch nicht das einzige Problem von Stromchiffren. Verwendet man den selben Schlüsselstrom des Generators G unter Verwendung des Seeds S zur Verschlüsselung zweier verschiedener Nachrichten M_1 und M_2, so gibt man damit Informationen über die Nachrichten preis [BGW01]:

$$\begin{aligned} C_1 &= M_1 \oplus G(S) \\ C_2 &= M_2 \oplus G(S) \\ \Rightarrow C_1 \oplus C_2 &= (M_1 \oplus G(S)) \oplus (M_2 \oplus G(S)) = M_1 \oplus M_2 \end{aligned} \tag{3}$$

Diese Schwäche ist geradezu prädestiniert für einen Known-Plaintext-Angriff. Kennt man eine Nachricht, so ergibt sich die zweite automatisch. Zudem gibt es Verfahren, die Rückschlüsse auf die enthaltenen Nachrichten ermöglichen, selbst wenn nur Teile der Nachrichten bekannt sind. Aus diesem Grund ist die bereits erwähnte Bedingung, dass der Schlüssel nur ein einziges Mal verwendet werden darf, durchaus ernst zu nehmen.

Moderne Stromchiffren sind folglich ebenso wie alle anderen Chiffren – mit Ausnahme des One-Time-Pads – nicht nach Definition 2.4 absolut sicher, sondern nur berechnungssicher nach Definition 2.3.

2.6.2. Berechnungssicherheit

Praktisch besteht die Sicherheit solcher Chiffren in der Komplexität bestimmter mathematischer Probleme. Bekannte Probleme sind zum Beispiel die Zerlegung großer Zahlen ($\gtrapprox 5 \cdot 10^{119}$) in ihre Primfaktoren [Wä04, Seite 72] (Basis des RSA Algorithmus) oder die Arithmetik auf Elliptischen Kurven[4] (Basis des ElGamal Verfahrens). Solche Probleme sind selbst für moderne Supercomputer zu komplex, um ohne Schlüssel gelöst zu werden.

Daraus folgt jedoch automatisch, dass es absolute Sicherheit in der Kryptologie – mit Ausnahme des One-Time-Pads – nicht gibt. Da sowohl die Mathematik als auch die Computer weiterentwickelt werden, darf man erwarten, dass zukünftig Computer die heute noch berechnungssicheren Algorithmen knacken können. Dennoch kann man modernen Chiffren der starken Kryptographie in der Regel vertrauen, nicht zuletzt da diese der kontinuierlichen Überprüfung durch die Fachwelt unterworfen sind.

Bedenklich sind viel mehr solche Chiffren, die nicht nach den Prinzipien der starken Kryptographie entwickelt wurden. Ein Beispiel dafür liefert das im folgenden Teil vorgestellte Wired Equivalent Privacy (WEP) Protokoll, dessen Kernbestandteil, der RC4

[4] Die Lösungsmenge einer Gleichung $y^2 = ax^3 + bx^2 + cx + d$ wird Elliptische Kurve genannt [Wä04, Seite 241]

Algorithmus, in Geheimhaltung entwickelt wurde und lange Zeit als Firmengeheimnis galt. Hier hatte die kryptologische Fachwelt keinerlei Gelegenheit, den Algorithmus noch während der Entwicklung zu überprüfen. Es handelt sich also nicht um starke Kryptographie – mit katastrophalen Folgen, wie sich zeigen wird.

3. Wired Equivalent Privacy (WEP)

WEP ist ein Standard zur Absicherung drahtloser Netzwerke. Er wurde im Jahr 1999 als Teil des Standards IEEE 802.11 [IEE99] verabschiedet, welcher den Aufbau drahtloser Netzwerke generell definiert.

3.1. Ablaufbeschreibung

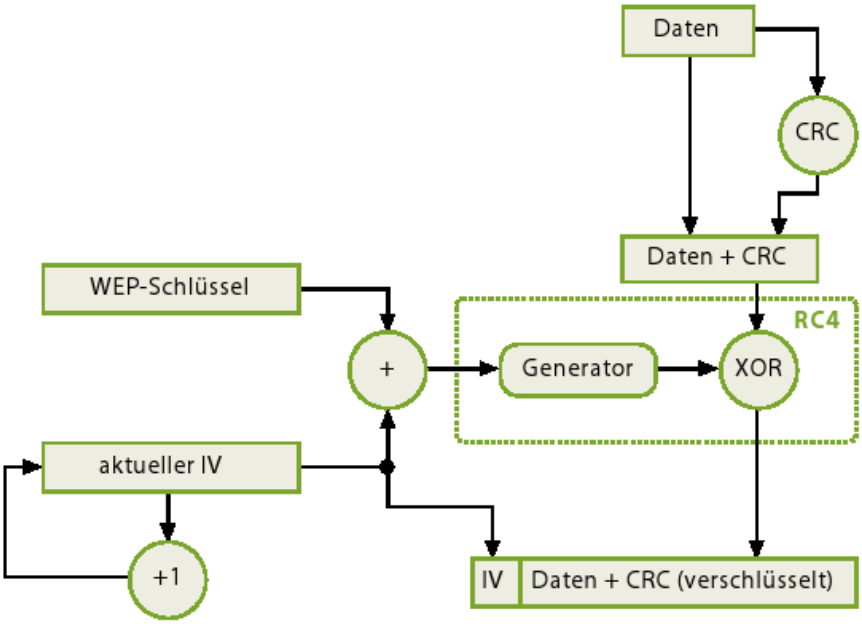

Abbildung 3: Blockdiagramm der WEP Verschlüsselung ([Arn04, Seite 215])

