Universität Ulm, Fakultät für Informatik
Abteilung Medieninformatik
Leiter: Prof. Dr. Michael Weber

Sicherheit in Mobilen Ad hoc Netzwerken

Dissertation

zur Erlangung des Doktorgrades
Dr.rer.nat.
der Fakultät für Informatik
der Universität Ulm

vorgelegt von
Frank Kargl
aus Werneck
2003

Amtierender Dekan:	Prof. Dr. Friedrich von Henke
Gutachter:	Prof. Dr. Michael Weber
Gutachter:	Prof. Dr. Jörg Kaiser
Tag der Promotion:	21. Oktober 2003

„Mache die Dinge so einfach wie möglich - aber nicht einfacher.“ *Albert Einstein*

Inhaltsverzeichnis

1. Einleitung

In den letzten Jahren erfahren drahtlose Funknetzwerke eine große Aufmerksamkeit. Auf der einen Seite stehen Funknetzwerke wie das *Global System for Mobile Communication* (*GSM*, ursprünglich *Groupe Spécial Mobile*), welche primär der mobilen Telefonie dienen, aber auch zunehmend für den Datentransport genutzt werden. Auf der anderen Seite haben sich verschiedene Standards für die mobile Vernetzung von elektronischen Geräten etabliert. Hierzu gehören beispielsweise *Wireless LAN (IEEE 802.11)*, *Bluetooth* oder *HomeRF*.

Alle diese Systeme haben eines gemeinsam: In der Regel überbrücken Sie lediglich die Distanz zwischen einer Basisstation und dem Endgerät des Benutzers über eine Funkschnittstelle. Hinter der Basisstation befindet sich dann eine mehr oder weniger komplexe Infrastruktur, welche der Verwaltung des Gesamtnetzwerks dient und die in der Regel über klassische, leitungsgebundene Netzwerke kommuniziert.

Bei GSM besteht diese Infrastruktur z.B. aus Home Location Register (HLR), diversen Service Nodes und Gateways zu anderen Funk- und Fest-Netzen. Bei Wireless LANs werden die Access Points typischerweise über ein Ethernet oder ein anderes, IP-basiertes Netz miteinander verbunden.

Insofern ist der Begriff des „Mobilfunks“ oder des „drahtlosen LANs“ eigentlich irreführend, da der größte Teil der involvierten Geräte mittels klassischer Leitungen verbunden sind und nur an der Schnittstelle zum Endnutzer wirklich eine Funkschnittstelle zum Einsatz kommt.

In jüngster Zeit entwickeln Forscher in aller Welt jedoch eine Vision von Netzwerken, welche ganz anders aufgebaut sind. Sogenannte *Mobile Ad hoc Netzwerke (MANETs)* sind vollkommen dezentral organisiert. Es gibt keine ausgezeichneten Knoten, deren Ausfall das Netzwerk zum Stillstand bringen könnte. Damit ermöglichen MANETs den Einsatz an Orten oder in Situationen, in denen der vorherige Aufbau einer Infrastruktur zur Vernetzung von Computern oder anderen elektronischen Geräten nicht möglich oder wünschenswert ist. Typische Beispiele sind Besprechungen, in denen die Notebooks der Teilnehmer vernetzt werden sollen, Einsätze von Rettungskräften in Katastrophengebieten, militärische Einheiten auf dem Schlachtfeld oder fahrende Pkws auf einer Autobahn. Zwangsläufig ergeben sich hieraus auch neue Formen von Anwendungen, welche diese spontan gebildeten Netze nutzen.

Bisher konzentrierten sich die Anstrengungen primär auf die Entwicklung geeigneter Routingprotokolle, welche die Verkehrslenkung in MANETs organisieren. Zu den Neuerungen gehören Protokolle, die erst bei Bedarf (on-demand) tätig werden, oder solche, welche die geographische Position oder die Signalstärke mit in Betracht ziehen. Kapitel 5 stellt MANET und das Routing in MANETs ausführlich vor.

Doch nicht nur das Routing in Ad hoc Netzen unterscheidet sich radikal von den traditionellen Mechanismen. Auch in anderen Bereichen wie der Service-Discovery oder der

Adressverteilung werden gänzlich neue Ansätze für MANETs entwickelt. Ein Aspekt, der bisher auch nur teilweise untersucht wurde, ist die Absicherung solcher Netze. Dabei treten eine Reihe von neuen Fragestellungen auf:

1. Wie werden Knoten oder deren Benutzer identifiziert?
2. Wie wird das Vertrauen in solchen Netzen organisiert, wenn sich die Teilnehmer zu Anfang nicht kennen und auch kein vertrauenswürdiger Dritter online verfügbar ist?
3. Wie geschieht die Authentisierung von Knoten oder Benutzern?
4. Können Knoten oder deren Benutzer genau lokalisiert werden? Lassen sich Bewegungsprofile erstellen? Wie kann man dies verhindern?
5. Wenn alle Knoten gleichzeitig auch Router sind, tragen auch alle Knoten zum Topologieaufbau und zur Routenfindung bei. Wie kann man verhindern, dass böswillige Knoten diesen Prozess stören und somit die Funktionsfähigkeit des Netzwerks beeinträchtigen?
6. Wie geht man mit egoistischen Knoten um, welche zwar die Leistung des MANETs nutzen, selbst aber nicht bereit sind, zum Aufbau des Netzes eigene Ressourcen beizutragen?
7. Können die Sicherheitsmechanismen den dynamischen Strukturen im MANET Rechnung tragen und sich daran anpassen?
8. Lassen sich die Sicherheitsmechanismen auch auf stark ressourcenbeschränkten Geräten wie PDAs oder Mobiltelefonen betreiben?

Wegen der besonderen Struktur von MANETs lassen sich die Lösungen aus dem Bereich herkömmlicher Netze nicht einfach auf MANETs übertragen. So wird eine klassische PKI Lösung in MANETs nicht ohne weiteres funktionieren, da zentrale CA Server meist nicht online erreichbar sind. Manche Fragestellungen sind auch gänzlich neu und treten nur in MANETs auf. Ein Beispiel dafür ist die Bekämpfung von egoistischen Knoten.

Die bisherigen Forschungsarbeiten in diesem Bereich, welche in dieser Dissertation ausführlich vorgestellt werden, konzentrieren sich zumeist auf ein sehr eng begrenztes Gebiet und liefern hier eine isolierte Lösung. Ein typisches Beispiel sind sichere Routingprotokolle für MANETs, wie sie in Kapitel 10 beschrieben werden. Viele setzen eine sichere Verteilung von geheimen oder öffentlichen kryptographischen Schlüsseln voraus. Wie diese Schlüssel aber ohne bestehende Routen effizient verteilt werden sollen, bleibt offen.

Die Sicherheitsfragestellungen in MANETs sind an vielen Punkten miteinander verwoben. Eine singuläre Betrachtung einzelner Aspekte erscheint daher nicht sinnvoll. Diese Arbeit geht einen anderen Weg und versucht, ein komplettes Sicherheitsrahmenwerk für Ad hoc Netze zu entwerfen, welches ohne initiale Annahmen auskommt und sämtliche der oben angesprochenen Fragestellungen diskutiert. Nach meinem Kenntnisstand ist dies der erste derartige Versuch.

Ein solcher Ansatz bedingt zwangsläufig, dass diese Arbeit sehr breit angelegt sein muss und viele Teilbereiche der Netzwerksicherheit umfasst. Dies wird auch im Umfang der Arbeit deutlich. Eine systematische Herangehensweise an Sicherheitsfragen

bedingt immer zunächst eine umfassende und strukturierte Sicherheitsanalyse, welche in Kapitel 6 erfolgt. Die bestehenden Arbeiten in dieser Richtung haben die Frage nach möglichen Angriffen meist nur lückenhaft und wenig strukturiert beleuchtet. Indem eine bisher ungekannte Vielzahl von Angriffsformen strukturiert in sogenannten Angriffsbäumen erfasst werden, leistet diese Arbeit hier einen wichtigen Beitrag.

Aufbauend auf dieser Analyse werden verschiedene Teilbereiche identifiziert, welche durch Sicherheitsmechanismen abgesichert werden müssen. Das Resultat ist die in Kapitel 7 vorgestellte *Sicherheitsarchitektur für Mobile Ad hoc Netze*, kurz *SAM*. SAM besteht aus insgesamt vier Komponenten, welche in den Kapiteln 8, 9, 10 und 11 betrachtet werden. Dabei werden in jedem Teil immer erst bestehende Arbeiten auf diesem Gebiet vorgestellt und deren Stärken und Schwächen analysiert. Darauf aufbauend wird ein eigener und in SAM integrierter Lösungsansatz entwickelt, welcher die Nachteile der anderen Lösungen zu vermeiden sucht. Indem diese Teillösung im Kontext der gesamten Sicherheitsarchitektur betrachtet wird, vermeidet SAM Probleme, die aus einem zu engen Blickwinkel resultieren.

Kapitel 8 widmet sich der Fragestellung, wie Knoten im Ad hoc Netz zu identifizieren sind. Breiten Raum nimmt dabei die Frage ein, was eigentlich eine Identität in einem Ad hoc Netz auszeichnet und wie diese beschaffen sein muss, um als Ausgangsbasis für eine Sicherheitsarchitektur dienen zu können. Dieser Punkt wurde in früheren Arbeiten stets vernachlässigt und wird hier erstmals ausführlich untersucht. Als Resultat dieser Überlegungen werden die *MANET-IDs* vorgestellt, ein System zur Identifizierung von Geräten, welches auch ohne ständigen Kontakt zu einer zentralen Infrastruktur genutzt werden kann und welches über effiziente Mechanismen zum Rückruf und zur Sperrung von Identitäten verfügt. Diese werden vor allem vom später vorgestellten Mobile Intrusion Detection System genutzt.

Daran schließt sich Kapitel 9 an, welches die Frage betrachtet, wie die Erstellung von Bewegungsprofilen in Ad hoc Netzen verhindert und somit die Privatsphäre der Nutzer geschützt werden kann. Dieses Problem wurde in früheren Arbeit noch überhaupt nicht betrachtet. Als Lösung dieser Problemstellung werden die MANET-IDs um die Unterstützung von *Pseudonymen* erweitert. Damit kann ein Benutzer oder Gerät im MANET seine Identität verschleiern, was den Wert von Bewegungsprofilen deutlich einschränkt.

Ein Kernaspekt von SAM ist die Absicherung des Routingprozesses vor unberechtigten Modifikationen. Das *Secure Dynamic Source Routing* Protokoll (*SDSR*) in Kapitel 10 schützt die übertragenen Daten in vielfältiger Weise. Verglichen mit anderen Protokollen wie SAODV, ARAN oder Ariadne zeigt sich auch hier der Vorteil des breiten Ansatzes dieser Arbeit. Während diese Protokolle sich alleine auf den Schutz der Routingdaten konzentrieren, sind in SDSR ebenso eine komplette Authentifizierung aller an einer Route beteiligten Knoten, die Verteilung öffentlicher Schlüssel und die Vereinbarung geheimer Sitzungsschlüssel integriert. Letztere werden unter anderem von der MobIDS Komponente benötigt.

MobIDS, das *Mobile Intrusion Detection System*, adressiert in Kapitel 11 eine letzte Problemstellung, welche die Routingprotokolle meist außen vor lassen. Wie schütze ich das Netz vor egoistischen Schmarotzern, welche zwar die angebotene Leistung der Weiterleitung von Paketen durch andere Knoten in Anspruch nehmen, selbst aber nicht bereit sind, im Gegenzug eigene Ressourcen aufzuwenden, um Verkehr anderer Kno-

ten zu transportieren? MobIDS verfügt über Sensoren, welche ein solches Verhalten erkennen. Gegebenenfalls können die Teilnehmer des MANET einen egoistischen Knoten durch Ausschluss aus dem Netz bestrafen. Im Vergleich zu bisherigen Systemen entwickelt MobIDS vor allem die Sensoren, welche ein Fehlverhalten erkennen sollen, weiter und stellt drei neue und leistungsfähigere Sensoren vor. Neu ist ebenfalls die in Kombination mit den MANET-IDs gegebene Möglichkeit, einen Knoten nicht nur aus dem aktuellen MANET auszuschließen, sondern zukünftig weltweit die Teilnahme an MANETs zu verhindern.

Die bisherigen Kapitel beschränken sich aus Gründen der Übersichtlichkeit auf eine funktionale Darstellung der Komponenten von SAM und einen Vergleich mit verwandten Arbeiten. Kapitel 12 analysiert im Anschluss die SAM-Komponenten im Hinblick auf ihre Funktionalität, auf verbleibende Angriffsmöglichkeiten und auf Aufwand und Effizienz. Als Werkzeuge kommen hier formale Methoden und Simulationen zum Einsatz. Es wird gezeigt, dass SAM den bisherigen Ansätzen oft überlegen, mindestens aber ebenbürtig und der entstehende Aufwand meist gering ist, zumindest aber noch im vertretbaren Rahmen bleibt.

Den Abschluss bildet Kapitel 13, in welchem zunächst nochmals die wesentlichen Aspekte dieser Arbeit zusammengefasst werden. Bedingt durch den breiten Ansatz konnten nicht alle Aspekte und Fragestellungen in beliebiger Tiefe verfolgt werden. Im Ausblick werden verschiedene dieser Punkte nochmals aufgegriffen und kurz andiskutiert. Einiges davon mag als Anregung für zukünftige Forschungstätigkeiten dienen.

Zunächst folgen aber einige einführende Kapitel zu Sicherheit, Kryptographie und mobilen Datennetzen, welche den Grundstock für das Verständnis der nachfolgenden Themen legen sollen.

2. Sicherheit

Es herrscht weitestgehend Einigkeit, dass Sicherheit eine für jegliches technisches System anzustrebende Eigenschaft ist. Dabei ist der Begriff zunächst sehr abstrakt und wenig fassbar. Im angloamerikanischen Sprachraum wird bereits grundsätzlich unterschieden zwischen *Security* und *Safety*. *Safety* beschreibt dabei die Tatsache, dass das System in sich sicher ist, d.h. dass es beim normalen Betrieb des Systems nicht zu Fehlern kommt. Im Gegensatz dazu beschreibt *Security* eine Sicherheit gegen gezielte Angriffe oder Störversuche. Wir wollen uns im Folgenden auf diese Form der Sicherheit konzentrieren, die Sicherheit in Form der *Safety* setzen wir dagegen voraus. Allerdings wird sich später zeigen, dass das Sicherheitssystem *SAM* (*Security Architecture for Mobile Ad hoc Networks*) das Ad hoc Netzwerk in gewissen Grenzen auch gegen fehlerhaft arbeitende Knoten schützen kann.

In den folgenden Abschnitten soll zunächst aber der abstrakte Begriff der Sicherheit (Security) konkreter analysiert und definiert werden. Dabei wird der Einfachheit halber von einem Kommunikationssystem (z.B. also einem MANET) ausgegangen, bei dem ein Sender eine Nachricht an einen oder mehrere Empfänger übertragen will. Die dargestellten Prinzipien lassen sich jedoch leicht auf beliebige IT Systeme (z.B. ein Betriebssystem) verallgemeinern.

Vor der Umsetzung jeglicher Sicherheitsmaßnahmen sind zunächst die Eigenschaften zu definieren, welche man von seinem Kommunikationssystem erwartet. Den nächsten Schritt stellt eine umfassende Analyse von möglichen Bedrohungen dar, welche diese Eigenschaften unter Umständen gefährden. Darauf aufbauend kann man dann das Kommunikationssystem geeignet planen oder im Nachhinein modifizieren, um möglichst viele dieser Bedrohungen zu eliminieren. Hierzu verwendet man verschiedene Sicherheitsmechanismen, wie sie am Ende dieses Kapitels vorgestellt werden.

2.1. Eigenschaften

Elektronische Kommunikationssysteme transportieren heute eine unvorstellbare Menge an Daten. Einige davon sind von hohem Wert und der Sender einer Nachricht geht oft implizit davon aus, dass diese sicher und unverändert den Empfänger (und nur diesen) erreicht.

Es lassen sich sechs wesentliche *Eigenschaften einer Sicherheitsarchitektur* feststellen. Diese sind hier jeweils zusammen mit dem gängigen englischsprachigen Begriff genannt. Die nachfolgende Aufstellung bezieht sich teilweise auf [Sch02], wurde aber überarbeitet und modifiziert.

1. Vertraulichkeit (Confidentiality)
2. Authentizität (Authenticity)

3. Integrität (Integrity)
4. Verfügbarkeit (Availability)
5. Verbindlichkeit (Accountability/Non-Repudiation)
6. Zugriffskontrolle (Access Control)

2.1.1. Vertraulichkeit

Der Sender einer Nachricht will in der Regel sicherstellen, dass nur der rechtmäßige Empfänger auf den Inhalt der Nachricht zugreifen kann. Sendet beispielsweise eine Entwicklungsabteilung Pläne eines neuen Produkts per E-Mail an die Fertigungsabteilung, so sollten diese möglichst nicht in die Hände eines Konkurrenzunternehmens fallen.

Während früher Briefumschläge, Siegel und vertrauenswürdige Boten zum Einsatz kamen, wird Vertraulichkeit in der Welt der elektronischen Kommunikation in der Regel mit Hilfe von Verschlüsselung realisiert.

Fasst man den Begriff der *Vertraulichkeit* noch etwas weiter, so ist bereits die Tatsache, dass eine Nachricht zwischen einem Sender und einem Empfänger ausgetauscht wird, eine Information, welche geheim zu halten ist. So könnte ein Angreifer beispielsweise alleine aus der Tatsache, dass umfangreiche Daten von der Entwicklungs- in die Fertigungsabteilung geschickt werden, schließen, dass die Einführung eines neuen Produkts unmittelbar bevor steht. Um diese Form der Vertraulichkeit zu erreichen, müssen die Identitäten von Sender und/oder Empfänger geschützt werden, was man als *Anonymität* oder *Pseudonymität* bezeichnet.

2.1.2. Authentizität

Der Begriff der *Authentizität* bezieht sich auf die Möglichkeit, die Identität eines Teilnehmers einer Kommunikation zweifelsfrei feststellen zu können. Insbesondere soll also ein Absender einer Nachricht zweifelsfrei zu erkennen sein.

Bei unmittelbarer Kommunikation ist die Identität des Kommunikationspartners in der Regel bereits durch den Augenschein gesichert, problematisch ist lediglich, die behauptete Identität beim ersten Treffen zu verifizieren. Dies geschieht typischerweise durch Vorlage eines Ausweises, welcher von einem vertrauenswürdigen Dritten (in der Regel vom Staat) ausgestellt wurde und hinreichend fälschungssicher sein muss.

Bei mittelbarer Kommunikation (z.B. über ein elektronisches Kommunikationssystem) ist diese leichte Überprüfbarkeit des Absenders einer Nachricht normalerweise nicht gegeben. Hierzu tragen zwei Komponenten bei: erstens wird als Absender normalerweise ein mehr oder weniger abstrakter Name verwendet (z.B. Emailadresse `nobody@gmx.net`), die Zuordnung zu einer bestimmten Person ist normalerweise nicht automatisch gegeben. Zweitens kann der Absender einer Nachricht normalerweise gefälscht werden. So kann jeder eine Email mit der Absenderadresse `nobody@gmx.net` erzeugen.

Beim klassischen Austausch von Dokumenten stellt normalerweise die Unterschrift des Absenders dessen Authentizität sicher. Beim elektronischen Nachrichtenaustausch leistet die elektronische Signatur vergleichbares, allerdings wesentlich zuverlässiger. Ver-

sieht also die Konstruktionsabteilung aus obigem Beispiel die Konstruktionspläne mit einer elektronischen Unterschrift, so kann die Fertigungsabteilung damit eindeutig den Urheber der Nachricht feststellen. Dies setzt allerdings voraus, dass die Identität und die Unterschrift der Fertigungsabteilung zweifelsfrei bekannt sind, damit niemand diese Unterschrift fälschen kann.

2.1.3. Integrität

Noch wichtiger als die Vertraulichkeit der Daten ist oft, dass diese während des Transports nicht verändert werden. Schickt z.B. oben erwähnte Entwicklungsabteilung die Pläne zur Fertigung, so wäre es verheerend, wenn diese während des Transports so manipuliert würden, dass das Produkt am Ende fehlerhaft ist.

Früher sollten Umschläge und Siegel sicherstellen, dass ein Brief während des Transports nicht geöffnet und verändert werden konnte. Dieses Konzept kann man in elektronischen Systemen mit kryptographischen Checksummen in weit leistungsfähigerer Form umsetzen. Allerdings muss auch hierbei die Identität überprüft werden, da sonst diese Checksummen gefälscht werden können.

2.1.4. Verfügbarkeit

Ein weiterer wichtiger Aspekt ist die Verfügbarkeit der genutzten IT-Systeme. Zunächst ist natürlich die korrekte und fehlerfreie Arbeitsweise der beteiligten IT-Komponenten sicherzustellen. Dies fällt in den Bereich der „Safety“ und soll, wie bereits oben erwähnt, hier nicht weiter diskutiert werden.

Darüber hinaus kann die Verfügbarkeit jedoch auch durch gezielte Angriffe von außen oder innen gestört werden. Dann ist auch die „Security“ gefragt, da sich ein verteiltes IT-System gegen derartige Angriffe schützen muss.

Dabei muss die Verfügbarkeit auf mehreren Schichten sichergestellt werden. Zunächst ist natürlich die Hardware selbst entsprechend abzusichern. So ist ohne entsprechende Schutzmaßnahmen bereits ein einfacher Stromausfall das Ende jeglicher Verfügbarkeit für alle IT-Systeme. Auch sollten die vorhandenen Ressourcen (wie Festplattenplatz, CPU-Leistung, Netzwerkkapazität) ausschließlich berechtigten Nutzern zugänglich sein. Sonst kann ein Angreifer durch gezieltes Überlasten bestimmter Komponenten die Verfügbarkeit dieser Ressourcen beliebig einschränken.

2.1.5. Verbindlichkeit

Je wichtiger elektronische Kommunikation z.B. bei Geschäftsprozessen wird, desto höher sind auch die Anforderungen an die zu Grunde liegenden Sicherheitssysteme. Spätestens wenn über ein solches Kommunikationssystem größere Finanz-Transaktionen abgewickelt werden, will der Empfänger nicht nur die Identität des Absenders zweifelsfrei prüfen können (Authentizität) und sicher sein, dass die Nachricht nicht verändert wurde (Integrität), er will im Falle eines Rechtsstreits dem Absender auch zweifelsfrei nachweisen können, dass dieser die Nachricht wirklich geschickt hat. Es soll also z.B.

ausgeschlossen sein, dass jemand eine gültige Nachricht eines Absenders dem Empfänger ein zweites Mal zustellt und so bspw. eine erneute Überweisung veranlasst.

Hierzu sind in der Regel neben digitalen Signaturen und kryptographischen Checksummen noch verlässliche Zeitstempel notwendig.

2.1.6. Zugriffskontrolle

Aufbauend auf der festgestellten Identität eines Kommunikationspartners soll eine Sicherheitsarchitektur dann den Zugriff auf bestimmte Dienste des Netzwerkes oder der darin verfügbaren Anwendungen gewähren oder verweigern. Dies ist naturgemäß eng mit der Eigenschaft der Authentizität verbunden.

So kann bestimmten Benutzern mit einem gültigen Account beispielsweise der Login auf einem Rechner erlaubt, anderen jedoch verweigert werden.

Je nach konkreter Anwendung sind alle diese Eigenschaften mehr oder weniger wichtig. Während ein Sicherheitssystem, welches für eine konkrete Anwendung konzipiert wird, dieser unterschiedlichen Gewichtung Rechnung tragen kann, sollte eine allgemeine Sicherheitsarchitektur alle oben genannten Eigenschaften aufweisen.

2.2. Mögliche Bedrohungen

Definition 2.1 (Bedrohung) *Eine* Bedrohung *(engl.* threat*) in einem Kommunikationssystem ist jedes Ereignis oder jede Folge von Handlungen, welche eine oder mehrere der oben genannten Eigenschaften (Vertraulichkeit, Authentizität, Integrität, Verfügbarkeit, Verbindlichkeit, Zugriffskontrolle) verletzt.*

Definition 2.2 (Angriff) *Eine Bedrohung manifestiert sich in einem* Angriff *(engl.* attack*) auf ein Kommunikationssystem, also in einer Folge von Handlungen, die das Ziel haben, eine oder mehrere der genannten Sicherheitseigenschaften zu verletzen.*

Dabei können diese Angriffe je nach Intention eines Angreifers recht unterschiedliche Formen annehmen. In der Literatur finden sich eine große Vielzahl verschiedener Klassifikationen (siehe beispielsweise [Sch96, Sch00, CB95, GS96, CZ95]). Manche trennen dabei primär nach der Intention des Angreifers, andere z.B. nach bestimmten technischen Aspekten des benutzten Angriffs. Je nach genutzter Skala sind dann aber manche Angriffsformen nicht mehr klar zu trennen. Erschwerend kommt hinzu, dass bestimmte Angriffe oft aus einzelnen Teilen aufgebaut sind, die unterschiedlich zu klassifizieren sind.

[JDHC97] setzt sich in Kapitel 6 ausführlich mit den Problemen der Klassifikation von Angreifern auseinander und gibt einen umfassenden Überblick über die verschiedenen Herangehensweisen. Aus den genannten Gründen kann keine der nun vorgestellten Kategorisierungen einen Anspruch auf Vollständigkeit erheben.

2.2.1. Klassifikation nach Intention

Bei der Klassifikation nach Intention wird primär nach dem Ziel des Angriffs gefragt. Das zweite Modell klassifiziert Angriffe eher aus technischer Sicht. Auch hier sind jeweils die gängigen englischsprachigen Fachbegriffe mit angegeben.

- Störung des Netzwerks (Denial of Service, DoS)
- Zugriff auf Informationen (Information Theft)
- Eindringen ins Netz/auf Knoten (Intrusion)
- Veränderung von Informationen (Tampering)

Störung des Netzwerks

Ist das Ziel eines Angriffs die Störung des Netzwerks, so sollen hier in der Regel andere Benutzer an der regulären Nutzung des Kommunikationssystems gehindert werden. So wurden im Internet bereits des Öfteren die Root-Nameserver des Domain Name Systems (DNS) durch solche *Denial of Service (DoS)* Angriffe außer Funktion gesetzt. Dies hatte zur Folge, dass Namen nicht in IP-Adresse aufgelöst werden konnten, weshalb die Nutzung von Diensten, die das DNS voraussetzen, nicht möglich war. Somit war beispielsweise der Zugriff auf Informationen im WWW in der Regel unmöglich, weil die Browser die IP Adressen der Web-Server nicht finden konnten.

Gängige Kommunikationssysteme sind oft sehr anfällig gegen solche DoS Angriffe, da die Sicherheit bei deren Design oft nicht oder nur unzureichend berücksichtigt wurde. So gehören DoS Attacken zu den am einfachsten durchzuführenden Angriffsformen. Der Angreifer zieht aus dem DoS Angriff in der Regel keinen direkten persönlichen Vorteil, die Vergangenheit zeigt jedoch, dass offensichtlich viele Personen trotzdem eine genügend hohe Motivation zu deren Durchführung besitzen.

Zugriff auf Informationen

Hier möchte der Angreifer auf Daten zugreifen, welche über das Kommunikationssystem übertragen werden oder in den angeschlossenen Rechnern gespeichert sind und die gemäß dem Prinzip der Vertraulichkeit nicht allgemein zugänglich sein sollen. Dies ist in der Regel recht einfach, da auch heute noch viele Daten unverschlüsselt übertragen werden. Während man in einem klassischen, leitungsgebundenen Netzwerk immerhin noch physikalischen Zugang zum Übertragungssystem (z.B. Ethernet-Buchse) benötigt, reicht bei drahtlosen Funknetzen bereits die räumliche Nähe zur Sendestation zum Abhören der Übertragung. So können Wireless LANs mit geeigneten Antennen aus mehreren Kilometern Entfernung abgehört werden [Mar02].

Ein anderer Weg, um unbefugt Zugriff auf Informationen zu erhalten, führt über die Anwendungen, welche in einem verteilten System zur Verfügung stehen. So lassen (bzw. ließen) sich viele Web-Server auf Grund von Implementierungsfehlern dazu bringen, Dateien außerhalb des Dokumentenverzeichnisses auszuliefern. Damit konnten dann beispielsweise Dateien mit Kundendaten und Kreditkartennummern aus dem Internet abgerufen werden.

Veränderung von Informationen

Schwieriger als der reine Zugriff auf Informationen ist unter Umständen deren Veränderung. Während beispielsweise in einem Funknetz Datenpakete relativ leicht abgehört werden können, ist es deutlich schwieriger (wenn auch keineswegs unmöglich), diese abzufangen und in veränderter Weise wieder ins Netz einzuspielen, ohne dass der Empfänger das ursprüngliche Datenpaket empfangen kann. Ein schönes Beispiel für die Veränderung von Informationen in einem Kommunikationssystem liefert [EF01]. Hier wurde im Rahmen einer Diplomarbeit ein WWW-Proxyserver an einer kleinen privaten Hochschule in Stuttgart so modifiziert, dass er den Text von Dokumenten gezielt veränderte, bevor er diese an die anfragenden Browser weiterleitete. Interessanterweise fielen diese Veränderungen praktisch niemandem auf.

Eindringen ins Netz/auf Knoten

Oft ist es das primäre Ziel eines Angreifers, generell Zugang zu einem Netzwerk oder einem Knoten im Netz zu erhalten. So mag das illegal genutzte Netzwerk als preiswerter Internet-Zugang dienen oder der fremde Account auf einem Fileserver dient als billige Ablage für die eigene MP3-Sammlung, welche über diesen Rechner gleichzeitig noch im Internet angeboten wird. Oft dient der Zugriff auf einen fremden Rechner auch als Ausgangsbasis für weitere Angriffe. So kompromittiert ein Angreifer bei den sogenannten Distributed Denial of Service Angriffen (DDoS) zunächst eine große Anzahl fremder Rechner. Deren aggregierte Netzwerkbandbreite wird dann für einen DoS Angriff auf einen leistungsstarken Internet-Server verwendet, indem dieser von allen Rechnern gleichzeitig mit Daten überschüttet wird [KMSW01].

2.2.2. Klassifikation nach Verfahren

Ein anderer Ansatz (nach [Sch02]) klassifiziert die Angriffe nach der Art den verwendeten Angriffstechniken:

- Verstellung (Masquerade)
- Mithören (Eavesdropping)
- Zugriffsverletzung (Authorization Violation)
- Verlust/Veränderung von Informationen
  (Loss or Modification of (transmitted) Information)
- Verleugnung des Kommunikationsvorgangs
  (Denial of Communication Act / Repudiation)
- Fälschen von Information (Forgery of Information)
- Sabotage

Unter Umständen kann hier, wie im Falle des Fälschens von Informationen, zwar das Verfahren mit der Intention übereinstimmen, im Allgemeinen sagt aber z.B. die Verstellung noch nichts darüber aus, welches Ziel der Angreifer damit erreichen will. Oft werden auch verschiedene Verfahren kombiniert, um das Ziel des Angriffs zu erreichen.

Verstellung

Hierbei verwendet ein Objekt im Netzwerk die Identität eines anderen Objekts. So kann sich ein Benutzer durch Angabe falscher Account-Informatinen beim Login-Vorgang als ein anderer Benutzer ausgeben und so möglicherweise umfangreichere Berechtigungen erlangen. In einem anderen Beispiel könnte ein Rechner unter der IP Adresse eines anderen Rechners Datenpakete verschicken und so dessen Identität vortäuschen. Dies kann beispielsweise dazu dienen, den Verursacher eines DoS Angriffs zu verschleiern.

Mithören

Wie schon erwähnt ist es in einem Kommunikationssystem mitunter sehr leicht, übertragene Daten mitzuhören. So dient das von den Amerikanern und anderen Nationen betriebene Echelon-Netzwerk [Röt00] ausschließlich dazu, internationale Kommunikationsleitungen abzuhören. In einem Fall wurde beispielsweise ein Fax von Airbus an eine arabische Fluggesellschaft mit einem Angebot über die Lieferung mehrerer Flugzeuge abgefangen und dem Konkurrenten Boeing zugespielt, worauf dieser Airbus in letzter Minute unterbieten konnte. Hieraus sieht man sehr schön, dass bereits das unerlaubte Mithören in einem Kommunikationssystem weitreichende Folgen haben kann.

Zugriffsverletzung

Bei der Zugriffsverletzung wird versucht, die eigenen Berechtigungen, die einem vom System zugestanden werden, zu erweitern. So könnte ein Angreifer in einem Kommunikationssystem mit Bandbreitenbeschränkungen beispielsweise versuchen, mehr als die ihm zustehende Bandbreite zu verwenden. In einem Unix System könnte ein regulärer Benutzer versuchen, Superuser-Rechte zu erlangen, um auf fremde Dateien zugreifen oder diese verändern zu können.

Verlust/Veränderung von Informationen

Die Löschung von Daten in einem Kommunikationssystem kann auf unterschiedlichen Ebenen erfolgen, so könnte in einem Wireless LAN ein Störsender gezielt Datenpakete von einem Sender stören, was einen effektiven DoS Angriff gegen diesen Knoten darstellt. Auch die Veränderung von Informationen kann einen Angriff darstellen. So könnte ein Angreifer beispielsweise die Daten einer Online-Überweisung so ändern, dass das Geld auf seinem eigenen Konto landet.

Verleugnung des Kommunikationsvorgangs

Wie schon in der ersten Kategorisierung angesprochen, sollte ein Kommunikationssystem sicherstellen, dass ein Urheber einer Nachricht zweifelsfrei festgestellt werden kann. Erfolgt zum Beispiel von einem Gerät A ein Angriff auf ein anderes Gerät B, dann wird vermutlich der Besitzer von A für den Angriff verantwortlich sein und entsprechende Konsequenzen zu befürchten haben. Leugnet dieser die Urheberschaft des Angriffs,

so sollte das Sicherheitssystem eine zweifelsfreie Aussage darüber ermöglichen, ob A wirklich der Ausgangspunkt war oder nicht.

Fälschen von Informationen

Oft ist es in einem Kommunikationssystem recht einfach, Informationen zu fälschen. Beispielsweise lassen sich im Internet E-Mails mit beliebigen Absenderadressen generieren. Auch die IP Adresse eines IP Pakets lässt sich beliebig fälschen. Erst durch zusätzliche (kryptographische) Verfahren lassen sich solche Fälschungen verhindern.

Sabotage

Sabotage ist ein sehr allgemeiner Begriff. In unserem Kontext ist damit jede gezielte Handlung gemeint, welche das Kommunikationssystem oder darauf aufsetzende Dienste oder daran beteiligte Knoten in ihrer Funktionsfähigkeit einschränkt. Damit ist diese Form des Angriffs weitestgehend identisch mit der bereits erwähnten „Störung des Netzwerks“.

Damit ist die Vorstellung verschiedener Angriffsformen beendet. Wie bereits zu Anfang festgestellt, gibt es unzählige Möglichkeit, die Sicherheit eines Kommunikationssystems zu beeinträchtigen. In der Literatur finden sich viele Versuche der Kategorisierung. Je nach Standpunkt des Autors betont eine Kategorisierung einzelne Aspekte eines Angriffs und vernachlässigt andere. Auch lassen sich nie alle Angriffe zweifelsfrei einer Kategorie zuordnen, es gibt immer Zwischenformen, die in mehrere Kategorien passen. Somit ist eine Kategorisierung niemals vollständig oder absolut richtig, vielmehr dient sie als grober Anhalts- und Orientierungspunkt, der bei der Entwicklung von Sicherheitssystemen hilfreich ist.

2.3. Sicherheitsmechanismen

Sicherheitsmechanismen in einem Kommunikationssystem sind in der Regel hierarchisch aufgebaut. Auf der untersten Ebene stehen die elementaren *kryptographischen Algorithmen*, welche zum Beispiel die Verschlüsselung von Daten oder die Bildung einer kryptographischen Prüfsumme ermöglichen. Aus diesen elementaren Bestandteilen werden komplexere *kryptographische Protokolle* aufgebaut. Verschiedene dieser kryptographischen Protokolle kommen dann in den Anwendungen zum Einsatz. Zusammen bilden sie das *Sicherheitssystem*.

[Sta03] definiert den Begriff Sicherheitsmechanismen wie folgt:

Definition 2.3 (Sicherheitsmechanismus) *Ein Mechanismus der dazu entwickelt wurde, einen Angriff (s.o.) zu erkennen bzw. zu verhindern oder die Folgen eines solchen Angriffs zu beheben.*

Gängige Sicherheitsmechanismen lassen sich wie folgt einordnen:

Verschlüsselung: Durch Verschlüsselung wird eine Nachricht so transformiert, dass ihr ursprünglicher Inhalt für einen Angreifer nicht mehr zu erkennen ist. Nur der rechtmäßige Empfänger verfügt über das Wissen, um die Transformation

rückgängig zu machen und so an den ursprünglichen Inhalt der Nachricht zu gelangen.

Prüfsummen: Prüfsummen über Nachrichten dienen dazu, unberechtigte Manipulationen festzustellen.

Signaturen: Ein Absender bestätigt durch eine digitale Signatur seine Urheberschaft an einer Nachricht. Der Empfänger prüft mittels einer Signatur die Identität des Absenders.

Frischekennzeichen: Elemente in einer Nachricht, welche nur einmalig verwendet werden. Somit kann ein Empfänger wiederholt abgespielte Nachrichten erkennen und verwerfen.

Zugriffskontrolle: Der Zugriffskontrollmechanismus prüft vor jeder Aktion die Berechtigung (Autorisation) des Benutzers.

Diese Mechanismen werden durch *kryptographische Algorithmen* realisiert, welche im folgenden Kapitel beschrieben sind. Ein kryptographischer Algorithmus alleine ist in der Praxis noch nicht zu verwenden. Es fehlt noch eine genaue Beschreibung, wie die Berechnungen von den beteiligten Parteien auszuführen sind. Dies leistet das *kryptographische Protokoll*.

Es gibt eine Vielzahl von kryptographischen Protokollen für alle möglichen Anwendungen, z.B. Schlüsselaustausch, elektronisches Geld, Authentifizierung, Bit-Commitment, gegenseitige Vertragsunterzeichnung uvm. Eine gute Übersicht über die Vielfalt kryptographischer Protokolle liefert [Sch96]. Für die weiteren Ausführungen relevante Protokolle werden im nächsten Kapitel vorgestellt.

