Seminararbeit
Datensicherheit und Authentizität in Sensornetzwerken

Gerrit-Arthur Gruben
Fachbereich Mathematik & Informatik
Freie Universität Berlin

***Abstract*—Mit ihren beschränkten Ressourcen fallen für drahtlose Sensornetzwerke bewährte Methoden aus der Computer-/Netzwerksicherheit weg. Ich gehe der Frage nach, welche Herausforderungen sich im Bezug auf Sensornetzwerke besonders darbieten und welche Angriffe auf diese besonders gefährlich sind. Erste Lösungsansätze werden genannt und kategorisiert, vor allem für das Schlüsselmanagement und DoS-Attacken. Zwei Architekturen für TinyOS, TinySec und LRSA[1], werden genauer betrachtet und verglichen. In den Fällen, wo ein Sensornetzwerk keine Informationssicherheit braucht, ist TinySec LRSA vorzuziehen, letzteres ist jedoch effizienter. Dabei wird in dieser Arbeit auf aktuelle Forschungsarbeiten bezug genommen und die Ergebnisse werden klar zusammengestellt. Es werden auch Vorschläge zur Erweiterungen dieser Architekturen gemacht, wie zum Beispiel die Implementierung von dem Advanced Encryption Standard (AES) in TinySec.**

***Index Terms*—Wireless Sensor Networks, Security, Authentification**

I. Einleitung

Die Anwendungsmöglichkeiten von Sensornetzwerken sind vielfältig, einige davon kritisch, wie militärische Anwendungen [1]. Es ist daher umso wichtiger die Aufmerksamkeit auf die Sicherheitsaspekte in Sensornetzwerken zu richten. Sensornetzwerke unterscheiden sich von klassischen Netzwerken vor allem durch die schiere Anzahl der Netzwerkknoten und ihrer Leichtgewichtigkeit. Das heisst praktisch gesehen begrenzte Energie und geringer Rechenleistung. Das ist selbstverständlich wenig förderlich für die Software-Implementierung aufwändiger Sicherheitsverfahren, die u.A. Rechenressourcen an den Knoten, eine höhere Kommunikationslast oder zusätzliche Hardware benötigen. Da die Sensoren gleichzeitig aber auch billig sein sollen, ist es mitunter schwierig einen integrierten Schutz für physische Zugriffe umzusetzen. Dieser Sachverhalt erzeugt Zielkonflikte beim Design von Sensornetzwerken zwischen Ökonomie, Sicherheit und Leistungsfähigkeit der Sensorknoten. Es muss also gezielt abgewägt werden, welches die kritischen Probleme des jeweiligen Anwendungsfeldes sind und welche Maßnahmen umsetzbar sind.

Daher erarbeite ich in diesem Papier zum einen einen Überblick über mögliche Bedrohungen. Dabei stellt sich heraus, dass die Sicherheitsgefahr immens ist und sich dieses wesentlich aus den Eigenschaften der Sensornetzwerke ableitet. Zum anderen werden mögliche Angriffe mit zugehörigen Abwehrmaßnahmen aufgelistet, kategorisiert und besprochen. Unter genauerer Betrachtung fällt der aktuelle Stand der Forschung, welche Systeme bisher implementiert wurden und welche Fragen offen sind. Zweifelsohne ist die Frage nach Sicherheit in Sensornetzwerken eine interessante, hat sie doch einen wichtigen Anwendungsbezug. Denn ohne Sicherheit zu bieten, fallen einige Anwendungsbereiche ganz weg. So zum Beispiel die militärische Auskundschaftung, als auch das automatisierte Ablesen vom Gas-/Wasser- und Stromverbrauch in privaten Haushalten.

Dazu werden im ersten Teil die grundsätzlichen Probleme genannt und erläutert, die vor allem in Sensornetzwerken auftreten. Zusätzlich werden Sensornetzwerke klar von anderen Netzwerktypen abgegrenzt. Daraufhin werden im zweiten Teil eine Palette von Lösungen besprochen: konzeptuelle, methodische und fertige Implementierungen, wie TinySec oder die Lightweight Rabbit based Security Architecture (kurz LRSA), die auf den RABBIT-Stromchiffre basiert. Es werden zusätzlich Abwehrmaßnahmen zu Denial-of-Service-Angriffen und Schlüsselverteilungsverfahren vorgestellt. Im dritten Teil werden die vorher beschriebenen Methoden und Implementierungen im Kontext der im ersten Teil vorgestellten Problemaspekte betrachtet und diskutiert. Die Darstellung des Themas wird mit einer Zusammenfassung über den derzeitigen Stand der Technik und einem Ausblick beendet, sowie Vorschläge zur konkreten Benutzung der beiden vorgestellten Architekturen TinySec und LRSA gemacht.

II. Aspekte der Sicherheit in Sensornetzwerken

Die Sicherheitsproblematik bei Netzwerken ist kein neues Thema, die Kryptographie als Teilaspekt existiert zum Beispiel schon seit mehr als 2000 Jahren, und es gibt viele zuverlässige, praktisch relevante Lösungen auf die man zurückgreifen kann. Die in dieser Arbeit geforderten Ziele für die Sicherheit von Sensornetzwerken sind die Informationssicherheit, Datenintegrität, Nachrichtenaktualität und Verfügbarkeit zu gewährleisten. In einigen Sensornetzen sollten auch die Selbstorganisation, Synchronisierung, Lokalisierung und Authentizität geschützt werden.

Es werden nun die vier wichtigen Sicherheitsziele für Sensornetzwerke präziser formuliert. Informationssicherheit bedeutet, dass Unbefugten das Lesen der Daten verwehrt bleibt. Datenintegrität beschreibt analog den Schreibefall, ohne die notwendigen Rechte soll es also nicht möglich sein, Daten zu schreiben. Wohingegen die Nachrichtenaktualität das Zeitintervall, in der Nachrichten ankommen sollten, beschränkt. Es wird also eine Garantie ausgesprochen bis zu welchem Zeitpunkt nach Versenden einer Nachricht diese ankommen muss. Die Verfügbarkeit eines Sensornetzes ist im Kontext der Netzintegrität zu sehen. Die Sensorknoten sollen also möglichst stets erreichbar sein.