WEP besteht eigentlich aus zwei getrennten Operationen besteht (Abbildung 3):

1. Im linken Teil der Abbildung 3 ist die Vorbereitung des Schlüssels K schematisch dargestellt, der mit einem sogenannten Intialisierungsvektor (IV) v zusammengeführt wird (Abbildung 4). Der IV ist dabei nichts weiter als eine beliebige Zeichenkette einer Länge von 24 Bit. Durch die Verwendung eines IV soll garantiert werden, dass einzelne Datenpakete trotz gleichen Schlüssels unterschiedlich verschlüsselt werden. So soll dem bereits erwähnten Problem aller Stromchiffren

(vgl. Gleichung 3) begegnet werden. Da der IV allerdings zur Entschlüsselung benötigt wird, muss er dem Kommunikationspartner bekannt sein. Er wird deswegen der verschlüsselten Nachricht unverschlüsselt vorangestellt.

2. Im oberen rechten Teil der Abbildung 3 zeigt die Abbildung schematisch die Vorbereitung des Klartextes, der mit einer Prüfsumme (*cyclic redundancy code, CRC* [Haa]) $c(M)$ versehen wird[5]. Durch die Berechnung der Prüfsumme des Klartextes kann der Empfänger später erkennen, ob der Klartext während der Übertragung verändert wurde. Dadurch sollen zufällige (beispielsweise durch Übertragungsfehler) oder beabsichtigte Manipulationen an der übertragenen Botschaft verhindert werden.

Abbildung 4: Konkatenation eines IVs (Länge 8 Bit) mit einem Schlüssel (Länge 16 Bit) (Sebastian Wiesner, 2007)

Die Ausgabe aus der ersten Teiloperation wird nun als Seed für den Generator verwendet. Als Generator dient dabei der RC4 Algorithmus (vgl. Anhang A), welcher nun den Schlüsselstrom liefert. Dieser Strom wird anschließend mit der Ausgabe der zweiten Teiloperation XOR-verknüpft, um den Chiffretext C zu erzeugen. Diese Operationen lassen sich wie folgt zusammenfassen[6]:

$$A \rightarrow B : v \circ (P \oplus RC4(v \circ K) \text{ wobei } P = M \circ c(M) \tag{4}$$

Der WEP Algorithmus wird für jedes Datenpaket[7] getrennt durchlaufen.

3.2. Analyse der Sicherheit

Bereits bei der Standardisierung des Protokolls zeichnete sich eine Schwachstelle ab. WEP legte die Schlüssellänge auf 40 Bits fest. Dies geschah allein aus dem Grund, dass zur damaligen Zeit eine größere Länge in den USA geltende Gesetze verletzt hätte. Diese Länge bietet aber keinen ausreichenden Schutz vor Brute-Force-Angriffen. Die Autoren von [FMS02] stellten fest, dass ein solcher Schlüssel in weniger als einem Tag gebrochen werden kann. Diese Sicherheitslücke konnte nach der Aufhebung der entsprechenden Gesetze in den USA jedoch beseitigt werden, indem die Hersteller von WLAN-Hardware die Länge des Schlüssels auf 128 Bit (eigentlich 104 Bit Schlüssel sowie 24 Bit IV) erhöhten[8].

[5] Die Prüfsumme des Textes „Hallo Welt“ wäre beispielsweise 01010000001010000001010001110101.

[6] Der Kreis ∘ steht hier für eine Konkatenation. Dabei werden zwei Zeichenketten zu einer neuen zusammengefügt. So ergibt „a“ ∘ „b“ die neue Zeichenkette „ab“.

[7] Im Bezug auf technische Protokolle wie WEP heißt eine Nachricht **Paket**.

[8] Diese Änderung ging jedoch nicht in den Standard IEEE 802.11 ein. Es handelt sich also nicht um einen offiziellen Standard, sondern um einen „Gebrauchsstandard“.

Viel gravierender wirken sich zwei andere Schwachstellen aus, welche die Berechnung der Prüfsumme sowie den RC4 Algorithmus selbst betreffen. Diese Schwachstellen sind nicht mit einfachen Erweiterungen zu beheben. Dazu waren umfangreiche Korrekturen am Protokoll selbst notwendig, die in den inoffiziellen Nachfolger WPA (vgl. Abschnitt 4) eingeflossen sind.

3.2.1. Die Prüfsumme

In [BGW01] wird gezeigt, dass es möglich ist, mit WEP übertragene Pakete beliebig zu manipulieren, ohne dass dies dem Empfänger auffällt.

Dabei wird eine besondere Eigenschaft der CRC-Prüfsumme ausgenutzt:

Eigenschaft 1 *Die CRC Prüfsumme wird ohne Verwendung eines Schlüssels berechnet.*

Dies ermöglicht jeder beliebigen Person, ohne Kenntnis des WEP-Schlüssels korrekte Prüfsummen für jede beliebige Nachricht zu erzeugen. Diese Nachrichten müssen nun nur noch in das drahtlose Netzwerk eingebracht werden. Dazu eröffnen sich zwei Wege.

Der erste führt über die Manipulation abgefangener Nachrichten. Dies erfolgt unter Ausnutzung einer weiteren besonderen Eigenschaft der CRC Prüfsumme:

Eigenschaft 2 *Die CRC Prüfsumme ist linear, es gilt also:* $c(x_1) \oplus c(x_2) = c(x_1 \oplus x_2)$ *für jedes Paar* x_1, x_2.