In der Praxis bestehen Anwendungen und die darin eingebetteten *Sicherheitssysteme* meist aus einer Vielzahl von Komponenten. Kommunikationsprotokolle wie SSL, TLS oder IPsec sichern den Transport der Daten ab, die Schlüssel selbst müssen im Rechner oder auf Chipkarten verwaltet werden und auch die Absicherung der Rechner durch Firewalls und Intrusion Detection Systeme muss berücksichtigt werden. Insgesamt gilt, dass die Absicherung einer ganzen Firmen-EDV mit einer Vielzahl von installierten Anwendungen, mit Netzverbindungen und Datenaustausch zu Geschäftspartnern, nur von geschulten Sicherheitsexperten geplant und durchgeführt werden kann. Dabei müssen diese oft auf ein umfangreiches Erfahrungswissen zurückgreifen, die formale Modellierung und Verifikation von Sicherheitssystemen und -protokollen steckt noch in den Kinderschuhen. Mehr hierzu im nächsten Kapitel.

2.4. Fazit

Wie man sieht, ist die Sicherheit in einem Kommunikationssystem vielfältigen Bedrohungen ausgesetzt. Entsprechend ist es unabdingbar, in jedes Anwendungssystem auch entsprechende Sicherheitsmechanismen zum Schutz vor eventuellen Angriffen einzubetten. Zum Verständnis der resultierenden, oftmals komplexen Sicherheitssysteme ist ein umfangreiches Fach- und Anwendungswissen notwendig. Im Rahmen dieser Arbeit soll das nächste Kapitel nun die notwendigen kryptographischen Grundlagen vermitteln, welche die Basis der später vorgestellten Sicherheitsarchitektur bilden.

3. Kryptographische Grundlagen

Laut [Ert01] ist *Kryptographie* die *„Lehre der Absicherung von Nachrichten durch Verschlüsselung“*. Im Gegensatz dazu ist *Kryptanalyse* die *„Kunst, Chiffretext aufzubrechen, d.h. den Klartext zu reproduzieren, ohne Kenntnis des Schlüssels“*. *Kryptologie* ist die Verbindung von Kryptographie und Kryptanalyse.

Dabei geht Kryptographie in der Realität weit über eine simple Verschlüsselung einer Nachricht hinaus und erfüllt vielfältige Aufgaben in der Absicherung von Kommunikationssystemen. Authentisierung von Benutzern, Anonymität, elektronisches Geld und vieles mehr wird heute mit Hilfe kryptographischer Algorithmen realisiert. Dieses Kapitel stellt die gängigsten kryptographischen Mechanismen vor und führt in die mathematischen Grundlagen ein.

3.1. Frischekennzeichen

Definition 3.1 (Nonce) *Eine Nonce ist ein* kryptographischer Einmalwert*, welcher die Frische einer Nachricht sicherstellt.*

Jede Nonce darf nur in genau einem Protokolldurchlauf eingesetzt und muss danach neu generiert werden. Um zu verhindern, dass ein Angreifer eine Nonce erraten kann, muss diese eine starke Zufallszahl sein. Kommen schwache Pseudozufallszahlengeneratoren zum Einsatz, so entstehen möglicherweise Angriffspunkte. Zur Generierung von Zufallszahlen siehe [Ert01, Sch96].

Nonces kommen in einer Vielzahl von kryptographischen Protokollen vor, wie dem Needham-Schroeder Schlüsselaustausch [NS78] oder dem Otway-Rees Protokoll [OR87]. Auch das in Kapitel 10 vorgestellte SDSR Protokoll benutzt Nonces.

Alternativ zu Nonces kommen oft auch *Zeitstempel* (engl. *Timestamps*) als Frischekennzeichen zum Einsatz. Die Überprüfung von Zeitstempeln setzt aber synchronisierte Uhren zwischen Sender und Empfänger einer Nachricht voraus, was oft die Möglichkeit von neuen Angriffen eröffnet.

3.2. Kryptographische Hashfunktionen

Oft soll über eine Nachricht eine Prüfsumme gebildet werden, z.B. um zu belegen, dass die Nachricht während des Transports über ein unsicheres Netzwerk nicht verändert wurde. Eine Funktion, welche eine solche Prüfsumme berechnet, nennt man *Hashfunktion*.

Definition 3.2 (Hashfunktion) *Eine Hashfunktion H ist eine Funktion, die einer beliebig langen Eingabe M einen Hashwert h fester Länge zuordnet. Der Funktionswert*

zu einer gegebenen Eingabe kann dabei mit geringem Aufwand berechnet werden (nach [Sch96, Wät02]).

Ein bekanntes Prüfsummenverfahren ist der *Cyclic Redundancy Check (CRC)*. In der Form CRC-32 ordnet er einer beliebigen Eingabe eine 32 Bit Prüfsumme zu. Für kryptographische Anwendungen sind solche herkömmlichen Prüfsummenverfahren aber ungeeignet, da sie es einem geschickten Angreifer ermöglichen, Änderungen an der Nachricht durchzuführen, bei denen die Prüfsumme gleich bleibt. *Kryptographische Hashfunktionen* hingegen müssen *Einwegfunktionen* sein.

Definition 3.3 (Einwegfunktion) *H ist eine Einwegfunktion (one-way function), wenn es nur mit sehr großem Aufwand möglich ist, zu einem vorgegebenen Hashwert h eine Nachricht M zu finden, so dass H(M) = h gilt.*

Je größer der Wertebereich von H, desto schwieriger ist dies natürlich. Weiterhin sollte die Funktion H die Werte möglichst zufällig im Wertebereich streuen. Wichtig ist in diesem Zusammenhang noch der Begriff der Kollisionsresistenz.

Definition 3.4 (Kollisionsresistenz) *H heißt schwach kollisionsresistent, wenn zu einer festen Nachricht M nur mit sehr großem Aufwand eine Nachricht $M' \neq M$ gefunden werden kann, für die $H(M) = H(M')$ gilt. H heißt stark kollisionsresistent, wenn es sehr schwierig ist, zwei beliebige Nachrichten M und M' ($M \neq M'$) zu finden mit $H(M) = H(M')$.*

CRC-32 ist in kryptographischem Sinne keine Einwegfunktion, sie ist somit auch weder stark noch schwach kollisionsresistent. Gängige kryptographische Hashfunktionen wie *MD5* [Riv92b] oder *SHA-1* [Sta94] erfüllen alle genannten Anforderungen. Die Berechnung der Hashfunktion erfolgt in der Regel rundenbasiert, wobei in jeder Runde ein neuer Block Eingabedaten verarbeitet wird. Zur genauen Funktionsweise der Algorithmen siehe z.B. [Sta03] oder [Sch96].

Kommt bei der Prüfsummenbildung zusätzlich noch ein geheimer Schlüssel k zum Einsatz, so spricht man von einem sogenannten *Keyed Hash* bzw. von einem *Message Authentication Code (MAC)*. Ein gängiges Beispiel hierfür ist der HMAC [KBC97], welcher sich einer beliebigen Hashfunktion bedient. In Kombination mit MD5 spricht man dann beispielsweise von HMAC_MD5. Bei der Bildung eines HMAC_MD5 Wertes über den Wert *text* bei Passwort K bildet man:

$$HMAC\_MD5(K, text) = MD5(K \oplus opad, MD5(K \oplus ipad, text))$$

Dabei ist *ipad* definiert als Byte 0x36 und *opad* als Byte 0x5c jeweils zur Blocklänge L der verwendeten Hashfunktion konkateniert. Für MD5 gilt dabei L=16 und für den analogen HMAC_SHA-1 ist L=20.

Hashchains

Ein interessanter Weg, Hashfunktionen einzusetzen, sind sogenannte *Hashchains*. Hierbei wird eine Hashfunktion H wiederholt auf einen Startwert s angewendet und es ergibt sich eine verkettete Folge von Hashwerten H_1, H_2, ..., H_n:

$$H_1 = H(s); H_2 = H(H_1) = H(H(s); ...$$

Dabei kann zwar bei Kenntnis von H_i das zugehörige H_{i+1} berechnet werden, aber nicht umgekehrt. Hashchains lassen sich zur effizienten Authentisierung einsetzen [Lam81, Gue02a, GA02]. Hierzu berechnet ein Knoten eine Hashchain, ausgehend von einem Startwert. Den letzten Wert dieser Kette H_n schickt er nun an einen anderen Knoten und sichert diese Information z.B. durch eine (aufwändige) Signatur ab. Will er dem Knoten später beweisen, dass er immer noch der gleiche Kommunikationspartner ist, so genügt es, wenn er H_{n-1} angibt. Der Empfänger prüft lediglich, ob $H(H_{n-1}) = H_n$. Da H eine Einwegfunktion ist, kann ohne Kenntnis von s kein anderer Knoten H_{n-1} bilden.

3.3. Verschlüsselung

Ein *Alphabet* A ist eine endliche Menge von Zeichen, $|A|$ ist die *Mächtigkeit von* A. Sei M der *Klartext (plaintext)* der Nachricht, gebildet über A. Beispielsweise sind *abcabc* oder *abbbba* Klartexte über dem Alphabet a, b, c. Der *Geheimtext* oder *Chiffretext (ciphertext)* C ist die verschlüsselte Nachricht über dem gleichen oder einem anderen Alphabet.

Die *Verschlüsselungsfunktion* oder *Chiffre (cipher)* ist eine Funktion E, welche unter Verwendung des *Schlüssels (key)* K den Klartext M nach C überführt. Die Umkehrfunktion $D = E^{-1}$ führt C in M über. Dies nennt man *Entschlüsselung*. Entsprechend dieser Definitionen gilt $E(M) = C$ und $D(C) = M$, woraus

$$D(E(M)) = M$$

folgt. D.h. nach erfolgter Ver- und Entschlüsselung kommt wieder der Klartext zum Vorschein. Bei der *symmetrischen Verschlüsselung* wird für Ver- und Entschlüsselung der gleiche Schlüssel K verwendet, daher die Bezeichnung symmetrisch. Formal ausgedrückt gilt

$$E_K(M) = C \text{ und } D_K(C) = M$$

Umgekehrt kommen bei *asymmetrischen Verschlüsselungsverfahren* für Ver- und Entschlüsselung getrennte Schlüssel PK und SK zum Einsatz, so dass gilt

$$E_{PK}(M) = C \text{ und } D_{SK}(C) = M$$

Ferner unterscheidet man *Stromchiffren* und *Blockchiffren*. Stromchiffren ver- bzw. entschlüsseln eine Klartextnachricht zeichenweise, während Blockchiffren auf ganzen Blöcken von Zeichen arbeiten. In der Frühzeit der Kryptographie kamen oft sogenannte *eingeschränkte Algorithmen* zum Einsatz, bei denen die Sicherheit des Verfahrens neben dem Schlüssel auch von der Geheimhaltung des Algorithmus abhängt. Dieses wird heute nicht mehr akzeptiert, bereits im 19. Jahrhundert forderte A. Kerkhoff [Kah91]:

> Die Sicherheit eines Verschlüsselungsverfahrens darf nur von der Geheimhaltung des Schlüssels abhängen, nicht jedoch von der Geheimhaltung des Algorithmus.

Viele Beispiele, wie die später geschilderte WEP Verschlüsselung (siehe Abschnitt 4.3.2, zeigen, dass nur die frühzeitige Offenlegung eines Kryptoalgorithmus und die breite Diskussion in der Fachgemeinschaft elementare Designfehler ausschließen und das Vertrauen in ein Verfahren stärken können. Dem Kryptoanalytiker stehen heute mächtige Werkzeuge wie differentielle oder lineare Kryptoanalyse bzw. Side-Channel-Angriffe wie Timing- und Power-Attacks zur Verfügung [Bau00]. Daneben werden oft Fehler im Design von kryptographischen Protokollen gemacht, welche mit der Kryptographie selbst zunächst nichts zu tun haben, sich aber genauso verheerend für die Sicherheit der Anwendungen auswirken. Aus diesem Grund ist es selbstverständlich Ziel dieser Arbeit, ein Sicherheitssystem zu realisieren, das einem Angreifer auch bei detaillierter Kenntnis der verwendeten Verfahren standhält.

3.3.1. Symmetrische Verschlüsselungsverfahren

Typische symmetrische Verschlüsselungsverfahren sind der *Data Encryption Standard (DES)* [Sta77], der *International Data Encryption Algorithm (IDEA)* [LM90] oder der *Advanced Encryption Standard (AES)* „Rijndael“ [DR99, DR00]. Allen gemeinsam ist, dass sie aus einem Satz elementarer Operationen aufgebaut sind. Es kommen Permutationen (Vertauschungen) und Substitutionen (Ersetzungen) zum Einsatz. Gemäß den Forderungen von Shannon [Sha49] sollen diese eine möglichst optimale Kombination aus *Konfusion* und *Diffusion* erzeugen. Konfusion bedeutet, dass die Klartextbits möglichst gut gemischt werden, um die statistischen Eigenschaften des Textes zu verbergen. Im Idealfall gleicht der Chiffretext am Ende einer Folge von Zufallszahlen. Unter Diffusion versteht man die Tatsache, dass der Informationsgehalt jedes Klartextbits möglichst gleichmäßig auf die Ausgangsbits des Chiffretexts verteilt werden soll. Dies erschwert Angriffe, die auf der Zuordnung eines Klartextteils zu einem bestimmten Chiffretextteil und dem Schlüssel beruhen. Aus Effizienzgründen arbeiten gängige Algorithmen allerdings nie auf dem gesamten Klar-/Chiffretext, sondern teilen diesen in Blöcke ein, was die Diffusion begrenzt. Die Blockgröße ist also ein Kompromiss zwischen Effizienz und Sicherheit.

AES

Im Folgenden wird der *Advanced Encryption Standard* als ein typischer Vertreter einer modernen symmetrischen Chiffre vorgestellt. Dies bietet sich an, da dieser auch im Rahmen der MANET Sicherheitsarchitektur zum Einsatz kommt. Im Jahr 1997 hat das *National Institute of Standards (NIST)* einen Wettbewerb zur Suche eines DES Nachfolgers initiiert [Sta97]. In einem mehrstufigen, offenen Verfahren mussten sich die Kandidaten intensiven Prüfungen unterziehen. Insbesondere wurden die folgenden Anforderungen gestellt [Sta97, Ert01]:

Formal sollte AES eine symmetrische Blockchiffre sein, die bei einer Blockgröße von 128 Bit Schlüssellängen von 128, 192 und 256 Bit unterstützt.

Sicherheit gegen Angriffe aller Art. Gefordert war insbesondere eine mathematische Verifikation der Sicherheit.

Einfachheit des Designs.

Flexibilität: AES sollte nach Möglichkeit auch weitere Block- und Schlüsselgrößen unterstützen.

Effizient: AES sollte effizienter sein als sein Vorgänger DES.

Implementierung in Hard- und Software sollte einfach und effizient möglich sein.

Nach einem langen Auswahlverfahren wurde am 2. Oktober 2000 die *Rijndael* Chiffre der belgischen Kryptographen Joan Daemen und Vincent Rijmen zum neuen Krypto-Standard erklärt. Rijndael ist ähnlich wie DES eine iterierte Blockchiffre, deren Blocklänge b und Schlüssellänge k unabhängig auf einen der Werte 128, 192 und 256 Bit gesetzt werden können. Abhängig von den Werten schwankt die Rundenzahl zwischen 10 und 14. Abbildung 3.1 zeigt den Ablauf für $b = 128$ und $k = 192$ mit 12 Runden.

Die Schlüsselexpansion erzeugt aus dem Schlüssel k abhängig von der Rundenzahl r $r + 1$ Rundenschlüssel k_i mit Länge b. Der Rundenschlüssel wird an den angegebenen Stellen mit dem aktuellen Zustand XOR verknüpft. Der Zustand wird dabei je nach Blockgröße als 4x4, 6x4 oder 8x4 Byte Matrix repräsentiert. Das Ergebnis der XOR Operation dient dann als Eingabe für die nächste Runde bzw. ergibt am Ende den Chiffretext. Jede Runde besteht aus den drei Operationen *ByteSub*, *ShiftRow* und *MixColumn*, wobei in der letzten Runde MixColumn wegfällt.

Die *ByteSub* Transformationen entspricht einer nicht-linearen S-Box und wird auf jedes Byte des Zustands angewendet. Dabei wird zunächst die multiplikative Inverse über dem Galois Feld $GF(2^8)$ berechnet (zu Details der Mathematik von Galois Feldern siehe [Ert01], Anhang A.4.2). Implementiert ist dies aus Effizienzgründen meist als Lookup in eine vordefinierte Tabelle. Danach schließt sich noch eine affine Transformation mit einer festen Transformationsmatrix an. Bei der Entschlüsselung muss dann lediglich die Transformationsmatrix invertiert werden.

ShiftRow verschiebt die Zeilen 1 bis 3 der Zustandsmatrix zyklisch um festgelegte Werte (abhängig von der Blockgröße) nach rechts. Zeile 0 bleibt unverändert. Zum Entschlüsseln wird analog nach links verschoben.

Die *MixColumn* Transformation schließlich modifiziert die Spalten $(a_{0,i}, a_{1,i}, a_{2,i}, a_{3,i})$ des Zustandes durch folgende Matrixmultiplikation

$$\begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} = \begin{pmatrix} a_{0,i} \\ a_{1,i} \\ a_{2,i} \\ a_{3,i} \end{pmatrix}$$

Die Operationen werden wieder in $GF(2^8)$ ausgeführt. Für die anderen Blockgrößen existieren entsprechende Matrizen und Invertierungen für die Entschlüsselung.

Da jede Operation von Rijndael invertierbar ist, kann die entsprechende Umkehrfunktion einfach angegeben werden. Hierzu sind lediglich alle Matrizen zu invertieren und der Ablauf umzukehren.

Erfolgt die Verschlüsselung eines längeren Klartextes mit Rijndael (oder jeder anderen Blockchiffre) blockweise, so könnte ein Angreifer durch Analyse des Chiffretextes diese Blockstruktur erkennen und sich bei bekanntem Klartext eine Art elektronisches Codebuch (*electronic codebook (ECB)*) generieren, in dem er sich zu jedem Klartext

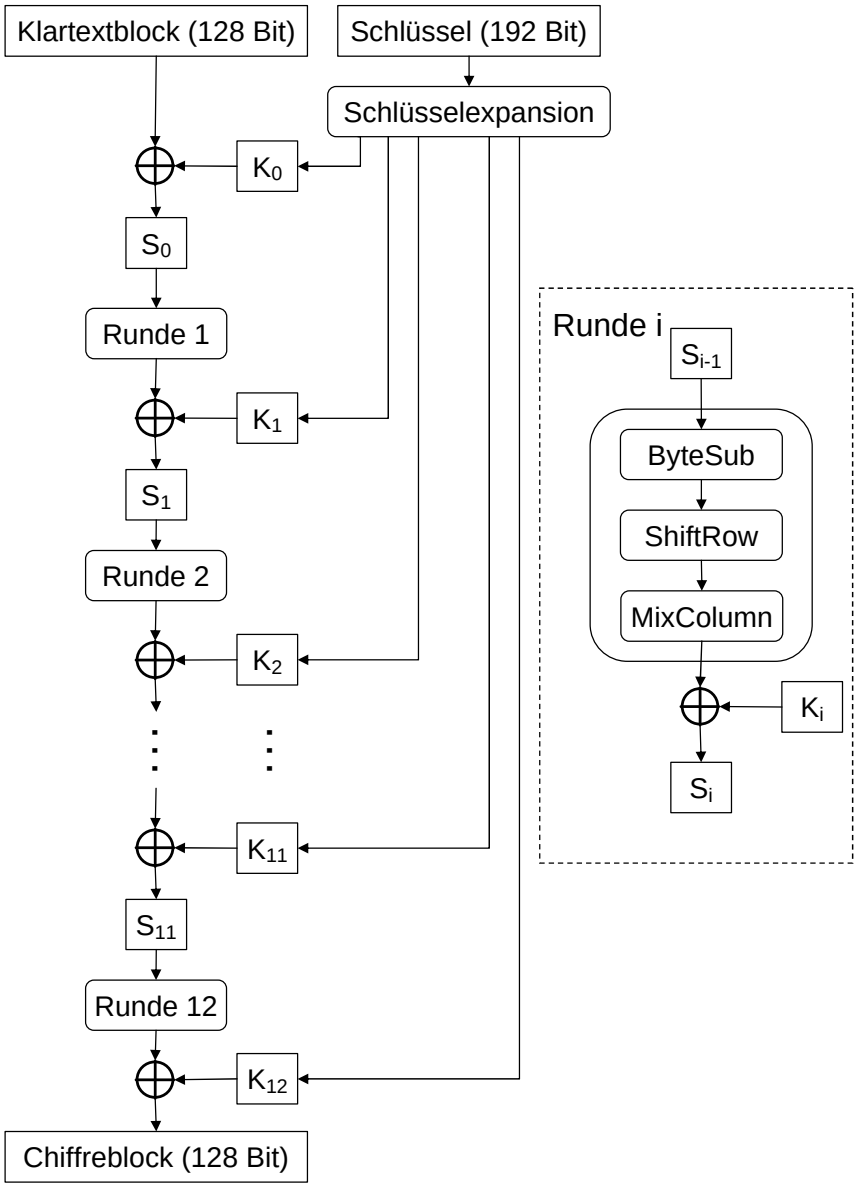

Abbildung 3.1.: Überblick Advanced Encryption Standard AES (aus [Ert01])

Block den zugehörigen Chiffretext Block merkt. Diese Art, eine Blockchiffre zu verwenden, nennt man deswegen auch *ECB-Modus*. Weiterhin könnte ein Angreifer einzelne Blöcke gegen andere vertauschen, ohne dass dies der Empfänger merkt. Um derartige Probleme zu umgehen, verkettet man die Blöcke im *CBC-Modus (cipher block chain)* miteinander:

$$\begin{aligned} C_0 &= E_K(M_0) \\ C_1 &= E_K(C_0 \oplus M_1) \\ &\vdots \\ C_i &= E_K(C_{i-1} \oplus M_i) \\ &\vdots \end{aligned}$$

Außer zur Verschlüsselung kommt Rijndael auch als *Message Authentication Code (MAC)*, als Einweg-Hashfunktion sowie als Stromchiffre und als Pseudozufallszahlengenerator zum Einsatz.

Analysen von Rijndael haben ergeben, dass der Algorithmus um ein Vielfaches schneller ist als DES. Es lassen sich auf gängigen PCs Software-Implementierungen realisieren, die mit mehreren hundert Mbps ver- und entschlüsseln können. Mit spezialisierter Hardware sind problemlos Leistungen im Gbps Bereich möglich. Damit bietet AES eine Leistung, die heutigen Hochgeschwindigkeitsnetzwerken gerecht wird.

AES ist darüber hinaus leicht parallelisierbar, was gerade für Software-Implementierungen auf entsprechend ausgelegten Prozessoren, wie dem Pentium-4 mit Hyperthreading, von Vorteil ist. Da die Elemente der Operationen Bytes sind, ist selbst auf einfachen Prozessoren in eingebetteten Systemen eine effiziente Implementierung möglich. So benötigt Rijndael auf einem Intel 8051 nur etwa ein Kilobyte Programmspeicher und bei 128 Bit Blocklänge je nach Schlüssellänge 36 bis 53 Byte RAM [Ert01].

Rijndael ist einer der in den letzten Jahren am meisten untersuchten Verschlüsselungsalgorithmen und ein gewisses Vertrauen in die Sicherheit dieses Verfahrens ist daher berechtigt. Allerdings soll nicht unerwähnt bleiben, dass einige Arbeiten etwas an der Sicherheit von Rijndael gekratzt haben. In [CP02] zeigen die Autoren, dass die S-Boxen von Rijndael in mathematischem Sinne überspezifiziert sind. Danach lässt sich Rijndael mit 256 Bit Schlüssellänge als ein System von 8000 quadratischen Gleichungen mit 1600 binären Unbekannten spezifizieren. Eine Lösung dieses Systems entspricht der erfolgreichen Kryptanalyse von Rijndael. Aufbauend auf dem XL-Algorithmus [SPCK02] entwickeln die Autoren eine XSL genannte Angriffsform, welche in [MR02] noch verfeinert wird. Ähnliche Ergebnisse finden Fuller und Millan in [FM02]. Diese und andere Arbeiten haben gezeigt, dass ein Angriff auf Rijndael möglich ist, dessen Komplexität nicht exponentiell mit der Zahl der Runden des Algorithmus wächst. Wie ein solches Gleichungssystem effizient zu lösen ist, ist allerdings weiterhin unbekannt, so dass die praktische Sicherheit von AES nicht gefährdet ist.

3.3.2. Asymmetrische Verschlüsselungsverfahren

Trotz aller beschriebenen Leistungen haben die symmetrischen Verschlüsselungsverfahren ein grundsätzliches Problem. Der geheime Schlüssel muss vor der Kommunikation über einen sicheren Kanal ausgetauscht werden. Gerade im Internet (aber auch in Mobilen Ad hoc Netzwerken) ist dies nicht immer möglich, da sich die Kommunikationspartner oft nicht kennen und auch kein sicherer Kanal zum Austausch zur Verfügung steht.

Ralph Merkle war der Erste, der bereits im Jahr 1974 eine Lösung für dieses Problem fand [Sch96]. Sein „Merkles Rätsel“ genanntes Verfahren ist jedoch in der Praxis kaum einsetzbar. Den ersten einsetzbaren Public Key Algorithmus veröffentlichten Diffie und Hellmann 1976 [DH76], da dieser jedoch nur zum Schlüsselaustausch verwendet werden kann, wird er in Abschnitt 3.4.1 besprochen.

Das grundsätzlich Neue an der asymmetrischen Kryptographie ist die Idee, für Ver- und Entschlüsselung zwei getrennte Schlüssel zu verwenden. An die Stelle des einen Schlüssels bei symmetrischen Verfahren tritt jetzt ein *Schlüsselpaar* SK[1] und PK[2]. Dabei darf SK nicht ohne unverhältnismäßig hohen Aufwand aus PK herleitbar sein. Ver- und Entschlüsselung funktionieren dann wie folgt:

$$E_{PK}(M) = C \text{ und } D_{SK}(C) = M$$

Den Schlüssel SK_A[3] hält Alice, die Besitzerin des Schlüsselpaares, typischerweise geheim, PK_A hingegen wird veröffentlicht. Aus diesem Grund spricht man auch von *Verschlüsselung mit öffentlichen Schlüsseln* oder *Public-Key Cryptography*. Mittels des öffentlichen Schlüssels von Alice PK_A kann Bob eine Nachricht für Alice verschlüsseln, die nur diese mit SK_A wieder entschlüsseln kann.

Im Vergleich mit symmetrischen Verschlüsselungsverfahren fallen einige Nachteile der asymmetrischen Verfahren auf:

1. Um den gleichen Grad an Sicherheit zu bieten, benötigen asymmetrische Verfahren deutlich längere Schlüssel. Während die meisten heute gängigen symmetrischen Verfahren Schlüssellängen zwischen 128 und 256 Bit verwenden, gelten beispielsweise bei RSA erst 1024 bis 2048 Bit als wirklich sicher.

2. Die Rechenoperationen bei asymmetrischen Verfahren sind in der Regel deutlich aufwändiger. Entsprechend liegt der erreichbare Durchsatz um Größenordnungen unter dem von schnellen symmetrischen Verfahren. Eine Software-Implementierung des später vorgestellten RSA Algorithmus ist etwa 100- bis 1000-mal langsamer als AES, bei Hardware-Implementierungen gelten sogar Faktoren von etwa 1000 bis 10000.

Aus diesem Grund werden in realen Systemen meist Kombinationen beider Verfahren eingesetzt. Das asymmetrische Verfahren dient zum Austausch einer Zufallszahl, welche dann als Schlüssel für ein symmetrisches Verfahren dient, mit dem der eigentliche Datenverkehr verschlüsselt wird.

[1] für *secret key*
[2] für *public key*
[3] für secret key von A

RSA

Der RSA Algorithmus [RSA78] ist benannt nach seinen Erfindern Ron Rivest, Adi Shamir und Leonard Adleman. RSA ist nach wie vor die populärste asymmetrische Chiffre, insbesondere da seit September 2000 das Patent von RSA Inc. abgelaufen ist und das Verfahren nun frei verwendet werden kann. Da auch dieses Verfahren später in der Sicherheitsinfrastruktur eingesetzt wird, soll es an dieser Stelle etwas genauer vorgestellt werden (siehe auch [Ert01]).

Von zentraler Bedeutung für RSA ist die korrekte Generierung der Schlüsselpaare, da hiervon die Sicherheit der verschlüsselten Daten abhängt. Anschließend können die generierten Schlüssel zur Ver- und Entschlüsselung genutzt werden.

Definition 3.5 (Schlüsselerzeugung für RSA)

1. *Wähle zufällig zwei große Primzahlen p und q. Die Größe dieser Zahlen liegt heute typischerweise in der Größenordnung von 512 bis 1024 Bit.*
2. *Berechne $n = pq$ (n hat dann also eine Länge von 1024 bis 2048 Bit).*
3. *Setze $\varphi(n) = (p-1)(q-1)$ und wähle e relativ prim zu $\varphi(n)$, d.h. es muss gelten $ggT(e, \varphi(n)) = 1$.*
4. *Berechne d als Lösung von $ed = 1 \bmod \varphi(n)$. Wie in [BRK95] und [CLR92] nachzulesen, ist d eindeutig bestimmt und kann mit dem erweiterten Euklidischen Algorithmus berechnet werden.*
5. $PK = (e, n)$ *ist der* öffentliche Schlüssel
6. $SK = (d, n)$ *ist der* geheime Schlüssel

Definition 3.6 (Anwendung von RSA)

Verschlüsseln: *Die Nachricht $M \in \mathbb{Z}_n$ (Körper der ganzen Zahlen modulo n) wird codiert durch*

$$C = E_{PK}(M) = M^e \bmod n$$

Entschlüsseln: *Ein Chiffretext $C \epsilon \mathbb{Z}_n$ wird dekodiert durch*

$$M = D_{SK}(C) = C^d \bmod n$$

Der effizienteste, bisher bekannte Angriff auf RSA beruht auf der Primfaktorzerlegung von n. Bei bekanntem p und q lässt sich dann $\varphi(n)$ und somit auch d berechnen. Damit ist der geheime Schlüssel dem Angreifer bekannt. Alternativ könnte man auch Nachrichten entschlüsseln, indem man $M^e \bmod n$ nach M auflöst. Dazu müsste man allerdings $\sqrt[e]{C} \bmod n$ berechnen. Die Sicherheit von RSA beruht also maßgeblich auf der Komplexität der Primfaktorzerlegung bzw. des diskreten Wurzelziehens $\bmod n$. Selbst die effizientesten bekannten Verfahren sind immer noch so aufwändig, dass RSA mit $|n| = 1024$ Bit noch bis etwa 2037 sicher sein sollte [Sil00]. Es kann allerdings nicht ausgeschlossen werden, dass neue bahnbrechende Verfahren entwickelt oder gänzlich neue Wege zur Entschlüsselung von RSA gefunden werden.

3.4. Kryptographische Protokolle

Ein kryptographischer Algorithmus alleine ist in der Praxis noch nicht zu verwenden. Es fehlt noch eine genaue Beschreibung, wie die Berechnungen von den beteiligten Parteien auszuführen sind. Dies leistet das *kryptographische Protokoll*. Erst durch ein kryptographisches Protokoll kann ein Algorithmus wie RSA wahlweise zur Verschlüsselung, für den Austausch von Sitzungsschlüsseln, die Signatur von Nachrichten oder zur Authentisierung von Kommunikationspartnern verwendet werden.

Definition 3.7 (Kryptographisches Protokoll) *Ein kryptographisches Protokoll regelt die Reihenfolge, in der zwei oder mehr Teilnehmer genau festgelegte Berechnungen durchführen oder Nachrichten austauschen, um ein definiertes Sicherheitsziel*[4] *zu erreichen.*

Es gibt eine Vielzahl von kryptographischen Protokollen für alle möglichen Anwendungen, z.B. Schlüsselaustausch, elektronisches Geld, Authentifizierung, Bit-Commitment, gegenseitige Vertragsunterzeichnung uvm. Eine gute Übersicht über die Vielfalt kryptographischer Protokolle liefert [Sch96]. Im Folgenden sollen nur Protokolle angesprochen werden, welche für die weiteren Ausführungen relevant sind.

3.4.1. Schlüsselaustausch

Wie wir gesehen haben, können zwei Partner, welche über einen gemeinsamen geheimen Schlüssel verfügen, durch Einsatz eines (symmetrischen) Verschlüsselungsverfahrens wie AES sicher miteinander kommunizieren. Das Problem der sicheren Kommunikation lässt sich also auf den sicheren Austausch eines vergleichsweise kurzen Schlüssels reduzieren. Obschon die asymmetrische Kryptographie eine Lösung anbietet, wird man in der Praxis vor allem aus Geschwindigkeitsgründen um den Einsatz symmetrischer Verfahren nicht herum kommen.

Dazu einigen sich die Kommunikationsteilnehmer in einer Sitzung auf einen temporären geheimen Schlüssen, welcher als *Sitzungsschlüssel* oder *Session Key* bezeichnet wird. Um einen solchen Schlüssel zwischen allen Partnern auszutauschen, kommen sogenannte *Schlüsselaustausch-Protokolle* zum Einsatz. Wünschenswerte Kriterien für ein solches Schlüsselaustausch-Protokoll sind [AG00]:

Geheimhaltung: Nur berechtigte Teilnehmer kennen am Ende des Protokolls den gemeinsamen Schlüssel. Kein Außenstehender kann ihn oder Teile davon erlangen.

Perfect Forward Secrecy: Wird ein ausgetauschter geheimer Schlüssel oder ein zur Schlüsselgenerierung verwendetes Passwort kompromittiert, so sind bei Protokollen mit dieser Eigenschaft alle früher ausgetauschten Schlüssel weiterhin geschützt.

Kooperative Schlüsselerzeugung: Alle Parteien, die am Schlüsselaustausch teilnehmen, tragen im gleichen Umfang zur Erzeugung des Schlüssels bei. Kein einzelner Teilnehmer (oder eine Gruppe von Teilnehmern) kann den Schlüssel bestimmen oder seine Stärke herabsetzen. Erzeugen mehrere Teilnehmer in dieser Form ge-

[4] z.B. Authentisierung

meinsam einen Schlüssel, spricht man von *key agreement* im Gegensatz zur *key distribution*, bei der ein Teilnehmer den Schlüssel erzeugt und dann verteilt.

Störungstoleranz: Kein Angreifer darf in der Lage sein, das Protokoll zu stören. Hierzu sollte es den Verlust (bzw. die absichtliche Löschung), das Einfügen, Verändern oder erneute Abspielen von Nachrichten ohne negative Beeinträchtigungen der Funktionalität verkraften.

Schlüsselverteilzentren

Einen einfachen Ansatz zur Verteilung von Sitzungsschlüssel stellen zentrale Schlüsselverteilzentren oder *Key Distribution Center (KDC)* dar. Hierbei wird vorausgesetzt, dass jeder Teilnehmer X über einen gemeinsamen Schlüssel K_{SX} mit dem Schlüsselverteilzentrum S verfügt. Möchte A nun mit B kommunizieren, so bittet er S, einen geheimen Sitzungsschlüssel K_{AB} zu generieren und diesen in verschlüsselter Form an beide Teilnehmer zu versenden.

Der Vorteil eines KDC ist, dass sich die Zahl der vorher über einen sicheren Kanal zu vereinbarenden Schlüssel drastisch reduziert. Ohne das KDC müsste man für jedes potentielle Kommunikationspaar (A, B) einen gemeinsamen, geheimen Schlüssel austauschen. In einem Netz mit n Teilnehmern sind dazu $\binom{n}{2} = \frac{n(n-1)}{2} = O(n^2)$ Schlüssel auszutauschen. Das Schlüsselverteilzentrum reduziert diesen Aufwand auf einen Schlüssel pro Knoten, also auf n. Diese n Schlüssel müssen aber nach wie vor abgesichert übertragen werden. Dies beheben die hybriden Kryptosysteme.

Hybride Kryptosysteme

Hier wird das Problem des *Schlüsselaustauschs* dadurch gelöst, dass ein Kommunikationspartner einen Sitzungsschlüssel zufällig generiert und diesen mit einem asymmetrischen Verschlüsselungsverfahren verschlüsselt zum Gegenüber schickt. Dieser entschlüsselt die Daten und kommt so in den Besitz eines temporären Schlüssels, den er für diese Sitzung verwendet.

Diffie-Hellmann Verfahren

Ein gängiges Verfahren zur Generierung von Sitzungsschlüsseln ist das Diffie-Hellmann Verfahren [DH76]. Der Protokollablauf ist wie folgt:

Definition 3.8 (Diffie-Hellmann Verfahren)

1. *Alice generiert eine große Primzahl n sowie ein z mit $0 < z < n$. n und z werden im Klartext übertragen oder im Vorfeld vereinbart.*
2. *Alice generiert eine geheime (große) Zufallszahl a und berechnet $A = z^a \bmod n$. Alice sendet A an Bob.*
3. *Bob generiert eine geheime (große) Zufallszahl b und berechnet $B = z^b \bmod n$. Bob sendet B an Alice.*
4. *Alice berechnet $K_a = B^a \bmod n$, Bob berechnet $K_b = A^b \bmod n$.*

Da $K_a = B^a \bmod n = (z^b \bmod n)^a \bmod n = (z^a \bmod n)^b \bmod n = A^b \bmod n = K_b$ gilt, verfügen Alice und Bob nun über einen identischen geheimen Schlüssel, den ein passiver Lauscher ohne Kenntnis der geheimen Zahlen a und b nicht berechnen kann. Hierzu müsste er $A = z^a \bmod n$ nach a auflösen (bzw. analog $B = z^b \bmod n$ nach b). Während dies mit reellen Zahlen einfach ist und der Berechnung von $a = \frac{\log A}{\log z}$ entspricht, ist der entsprechende *diskrete Logarithmus* in $\mathbb{Z}_n$ nur sehr schwer zu berechnen. Während im naiven Fall $O(2^n)$ Berechnungen notwendig sind, haben die schnellsten bekannten Verfahren mit $O(2^{\frac{n}{2}})$ und $O(2^{\sqrt{n}})$ immer noch eine exponentielle Laufzeit [Sho97].

Diffie-Hellmann ist, wie jedes Public-Key Verfahren, anfällig für Man-in-the-Middle Angriffe, weshalb hier immer eine verlässliche Authentisierung der beteiligten Parteien notwendig ist. Wie eine solche Authentisierung realisiert werden kann, wird später erläutert.