Die Intention dieser Seminararbeit ist es sich der Probleme bewusst zu werden, die ein Erreichen dieser Sicherheitsziele verhindert. Ein erstes Konzept ist eine Selbstorganisation des Sensornetzwerkes. Sie umschliesst weitere Konzepte, wie eine dynamische Integration neuer Knoten, Verwaltung

[1] Lightweight Rabbit based Security Architecture

einer Vertrauensstruktur oder das Schlüsselmanagement. Fehlt diese, so könnte ein Angreifer beispielsweise einen bösartigen Knoten einschleusen und das System kompromittieren. Es ist daher von Nöten für die Authentizität in Sensornetzwerken zu sorgen, die aber im Kontext der begrenzten Ressourcen zu einer Herausforderung anwächst, da Authentizitätssysteme oft auf teure Public-Key-Verfahren basieren. Die meisten Sensornetzwerke verfügen über eine Funktion, die die Kommunikationseinheit von den Sensorknoten zum sparen von Energie abschaltet. Die zeitliche Synchronisation dieser Funktion bezeichne ich kurz mit Synchronisierung. Ein Sicherheitsproblem kann dies im Rahmen von Denial-of-Service Attacken werden, wenn ein Angreifer einen Knoten dazu zwingt, permanent den Sender und Empfänger aktiviert zu haben, welches ein Großteil seiner Energie verbraucht, wie man in [1] nachlesen kann. Häufig müssen Sensornetzwerke dazu fähig sein, bestimmte Knoten zu lokalisieren. Diese Lokalisierung muss auch vor bösartigen Absichten geschützt werden, um bestimmte Angriffe, wie eine Maskerade zu vermeiden. All diese Konzepte müssen zum einen sicher sein, aber sie können auch zur Umsetzung der postulierten Sicherheitsziele dienen. So können clevere Selbstorganisationsverfahren das Netz vor der Injektion bösartiger Knoten schützen.

Die Sensornetzwerke sind klar von anderen Netzwerktypen abzugrenzen, um dies zu begründen, liste ich einige fundamentale Unterschiede zu anderen Netzwerktypen auf:

- Sensornetzwerke verfügen über tausende von Netzwerkknoten, dementsprechend schwierig ist es Protokolle für die Sicherungs-, Vermittlungs- und Transportschicht zu entwickeln, die zugleich den notwendigen Effizienz- und Sicherheitskriterien entsprechen. Die Verteilung von Schlüsseln für kryptographische Verfahren ist eine zusätzliche Schwierigkeit, da einige Verfahren nicht effizient skalierbar sind.
- Die Knoten sind mit vergleichsweiser schwacher Hardware ausgerüstet, daher verfügen sie nur über eine schwache Kommunikationseinheit, wenig Rechenleistung und begrenztem Speicher. Was auf üblichen Rechensystemen performant läuft, muss dies bei Sensorknoten nicht mehr tun. So berichtet [2] von mehr als 30 Sekunden Vollauslastung des CPU für eine Durchführung des Diffie-Hellman Schlüsselaustausches (siehe Glossar). Desweiteren sind Störsender für die Kommunikation der Sensorknoten ein Problem, da diese wegen ihrer vergleichsweise schwachen Sende- und Empfangseinheit besonders anfällig darauf reagieren.
- Die Knoten verfügen nur über beschränkte Energiereserven, womit ein vorsichtiger Umgang mit den Ressourcen erzwungen wird. Dass die Knoten ausfallen können, stellt eine weitere Problematik beim Entwurf sicherer Systeme dar, da dynamische Lösungen benötigt werden.
- Physikalische Angriffe auf Sensoren sind einfacher durchzuführen als bei klassischen Systemen. Dementsprechend können auf dem Speicher liegende Daten - wie z.B. Schlüssel für kryptographische Verfahren - ausgelesen werden und für bösärtige Zwecke verwendet werden.

In Routingaspekten sind Sensornetzwerken aufgrund ihrer Dynamik den drahtlosen Ad-Hoc-Netzwerken ähnlich. Ferner, kann man davon ausgehen, dass es vergleichsweise mächtige Basisstationen unter den Sensorknoten gibt. Diese sind unter anderem für die Datenaggregration zuständig. Oft wird ihnen auch eine zentrale Rolle in hierarchischen Lösungsansätze für das Routing oder für das Schlüsselmanagement zugewiesen. Daher ist es besonders gefährlich, wenn eine Basisstation kompromittiert wird.
Es darf nicht von leistungsfähigeren Knoten ausgegangen werden, denn aus ökonomischen Gründen ist eher davon auszugehen, dass der technische Fortschritt für billigere Stückpreise der Sensoren statt für bessere Hardware genutzt wird. Es werden daher gut skalierbare, sowie effiziente, aber dennoch sichere Verfahren benötigt.

Die möglichen Bedrohungen für Sensornetzwerke werden im folgendem strukturiert wiedergegeben. Dabei wird von der Bitübertragunsschicht bis zur Anwendungsschicht von unten nach oben vertikal im OSI-Modells vorgegangen. Die Sitzungs- und Darstellungsschicht wird klassischerweise ausgelassen, da sie für Sensornetzwerke keine Rolle spielen. Es wurde diese Vorgehensweise gewählt, da per Entwurf die Schichten des OSI-Modells logisch strukturiert sind und vor allem weitesgehend abgeschlossen, weshalb es eine sinnvolle Strukturierung liefert. Im wesentlichen halte ich mich an die Darstellung aus den Übersichtsartikeln von Xiao et al. ([2]) und Avancha et al. [3]. Der Begriff des bösartigen Knotens bezeichnet in der folgenden Beschreibung einen vom Angreifer eingesetzten oder übernommenen Knoten. Eventuell ist auch ein durch äußere Einwirkungen beschädigter Knoten bösartig, da er sich unkonventionell verhalten könnte, ohne einen Angreifer mit dementsprechenden Absichten als Verursacher zu haben. Die Schäden sind in diesen Fällen möglicherweise jedoch die gleichen.

A. Bitübertragungsschicht

In der Bitübertragungsschicht stellt sich die Leichtgewichtigkeit der Kommunikationseinheiten der Sensorknoten, als auch die Möglichkeit zum physikalischen Zugriff von Angreifern als das grundlegende Problem für die Sicherheit heraus. Eine weitere Charakteristika von Sensornetzwerken ist, dass weniger Bandbreite auf dem Frequenzband verglichen zu normalen Funknetzwerken genutzt wird, daher kann ein Angreifer auch nur eine kleinere Anzahl an Pakete pro Sekunde injezieren beziehungsweise abhören.

Physikalischer Zugriff: Ein mögliches Anwendungsgebiet von Sensornetzwerken besteht in der militärischen Auskundschaftung des Gegners in der die Sensoren hinter feindlichen Linien liegen, sodass die Angreifer Sensoren aufspüren können, um mechanischen Zugriff auf

diese zu gelangen. Dies kann dazu führen, dass Schlüssel für symmetrische Kryptographieverfahren oder Daten für die Authentifizierung ausgelesen oder die Software des Sensorknotens manipuliert werden, welche weitere Angriffe in anderen Schichten ermöglicht. Da die Reichweite der Sensoren beschränkt ist, bietet die Übernahme eines Knotens die Möglichkeit, alle darüber laufenden Routingpfade abzuhören. Die Sensoreinheiten können auch einfach zerstört oder beschädigt werden, sodass diese ganz oder teilweise ausfallen und ihre Funktion nicht mehr korrekt warnehmen können. Besonders schädlich sind Attacken auf die Knoten, die Daten aggregrieren, da diese eine Art Sammelpunkte sind, deren Ausfall die Funktion des Sensornetzwerkes außer Kraft setzen würde.