Bei einer beliebigen übertragenen Nachricht $A = v \circ C$ gilt (Gleichung 4):

$$C = RC4(v \circ K) \oplus (M \circ c(M))$$

Es ist nun möglich, eine neue Nachricht M' zu erzeugen. Danach gilt:

$$C' = RC4(v \circ K) \oplus (M' \circ c(M'))$$

M' wird dabei als XOR-Verknüpfung der originalen Nachricht M und einer beliebig wählbaren Abweichung Δ gebildet. Um den neuen Chiffretext C' zu erhalten, berechnet man einfach:

$$C' = C \oplus (\Delta \circ c(\Delta))$$

Um zu zeigen, dass diese Gleichung korrekt ist, löst man den rechten Teil der Gleichung auf:

$$\begin{aligned} C' &= RC4(v \circ K) \oplus ((M \circ c(M)) \oplus (\Delta \circ c(\Delta)) \\ &= RC4(v \circ K) \oplus ((M \oplus \Delta) \circ (c(M) \oplus c(\Delta)) \\ &= RC4(v \circ K) \oplus (M' \circ (c(M) \oplus c(\Delta)) \\ &= RC4(v \circ K) \oplus (M' \circ c(M')) \end{aligned} \tag{5}$$

In Gleichung 5 wird die Prüfsumme der manipulierten Nachricht M' berechnet. Dazu werden die Prüfsummen der Abweichung Δ sowie der Originalnachricht M verknüpft. Gemäß der Eigenschaft 2 ergibt dies die Prüfsumme der manipulierten Nachricht M'.

Bemerkenswert ist, dass für diese Manipulation weder die Kenntnis der Nachricht M noch des Schlüsselstroms $RC4(v \circ K)$ notwendig ist!

Der zweite Weg ermöglicht ebenfalls das Einschleusen beliebiger Nachrichten. Diesmal kann die Nachricht ohne den Umweg über eine Abweichung Δ direkt erzeugt werden. Dazu ist allerdings die Kenntnis des Schlüsselstroms $RC4(v \circ K)$ (aber nicht des Schlüssels K) nötig. Den Schlüsselstrom erhält man, wenn man den Klartext P einer Nachricht mit dem Chiffretext C verknüpft:

$$P \oplus C = P \oplus (P \oplus RC4(v \circ K)) = RC4(v \circ K)$$

Problematisch ist natürlich, dass die Kenntnis des Klartextes P erforderlich ist, um diese Operation durchzuführen. In der Praxis ist dies jedoch leichter als es scheint. Ein Angreifer könnte beispielsweise dem Opfer eine E-Mail senden und darauf warten, dass diese über das drahtlose Netzwerk abgerufen wird. Dann ist er in der Lage den Schlüsselstrom zu rekonstruieren.

Mit diesem Schlüsselstrom kann er dann beliebige Nachrichten erzeugen und in das Netzwerk einschleusen:

$$C' = RC4(v \circ K) \oplus (M' \circ c(M'))$$

3.2.2. Die FMS-Attacke – Eine Schwäche in RC4

Der Key-Scheduling Algorithm (KSA) ist der erste Teil des RC4 Algorithmus, welcher eine geheime, interne Menge (S-Box) erzeugt. Aus dieser Menge entnimmt der Pseudo Random Number Generator (PRNG) später die Pseudozufallszahlen zur Verschlüsselung des Klartextes. Für eine detaillierte Darstellung des Algorithmus sei auf den Anhang A verwiesen.

Die Schwäche besteht darin, dass sich aufgrund des relativ einfachen Aufbaus des KSA unter bestimmten Bedingungen Rückschlüsse über die Bytes des Schlüssels ziehen lassen. Sind diese Bedingungen eingetreten, so wird der Zustand des RC4 Algorithmus als *resolved* (aufgelöst) bezeichnet.

Um nun die Schlüsselbytes zu berechnen, ist das Wissen um einige Bestandteile des RC4 Algorithmus nötig. Zum einen benötigt der Angreifer Kenntnis der ersten Bytes des Schlüssels. Zwei Eigenschaften des WEP Protokolls sorgen dafür, dass der Angreifer diese auch erhält:

Eigenschaft 3 *Der eigentliche Schlüssel wird durch eine Konkatenation aus dem IV und dem geheimen WEP-Schlüssel*[9] *gebildet.*

Eigenschaft 4 *Der IV wird der chiffrierten Nachricht im Klartext vorangestellt. Er ist also auch dem Angreifer bekannt.*

Dem Angreifer sind also bedingt durch diese beiden Eigenschaften die ersten 3 Byte (24 Bit) des Schlüssels bekannt. Allerdings ist nur eine bestimmte Menge an schwachen Schlüsseln betroffen (*x-good keys*). Es ist also die Kenntnis eines IVs notwendig, der zu so einem schwachen Schlüssel führt.