3.5. Digitale Signaturen

Oft will man eine Nachricht gar nicht verschlüsseln, sondern lediglich den Absender zweifelsfrei feststellen und spätere Veränderungen an einer Nachricht verhindern. Hierzu dient in der Regel die *digitale Signatur*, auch *digitale Unterschrift* genannt. Eine Unterschrift (egal ob digital oder analog) sollte laut [Sch96, Ert01] folgende Eigenschaften besitzen:

1. Sie ist *authentisch*, d.h. sie zeigt, dass der Unterzeichner das Dokument willentlich unterschrieben hat.
2. Sie ist *fälschungssicher*, d.h. sie beweist, dass der Unterzeichner und kein anderer das Dokument unterschrieben hat.
3. Sie ist *nicht wiederverwendbar*. Die Unterschrift kann nicht auf ein anderes Dokument kopiert werden.
4. Das unterzeichnete Dokument ist *unveränderbar*. Nach der Unterzeichnung kann es nicht verändert werden (ohne dass die Unterschrift ungültig wird).
5. Die Unterschrift ist *bindend*. Der Unterzeichner kann später nicht behaupten, dass er das Dokument nicht unterschrieben hat.

Wie man sieht, erfüllt die klassische Unterschrift keine dieser Aussagen vollständig, bzw. die Korrektheit lässt sich nur schwer feststellen. Im Gegensatz erfüllen digitale Signaturen die Anforderungen bis auf 1. in hohem Masse. 1. kann prinzipiell nicht garantiert werden, da der Unterzeichner z.B. durch Gewaltandrohung zur Unterschrift gezwungen werden kann. Auch könnte man ihm bei Kontrolle über dessen Rechner ein Dokument am Bildschirm anzeigen und ein anderes signieren lassen.

Eine digitale Signatur wird typischerweise mit Public-Key Verfahren erzeugt, indem man die Nachricht einmal im Klartext und einmal mit dem digitalen Schlüssel des Absenders A verschickt.

$$(M, E_{SK_A}(M))$$

Allerdings ist $E_{SK_A}(M)$ genauso groß wie M selbst, man verschickt die Nachricht also zweimal und verschwendet dadurch die Hälfte der zur Verfügung stehenden Bandbreite. Deshalb bildet man in der Praxis zunächst mit einer kryptographischen Hashfunktion h eine Prüfsumme über das Dokument und verschlüsselt diese mit dem geheimen Schlüssel des Absenders A. Die verschlüsselte Prüfsumme wird dann zusammen mit dem Text verschickt. Der Empfänger kann jetzt die Signatur verifizieren, indem er die verschlüsselte Prüfsumme mit dem öffentlichen Schlüssel des Absenders verifiziert und das Ergebnis mit einer selbst erstellten Prüfsumme vergleicht:

$$(M, E_{SK_A}(h(M)))$$

Neben RSA lassen sich prinzipiell ähnliche Berechnungen auch mit anderen Public-Key Algorithmen wie ElGamal durchführen. Nicht zum Verschlüsseln sondern ausschließlich zum Signieren eignet sich hingegen DSA. Im Jahr 1991 beauftragte das amerikanische National Institute of Standards and Technology (NIST) die National Security Agency (NSA) mit der Entwicklung eines neuen Standardverfahrens für digitale Signaturen. Das Ergebnis ist der *Digital Signature Algorithm (DSA)*, gelegentlich auch als *Digital Signature Standard (DSS)* bezeichnet [DSS94]. In Kombination mit dem Verschlüsselungsverfahren ElGamal wird er unter der Bezeichnung DSS/DH z.B. im Verschlüsselungsprogramm PGP [PGP03] ab Version 5 verwendet.

Theoretisch könnte man statt der asymmetrischen Verschlüsselung eine Signatur auch mittels eines symmetrischen Verfahrens realisieren. Hierzu müssen zwei Teilnehmer (Alice und Bob) einen gemeinsamen geheimen Schlüssel vereinbaren. Schickt Alice nun eine Nachricht an Bob, so bildet sie analog zu oben wieder die Prüfsumme und verschlüsselt diese mit dem geheimen Schlüssel. Bob kann dann die Prüfsumme kontrollieren und ist sich danach sicher, dass die Nachricht von Alice kommt. Dieses Verfahren erfüllt allerdings nicht die Anforderungen 2., 3. und 4. an digitale Signaturen, denn neben Alice kann ja auch Bob selbst das Dokument erzeugt haben und später behaupten, dass es von Alice käme. Auch kann er keinem Dritten beweisen, dass eine Nachricht von Alice unterzeichnet ist, ohne den geheimen Schlüssel offen zu legen. Eine Lösung bietet die Einbeziehung einer sogenannte *Trusted Third Party (TTP)* also eines vertrauenswürdigen Dritten, der über gemeinsame Kommunikationsschlüssel mit Alice und Bob verfügt. Er entschlüsselt die Nachricht, die Alice an ihn schickt, vermerkt die erfolgreiche Prüfung und schickt sie neu verschlüsselt an Bob weiter. Da ein solcher vertrauenswürdiger Dritter meist nicht vorhanden ist, kommen für Signaturen heute fast ausschließlich asymmetrische Verfahren zum Einsatz.

3.6. Authentisierung

3.6.1. Begriffe

Während im englischen Sprachgebrauch die Prüfung der Identität eines Kommunikationspartners mit dem Begriff *authentication* umfassend und eindeutig beschrieben ist, herrscht im Deutschen hier eine regelrechte Sprachverwirrung, da mal der Begriff *Authentifizierung* und mal *Authentisierung* verwendet wird. Die meisten Leute verwenden diese Begriffe völlig synonym, während für andere hier ein tatsächlicher Unter-

schied besteht. Daher stehen an dieser Stelle zunächst klare Definitionen der Begriffe (aus [BSI94]):

Definition 3.9 (Authentisierung) *Unter einer Authentisierung (engl. authentication) versteht man die Vorlage eines Nachweises eines Kommunikationspartners, in dem bestätigt wird, dass er tatsächlich derjenige ist, der er vorgibt zu sein.*

Definition 3.10 (Authentifizierung) *Unter einer Authentifizierung (engl. authentication) versteht man die Prüfung einer Authentisierung, d.h. die Überprüfung, dass ein Kommunikationspartner tatsächlich derjenige ist, der er vorgibt zu sein.*

Definition 3.11 (Authentizität) *Unter dem Begriff Authentizität (engl. authenticity) versteht man die Eigenschaft, die gewährleistet, dass der Kommunikationspartner tatsächlich derjenige ist, der er vorgibt zu sein bzw. dass die vorliegenden Informationen von der angegebenen Quelle erstellt wurden.*

Bezogen auf die digitale Signatur wäre also das Erstellen der Signatur durch den Absender eine Authentisierung, das Prüfen der Signatur durch den Empfänger die Authentifizierung. Bei einem Login-Vorgang an einem Rechner authentisiert sich der Benutzer, der Rechner hingegen authentifiziert ihn. Da jedoch die Rollen meist klar verteilt sind, schadet es in der Praxis meist wenig, wenn diese Begriffe synonym verwendet werden.

Die Authentizität des Absenders einer Nachricht wird in aller Regel durch eine digitale Signatur sichergestellt. Dies schützt aber nicht gegen sogenannte *Replay-Angriffe*, bei denen eine alte Nachricht mit gültiger Signatur nochmals wiederholt wird. Wollen sich also zwei Kommunikationspartner in einem Protokollablauf auch der Frische der Nachrichten versichern, so kommen typischerweise Zeitstempel oder Nonces zum Einsatz, die während eines kryptographischen Protokolls ausgetauscht werden.

Entsprechende Authentisierungsprotokolle für symmetrische und asymmetrische Verschlüsselung haben beispielsweise Needham und Schroeder schon 1978 vorgeschlagen [NS78][5].

3.6.2. Public-Key-Infrastrukturen

Ein zentraler Aspekt der Authentisierung ist die Überprüfung von öffentlichen Schlüsseln. Bei der Verteilung eines öffentlichen Schlüssels von Alice P_A entsteht ein Problem. Würde man den Schlüssel direkt veröffentlichen, z.B. über einen Webserver oder direkt verschicken, dann könnte ein Angreifer, nennen wir ihn Mallory, den Schlüssel gegen seinen eigenen öffentlichen Schlüssel PK_M austauschen. Wie in Abbildung 3.2 gezeigt, würde Bob dann unwissentlich seine Nachrichten mit PK_M verschlüsseln, diese könnte Mallory abfangen, mit SK_M entschlüsseln und mit dem richtigen PK_A verschlüsselt an Alice weiterleiten. Weder Alice noch Bob können diesen Angriff bemerken, der auch als *Man-in-the-Middle Angriff* bezeichnet wird. Daher muss sich Bob versichern, dass er wirklich den öffentlichen Schlüssel von Alice verwendet.

Solange Bob den öffentlichen Schlüssel über einen unsicheren Kanal erhält, kann er niemals sicher sein, dass dies tatsächlich der öffentliche Schlüssel seiner Kommunikationspartnerin Alice ist. Es könnte immer ein Man-in-the-Middle Angriff stattfinden,

[5] Es sei noch erwähnt, dass der originale Needham-Schroeder Algorithmus ein ernsthaftes Sicherheitsproblem aufwies, welches erst 1996 dank formaler Analyseverfahren entdeckt und behoben wurde [Low96].

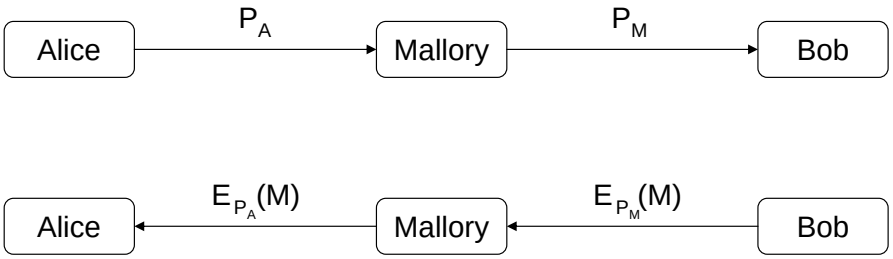

Abbildung 3.2.: *Man-in-the-Middle* Angriff

bei dem der Angreifer den Schlüssel ausgetauscht hat. Sicher kann er sich nur sein, wenn der Schlüssel über einen sicheren Kanal ausgetauscht wird (z.B. direkte Übergabe auf Diskette) oder er den Schlüssel beispielsweise durch ein Telefonat mit Alice verifiziert.

Für viele Kommunikationsvorgänge ist eine solche Verifikation nicht praktikabel, weil sich die Kommunikationspartner nicht persönlich kennen und kein sicherer Kanal zur Verfügung steht. Daher führt man auch hier einen Vermittler ein, dem alle Kommunikationspartner vertrauen, ein sogenanntes *Trustcenter*. Eine Zertifizierungsstelle (engl. *Certification Authority (CA)*) authentifiziert den Zusammenhang zwischen einem Benutzer und dessen öffentlichem Schlüssel (z.B. durch persönlichen Kontakt und Prüfung des Personalausweises) und bestätigt dies durch eine digitale Signatur des öffentlichen Schlüssels, ein sogenanntes *Zertifikat*. Ein Zertifikat besteht aus einer Signatur über eine Identität ID und einen zugehörigen öffentlichen Schlüssel PK.

$$Cert = (ID, PK, E_{SK_{CA}}(ID, PK))$$

Um Platz zu sparen, wird hier in der Regel wieder mit einer Hashfunktion h gearbeitet.

$$Cert = (ID, PK, E_{SK_{CA}}(h(ID, PK)))$$

Zur Verifikation eines Zertifikats entschlüsselt der Empfänger das Zertifikat mit dem öffentlichen Schlüssel der CA PK_{CA} und prüft, ob die vorgegebene Identität der im Zertifikat enthaltenen entspricht. Natürlich kann ein Schlüsselpaar auf diesem Weg nicht nur einer Person zugeordnet werden, sondern auch einer Firma oder einem Rechner, beispielsweise einem Knoten in einem Ad hoc Netzwerk.

Manchmal ordnet man die Zertifizierungsstellen auch hierarchisch an. Dabei signiert eine sogenannte *Wurzel-CA (Root-CA)* zunächst die öffentlichen Schlüssel der untergeordneten CAs usw., so dass sich ein CA Baum ergibt. Auf der untersten Ebene zertifizieren die CAs dann die Schlüssel der Benutzer. Ein komplettes System zur Verwaltung und Verifikation von Schlüssel nennt man *Public-Key-Infrastruktur (PKI)*.

Definition 3.12 *Eine* Public Key Infrastruktur *besteht aus der Hardware, Software, den Personen, Richtlinien und Verfahren, die zur Erzeugung, Verwaltung, Speicherung, Verteilung und zum Rückruf von Public-Key Zertifikaten auf Basis von asymmetrischer Kryptographie benötigt werden [AT02, Abschnitt 1.2].*

Das grundlegende Problem bleibt aber bei all diesen Maßnahmen erhalten. Letztendlich löst man das Problem, ob man einem bestimmten öffentlichen Schlüssel vertrauen kann, nicht wirklich. Jetzt muss man nämlich die Korrektheit des öffentlichen Schlüssels der (Wurzel-)CA sicherstellen. Im Rahmen des Vertrauens in die korrekte und sichere Arbeitsweise einer solchen CA ist man dann in der Lage, die Schlüssel aller Benutzer zu verifizieren, welche diese CA zertifiziert hat. Somit reduziert sich der Aufwand bei der Kommunikation mit mehreren Benutzern deutlich, vor allem weil man die Zertifikate wichtiger CAs in der Regel fest in die entsprechende Software wie Web-Browser oder Mail-Clients einbaut und somit die Manipulation der CA Schlüssel erschwert wird.

Trotzdem bleiben natürlich Angriffsmöglichkeiten bestehen. Unter Umständen kommt Mallory in den Besitz des geheimen Schlüssels von Alice. Das Zertifikat bestätigt ja lediglich, dass dieser Schlüssel Alice gehört, es bleibt also weiterhin gültig. Anschließend kann sich Mallory im Netz für Alice ausgeben, ohne dass dies andere Teilnehmer bemerken können.

Um die Auswirkungen eines solchen Schlüsselverlusts zu begrenzen, haben Zertifikate (und/oder Schlüssel) meist eine *beschränkte Gültigkeit*. Nach Ablauf dieser Gültigkeit muss das Zertifikat bei der CA erneuert (oder ein neuer Schlüssel generiert und zertifiziert) werden. Nach Ablauf des Zertifikats wird der Schlüssel für Mallory also wertlos, da er kaum in der Lage sein wird, die CA zur Ausgabe eines neuen Zertifikats auf den Namen Alice zu bewegen. Zusätzlich unterstützen manche PKI Systeme noch sogenannte *Certificate Revocation Lists (CRLs)* [NN98], also Listen von zurückgerufenen Zertifikaten. Hier gibt die Zertifizierungsstelle in regelmäßigen Abständen Listen an alle Knoten aus, welche ungültige Zertifikate enthalten, deren Gültigkeitszeitraum noch nicht abgelaufen ist. Verliert ein Benutzer seinen Schlüssel (und bemerkt diesen Verlust), kann er sich an die CA wenden und um Sperrung bitten. Spätestens mit der nächsten CRL wird der Schlüssel dann für Mallory wertlos.

3.6.3. Schwellwert-Kryptographie

Wird eine CA erfolgreich angegriffen und deren Signatur-Schlüssel kompromittiert, so hat dies verheerende Auswirkungen. Der Angreifer kann nun beliebig Zertifikate erstellen und zurückziehen. Um die Verwundbarkeit der CA zu reduzieren, kann man die sogenannte *Schwellwert-Kryptographie* (engl. *threshold cryptography* oder auch *secret sharing*) einsetzen.

Mit einem (k, n)-Schwellwert-Verfahren ist es möglich, ein Geheimnis so auf n Teilnehmer zu verteilen, dass mindestens k Teilnehmer kooperieren müssen, um das Geheimnis rekonstruieren zu können. Versuchen sich weniger als k Teilnehmer an einer solchen Rekonstruktion, so erlangen sie hierbei keinerlei Information über das ursprüngliche Geheimnis. Shamir [Sha79] und Blakley [Bla79] entwickelten dieses Verfahren etwa zeitgleich. Zunächst wurden hierbei lediglich Geheimnisse verteilt (*secret sharing*), was den Nachteil hat, dass nach einer Rekonstruktion jeder Teilnehmer das Geheimnis kennt. Deshalb wurden später Verfahren entwickelt, die nicht ein Geheimnis sondern einen ganzen kryptographischen Algorithmus verteilt ausführen (*function sharing*) [GJKR96].

In diesem Kontext sind insbesondere Verfahren wie beispielsweise das in [DF89, FD92] interessant, bei denen ein geheimer Schlüssel so verteilt wird, dass k aus n Teilnehmer

eine gültige Signatur erzeugen können, ohne dass es nötig ist, den geheimen Schlüssel vorher vollständig zu rekonstruieren. Durch geeignete Wahl von k und n lässt sich die Sicherheit des Verfahrens einstellen. Je höher k ist, desto schwieriger wird es für einen Angreifer, genügend Knoten zu kontrollieren. Gleichzeitig wird aber auch die reguläre Kooperation erschwert.

Kann garantiert werden, dass ein Angreifer niemals k Knoten gleichzeitig kontrolliert, so ist die Kompromittierung von maximal $k-1$ Knoten unkritisch. Allerdings könnte ein Angreifer im Laufe einer längeren Zeit möglicherweise doch beliebig viele Knoten kontrollieren. Deshalb bietet es sich an, die Teilgeheimnisse in regelmäßigen Zeitabständen T vorsorglich (proaktiv) zu erneuern [HJKY95]. Dann muss ein Angreifer in *einem* Zeitintervall T mindestens k Knoten unter seine Kontrolle bringen, um die Sicherheit des Systems zu beeinträchtigen. Das genannte Verfahren bietet auch die Möglichkeit, die Parameter k und n am Ende eines Intervalls neu zu setzen. So könnte bei einem erkannten Angriff möglicherweise der Parameter k für einige Zeit erhöht werden.

Ein weiteres Problem bei der Schwellwert-Kryptographie besteht darin, dass jeder der beteiligten Knoten das Verfahren boykottieren kann, indem er seinen Teil des Geheimnisses verfälscht oder seine Berechnungen beim *function sharing* falsch durchführt. Beim *verifyable secret sharing* lassen sich solche Angriffe verhindern.

3.6.4. Identitätsbasierte Kryptographie

Das Problem der Zuordnung eines öffentlichen Schlüssels zu einer Identität ließe sich auf sehr einfache Weise lösen, wenn Identität und öffentlicher Schlüssel untrennbar miteinander verknüpft wären, eine CA wäre dann nicht mehr notwendig. Aus diesem Gedankengang entstanden die 1984 von Adi Shamir erstmals vorgeschlagenen Systeme mit identitätsbasierter Kryptographie, die auch *Non-Interactive Key-Sharing Systeme (NIKS)* genannt werden [Sha84b]. Hierbei wird aus der Identität eines Benutzers direkt dessen öffentlicher Schlüssel abgeleitet. Eine separate Verknüpfung ist unnötig, auch die explizite Verteilung von Schlüsseln und Zertifikaten entfällt. Bei Kenntnis einer Emailadresse kann man sofort eine verschlüsselte Email an die betreffende Person schicken.

Ein Nachteil dieses Vorgehens zeigt sich, wenn ein Benutzer seinen geheimen Schlüssel verliert. In diesem Fall ist er nämlich gezwungen, sich eine neue Identität und daraus ein neues Schlüsselpaar zu generieren. Sämtliche Kommunikationspartner müssen dann über diesen Identitätswechsel informiert werden, damit sie keine Nachrichten mehr für die alte Identität (und somit für den Angreifer lesbar) verschlüsseln.

Generell sind sichere und praktikable identitätsbasierte Kryptosysteme nur schwer zu entwickeln. Einige Informationen hierzu finden sich in [Sch96]. Ein neueres System ist beispielsweise [BF01]. Der umgekehrte Fall der kryptobasierten Identitäten wird in Kapitel 8 vorgestellt.

3.7. Automatische Protokollverifikation mit BAN Logik

Bisher funktioniert der Entwurf von kryptographischen Protokollen eher nach dem „Versuch und Irrtum“ Prinzip. Erfahrene Sicherheitsspezialisten überlegen sich ein Ver-

fahren, publizieren dieses und andere Forscher versuchen, Schwachstellen zu finden. Leider kann es manchmal lange dauern, bis eine Schwachstelle entdeckt wird; auch in scheinbar sicheren Protokollen wurden nach Jahren noch Fehler gefunden [DS81, BAN90a, Sim85]. In der Zwischenzeit ist das Protokoll unter Umständen in einer Vielzahl von Systemen implementiert und entsprechend schwer ist es, den Fehler zu beheben oder auf ein anderes Protokoll umzustellen.

Eine ähnliches Problem ist die Korrektheit von Software. Hier gibt es seit Langem Verfahren wie das Hoare-Kalkül [Hoa69], mit denen man die Korrektheit eines Programms bezüglich einer formalen Spezifikation beweisen kann. Allerdings stoßen auch diese Verfahren mit zunehmender Komplexität der Programme schnell an ihre Grenzen.

Noch problematischer ist der Beweis der Sicherheit von kryptographischen Protokollen. Hier will man nämlich nicht nur beweisen, dass ein Protokoll innerhalb einer Spezifikation bei korrekten Eingaben korrekte Ausgaben liefert. Vielmehr soll auch bewiesen werden, dass das Protokoll unter allen denkbaren Umständen – insbesondere also bei allen möglichen Angriffen – kein unerwünschtes Verhalten zeigt, also der Angreifer beispielsweise keine Kenntnis über kryptographische Schlüssel erlangt.

Im Fall des bereits erwähnten Needham-Schroeder-Protokolls [NS78] stellt sich z.B. die Frage, ob die Teilnehmer nach Ablauf des Protokolls wirklich von der Authentizität des Kommunikationspartners überzeugt sein können oder ob es subtile Täuschungsmöglichkeiten gibt? Oder kann ein Angreifer aus dem Ablauf des Protokolls vielleicht Informationen gewinnen, die er für zukünftige Maskeradeangriffe nutzen kann?

Michael Burrows, Martín Abadi und Roger Needham publizierten 1990 die nach den Anfangsbuchstaben ihrer Nachnamen benannte BAN-Logik [BAN90a]. Sie ermöglicht die schrittweise formale Analyse von Authentifikationsprotokollen inklusive aller Vorbedingungen. Damit wird auch ein Vergleich verschiedener Protokolle hinsichtlich ihrer jeweiligen Annahmen möglich und überflüssige Protokollschritte können erkannt werden. Da die BAN Notation in Kapitel 12 verwendet wird, um die Korrektheit des SDSR Protokolls zu untersuchen, folgt hier eine kurze Einführung. Eine detailliertere Untersuchung verschiedener formaler Methoden zur Verifikation von kryptographischen Protokollen ist beispielsweise [KMM94] zu entnehmen.

3.7.1. Notation

Die BAN-Logik unterscheidet Protokollteilnehmer (*principals*), Schlüssel (*encryption keys*) und Formeln bzw. Aussagen (*statements*). In der folgenden Notationsübersicht bezeichnen P, Q und R beliebige Teilnehmer, K steht für Schlüssel und X und Y für Aussagen bzw. Formeln.

$P \models X$: **P glaubt X**
: P verhält sich so, als ob X wahr ist.

$P \lhd X$: **P sieht X**
: P hat die Nachricht X (von einem nicht näher bezeichneten Absender) erhalten und kann X (evtl. nach Entschlüsselung) lesen.

$P \mid\sim X$: **P sagte X**
: Hierbei wird keine Aussage gemacht, ob X im aktuellen Protokolllauf (Gegen-

wart) oder in einem früheren Durchlauf (Vergangenheit) gesendet wurde; zum ursprünglichen Sendezeitpunkt galt jedoch auf jeden Fall: $P \models X$.

$P \models\!\!\Rightarrow X$: **P hat Autorität über X**
P ist vertrauenswürdig in Bezug auf X; P kann z. B. ein Server mit spezieller Funktionalität X sein (Schlüsselerzeugung, Signatur etc.).

$\sharp(X)$: **X ist frisch**
X wurde noch in keinem früheren Protokolllauf verwendet; X wird auch als *Nonce* bezeichnet.

$P \overset{K}{\leftrightarrow} Q$: **P und Q besitzen einen geheimen Schlüssel K**
Niemand sonst kennt K oder kann K erlangen, außer P oder Q vertrauen ihm.

$\overset{K}{\mapsto} P$: **K ist öffentlicher Schlüssel von P**
Den zugehörigen geheimen Schlüssel K^{-1} kennen nur P und evtl. seine Vertrauten.

$P \overset{X}{=\!=} Q$: **P und Q besitzen ein gemeinsames Geheimnis X**
Evtl. kennen noch weitere vertrauenswürdige Instanzen X, nur P und Q dürfen jedoch X zum gegenseitigen Identitätsnachweis verwenden.

$\{X\}_K$: **X ist mit K verschlüsselt**
Jeweils abhängig vom Typ von K wird symmetrisch oder asymmetrisch verschlüsselt. Kein Knoten interpretiert von ihm selbst verschlüsselte Nachrichten.

$\langle X \rangle_Y$: **Y beweist die Identität des Absenders von X**
Y kann z. B. ein Passwort sein, mit welchem durch eine *keyed hash* Funktion eine Signatur von X erzeugt wird.

3.7.2. Schlussregeln

Anhand der nun folgenden Schlussregeln können aus einer oder mehreren logischen Aussagen neue, gültige Aussagen gefolgert (abgeleitet) werden. Die Notation $\frac{X}{Y}$ ist dabei so zu verstehen, dass, falls der Teil X oberhalb des Strichs gilt, daraus der Teil Y unter dem Strich abgeleitet werden kann. Ein Komma bezeichnet die UND-Verknüpfung (Konjunktion) der beiden Aussagen.

Die sogenannten *message meaning*-Regeln (3.1) bestimmen, wie auf den Absender einer Nachricht geschlossen werden kann. Die Regel links beschreibt dabei z. B. folgenden Sachverhalt: Wenn P glaubt, dass K sein gemeinsamer geheimer Schlüssel mit Q ist, und P die Nachricht X empfängt, die mit K verschlüsselt wurde, dann ist P davon überzeugt, dass X *ursprünglich* von Q abgeschickt wurde[6]. Die anderen beiden Regeln gelten analog für asymmetrisch verschlüsselte bzw. mit einem gemeinsamen Geheimnis (Passwort) gesicherte Nachrichten.

$$\frac{P \models Q \overset{K}{\leftrightarrow} P, \quad P \lhd \{X\}_K}{P \models (Q \mid\sim X)} \qquad \frac{P \models \overset{K}{\mapsto} Q, \quad P \lhd \{X\}_{K^{-1}}}{P \models (Q \mid\sim X)} \qquad \frac{P \models Q \overset{Y}{=\!=} P, \quad P \lhd \langle X \rangle_Y}{P \models (Q \mid\sim X)} \tag{3.1}$$

[6] $\{X\}_K$ impliziert, dass die Nachricht wirklich von außen kam; kein Knoten interpretiert von ihm selbst verschlüsselte Nachrichten.

Die *nonce verification*-Regel (3.2) drückt aus, wie der Empfänger durch eine Nonce Gewissheit über die Aktualität einer Nachricht (Aussage) erlangen kann. Dies ist die einzige Regel, die aus $\mid\sim$ („sagte“) ein $\mid\equiv$ („glaubt“) macht.

$$\frac{P \mid\equiv \sharp(X), \quad P \mid\equiv (Q \mid\sim X)}{P \mid\equiv (Q \mid\equiv X)} \tag{3.2}$$

Die *jurisdiction*-Regel (3.3) beruht auf der Semantik von $\mid\Rightarrow$:

$$\frac{P \mid\equiv (Q \mid\Rightarrow X), \quad P \mid\equiv (Q \mid\equiv X)}{P \mid\equiv X} \tag{3.3}$$

(3.4) und (3.5) sind einige leicht einzusehende Regeln über die Interpretation zusammengesetzter Nachrichten (Aussagen):

$$\frac{P \mid\equiv X, \quad P \mid\equiv Y}{P \mid\equiv (X,Y)} \qquad \frac{P \mid\equiv (X,Y)}{P \mid\equiv X} \qquad \frac{P \mid\equiv (Q \mid\equiv (X,Y))}{P \mid\equiv (Q \mid\equiv X)} \tag{3.4}$$

$$\frac{P \mid\equiv (Q \mid\sim (X,Y))}{P \mid\equiv (Q \mid\sim X)} \qquad \frac{P \lhd (X,Y)}{P \lhd X} \qquad \frac{P \mid\equiv \sharp(X)}{P \mid\equiv \sharp(X,Y)} \tag{3.5}$$

(3.6) und (3.7) beschreiben die Entschlüsselung von Nachrichten:

$$\frac{P \lhd \langle X \rangle_Y}{P \lhd X} \qquad \frac{P \mid\equiv (Q \overset{K}{\leftrightarrow} P), \quad P \lhd \{X\}_K}{P \lhd X} \tag{3.6}$$

$$\frac{P \mid\equiv (\overset{K}{\mapsto} P), \quad P \lhd \{X\}_K}{P \lhd X} \qquad \frac{P \mid\equiv (\overset{K}{\mapsto} Q), \quad P \lhd \{X\}_{K^{-1}}}{P \lhd X} \tag{3.7}$$

(3.8) und (3.9) beschreiben schließlich die Kommutativität verschiedener Operatoren:

$$\frac{P \mid\equiv (R \overset{K}{\leftrightarrow} R')}{P \mid\equiv (R' \overset{K}{\leftrightarrow} R)} \qquad \frac{P \mid\equiv (Q \mid\equiv (R \overset{K}{\leftrightarrow} R'))}{P \mid\equiv (Q \mid\equiv R' \overset{K}{\leftrightarrow} R))} \tag{3.8}$$

$$\frac{P \mid\equiv (R \overset{X}{\rightleftharpoons} R')}{P \mid\equiv (R' \overset{X}{\rightleftharpoons} R)} \qquad \frac{P \mid\equiv (Q \mid\equiv (R \overset{X}{\rightleftharpoons} R'))}{P \mid\equiv (Q \mid\equiv (R' \overset{X}{\rightleftharpoons} R))} \tag{3.9}$$

Wie in [Nes90, BAN90b] dargelegt, sind die von Burrows, Abadi und Needham aufgestellten Schlussregeln keineswegs vollständig. Daher sind für den späteren Beweis von SDSR noch einige erweiterte Schlussregeln notwendig, deren Korrektheit intuitiv einleuchtet:

$$\frac{P \mid\Rightarrow X, \quad \sharp X}{P \mid\equiv \sharp X} \tag{3.10}$$

Hat ein Knoten die Kontrolle über X und ist X frisch (weil es z.B. Knoten in jedem Protokolldurchlauf neu generiert wird), so glaubt er auch an die Frische von X.

$$\frac{P \mid\equiv \ Q \mid\equiv h(X)}{P \mid\equiv \ Q \mid\equiv X} \tag{3.11}$$

Wenn P glaubt, dass Q $h(X)$ glaubt[7], dann ist P auch davon überzeugt, dass Q an X glaubt.

[7] und h eine kryptographische Hashfunktion ist

3.7.3. Protokollanalyse

Um ein konkretes Protokoll mittels BAN-Logik zu analysieren, sind vier Schritte notwendig, die eventuell wiederholt werden müssen, wenn neue Bedingungen gefunden werden oder die formale Protokollbeschreibung verfeinert wird.

1. **Idealisierung des Protokolls:** Das ursprüngliche Protokoll wird in idealisierter Form in der vorgestellten Notation beschrieben. Die Formeln enthalten dabei nicht unbedingt die gleichen Bestandteile, die auch im realen Protokoll übermittelt werden, sondern nur die jeweils für den Authentifizierungsprozess relevanten Informationen.

2. **Anfangsbedingungen ergänzen:** Bedingungen, die den Anfangszustand des Protokolls beschreiben, werden hinzugefügt.

3. **Nachbedingungen finden:** Jedem Protokollschritt wird eine Formel zugeordnet, die den Zustand des Systems nach jedem Schritt beschreibt.

4. **Schlussregeln anwenden:** Durch Anwendung der Schlussregeln auf die gefundenen Behauptungen und Bedingungen wird schrittweise ermittelt, was die Protokollteilnehmer glauben (im Sinne von $\models$).

In [BAN90a] wird diese Vorgehensweise exemplarisch auf das Needham-Schroeder-Protokoll [NS78] angewendet.

3.7.4. Kritik und Alternativen

Mit Hilfe der BAN-Logik wurden in zahlreichen Protokollen (u. a. Needham-Schroeder, Kerberos [NT94], X.509 [CCI87]) Fehler und Redundanzen gefunden; in vielen Arbeiten wird sie zudem zur Untermauerung der aufgestellten Thesen herangezogen [Sch96]. Trotzdem stellt die BAN-Logik kein Allheilmittel dar, wie im folgenden kurz erläutert werden soll.

Die Arbeit von Burrows et al. enthält keine vollständige formale Definition von Syntax oder Semantik der BAN-Logik-Konstrukte. Somit ist auch kein Beweis für die Korrektheit und Vollständigkeit der Schlussregeln möglich. Es könnte also Aussagen geben, die zwar formal korrekt abgeleitet werden können, die intuitiv aber falsch sind. Umgekehrt können intuitiv richtige Folgerungen eventuell nicht anhand der Schlussregeln abgeleitet werden [Nes90, BAN90b]. Aus diesem Grund mussten die Schlussregeln auch in dieser Arbeit um die Formeln 3.10 und 3.11 erweitert werden.

Die BAN-Logik geht von einigen Grundannahmen aus, die nicht weiter analysiert werden, so z. B. der Sicherheit der verwendeten kryptographischen Algorithmen, der Ehrlichkeit der Teilnehmer oder der Zuverlässigkeit der vertrauenswürdigen Instanzen. Zur Anwendung der BAN-Logik muss das Protokoll zudem in einer idealisierten Form vorliegen. Diese kann jedoch nicht einfach durch Umformung anhand fester Regeln aus der üblichen Protokollspezifikationen (A sendet Nachricht M an B) gewonnen werden. Es ist vielmehr ein genaues Verständnis des Protokolls und des „tieferen Sinns“ jedes einzelnen Schrittes nötig:

> „However, the idealized form of each message cannot be determined by looking merely at a single protocol step by itself. Only knowledge of the entire protocol can determine the essential logical contents of the message.“ [BAN90a, Abschnitt „Idealized protocols“]

Dadurch stellt sich die Frage, ob eine gefundene Idealisierung wirklich dem zugrunde liegenden Protokoll entspricht, oder ob bei der Umsetzung Fehler gemacht wurden. Ähnliche Schwierigkeiten gibt es auch beim Finden der Anfangsbedingungen. Über die Korrektheit einer *konkreten Implementierung* eines Protokolls kann die BAN-Logik natürlich erst recht keine Aussage treffen.

Schließlich kann die BAN-Logik keinen Beweis für die Sicherheit eines Protokolls liefern; sie kümmert sich ausschließlich um die Frage, ob die übertragenen Informationen ausreichen, um die Teilnehmer zweifelsfrei zu authentifizieren und ggf. einen gemeinsamen Schlüssel zu vereinbaren. Falls im Rahmen des Protokolls sensible Informationen ungeschützt übertragen werden und somit einem Angreifer in die Hände fallen können, fällt dies bei der Analyse leider nicht auf [Nes90].

Es existieren etliche Erweiterungen der BAN-Logik, u. a. die GNY-Logik [GNY90] und die Logik von Abadi und Tuttle [AT91], die mit dem Ziel entworfen wurden, die Schwächen der BAN-Logik zumindest teilweise zu beheben. Gänzlich andere Ansätze werden beim *Protocol Analyzer* des U. S. Naval Research Laboratory (NRL) [Mea96] und beim *Interrogator*-System [MCF87] verfolgt. Diese Systeme modellieren und analysieren das Protokoll als algebraisches System bzw. durch ein Expertensystem.

Alle diese Systeme besitzen eine Vielzahl von inhärenten Fehlerquellen. Bereits bei der Abstraktion und Modellierung des Protokolls können Probleme entstehen, so dass das Modell nicht mehr dem tatsächlichen Protokoll entspricht. Manche der Systeme verwenden Zustandsautomaten, um die Sicherheit eines Systems zu analysieren. Hierbei erreicht der zu durchsuchende Zustandsraum jedoch schnell eine Größenordnung, bei welcher der Benutzer den Suchraum beschränken muss, um in endlicher Zeit zu einem Ergebnis zu kommen. Wählt der Benutzer diese Einschränkungen falsch, werden mögliche Angriffe und Schwachstellen übersehen.

Insgesamt sind wir heute von einem automatisierten Beweis der Sicherheit eines kryptographischen Protokolls noch weit entfernt. Trotz aller Schwächen leisten die bestehenden Ansätze jedoch wertvolle Dienste, wenn es darum geht, Protokolle zu analysieren und das Vertrauen in deren Zuverlässigkeit zu erhöhen. Deshalb wird die BAN-Logik in Kapitel 12 eingesetzt, um das Secure-DSR Protokoll auf Korrektheit zu untersuchen.

3.8. Fazit

Nach dieser Übersicht über wesentliche kryptographische Mechanismen werden in den nächsten Kapiteln noch weitere Grundlagen zu mobiler Datenkommunikation und Mobilen Ad hoc Netzen beschrieben. Aufbauend auf diesen Grundlagen kann dann in Kapitel 6 zunächst eine Analyse der Sicherheitsaspekte von Mobilen Ad hoc Netzen erfolgen. Bei der anschließenden Vorstellung der Sicherheitsarchitektur für Mobile Ad hoc Netze wird dann intensiv von den hier vorgestellten Algorithmen wie RSA oder Diffie-Hellmann Gebrauch gemacht. Auch bei der Analyse (siehe Kapitel 12) leisten die hier vorgestellten Verfahren wertvolle Dienste.

4. Mobile Datenkommunikation

In den sechziger Jahren legten Forscher die Grundlagen zu den paketorientierten, leitungsgebundenen Datennetzen, welche auch heute noch die Landschaft der Computernetze bestimmen. Beispielsweise entwickelte Bob Metcalf bei Xerox PARC das Ethernet und mit dem ARPANET entstand der Vorläufer des heutigen Internet.

Doch bereits zu diesem frühen Zeitpunkt wurden auch drahtlose Alternativen entwickelt, um die Daten mittels elektromagnetischer Strahlung anstatt durch Leitungen zu übermitteln.

Zu Beginn der 70er Jahre entwickelte Norman Abramson auf Hawaii das sogenannte *ALOHA Netzwerk* [MW77, Abr70]. Die dortige Universität hatte das Problem, dass sie Standorte auf unterschiedlichen Inseln miteinander vernetzen und an das ARPANET anschließen wollte. Doch obwohl die Inseln teilweise nur einige Dutzend Kilometer auseinander liegen, trennen sie mehrere Tausend Meter tiefe Unterseegräben. Da die Verlegung von Unterwasserkabeln sehr aufwändig ist, entstand also die Idee, die damals aufkommenden paketorientierten Netze über Funk statt über Kupferkabel zu betreiben. Dabei waren verschiedene technische Probleme zu lösen, vor allem das des koordinierten Zugriffs auf das Funkmedium. Das auf Hawaii entwickelte Zugriffsverfahren trägt den Namen *ALOHA* und ist ein einfacher Vorläufer des *Carrier Sense Multiple Access (CSMA)* Verfahrens, welches in seiner Weiterentwicklung als *CSMA with Collision Detection (CSMA/CD)* beim herkömmlichen *Ethernet* zum Einsatz kommt.