Abhören des Kommunikationskanals: Das klassische Eavesdropping ist auch bei Sensornetzwerken ein Problem. Es ist jedoch wegen der schieren Anzahl an Knoten schwierig dies gezielt zu tun, sodass man als Angreifer Filtertechniken oder eine Traffic-Analyse nutzen muss, um die zentralen Knoten des Netzwerks ausfindig zu machen und gerade die abzuhören, denn diese beinhalten die interessantesten Informationen.

Störungen: Das induzieren von Störsignalen (Jamming) ist, wegen der schwachen Kommunikationseinheit von Sensorknoten, ein Problem besonders für das Routing, also der Vermittlungsschicht, weil die Kommunikation in ganzen Gebieten ausgesetzt werden kann. Dies kann auch dazu genutzt werden in einer Denial-of-Service (DoS) Attacke die Energiereserven von Knoten gezielt zu erschöpfen, falls diese mehrfach versuchen ihre Datenpakete zu schicken. Man unterscheidet ferner zwischen einer permanenten (constant jamming) und einer periodischen (intermittent jamming) Störung. Die zweite Methode führt dazu, dass die gestörten Teile des Netzwerkes nicht so schnell als solche erkannt werden können, sodass der Verkehr umgeleitet werden muss. Folglich werden weiterhin Datenpakete durch Gebiete mit höherer Fehlerrate dirigiert und die Fehlerrate steigt global, was die Effizienz verringert.

B. Sicherungsschicht

In der Sicherungsschicht bietet sich vor allem die Möglichkeit mehrerer DoS Angriffe.

DoS über Kollisionen: Anstatt ein gesamtes Gebiet mit Störsendern auszuschalten, kann man dazu übergehen gezielter Kollisionen zu erzeugen mit Hilfe eines bösartigen Knotens. Der Unterschied ist, dass hier flexible Routingalgorithmen eventuell das Gebiet nicht als für die Kommunikation gestört erkennen.

DoS über multiple Anforderungen: Es wird hier von einem bösartigen Knoten mehrfach vorgetäuscht, ein Paket nicht empfangen zu können. Falls der Automated-Repeat-Request (ARQ) Modus im Netzwerk genutzt wird, führt dies zur einer Erschöpfung des permanent sendenden Knoten.

C. Vermittlungsschicht

Es tut sich beim Routing in Sensornetzwerken vor allem das Problem auf, dass der Datenverkehr in der Nähe von den Basisstationen zunimmt und diese damit erkennbar macht. Man kann jedoch nicht davon ausgehen, dass jeder Knoten eine eigene Adresse hat, stattdessen muss man von einer Adressierung ausgehen, die konditionsbasiert arbeitet (z.B. 'Alle Knoten in Reichweite' oder 'Alle Knoten, die eine Temperatur $\geq 30°C$ messen).

Traffic-Analyse: Der Verkehr im Sensornetzwerk kann analysiert werden, um die für den Angreifer wichtigen Basisstationen zu erkennen oder Regionen mit einer besonders hohen Verkehrsdichte ausfindig zu machen um diese durch Störsender auszuschalten.

Manipulierte Weiterleitung: Übernomme Knoten können dahingehend modifiziert werden, dass sie Pakete an bestimmte Adressaten nicht weitersenden oder gar nur bei denen modifizieren. Dies macht es besonders schwer für Algorithmen, die auf Vertrauen zwischen Knoten basieren. Da nur ein Teil der Pakete manipuliert wird und ein dynamisches Erkennen von bösartigen Knoten somit deutlich schwieriger wird.

Maskerade: Ein bösartiger Knoten 'verkleidet' sich um eine bestimmte Funktion einzunehmen. Im Falle der Vermittlungsschicht könnte er sich zum Beispiel als Basisstation identifizieren und die ankommenden Pakete verwerfen oder den Zugriff von außen, der über die Basisstationen verläuft, blockieren.

Sybil-Attacke: Die Sybil-Attacke kann man zusammenfassend beschreiben, als dass von einem bösartigen Knoten mehrere Identitäten angenommen werden um das Netzwerk zu schaden. Die möglichen Angriffsziele sind Routingalgorithmen, Datenaggregation, wahlbasierte Systeme, faire Ressourcenzuteilung und Systeme zur automatischen Fehlverhaltenserkennung. Die Sybil-Attacke kann auf die Datenaggregation, Manipulation wahlbasierter Systemen und der Ressourcenzuteilung beschaffen abzielen. Der bösartige Knoten des Angreifers emuliert mehrere Knoten um damit mehr Einfluss/Kontrolle über das Netzwerk zu erlangen. Ein Angriff auf das Routing besteht daraus, dass durch die Übernahme mehrerer Identitäten, viele Routingpfade über den bösartigen Knoten gehen. Das ermöglicht dem Angreifer weitere potentielle Man-in-the-Middle-Attacken. Sollte eine automatische Fehlverhaltenserkennung im Sensornetzwerk vorhanden sein, so müssen alle Identitäten des Angreiferknotens aufgespürt werden.

Nachspielattacke: Die Nachspielattacke ist eine Man-in-the-Middle Attacke, in der der Angreifer eine Nachricht abfängt und später in genau dieser Form nocheinmal sendet. Im Normalfall ist davon auszugehen, dass der Empfänger diese dann auch annimmt, selbst wenn sie verschlüsselt ist. Ein Beispiel wäre ein Verteidigungssystem zur Erkennung von feindlichen U-Booten. Ein U-Boot eines Angreifers könnte die Meldung 'Keine Bedrohung vorhanden' abhören und falls es in Reichweite der Sensoren kommt dies wiederholt zur Basisstation schicken um sich den Weg durch das Verteidigungssystem zu ebnen.

D. Transportschicht

Sollte ein Sensornetzwerk verbindungsorientiert sein, so ist eine weitere Denial-of-Service Attacke möglich: Es werden

mehrere Verbindungen, die beiden Kommunikationsparteien Ressourcen kosten, aufgebaut, um die Speicherreserven eines Knoten gezielt zu erschöpfen. Vor allem in Sensornetzwerken, die vom Internet aus ansprechbar sein sollen, unterstützen diese in der Regel eine Form von TCP/IP, sodass diese DoS-Attacke sogar aus dem Internet möglich wäre.

E. Anwendungsschicht

Auf der Anwendungsschicht befinden sich in Sensornetzwerken im wesentlichen Verwaltungssysteme für die Sensorknoten um diese von außen zu steuern oder Programmcode dynamisch nachzuladen. Das Schlüsselmanagement zählt auch zur Anwendungsschicht. Eine Übernahme des Angreifers solcher Systeme ist in verheerend, denn es erlaubt ihm die gleiche Kontrolle, wie des Netzwerkbesitzers. **Übernahme der Netzwerkverwaltung**: Falls das Sensornetz eine Möglichkeit zur Fernsteuerung von Knoten besitzt, so kann der Angreifer versuchen dies zu übernehmen. Die Folgen sind katastrophal, denn er hat damit das ganze Sensornetzwerk übernommen.
Angriff auf das Schlüsselmanagement: Sollte der Angreifer es schaffen, Teile des Schlüsselmanagements zu übernehmen, so verfügt er zum einen über die Möglichkeit gezielt Schlüssel, die er kennt, zu injezieren, sodass er den betreffenden Teil der Kommunikation abhören kann. Es erleichtert ihm auch die Integration von bösartigen Knoten, da er diese mit Schlüsseln ausstatten kann.
Modifikation von Programmcode: Werden Aktualisierungen für den Programmcode dynamisch nachgeladen, so kann der Angreifer dies über Man-in-the-Middle Attacken in einer für seine Zwecke förderlichen Hinsicht manipulieren. Daher ist darauf zu achten, die Integrität und Authentizität der Updates zu schützen.