Zum anderen benötigt der Angreifer das erste Byte z der Ausgabe des PRNG. An dieses zu gelangen ist bereits schwieriger, da der Angreifer nur das Byte c_1 des Chiffretextes C abfangen kann. Um an z zu gelangen, müsste das erste Byte p_1 des Klartextes P ebenfalls bekannt sein:

$$\begin{aligned} c_1 &= p_1 \oplus z \\ \Rightarrow z &= c_1 \oplus p_1 \end{aligned}$$

WEP macht dem Angreifer jedoch auch hier das Leben leicht. Das erste Klartextbyte ist nämlich immer exakt gleich, wenn das WLAN IP [RFC81, RFC98] oder ARP [RFC82] Verkehr transportiert [SIR01]. Da diese beiden Protokolle in modernen Netzen grundlegend sind, ist dies fast immer der Fall; das erste Byte des Klartextes ist also immer bekannt. Nun ist es keine große Schwierigkeit, z zu berechnen.

Mit diesen Informationen versucht man nun, dass nächste Byte des Schlüssels zu „erraten“. Dazu vollzieht man die Arbeit des KSA solange nach, wie es mit der vorhandenen Information möglich ist, und erhält so einen bestimmten Zustand der S-Box. Nun nimmt man einfach an, dass das Ausgabebyte x aus einem bekannten Feld der S-Box kommt. Durch diese Vermutung erhält man nun alle nötigen Parameter, um das nächste Byte des Schlüssels zu berechnen. Allerdings ist die Annahme und folglich auch die Berechnung nicht notwendigerweise korrekt, da sowohl der KSA als auch der PRNG die S-Box im späteren Verlauf des Algorithmus noch verändern und das Ergebnis somit verfälschen könnten. Interessant ist jedoch, dass mit einer Wahrscheinlichkeit von 5% [FMS01, Seite 11] weder der KSA noch der PRNG die bekannten Felder der S-Box verändern.

Diese Eigenschaft ermöglicht es, eine Art „Abstimmung“ durchzuführen. Dazu sammelt man eine gewisse Anzahl an Nachrichten, die mit einem der benötigten Schlüssel chiffriert wurden und führt die obigen Berechnungen für jede Nachricht durch. Jede Nachricht „stimmt“ dann quasi für einen bestimmten Wert ab. Eine beispielhafte „Stimmen“-Verteilung ist in Abbildung 5 zu sehen. Der „Sieger“ dieser Abstimmung

[9] Dies ist der Schlüssel, welchen der Nutzer in der Konfigurationsoberfläche der WLAN-Hardware eingibt.

(im Beispiel das Byte mit dem Wert 19) ist dann das gesuchte Schlüsselbyte. Eine Untersuchung von 60 Nachrichten reicht aus, um den Sieger mit einer Wahrscheinlichkeit von mehr als 50% [FMS01, Seite 11] korrekt zu bestimmen.

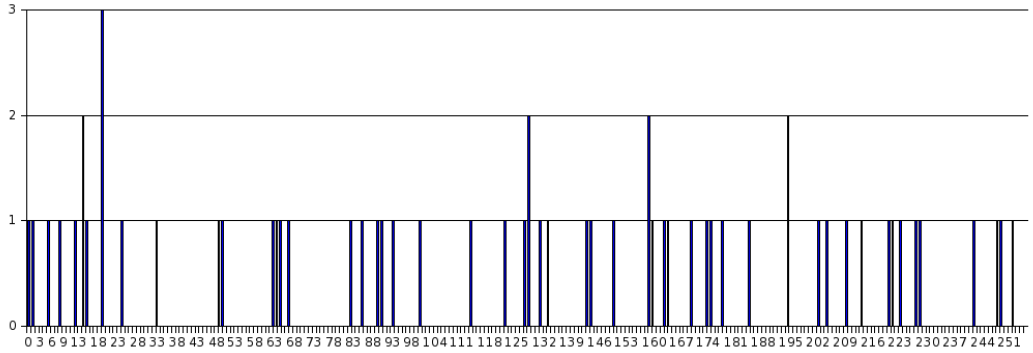

Abbildung 5: Ein beispielhaftes Ergebnis einer FMS-Attacke (Sebastian Wiesner, 2007)

Hat man nun ein weiteres Schlüsselbyte „erraten“, so zählt man dieses zum bereits bekannten Schlüsselteil dazu und beginnt von vorne. Auf diese Weise kann man Schritt für Schritt den gesamten Schlüssel rekonstruieren.

Um diese Schwachstelle auszunutzen, muss man allerdings viele Nachrichten abfangen, um eine ausreichende Anzahl an passenden Initialisierungsvektoren zu erlangen. Die Anzahl der benötigten Pakete wurde zunächst etwa vier Millionen geschätzt [FMS02, Seite 8], in der Praxis erwies sich eine Anzahl von fünf bis sechs Millionen Paketen als erforderlich [SIR01].

3.3. Schlussfolgerung

Die im vorigen Abschnitt besprochene Schwäche im RC4 Algorithmus führt dazu, dass WEP in halbwegs ausgelasteten Netzwerken innerhalb einiger Stunden vollständig aufgebrochen werden kann, wobei das Sammeln der benötigten Nachrichten die meiste Zeit benötigt. Das Berechnen der Schlüsselbytes selbst ist auf modernen Computern eine Sache von Sekunden.

Kombiniert man diesen Angriff mit der in Abschnitt 3.2.1 erläuterten Möglichkeit, beliebige Pakete in das Netzwerk einzuschleusen, kann man auch mäßig bis schwach ausgelastete Netzwerke in annehmbarer Zeit brechen. Die dazu benötigten Pakete erzeugt der Angreifer einfach selbst.