Weitere Entwicklungen in dieser Zeit waren die paketorientierte Satellitenkommunikation im *SATNET* Projekt [JBH78], welches 64 kbps Kanäle auf Intelsat-Satelliten benutzte, sowie ein erster Ansatz zu spontaner Ad hoc Kommunikation, das *Packet Radio Network (PRNET)* [K+78]. In diesem Projekt wurden mobile und stationäre Einheiten in der San Francisco Bay Area über Funkkanäle mit 400 bzw. 100 kbps dynamisch vernetzt, wobei jede Einheit auch als Relay für andere Einheiten dienen konnte. Dies war ein signifikanter Unterschied zum ALOHA Netzwerk, bei dem sich alle Sender in gegenseitiger Reichweite befinden mussten. Mit dem PRNET war die Idee der *Mobilen Multi-Hop Ad hoc Netzwerke (MANETs)* geboren.

Einige Jahre später (1976) kam die Idee auf, das *ARPANET*, das *SATNET* und das *PRNET* zu verbinden [L+97]. Ziel war es, eine Datenkommunikation zwischen einem fahrenden LKW in San Francisco und einem stationären Rechner in England zu etablieren. Dazu war es notwendig, dass die drei eigentlich heterogenen Netzwerke ein einheitliches Paketformat und vor allem einheitliche Adressen verwendeten. Folglich wurde ein neues Protokoll entwickelt, welches es erlaubte, die Unterschiede der drei Netzwerkarten zu verstecken: das *Transmission Control Protocol* oder kurz *TCP*. Mittels TCP konnten Datenpakete zwischen den Teilnetzen ausgetauscht werden, man spricht hier vom sog. *Routing*. Dies war die Geburtsstunde des *Internetworking* und somit des heutigen *Internet*.

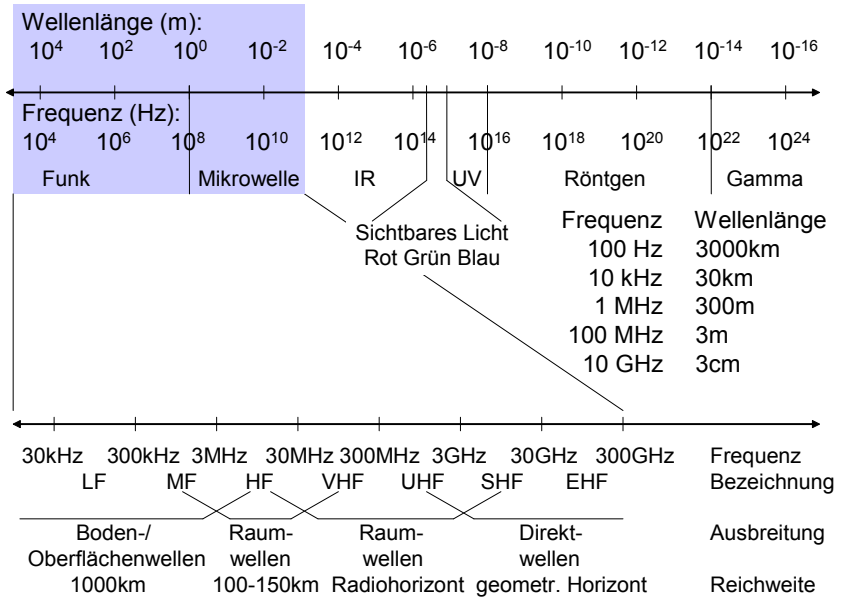

Abbildung 4.1.: Frequenzspektrum

Später wurde das TCP Protokoll in zwei separate Protokolle namens TCP und *IP (Internet Protocol)* aufgeteilt, wobei das IP für die Weiterleitung der Datenpakete und das TCP für die Ende-zu-Ende Sicherung einer virtuellen Verbindung zuständig ist.

Wie man sieht, waren zu Beginn die Forschungen über paketorientierte Kommunikation in leitungsgebundenen und drahtlosen Netzwerken eng miteinander verknüpft. Dann erlebten Ethernet und Glasfasernetze einen wahren Boom und erreichten ungeahnte Datenraten. Erst in den letzten Jahren hat sich das Interesse an den drahtlosen Netzen wieder verstärkt und heute sind Wireless LANs, GPRS, Richtfunk- oder Satellitenstrecken aus der Kommunikationsinfrastruktur nicht mehr wegzudenken.

4.1. Grundlagen der drahtlosen Datenübertragung

4.1.1. Elektromagnetische Wellen

Die Grundlage jeder drahtloser Datenübertragung ist eine elektromagnetische Welle. Auf diese wird ein Datensignal aufmoduliert. Dabei steht prinzipiell das gesamte elektromagnetische Spektrum zur Verfügung, allerdings unterscheiden sich die unterschiedlichen Frequenzen zum Beispiel bezüglich Reichweite, nutzbarer Bandbreite oder der Störungsempfindlichkeit durch Hindernisse.

Bild 4.1 zeigt einen Überblick über das gesamte Frequenzspektrum [Tan96, Sik01]. Nyquist [Nyq24] und Shannon [Sha48] haben schon sehr früh gezeigt, dass die maximal erreichbare Datenrate eines Kommunikationskanals prinzipiell beschränkt ist. Dies hängt unter anderem von der verwendeten Sendefrequenz und dem sogenannten *Signal-Rauschabstand* ab. Dieser Zusammenhang lässt sich nach Shannon mathematisch folgendermaßen ausdrücken:

$$\max\left(\frac{Zeichen}{s}\right) = H\ \log_2\left(1+\frac{S}{N}\right)$$

Dabei ist H die Frequenz (in Hz); $\frac{S}{N}$ bezeichnet den Signal-Rauschabstand (in dB). Wie man sieht, eignen sich also hohe Frequenzen besser für die schnelle Übertragung großer Datenmengen. Dabei ist der von Shannon angegebene Wert als Obergrenze zu verstehen, der in der Praxis kaum erreicht werden wird. Allerdings redet Shannon hier von Zeichen pro Sekunde, dabei kann ein Zeichen durchaus einen Informationsgehalt von mehreren Bit haben.

Obwohl es aus Sicht der maximalen Übertragungsrate wünschenswert ist, möglichst hohe Frequenzen zur Übertragung zu nutzen, unterliegen diese beim Transport abhängig vom Medium, in dem sie sich ausbreiten, höheren Dämpfungen, was wiederum die Reichweite einschränkt. Beim Entwurf von Funk-Kommunikationssystemen ist also ein Kompromiss gefragt.

Das Frequenzspektrum ist laut *International Telecommunication Union (ITU)* [ITU] in verschiedene *Frequenzbänder* eingeteilt. Am unteren Ende im Bereich von 1 kHz bis 10 kHz liegt das *Very Low Frequency (VLF)* Band. Dieses bietet für Datenanwendungen nur eine minimale Übertragungskapazität und wird daher ausschließlich bei Spezialanwendungen genutzt. So gibt es ein System zur Telegraphie zwischen U-Booten, was diese Frequenzen nutzt, weil höhere Frequenzen im Wasser zu stark gedämpft werden.

Der Bereich zwischen 30 kHz und 300 kHz wird als *Low Frequency (LF)* Band oder Langwelle bezeichnet. Eine Anwendung dieser Frequenz ist das *LORAN-C (Long Range Navigation)* System, welches bei 100 kHz Peilsignale aussendet, die beispielsweise von Schiffen zur Positionsbestimmung genutzt werden. Allerdings wird dieses System immer mehr vom Satellitennavigationssystem *GPS (Global Positioning System)* verdrängt.

Der Bereich zwischen 300 kHz und 3 MHz ist das *Medium Frequency (MF)* Band. Dieses wird hauptsächlich von Mittelwelle-Radiosendern mit Amplitudenmodulation verwendet. Die Reichweite beträgt bei Tag etwa 150 km und kann bei Nacht durch die dann höhere ionosphärische Reflektion auf mehrere hundert Kilometer ansteigen.

Das *High Frequency (HF)* Band oder auf deutsch Kurzwelle liegt zwischen 3 MHz und 30 MHz. In diesem Bereich arbeiten zum Beispiel die Funkgeräte der Amateurfunker. Da die Technik zu der Zeit, als diese Bandeinteilung gemacht wurde, kaum geeignet war, mit Frequenzen oberhalb von 10 MHz zu arbeiten, wurde diese Einteilung als ausreichend angesehen. In dem Maße, wie sich die Technik weiter entwickelte, wurden die hohen Frequenzen mit mehr oder weniger phantasievollen Bezeichnungen weiter unterteilt.

So folgt als nächstes *Very High Frequency (VHF)*, die deutsche Ultrakurzwelle (UKW) mit Frequenzen von 30 MHz bis 300 MHz. Neben den bekannten UKW Radiosendern sind hier auch die terrestrischen Fernsehkanäle 2 – 13 mit je 6 MHz Bandbreite untergebracht. Außerdem gibt es hier einen „freien“ Frequenzbereich, der z. B. von älteren drahtlosen Telefonen verwendet wird.

Als nächstes schließt sich von 300 MHz bis 3 GHz das *Ultra High Frequency (UHF)* Band an. Hier liegen weitere Fernsehkanäle (14 – 69) wobei der Kanal 37 reserviert ist,

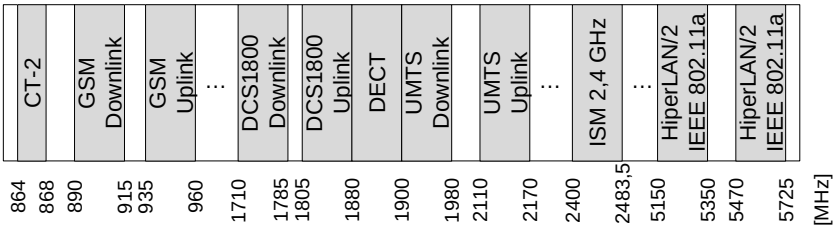

Abbildung 4.2.: Frequenzzuweisungen in Europa (aus [Sik01])

da sich hier ein Beobachtungsfenster für die Radioastronomie befindet. Ursprünglich waren Fernsehkanäle bis 83 vorgesehen, diese wurden jedoch zugunsten von Mobilfunk-Systemen wie GSM 900 gekürzt. In diesem Frequenzbereich arbeitet heute ein Großteil der Systeme zur drahtlosen Datenübertragung, allerdings wird bereits das nächste Band erschlossen.

Das *Super High Frequency (SHF)* Band von 3 bis 30 GHz dient im Bereich von 5 GHz für die Wireless LAN Systeme der nächsten Generation (IEEE 802.11a). Darüber schließt sich das sogenannte *Extra High Frequency (EHF)* Band (30 – 300 GHz) an. Sollte das Spektrum darüber hinaus erschlossen werden, dürfte die Benennung langsam schwierig werden. [Tan96] schlägt dann die Bezeichnungen „Incredible, Astonishingly, and Prodigiously High Frequency (IHR, AHF, and PHF)“ vor.

Die heute für MANETs interessanten Frequenzen liegen etwa zwischen 1 und 6 GHz, lediglich Sensornetze für geringe Datenraten arbeiten manchmal unterhalb dieser Frequenzen. Allerdings besteht von verschiedensten Nutzern wie Militär oder Mobilfunkbetreibern ein Interesse an einer möglichst exklusiven Nutzung eines möglichst breiten Frequenzbandes aus diesem Bereich.

Deshalb wurden durch die nationalen Regulierungsgremien verschiedene Frequenzbänder bestimmten Anwendungen zugewiesen (siehe Abbildung 4.2). Wichtig sind in unserem Zusammenhang das sogenannte *Industrial-Scientific-Medical (ISM) Band* bei 2,4 GHz und im Bereich von 5 GHz das nordamerikanische *Unlicensed National Information Infrastructure (UNII) Band* bzw. das entsprechende europäische *License-Exempt-Frequenzband*. Geräte, die in diesen Frequenzbereichen arbeiten, müssen zwar bestimmte Vorgaben hinsichtlich Sendestärke, Kanalzugriffsverfahren usw. einhalten, darüber hinaus dürfen sie aber, nach einer gerätespezifischen Zulassung, ohne weitere Lizenzen betrieben werden.

Im Gegensatz dazu benötigen Netzbetreiber zum Beispiel im GSM- oder UMTS-Band eine staatliche Lizenz. Wie die Versteigerungen der UMTS Lizenzen in verschiedenen europäischen Ländern gezeigt haben, kann dies sehr teuer werden. Deshalb kommen zum Aufbau von MANETs praktisch ausschließlich Technologien zum Einsatz, welche in den freien Frequenzbändern arbeiten. Weitere Bänder, welche dem ISM Bereich zugerechnet werden, liegen bei 434 und 868 MHz. Diese werden allerdings eher für Anwendungen mit niedriger Datenrate (z.B. Garagentoröffner, Schließsysteme für Pkws etc.) genutzt.

Die zuständigen Gremien bemühen sich, weltweit identische Frequenzbereiche zu reservieren, damit Herstellerfirmen ihre Produkte überall ohne Modifikationen anbieten

können. Allerdings kommt es dabei immer wieder zu Interessenskonflikten, da in manchen Ländern bestimmte Frequenzen zum Beispiel durch militärische Anwendungen bereits blockiert sind. Ein Beispiel hierzu folgt später bei der Beschreibung von IEEE 802.11a/h.

Die weiteren Aspekte der Übertragung elektromagnetischer Wellen, wie Antennenformen, Ausbreitungscharakteristiken usw., sollen hier nicht vertieft werden. Siehe hierzu [Tan96, Sik01, Wal00a].

4.1.2. Medienzugriff

Senden in einem drahtlosen Kommunikationssystem mehrere Sender gleichzeitig auf einer Frequenz, so findet eine Überlagerung der Signale statt. Für einen Empfänger ist es dabei zunächst unmöglich, ein einzelnes Nutzsignal zu isolieren. Aus diesem Grund muss der Zugriff auf das Übertragungsmedium „Funkkanal“ kontrolliert und zwischen den Sendern abgestimmt werden. Hier kommen im Wesentlichen vier unterschiedliche Arten von Medienzugriffsverfahren zum Einsatz [Sik01]:

- das Zeitmultiplexverfahren (*Time Division Multiple Access (TDMA)*)
- das Frequenzmultiplexverfahren (*Frequency Division Multiple Access (FDMA)*) bzw. das darauf aufbauende Orthogonale Frequenzmultiplexverfahren (*Orthogonal Frequency Division Multiplex (OFDM)*)
- das Raummultiplexverfahren (*Space Division Multiple Access (SDMA)*) und
- das Codemultiplexverfahren (*Code Division Multiple Access (CDMA)*)

Bei *TDMA* übertragen die Sender ihre Daten in kurzen Abständen hintereinander. Hierzu ist der Kommunikationskanal meist in kurze Zeitschlitze aufgeteilt, welche den potentiellen Sendern nach einem bestimmten Schema zugeteilt werden.

Das *FDMA*-Verfahren teilt das zur Verfügung stehende Frequenzband in verschiedene Unterbänder und weist jedem der Sender eines dieser Unterbänder zu, so dass es zu keiner Interferenz kommen kann.

SDMA teilt die zu versorgende Fläche in mehrere sog. Zellen und verwendet in benachbarten Zellen unterschiedliche Frequenzen. Damit können Sender Daten an Empfänger in ihrem Bereich schicken, ohne sich gegenseitig zu stören. Durch den räumlichen Abstand und die Dämpfung der elektromagnetischen Wellen sind die Signale von weiter entfernten Sendern auf gleicher Frequenz soweit abgeschwächt, dass für den Empfänger eine klare Trennung möglich ist.

Mobile Ad hoc Netze realisieren in gewisser Weise eine Form von SDMA, da sie eine bestimmte zu versorgende Fläche nicht mit einem leistungsstarken Sender, sondern mit vielen kleinen Sendern geringer Reichweite abdecken. Dadurch erhöht sich die spektrale Effizienz und der maximale Datendurchsatz des Gesamtnetzes.

CDMA gehört zu den sogenannten Frequenzspreizverfahren und wird im nächsten Abschnitt ausführlich beschrieben. Gängige Systeme setzen meist eine Mischung der beschriebenen Verfahren ein, um Störungen zwischen Teilnehmern weitestgehend zu eliminieren.

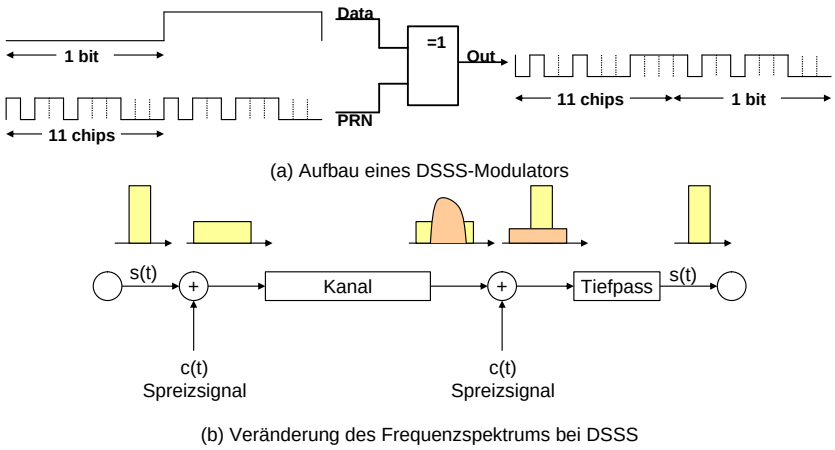

Abbildung 4.3.: Direct Sequence Spread Spectrum (DSSS)

Frequenzspreizverfahren

Die sogenannten Frequenzspreizverfahren (engl. *Spread Spectrum* [FCA+87]) verteilen das schmalbandige Nutzsignal auf einen größeren Frequenzbereich, als dies eigentlich der Symbolrate nach notwendig wäre. Dies wird dazu genutzt, um die Sicherheit einer Übertragung gegen Störsender oder parallel ablaufende Übertragungen zu verbessern. Dabei wird davon ausgegangen, dass ein (Stör-)Sender normalerweise nur auf einem schmalen Frequenzband sendet.

Beim Frequenzsprungverfahren (*Frequency Hopping Spread Spectrum (FHSS)*) wird das zur Verfügung stehende Frequenzband in kleinere Kanäle zerlegt. Sowohl Sender als auch Empfänger wechseln nun in schneller Folge gemäß einer gemeinsamen Pseudozufallsfolge (*Pseudo Random Sequence (PRS)*) den Kanal. Gemäß obiger Annahme wird ein Störsender nur einige wenige Kanäle stören können. Genauso sind Kollisionen, bei denen verschiedene Sender in ihrer PRS zeitgleich den gleichen Kanal belegen, relativ selten und nur von kurzer Dauer. Daher fallen diese Effekte nicht zu stark ins Gewicht und können ignoriert werden.

Anders geht das *Direct Sequence Spread Spectrum (DSSS)* Verfahren vor. Hier wird jedes Bit des Nutzsignals mittels einer Zufalls-Bitfolge (PRS) von z.B. 11 sogenannten Chips kodiert und dann breitbandig übertragen. Der Empfänger kann bei Kenntnis der PRS aus dem breitbandigen Übertragungssignal wieder das schmalbandige Nutzsignal regenerieren. Schmalbandige Störsignale (von Störsendern oder Sendern mit anderer PRS) werden bei dieser Rückwandlung zu breitbandigen Signalen gewandelt und können mit Tiefpassfiltern einfach entfernt werden. Abbildung 4.3 verdeutlicht diesen Vorgang. Im oberen Teil ist die Kodierung des Nutzsignals mit der hochfrequenten Chipsequenz gezeigt, unten sieht man, wie das gelbe Nutzsignal erst gespreizt und dann durch das rote Störsignal überlagert wird. Nach der „Entspreizung“ kann das Störsignal einfach ausgefiltert werden.

Manchmal werden das FHSS und vor allem das DSSS Verfahren als ein Sicherheitsmechanismus bezeichnet, da ohne Kenntnis der PRS das Signal nicht ohne weiteres wieder rekonstruiert werden kann. Da aber sowohl Sender als auch Empfänger über den Startwert der PRS verfügen müssen, wird dieser in heutigen Systemen in der Regel

auf einem ausgezeichneten Kanal vor Beginn der eigentlichen Übertragung im Klartext gesendet oder die Folge ist fest vorgegeben. Somit kann sich ein Angreifer recht einfach auf eine Übertragung aufsynchronisieren und das Frequenzspreizverfahren ist aus sicherheitstechnischer Sicht wirkungslos.

Bei geheimer Übertragung des Startwertes könnte es allerdings wirklich als effektiver Schutz der Signalübertragung auf der Sicherungsschicht (OSI Schicht 2) verwendet werden. Das DSSS-Frequenzspreizverfahren ist auch insoweit interessant, als bei hinreichend starker Spreizung die Amplitude des gespreizten Signals unter das Umgebungsrauschen fallen und somit die Sendung ohne Kenntnis der PRS nicht festgestellt werden kann. Aus diesem Grund wurden Frequenzspreizverfahren ursprünglich im militärischen Bereich entwickelt. Eine aktuelle Forschungsrichtung, die sich mit dieser Anwendung von DSSS beschäftigt nennt sich *Ultra Wide Band (UWB)* [CS02].

4.2. Anwendungen

Drahtlose Netzwerke werden heute in einer Reihe von unterschiedlichen Szenarien eingesetzt. Sogenannte *Wireless Personal Area Networks (WPANs)* erschließen einen Bereich von etwa 10 Metern im Umkreis des Benutzers. Dies kann zum Beispiel zur drahtlosen Anbindung von Peripheriegeräten (Drucker, Scanner usw.) an einen PC dienen. Hier ist das drahtlose Netz also ein reiner Kabelersatz. Darüber hinaus können mit WPANs auch mobile Kleingeräte wie Mobiltelefon, PDA, Headset usw. verbunden werden. Hier ersetzt das WPAN ebenfalls Kabel oder löst ältere Techniken wie Infrarot-Schnittstellen ab. Typischerweise kommen in WPANs Anwendungen zum Einsatz, welche eher moderate Bandbreitenanforderungen (bis 1 Mbps) haben. Teilweise werden WPAN-Technologien jedoch auch zur Rechner-Kopplung eingesetzt und gehen damit nahtlos in den Bereich der WLANs über.

Wireless Local Area Networks (WLANs) sollen die klassischen LANs wie Ethernet ersetzen bzw. ergänzen. Typischerweise sollen ganze Gebäude- oder Firmengelände abgedeckt werden, was durch Einsatz von mehreren *Access Points* geschieht, die jeder eine Fläche von einigen zehn bis wenigen hundert Meter Radius abdeckt. Die Bandbreitenanforderungen sind normalerweise deutlich höher als bei WPANs, der tatsächlich erzielbare Durchsatz liegt mit 5 bis 25 Megabit/s (Mbps) aber eine Größenordnung unter den gängigen drahtgebundenen LANs.

In Zusammenarbeit mit Richtfunk-Antennen wird WLAN Technologie heute auch zur *Punkt-zu-Punkt Verbindung* von LANs oder LANs und WANs verwendet. Im letzten Fall spricht man auch von der sogenannten *Wireless Local Loop (WLL).*

Zur Abdeckung größerer Flächen kommen normalerweise sog. zelluläre Netze zum Einsatz; im Englischen wird diese Technik meist als *cellular radio* bezeichnet. Hier wird (ähnlich wie bei den WLAN Access Points) die zu versorgende Fläche in Zellen aufgeteilt, die jeweils durch eine Basisstation abgedeckt werden. Benachbarte Zellen verwenden typischerweise unterschiedliche Frequenzen, es kommt also ein SDMA Verfahren zum Einsatz. Bei mehreren Teilnehmern innerhalb einer Zelle wird zusätzlich eines der anderen beschriebenen Medienzugriffsverfahren verwendet. Die Anwendungen solcher Systeme reichen von einfachen unidirektionalen Pagingsystemen bis zu den Mobiltelefonsystemen, die heute allgemein verbreitet sind.

4.3. Beispiele

Der folgende Abschnitt stellt für die oben genannten Anwendungsfälle WPAN und WLAN jeweils einen typischen und verbreiteten Vertreter vor. Im Falle von WPANs ist dies Bluetooth [BS01], bei den WLANs wird IEEE 802.11 [IEEc] als typischer Vertreter beschrieben. Beide Systeme sind für diese Arbeit besonders relevant, da diese typischerweise zum Aufbau von MANETs eingesetzt werden. Am Ende des Kapitels werden dann noch weitere Systeme kurz aufgeführt.

4.3.1. Bluetooth

Die Zahl der elektronischen Geräte, mit denen wir uns umgeben, wächst ständig. So besitzen viele Leute heute einen PC, einen PDA, ein Mobiltelefon mit Freisprecheinrichtung (Headset), mobile elektronische Unterhaltungselektronik wie MP3-Spieler und vieles mehr. Oft ist es sinnvoll und wünschenswert, diese Geräte untereinander drahtlos zu vernetzen. So könnte der MP3 Spieler seine Musik über das drahtlose Headset abspielen, welches bei ankommenden Anrufen vom Mobiltelefon genutzt wird. Der PDA könnte via Mobiltelefon auf Daten im Internet zugreifen und seine Adressdaten mit dem PC oder den PDAs anderer Leute abgleichen. Mit Kabeln sind all diese Anwendungen nur aufwändig und für den Benutzer sehr unbequem zu realisieren. Ähnliches gilt im PC Umfeld. Auch hier möchte man viele Geräte wie Drucker, Digitalkamera, MP-3 Spieler, Tastatur oder Maus viel lieber drahtlos an den PC anschließen. Herstellerspezifische Lösungen haben den Nachteil, dass man in den Geräten jeweils mehrere Sender und Empfänger benötigt, die Interoperabilität nicht zwischen allen Gerätekombinationen gegeben ist und es bei Verwendung der gleichen Frequenzen (z.B. im ISM-Band) zu Interferenzen kommt.

Aus diesem Grund ist es naheliegend, einen einheitlichen Kommunikationsstandard zu verwenden. Ericsson gab bereits im Jahr 1994 eine Machbarkeitsstudie für einen solchen Standard in Auftrag. Anfang 1998 gründete Ericsson daraufhin zusammen mit den vier anderen Firmen IBM, Toshiba, Intel und Nokia die *Bluetooth Special Interest Group (BSIG)*, welche im Jahr 1999 die Version 1.0 des sogenannten *Bluetooth* Standards vorlegte. Der Name Bluetooth („Blauzahn“) leitet sich von dänischen König Harald II. Blaatand ab, ein Tribut an die skandinavische Herkunft. Aufgrund verschiedener Schwächen in Version 1.0 folgte im Februar 2001 die Spezifikation 1.1 [Blu01a, Blu01b, Blu01c], welche als Basis für die meisten heute erhältlichen Geräte dient. Zurzeit ist Version 1.2 in Arbeit. Von der Intention her ist Bluetooth eine Art „Wireless USB“. Um eine ähnlich große Verbreitung wie USB zu erreichen, muss Bluetooth vor allem

- ... eine hinreichend hohe *Bandbreite* bieten. Für manche der heute gängigen Anwendungen ist die Datenrate von 1 Mbps (brutto) allerdings zu klein, so dass schon über eine Erweiterung des Standards hin zu höheren Bandbreiten nachgedacht wird.

- ... kostengünstig zu produzieren sein. Das Ziel der BSIG ist es, eine komplette Bluetooth-Einheit in einem Gerät zu Fertigungskosten von unter 5 US Dollar zu ermöglichen. Dieses Ziel scheint realistisch zu sein, so sind immerhin schon USB Bluetooth Module für PCs im Preisrahmen unter 50 Euro erhältlich, die

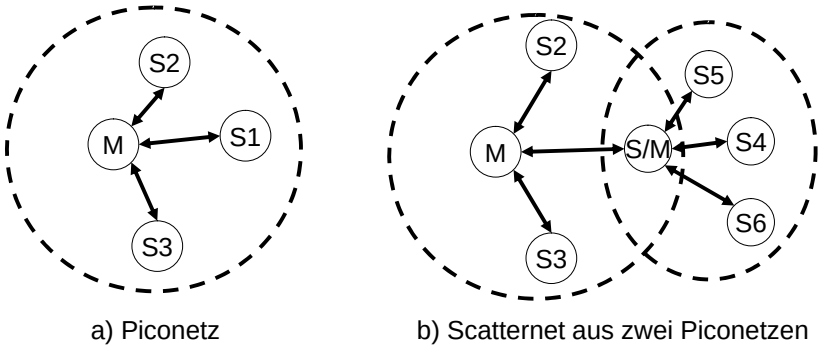

Abbildung 4.4.: Bluetooth: Pico- und Scatternetz

reinen Fertigungskosten einer Einbaulösung dürften deutlich darunter liegen und bei steigenden Stückzahlen weiter sinken.

- ... energieeffizient arbeiten. Insbesondere in kleinen mobilen Elektronikgeräten wie Mobiltelefonen ist Energie eine knappe Ressource. Trotz Fortschritten bei der Herstellung von Batterien und Akkus würde Bluetooth kaum akzeptiert werden, wenn der Akku des Mobiltelefons nach einer halben Stunde leer wäre. Entsprechend legt Bluetooth großen Wert auf Energiesparmaßnahmen und ist hier deutlich effizienter als beispielsweise das später vorgestellte IEEE 802.11 Wireless LAN. Es gibt drei verschiedene Energiesparmodi mit Namen SNIFF, HOLD und PARK, die es einem Gerät erlauben, sich schrittweise aus der aktiven Kommunikation zurückzuziehen und nur noch periodisch den Datenverkehr abzuhören.

[BS01] gibt einen guten Überblick über die Technik von Bluetooth. Es kommt ein FHSS-System zum Einsatz. Dieses gruppiert die Netzwerk-Teilnehmer in sogenannte *Piconetze*, wie in Abbildung 4.4 a zu sehen ist. In einem Piconetz benutzen alle Teilnehmer die gleiche Sprungsequenz (*Hopping Sequence*), die vom sogenannten Master vorgegeben wird. Vor einer Kommunikation müssen zwei Geräte zunächst explizit eine Verbindung aufbauen. Der Initiator wird dabei automatisch *Master* des Piconetzes, der Knoten zu dem eine Verbindung aufgebaut wird ist ein *Slave*. Ein Master kann bis zu sieben Slave-Knoten kontaktieren, dann ist die Kapazität des Piconetzes erschöpft. Die Kommunikation im Piconetz läuft ausschließlich über den Master, der die Slaves via Polling steuert. Zwei Slaves können nicht direkt kommunizieren.

Kontaktiert einer der Slaves einen weiteren Knoten, so wird dieser zum Master in einem neuen Piconetz. Der Knoten ist somit Slave im ursprünglichen Piconetz und Master in einem neuen Piconetz. Er wird in diesem Fall als *Bridge-Knoten* bezeichnet, eine Ansammlung mehrerer, durch Bridge-Knoten verbundener Piconetze nennt man *Scatternetz* (Abbildung 4.4 b). Dies kommt der Idee der Mobilen Multi-Hop Ad hoc Netze, wie sie im folgenden Kapitel beschrieben werden, schon sehr nahe. Der Standard beschreibt allerdings die Funktionsweise eines Piconets nur sehr unzureichend. So ist beispielsweise die Verkehrslenkung (*Routing*) im Scatternet nicht weiter ausgeführt. In Abschnitt 5.3.4 erläutern wir ein von uns entwickeltes Routing Verfahren für Bluetooth Scatternetze und weisen auch auf einige andere Forschungsarbeiten zu diesem Thema hin.

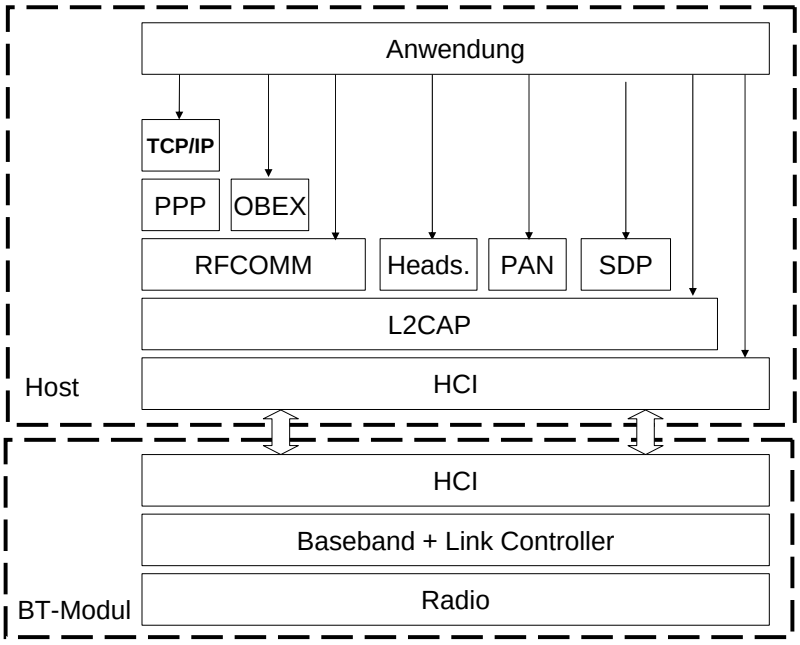

Abbildung 4.5.: Bluetooth: Stack

Es gibt zwei Arten von Verbindungen zwischen Bluetooth-Knoten: *Asynchronous Connection-Less (ACL)* und *Synchronous Connection Oriented (SCO)*. ACL dient zur Übertragung von herkömmlichem Datenverkehr. Dabei findet eine Sicherung durch das Verfahren „Bestätigung mit erneuter Übertragung“ (engl. *Acknowledgment with Retransmit*) statt. SCO Verbindungen kennen diese Sicherung nicht, sie dienen zur Übertragung von isochronen Daten mit fester Datenrate von 64 kbps, insbesondere wurden sie zur Sprachübertragung (z.B. vom Headset zum Mobiltelefon) entwickelt. Beide Verbindungsarten kennen eine optionale redundante Kodierung mit *Forward Error Correction*.

Zur Untergliederung der Funktionalität eines Bluetooth-Systems hat die BSIG ein mehrschichtiges Referenzmodell, den sogenannten *Bluetooth Stack*, entwickelt (siehe Abbildung 4.5).

Der *Radio-Layer* regelt die Funkübertragung der Daten. Wie bereits angesprochen kommt ein FHSS Verfahren im 2,4 GHz ISM Band zum Einsatz, welches die zur Verfügung stehende Bandbreite in 79 Kanäle zu 1 MHz unterteilt. Die Daten werden mit 1 Megabaud (1 Million Signale pro Sekunde) mit einer GFSK (*Gaussian Frequency Shift Keying*) Modulation übertragen, was zur erwähnten Bandbreite von 1 Mbps führt. Bluetooth definiert drei verschiedene Sendeklassen. *Klasse 1* Geräte senden mit 100 mW und haben eine Reichweite von bis zu 100 Metern, *Klasse 2* sendet mit 2,5 mW und *Klasse 3* erreicht mit 1 mW noch etwa 10 Meter Reichweite. Die meisten der heute verfügbaren Geräte gehören zur Klasse 3.

Die *Baseband*-Schicht ist für die Datenübertragung, Fehlererkennung und -korrektur und grundlegende Verschlüsselung zuständig. Der *Link-Controller* regelt den Verbindungsauf- und -abbau und die Suche nach Nachbarn (engl. *Inquiry*). Hierzu wird das Modell eines Zustandsautomaten verwendet. Der lokale Bluetooth-Knoten kann, wie

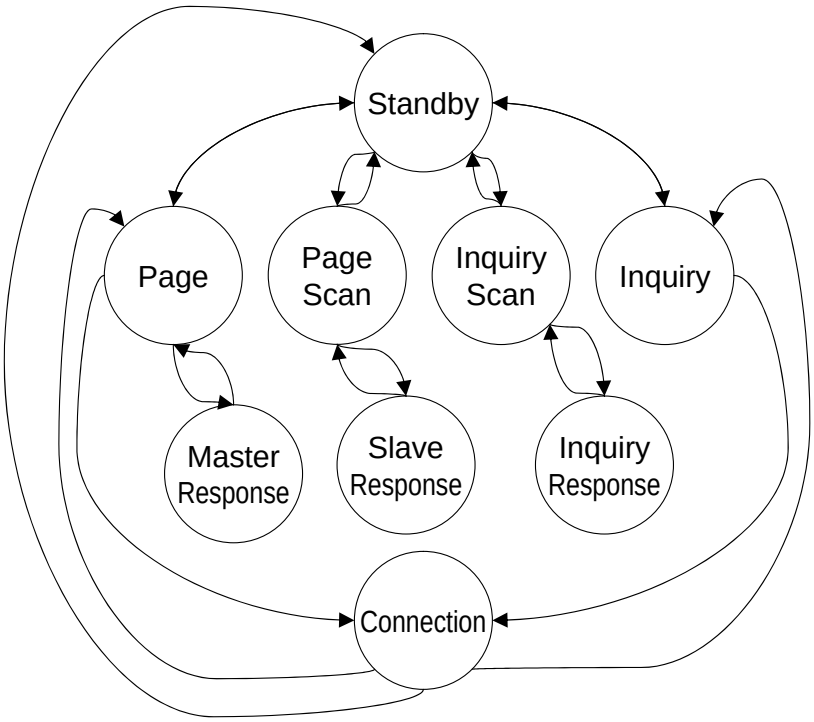

Abbildung 4.6.: Bluetooth: Link Controller (aus [BS01])

in Abbildung 4.6 gezeigt, zwischen den Zuständen wechseln, aber niemals in mehreren Zuständen gleichzeitig sein. Deshalb kann er beispielsweise nicht eine Verbindung aufbauen und gleichzeitig eine Inquiry beantworten. Als Konsequenz sind die Zeiten für Verbindungsaufbau und Inquiry relativ lang, da eine Anfrage oft mehrmals wiederholt werden muss, bis sich das Zielgerät im richtigen Zustand befindet.

Das *Host Controller Interface (HCI)* stellt eine standardisierte Schnittstelle zwischen dem Bluetooth-Modul und dem Host-Rechner dar. Der Standard beschreibt eine Anbindung via UART-Chip, RS232-Interface oder über USB. Andere Anbindungen z.B. über einen CAN Bus im Auto sind natürlich denkbar. Dank genormtem HCI gestaltet sich das Zusammenspiel zwischen Bluetooth-Hardware und dem Software-Teil im Rechner in der Regel sehr problemlos.