F. Zusammenfassung

Charakteristisch für die Attacken auf Sensornetzwerke ist es, dass neben den klassischen Methoden vor allem solche zu finden sind, die gezielt die intrinsischen Schwächen der Sensornetzwerke direkt (Energiereserven, schwache Kommunikation) oder indirekt (Schwächen in Protokollen) ausnutzen. Es wurden Methoden vorgestellt, die insbesondere die Informationssicherheit, Datenintegrität und Authentizität angreifen oder auch Techniken zum Ausschalten einzelner Sensoren, zur Übernahme gewisser Teilmengen des Systems und deren Möglichkeiten zur weiteren Ausnutzung, sowie zur Injektion eigener Knoten mit den dazugehörigen sich ergebenden Angriffstechniken, vorgestellt. Zu beachten ist, dass sich je nach Typ des Sensornetzwerkes verschiedene Attacken ergeben. Zu Gute kommt dem Angreifer, wie im nächsten Abschnitt beschrieben, dass bisher keine fertigen Komplettlösungen existieren, die sämtliche potentielle Angriffe ausschliessen. Denn schließlich wären einige Schwächen (beschränkte Energie) nur durch mehr Kosten entfernbar: die Verantwortlichen für die Netzwerkonstruktion müssen hier also eine Entscheidung treffen, zwischen wirtschaftlichen und sicherheitstechnischen Aspekten.

III. State-of-the-Art Methoden und Lösungen

Es werden nun die wichtigsten Gegenmaßnahmen gegen die im vorherigen Abschnitt vorgestellten Angriffe genannt. Weitere Referenzen findet man unter [2] und [3]. Ferner wird TinySec [4] vorgestellt, welches eine Architektur für die Sicherheitsschicht von drahtlosen Sensornetzwerken ist. Es handelt sich hierbei um eine schon fertig implementierte Lösung. Daraufhin wird der RABBIT-Stromchiffre vorgestellt, der in [5] beschrieben ist. Eine darauf aufbauende Architektur, die 'Lightweight Rabbit based Security Architecture' (LRSA), beschrieben in [6], wird behandelt. Mit einer Schlüssellange von 128 bit und einer Initialisierungsvektorlänge von 64 bit und bisherigen Kryptoanalysen stand haltend, stellt sich heraus, dass der RABBIT-Stromchiffre effizient gute Sicherheit bietet. Wir integrieren bei der Vorstellung von der LRSA und TinySec die Ergebnisse der Performance-Analysen aus den Originalarbeiten der beiden Architekturen.

A. Schlüsselverteilung

Das Problem der Schlüsselverteilung ist es, den einzelnen Knoten Schlüssel für symmetrische Verfahren zum Chiffrieren zur Verfügung zu stellen. Traditionell wird dies über teure asymmetrische Verfahren getan. Ein Beispiel ist, der im Glossar dargestellte Diffie-Hellman Schlüsselaustausch. Die Berechnung asymmetrischer Verfahren ist jedoch für Sensornetzwerke derart aufwändig, dass ein damit durchgeführter Schlüsselaustausch bei jedem Verbindungsaufbau unpraktisch wäre. Unabhängig vom Problem der Schlüsselverteilung werden in [7] symmetrische Verfahren für die Anforderungen von Sensornetzwerken gegenüber gestellt, die Gewinner sind je nach Kategorie (Geschwindigkeit, Speicherlast) die Chiffren MISTY1, RC5-32 und Rijndael (AES). Notwendig für diese ist es jedoch, dass die Kommunikationspartner einen Schlüssel teilen, wobei dies gegen Abhörattacken (Eavesdropping) abgesichert sein muss.

Ein Konzept probabalistischer Natur für den Schlüsselaustausch durchzuführen, ist es eine Menge von Schlüsseln $\mathcal{K}$ zu erzeugen und jedem Sensorknoten i eine Teilmenge $A_i \subseteq \mathcal{K}$ zukommen zu lassen. Zwei Knoten i, j können genau dann kommunizieren, wenn $A_i \cap A_j \neq \emptyset$. [2] verweist auf weitere Ergänzungen zu diesen Konzept, die vor allem eine dynamische Anpassung der Schlüssel erlaubt, um zum Beispiel auf einer Kompromittierung einzelner Knoten reagieren zu können. Zwei weitere Lösungen sind erstens das LEAP Protokoll von Zhu et al. [8], wo jeder Knoten einen Ausgangsschlüssel erhält und damit verschlüsselt weitere ausgetauscht werden können. Zweitens das PIKE Protokoll [9] von Chan und Perrig, wo über eine Art binäre Vertrauensrelation über die Menge der Knoten über transitive Abschlüsse sichere Routen berechnet werden. Ein weiteres interessantes Konzept ist eine hybride Lösung in der die Basisstationen und die Sensorknoten logisch getrennt werden und die mächtigeren Basisstationen einen Großteil des Aufwandes übernehmen, für weiteres siehe [10].

B. Denial-of-Service-Attacken

Die möglichen Denial-of-Service-Attacken auf Sensornetzwerke sind wegen der beschränkten Energiereserven besonders gefährlich, deshalb werden einige Verteidigungsmethoden kurz vorgestellt. Um gezielten Störungen zu begegnen, kann man das Routingprotokoll so anpassen, dass die gestörten Regionen in den Routinalgorithmen des Sensornetzwerkes erkannt und diese vermieden werden. Wood und Stankovic [11] schlagen vor, dass die Knoten am Rand des gestörten Gebietes diese Störung erkennen und Informationen darüber an ihre Nachbarn über Broadcasting weitergeben. Natürlich muss dieser Prozess über Authentizität geschützt sein, denn sonst könnte ein bösartiger Knoten funktionierende Gebiete als gestört deklarieren und die anderen Knoten routen fortan nicht mehr durch diese Gebiete. Aufgrund der erhöhten durchschnittlichen Pfadlängen für die Kommunikation steigt so der Gesamtenergieverbrauch. Um DoS-Attacken über Kollisionen zu begegnen, kann eine Kontrolle auf der Sicherungsschicht statt finden, die nur Pakete von vertraulichen Knoten annimmt. Der DoS-Attacke auf der Transportschicht kann mit den bekannten SYN-Cookies begegnet werden.