Zusätzlich wurden weitere Verbesserungen an der eigentlichen Attacke vorgenommen. Durch die KoreK-Attacke [Sch05] kann jedes beliebige Datenpaket zum Knacken verwendet werden, anstatt nur auf mit bestimmten Schlüsseln chiffrierte Pakete zurückzugreifen. So sind wesentlich weniger Pakete (zwischen 250.000 und 500.000 verschiedene Initialisierungsvektoren) für einen erfolgreichen Angriff nötig.

Diese Erkenntnissen zeigen, dass WEP weder die Geheimhaltungs- noch die Authentizitätsforderung (vgl. Abschnitt 2.3) erfüllt. Es bietet also keinerlei Sicherheit mehr. Glücklicherweise gibt es Alternativen zum Schutz eines WLANs. Der „inoffizielle“ Nachfolger des WEP Protokolls, das WPA Protokoll, ist eine davon.

4. WiFi-Protected Access (WPA)

Nach dem WEP gebrochen war, bestand die Notwendigkeit eines alternativen Protokolls zum Schutz drahtloser Netzwerke. Die Verabschiedung des nächsten Standards IEEE 802.11i [IEE04], welcher neue Sicherheitsmaßenahmen definierte, war allerdings noch nicht absehbar. Zudem war klar, dass dieser Standard mit älteren WLAN-Geräten inkompatibel sein würde. Angesichts dieser Probleme schuf die Wi-Fi Alliance [wif], ein Konsortium der führenden Hersteller von WLAN Hard- und Software, das *WPA* (Wi-Fi Protected Access Protokoll) Protokoll als „inoffiziellen“ Nachfolger von WEP. Dabei wurden einige der Ideen des neuen Standards aufgegriffen, allerdings immer unter Berücksichtigung der Kompatibilität zu älteren Geräten.

4.1. Ablaufbeschreibung

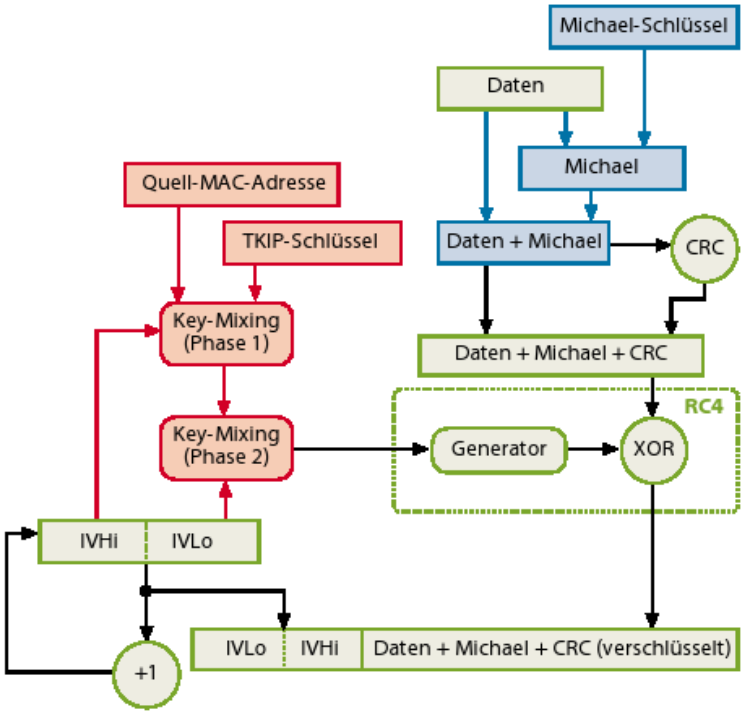

Abbildung 6: Blockdiagramm der WPA Verschlüsselung ([Arn04, Seite 216])

Man erkennt in Abbildung 6, dass die eigentliche Verschlüsselung (grün markiert) noch immer unverändert ist. Wie auch in WEP wird hier der Schlüsselstrom des RC4 Algorithmus mit dem Klartext XOR-verknüpft. Interessant sind jedoch die rot und blau markierten Bestandteile des WPA Protokolls sowie der deutlich vergrößerte IV, dessen Länge auf 48 Bit verdoppelt wurde.

Der blau markierte Teil zeigt die Änderungen bei der Berechnung der Prüfsumme. Hier wird nun zusätzlich der eigens entwickelte Michael-MIC (Message Integrity Check) eingesetzt. Die Besonderheit dieses Algorithmus besteht darin, dass er einen eigenen Schlüssel K_M zur Berechnung der Prüfsumme verwendet.

Weitere Änderungen betreffen die Erzeugung des Seeds für den RC4 Algorithmus. Hier wird nun das Temporal Key Integrity Protocol (TKIP) verwendet, um aus dem geheimen TKIP-Schlüssel einen temporären Schlüssel abzuleiten, welcher zum Chiffrieren einzelner Nachrichten dient. Der TKIP-Schlüssel wird dabei auch als PTK (Pair-Wise Transient Key) bezeichnet. Die Kommunikationspartner handeln diesen PTK beim Verbinden mit dem drahtlosen Netzwerk individuell aus. Als Basis dient dabei der sogenannte PMK (Pair-Wise Master Key). Der PMK wiederrum wird aus einer Kombination zwischen dem Namen des Netzwerks sowie des Passwortes, die der Anwender bei der Konfiguration der WLAN-Hardware angegeben hat, berechnet.