L2CAP steht für *Logical Link Control and Adaptation Protocol*. Im Wesentlichen ist diese Schicht ein Paketmultiplexer, welcher mehrere logische Verbindungen zu einem Gerät verwaltet und für die Fragmentierung und Defragmentierung der Datenpakete zuständig ist. Auf L2CAP setzen die sogenannten Bluetooth-Profile auf, welche Protokolle für bestimmte Anwendungsprofile definieren. So gibt es mit *RFCOMM* ein Profil, welches eine serielle Schnittstelle inklusive Steuerleitungen über Bluetooth emuliert. Oft werden auf RFCOMM weitere Protokolle aufgesetzt, die man sonst bei serieller Verbindung von Geräten nutzt. Zusammen mit PPP kann so eine Punkt-zu-Punkt Netzwerkverbindung realisiert werden. Mit dem *Object Exchange (OBEX)* Protokoll lassen sich Adressbücher und Termine abgleichen. Sehr speziell ist das *Headset-Profil*, welches die Audio-Übertragung mit Headsets regelt. *PAN* steht für *Personal Area Networking* und erlaubt den effizienten Aufbau von Datennetzen zwischen Bluetooth-

Knoten. PAN soll die Kombination RFCOMM + PPP ersetzen. Einen Überblick über die bisher spezifizierten Profile liefern [Blu01a] und [Blu01c].

Ein wichtiges Element des Protokoll Stacks ist das *Service Discovery Protocol (SDP)*. Dieses definiert in XML eine Dienste-Beschreibungssprache, über die ein Knoten anderen Knoten mitteilen kann, welche Profile er unterstützt.

Bluetooth ist natürlich deutlich komplexer als hier verkürzt dargestellt. Umfangreichere Informationen finden sich in [BS01]. Im Folgenden soll lediglich noch auf die Sicherheitsfunktionen von Bluetooth eingegangen werden, die einen essentiellen Teil des Standards ausmachen.

Bluetooth kennt drei unterschiedliche Sicherheits-Modi: Modus 1 deaktiviert alle Sicherheitsfunktionen. Daten werden nicht verschlüsselt, Kommunikationspartner werden bei Verbindungsaufbau nicht authentisiert. Modus 2 realisiert die Sicherheit auf dem Service-Level, d.h. pro logischer Verbindung, während Modus 3 Sicherheit auf der Verbindungsebene erzwingt, d.h. die Authentisierung findet bereits statt, bevor eine ACL oder SCO Verbindung aufgebaut wird.

Bluetooth kennt unterschiedliche Arten von Schlüsseln, die durch verschiedene Algorithmen generiert werden. $E0$ dient dabei der Generierung einer Pseudozufallszahlenfolge zur Verschlüsselung der Nutzdaten, $E1$ ist für die Authentisierung zweier Geräte und Absicherung des Schlüsselaustauschs zuständig. $E2$ umfasst verschiedene Algorithmen zur Generierung der sogenannten *Link Keys*, welche zur Identifizierung einer Verbindung zwischen zwei Geräten und zur Generierung des eigentlichen *Encryption Keys* verwendet werden. Schließlich dient $E3$ der Erzeugung dieses Encryption Keys, der in $E0$ Eingang findet.

Der *Link Key* hat eine Länge von 128 Bit und wird durch verschiedene Abwandlungen des sogenannten SAFER+ Algorithmus [MKK98] erstellt. Er gilt entweder temporär während einer Sitzung oder semi-permanent über mehrere Sitzungen hinweg. Dabei gibt es vier unterschiedliche Arten des Link Keys:

- *Initialization Key*
- *Unit Key*
- *Combination Key*
- *Master Key*

Der *Initialization Key* wird beim erstmaligen Treffen zweier Bluetooth-Geräte verwendet, um die Initialisierungsphase abzusichern, solange noch keine weiteren Schlüssel ausgetauscht wurden. Er wird mit Variante E_{22} von SAFER+ aus der Bluetooth-Adresse, einer PIN und einer Zufallszahl generiert. E_{22} kommt ebenfalls zum Einsatz, wenn der Piconetz Master einen sogenannten *Master Key* generieren möchte. Dieser wird benötigt, wenn der Master eine Punkt-zu-Multipunkt Nachricht an alle Slaves schicken möchte. Dabei wird die Verschlüsselung kurz auf den Master-Key umgeschaltet, im Anschluss werden dann wieder die ursprünglichen Link-Keys verwendet.

$$E_{22}(Random; [PIN;]ADDR) \longrightarrow initkey\ [128bit]$$

Der *Unit Key* wird typischerweise während des Produktionsprozesses eines Bluetooth-Geräts erzeugt und permanent im Gerät gespeichert. Der *Combination Key* wird von

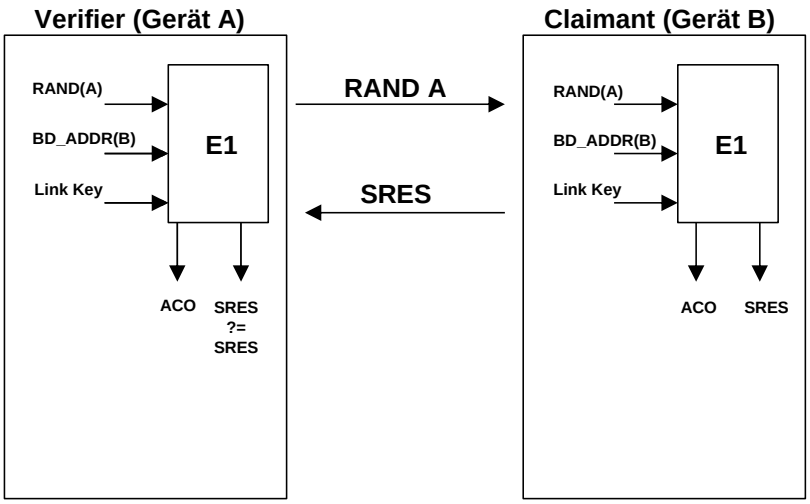

Abbildung 4.7.: Bluetooth Authentisierung

zwei Geräten gemeinsam erzeugt, die beide jeweils eine Zufallszahl beisteuern. Beide Schlüssel werden mit der E_{21} Variante von SAFER+ generiert.

$$E_{21}(Random; ADDR) \longrightarrow unit/combinationkey\ [128bit]$$

Aus dem jeweils aktuell gültigen Link Key wird anschließend mittels der E_3 Variante der sogenannte *Encryption Key* gebildet. Darin findet neben der Zufallszahl und dem Link Key noch der *Ciphering Offset COF* Eingang. Der Encryption Key wird für jede Verbindung neu berechnet.

$$E_3(Random; COF; LinkKey) \longrightarrow encryptionkey\ [128bit]$$

Schließlich wird aus dem Encryption Key mittels des E_0 Algorithmus eine Pseudozufallszahlenfolge (der *Cipher Stream*) gebildet, der dann mit den Nutzdaten XOR verknüpft wird.

Abbildung 4.7 zeigt, wie mittels $E1$ eine Challenge-Response Authentisierung stattfindet. Existiert schon ein Link-Key, so wird dieser verwendet, andernfalls wird mittels einer PIN ein Initialization Key gebildet. In Abbildung 4.8 wird gezeigt, wie mittels des Encryption Keys ein Keystream gebildet wird, der zur Ver- und Entschlüsselung des Datenverkehrs dient.

Das Sicherheitssystem von Bluetooth wurde bereits kurz nach Veröffentlichung des Standards kritisiert. So sind in [JW01] verschiedene mögliche Angriffe gegen das Authentisierungsverfahren erläutert. Das Abhören einer Verbindung ist trotz Frequency Hopping relativ einfach, da sämtliche Parameter zur Bestimmung der Hopping Sequence (Master Address und Master Clock) zu Beginn einer Verbindung im Klartext übertragen werden. Damit lassen sich Zufallszahlen und Parameter des Authentisierungsverfahrens abhören. Anschließend kann man versuchen, die PIN zu ermitteln. Dazu probiert man verschiedene PINs offline aus und vergleicht die Ergebnisse mit

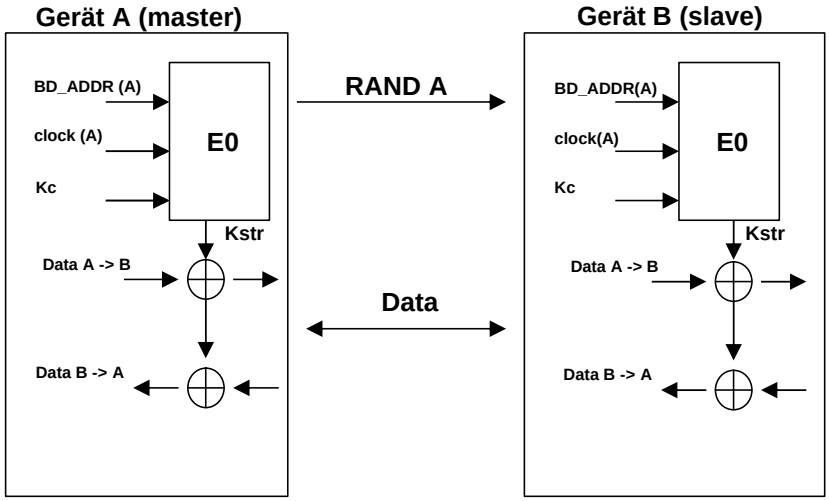

Abbildung 4.8.: Bluetooth Verschlüsselung

den aufgezeichneten Daten. Stimmen diese überein, hat man die PIN gefunden. Geräte ohne Eingabemöglichkeit verfügen oft über eine fest eingestellte PIN, die sich auch nicht ändern lässt und meist aus nur wenigen Ziffern besteht. Oft ist diese PIN sogar „0000“. Hier ist das Ausprobieren besonders einfach.

Ein weiteres Problem sind Geräte, die auf Suchanfragen (*Inquiries*) antworten. Stellt beispielsweise ein großer Supermarkt in seinen Verkaufsräumen Stationen auf, die ständig nach Bluetooth-Geräten suchen, so lassen sich daraus recht zuverlässig Bewegungsmuster ermitteln. Diese können dann zu Marketingzwecken verwendet werden oder um die Warenanordnung zu optimieren. Aus Datenschutzgründen sind solche Praktiken natürlich abzulehnen.

4.3.2. IEEE 802.11

Sämtliche heute relevanten LAN Standards wurden durch das *Institute of Electrical and Electronic Engineers (IEEE)* [IEEa] standardisiert. Grafik 4.9 zeigt einen Überblick über die verschiedenen Standards des *IEEE 802 LAN/MAN Standards Committee* [IEEb], welche Netzwerktechnologien vom Personal- bis zum Metropolitan-Area Bereich umfassen.

Obwohl die meisten der später vorgestellten Verfahren zur Bildung von Ad hoc Netzen prinzipiell von der eingesetzten Funknetz-Technologie unabhängig sind, verwenden praktisch alle Arbeitsgruppen für Prototypen oder Simulationen Funknetzwerke nach IEEE 802.11 [IEEc]. Deshalb soll dieser Standard im Folgenden etwas ausführlicher erklärt werden. Eine ausführliche Behandlung des Themas findet sich in [Sik01].

Der 802.11 Standard bzw. dessen Substandards definieren die Bitübertragungsschicht und die Sicherungsschicht eines Netzwerks, wobei die Sicherungsschicht nochmals in *Medium Access Control (MAC)* und *Logical Link Control (LLC)* unterteilt wird. Die LLC-Schicht IEEE 802.2 ist dabei für alle Standards nach IEEE 802 gleich, was eine einfache Interoperabilität (*Bridging*) zwischen den verschiedenen Systemen ermöglicht.

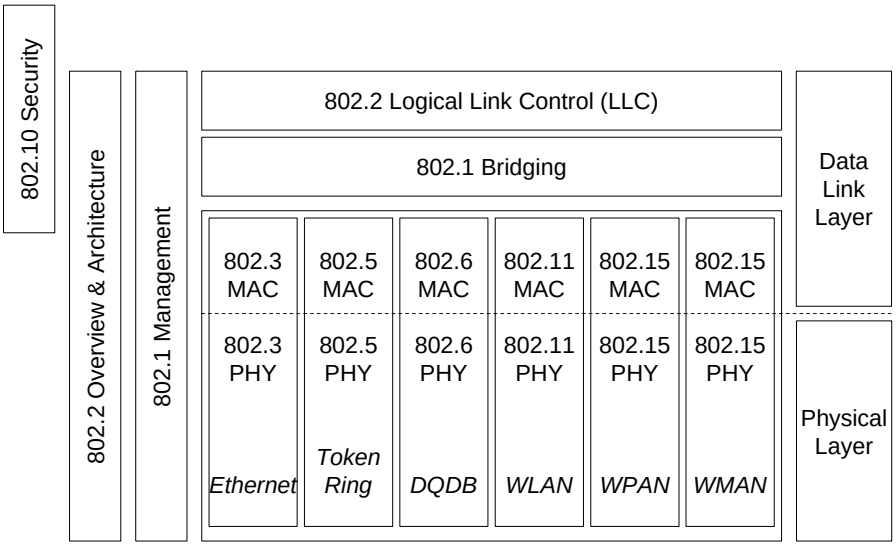

Abbildung 4.9.: Übersicht über die Standards IEEE 802.x (aus [Sik01])

Deshalb kann beispielsweise ein WLAN Access Point sehr einfach als Bridge zu einem herkömmlichen Ethernet (IEEE 802.3) dienen. Innerhalb des IEEE 802.11 WLAN Standards ist wiederum die MAC Schicht weitestgehend identisch, wohingegen sich die physikalische Schicht unterscheiden kann.

Der ursprüngliche Standard von 1997 [IEEd] (erweitert 1999 [IEEe]) sieht drei verschiedene Übertragungsmethoden vor: Infrarot, Funk mit FHSS und Funk mit DSSS bei Datenraten von jeweils 1 oder 2 Mbps. Dabei nutzt der Standard das ISM Band bei 2,4 GHz. Heute spielt eigentlich nur noch Funk mit DSSS eine Rolle, die anderen Technologien sind vom Markt verschwunden. Dies hängt auch damit zusammen, dass alle späteren Erweiterungen mit höherer Geschwindigkeit ausschließlich DSSS einsetzen.

1999 wurde der Standard um IEEE 802.11b [IEEh, IEEg] erweitert. Dieser ermöglicht durch Einsatz einer leistungsfähigeren Modulation (*Quadrature Phase Shift Keying (QPSK)*) zusätzliche Datenraten von 5,5 und 11 Mbps. Parallel dazu wurde der Standard IEEE 802.11a [IEEf] vorgestellt, welcher bei 5 GHz mittels OFDM Bruttodatenraten von 54 Mbps erreicht. Die Einführung verzögerte sich wegen diverser Probleme mit Technik und Frequenzfreigabe, so dass erst 2002 erste Produkte verfügbar waren. In Europa waren weitere Anpassungen notwendig, um Störungen mit anderen Systemen im gleichen Frequenzbereich (z.B. Radareinrichtungen) zu vermeiden, diese beschreibt IEEE 802.11h [IEEl]. IEEE 802.11g [IEEk] versucht, auch bei 2,4 GHz höhere Datenraten (bis 54 Mbps) zu erreichen. Daneben existiert eine Vielzahl weiterer Aktivitäten, die sich beispielsweise mit Sicherheit [IEEm] oder Quality of Service [IEEi] befassen.

WLANs nach IEEE 802.11 können entweder im sogenannten Ad hoc Modus oder im Infrastruktur-Modus betrieben werden. Beim Ad hoc Modus bilden verschiedene Knoten spontan ein Layer 2 Netz, wobei sich alle Knoten in gegenseitiger Funkreichweite befinden müssen. Eine Verkehrsweiterleitung durch Zwischenknoten findet (im Gegensatz zu den in Kapitel 5 beschriebenen MANETs) nicht statt. Im Infrastrukturmodus

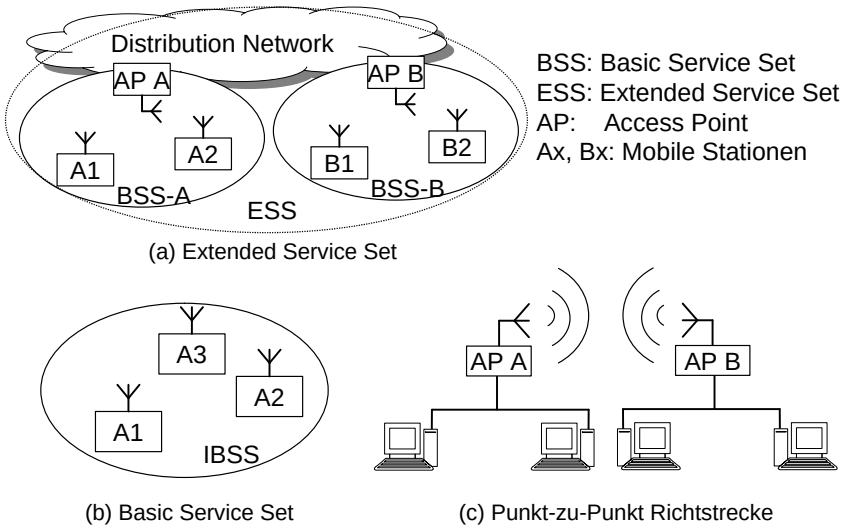

Abbildung 4.10.: IEEE 802.11: Verschiedene mögliche Konfigurationen von IEEE 802.11 Wireless LANs

melden sich alle Knoten bei einem zentralen Access Point (AP) an, über den sämtliche Kommunikation läuft.

Die Knoten eines Ad hoc Netzes bilden ein sog. *Independent Basic Service Set (IBSS)*, alle Knoten in Reichweite eines Access Points nennt man *Basic Service Set (BSS)*. Der Standard definiert darüber hinaus noch das *Extended Basic Service Set (EBSS)*. Dieses besteht aus mehreren BSSs plus einem sogenannten *Distribution System (DS)*, welches die Access Points der BSSs verbindet. Über das DS tauschen die BSSs Verkehr aus und organisieren die Lokalisierung und das Roaming von Teilnehmern zwischen verschiedenen BSSs. Da der Standard hierzu allerdings kein Protokoll beschreibt, bieten heutige Systeme eigentlich kein EBSS. Vielmehr werden mehrere unabhängige APs meist über Ethernet-Switches verbunden. Wechselt ein Teilnehmer zu einem anderen AP findet dort eine neue Anmeldung statt und es ist Aufgabe des Ethernets, die Daten wieder zum richtigen AP zu transportieren. Entsprechende Anstrengungen zur Entwicklung eines EBSS Protokolls unternimmt [IEEj].

Abbildung 4.10 zeigt die verschiedenen Formen von IEEE 802.11 WLANs, wobei zusätzlich noch eine Punkt-zu-Punkt Richtfunkstrecke abgebildet ist. Diese Konfiguration ist zwar im Standard nicht vorgesehen, wird aber in der Praxis häufig zur LAN-LAN-Kopplung verwendet.

Bei IEEE 802.11(b) mit DSSS wird der zur Verfügung stehende Frequenzbereich von 2,4 bis 2,4835 GHz in 14 Kanäle von jeweils 22 MHz Bandbreite eingeteilt (regional sind evtl. weniger Kanäle verfügbar). Wie man in Abbildung 4.11 sieht, überlappen sich diese Kanäle, so dass maximal drei BSS störungsfrei parallel betrieben werden können, zum Beispiel auf den Kanälen 1, 6 und 11. Es ist Aufgabe des WLAN-Administrators, die Kanäle der APs so einzustellen, dass benachbarte APs keine überlappenden Frequenzen verwenden. Sonst kommt es zu Interferenzen und Geschwindigkeitseinbußen.

Beim Kanalzugriff unterstützt IEEE 802.11 zwei verschiedene Verfahren. Im Normalfall kommt die *Distributed Coordination Function (DCF)* zum Einsatz, welche ein CS-

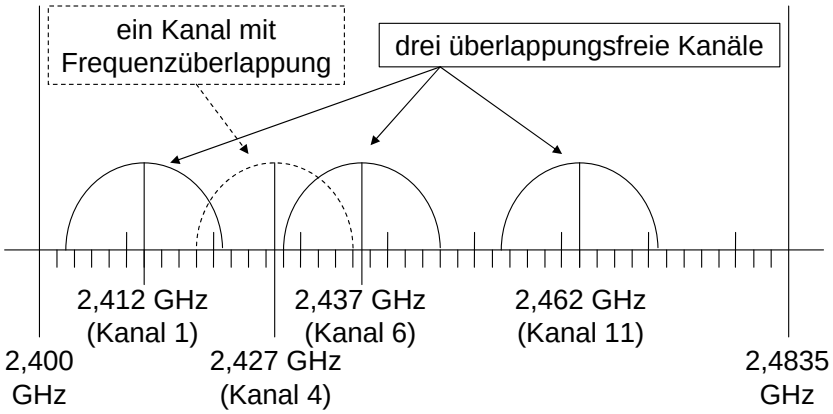

Abbildung 4.11.: IEEE 802.11: Kanäle bei IEEE 802.11 Wireless LANs(aus [Sik01]

MA/CA (*Carrier Sense Multiple Access/Collision Avoidance*) realisiert. Dabei muss ein Teilnehmer mit Sendewunsch zunächst für eine kurze Zeitspanne prüfen, ob das Medium frei ist. In diesem Fall kann er nach Ablauf dieser Zeit mit dem Senden beginnen. Falls das Medium belegt ist, wird sofort ein Wartezyklus (*Backoff*) durchgeführt, bevor erneut geprüft wird. Die Wartezeit wird dabei teilweise zufällig bestimmt, so dass Kollisionen relativ unwahrscheinlich sind. Pakete müssen grundsätzlich mit einem *Acknowledge* bestätigt werden, so dass eine auftretende Kollision oder eine sonstige Störung erkannt und das Paket nochmals übertragen wird. Dies gilt aber nur für Unicast Pakete, Broad- und Multicast wird nicht über diesen Weg abgesichert, was sich beim Fluten in MANETs gelegentlich negativ bemerkbar macht.

Treten in Netzen mit vielen Knoten und großer geographischer Verteilung häufiger Kollisionen auf, so kann zusätzlich ein *RTS-CTS-Mechanismus* aktiviert werden. Hier schickt eine sendewillige Station zunächst ein *Ready-To-Send* Paket an den Empfänger. Kann dieser das RTS problemlos empfangen, so antwortet er mit einem *Clear-To-Send*, was gleichzeitig Sendungen von benachbarten Stationen für die Dauer der Übertragung blockiert. Damit löst RTS-CTS insbesondere das *Hidden-Station-Problem*, da ein Sender trotz Carrier-Sense eigentlich keine Aussage über den Zustand des Mediums beim Empfänger machen kann.

Für die Unterstützung von zeitkritischen Diensten sieht IEEE 802.11 die *Point Coordination Function (PCF)* vor. Hier teilt ein zentraler Knoten (der *Point Coordinator (PC)*, normalerweise der Access Point), die Zeit in Intervalle ein. Jedes dieser Intervalle gliedert sich in eine wettbewerbsfreie Zeit (*Contention Free Period (CFP)*) und eine Wettbewerbsperiode (*Contention Period (CP)*), in der die normale DCF zum Einsatz kommt. Während der CFP steuert der PC die Kommunikation durch ein Polling-Verfahren und ist so in der Lage, den Verkehr zu steuern und zu priorisieren. Kollisionen (innerhalb des BSS) werden solange vermieden. Die PCF realisiert damit eine einfache und effiziente Basis zur Implementierung von Dienstegüte-Mechanismen. Allerdings ist die PCF in den meisten der heute verfügbaren Systeme nicht implementiert und auch der Standard weist noch etliche Lücken auf, so dass existierende Lösungen oft nicht interoperabel sind.

IEEE 802.11 Sicherheit

Den Entwicklern von 802.11 war klar, dass sich ein Funk-LAN nur schwer verkaufen lassen würde, wenn jeder den Datenverkehr einer Firma noch in 300 Metern Entfernung mithören könnte. Daher wurde ein Sicherheitssystem mit dem Namen *Wired Equivalent Privacy (WEP)* in den Standard integriert. Wie der Name schon andeutet, war es dabei nicht das Ziel, ein umfangreiches Sicherheitsframework wie IPsec zu entwickeln, vielmehr sollte dem WLAN-Nutzer ein ähnliches Sicherheitsniveau geboten werden, wie dem LAN Nutzer in einem Ethernet, dessen Kabel geschützt und weitgehend abhörsicher im Gebäude verlegt sind. Im Einzelnen waren die Ziele:

1. Kein Abhören des Datenverkehrs durch Unbefugte
2. Keine Modifikation von Daten oder Erzeugen neuer Pakete
3. Kontrolle des Netzwerkzugangs

Bei WEP kommt das RC4 Verfahren [Riv92a] mit 64 oder 128 Bit Schlüssellänge zum Einsatz. Dabei wird aus dem Schlüssel ein Pseudozufallszahlenstrom (*Pseudo-random Keystream (PRK)*) generiert. Dieser PRK wird anschließend mit den Daten durch ein Exklusiv-Oder (XOR) verknüpft. Dabei bleibt, ähnlich dem Onetime-Keypad, die zufällige Verteilung des PRKs erhalten. Allerdings gelten die für Onetime-Keypads gemachten Einschränkungen, wonach ein PRK niemals mehrfach verwendet werden darf. Lägen nämlich einem Angreifer mehrere mit dem gleichen PRK verschlüsselte Datenpakete vor, so könnte er relativ einfach die Verschlüsselung knacken. Dies führt, wie im Folgenden gezeigt, zu einer Vielzahl von Unsicherheiten bei WEP.

Seien $C1$ und $C2$ der verschlüsselte Text aus Paket 1 und 2 (C = ciphertext). Seien weiter $P1$ und $P2$ die zugehörigen Klartexte (P = plaintext). Sei $RC4(k)$ der mit dem Schlüssel k generierte RC4 PRK. Dann gilt

$$\begin{aligned} C1 &= P1 \oplus RC4(k) \\ C2 &= P2 \oplus RC4(k) \\ \text{daraus folgt} & \\ C1 \oplus C2 &= P1 \oplus P2 \end{aligned}$$

Da in $P1 \oplus P2$ der PRK eliminiert ist, lässt sich bei bekanntem $P1$ oder $P2$ der jeweils andere Klartext direkt berechnen. Auch ohne Kenntnis eines Klartextes sind derartige Codes durch statistische Analysen recht einfach zu brechen. Um also nicht jedes Paket mit dem gleichen PRK zu verschlüsseln, wird bei WEP ein sogenannter Initialisierungsvektor (*Initialization Vector (IV)*) eingesetzt. Dieser hat eine Länge von 24 Bit und wird in jedem Paket am Anfang im Klartext mitgeschickt. Die effektive Schlüssellänge verkürzt sich dadurch von 64/128 auf 40/104 Bit.

Zusätzlich wird über die Nutzdaten des Pakets mittels CRC-32 eine Prüfsumme, der sogenannte *Integrity Check Value (ICV)*, gebildet, die am Ende angehängt wird. Nutzdaten plus Prüfsumme werden dann mit RC4 verschlüsselt, wobei als Schlüssel die Konkatenation von IV und konfiguriertem Schlüssel verwendet wird.

Neben der Verschlüsselung sieht IEEE 802.11 auch vor, dieses Verfahren zur Authentisierung von Teilnehmern an einem BSS zu nutzen. Hierzu sendet der AP eine zufällige

Herausforderung C (*Challenge*) an den Teilnehmer. Dieser muss C mit dem korrekten RC4 Schlüssel k verschlüsseln und an den AP zurückschicken. Dieser entschlüsselt den Text und prüft das Ergebnis auf Gleichheit mit der ursprünglichen Challenge. Erst danach wird ein Zugang zum WLAN gewährt.

$$\begin{aligned} \text{AP} &\longleftrightarrow \text{Client} \\ C &\longrightarrow \\ &\longleftarrow \quad M = C \oplus RC4(k) \\ C \oplus RC4(k) &\stackrel{?}{=} M \end{aligned}$$

Verschiedene Forschergruppen haben zwischen Oktober 2001 und August 2002 Ergebnisse veröffentlicht, die schwere Sicherheitsmängel in WEP nachweisen. Dabei werden den Entwicklern von WEP schwere handwerkliche Mängel vorgeworfen, insbesondere die Verwendung viel zu kurzer IVs und die Nutzung des für kryptographische Zwecke ungeeigneten Prüfsummenverfahrens CRC-32.

Im Oktober 2000 legte Jesse Walker der IEEE 802.11 Arbeitsgruppe einen Beitrag mit dem Titel „Unsafe at any key size; An Analysis of the WEP encapsulation“ vor, in dem er auf einige grundlegende Probleme mit WEP hinwies [Wal00b]. Kurze Zeit später veröffentlichten Goldberg, Borisov und Wagner eine Arbeit [BGW01], in der sie zeigten, dass es bei WEP zu IV Kollisionen kommt und man damit Datenpakete entschlüsseln kann. Weiterhin zeigten Sie, dass sich der Paketinhalt ändern lässt, ohne dass der Empfänger dies merkt. Die CRC-32 Checksumme ist linear bezüglich der XOR Funktion. So ist es möglich, Bits im Datenteil zu verändern und gleichzeitig die Checksumme so anzupassen, dass diese gültig bleibt. Dies funktioniert, ohne das Paket selbst entschlüsseln zu müssen oder den Schlüssel zu kennen. Denn wenn $RC4(k, M)$ die RC-4 Verschlüsselung der Nachricht M mit Schlüssel k und $CRC(M)$ die CRC-32 Prüfsumme über M ist, dann gilt:

$$\begin{aligned} CRC(X \oplus Y) &= CRC(X) \oplus CRC(Y) \\ RC4(k, X \oplus Y) &= RC4(k, X) \oplus Y \\ \text{und somit} & \\ RC4(k, CRC(X \oplus Y)) &= RC4(k, CRC(X)) \oplus CRC(Y) \end{aligned}$$

Darauf aufbauend lassen sich Angriffe konstruieren, bei denen man Pakete zum Angreifer umleiten oder über bestimmte Reaktionen des TCP Protokolls den Klartext entschlüsseln kann. Arbaugh zeigte, wie sich durch ein systematisches Vorgehen ein umfangreiches „*Keystream-Dictionary*“ aufbauen lässt, das die Entschlüsselung des kompletten Netzwerkverkehrs erlaubt [ASW01, Arb01]. In den gleichen Arbeiten wird darüber hinaus noch gezeigt, wie man die WEP-Authentisierung umgehen kann. Zur gleichen Zeit entwickelte Tim Newsham [New01] ein System, welches eine „*Dictionary Attack*“ auf den WEP-Schlüssel durchführt, da der WEP-Schlüssel oftmals aus einem ASCII-Wort generiert und durch die verwendete Umsetzung der mögliche Schlüsselraum drastisch eingeschränkt wird.

Während die bisherigen Angriffe nur einzelne Aspekte der WEP-Sicherheit angriffen und viele Hersteller weiterhin behaupteten, dass ihre Systeme trotzdem sicher seien, zeigten Fluhrer, Mantin und Shamir im August 2001, dass WEP hoffnungslos unsicher war [FMS01]. Sie fanden heraus, dass während der Generierung der Zufallszahlenfolge PRK, bei Verwendung von bestimmten IVs, einzelne Bits des Schlüssels einen höheren Einfluss auf die ersten Bytes des PRK haben als andere. Sammelt man genügend verschlüsselte Datenpakete mit solchen sogenannten *Weak IVs*, lässt sich durch eine statistische Analyse der komplette Schlüssel finden. Schon einen Monat später berichtete Stubblefield, dass er zusammen mit Kollegen diesen theoretischen Angriff erfolgreich in die Praxis umgesetzt hat [SIR01]. Zwar hielten sie ihre Programme unter Verschluss, doch schon bald tauchten im Internet die ersten öffentlich verfügbaren Tools auf [AIR].

Damit war klar, dass WEP keines der gesteckten Ziele erreicht. Die Hersteller versuchten, durch proprietäre Patches die Sicherheit ihrer Produkte zumindest etwas zu erhöhen. Parallel wurde von der IEEE eine neue Version von WEP entwickelt, die jedoch wegen des schlechten Rufes von WEP in *Temporal Key Integrity Protocol (TKIP)* [IEEm] umbenannt wurde. Zusammen mit dem Authentifizierungsprotokoll 802.1x soll damit die Sicherheit der Wireless LANs wieder hergestellt werden.

Zusammenfassend lässt sich sagen, dass man sich zur Absicherung eines WLANs nicht nur auf die im Standard vorgesehenen Sicherheitsmechanismen verlassen sollte, sondern zusätzliche Vorkehrungen zum Beispiel in Form von Firewalls und Virtuellen Privaten Netzen nötig sind. Die Erfahrung zeigt gleichwohl, dass circa drei Viertel der heute installierten WLANs keinerlei Sicherheitssysteme verwenden, also nicht einmal das schwache WEP einsetzen. Unter diesen Umständen stellen WLANs heute eine der größten Bedrohungen für die IT-Sicherheit der Firmen dar.

4.3.3. Andere Systeme

Neben den vorgestellten Systemen gibt es noch eine Reihe weiterer Technologien, die zum digitalen drahtlosen Datenaustausch eingesetzt werden können. Da diese sich jedoch nur schlecht für den Aufbau von MANETs eignen, werden sie hier nur kurz erwähnt.

Das *Digital European Cordless Telecommunication (DECT)* System wird, wie der Name schon andeutet, primär für schnurlose Telefone verwendet. Nichtsdestotrotz sind auch Datenanwendungen mit Bandbreite bis zu 552 kbps möglich, die sich unterschiedlich auf Hin- und Rückkanal verteilen lässt. Da jedoch die Existenz von zentralen Basisstationen vorausgesetzt wird, ist ein dezentrales MANET nur schlecht realisierbar. [Sik] und [Mül] geben einen Überblick über die DECT Technologie.

Der *HomeRF* Standard wurde 1998 von den Firmen Compaq, Hewlett-Packard, IBM, Intel und Microsoft ins Leben gerufen. Er kann als eine Art Kombination aus WLAN und DECT angesehen werden, der sowohl für Daten- wie Telefonieanwendungen geeignet ist. Während der ältere Standard V1.2 eine Datenrate von lediglich 1,6 Mbps spezifiziert, liefern die neueren Versionen 2.0 (von 2001) und 2.1 (geplant) jeweils 10 Mbps und 20 Mbps. HomeRF kann als direkter Konkurrent zu IEEE 802.11 gesehen werden, dem es auch architektonisch sehr ähnelt. HomeRF unterstützt aber die Übertragung von Multimediadaten und Sprache besser, was es als Nachfolger für DECT

Standards im Überblick

Netz	Frequenz	Reichweite	Geschwindigkeit	Einsatzgebiet	Eignung MANET
Bluetooth	2,4 GHz	10m (bis 100m)	max. 1 Mbps	Personal Area Networks	⊕
DECT	1880 - 1900 MHz	50 m in Gebäuden, 300 m im Freien	max 552 kbps	lokale Sprach- und Datendienste	⊖
HomeRF	2,4 GHz	50 - 100 m	10 Mbps	Heimvernetzung/ SOHO Netzwerke	⊕
HiperLAN/2	5 GHz	50 - 100 m	max. 54 Mbps	Wireless LAN	⊕⊕
GSM	900 / 1800 MHz	1 - 5 km	9,6 kbps (Kanalbündelung möglich)	Mobilfunk	⊖⊖
GPRS	900 / 1800 MHz	1 - 5 km	53 kbps	Datenmobilfunk	⊖
UMTS	1900 - 2000 MHz und 2100 - 2200 MHz	ähnlich GSM/GPRS	max. 2 Mbps	Daten, Sprache, Multimedia (geringe Datenrate)	⊖
802.11b	2.4 GHz	30 - 50 m in Gebäuden, 300 m im Freien	max. 11 Mbps	Wireless LAN	⊕⊕
802.11a	5 GHz	30 in Gebäuden, 150 m im Freien	max. 54 mbps	Wireless LAN	⊕⊕

Tabelle 4.1.: Mobilfunkstandards im Überblick

prädestiniert. Trotzdem bleibt der kommerzielle Erfolg dieser Technologie bisher weit hinter IEEE 802.11 zurück. Für eine Übersicht und weiterführende Verweise siehe [Mül].

Eine weitere Technologie für Wireless LANs ist *HiperLAN/2*, welches im Rahmen des europäischen Forschungsprogramms BRAN entstanden ist und vom *HiperLAN/2 Global Forum* [HIP] standardisiert wird. HiperLAN/2 arbeitet im 5 GHz Band und erreicht Datenraten bis 54 Mbps. Entsprechend der Zielsetzung der Forschungsprojekte entspricht HiperLAN/2 einer Art drahtlosem ATM, was für eine gute Unterstützung von Dienstegüte sorgt. HiperLAN/2 unterstützt auch einen Ad hoc Modus ohne zentralen Access Point und wäre somit auch für den Aufbau von MANETs geeignet. Mangels Marktverfügbarkeit entsprechender Geräte tritt es jedoch im Vergleich zu konkurrierenden Technologien wie IEEE 802.11a in den Hintergrund.

Schließlich sollen noch die zellulären Mobilfunknetze erwähnt werden, die auch eine Datenübertragung erlauben. Der ursprüngliche *GSM (Global System for Mobile Communication)* Standard sah lediglich eine verbindungsorientierte Datenübertragung mit 9,6 kbps vor. Dabei verwendet GSM eine feste Infrastruktur von Basisstationen, welche die zu versorgende Fläche in sogenannte Zellen einteilt. Eine direkte Kommunikation zwischen Teilnehmern ist nicht möglich, was diese Technologie zum Aufbau von MANETs disqualifiziert. Wegen der großen geographischen Verbreitung ist diese Technologie aber auch für MANETs interessant, die so beispielsweise eine Anbindung ans Internet realisieren können.

GSM arbeitet in Europa auf Frequenzbändern von 900 und 1800 MHz. Da der leitungsorientierte Transport von Daten nur schlecht zu den heutigen paketorientierten Netzen passt, wurden später Erweiterungen wie der *General Packet Radio Service (GPRS)* entwickelt. GPRS erlaubt paketorientierte Übertragungen in GSM Netzen bis 53 kbps. Diese Technologie soll durch das *Universal Mobile Telecommunication System (UMTS)* abgelöst werden, welches im Bereich um 2 GHz Datenraten von bis zu 2 Mbps liefern soll. Ausführliche Informationen zu GSM liefert beispielsweise [EVB01].

Tabelle 4.1 fasst noch einmal alle Technologien kurz zusammen und gibt eine Bewertung der Eignung als Basistechnologie für MANETs ab.