C. TinySec

TinySec ist eine Architektur für die Datensicherungsschicht entwickelt von C. Karlof, N. Sastry und D. Wagner, die in [4] vorgestellt wird. Grundlegend für ihr Design sind die Nebenbedingungen, die in Sensornetzwerken herrschen: begrenzte Energie, wenig Speicher und Rechenressourcen und eine schwache Kommunikationseinheit. Eine fertige Implementierung existiert für TinyOS, ein Betriebssystem für eingebettete, drahtlose Sensornetzwerke. Als Sicherheitsziele verfolgt TinySec erstens die Zugangskontrolle, das ist Bestandteil der Authentizität und sagt aus, dass nicht authorisierte Knoten keinen Zugang zum Netz haben sollen, und die Datenintegrität. Zweitens soll TinySec die Informationssicherheit gewährleisten. Drittens sollen Nachspielattacken verhindert werden. Zusätzlich zu diesen sicherheitsrelevanten Zielen, soll auch der Ressourcenverbrauch minimiert werden. Per Design ist TinySec auf der Datensicherungsschicht tätig, daher sind weitere Protokolle für die höheren Schichten notwendig, diese können aber auf TinySec ganz im Sinne der Modularität des OSI-Modells aufbauen .

Es wird in diesem Absatz die Funktionsweise von TinySec genauer erläutert. Dazu sollten die Message Authentication Codes (MACs) und das Konzept des Initialisierungsvektors (IVs) bekannt sein (siehe Glossar). TinySec unterstützt zwei Optionen: zum einen 'authenticated encryption' (TinySec-AE) und 'authentication only' (TinySec-Auth). Im ersten Fall verschlüsselt TinySec den Datenteil der Pakete und authentifiziert diese mit einem MAC. In der zweiten Option fällt die Verschlüsselung weg. Das Verschlüsselungsverfahren von TinySec benutzt einen Blockchiffre mit einen 8 Byte IV im CBC-Modus mit der Modifikation, dass der IV einmal verschlüsselt wird. Das hat denn Vorteil, dass

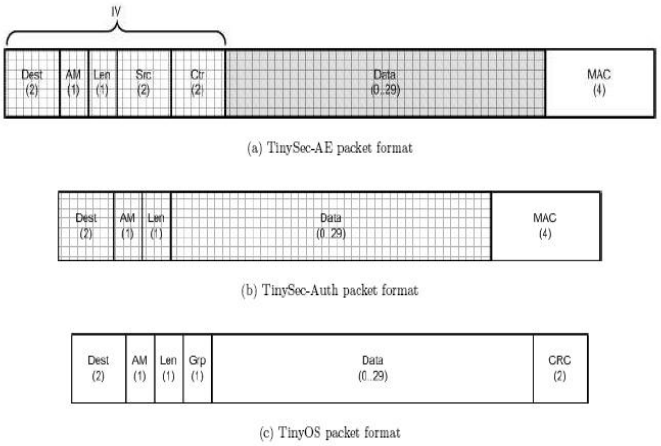

Fig. 1. Header von TinySec aus [4]

es wahrscheinlich keine geringe Abweichung zwischen zwei aufeinander folgenden IVs und der einhergehenden Sicherheitsgefahr gibt.

Der IV setzt sich aus [Zieladresse;Typ;Länge;Quelladdresse;Zähler] zusammen, die als Teilmenge des Headers von Paketen genutzt wird. Die Adressen sowie der Zähler sind 2 Byte lang. Der Typ und die Länge der Nachricht jeweils ein Byte. Der Typ heisst auch 'Active message type' und erfüllt eine ähnliche Funktion wie die Ports bei TCP/UDP. Da in Sensornetzwerken üblicherweise über Broadcasting übertragen wird, sind die diese Felder in den Paketen unverschlüsselt damit Pakete frühzeitig abgelehnt werden können. Als symmetrischer Chiffre wird von den Entwicklern Skipjack ([12]) vorgeschlagen und als Standardverfahren für TinySec verwendet. Als MAC wird die CBC-MAC Konstruktion aus [13] verwendet. Das Paketformat ist ferner an das von TinyOS angelehnt, der wesentliche Unterschied besteht in der Ersetzung des CRC zugunsten des MACs. Ferner da der MAC Authentifizierungsmechanismus auch die Zugangskontrolle regelt, ist das 'Group' Feld aus dem TinyOS Paket-Header nicht mehr zu finden, als Ersatz wird vorgeschlagen für verschiedene Netzwerke verschiedene Schlüssel zu benutzen. Im TinySec-Auth findet man auch nicht die Quelladresse und den Zähler.

Damit ergibt sich für TinySec-Auth 4 Byte Header, 4 für den MAC und 0 bis 29 Byte für die Nutzdaten. TinySec-AE hat dagegen 8 Byte für den Header, der gleichzeitig der IV ist, 4 für den MAC und ebenso 0 bis 29 Byte für die Nutzdaten. Sollte ein Angreifer versuchen die MAC zu fälschen, so braucht er bei zufälligem Raten 2^{31} erwartete Versuche. Da die Übertragungsrate in Sensornetzwerken beschränkt ist, reicht dies im Gegensatz zu konventionellen Netzwerken aus. Als Abwehrmaßnahme ist es auch möglich eine simple Heuristik zu implementieren, die nach einer festen Zahl von falschen MACs Blockierungsmaßnahmen durchführt. Die Entwickler von TinySec schlagen vor, dass bevor eine IV-Kollision auftritt, ein erneuter Schlüsselaustausch statt

finden sollte. Zu bedenken ist, dass man einen für die Chiffrierung und einen für den MAC auszutauschen hat. Das ist jedoch Aufgabe von übergeordneten Protokollen, dazu siehe den Abschnitt zur Schlüsselverteilung.

Abschliessend wird im folgendem auf die Implementierungsaspekte von TinySec eingegangen. TinySec wurde schon für die Berkeley Sensorknoten implementiert, so z.B. Mica, Mica2 und Mica2Dot. Es bedarf für die Referenzimplementierung auf dem TinyOS 728 Bytes Arbeitsspeicher und 7146 Bytes für das Programm. Zusätzlich zum Skipjack-Chiffre wurde noch RC5 implementiert und TinySec konnte mit beiden erfolgreich getestet werden, per Design ist es auch Chiffreunabhängig. Bei den Effizienztests wurde bei TinySec-Auth ein Anstieg der Energiekosten von etwa 3% und bei TinySec-AE von 10% gemessen. In beiden Fällen schreiben die Autoren ca. $2/3$ der gestiegenen Kosten der Paketlänge zu, der Rest fällt für die kryptographischen Berechnungen an. Die benötigte Datenbreite und Latenz steigt beides ebenfalls weniger als 10%.

D. RABBIT und LRSA

Es werden nun der RABBIT-Stromchiffre und die 'Lightweight Rabbit based Security Architecture' kurz LRSA, eine Sicherheitsarchitektur für Sensornetzwerke, die den RABBIT-Stromchiffre benutzt, vorgestellt. Der RABBIT-Stromchiffre basiert zum generieren der Schlüsselbitfolge auf besonders chaotischen, nichtlinearen Abbildungen. Damit wird eine pseudo-zufällige Folge als Schlüssel besonders effizient generiert. Er bedarf einen geheimen 128 bit Schlüssel mit 64 bit IV als Nebeneingabe um daraus die Schlüsselstromfolge zu generieren.