Im Zuge dieses Protokolls werden zwei sogenannte Key-Mixing-Phasen durchlaufen. Die erste Phase mischt den geheimen TKIP-Schlüssel und die Hardware-Adresse des Senders mit einem 32 Bit langen Teilstück (Hi-Teil) des IVs. Das Ergebnis dieser Operation wird nun in der zweiten Phase mit dem übrig gebliebenen 16 Bit großen Teilstück (Lo-Teil) des IV vermischt. Daraus resultiert der eigentliche Seed, welchen der RC4 Algorithmus nun für die Erzeugung der Pseudozufallszahlen verwendet.

Diese Aufteilung geschieht, weil die erste Key-Mixing-Phase relativ aufwändig ist. Da der Hi-Teil jedoch für ungefähr 65.000 Nachrichten konstant bleibt, muss die erste Phase nicht für jedes Paket durchgeführt werden. Der Lo-Teil jedoch wird für jede Nachricht verändert. So ist gewährleistet, dass jede Nachricht mit einem anderen Seed verschlüsselt wird. Es findet also ein Kompromiss zwischen der rechenintensiven Einbindung des TKIP-Schlüssels und der Variation des Seeds statt.

Da auch hier, wie bei WEP, der komplette IV (Lo- und Hi-Teil) zur Entschlüsselung notwendig ist, wird er ebenfalls dem Chiffretext unverschlüsselt vorrangestellt.

4.2. Analyse der Sicherheit

Bei der Sicherheit von WPA bietet sich ein weitaus besseres Bild. Die Schwäche der Prüfsumme (Abschnitt 3.2.1) durch die Verwendung des Michael-Algorithmus verhindert. Da dieser Algorithmus einen eigenen Schlüssel verwendet, kann ein Angreifer für manipulierte Nachrichten keine korrekte Prüfsumme erzeugen. Das Einbringen manipulierter Nachrichten in das Netzwerk fällt also sofort auf.

Auch die in Abschnitt 3.2.2 vorgestellte Schwäche wird erfolgreich beseitigt. Durch das aufwändige Key-Mixing ist sichergestellt, dass sich der RC4-Schlüssel für jedes Paket ändert. Folglich nutzt dem Angreifer die Kenntnis eines einzigen RC4-Schlüssels nichts mehr. Zudem ist auch der Angriff selbst unmöglich geworden. Durch das Key-Mixing ist sichergestellt, dass der Angreifer nicht länger Kenntnis der ersten Schlüsselbytes erlangt. Das Wissen um die ersten Schlüsselbytes kann der Angreifer auch nicht durch das Nachvollziehen des Key-Mixings erlangen. Dazu wäre der geheimen TKIP-Schlüssel nötig, welchen der Angreifer nicht besitzt. Somit ist eine wesentliche Bedingung für den Erfolg des Angriffes nicht mehr gegeben.

Allerdings ist auch WPA nicht vor Angriffen gefeit. WPA leidet an einer prinzipiellen Schwäche aller Passwort-basierten Verfahren: Ist das Passwort zu kurz, so kann der Angreifer dieses durch einen Brute-Force-Angriff einfach erraten. Dabei braucht der Angreifer in der Regel noch nicht einmal jede mögliche Zeichenkombination durchzuprobieren, da viele Nutzer einfache Passwörter wie Geburtstage oder Namen berühmter Persönlichkeiten wählen. Es reicht also, wenn der Angreifer eine Liste mit wahrscheinlichen Passwörtern anlegt und diese dann als Basis für einen Brute-Force-Angriff nutzt. Einen auf diese Weise verbesserten Brute-Force-Angriff nennt man **Wörterbuchattacke**. Die Erfolgschancen eines derartigen Angriffes sind wegen der Nachlässigkeit der meisten Nutzer sogar relativ hoch.

Angesichts dieser Gefahr wurde die minimale Länge der Passphrase in WPA auf 8 Zeichen festgelegt. Dies reicht heute allerdings nicht mehr aus, weshalb die Wi-Fi Alliance ein Minimum von 20 Zeichen empfiehlt. Auch sollte das Passwort eine möglichst zufällige Zeichenfolge sein, um Wörterbuchattacken zu erschweren.

Zusammenfassend lässt sich sagen, dass WPA die Schwächen von WEP erfolgreich beseitigt. Die Gefahr eines Wörterbuch- oder Brute-Force-Angriffes ist zwar prinzipiell gegeben, lässt sich allerdings durch die Wahl eines entsprechend sicheren Passwortes leicht vermeiden.

5. Ausblick auf WPA2

WPA ist zwar (noch) sicher, allerdings nicht standardisiert. Es handelte sich vielmehr um eine Art Zwischenlösung bis zur Einführungen des nächsten WLAN-Standards. Dieser kam im Jahr 2004 unter dem Namen IEEE 802.111i [IEE04] und definiert mit AES-CCM eine neues Protokoll zur Sicherung drahtloser Netzwerke. Von der Wi-Fi Alliance erhielt es den einprägsamen Namen WPA2.

Der Name des Protokoll deutet bereits an, dass es auf dem AES Algorithmus basiert. Dieser Algorithmus wurde im Rahmen eines öffentlichen Wettbewerbs des NIST (National Institute for Standards in Technology) nach dem Modell der starken Kryptographie entwickelt und im Jahr 2001 unter dem Namen FIPS 197 offiziell zum Standard-Algorithmus für Regierungsinstitutionen der USA ernannt [FIP01].