4.4. Fazit

Wie man sieht gibt es eine ganze Reihe unterschiedlicher Standards für drahtlose Datenkommunikation, die sich prinzipiell für den Aufbau von MANETs eignen. Im Idealfall sollte der Aufbau eines Multi-Hop Ad hoc Netzes vom verwendeten Funknetz unabhängig sein, ähnlich wie dies heute für das drahtgebundene Internet gilt. Hier spielt es auch keine große Rolle, ob die Daten über Glasfasern, via Ethernet oder ADSL transportiert werden. Obwohl viele Technologien über eigene Sicherheitslösungen verfügen, sind diese meist nicht für eine Ende-zu-Ende Absicherung im Rahmen eines IP Netzes gedacht. Vielmehr wird hier explizit das Schicht-2 Netz bspw. gegen Abhören gesichert. Für ein Mobiles Ad hoc Netzwerk sind die Mechanismen daher wenig geeignet.

Das nächste Kapitel widmet sich nun im Detail dem Aufbau von Ad hoc Netzen. Dabei wird sich zeigen, dass viele Aspekte doch nicht gänzlich unabhängig vom jeweiligen Funksystem betrachtet werden können, so dass ein grundlegendes Verständnis der drahtlosen Netzwerktechnik für die Realisierung von MANETs unabdingbar ist.

5. Mobile Ad hoc Netzwerke

5.1. Grundlagen und Anwendungen

Wie bereits in Kapitel 4 deutlich wurde, sind die meisten der heute existierenden drahtlosen Netzwerke hauptsächlich eine Verlängerung der drahtgebundenen Netze. Allerdings bieten viele Technologien wie Bluetooth, IEEE 802.11 oder HiperLAN/2 auch die heute wenig genutzte Möglichkeit, dass sich zwei Teilnehmer automatisch vernetzen, sobald sie sich in gegenseitiger Funkreichweite befinden. Im Idealfall ist hierzu keinerlei Eingriff des Benutzers und keine Infrastruktur notwendig, die Kommunikationsmöglichkeit entsteht völlig spontan. Bei IEEE 802.11 ist das im sogenannten Ad hoc Modus gegeben, Bluetooth verlangt hingegen eine explizite Inquiry und einen Verbindungsaufbau[1].

Dabei wird implizit eine drahtlose Kommunikationsform vorausgesetzt, da die Herstellung von Kabelverbindungen in keinem Fall als automatische Vernetzung gelten kann. Die gängigen Systeme nutzen hierzu Funkschnittstellen, weil Alternativen wie Infrarotlicht eine Reihe unerwünschter Eigenschaften aufweisen. Die direkte Ausbreitung des Infrarotlichts bedingt beispielsweise, dass Sender und Empfänger über eine Sichtverbindung verfügen müssen.

Definition 5.1 (Ad hoc Netzwerk) *Ein Ad hoc Netzwerk ist ein (funkbasiertes) Kommunikationsnetz aus zwei oder mehr Teilnehmern, welches sich spontan und ohne Benutzereingriff bildet, sobald sich die Teilnehmer in Reichweite befinden.*

Hierzu einige Anmerkungen. Die Etablierung eines solchen Netzes bedeutet einen deutlich höheren Aufwand als nur den Aufbau einer Funkverbindung zwischen den Knoten. Zunächst muss der Kommunikationspartner gefunden werden. Unter Umständen müssen auch Adressen automatisch vergeben und Sicherheitsprotokolle initiiert werden. Auch ist die in der Definition genannte Reichweite nicht notwendigerweise die Reichweite des eingesetzten Funksystems. Während die einfachen *Single-Hop Ad hoc Netzwerke* nur die Kommunikation zwischen Knoten erlauben, die sich direkt in Funkreichweite zueinander befinden, ermöglichen die *Multi-Hop Ad hoc Netzwerke* auch eine Kommunikation zwischen weiter entfernten Knoten, solange dazwischenliegende Knoten erreichbar sind, die den Datenverkehr weiterleiten.

Definition 5.2 (Single-Hop Ad hoc Netzwerk) *Ein Single-Hop Ad hoc Netzwerk ist ein Ad hoc Netzwerk, welches Knoten nur dann eine Kommunikation erlaubt, wenn sie sich in direkter Funkreichweite zueinander befinden.*

Definition 5.3 (Multi-Hop Ad hoc Netzwerk) *Ein Multi-Hop Ad hoc Netzwerk ist ein Ad hoc Netzwerk, in welchem Daten zweier Kommunikationspartner auch über Zwischenknoten weitergeleitet werden.*

[1] evtl. sogar mit Eingabe von PIN-Nummern zur Absicherung

Im englischen Sprachraum werden Ad hoc Netzwerke meist auch als *Mobile Ad hoc Networks* bezeichnet und mit *MANET* abgekürzt. Die Sprechweise unterscheidet sich hierbei. Während manche das *NET* in MANET wie im englischen „network“ sprechen, betonen andere das Wort wie den französischen Maler. Es ist auch nicht einheitlich geregelt, ob der Begriff Single- oder Multi-Hop Ad hoc Netzwerke bezeichnet. In dieser Arbeit wird die Abkürzung MANET ausschließlich für Multi-Hop Netzwerke verwendet.

Definition 5.4 (MANET) *Ein MANET ist ein Multi-Hop Ad hoc Netzwerk.*

Die beiden Bücher [Per01] und [Toh02] liefern eine gute Einführung in die vielfältigen Aspekte von MANETs. Für die Zwecke dieser Arbeit sind vor allem die möglichen Einsatzszenarien und die technische Realisierung des Routings von Interesse. Andere Aspekte wie Interoperabilität mit dem Internet, Verhalten von TCP in MANETs [SSS02] oder Aufbau von Applikationen werden hingegen nicht betrachtet.

5.1.1. Geschichte

Wie in Kapitel 4 schon erwähnt, entstanden die ersten Ideen für MANETs bereits sehr früh. Der vermutlich erste Vertreter eines MANET war das *Packet Radio Network (PRNET)* [K+78], in welchem seit 1972 die Grundlagen der gemeinsamen Nutzung von Radiofrequenzen durch ein paketorientiertes Multi-Hop Netzwerk erforscht wurden. Die Ergebnissen hatten auch Auswirkungen auf andere Vernetzungstechniken mit gemeinsamem Kanalzugriff (wie Satelliten-Netze oder Ethernet [LNT87b]).

Allerdings waren diese frühen Netze der 70er Jahre durch verschiedene Faktoren limitiert: Die Geräte waren, bedingt durch den geringen Integrationsgrad, sehr groß und verbrauchten viel Energie, so dass bestenfalls an den Einsatz in Autos zu denken war. Der Datendurchsatz war, gemessen an den heutigen Leistungen, mit bestenfalls einigen hundert kbps sehr gering. Auch waren die Routingalgorithmen noch sehr einfach und die Skalierbarkeit und Robustheit damit relativ gering [LNT87a].

Aus diesem Grund wurde in den 80er Jahren von der DARPA das SURAN Projekt (*Survivable Radio Networks*) ins Leben gerufen [SW87]. Die Ziele waren, kleinere und leistungsfähigere Radiomodule zu entwickeln, sowie die Skalierbarkeit der Routingalgorithmen und die Störanfälligkeit,besonders gegen bewusste Störversuche von außen, zu verbessern. Auch innerhalb des amerikanischen Militärs selbst gab es verschiedene Forschungsarbeiten, so bei der Army [LKG86, Fra86, Sas99], der Navy [EWB87, Bak97] und der Air Force [Fra86]. Auch außerhalb der USA gab es in dieser Zeit militärische Forschungsprojekte zu selbstorganisierenden Packet-Radio Netzen, beispielsweise in England [BHGV87].

In den 90er Jahren hat sich vor allem die zur Verfügung stehende Technologie dramatisch weiterentwickelt. Kleine und leistungsfähige mobile Geräte sind für jedermann zugänglich, Wireless LAN nach IEEE 802.11 erlaubt den kostengünstigen Aufbau auch größerer MANETs und die Bandbreite der Netze und Batterieleistung ist zumindest stetig gestiegen. Militärische Entwicklungen wie *Spread-Spectrum* fanden Eingang in kommerzielle Produkte. Auch das Militär ist nach wie vor stark an einer Weiterentwicklung der MANET Technologie interessiert, wie eine Vielzahl von Projekten dokumentiert: das „US Army’s Task Force XXI Advanced Warfighting Experiment (AWE)“,

die „U.S. Navy and Marines' Extending the Litoral Battlespace (ELB) Advanced Concept Technology Demonstration (ACTD)“ [Alt99] oder das „DARPA Global Mobile (GloMo) Information Systems Program“ [LRS96, Rut98].

Neben diesen Forschungsprojekten engagiert sich das Militär aber auch immer mehr in öffentlichen Forschungs- und Standardisierungsbemühungen, vor allem seitdem in der „Joint Technical Architecture (JTA)“ [JTA97] festgelegt wurde, dass auch im militärischen Bereich kommerzielle und offene Standards zum Einsatz kommen sollen. Aus diesem Grund waren auch militärische Stellen beteiligt, als im August 1997 die *IETF MANET Working Group (MANET WG)* [MAN] gegründet wurde. Ziel dieser Arbeitsgruppe ist die Entwicklung eines oder mehrerer Standard-Routingprotokolle für Ad hoc Netzwerke. In dieser Funktion entwickelte sie sich schnell zum zentralen Gremium aller Forschungsaktivitäten rund um MANETs. Insbesondere werden sämtliche heute relevanten Routingprotokolle in dieser WG weiterentwickelt. Wie sich gezeigt hat, ist das Forschungsgebiet MANET sehr vielfältig und die Anforderungen an ein Routingprotokoll sind so heterogen, dass sich noch kein Standard für ein Routingprotokoll abzeichnet. Nachdem zunächst mehrere dutzend Protokolle als Drafts entwickelt und diskutiert wurden, gab es in jüngster Zeit eine deutliche Umstellung. Die IETF Working Group soll sich darauf konzentrieren, die Standardisierung der Protokolle AODV, DSR, OLSR und TBRPF zum Abschluss zu bringen. Forschungsaktivitäten rund um die anderen Protokolle und insbesondere im Bereich der Skalierbarkeit sollen zukünftig in die neu gegründete *IRTF RRG Ad hoc Network Scaling Research Subgroup* der *IRTF Routing Research Group* eingebracht werden [ANS].

5.1.2. Anwendungsszenarien

Bevor man sich mit technischen Details von MANETs beschäftigt, erscheint es sinnvoll, zunächst nach möglichen Anwendungen zu fragen, welche diese Netze nutzen können. Wie schon erwähnt, waren ursprünglich vor allem militärische Anwender an der Nutzung von MANETs interessiert. Die Gründe sind naheliegend: in einem feindlichen Kampfgebiet kann man schlecht eine Kommunikations-Infrastruktur installieren, denn diese ist ständig in Gefahr, durch feindliche Angriffe zerstört zu werden. Ein dynamisches, selbstorganisierendes Netz, welches ohne Infrastruktur auskommt und auch den Ausfall von Knoten toleriert, bietet hier optimale Eigenschaften. Pressemeldungen zu Folge waren mit MANET-Technologie ausgerüstete Armee-Einheiten bereits im Afghanistan-Krieg im Einsatz [USA02].

Ein ähnliches Szenario im zivilen Umfeld bieten Rettungseinsätze im Katastrophenfall. Wenn nach einem schweren Erdbeben, nach Vulkanausbrüchen oder Lawinenabgängen die herkömmliche Kommunikationsinfrastruktur zerstört ist, könnten Rettungskräfte mit MANET Systemen ausgerüstet werden, um schnell ein Kommunikationsnetz zu realisieren, über welches alle Beteiligten Status-Informationen austauschen können.

Die naheliegendste Idee einer alltäglichen Nutzung von MANETs ist die Erhöhung der Reichweite eines Access Points durch ein MANET. Hierzu leiten Knoten, die sich in Reichweite des APs finden, den Verkehr für entferntere Knoten weiter. Abbildung 5.1 zeigt dieses Szenario. Wie man sieht, kann es durchaus vorkommen, dass ein Knoten außerhalb der AP Reichweite nicht versorgt werden kann, weil sich kein anderer Knoten in einer günstigen Position zur Weiterleitung von Daten befindet. Auch können

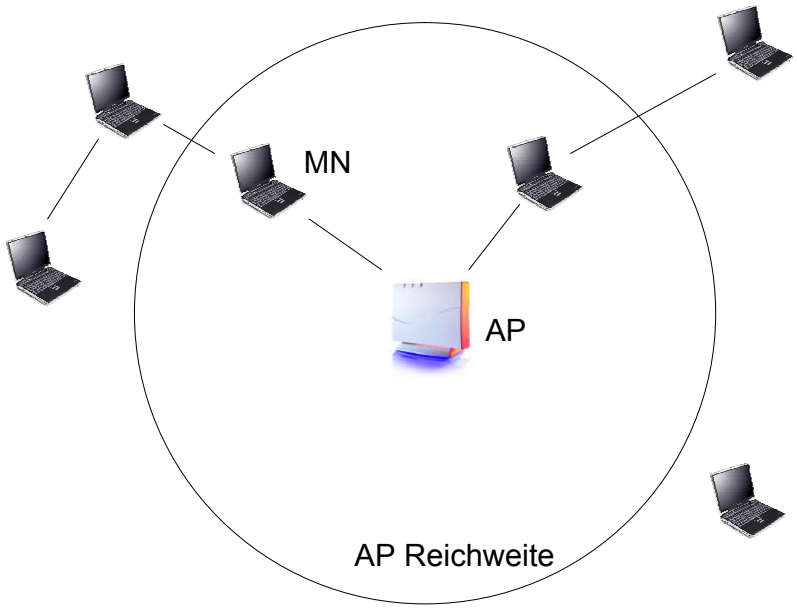

Abbildung 5.1.: Einsatz von MANETs zur Reichweitenerhöhung eines APs

unter Umständen mehrere Zwischenstationen (*Hops*) notwendig sein, um einen anderen Knoten zu erreichen.

Es sei noch angemerkt, dass die spektrale Effizienz und somit der Gesamtdurchsatz des Netzes durch *Space Division Multiple Access (SDMA)* erhöht werden kann, wenn die Knoten ihre Sendestärke so regeln können, dass der Empfangsknoten eines Datenpaketes das Paket gerade noch empfangen kann. Somit können MANETs auch ein Mittel zur Erhöhung der Leistungsfähigkeit von Funknetzen sein.

Die Anwendungen in einem solchen Szenario entsprechen den klassischen Internet-Anwendungen - Email, Web, Filetransfer usw. Es sind jedoch auch ganz andere Applikationen denkbar, bei welchen MANETs komplett ohne klassische Infrastruktur eingesetzt werden können.

In [KRSW03] und [Rib02] haben wir beschrieben, wie sich mit Bluetooth ausgerüstete Mobiltelefone spontan zu einem MANET verbinden können und wie Telefongespräche statt über die teure GSM-Infrastruktur kostenlos via Bluetooth geführt werden. Hierzu wurde ein speziell an Bluetooth angepasstes Routing Protokoll namens *Bluetooth Scatternet Routing (BSR)* entwickelt, welches später noch vorgestellt wird. Abbildung 5.2 zeigt ein typisches Szenario, bei dem zwei Personen in einem Bürogebäude ein BSR-vermitteltes Telefonat führen, welches über weitere dazwischen liegende Mobiltelefone vermittelt wird.

Es existiert eine Vielzahl von weiteren Anwendungsszenarien im Bereich Ubiquitous Computing, bei denen sich spontan Gruppen von Benutzern zusammenschließen und neuartige Anwendungen einsetzen. So baut das *UbiBay* System der Uni Trier im Ad hoc Netz eine Auktionsplattform auf, bei der man mit benachbarten Personen automatisiert Auktionen abhalten kann [FLS02]. Weitere derartige Anwendungen werden im Forschungsschwerpunkt „Basissoftware für selbstorganisierende Infrastrukturen für

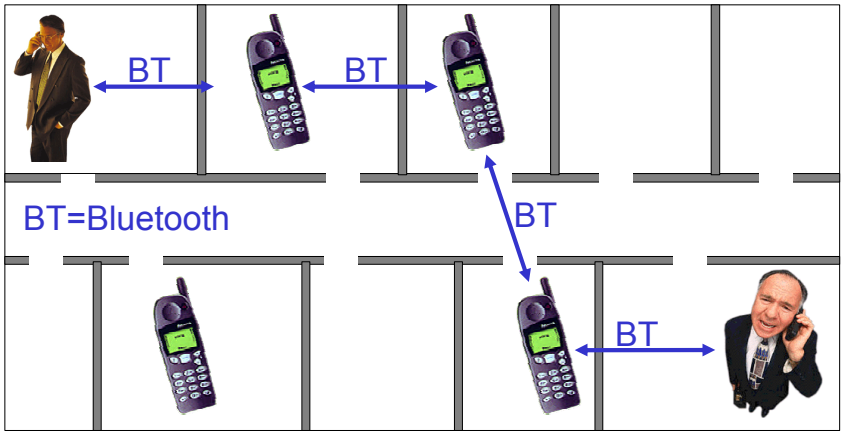

Abbildung 5.2.: Ein typisches Szenario für den Einsatz von Bluetooth Scatternet Routing (BSR)

vernetzte mobile Systeme“ der Deutschen Forschungsgesellschaft (DFG) entwickelt. Wir haben mit dem *SmartReminder* [KDIW02, KID+02] ein System zur persönlichen Assistenz realisiert, welches bei einem spontanen Treffen zweier Personen nützliche Hinweise und Erinnerung zum Gegenüber präsentiert. Hierzu schließen sich die mobilen Rechner beider Personen kurzzeitig zu einem (allerdings bisher nur Single-Hop) Ad hoc Netzwerk zusammen. Weitere große Projekte mit Bezug zu MANETs sind beispielsweise Fleetnet [Fle] oder IPonAir [IPo].

Ein spezieller Anwendungsfall für MANETs sind sogenannte *Sensor Networks* [Cal03]. Die Idee hierbei ist es, eine große Anzahl sehr kleiner und autarker Sensor-Module im Einsatzgebiet auszubringen. Das können installierte Geräte in einem Gebäude sein ebenso wie kleine Platinen, die von einem Flugzeug in großer Stückzahl über einem Waldgebiet abgeworfen werden. Dort könnten sie dann beispielsweise Umweltmesswerte mit hoher geographischer Auflösung erfassen. Da die Module nur über sehr beschränkte Energieressourcen verfügen und auch keine Sender mit großer Reichweite besitzen, bilden sie ein MANET, um die Daten z.B. zu einer Forschungsstation am Waldrand zu transportieren. Unterwegs können die Knoten bereits Berechnungen wie Mittelwertbildung auf den Daten durchführen und so die zu übertragende Datenmenge reduzieren.

Wie man sieht, können Mobile Ad hoc Netze für sehr unterschiedliche Anwendungsbereiche genutzt werden. Die konkrete Anwendung hat jedoch signifikante Auswirkungen auf die Sicherheit eines solchen Systems. Wie in Abbildung 5.3 gezeigt, ist zunächst zu unterscheiden, ob es sich um ein öffentliches oder ein geschlossenes MANET handelt. Ein geschlossenes MANET sei wie folgt definiert:

Definition 5.5 (Geschlossenes MANET) *Ein geschlossenes MANET ist ein Mobiles Ad hoc Netzwerk, welches von einem genau definierten Benutzerkreis gebildet und ausschließlich von diesem genutzt wird. Der Zweck der Kooperation ist in der Regel genau bekannt und wird von allen unterstützt, gegenseitige Angriffe kommen nicht vor.*

Geschlossene Systeme wird man beispielsweise innerhalb von Firmen finden. Diese sollen nur den Mitarbeitern zugänglich sein und man kann davon ausgehen, dass die Mitarbeiter sich gegenseitig unterstützen und vertrauen. Sicherheitsmaßnahmen beschränken sich dann darauf, fremde Benutzer und Knoten aus dem Netz fernzuhalten.

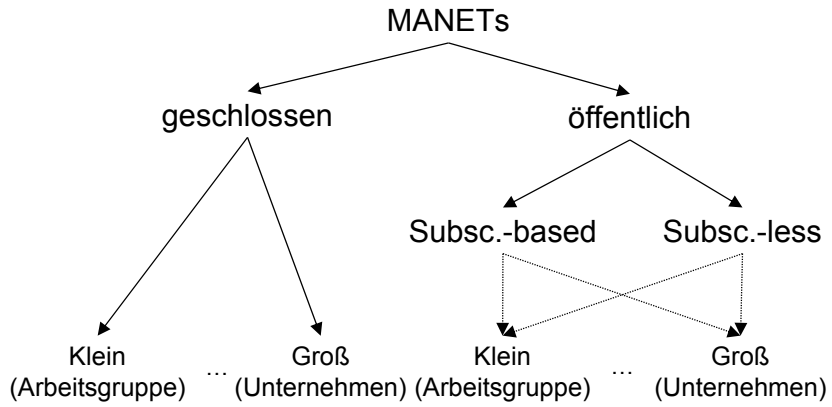

Abbildung 5.3.: Kategorisierung verschiedener Typen von MANETs

Die Verteilung von gegenseitig bekannten Schlüsseln kann relativ einfach organisiert werden. Damit treten viele der später beschriebenen Sicherheitsprobleme gar nicht auf. Allerdings muss man im schlimmsten Fall trotzdem davon ausgehen, dass beispielsweise ein Notebook gestohlen wird und die darauf hinterlegten Schlüssel in die Hände eines Angreifers gelangen. Es sollte also Möglichkeiten geben, diesen Benutzer bzw. dessen Rechner in einem solchen Fall aus dem Netz auszuschließen.

Definition 5.6 (Öffentliches MANET) *Ein öffentliches MANET gewährt prinzipiell jedem Zugang zum Netz und seinen Ressourcen. Die Benutzer kennen sich nicht und verfolgen nicht notwendigerweise gemeinsame Ziele. Es muss damit gerechnet werden, dass einzelne Benutzer die Funktionsfähigkeit des Netzes stören wollen.*

Bewegt man sich in einem öffentlichen MANET, so sind die Bedingungen schwieriger. Bei Anwendungen, wie dem oben beschrieben UbiBay oder der MANET-Telefonie, kennen sich die Benutzer normalerweise nicht gegenseitig. Somit besteht zunächst kein Grund zur Kooperation und zum gegenseitigen Vertrauen. Jedoch entsteht die Ressource MANET erst durch die Kooperation der Benutzer und somit sollte jeder, der diese Ressource selbst nutzen will, ebenfalls zum Aufbau des MANETs beitragen. Man muss aber damit rechnen, dass einzelne egoistische Knoten ausschließlich Ressourcen konsumieren, ohne selbst Datenverkehr von anderen weiterzuleiten. Wie wir später zeigen werden, leidet die Gesamtleistungsfähigkeit des MANETs darunter zunächst nur wenig. Erst bei einer größeren Anzahl solcher Knoten kommt es zu starken Einbrüchen. Darüber muss man auch mit böswillige Angreifern im Netz rechnen, welche gar nicht an einer reibungslosen Funktion des Netzes interessiert sind, sondern das Netzwerk gezielt stören oder die Daten der Teilnehmer abhören wollen.

Im Hinblick auf die zu realisierenden Sicherheitsmaßnahmen muss man bei öffentlichen MANETs zwischen *subscription-based* und *subscription-less* MANETs unterscheiden. Im ersten Fall muss ein Benutzer zunächst eine, wie auch immer geartete, Anmeldeprozedur bei einem Netz-Koordinator durchlaufen, bevor er am MANET teilnehmen kann. Im Rahmen dieser Anmeldung könnten beispielsweise die Identitäten der Benutzer festgestellt und kryptographische Schlüssel ausgetauscht werden. Der zentrale Netzkoordinator ist in diesem Fall eine Art Trusted-Third-Party, welcher alle Netzteilnehmer vertrauen. Er kann eine faire Netznutzung regeln. Im Gegensatz dazu kommen

subscription-less MANETs ohne einen Anmeldeprozess oder eine zentrale Instanz aus. Ein beliebiger Benutzer kann also sofort am MANET teilnehmen. Damit realisiert diese Art die reinste Form eines MANET, hier wird wirklich ein *Ad hoc* Netzwerk aufgebaut. Unter Sicherheitsgesichtspunkten ist ein solches Netz natürlich sehr kritisch zu bewerten. Die Teilnehmer können eigentlich nur aus ihren Beobachtungen der anderen Knoten Schlüsse über deren Kooperation oder Nicht-Kooperation ziehen.

Neben diesen Einteilungen spielt noch die Größe des Netzes eine Rolle. Bei geringen Teilnehmerzahlen sind natürlich Lösungen wie ein manueller Austausch von Schlüsseln oder Passwörtern denkbar, die bei größeren Netzen nicht mehr in Frage kommen.

Der Rest dieser Arbeit konzentriert sich auf den komplexesten Fall der *öffentlichen subscription-less MANETs*. Dazu sollen zunächst die Rahmenbedingungen noch etwas genauer definiert werden. Das zu Grunde liegende Szenario für unsere Sicherheitsinfrastruktur lässt sich wie folgt beschreiben:

1. Es handelt sich um ein *öffentliches subscription-less Ad hoc Netzwerk*, d.h. jeder kann ohne vorherige Anmeldung teilnehmen.
2. Es kommt ein *gängiges MANET Routing Protokoll* zum Einsatz (z.B. DSR, AODV, OLSR). Zu Demonstrationszwecken wird DSR verwendet. Dieses wird um geeignete Sicherheitsfunktionen erweitern.
3. Im Netz werden *Peer-to-Peer-Dienste* oder *Client/Server Anwendungen* verwendet. Weiterhin dient das Netz zum Zugriff auf das öffentliche Internet über entsprechende Gateways.
4. Der *Datenschutz* soll technisch unterstützt werden. Daher soll Teilnehmern die Identität und/oder Lokation von anderen Teilnehmern nicht ohne weiteres zugänglich sein. Dies gilt insbesondere für die Erstellung von Bewegungsprofilen.
5. Die Betrachtungen konzentrieren sich auf die *Netzwerkschicht*. Aspekte der Absicherung von Betriebssystemen oder Anwendungen sind kein Bestandteil der Analyse.

5.2. MANET Routing

5.2.1. Routing

Jedes Netzwerk kann als ein gerichteter oder ungerichteter Graph $\mathcal{G}(\mathcal{N}, \mathcal{E})$ mit den Knoten $\mathcal{N}$ und den Kanten $\mathcal{E} = \{(n_1, n_2) | n_1 \neq n_2 \wedge n_1, n_2 \in \mathcal{N}\}$ betrachtet werden. Die Knoten entsprechen den Rechnern im Netzwerk, eine Kante – auch *Link* genannt – repräsentiert eine Kommunikationsverbindung zwischen zwei Knoten. Können in einem Netzwerk unidirektionale Verbindungen auftreten, so modelliert man das Netz sinnvollerweise als gerichteten Graph, gibt es ausschließlich bidirektionale Verbindungen, so genügt ein ungerichteter Graph. Im Falle eines Funknetzes werden zwei Knoten N_1 und N_2 genau dann mit einer Kante $e = (N_1, N_2) \in \mathcal{E}$ verknüpft, wenn N_1 und N_2 sich in Kommunikationsreichweite zueinander befinden. Die Kanten können dabei auch gewichtet sein, d.h. $g(e) = x$ gibt das Kantengewicht einer Kante e an. Im einfachsten Fall ist das Gewicht jeder Kante 1, es sind jedoch auch komplexere Gewichtungsfunktionen denkbar, die z.B. die verfügbare Bandbreite mit in Betracht ziehen.

Als *Routing* bezeichnet man den Transport einer Nachricht von Knoten S zu Knoten D über die dazwischenliegenden Knoten $(R_1, R_2, \ldots, R_n)$. Es ist Aufgabe des *Routingverfahrens*, die Route $\mathcal{R} = \{e_i | i = 1 \ldots n-1 \wedge e_i = (R_i, R_{i+1})\}$ so zu wählen, dass $\sum_i g(e_i)\ \forall\, i \,|\, e_i \in \mathcal{R}$ minimal wird. Dies ist auch als das *Shortest Path* Problem bekannt [SS80].

Es existieren effiziente Algorithmen, um in einem gegebenen Graphen eine derartige kürzeste Route zu finden. Am Bekanntesten ist der *Shortest Path First (SPF)* Algorithmus von E.J. Dijkstra aus dem Jahr 1959 [Dij59]. Es gibt zwei grundsätzlich unterschiedliche Herangehensweisen an das Routing. Im einen Fall wird die Topologie-Information des Graphen komplett zwischen allen Knoten ausgetauscht und anschließend führt jeder Knoten unabhängig vom Rest eine Routenberechnung z.B. mittels SPF durch. Dieser Ansatz wird *Link-State-Routing* genannt und ist beispielsweise im OSPF Protokoll realisiert [Moy91, Hui00]. Alternativ kann die Berechnung auch verteilt im Netz erfolgen, so dass nie ein einzelner Knoten eine komplette Sicht auf das Netz hat. Bellman [Bel58], Ford und Fulkerson [FF62] waren wesentlich an der Entwicklung des nach ihnen benannten Verfahrens beteiligt, auf welchem Protokolle wie RIP aufbauen [Hed88, Hui00]. Beide Verfahren müssen das Problem lösen, dass ein Netzwerkgraph nie statisch ist, sondern sich ständig ändert, indem entweder alte Verbindungen wegfallen oder neue aufgebaut werden. Dabei können temporäre Inkonsistenzen entstehen, die zum Beispiel zu Schleifen, sogenannten *Routing Loops* führen [MRR80]. Wie derartige Probleme gelöst werden, soll für den allgemeinen Fall hier jedoch nicht weiter erörtert werden. Interessierte Leser seien an [Hui00] verwiesen.

5.2.2. Anforderungen an MANET Routing

Besonders schwer wiegt das vorher genannte Probleme der ständig wechselnden Netz-Topologie natürlich in MANETs, welche ihre Topologie, bedingt durch die mobilen Teilnehmer, mit weit höherer Geschwindigkeit ändern, als dies herkömmliche drahtgebundene Netzwerke tun. Aber auch andere Eigenschaften dieser Netze machen das Routing in MANETs zu einer besonderen Herausforderung. Bevor im nächsten Abschnitt konkreter auf Routingverfahren für MANETs eingegangen wird, sollen zunächst die wesentlichen Eigenschaften und Anforderungen an ein solches Routing Protokoll für Ad hoc Netzwerke genauer betrachtet werden. Die IETF MANET WG hat diese in RFC 2501 [CM99] zusammengestellt:

Häufige/komplexe Topologiewechsel: Bedingt durch die Mobilität der Benutzer entstehen in einem Ad hoc Netz ständig neue Verbindungen, wenn sich zwei Teilnehmer auf Funkreichweite nähern. Umgekehrt verschwinden bestehende Verbindungen, wenn sich zwei Knoten zu sehr voneinander entfernen. Dabei sind diese Link-Wechsel *zufällig* und nicht vorhersehbar, da auch die Bewegungen der beteiligten Kommunikationspartner kaum vorherzusehen sind. In besonderen Szenarien (z.B. Fahrzeuge auf einer Straße [Fle]) mag allerdings eine begrenzte Vorhersagbarkeit möglich sein. Weiterhin gibt es in manchen Funktechnologien die Möglichkeit von *unidirektionalen Links*, d.h. Verbindungen, bei denen zwar A eine Nachricht an B schicken kann, aber nicht umgekehrt.

Geringe Bandbreite: Wie wir in Kapitel 4 gesehen haben, ist die über eine elektromagnetische Welle übertragbare Datenmenge pro Zeiteinheit grundsätzlich beschränkt. Deshalb wird die Bandbreite in Funknetzen, verglichen mit drahtgebun-

denen Netzen, trotz aller Anstrengungen immer geringer sein. Momentan hinken die Funknetze mit Datenraten um die 50 Mbps den drahtgebundenen Netzen wie 10 Gbps Ethernet etwa um den Faktor 200 hinterher. Bei gleichen Anforderungen durch die Anwendungen ist also die Datenrate eine knappe Ressource, die entsprechend sparsam einzusetzen ist.

Geringe Leistungsfähigkeit der Komponenten: In einem mobilen Szenario kommen sinnvollerweise auch kleine, mobile Geräte zum Einsatz. Egal ob es sich hierbei um PDAs, Mobiltelefone oder Notebooks handelt, die zur Verfügung stehende Prozessorleistung und Speicherkapazität wird nur selten mit einem Desktop-PC konkurrieren können. Notwendig wird diese Beschränkung primär deshalb, weil mobile Geräte nicht zu groß und zu schwer werden dürfen und somit auch die Energiekapazität in Form von Batterien oder Akkus beschränkt ist.

Eingeschränkte physikalische Sicherheit: Vom Standpunkt der Sicherheit gesehen ist dieser Aspekt besonders relevant. Während ein klassischer IBM Mainframe nur mit erheblichem Aufwand aus einem gesicherten Rechenzentrum gestohlen werden kann, besteht bei einem PDA oder Mobiltelefon viel eher die Gefahr, dass diese durch Diebstahl abhanden kommen oder schlicht verloren werden. Enthaltene sicherheitskritische Daten – insbesondere kryptographische Schlüssel – gehen dann natürlich auch verloren.

Aus diesen Eigenschaften leitet die MANET WG in RFC2501 [CM99] diverse Anforderungen an ein Routing-Protokoll für MANETs ab.

Verteilte Arbeitsweise: Ein zentralisiertes Routingverfahren ist nicht praktikabel. Daher muss ein verteilter Algorithmus zum Einsatz kommen, der nicht auf der Erreichbarkeit von irgendeinem zentralen Knoten beruht.

Schleifenfreiheit: Um nicht unnötig Netzkapazität zu verbrauchen, muss das Routing-Protokoll in jedem Fall Routen ohne Schleifen berechnen.

Reaktive vs. proaktive Arbeitsweise: RFC 2501 unterscheidet zwei Arten von Routingprotokollen, nämlich reaktive und proaktive. Auf diese Unterscheidung wird später in einem eigenen Abschnitt eingegangen.

Sicherheit: Schon sehr früh war der Working Group klar, dass Sicherheit ein zentraler Aspekt für den praktischen Einsatz von MANETs sein würde. Entsprechend soll ein Routing Protokoll wenig Angriffsfläche für böswillige Attacken bieten.

´Sleep´ Period Operation: Hierunter versteht man die Möglichkeit, dass sich ein Knoten bei Bedarf für eine gewisse Zeit in einen Energiesparmodus begeben kann, ohne dass hierunter die Funktionsfähigkeit des MANETs leidet.

Unterstützung unidirektionaler Links: Das Routingprotokoll sollte nach Möglichkeit mit der Situation umgehen können, dass in einem MANET auch unidirektionale Links auftreten.

5.2.3. Positionierung im OSI Schichtenmodell

Eine grundsätzliche Frage, welche vor allem zu Beginn der IETF-Arbeit für viele Diskussionen gesorgt hat, war, ob ein solches Routingprotokoll für MANETs auf *Schicht*

zwei oder *Schicht drei* des OSI Schichtenmodells zu positionieren sei, oder, mit der Nomenklatur der IETF gesprochen, ob es im *Physical Layer* oder im *IP Layer* anzusiedeln sei. Für den IP Layer spricht, dass Routing grundsätzlich eine Funktion der Netzwerkschicht ist und somit besser hier positioniert wird. Außerdem kann auf dieser Ebene von Details der verwendeten Funknetztechnologie (z.B. dem Adressformat) abstrahiert werden. Damit können gleichzeitig unterschiedliche Funknetztechnologien in einem MANET unterstützt werden. Ein Nachteil ist, dass in einem MANET sehr viele IP Host-Routen auftauchen und eine klare Trennung in IP-Subnetze nicht so intuitiv möglich ist, wie bei drahtgebundenen Netzen. Siedelt man das MANET hingegen auf dem Physical Layer an, dann entspricht das gesamte MANET aus Sicht von IP einem Layer-2 Netz, also beispielsweise einem Ethernet-Segment. Die Verteilung von IP-Adressen und die Anbindung an das Internet kann in diesem Fall ganz genauso erfolgen, wie dies auch in einem Ethernet geschieht. Ein Nachteil ist allerdings, dass IP ohne Kenntnis des darunter liegenden MANETs viele Subnetz-Broadcasts auslöst (z.B. für ARP) und somit das Netz häufig mit Daten geflutet werden muss.

Die Diskussion um die korrekte Ansiedlung ist heute weitgehend zugunsten von IP entschieden, lediglich bei Sensor Networks verzichtet man manchmal aus Effizienzgründen auf IP. Die auftretenden Probleme wurden durch entsprechende Forschungsarbeiten ansatzweise gelöst [WMP+02, BRSP01, PMW+01].

5.2.4. Proaktiv vs. Reaktiv

Die prominentesten Vertreter für MANET Routingprotokolle lassen sich in zwei unterschiedliche Lager einteilen. Die sogenannten *proaktiven* oder *table-driven* Protokolle wie OLSR [ACJ+03] und DSDV [PB94] versuchen, ständig aktuelle Routen für das gesamte MANET zu unterhalten. Dies entspricht der Vorgehensweise in klassischen IP Netzen. Im Gegensatz dazu werden die *reaktiven* oder *on-demand* Protokolle erst bei Bedarf aktiv. Schickt ein Knoten A ein Paket an Knoten B, so führt das Routing Protokoll zunächst eine Pfadsuche (engl. *Path-* oder *Route-Discovery*) durch. Erst wenn ein gültiger Pfad gefunden wurde, kann das Paket verschickt werden. Dieser Vorgang benötigt natürlich einige Zeit, so dass es bei typischen on-demand Protokollen wie DSR [JMHJ03] oder AODV [PRD03] zu Beginn einer Verbindung zu einer Verzögerung kommt. Dafür gehen on-demand Protokolle in der Regel sparsamer mit der Bandbreite um und erzeugen weniger Routing-Overhead.

Im Gegensatz dazu steht bei proaktiven Protokollen die Route sofort zur Verfügung, unter Umständen aber um den Preis von viel unnötiger Kommunikation durch das Routing Protokoll. Wie hoch dieser Overhead in der Praxis wirklich ist, hängt von einer Vielzahl von Faktoren ab, z.B. der Anzahl der Knoten im Netz, deren Bewegungsmustern, dem Kommunikationsverhalten uvm. Somit lässt sich keine allgemeingültige Antwort auf die Frage geben, welche Protokolle nun besser oder schlechter sind. Abhängig vom Einsatzszenario arbeiten mal die einen und mal die anderen effizienter. Untersuchungen zeigen, dass tendenziell die proaktiven Protokolle in dichtbevölkerten Netzen mit starker Kommunikation zwischen vielen unterschiedlichen Partnern besser geeignet sind, wohingegen die reaktiven Protokolle in dünnbesetzten Netzen mit Kommunikation zwischen wenigen Teilnehmern vorzuziehen sind [JLH+99, Qua00].