LRSA stellt ähnlich wie TinySec zu symmetrischen Verfahren eine Architektur für den RABBIT-Stromchiffre zur Verfügung. Die Designziele sind Sicherheit, Effizienz und eine einfache Benutzung. Dem grundsätzlichen Dilemma des Zielkonfliktes der Effizienz und der Sicherheit in Sensornetzwerken haben die Designer von LRSA von Anfang an im Auge gehabt: daher benutzen sie wie TinySec dasselbe Konzept den IV aus dem Paket Header zu entnehmen und nur die Nutzdaten zu verschlüsseln. Der Zusatzaufwand in der Kommunikation und der Berechnung soll möglichst minimiert werden und alle Schlüssel, die der Schlüsselungsverteilungsmechanismus zur Verfügung stellt, sollen optimal genutzt werden.

Wir erläutern nun, wie die 64 Bit des IVs aus dem Paket Header ermittelt werden. Das Paket besteht aus (Zieladresse,Typ,Länge,Quelladresse,Zähler,–Daten–,CRC).
Bis auf das 2-Byte CRC Feld, welche den 4-byte MAC ersetzt, ist es exakt gleich zum Paketdesign vom TinySec-AE. Das ist kein Zufall, denn beide Architekturen basieren auf TinyOS. Auch hier wird wiederrum das Group Feld weggelassen, denn dies wird durch den gleichen Schlüssel für die gleiche Gruppe ersetzt. Der IV besteht, wie bei TinySec, aus den ersten 8 Byte des Headers, also allem außer den Nutzdaten und dem CRC. Testläufe wurden mit TOSSIM, einem Simulator für TinyOS, durchgeführt. Die Ergebnisse sind, dass die Ausführungszeit einer Ver- oder Entschlüsselung eines Blocks der Größe 128 Bit ungefähr 20μs ohne und ungefähr 39μs mit benötigt. Sie verdoppelt sich somit etwa. Der RAM Verbrauch verdoppelt sich ebenfalls und der Festspeicherverbrauch um etwa 40%. Der Energieverbrauch hingegen steigt jedoch fast garnicht. Die meisten Energiekosten eines Sensornetzwerkes fallen für die Übertragung an. In [1] findet man den Vergleich, dass bei der Übertragung von 1 Kbyte auf 100m die gleiche Kosten anfallen, wie für die Ausführung von 3 millionen Instruktionen auf einem Prozessor, der 100 millionen Instruktionen pro Sekunde ausführen kann bei 1 [(MIPS)/W][2].

IV. Diskussion

In diesem Abschnitt wird TinySec und LRSA miteinander verglichen. Die Ähnlichkeiten bezüglich der Paketkonstruktion wurden schon im vorherigen Teil angesprochen: aufgrund der Ausrichtung auf TinyOS haben beide Architekturen das Group Feld gestrichen und verweisen auf die Schlüsselverteilung um das Groups im Netzwerk zu unterstützen. Die wesentlichen Unterschiede zwischen TinySec und dem LRSA sind zum einen die Wahl des Chiffres zum anderen die Benutzung eines MACs. Es wird zuerst auf die Wahl des Chiffres und den Authentifizierungsmechanismen eingegangen, sowie darauf folgend weitere funktionale Aspekte betrachtet.

Symmetrische Blockchiffren und Stromchiffren gelten als die effizientesten Verschlüsselungsklassen. Die Designer von TinySec haben sich für die symmetrischen Blockchiffreverfahren entschieden, wobei sie die konkrete Wahl offen lassen und nur eine Empfehlung für Skipjack aussprechen, und die von LRSA für den RABBIT-Stromchiffre, der erst 2003 in [5] vorgeschlagen wurde. In dieser Arbeit wurde auch eine Kryptoanalyse durchgeführt und Divide-and-Conquer-, Guess-and-Determine- und Attacken auf eine Teilfunktion des RABBIT ausgeschlossen. Ferner wurden die typischen Angriffsmöglichkeiten für Stromchiffren dem Korrelationsangriff und algebraische Angriffe analysiert und als sicher eingestuft. Einen Angriff auf den RABBIT-Chiffre findet man in [14]. Sie benötigt etwa 2^{158} Versuche, also mehr als klassische Brute-Force-Attacken auf den 128-bit-Schlüsselraum, welche 2^{128} Versuche benötigen würde. In ihrer Performance Analyse kamen Boesgaard et al. auf die selben Ergebnisse, die Tahir et. al in ihren Simulationsläufen von LRSA mit dem Simulator TOSSIP kamen. Daher ist davon auszugehen, dass der RABBIT-Stromchiffre zur Zeit sicher.

Karlof, Sastry und Wagner haben in ihrer Arbeit [4], wo sie TinySec vorstellen, gegen die Benutzung von Stromchiffren ausgesprochen. Sie begründen die Ablehnung im wesentlichen mit der schwachen Resistenz gegenüber Kollisionen bei IVs mit gleichem Schlüssel, welche es unter Umständen erlaubt, den Klartext auszurechnen. Da aber beide Architekturen über 2 Byte-Zähler als Teil des IVs verfügen und man ja durchaus

[2]Millionen Instruktionen pro Sekunde pro Watt

sicher einen Schlüsselaustausch durchführen könnte, ist dies bei der niedrigen Datenrate von Sensornetzwerken äußerst unwahrscheinlich. Sie räumen sogar ein, dass die schnellsten Stromchiffren schneller als die schnellsten Blockchiffren sind. Die Experimente der Performance von TinySec und LRSA sind kaum miteinander vergleichbar, da sie verschiedene Merkmale getestet haben. Aus den vorliegenden Daten lässt sich jedoch schliessen, dass LRSA effizienter ist.

Da jedoch nur TinySec über ein Authentifizierungsmechanismus verfügt und mit diesem gleich die Fehlererkennung mitliefert. Dabei aber die Verschlüsselung optional hält, bietet es in einigen Anwendungsfällen die bessere Alternative.

V. Folgerungen und Ausblick

Wir haben in dieser Arbeit die Sicherheitsaspekte von drahtlosen Sensornetzwerken durchgesprochen, Problemfelder und deren ersten konzeptuellen Lösungen angesprochen, sowie zwei Architekturen und deren Eigenschaften vorgestellt. Eine Gesamtlösung für die Sicherheit in Sensornetzwerken findet sich nicht, jedoch einige für spezielle Probleme angepasste Lösungen. In Anbetracht der geringen Ressourcen ist es auch nicht anders möglich als, je nach Typ des Sensornetzwerkes auf dazu passende Lösungen zurückzugreifen. Benötigt das Sensornetz Authentifizierungsverfahren, so ist die Benutzung von TinySec anzuraten, da eine fertige Implementierung existiert und Sicherheit geboten wird. Benötigt es diese nicht, so sollte man wegen des geringeren Ressourcenverbrauchs zur LRSA greifen.