Trotz des hochwertigen Charakters dieses Protokolls ist nicht zu erwarten, dass WPA2 in den nächsten Jahren außerhalb von Firmen und Regierungsorganisationen weite Verbreitung finden wird. AES-CCM ist mit älterer WLAN-Hardware nicht kompatibel und würde somit somit den Austauschen aller am WLAN beteiligter Hardware-Komponenten erfordern. Da WPA selbst bisher noch nicht gebrochen wurde, stellt sich hier die Frage der Verhältnismäßigkeit gerade für den privaten Nutzer.

Mit WPA2 wird die Sicherung drahtloser Netze erstmals auf die bewährten Grundlagen der starken Kryptographie gestellt. Nach Karl Valentin gilt zwar immer noch, dass nichts sicher ist. Dennoch ist mit WPA bis auf weiteres und mit WPA2 auch mittelfristig der weitgehend sichere Betrieb von drahtlosen Netzwerken gewährleistet.

A. Der RC4 Algorithmus

RC4 wurde 1987 von Ron Rivest im Auftrag des Unternehmens RSA Security Inc. entwickelt. Er galt als Firmengeheimnis, bis er 1994 anonym auf der Cypherpunks Mailingsliste veröffentlicht wurde [cyp]. Heute ist der Algorithmus einer der meistgenutzten Stromchiffren und wird unter anderem auch im SSL Protokoll verwendet, um Übertragungen im Internet zu schützen.

RC4 läuft in zwei Schritten ab. Zuerst wird im Key-Scheduling Algorithm (KSA) eine geheime, internen Menge (S-Box) S erzeugt, welche normalerweise eine Länge von $N = 2^8 = 256$ hat. Jedes Element dieser Menge enthält dabei ein Byte. Die Ausführung des KSA ist für einen bestimmten Schlüssel K nur ein einziges Mal notwendig. Die eigentlichen Pseudozufallszahlen werden dann mittels des Pseudo Random Number Generator (PRNG) auf Basis der S-Box berechnet.

A.1. Der Key-Scheduling Algorithm

Der KSA berechnet die S-Box auf Basis des Schlüssels, indem er sie zuerst mit $S = \{0, \ldots, 255\}$ initialisiert und dann jedes Element durch ein mit Hilfe des Schlüssels ausgewähltes Element ersetzt.

Zuerst wird die S-Box einfach initialisert:

$$S = \{0, \ldots, 255\}$$

Im folgenden wird sie nun quasi-zufällig „durcheinandergewürfelt“. l sei die Länge des Schlüssels K in Byte. Für jeden Index i in S wird nun folgendes durchgeführt:

$$\begin{aligned} j &= j + S[i] + K[i \bmod l] \\ S[i] &\leftrightarrow S[j] \end{aligned} \tag{6}$$

Zu beachten ist hier noch, dass in Gleichung 6 keine einzelnen Schlüsselbits verwendet werden, sondern immer ein ganzes Byte. Deswegen ist auch die modulare Division $i \bmod l$ notwendig. Für den wahrscheinlichen Fall, dass der Schlüssel kürzer ist als die S-Box, wird so garantiert, dass der Index immer innerhalb der Schlüssellänge (in Byte) liegt. Hat der Schlüssel beispielsweise die Länge 40 Bits, also 5 Byte, so würde der Zugriff auf ein Schlüsselbyte nicht mehr möglich sein, sobald $i = 6$ gilt. Durch die modulare Division wird der Index wieder an die Schlüssellänge angeglichen: $6 \bmod 5 = 1$. Der Zugriff auf den Schlüssel erfolgt also wieder beim ersten Byte.

A.2. Der Pseudo Random Number Generator

Die zweite Operation liefert auf Basis der durch den KSA erzeugten S-Box eine Folge von Pseudozufallszahlen. Dabei werden zwei Zähler i und j mit 0 intialisiert. Um nun die nächste Pseudozufallszahl z zu erzeugen, wird folgendes durchgeführt:

$$
\begin{aligned}
i &= i + 1 \\
j &= j + S[i] \\
S[i] &\leftrightarrow S[j] && (7) \\
t &= S[i] + S[j] && (8) \\
z &= S[t]
\end{aligned}
$$

Zuerst werden die beiden Zähler berechnet. Dabei ist zu beachten, dass diese nach der Ausgabe einer Zahl nicht neu initialisiert werden. Es wird also immer mit den Werten der vorherigen Operation weitergerechnet. Nach der Änderung der Zähler werden in Gleichung 7 zwei Elemente der S-Box vertauscht. Dadurch wird gewährleistet, dass sich die S-Box während der Laufzeit des Algorithmus ändert. Nun kann in Gleichung 8 die Zufallszahl berechnet werden. Dazu werden die Werte zweier Elemente der S-Box zu einem neuem Index t addiert. Das Element des S-Box, welches an diesem Index liegt, wird dann als Pseudozufallszahl ausgegeben. Überschreitet i die Länge der S-Box, so werden i und j erneut mit 0 initialisiert.

Nun erklärt sich auch die Notwendigkeit der Tauschoperation in Gleichung 7. Würde diese nämlich entfallen, so bliebe die S-Box konstant. Dadurch würde der PRNG für jeden kompletten Durchlauf der S-Box (bis zur erneuten Initialisierung von i) exakt die selbe Folge liefern. Bei einer S-Box Länge von 256 Byte wäre also jede 256ste Zahl immer gleich.