5.3. MANET Routing Protokolle

In diesem Abschnitt sollen exemplarisch einige MANET Routing Protokolle etwas detaillierter vorgestellt werden. Am Anfang steht mit OLSR ein Vertreter der proaktiven Protokolle, gefolgt von den beiden verbreitetsten reaktiven Protokollen AODV und DSR. Im Anschluss folgt ein Protokoll, welches wir für einen speziellen Einsatzzweck, nämlich das Routing in Bluetooth-basierten Netzen, entwickelt haben: das *Bluetooth Scatternet Routing (BSR)* [KRSW03]. Am Ende folgt eine Übersicht über weitere Ansätze und Protokolle.

5.3.1. Optimized Link-State Routing

Optimized Link-State Routing (OLSR) [CJL+01, ACJ+03] ist ein typischer Vertreter der proaktiven Protokolle. Es lehnt sich eng an klassische Link-State Protokolle wie OSPF [Hui00] an. Ein Link-State Routing Protokoll gliedert sich normalerweise in zwei Teile: zum einen die Verteilung der Topologie-Information, insbesondere der Änderungen im laufenden Betrieb, und zum anderen der lokalen Berechnung von Routen basierend auf diesen Daten. Für den letzten Schritt kann ein beliebiger Algorithmus eingesetzt werden, welcher das Shortest-Path Problem in Graphen löst, beispielsweise der schon erwähnten SPF Algorithmus von Dijkstra [Dij59]. Hier unterscheidet sich OLSR nicht wesentlich von den herkömmlichen Protokollen wie OSPF. Der Schwerpunkt von OLSR liegt auf der Optimierung der Topologie-Verteilung in MANETs.

Hierzu verwendet OLSR das Konzept der *Multipoint-Relays (MPR)*. Aus der sogenannten *One Hop Neighborhood* wird zunächst eine minimale Menge von Knoten (das *MPR Set*) ausgewählt, über welche alle Knoten erreicht werden können, die vom aktuellen

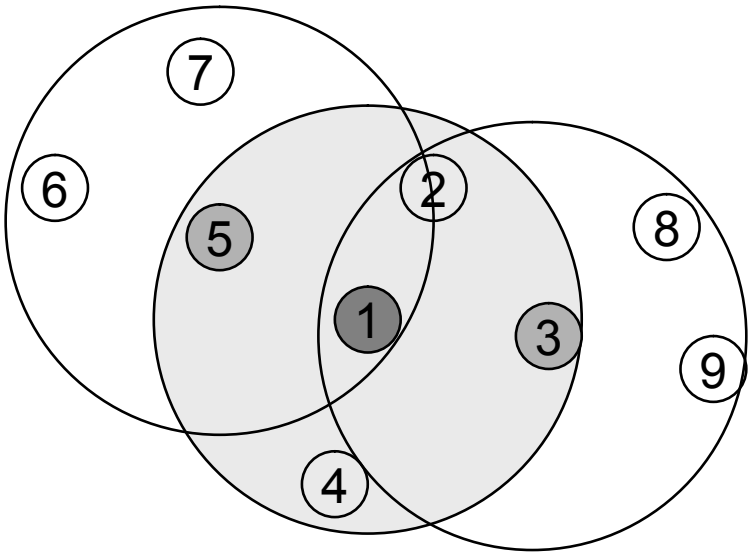

Abbildung 5.4.: OLSR Multipoint Relays

Knoten zwei Hops entfernt sind. In Abbildung 5.4 bilden die Knoten 2, 3, 4 und 5 die One Hop Neighborhood von Knoten 1, die Knoten 6 bis 9 sind Two Hop Neighbors. Will 1 nun eine Nachricht (z.B. ein Link-Update) durchs Netz fluten, so genügt es, wenn 3 und 5 diese weiterleiten, da hiermit ja bereits alle Two Hop Neighbors erreicht werden. Leitet auch Knoten 2 die Nachricht weiter, so wird damit kein zusätzlicher Knoten erreicht, es entsteht nur unnötiger Netzverkehr. Bei OLSR erfährt der Knoten 1 über seine One Hop Neighborhood, welche Two Hop Neighbors über welchen One Hop Neighbor zu erreichen sind. Daraus bildet er dann das MPR Set und informiert diese Knoten darüber, dass er sie als MPR betrachtet. Für diese Knoten ist 1 dann ein sogenannter *Multipoint Relay Selector*.

Mit dieser Methode wird die Zahl der Nachrichten, die verschickt werden, wenn eine Information im Netz geflutet werden soll, deutlich reduziert. Außerdem verteilt OLSR nur Informationen über Verbindungen zu MPR Selektoren im Netz, so dass insgesamt die Topologie optimiert wird. Mit diesen Eigenschaften eignet sich OLSR vor allem für dicht besetzte Netzwerke mit sehr engen Kommunikationsbeziehungen zwischen vielen Knoten.

5.3.2. Ad Hoc On-Demand Distance-Vector Routing

Das *Ad Hoc On-Demand Distance-Vector Routing (AODV)* [PR99, PRD03] lehnt sich inhaltlich an die Distance-Vector Protokolle wie RIP [Hed88, Hui00] an. Die Routenberechnung erfolgt also nicht ausgehend von einer kompletten Topologie-Information, sondern verteilt im Netz. Im Gegensatz zum Vorläufer DSDV [PB94] ist AODV allerdings kein proaktives Protokoll, vielmehr arbeitet es reaktiv. Will ein Knoten ein Datenpaket an einen Empfänger verschicken, zu dem noch keine Route existiert, so startet er eine *Route Discovery*. Hierzu wird ein *Route Request (RREQ)* im Netz geflutet (Abbildung 5.5 a). Jeder RREQ beinhaltet die Adresse des Absenders, eine für den Absender eindeutige Sequenznummer, eine Broadcast-ID und einen Hop Count, d.h. die Anzahl der bisher durchlaufenen Knoten.

Ein Empfänger prüft zunächst anhand der Absenderadresse und Broadcast-ID, ob er den RREQ schon einmal weitergeleitet hat. Wenn ja wird der RREQ verworfen, was automatisch Schleifen vermeidet. Wenn nicht, wird zunächst überprüft, ob eine gültige Route zum gewünschten Ziel bekannt oder man selbst das Ziel ist. In diesem Fall wird der RREQ mit einem *Route Reply (RREP)* beantwortet. Sonst erhöht der Knoten den Hop Count um eins und leitet das Paket als Broadcast weiter. Gleichzeitig merkt er sich in einer Tabelle eine sogenannte *Reverse Route*, die angibt, von welchem Knoten er einen RREQ mit einer bestimmten Sequenznummer empfangen hat. Damit weiß der Knoten, wie er einen RREP an den Absender zurückschicken kann. Um die Ausbreitung der RREQs in einem großen MANET zu begrenzen, wird die Route Discovery typischerweise als *Expanding Ring Search* durchgeführt. Hierzu wird das *Time-To-Live (TTL)* Feld des ersten RREQ Pakets zunächst auf einen sehr kleinen Wert gesetzt. Erfolgt darauf keine Antwort, wird ein erneuter RREQ mit höherem TTL geschickt und so weiter.

In Abbildung 5.5 a wird davon ausgegangen, dass Knoten 5 eine gültige Route zu Knoten 8 mit Hop Count 2 kennt. Daher leitet er den Route Request nicht weiter, sondern antwortet direkt mit einem Route Reply (Abbildung 5.5 b). Auf dem unteren Pfad kennt kein Knoten eine gültige Route zu Knoten 8, so dass der RREQ bis zum Ziel

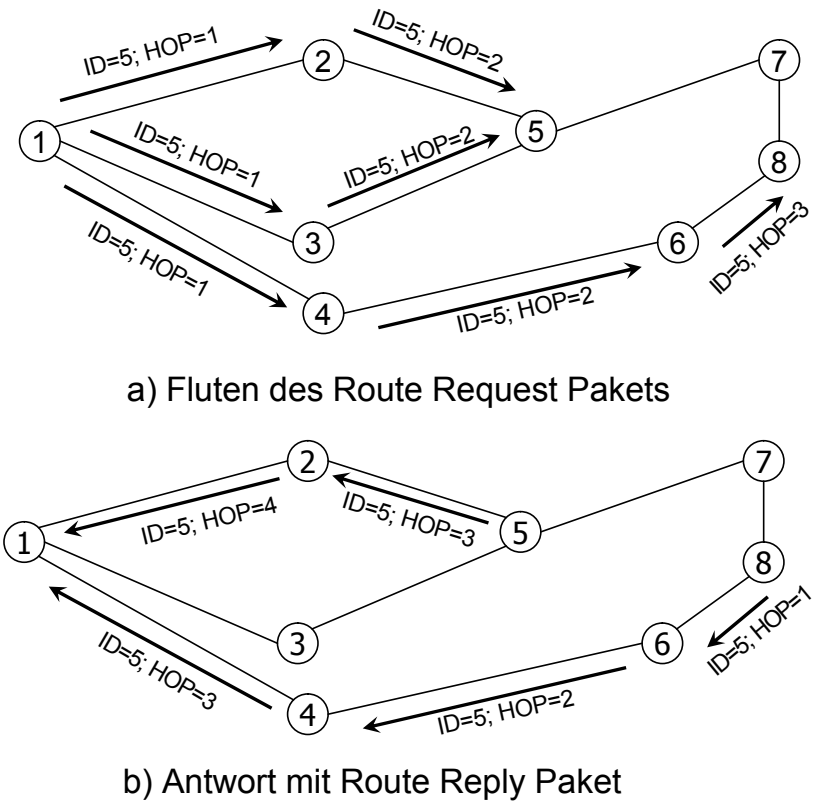

Abbildung 5.5.: Beispiel für Ad Hoc On-Demand Distance Vector Routing (AODV)

geflutet wird. Dieses erzeugt dann ebenfalls einen RREP. Erhält ein Zwischenknoten einen der Route Replies, so trägt er eine *Forward Route* in seine Routing Tabelle ein. Diese Forward Route zeigt in Richtung des RREP Absenders und enthält die Zahl der Hops bis zum Ziel. Danach wird der RREP in Richtung der jeweiligen Reverse Route weitergeleitet. Erhält ein Knoten RREPs von mehreren Nachbarn, so leitet er den ersten weiter. Folgende RREPs werden nur dann weitergeleitet, wenn sie einen kürzeren Hop-Count haben.

Erreicht der Route Reply den Ausgangspunkt der Route Discovery, so trägt dieser Knoten ebenfalls eine Forward Route ein und kann sofort beginnen, Daten über diese Route zu schicken. Empfängt er später weitere RREPs mit einer besseren Metrik (geringerem Hop Count), so kann er die Route gegebenenfalls anpassen. Entdeckt ein Knoten den Ausfall eines Links, so tritt die *Route Maintenance* in Aktion. Die Überwachung von Links wird entweder vom Link-Level unterstützt (z.b. bei Bluetooth) oder es wird ein HELLO Protokoll verwendet, welches die Erreichbarkeit der Nachbarn zyklisch überwacht.

Fällt eine Forward Route aus, so schickt der Knoten, der in einer Route *vor* dem ausgefallenen Teilstück sitzt, eine *Route Error (RERR)* Meldung entlang der Reverse Route zum Quellknoten einer Route. Dieser kann dann einen neuen Route Request anstoßen. Über die hier gemachten Ausführungen hinaus enthält AODV detaillierte Angaben darüber, wie lange Routen als gültig zu betrachten und wann sie zu verwerfen sind. Außerdem spezifiziert der Protokoll-RFC [PRD03] noch einige weitere Optimierungen

wie *Local Repair* und zusätzliche Angaben darüber, wie sich ein Knoten beispielsweise nach einem Reboot zu verhalten hat.

5.3.3. Dynamic Source Routing

Dynamic Source Routing (DSR) [JMHJ03, JMB01] ist ein proaktives Protokoll ähnlich AODV. Auch hier gibt es eine *Route Discovery*, bei der ein Route Request vom Quellknoten durchs Netz geflutet wird. Wie man in Abbildung 5.6 a sieht, startet der RREQ bei Knoten 1. Bei der Weiterleitung durch andere Knoten wird im RREQ Paket eine Liste der bereits passierten Knoten aufgebaut. Anhand dieser Liste kann ein Knoten auch erkennen, ob er einen bestimmten RREQ schon einmal weitergeleitet hat. In diesem Fall wird das Paket verworfen, was Schleifen in Routen vermeidet.

Erreicht der Route Request den Zielknoten 8, sendet dieser einen Route Reply, welcher die komplette Route von Quellknoten 1 zu Zielknoten 8 aus dem RREQ enthält (Abbildung 5.6 b). Besteht das Netz ausschließlich aus bidirektionalen Links, kann 8 den RREP einfach über die reverse Route transportieren. Wenn auch unidirektionale Links vorkommen können, muss 8 seinerseits einen Route Request nach 1 anstoßen und dabei den RREP an das RREQ Paket anhängen.

Sobald Knoten 1 einen Route Request erhält, speichert er die darin enthaltene Route in seinem Route Cache. Soll nun ein Paket an das Ziel 8 geschickt werden, wird die entsprechende Route in einem *Source-Routing Header* im Paket vermerkt, die Zwischenknoten müssen dann das Paket lediglich entlang dieser Route zum Ziel transportieren.

Fallen Verbindungen aus, so tritt eine *Route-Maintenance* ähnlich der von AODV in Aktion. Kann ein Datenpaket nicht über eine Route transportiert werden, weil ein Link ausgefallen ist, wird ein Route Error Paket entlang der reversen Route zurück zum Absender des ursprünglichen Datenpaketes geschickt. Unterwegs invalidieren alle Knoten ihre Route-Caches bezüglich dieser Route.

Neben diesem einfachen Basismechanismus enthält die DSR Spezifikation [JMHJ03] eine Vielzahl von möglichen Optimierungen. So gibt es die Möglichkeit zum „*Caching of Overheard Routing Information*“. Dabei fügen Knoten Topologieinformationen in ihren Routing-Cache ein, welche sie aus verschiedenen Quellen[2] erhalten haben.

Besitzt ein Knoten bereits eine gültige Route zum gewünschten Ziel, so kann er einen RREQ auch direkt beantworten, ohne ihn weiterzuleiten. Hierzu konstruiert er einen RREP aus seiner eigenen Route plus der Route im RREQ. In Abbildung 5.6 b wird davon ausgegangen, dass Knoten 5 bereits eine Route zu Knoten 8 über Knoten 7 kennt und deshalb die eingehenden RREQs direkt beantwortet. Andere Optimierungen wie *Package Salvaging*, *Automatic Route Shortening* oder *Caching Negative Information* sind in [JMHJ03, JMB01] aufgeführt. Eine ebenfalls interessante Erweiterung verwendet Hashchains statt Source Routen zur effektiveren Datenübertragung [Cas03]. Da sich viele dieser Erweiterungen nicht mit den später vorgestellten Sicherungsmechanismen vertragen, sind sie hier aber von untergeordnetem Interesse.

[2] z.B. durchlaufende RREQs/RREPs oder Empfang von Routing-Nachrichten im Promiscuous Mode

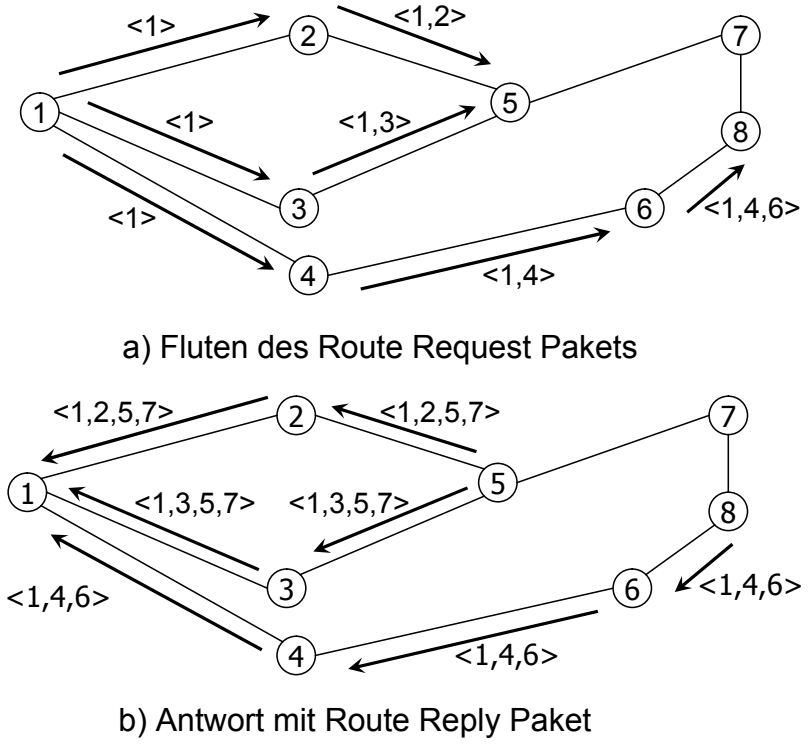

Abbildung 5.6.: Beispiel für Dynamic Source Routing (DSR)

5.3.4. Bluetooth Scatternet Routing

Die vorgestellten Protokolle setzen alle mehr oder weniger ein paketorientiertes Netzwerk wie IEEE 802.11 voraus. Bluetooth entspricht dem allerdings nur bedingt, da es wesentlich auf einer expliziten Verbindung zwischen Knoten aufbaut. Auch die sonstigen Eigenschaften sprechen nicht unbedingt für einen Einsatz in MANETs (siehe Abschnitt 4.3.1). Aus diesem Grund haben wir ein Routing Protokoll entwickelt, welches auf diese speziellen Eigenschaften von Bluetooth zugeschnitten ist, das *Bluetooth Scatternet Routing* (BSR) [KRSW03, Rib02].

Dieses überträgt das Konzept des *Circuit Switching*, wie es beispielsweise von ATM bekannt ist, auf die Verbindungen von Bluetooth und erlaubt so den Aufbau größerer MANETs mit Bluetooth.

BSR macht deutlich, dass man bei der Konzeption von Ad hoc Routing Protokollen durchaus von den Gegebenheit des Schicht-2 Netzes abhängt.

5.3.5. Weitere Ansätze

Neben den vorgestellten Protokollen haben Forschergruppen rund um die MANET WG eine Vielzahl weiterer interessanter Ideen publiziert und in Routing Protokollen umgesetzt. Beispielsweise wurden verschiedene hierarchische Routingprotokolle entwickelt [Lau86, LG97, RBS87, RS98, Sha84a, Sha85, Sha96], die das Netz in Cluster oder Zonen einteilen.

Das *Zone Routing Protocol (ZRP)* [HP98, HPS02d] geht dabei den Weg, innerhalb einer Zone ein proaktives (das *Intra-Zone-Routing-Protocol (IARP)* [HPS02c]) und zur Verbindung von Zonen ein reaktives Protokoll (das *Inter-Zone-Routing Protocol (IERP)* [HPS02b]) zu verwenden. Zur Routensuche wird das *Bordercast Resolution Protocol (BRP)* [HPS02a] eingesetzt.

Das *Fisheye State Routing (FSR)* [ICP+99] ist entfernt mit diesem hierarchischen Ansatz verwandt. FSR ist ein proaktives Protokoll, welches Topologieänderungen, die nahe an einem Knoten stattfinden, mit hoher zeitlicher Auflösung meldet, wohingegen ein Knoten nur noch ungenaue und verzögerte Informationen über weiter entfernte Änderungen erhält. Sendet ein Knoten ein Datenpaket an einen entfernten Knoten, so hat er zwar nur eine ungenaue Vorstellung davon, in welche Richtung er das Paket losschicken soll. Je näher das Paket aber dem Ziel kommt, umso genauer sind die Knoten informiert und umso genauer wird die Route. Man kann FSR also als eine Art ZRP mit einem nahtlosen Übergang zwischen Intra-Zone und Inter-Zone Routing ansehen.

Eine andere Idee ist das sogenannte *Position-based Routing* [MWH01], welches in Protokollen wie dem *Location-Aided Routing (LAR)* [KV98] oder im *Fleetnet Projekt* [Fle] umgesetzt wurde. Dabei gehen Lokations-Informationen (z.B. von einem GPS-Empfänger) in die Routing Entscheidung mit ein.

Ein weiterer Ansatz nennt sich *Power-aware Routing* und bezieht die Batteriepegel der einzelnen Geräte mit ein. *Associativity Based Routing (ABR)* [Toh97b, Toh97a, Toh02] versucht beispielsweise, zu frühe Netzwerk-Partitionierungen dadurch zu vermeiden, dass Knoten in zentralen Positionen aber mit geringem Batteriestand nur in unvermeidbaren Fällen als Zwischenknoten verwendet werden, um deren Batterie zu schonen. Generell bemüht sich ABR, nur möglichst stabile Links in eine Route aufzunehmen.

In eine ähnliche Richtung gehen Routing Protokolle, welche die Signalstärke eines Links mit in Betracht ziehen. Reduziert sich die Signalstärke einer Verbindung zwischen zwei Knoten, so deutet das unter Umständen darauf hin, dass sich die zwei Knoten voneinander entfernen. Hier könnte man also versuchen, frühzeitig nach einem alternativen Pfad zu suchen. Dieser Ansatz wird *Signal-Stability Routing* genannt und ist unter anderem im *Signal Stability-Based Adaptive Routing (SSA)* Protokoll umgesetzt [DRWT97].

Eine weitere Gruppe von Protokollen widmet sich dem Multicast [DFJ+94] in MANETs. So kennt beispielsweise AODV eine Variante M-AODV [RP] für Multicast-Unterstützung. Eine Übersicht über weitere Protokolle wie *Differential Destination Multicast (DDM)* oder *On-Demand Multicast Routing Protocol (ODMRP)* findet sich in [RT99].

Ausführlichere Beschreibungen vieler MANET Routing Protokolle finden sich beispielsweise in [Per01] und [Toh02], wohingegen [RT99] einen guten Überblick gibt.

5.4. Weitere Forschungen

Ein Großteil der Bemühungen der letzten Jahre zielte auf die Entwicklung der Routing Protokolle ab. Im Hinblick auf einen realen Einsatz von MANETs wird man jedoch auch eine Vielzahl weiterer Fragen untersuchen und beantworten müssen. [Per01] führt hierzu folgende Punkte an:

Skalierbarkeit: Wie groß können MANETs werden?

Quality of Service: Können QoS-kritische Anwendungen wie Videokonferenzen oder VoIP in MANETs genutzt werden?

Ablösung des Client-Server Paradigmas: Welche alternativen Formen der Kommunikation sind in MANETs sinnvoll?

Security: Wie schützt man sich vor Angriffen durch böswillige Knoten?

Interoperabilität mit dem Internet: Wie verbindet man ein MANET dynamisch mittels wechselnder Gateways mit dem Internet?

Power Control: Wie maximiert man die Betriebszeit der Knoten?

Seit dem Erscheinen von Perkins Buch wurden in vielen dieser Punkte bereits signifikante Anstrengungen unternommen, insbesondere auch im Hinblick auf Sicherheit. Aus dem Inhalt:

> „Clearly, security has so far not been satisfactorily investigated for ad hoc network protocols. [...] A quick scan through the chapters of this book will show that the topic of security for ad hoc protocols is impressive by its almost total absence."

Tatsächlich werden wir später sehen, dass das Thema Security in den Beschreibungen der Routing-Protokolle nicht behandelt wird oder es werden zumindest keine Lösungen angeboten.

5.5. Fazit

Dieses Kapitel hat deutlich gemacht, dass es sich bei den Ad hoc Netzen um ein breites und interessantes Forschungsgebiet mit einer Vielzahl von interessanten Fragestellungen handelt. Inwieweit MANETs in der Zukunft tatsächlich zu einer alltäglichen Art der Vernetzung werden, lässt sich momentan noch schwer abschätzen. Sicher ist aber, dass die Lösung der anstehenden Sicherheitsfragen eine wichtige Rolle für die weitere Entwicklung der Ad hoc Netze spielen wird.

Der Rest dieser Arbeit widmet sich nun ausschließlich der Sicherheit in MANETs. Wie wir später sehen werden, gibt es mittlerweile bereits eine Anzahl von Forschungsgruppen, die sich dieses Themas in den letzten Jahren angenommen haben. Schwachpunkt vieler dieser Arbeiten ist das Fehlen einer wirklich umfassenden Sicherheitsanalyse. Diese soll im folgenden Kapitel durchgeführt werden. Daran anschließend werden zunächst verwandte Arbeiten vorgestellt, bevor ein eigener Vorschlag einer Sicherheitsarchitektur für MANETs unterbreitet und analysiert wird.

6. Sicherheitsaspekte von MANETs

6.1. MANET Besonderheiten

Verglichen mit herkömmlichen Netzen weisen Mobile Ad hoc Netzwerke eine Reihe von Besonderheiten im Hinblick auf deren Sicherheit auf. Diese Unterschiede müssen bei der Konzeption eines MANETs berücksichtigt werden, da sonst sehr leicht sämtliche der im ersten Kapitel genannten Sicherheitsziele verfehlt werden.

Folgende grundlegenden Unterschiede sind zu berücksichtigen:

- Es kommen ausschließlich mobile Komponenten zum Einsatz. Diese können verloren gehen oder gestohlen werden. Damit fallen (leichter als bei herkömmlichen PCs oder Routern) sämtliche Daten auf diesem Gerät in die Hand eines Angreifers: Passworte, kryptographische Schlüssel uvm. Entsprechend darf die Gesamtsicherheit des Systems keinesfalls auf einer einzigen Komponente beruhen.
- Die Kommunikation erfolgt über ein Funknetz, welches von jedermann problemlos abgehört werden kann. Deshalb darf die Sicherheit des Systems nicht von der Kenntnis der Nachrichten abhängen. Während es also eventuell ausreicht, wenn sich zwei herkömmliche Router auf einer Punkt-zu-Punkt-Verbindung über ein Klartext-Passwort authentifizieren, ist das bei MANETs inakzeptabel.
- In klassischen Netzwerken ist eine Trennung zwischen einer (zentral administrierten) Routing-Infrastruktur und normalen Knoten möglich. Somit kann sich die Routing-Infrastruktur leicht durch Passwörter oder Message Authentication Codes (MAC) vor den normalen Knoten abschotten. Normale Knoten können dann keine Konfigurationsänderungen durchführen und auch nicht in das Routing-Protokoll eingreifen. Dies ist bei MANETs nicht gegeben, da hier jeder Knoten gleichzeitig auch ein Router ist und die Knoten (und damit die Routing-Komponente) jeweils von deren Benutzern verwaltet werden.

Zusätzlich gibt es für die Knoten auch neue Motivationen, sich nicht an der gemeinsamen Routing-Infrastruktur zu beteiligen. Neben den klassischen Angreifern, wie sie im ersten Kapitel vorgestellt wurden, müssen wir in MANETs auch noch Knoten berücksichtigen, welche durch Fehlfunktionen oder aus egoistischen Gründen eine Störung des Netzwerkes hervorrufen.

Wir unterscheiden also drei Arten von Knoten, welche die Sicherheit und Funktionsfähigkeit des Netzwerkes beeinträchtigen können:

1. Fehlerhafte Knoten
2. Egoistische Knoten
3. Böswillige Knoten

Wo eine Unterscheidung nicht notwendig ist, werden diese im Folgenden mit *FEB-Knoten* abgekürzt. Die Unterscheidung ergibt sich aus der Motivation der Knoten bzw. der Motivation ihrer Benutzer.

6.1.1. Fehlerhafte Knoten

Hier liegt schlicht eine Fehlfunktion eines Knotens vor. Dadurch werden falsche Topologie-Informationen verbreitet oder Datenpakete nicht weitergeleitet. In der Praxis können solche Knoten leicht durch Programmierfehler entstehen. Selbst wenn diese behoben sind, muss man davon ausgehen, dass Teilnehmer in einem MANET diese Aktualisierungen nicht auf ihren Rechnern installiert haben und somit alte Versionen der Routing-Software die Funktionalität des Netzes stören. Obwohl es sich also um keinen Angriff im engeren Sinne handelt, sind die Auswirkungen auf das MANET durchaus vergleichbar. Es ist wünschenswert, wenn ein Sicherheitssystem solche Knoten erkennt und solange aus dem Netz ausschließt, bis der Besitzer des Knotens den Grund für die Fehlfunktion behoben hat.

6.1.2. Egoistische Knoten

In einem MANET erbringen alle Knoten gemeinsam eine Leistung, von der wiederum alle profitieren. Das Ergebnis dieser Leistung ist die Konnektivität, zu welcher alle beitragen und die alle benutzen. Dabei wendet ein Knoten einen Teil seiner Ressourcen (CPU, Bandbreite, Batterie) auf, um den Verkehr von anderen weiterzuleiten, immer in der Hoffnung, dass diese einen Teil ihrer Ressourcen dazu aufwenden, seine Datenpakete zu transportieren.

Die Verlockung ist natürlich groß, die eigenen Aufwendungen für andere Knoten einzusparen, d.h. selbst keine Datenpakete für andere Knoten weiterzuleiten, die Leistung der anderen Knoten für den Datentransport aber in Anspruch zu nehmen. In dem Maße, wie die Zahl der egoistischen Knoten in einem Netzwerk ansteigt, sinkt die Leistung des Gesamtnetzes ab, d.h. wenn die Zahl der kooperierenden Knoten zu stark abnimmt, wird die im Netz zur Verfügung stehende Gesamtbandbreite kleiner und auch die Zahl der fehlerhaften Routen und verlorenen Pakete wird ansteigen.

Für jeden Knoten lässt sich die Nützlichkeit des Netzwerkes als eine Funktion berechnen, welche die eigenen Kosten in Relation zum eigenen Nutzen setzt. In Anlehnung an die Arbeiten von Buttyan und Hubaux [BH01b, BH03] zum Thema Nuglets, kann diese Funktion, wie in Abbildung 6.1 gezeigt, definiert werden.

Nach diesem Modell hat ein Knoten eine bestimmte Lebensdauer. Diese hängt typischerweise von der initialen Batteriekapazität B, dem Kommunikationsaufkommen und insbesondere von der Zahl der gesendeten Pakete ab. Während seiner Lebensdauer empfängt er Pakete für sich selbst (IN_o) sowie Pakete, die er an andere Knoten weiterleiten soll (IN_f). Ebenso sendet er eine bestimmte Anzahl von Paketen (OUT) an andere Knoten, wobei sich diese in eigene Pakete (OUT_o) und weitergeleitete Pakete (OUT_f) unterteilen lassen. Verfügt ein Knoten nicht über genügend Energie, so kann er unter Umständen eigene Pakete nicht mehr abschicken, diese werden verworfen (DRP_o). Um Energie zu sparen, kann er sich entschließen, fremde Pakete nicht weiterzuleiten, sondern ebenfalls zu verwerfen (DRP_f).

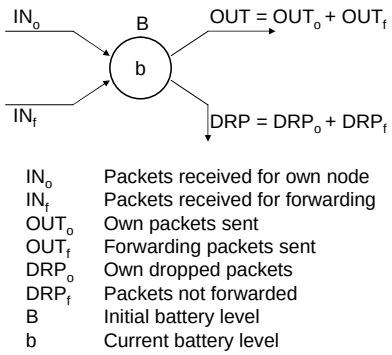

Abbildung 6.1.: MANET Nützlichkeitsfunktion

Ziel eines Knoten wird es in diesem Modell also sein, Energie zu sparen, um die Lebensdauer zu maximieren und möglichst wenige eigene Pakete zu verwerfen (d.h. DRP_o zu minimieren). Trivial lässt sich dies erreichen, wenn man keine Pakete weiterleitet, sondern alle fremden Pakete verwirft ($OUT_f = 0$, $DRP_f = IN_f$). Später wird durch Simulationen gezeigt, wie sich derartige Knoten auf den Durchsatz eines Netzes auswirken.

Etwas weniger egoistische Knoten verwerfen vielleicht nicht alle Pakete, sondern entscheiden anhand der verbleibenden Batteriekapazität b, ab wann Pakete nicht mehr weitergeleitet werden. Es bleibt Aufgabe eines Sicherheitssystems, den Umfang zu definieren, in welchem derartiges Verhalten akzeptiert wird.

Die Spieletheorie beschäftigt sich seit Langem mit den Auswirkungen von egoistischem und altruistischem Verhalten bei der Interaktion von Menschen. Deren Ergebnisse lassen sich auch auf das Verhalten von Knoten in MANETs übertragen, wie verschiedene Arbeiten zeigen [BH01a, BHČ02, BH03, MM03]. Anhand der verwendeten Modelle lässt sich der Nutzen abschätzen, den ein Knoten durch sein egoistisches Verhalten erreicht. Inwieweit das modellhafte Verhalten allerdings dem echter Nutzer entspricht, lässt sich mangels real verfügbarer Ad hoc Netze von nennenswerter Größe noch nicht sagen.

Die englischsprachige Literatur nennt das Konzept der egoistischen Knoten bei MANETs *Selfishness*. In einer anderen Vernetzungsvariante, den sog. *Peer-to-Peer*- oder *Filesharing*-Netzwerken, tritt ein ähnliches Phänomen auf, wenn Teilnehmer zwar Dateien wie Musikstücke aus dem Netz laden, selbst aber keinen Speicherplatz und keine Dateien zur Verfügung stellen. Dieses Phänomen wird als *Freeriding* bezeichnet. Untersuchungen haben gezeigt, dass etwa drei Viertel der Teilnehmer an populären P2P Netzen dieses Verhalten zeigen; mit negativen Auswirkungen auf die Leistungsfähigkeit des Gesamtnetzes [RL03].

6.1.3. Böswillige Knoten

Im Englischen als *Malicious Nodes* bezeichnet, entsprechen diese MANET-Teilnehmer weitgehend den Angreifern, wie sie in Abschnitt 2.2 beschrieben wurden. Allerdings erlaubt die Routing-Funktion der Knoten eine ganze Klasse von neuen Angriffsverfahren,

welche auf die Routing-Infrastruktur eines MANETs abzielen und insbesondere Denial-of-Service Angriffe ermöglichen. Da der Knoten ja selbst elementarer Bestandteil dieser Routing-Infrastruktur ist, fällt es umso schwerer, derartige Angriffe zu erkennen und ihnen zu begegnen. Im Gegensatz zu den egoistischen Knoten, welche selbst Wert auf die Funktionsfähigkeit des Netzes legen, geht es böswilligen Knoten oft gerade darum, diese Funktionsfähigkeit zu zerstören, auch um den Preis, dass der eigene Knoten dann nicht mehr kommunizieren kann.

6.2. Angriffsanalyse

Um wirkungsvolle Schutzmaßnahmen zu entwerfen, ist es prinzipiell hilfreich, zunächst eine möglichst große Anzahl von Angriffen strukturiert zu erfassen. Anhand dieser Aufstellung kann man später untersuchen, welche Arten von Angriffen das Schutzsystem verhindert und in welchen Teilen noch Schwächen bzw. Nachbesserungsbedarf besteht. Dabei kann eine solche Aufstellung niemals einen Anspruch auf Vollständigkeit haben, da regelmäßig neue Angriffe entdeckt werden, die in dieser Aufstellung nicht enthalten sind.

Denkbar wäre natürlich auch der umgekehrte Ansatz, nämlich ein Sicherheitssystem auf analytischem Weg als prinzipiell sicher gegen Angriffe zu entwerfen bzw. diese Sicherheit zu beweisen. Existierende Arbeiten auf diesem Gebiet sind in [KMM94] zusammengefasst. Ein Beispiel für die formale Analyse eines Protokolls, bei welcher ein Fehler entdeckt wurde, findet sich in [Low96]. In Abschnitt 3.7 wurde mit der BAN-Logik bereits ein Verfahren vorgestellt, mit welchem sich eine formale Analyse von kryptographischen Protokollen durchführen lässt.

Die dabei auftretenden Probleme sind vielfältig. Nur wenn man einen Angriff bzw. eine Schwachstelle kennt und somit formal beschreiben kann, ist man in der Lage, ein Sicherheitssystem entsprechend resistent auszulegen bzw. zu beweisen, dass es gegen diesen Angriff schützt. Gleichzeitig sind solche Beweise bzw. Designverfahren recht komplex und lassen sich mit realistischem Aufwand nicht auf ein umfangreiches Sicherheitssystem wie im vorliegenden Fall anwenden. Trotzdem wird bei der Analyse des Routingprotokolls SDSR mittels BAN Logik auf ein derartige Verfahren zurückgegriffen.

Für die Darstellung und Analyse von Angriffsformen bieten sich hierarchische Verfahren an. In [Sch99] stellt Bruce Schneier die sogenannten *Attack Trees* (Angriffsbäume) vor. Ausgehend von einem Ziel bzw. einer Motivation wird hier ein hierarchischer Baum mit Wegen aufgestellt, wie das Ziel eines Angriffs zu erreichen ist. Daraus lässt sich dann umgekehrt ableiten, welche Angriffe durch eine Schutzmaßnahme unterbunden werden. Neue Angriffe lassen sich an den entsprechenden Stellen im Baum einfügen.

Bei der gewählten Notation wird der Baum in Textform dargestellt. Die Einrückung gibt die Tiefe des Baums wider. Die verschiedenen Bäume werden mit Großbuchstaben bezeichnet. An der Wurzel eines Baumes steht das Ziel eines Angreifers, beispielsweise möchte er in Baum A Energie sparen. Gibt es mehrere Möglichkeiten, dieses Ziel zu erreichen, so werden diese eine Ebene tiefer als Teilbäume aufgeführt und mit *OR* bezeichnet. Sind zum Erreichen eines Zieles mehrere Schritte auszuführen, so wird

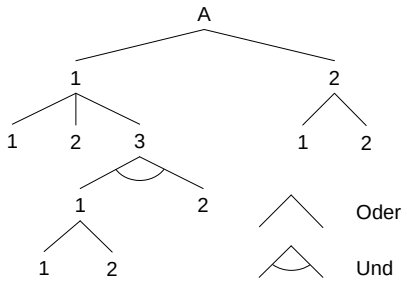

Abbildung 6.2.: Graphische Darstellung von Angriffsbaum A (Zweige unter A.1.1 und A.1.2 sind der Übersicht wegen nicht dargestellt.)

stattdessen die Bezeichnung *AND* verwendet. Alle Knoten einer Ebene sind durchnummeriert, so dass sich jeder Knoten einfach benennen lässt.

Beispielsweise ist mit A.1 der erste Knoten unterhalb der Wurzel von Baum A („Keine Teilnahme am Routing“, siehe Tabelle 6.1) gemeint. Um dieses Ziel zu erreichen, hat er mehrere Möglichkeiten. Eine davon (A.1.3) besteht darin, aus einer aktiven Route auszusteigen. Hierzu muss der Knoten einen Route Error auslösen und er darf dann nicht an der folgenden Route Discovery teilnehmen. Wieder gibt es dazu mehrere Möglichkeiten, welche als Querverweise in andere Teile des Baumes (z.B. A.1.2.1.1) aufgeführt sind. Zum Vergleich ist in Abbildung 6.2 ein Teil von Angriffsbaum A graphisch dargestellt.