Die Entwickler von TinySec haben bei der ersten Implementierung davon abgesehen, die klassischen symmetrische Chiffres Triple-DES und AES zu verwenden, da diese zu teuer seien. Jedoch geben sie schon in ihrer Arbeit zu, dass AES wohl auch günstig implementiert werden können, dies würde TinySec schon wesentlich sicherer machen. Auf TinySec aufbauend sind TinyPK [15], welche das RSA-Kryptographiesystem zum Schlüsselaustausch benutzt, und TinyECC [16], welches Elliptische Kurven Kryptographie verwendet. Wegen der großen Ähnlichkeit von TinySec und LRSA, sowie der Modularisierung der Architekturen, können diese wahrscheinlich auch ohne große Probleme mit LRSA benutzt werden, sofern sie den MAC nicht benötigen. Aufgrund der verschiedenen Performancetests in den Arbeiten zu TinySec und LRSA, wäre es wünschenswert beide ausführlich gegeneinander zu testen. Eine Möglichkeit zur Authentifizierung wäre für LRSA wünschenswert.

Konzeptuell ist die Computer-/Netzwerksicherheit schon sehr weit, jedoch treten mit Sensornetzwerken die Probleme hoher Knotenanzahl, begrenzten Ressourcen und wenig Speicher/Rechenkraft, wie auch den möglichen physikalischen Zugriff, auf. Erste speziell zu den Anforderungen von Sensornetzwerkenden passende Lösungen z.B. für das Schlüsselmanagement, Denial-of-Service Attacken oder der Sicherung auf der Datensicherungsschicht befinden sich in Entwicklung und erste Ergebnisse wurden in dieser Arbeit vorgestellt. Jedoch sind Sensornetze noch weit davon entfernt, die Sicherheit und Verlässlichkeit konventioneller Netze bieten zu können. Man muss sich in der Entwicklung auf jeder Schicht im Sinne des OSI-Modells Gedanken um die Sicherheitsaspekte machen: von induzierten Störungen auf der physischen Ebene zu Maskeraden-Attacken auf Managementsysteme auf der Anwendungsschicht.

Danksagungen

Mein Dank geht an Heiko Will für die Betreuung dieser Seminararbeit und an die anonymen Reviewer die gute Hinweise zur Verbesserung dieser Arbeit gegeben haben.

References

[1] I. F. Akyildiz, Su Weilian, Y. Sankarasubramaniam, and E. E. Cayirci, "A survey on sensor networks", *IEEE Communications Magazine*, vol. 40, no. 8, pp. 102–114, 2002.

[2] Yang Xiao, *Security in Distributed, Grid, Mobile, and Persasive Computing*, CRC Press, 2007.

[3] Sasikanth Avancha, Jeffrey Undercoffer, Anupam Joshi, and John Pinkston, "Security for wireless sensor networks", pp. 253–275, 2004.

[4] Chris Karlof, Naveen Sastry, and David Wagner, "TinySec: A link layer security architecture for wireless sensor networks", in *Proceedings of the Second ACM Conference on Embedded Networked Sensor Systems (SenSys 2004)*, Nov. 2004.

[5] Boesgaard, Vesterager, Pedersen, Christiansen, and Scavenius, "Rabbit: A new high-performance stream cipher", in *IWFSE: International Workshop on Fast Software Encryption, LNCS*, 2003.

[6] Ruhma Tahir, Muhammad Younas Javed, Muhammad Tahiar, and Firdous Imam, "LRSA: Lightweight Rabbit based Security Architecture for wireless sensor networks", *Second International Symposioum on Intelligent Information Technology Application*, pp. 679–683, 2008.

[7] Yee Wei Law, Jeroen Doumen, and Pieter Hartel, "Survey and benchmark of block ciphers for wireless sensor networks", *ACM Transactions on Sensor Networks*, vol. 2, no. 1, pp. 65–93, Feb. 2006.

[8] Sencun Zhu, Sanjeev Setia, and Sushil Jajodia, "LEAP+: Efficient security mechanisms for large-scale distributed sensor networks", *TOSN*, vol. 2, no. 4, pp. 500–528, 2006.

[9] Haowen Chan and Adrian Perrig, "PIKE: peer intermediaries for key establishment in sensor networks", in *INFOCOM*. 2005, pp. 524–535, IEEE.

[10] Qiang Huang, Johnas Cukier, Hisashi Kobayashi, Bede Liu, and Jinyun Zhang, "Fast authenticated key establishment protocols for self-organizing sensor networks", pp. 141–150.

[11] Anthony D. Wood and John A. Stankovic, "Denial of service in sensor networks", *IEEE Computer*, vol. 35, no. 10, pp. 54–62, 2002.

[12] National Institute of Standards and Technology (NIST), "SKIPJACK and KEA algorithm specifications", 1998, `http://csrc.nist.gov/encryption/skipjack-1.pdf`, `skipjack-2.pdf`.

[13] Bellare, Kilian, and Rogaway, "The security of the cipher block chaining message authentication code", *JCSS: Journal of Computer and System Sciences*, vol. 61, 2000.

[14] Yi Lu, Huaxiong Wang, and San Ling, "Cryptanalysis of rabbit", in *ISC*, Tzong-Chen Wu, Chin-Laung Lei, Vincent Rijmen, and Der-Tsai Lee, Eds. 2008, vol. 5222 of *Lecture Notes in Computer Science*, pp. 204–214, Springer.

[15] Ronald J. Watro, Derrick Kong, Sue fen Cuti, Charles Gardiner, Charles Lynn, and Peter Kruus, "TinyPK: securing sensor networks with public key technology", in *SASN*, Sanjeev Setia and Vipin Swarup, Eds. 2004, pp. 59–64, ACM.

[16] An Liu and Peng Ning, "Tinyecc: A configurable library for elliptic curve cryptography in wireless sensor networks", in *IPSN '08: Proceedings of the 7th international conference on Information processing in sensor networks*, Washington, DC, USA, 2008, pp. 245–256, IEEE Computer Society.

[17] Johannes Buchmann, *Einführung in die Kryptographie*, Springer-Verlag, Berlin-Heidelberg-New York-Barcelona-Budapest-Hong Kong-London-Mailand-Paris-Santa Clara-Singapur-Tokyo, 1999.

[18] Bellare, Desai, Jokipii, and Rogaway, "A concrete security treatment of symmetric encryption", in *FOCS: IEEE Symposium on Foundations of Computer Science (FOCS)*, 1997.

GLOSSAR

Der Anhang ist vor allem dazu gedacht dem Leser dieser Seminararbeit allgemeiner hinsichtlich der Kryptographie weiterzubilden und einige wichtige Konzepte kurz vorzustellen. Die Hauptquelle ist [17].