Literatur

[Abo] ABOBA, Bernard: *The Unofficial 802.11 Security Web Page*. http://www.drizzle.com/~aboba/IEEE/, Abruf: 16.10.2006

[Ahl04] AHLERS, Ernst: *Angriffe auf WPA*. Version: 2004. http://www.heise.de/security/artikel/53014, Abruf: 18.11.2006

[Arn04] ARNOLD, Alfred: Jenseits von WEP – WLAN-Verschlüsselung durchleuchtet. In: *c't* (2004), Nr. 21, S. 214–219

[BGW01] BORISOV, Nikita ; GOLDBERG, Ian ; WAGNER, David: Intercepting Mobile Communications: The Insecurity of 802.11 (DRAFT). 2001. – Forschungsbericht

[BSI03] BUNDESAMT FÜR SICHERHEIT IN DER INFORMATIONSTECHNIK: Sicherheit im Funk-LAN (WLAN, IEEE 802.11). 2003. – Forschungsbericht

[cyp] *Thank you Bob Anderson*. http://cypherpunks.venona.com/archive/1994/09/msg00304.html. – Der Quellcode des RC4 Algorithmus

[Ert03] ERTEL, Wolfgang: *Angewandte Kryptographie*. Fachbuchverlag Leipzig im Carl Hanser Verlag, 2003

[FIP01] Norm FIPS 197 November 2001. *Announcing the Advanced Encryption Standard (AES)*

[FMS01] FLUHRER, Scott ; MANTIN, Itsik ; SHAMIR, Adi: Weaknesses in the Key Scheduling Algorithm of RC4. 2001. – Forschungsbericht

[FMS02] FLUHRER, Scott ; MANTIN, Itsik ; SHAMIR, Adi: Attacks on RC4 and WEP. 2002. – Forschungsbericht

[Haa] HAASE, Jan: *Wie funktioniert CRC?* http://www.informatik.uni-frankfurt.de/~haase/crc.html, Abruf: 03.01.2007

[Hei] *Unverschlüsseltes WLAN hat Folgen*. http://www.heise.de/newsticker/meldung/77921, Abruf: 09.01.2007

[IEE99] Norm ANSI/IEEE Std. 802.11 April 1999. *Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*

[IEE04] Norm IEEE Std. 802.11i Juli 2004. *MAC Security Enhancements*

[Man] MANTIN, Itsik: *RC4 Page*. http://www.wisdom.weizmann.ac.il/~itsik/RC4/rc4.html, Abruf: 18.11.2006

[NIS] *Overview of the AES Development Effort*. http://csrc.nist.gov/CryptoToolkit/aes/index2.html#overview, Abruf: 27.12.2006

[RFC81] Norm RFC 791 September 1981. *Internet Protocol*

[RFC82] Norm RFC 826 November 1982. *An Ethernet Address Resolution Protocol*

[RFC98] Norm RFC 2460 Dezember 1998. *Internet Protocol, Version 6 (IPv6)*

[Sch96] SCHNEIER, Bruce: *Angewandte Kryptographie: Protokolle, Algorithmen und Sourcecode in C*. Addison-Wesley, 1996

[Sch02] SCHNEIER, Bruce: *Secrecy, Security, and Obscurity*. Version: 2002. `http://www.schneier.com/crypto-gram-0205.html#1`, Abruf: 27.12.2006

[Sch05] SCHMIDT, Michael: *Der WEP-Wall bricht*. Version: 2005. `http://www.heise.de/security/artikel/59098`, Abruf: 18.11.2006

[SIR01] STUBBLEFIELD, Adam ; IOANNIDIS, John ; RUBIN, Aviel D.: Using the Fluhrer, Mantin, and Shamir Attack to Break WEP / AT&T Labs. 2001. – Forschungsbericht

[Tak] TAKAHASHI, Takehiro: WPA Passive Dictionary Attack Overview. `http://www.personalwireless.org/tools/WPA-Cracker/WPA_Passive_Dictionary_Attack_Overview.pdf`, Abruf: 18.11.2006. – Forschungsbericht

[wif] *Wi-Fi Alliance*. `http://www.wi-fi.org`

[Wika] *Auguste Kerckhoffs*. `http://de.wikipedia.org/wiki/Kerckhoffs`, Abruf: 27.12.2006

[Wikb] *Kerckhoffs-Prinzip*. `http://de.wikipedia.org/wiki/Kerckhoffs-Prinzip`, Abruf: 27.12.2006

[Wikc] *One-Time-Pad*. `http://de.wikipedia.org/wiki/One-Time_Pad`, Abruf: 30.12.2006

[Wikd] *Pishing*. `http://de.wikipedia.org/wiki/Pishing`, Abruf: 30.12.2006

[Wike] *RC4*. `http://de.wikipedia.org/wiki/RC4`, Abruf: 10.09.2006

[Wikf] *Wi-Fi Protected Access*. `http://de.wikipedia.org/wiki/Wi-Fi_Protected_Access`, Abruf: 10.09.2006

[Wikg] *Wired Equivalent Privacy*. `http://de.wikipedia.org/wiki/Wired_Equivalent_Privacy`, Abruf: 10.09.2006

[Wä04] WÄTJEN, Dietmar: *Kryptographie: Grundlagen, Algorithmen, Protokolle*. Spektrum Akademischer Verlag, 2004

Erklärung

Ich erkläre hiermit, dass ich die Facharbeit ohne fremde Hilfe angefertigt und nur die im Literaturverzeichnis angeführten Quellen und Hilfsmittel benützt habe.

Ich bin damit einverstanden, dass diese Facharbeit in der Lehrer- und Kollegiaten-Bücherei veröffentlicht wird.

Gilching, den 17. Mai 2007

Unterschrift des Kollegiaten