Der Diffie-Hellman Schlüsselaustausch: Sei $(G, \cdot)$ eine endliche von $g \in G$ erzeugte zyklische Gruppe (die dann auch abelsch ist, d.h. die Verknüpfung $\cdot$ ist kommutativ) mit Ordnung n. Die beiden Kommunikationsseiten 1 und 2 wählen zufällig $a_1, a_2 \in \{2, \ldots, n-1\}$ und senden sich $A_1 := g^{a_1}$ bzw. $A_2 := g^{a_2}$. Sie berechnen dann beide jeweils $K = A_1^{a_2} = g^{a_1 \cdot a_2} = g^{a_2 \cdot a_1} = A_2^{a_1}$, welches den Schlüssel darstellt. Man kann z.B. für eine Primzahl p die Einheitengruppe (multiplikative Gruppe) von $\mathbb{Z}_p$ wählen.

Initialisierungsvektoren: Um semantische Sicherheit (d.h. der Angreifer kann vom Chiffretext mit keiner Wahrscheinlichkeit besser als 50% Wahr/Falsch Fragen über den zugehörigen Klartext zu erreichen) zu erreichen, wird bei jeder Ausführung eines Verschlüsselungsalgorithmus zusätzlich zum Schlüssel noch ein weiterer Parameter, der nicht notwendigerweise geheim sein muss, mit verwoben. Diesen nennt man Initialisierungsvektor kurz IV. Es sollen ähnliche Muster im Klartext im unterschiedlichen Kontext (aus dem der IV stammt) im Chiffretext so verwaschener aussehen. Ein Beispiel in dem ein IV genutzt wird, ist der Cipherblock-Chaining-Mode (CBC-Mode), den wir nun für ein symmetrisches Kryptosystem beschreiben. Er ist beweisbar [18] sicher, falls die IV nicht mehrfach genutzt werden, d.h. kollidieren. Sei dazu $\Sigma = \{0, 1\}$ das Alphabet. Wir betrachten einen Blockchiffre der Länge $n \in \mathbb{N}$ mit Schlüsselraum $\mathcal{K}$ und Ver-/Entschlüsselungsfunktionen $(E_k)_{k \in \mathcal{K}}$ bzw. $(D_k)_{k \in \mathcal{K}}$. Sei IV $\in \Sigma^n$ der Initialisierungsvektor, $k \in \mathcal{K}$ der symmetrische Schlüssel und $(m_1, \ldots, m_r) \in (\Sigma^n)^r$ die zu verschlüsselnden Blöcke. Setze $c_0 =$ IV. und $c_j = E_k(c_{j-1} \oplus m_j)$ $(1 \leq j \leq r)$ mit $\oplus$ bitweises XOR, dann ist $(c_1, \ldots, c_r)$ der Chiffretext. Entschlüsselt wird nun mit $c_0 =$ IV und $m'_j = c_{j-1} \oplus D_k(c_j) = c_{j-1} \oplus D_k(E_k(c_{j-1} \oplus m_j)) = c_{j-1} \oplus c_{j-1} \oplus m_j = m_j$ für $j = 1..r$. Bei festem Schlüssel sollte der gleiche IV nicht zweimal zum Verschlüsseln ausgewählt werden. Unterstellen wir eine Gleichverteilung auf der Menge der möglichen IVs und eine Bitlänge von n, so wird es wegen des Schubfachprinzips spätestens nach $2^n + 1$ Wahlen des IVs eine Kollision geben. Nach dem Geburtstagsparadox jedoch, können wir schon nach $2^{n/2}$ Wahlen eine Kollision probabalistisch erwarten.

Message Authentification Codes (MACs)

Message Authentification Codes (MACs) stellen einen anhand eines geheimen Schlüssels aus der Nachricht erzeugten Prüfteil dar, der mitgeschickt wird. Damit kann der Empfänger, der den Schlüssel ebenfalls kennt, analog diesen Prüfteil erzeugen und mit den empfangenen vergleichen um die Authenzität und Integrität der Nachricht zu sichern. Wir beschreiben nun formaler MACs, die auf Hashfunktionen, das sind Funktionen die für ein Alphabet Σ Σ^* auf Σ^n für ein $n \in \mathbb{N}$ abbilden, basieren.. Sei $\mathcal{K}$ eine Schlüsselmenge, dann heißt $(h_k)_{k \in \mathcal{K}}$ mit h_k Hashfunktion für festes $k \in \mathcal{K}$ parametrisierte Hasfunktionen, die man auch MAC nennt. Als Beispiel kann man für eine Hasfunktion $g : \Sigma^* \to \Sigma^n$ und $k \in \mathcal{K} = \Sigma^n$ einen MAC mit $h_k : x \mapsto g(x) \oplus k$ konstruieren. Will man nun eine Nachricht $m \in \Sigma^*$ authentifizieren, so wird ein Schlüssel $k \in \mathcal{K}$ zwischen den Kommunikationsparteien ausgetauscht. Der Sender schickt dann $(m, h_k(m))$ und der Empfänger erhält (m', y). Er prüft ob $y' = h_k(m') = y$ ist und falls ja, so ist die Nachricht authentizifizert. Es soll für einen Angreifer unmöglich sein, ohne der Kenntniss von k ein Paar $(m, h_k(m))$ zu berechnen.

Stromchiffren: Nach [17] verfolgt ein Stromchiffre das Konzept z.B. aus einer Saat (Seed) einen Schlüssel fortlaufend zu erzeugen, der mit den zu verschlüsselnden Daten bitweise mit XOR verknüpft wird[3] und dementsprechend wieder entschlüsselt wird. Die Idee hat die Grundlage, dass das sogenannte One-Time-Pad von Vernam perfekte Sicherheit im Sinne der Shannonschen Definition[4] bietet. Das One-Time-Pad besitzt einen Schlüssel, der so lang ist wie der Klartext, und verknüpft dies bitweise mit XOR zum Ent- und Verschlüsseln. Die Vorteile eines Stromchiffres sind nun, dass man keinen Schlüssel angeben muss, der so lang wie der Klartext ist, aber trotzdem sicher chiffriert wird, sofern die Erzeugung des Schlüssels dem Angreifer unzugänglich bleibt. Da der Schlüssel fortwährend erzeugt wird, kann der Klartext auch nach und nach verschlüsselt und versetzt werden. Die meisten Stromchiffren brauchen zur Erzeugung der nächsten Bits nur eine konstante Anzahl der vorherigen Bits und je nach Modi auch einen Initialisierungsvektor, d.h. andere Daten (die dem Sender als auch dem Empfänger bekannt sein müssen), die in die Berechnung einfliessen. Somit ist die Speicherbelastung vergleichsweise gering und je nach Verfahren kann die Effizienz der Schlüsselfolgenerzeugung variieren.

[3] Oder auch eine Addition im Galoiskörper $\mathbb{F}_2$.

[4] Perfekte Sicherheit im Shannonschen Sinne bedeutet, dass man keine wahrscheinlichkeitstheoritischen Schlüsse aus dem Chiffretext über den Klartext ziehen kann. ($P(p|c) = P(p)$, p Klar- und c Schlüsseltext). Dieser Definitionsansatz für Sicherheit aus der Wahrscheinlichkeitstheorie überrascht nicht: ähnlich wurde von Shannon die Informationstheorie eingeführt und entwickelt.