Verhindert ein Sicherheitssystem die Durchführung *irgendeines* AND Unterknotens bzw. *aller* OR Unterknoten, so kann der zugehörige Teilbaum nicht erfüllt werden. Wenn also ein solches System einen ganzen Angriffsbaum unerfüllbar macht, kann ein Angreifer sein Ziel nicht erreichen, das System ist sicher.

Das gilt jedoch nur solange, wie der Angriffsbaum auch wirklich vollständig ist. Dies ist nahezu unmöglich sicherzustellen. Trotzdem sind Angriffsbäume ein wertvolles Hilfsmittel um Bedrohungen für IT-Systeme zu strukturieren und zu analysieren. Diese Analyse sollte immer den ersten Schritt bei der Entwicklung eines Sicherheitssystems sein.

Im Rahmen dieser Arbeit wurde am Beispiel des DSR Protokolls eine umfassende Sicherheitsanalyse unter Verwendung von Angriffsbäumen durchgeführt. Die Konzentration auf DSR als ein weit verbreitetes On-Demand Routing Protokoll ist sinnvoll, da sonst die Angriffe sehr allgemein und diffus beschrieben werden müssten. Generell sind jedoch mindestens die oberen Schichten der Angriffsbäume protokollunabhängig und die dargestellten Bäume lassen sich bei genauer Kenntnis eines Routingprotokolls mit vertretbarem Aufwand geeignet abwandeln.

6.2.1. Egoistische Knoten

Angriffsbaum A in Tabelle 6.1 zeigt, welche Möglichkeiten einem egoistischen Knoten zur Verfügung stehen, wenn er in einem DSR-basierten Ad hoc Netzwerk die eigenen Ressourcen schonen will.

Dazu hat der Knoten die Wahl, ob er bereits den Routingprozess stört (A.1.2) oder gar

Baum A: Ressourcen einsparen					
OR	1.	Keine Teilnahme am Routing			
	OR	1.	Keine Weiterleitung von Routing-Daten		
		OR	1.	Route Request nicht weiterleiten	
			2.	Route Reply nicht weiterleiten	
			3.	Hop-Limit/TTL in Route Request/Reply auf 0 (bzw. kleinen Wert) setzen	
		2.	Routing Daten/Topologie modifizieren		
		OR	1.	Route Request fälschen	
			OR	1.	Zusätzliche Hops in Route Request einbauen (Route wird länger)
			2.	Route Reply fälschen	
			OR	1.	Eigene ID im RREP durch Umleitung über benachbarte Knoten ersetzen
				2.	Komplett falsche Route angeben, dann RERR und Salvaging
				3.	Zusätzliche Hops in Route Reply einbauen (Route wird länger)
			3.	Eigenen Interface Index in RREP als extern deklarieren	
		3.	Aus aktiver Route aussteigen		
		AND	1.	Route Errors erzeugen	
			OR	1.	Spontan RERR verschicken
				2.	Keinen ACK schicken, führt zu RERR bei anderen Knoten
			2.	Bei neuem Route Request nicht teilnehmen (siehe z.B. A.1.1.1 oder A.1.2.1.1)	
	2.	Keine Weiterleitung von Datenpaketen			
	OR	1.	Datenpakete löschen		
		2.	Hop-Limit/TTL auf 0/1 setzen (dadurch entstehen RERR)		

Tabelle 6.1.: Angriffsbaum A: Ressourcen einsparen

nicht an ihm teilnimmt (A.1.1), oder ob er aus einer aktiven Route aussteigt (A.1.3). Alternativ kann er sich auch schlicht weigern, Datenpakete weiterzuleiten, obwohl er auf einer gültigen Route liegt (A.2).

Interessant sind für den Angreifer insbesondere die Fälle, bei denen nicht direkt auf den Knoten zurück geschlossen werden kann, weil Fehler an ganz anderen Stellen im Netz auftreten. Das ist z.B. beim Verändern der TTLs der Fall. Zwei Fälle (A.1.1 und A.2) werden in Abschnitt 6.3 durch Simulationen näher untersucht.

6.2.2. Böswillige Knoten

Neben egoistischen können auch böswillige Knoten das Netz stören. Hierbei kann man unterscheiden zwischen Knoten, welche das Netz lahmlegen wollen (DoS) und Knoten die unbefugt Informationen mithören und modifizieren wollen. Schließlich gibt es noch

den klassischen Angriff gegen einen einzelne Knoten, mit dem Ziel, diesen unter die eigene Kontrolle zu bekommen.

Teilziel DoS

Angriffsbaum B (Tabellen 6.2 und 6.3) listet Möglichkeiten zur gezielten Beeinträchtigung des gesamten Netzwerks oder nur einzelner Teilnehmer auf.

Angriffe, bei denen die Funkübertragung durch Störmaßnahmen beeinträchtigt wird (B.1), werden im weiteren Verlauf nicht näher betrachtet. Alternativ stehen einem Knoten vielfältige Möglichkeiten zur Verfügung, das MANET zu überlasten (B.2) oder den Aufbau korrekter Routen zu verhindern (B.3).

Teilziel Zugriff auf Information

Zunächst ist zu definieren, welche Art von Information für den Angreifer von Interesse ist. Ist allgemein der Datenverkehr interessant, dann müssen Pakete im Netzwerk abgehört und dazu gegebenenfalls auch zum Knoten des Angreifers umgelenkt werden. Manchmal ist ein Angreifer an speziellen Informationen zum Verhalten der Netzteilnehmer interessiert, z.B. zur Analyse von Bewegungsmustern und Kaufgewohnheiten, oder er will mehr über die Identität eines Teilnehmers herausfinden. Neben den Informationen in den Datenpaketen können hier auch Informationen aus dem Routingprotokoll von Interesse sein.

Baum C in Tabelle 6.4 zeigt zunächst verschiedene Möglichkeiten, auf den Inhalt von Datenpaketen zuzugreifen. Zunächst kann ein böswilliger Knoten natürlich auf Datenpakete zugreifen, welche er im Rahmen des normalen Routings transportiert (C.1). Genügt das nicht, so wird er versuchen, zusätzlichen Verkehr zu sich umzuleiten (C.2). Hierzu bietet das Routingprotokoll verschiedene Ansatzpunkte.

Baum D (Tabelle 6.5) erläutert schließlich, wie man durch Analyse des Routingprotokolls und der Verkehrsflüsse Informationen über einen Teilnehmer erlangt. Im Kapitel 9 interessiert vor allem das Erstellen von Bewegungsprofilen (D.1). Zusätzlich kann ein Knoten auch versuchen, die tatsächliche Identität eines Knotens oder seines Benutzers zu ermitteln (D.2). Aus beiden Informationen und Daten der Applikationsebene lassen sich dann komplexe Nutzerprofile erstellen (D.3).

Das es tatsächlich möglich ist, aus den Routinginformationen die Position eines Knotens zu erschließen (D.1.1.2) wurde in [ČHH02] gezeigt. Eine detaillierte Diskussion zu Lokation Tracking und Nutzungsprofilen bei MANETs folgt in Kapitel 9.

Teilziel Modifikation von Information

Die Modifikation von Informationen des Routingprotokolls wurde bereits an verschiedenen Stellen der obigen Angriffsbäume diskutiert. Geht es darum, sich selbst aus dem Routing-Prozess auszuklinken, so ist Angriffsbaum A.1.2 relevant. Ist das Ziel, die Funktionalität des Netzwerkes durch Modifikationen im Routing-Protokoll zu stören, so gilt Angriffsbaum B.2. Geht es dem Angreifer um die Modifikation von Anwendungsdaten, so kann es hierzu eventuell notwendig sein, diese Datenpakete durch den

Baum B: Gesamtnetz/einzelnen Knoten in Funktionsfähigkeit einschränken

OR 1. Beliebige/alle Datenpakete zerstören
- OR 1. Angriff auf Funkschnittstelle
  - OR 1. Gesamtes Frequenzband stören
    - OR 1. Starken breitbandigen Störsender verwenden
      2. Schwachen breitbandigen Störsender nahe am Ziel aufstellen
    2. Gezielte Kollisionen erzeugen (evtl. muss Spreading Code, Hopping Sequence o.Ä. bekannt sein)
    - OR 1. Dauerhaft schwaches Störsignal senden
      2. Gezielt bestimmten Knoten/bestimmte Nachrichten stören
      - AND 1. Versand von Daten in Nachbarschaft erkennen
        2. Sofort Störsignal senden
- 2. Überlasten von Komponenten (Bandbreite, CPU, Batterie, ...)
- OR 1. Direkte Nachbarn überlasten
  - OR 1. Nachbarn beliebige falsche Pakete schicken
    2. Beliebige/mehrere Knoten in einer Route/im Gesamtnetz überlasten
  - OR 1. Unnötige/viele Route Requests verschicken (Discovery Sturm)
    2. Alte Route Replies wiederholt senden
    3. Normales Datenpaket wiederholt senden
    4. Künstlich lange Pfade im Netz erzeugen
    5. Unmotivierte Route Errors im Netz verschicken (dadurch neue Route Requests)
    6. Routingschleifen erzeugen
    - OR 1. Schleife bei Route Request oder Route Reply Weiterleitung einbauen
      2. Piggybacked Route Reply in Route Request löschen
    7. Möglichst viele Routen auf einzelnen Knoten umlenken (siehe B.3)

Fortsetzung auf der nächsten Seite

Tabelle 6.2.: Angriffsbaum B: Gesamtnetz/einzelnen Knoten in Funktionsfähigkeit einschränken

Fortsetzung Baum B			
OR	3.	Korrekte Routingfunktion stören	
	OR	1.	Pakete gehen im Netz verloren („Black Hole Routing“)
		OR	1. Falsches Acknowledgment/Route Reply für nicht erreichbare Nachbarn X senden
			2. Route Errors verwerfen, Routendefekt wird nicht erkannt
			3. Selbst als „Black Hole“ fungieren
			AND 1. Alle RREQ mit möglichst kurzem Pfad beantworten
			2. Datenpakete verwerfen
			4. Nachbarschaft des Zielknotens in den Pfad der Route-Request aufnehmen. Diese erkennen eine Schleife, Route Request erreicht Ziel nicht
			5. Discovery ID im Route Request auf alte ID setzen, Antworten gehen verloren/werden verworfen
			6. Gefälschten Route Request mit gefälschtem Absender X und zukünftiger ID verschicken, Anfragen durch X werden als ungültig verworfen
		2.	Topologiedaten zerstören
		OR	1. Fehlerhafte RREPs verteilen, durch Overhearing landen diese in den Routing Caches der Knoten („Cache Pollution“)

Tabelle 6.3.: Angriffsbaum B: Gesamtnetz/einzelnen Knoten in Funktionsfähigkeit einschränken

Baum C: Zugriff auf Informationen in Datenpaketen			
OR	1.	Datenpakete, welche im Verlauf der normalen Nutzung durch eigenen Knoten weitergeleitet werden, protokollieren/analysieren	
	2.	Datenpakete gezielt zum eigenen Knoten umleiten	
	OR	1.	Knoten geeignet positionieren, dass er auf der Verbindungslinie zwischen Quelle und Ziel liegt
		2.	Möglichst geringen Hopcount für eigene Verbindung zum Ziel vortäuschen (Idealfall: Ziel == next hop)
		OR	1. Route Reply mit Ziel als nächstem Hop angeben, separaten Route Request zum Ziel schicken, Pakete vor Weiterleitung geeignet umschreiben
			2. Wurmloch Angriff
			AND 1. Kooperierenden Knoten nahe am Ziel positionieren
			2. Tunnel zu diesem Knoten aufbauen
			3. Route Request/Reply durch diesen Tunnel schicken, der Tunnel zählt als ein Hop, daher hat Route bessere Metrik
			4. Datenpakete durch den Tunnel weiterleiten

Tabelle 6.4.: Angriffsbaum C: Zugriff auf Informationen in Datenpaketen

Baum D: Gewinnung von Informationen über Netzteilnehmer
OR 1. Ermitteln von Bewegungsprofilen (eines/aller Benutzer)
AND 1. Tracking eines Knotens (dabei evtl. Benutzung von Hilfsknoten mit bekannter Position)
OR 1. Positionsbestimmung anhand von Funksignal (Richtantenne, Signalstärke)
2. Positionsbestimmung anhand von Routinginformationen
OR 1. Regelmäßiges Senden von RREQ, Auswerten der RREP
2. Auswerten der Routinginformationen (RREQ/RREP), welche am Knoten „vorbeikommen"
3. Umleiten von Datenpaketen durch eigenen Knoten (siehe C.2), Analyse aller durch den Knoten laufenden Source-Routen
3. Auswerten zusätzlicher Informationen im Netz (z.B. GPS-Informationen)
2. Zuordnung einer Knoten-ID zu einem Benutzer (auch bei wechselnder Knoten-ID, siehe D.2)
2. Ermitteln der Identität von Benutzern
OR 1. Analyse der durch den Knoten laufenden Datenpakete (evtl. Umleiten siehe C.2)
2. Angriff auf Anmeldeprozess bzw. Vergabe/Erzeugung der Knoten-ID
3. Erstellen von Nutzungsprofilen
OR 1. Analyse der durch den Knoten laufenden Datenpakete (evtl. Umleiten siehe C.2)

Tabelle 6.5.: Angriffsbaum D: Gewinnung von Informationen über Netzteilnehmer

Parameter	Wert
Anzahl Knoten (numnodes)	50
Raumgröße X (maxx/m)	1500
Raumgröße Y (maxy/m)	300
Verkehrsmodell (traffic type)	cbr
Senderate (send rate)	4.0
Zufalls-Initialisierung (random seed)	1
Max. Zahl von Verbindungen (max connections)	20
Paketgröße (pktsize/byte)	512
Pause (pause time/s)	0
Simulationszeit (sim time/s)	900

Tabelle 6.6.: Parameter für ns-2 Simulationen

eigenen Knoten umzuleiten. Dies ist in Baum C erläutert. Sind diese Daten nicht durch zusätzliche Maßnahmen geschützt, kann ein Zwischenknoten diese Daten vor der Weiterleitung natürlich auch beliebig modifizieren.

Teilziel Eindringen ins Netz/Knoten

Hier ist das Ziel ein Eindringen in die am Aufbau des MANETs beteiligten Knoten oder generell die (unerlaubte) Teilnahme am MANET. Die Schutzfunktionen der Knoten sind in der Regel auf Betriebssystemebene angesiedelt und unterscheiden sich bei MANETs nicht wesentlich von Rechnern in klassischen Netzwerken. Daher wird dieser Aspekt nicht weiter betrachtet. Manchmal ist für das Eindringen in einen Knoten das vorherige Belauschen (z.B. Mithören von unverschlüsselten Passwörtern) oder eine Modifikation von dessen Netzwerkkommunikation notwendig. Siehe hierzu vorherige Teilziele.

Bei der unerlaubten Teilnahme am Netzwerk ist zunächst zu definieren, durch welche Authentifizierungsmechanismen sich ein potentieller Teilnehmer legitimieren muss. Da wir in unserem Szenario von öffentlichen Ad hoc Netzwerken ausgehen, steht das MANET zunächst jedem Knoten offen. Sollte allerdings ein Knoten als FEB erkannt und aus dem Netz ausgeschlossen werden, so muss das Sicherheitssystem sicherstellen, dass er nicht umgehend unter einer geänderten Identität wieder am Netz teilnimmt.

6.3. Auswirkungen von Angriffen

Um den Einfluss von Angriffen auf ungeschützte Netzwerke zu analysieren, haben wir mit dem Simulationstool ns2 [NS2] eine Reihe von Simulationen durchgeführt, welche diese Effekte veranschaulichen sollen. Als Protokolle für den Angriff haben wir DSR und AODV verwendet, da diese die wohl am häufigsten eingesetzten und untersuchten MANET Routing Protokolle sind.

Die Protokolle wurden derart modifiziert, dass eine frei wählbare Anzahl von Knoten ein bestimmtes egoistisches Verhalten zeigt. Die Rahmenbedingungen der Simulation blieben dabei immer gleich (siehe Tabelle 6.6).

6.3.1. Egoistische Knoten

Es wurden zwei Arten von egoistischen Knoten modelliert. Der Knoten vom Typ *Egoistisch-1* leitet gar keine Pakete weiter, es werden sowohl Kontroll- als auch Datenpakete verworfen. Damit spart er entsprechend der vorher aufgestellten Nützlichkeitsfunktion Ressourcen ein, da er diese ausschließlich für sich selbst einsetzt und keine Energie in die Weiterleitung fremder Pakete steckt.

In der Implementierung reicht es, sämtliche Route-Requests zu verwerfen, da somit keine gültigen Routen durch diesen Knoten zu Stande kommen und er folglich auch nicht zur Weiterleitung von Verkehr herangezogen wird. Somit entspricht das Verhalten des Knotens dem Angriff A.1.1.1 aus Angriffsbaum A. Der Knoten generiert im Übrigen selbst ungehindert Verkehr, wie jeder andere Knoten im Netz.

Ein Nachteil des Verhaltens *Egoistisch-1* ist, dass ein solcher Knoten durch unser später vorgestelltes Intrusion Detection System MobIDS recht leicht erkannt werden kann, da er bei jeder Route Discovery Pakete verwirft.

Im Gegensatz dazu nehmen Knoten vom Typ *Egoistisch-2* zwar ganz normal an der Route Discovery teil, d.h. sie können auch Bestandteil einer Source-Route werden, allerdings weigern sie sich, den dann folgenden Datenverkehr weiterzuleiten. Ein Knoten weigert sich also erst dann zu kooperieren, wenn er tatsächlich als Bestandteil einer Route ausgewählt wird. Da ein Knoten, der einen Route Request weiterleitet, nur in relativ wenigen Fällen später auch tatsächlich zur Route gehört, verringern sich damit die Chancen eines IDS, ein Fehlverhalten festzustellen.

Die Abbildungen 6.3 und 6.4 zeigen die Ergebnisse der Simulationen für DSR. Deutlich sind mehrere Punkte festzustellen: In dem Maß, wie die Zahl der egoistischen Knoten steigt, sinkt die Empfangsrate, also der Prozentsatz von gesendeten Paketen, die ihren Empfänger erreichen. Weiterhin hat die Bewegungsgeschwindigkeit der Knoten einen gewissen Einfluss darauf, wie gravierend dieser Effekt ist. Und drittens sind egoistische Knoten vom Typ 2 offensichtlich schädlicher für das Netz als solche vom Typ 1.

Doch warum ist das so? In einem MANET mit egoistischen Knoten vom Typ 1 ist die Wahrscheinlichkeit geringer, dass eine Route zwischen zwei Teilnehmern gefunden werden kann, als in einem normal funktionierenden Netzwerk. Da die egoistischen Knoten hier Route-Requests nicht weiterleiten, fallen sie für die Netz-Konnektivität aus. Existiert keine Alternativ-Route, können Pakete nicht zugestellt werden und müssen verworfen werden. Bei einer höheren Bewegungsgeschwindigkeit fällt dieser Effekt offenbar noch gravierender aus als bei langsamen Teilnehmern.

Trotzdem hat hier das Ad hoc Netz noch eine Chance, derart unkooperative Teilnehmer auf anderen Routen zu umgehen. Anders bei egoistischen Knoten vom Typ 2. Diese bieten zunächst ihre Dienste an, indem sie Route Requests und auch Route Replies weiterleiten. Auch sonst verhalten sie sich im Rahmen des DSR Protokolls „normal“. Wenn sie dann in einer etablierten Route als Zwischenknoten auftauchen, werden jedoch alle Pakete stillschweigend verworfen. Das DSR Protokoll bekommt davon nichts mit und versucht auch nicht, eine alternative Route zu finden. Bei hoher Bewegungsgeschwindigkeit reichen hier bereits 20% egoistische Knoten, damit die Datenpakete nur noch mit 50% Wahrscheinlichkeit zugestellt werden können.

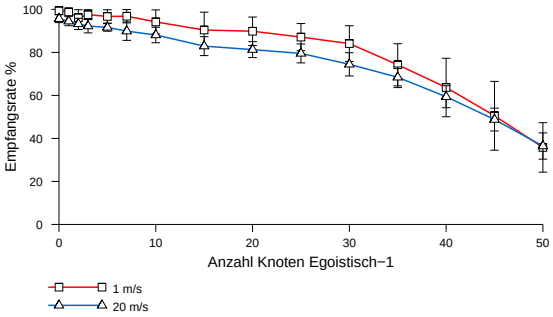

Abbildung 6.3.: Egoistisches Verhalten bei DSR – Fall 1

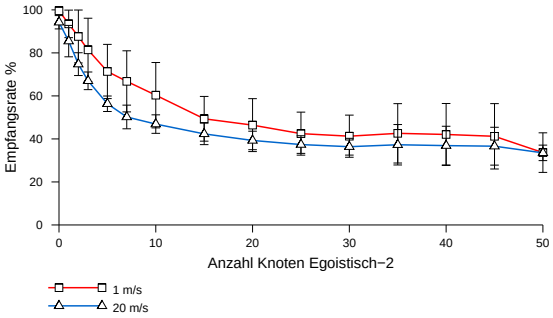

Abbildung 6.4.: Egoistisches Verhalten bei DSR – Fall 2

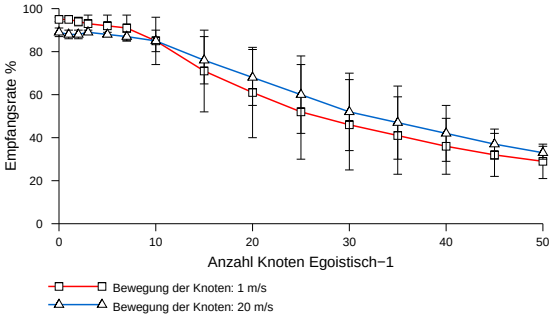

Abbildung 6.5.: Egoistisches Verhalten bei AODV – Fall 1

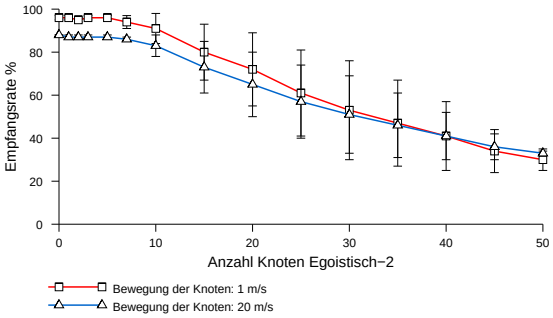

Abbildung 6.6.: Egoistisches Verhalten bei AODV – Fall 2

An dieser Stelle zeigt sich, dass der Übergang zwischen egoistischem und böswilligem Verhalten fließend ist. Ob ein Verhalten wie *Egoistisch-2* noch als egoistisch gelten kann oder schon als böswillig zu bezeichnen ist, hängt nur von der Intention des Benutzers ab.

Die Graphen zu AODV zeigen, dass sich dieses Protokoll bei Angriffen prinzipiell ähnlich verhält. Die Art des Abfalls ist etwas flacher und konstanter als bei DSR, letztendlich landen aber beide Protokolle am Ende bei einer Empfangsrate um die 30%. Da in diesem Extremfall kein Knoten mehr Verkehr weiterleitet, spielt das Routingprotokoll auch keine Rolle mehr. Die Gründe für die Abnahme der Empfangsrate sind analog zu DSR.

Zusammenfassend lässt sich sagen, dass die untersuchten MANET Routingprotokolle relativ empfindlich gegen die simulierten Angriffe sind und als Folge die Empfangsrate stark einbricht. Dabei ist dieser Befund nicht auf DSR beschränkt, sondern lässt sich auch bei anderen Protokollen wie AODV reproduzieren. Dies macht deutlich, dass ein wirksamer Schutz gegen egoistische und böswillige Knoten unbedingt notwendig ist.

6.4. Schutzmaßnahmen

Betrachtet man die Konzepte, die bisher zum Schutz von MANETs vorgeschlagen wurden, so zeigen sich im Wesentlichen drei Arten von Mechanismen, die den Schutz von MANETs leisten sollen:

1. Die Identifizierung der beteiligten Knoten.
2. Die Absicherung des Routing Protokolls gegen Manipulationen.
3. Die Verhinderung von egoistischem Verhalten durch Motivation oder Bestrafung.

Zunächst ist eine klare *Identifizierung der Knoten* notwendig, um zu verhindern, dass ein Knoten unter beliebigen, schnell wechselnden Identitäten im Netz aktiv wird. Ist dies nicht gegeben, so laufen jegliche Schutz- bzw. Strafmaßnahmen zwangsläufig ins Leere. Die Diskussion, wie eine solche Identifizierung ablaufen soll, ist noch nicht abgeschlossen. Manche Arbeiten schlagen eine zentrale oder verteile Zertifizierungshierarchie vor, wieder andere versuchen eher ein verteiltes „Web-of-Trust" in MANETs zu etablieren. Was bisher nicht betrachtet wird, sind Konzepte zur Verschleierung dieser Identitäten z.B. mittels *Pseudonymen*, was dem Schutz der Privatsphäre dient (siehe Angriffsbaum D).

Der zweite große Bereich beschäftigt sich mit der *Absicherheit der Routing Protokolle*. Hier geht es primär darum, mutwillige Modifikationen der Topologie-Informationen durch *FEB-Knoten* zu verhindern. Es kommen in der Regel kryptographische Verfahren zum Einsatz, die eine solche Veränderung verhindern sollen. Ziel muss dabei sein, den Aufwand für die Berechnung und den Transport der zusätzlichen Daten zu minimieren. Mit derartigen Verfahren lassen sich viele der Angriffe in den Angriffsbäumen A und B verhindern. Ein weiterer Aspekt, der bisher noch nicht betrachtet wurde, ist die Absicherung des späteren Datenverkehrs. Tauscht man im Rahmen des Routingprotokolls *Sitzungsschlüsseln* zwischen den Knoten aus, kann die Datenverbindung danach durch *Verschlüsselung* sicher vor Abhören und Modifikationen (siehe Angriffsbaum D) geschützt werden.

Allerdings lassen sich nicht alle Attacken verlässlich ausschließen, dies gilt insbesondere für viele egoistische Angriffe (siehe Angriffsbaum A.1.1 und A.2.1), wenn ein Knoten nicht am Routing oder der Weiterleitung teilnimmt. Daher versuchen wieder andere Sicherheitsmechanismen, *egoistisches Verhalten zu verhindern.* Dabei gibt es im Wesentlichen zwei Ansätze:

- Der *Motivations-basierte Ansatz* versucht, die Knoten zur aktiven Teilnahme zu veranlassen. Hier kommen beispielsweise virtuelle Währungen zum Einsatz, die sich ein Knoten durch aktive Teilnahme am Netz erwerben kann. Diese Währung wird im Gegenzug eingesetzt, wenn der Knoten selbst Verkehr generieren will.
- Der andere Ansatz zielt eher auf eine kooperative *Erkennung von egoistischen oder böswilligen Knoten* ab. Ist ein Knoten erkannt, wird versucht, diesen aus dem Ad hoc Netzwerk auszuschließen.

6.5. Überblick über den aktuellen Stand der Forschung

Wie sich auf [Zhu] ablesen lässt, gab es bis etwa Mitte 2000 kaum Forschung im Bereich der MANET Sicherheit. Die Anstrengungen fokussierten sich hauptsächlich auf die Entwicklung leistungsfähiger Routing-Protokolle für verschiedene Anwendungsfälle, Sicherheit wurde meist außen vor gelassen.

So ist beispielsweise im RFC zu AODV [PRD03] unter „10. Security Considerations“ zu lesen „Currently, AODV *does not specify any special security measures* ...“. Auch im Draft zum DSR Protokoll [JMHJ03] findet sich Vergleichbares:

> „11. Security Considerations
>
> This document does not specifically address security concerns. *This document does assume that all nodes participating in the DSR protocol do so in good faith and without malicious intent to corrupt the routing ability of the network.* In mission-oriented environments where all the nodes participating in the DSR protocol share a common goal that motivates their participation in the protocol, the communications between the nodes can be encrypted at the physical channel or link layer to prevent attack by outsiders.“

Ähnliches gilt für die anderen Routing-Protokolle. Manche verweisen explizit auf den Einsatz von IPsec als Sicherheitsframework. Dies wurde aber für ein ganz anderes Einsatzszenario entwickelt und wird den Anforderungen von MANETs aus verschiedenen Gründen nicht gerecht:

1. Mit dem Internet Key Exchange Protokoll *IKE* verfügt IPsec über einen äußerst komplexen, um nicht zu sagen „aufgeblasenen“ Mechanismus zum Austausch von Schlüsseln, welcher insbesondere für kleine mobile Geräte kaum geeignet ist. Außerdem verzögert dies den Verbindungsaufbau unnötig lange.
2. IPsec geht von einer etablierter und ständig verfügbaren Public Key Infrastruktur zur Validierung öffentlicher Schlüssel aus. Während dies im Internet-Kontext durchaus Sinn macht, kann davon in einem MANET nicht ausgegangen werden.

3. Schließlich werden spezifische Probleme von MANETs überhaupt nicht adressiert: das Fehlen initialer Vertrauensbeziehung, die Erstellung von Bewegungsprofilen, egoistische Knoten uvm.

Etwa seit dem Jahr 2000 haben sich verschiedene Forschergruppen dieser Problematik angenommen und entwickeln spezifische Sicherheitslösungen für Ad hoc Netze. Feng Zhu betreut eine umfassende Liste mit Literaturangaben zu Sicherheit in Ad hoc Netzwerken [Zhu]. Besonders relevant sind in diesem Zusammenhang die Aktivitäten der folgenden Forschergruppen:

- Marti, Giuli und Baker haben bereits 2000 eine Arbeit zur Erkennung von egoistischen Knoten in Ad hoc Netzen vorgestellt [MGLB00].
- Im gleichen Jahr haben N. Asokan und Philip Ginzboorg eine Arbeit über effizienten Schlüsselaustausch in Ad hoc Netzwerken veröffentlicht [AG00].
- Yongguang Zhang und Wenke Lee stellten ebenfalls 2000 eine Veröffentlichung über ein verteiltes IDS System für Ad hoc Netze vor [ZL00, ZLH03].
- Etwa zeitgleich haben Lidong Zhou und Zygmunt J. Haas eine Arbeit über eine verteilte Schlüsselverwaltung vorgestellt [ZH99]. Zur Zeit arbeitet Prof. Haas zusammen mit Panagiotis Papadimitratos am *Secure Routing Protocol (SRP)* [PH02b, PH02c, PH02d, PHS02, PH03].
- Adrian Perrig, Dave Johnson und Yih-Chun Hu von der Carnegie Mellon bzw. Rice University haben verschiedene Arbeiten zum Thema *Sicheres Routing* veröffentlicht, insbesondere das Ariadne Protokoll [HPJ02], SEAD [HJP02], TESLA Broadcast Authentication [PCTS02] und Packet Leashes [HPJ03], ein Verfahren zur Vermeidung von Tunnel-Angriffen.
- Das ARAN Protokoll [SDL+02] wird von einer Gruppe von Wissenschaftlern der University of Massachusetts, Amherst, der University of California, Santa Barbara und der Georgetown University entwickelt.
- Manel Guerrero Zapata arbeitet bei Nokia an einer sicheren Form von AODV, genannt SAODV [Gue02a, GA02].
- Im Rahmen des Terminodes-Projekt [HGBV01] entwickeln Prof. Hubaux und Mitarbeiter an der EPFL in Lausanne, Schweiz ein, dem PGP *Web-of-Trust* ähnliches, verteiltes Schlüsselmanagement [HBČ01]. In der gleichen Gruppe entstand mit den *Nuglets* ein Verfahren, welches mittels einer virtuellen Währung die Motivation zur Weiterleitung von Datenverkehr steigern will.
- Sonja Buchegger [BB02a, BB02b] von der EPFL Lausanne (früher IBM Zürich) arbeitet im Rahmen einer Dissertation ebenfalls an einem System zur Förderung der Kooperation in Ad hoc Netzwerken, welches auf Erkennung von FEB Knoten und deren Bestrafung beruht.
- Bei Eurecom arbeiten Pietro Michiardi und Refik Molva an CORE, einem System, welches auf Basis gegenseitiger Reputation die Kooperation in Ad hoc Netzen verstärken soll [MM, MM02].

Je nach ihrer thematischen Ausrichtung werden diese Arbeiten in späteren Kapiteln zu *Identitäten und Pseudonymen*, *sicherem Routing und Schlüsselaustausch* und dem *Mobile Intrusion Detection System* ausführlich vorgestellt.

6.6. Fazit

Bei der Konzeption und Planung von Mobilen Ad hoc Netzen sieht man sich einer Vielzahl von neuen Sicherheitsproblemen gegenüber. Die bisherigen Bemühungen zur Entwicklung von Routingprotokollen haben diese weitgehend ausgespart. Der mögliche Schaden durch egoistische oder böswillige Knoten ist jedoch so groß, dass der Betrieb von MANETs ohne Sicherheitsmechanismen kaum möglich erscheint. Etwa seit dem Jahr 2000 gibt es verschiedene Arbeiten, die sich mit der Absicherung von Ad hoc Netzen beschäftigen. Die verschiedenen Projekte betrachten allerdings alle isoliert einzelne Aspekte der Themen Identifizierung, sicheres Routing und Verhinderung von Egoismus.

Im Rahmen dieser Arbeit soll im Gegensatz dazu eine umfassende Sicherheitsarchitektur entwickelt werden, welche vor einer möglichst großen Zahl der geschilderten Angriffsmöglichkeiten schützt und alle genannten Bereiche adressiert. Dieses Framework wird im folgenden Kapitel übersichtsartig vorgestellt, danach folgen detaillierte Beschreibungen der einzelnen Komponenten.

7. SAM - eine Sicherheitsarchitektur für Mobile Ad hoc Netzwerke

7.1. Übersicht

Die bisherigen Ansätze und Projekte zu Sicherheit in Ad hoc Netzen adressieren immer nur einen Teil der auftretenden Sicherheitsprobleme, oft wird beispielsweise in Arbeiten zu sicheren Routingprotokollen das Vorhandensein gegenseitiger geheimer Schlüssel vorausgesetzt. Wie diese Schlüssel im Vorfeld (ohne ein funktionierendes Routing) auszutauschen sind, bleibt ungeklärt.

Im Gegensatz dazu ist es das Ziel der in dieser Arbeit vorgestellten *Sicherheitsarchitektur für Mobile Ad hoc Netzwerke* (kurz *SAM*), ausgehend von der bereits durchgeführten Sicherheitsanalyse eine umfassende und in den Teilkomponenten aufeinander abgestimmte Sicherheitslösung für Mobile Ad hoc Netzwerke zu realisieren.

Entsprechend den im letzten Kapitel beschriebenen Schutzmechanismen für MANETs gliedert sich SAM in drei Komponenten. Diese bauen aufeinander auf, wobei die gegenseitigen Abhängigkeiten und Aufgaben genau definiert sind.

1. Identifizierung und Pseudonyme
2. Sicheres Routing und Schlüsselaustausch (SDSR)
3. Mobile Intrusion Detection System (MobIDS)

Abbildung 7.1 zeigt einen Überblick über das Gesamtsystem. Die *Identifizierungskomponente* ist für eine eindeutige Identifizierung der Netzteilnehmer zuständig. Alle weiteren Komponenten bauen darauf auf. Bei Bedarf erlaubt die Komponente *Pseudonyme* die Verwendung von pseudonymen Identifikatoren, um einen direkten Rückschluss auf die kommunizierenden Benutzer oder Geräte zu verhindern. Das *SDSR* Modul verhindert, zusammen mit der *Identifizierung*, Fälschungen von Topologiedaten. Das *MobIDS* schließlich entdeckt und bestraft egoistisches Verhalten. Es hängt von einer korrekten Funktion der *Identifizierung* und des *Sicheren Routing* ab. Insbesondere geht das MobIDS davon aus, dass zwischen Knoten einer Route paarweise geheime Schlüssel vereinbart wurden. Entdeckt das *SDSR* Modul eine versuchte Fälschung von Topologiedaten und kann es diese eindeutig einem Knoten zuordnen, so kann es via *MobIDS* eine Sperrung des Knoten im MANET veranlassen.

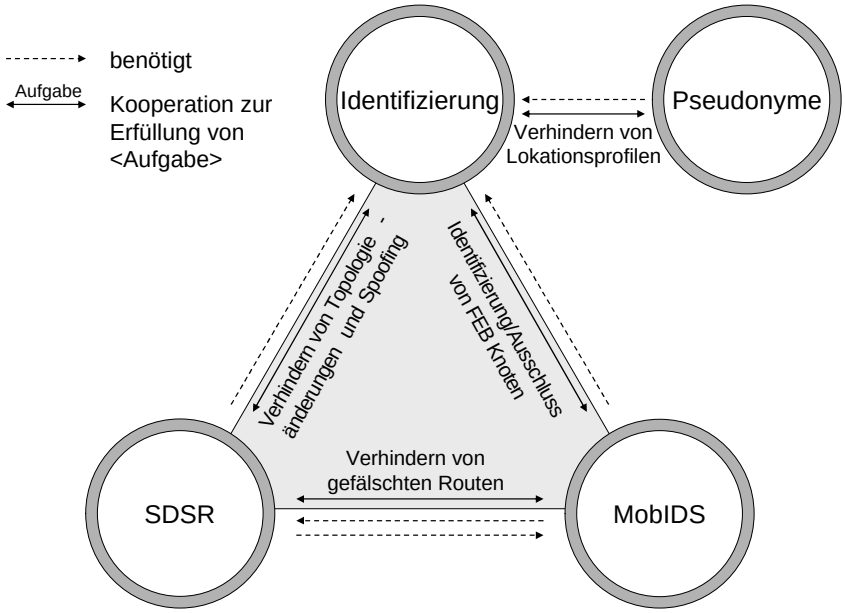

Abbildung 7.1.: Aufbau von SAM

7.2. Komponenten

Im Folgenden wird für jeden dieser Blöcke genau erläutert, von welchen Voraussetzungen die Komponente ausgeht, d.h. was sie von anderen Komponenten erwartet. Danach wird die Hauptaufgabe dieser Komponente im Sicherheitsframework erläutert. Schließlich wird unter Anforderungen beschrieben, welche Randbedingungen in Bezug auf die anderen Komponenten bei der Realisierung dieser Komponente zu beachten sind. Dabei gibt dieses Kapitel zunächst nur einen groben Überblick, damit der Leser die Komponenten besser zueinander in Beziehung setzen kann. Genaue Details liefern dann die folgenden Kapitel.

Identifizierung und Pseudonyme

Voraussetzungen: Die MANET-ID Komponente stellt den Ausgangspunkt unseres Sicherheitssystems dar und baut nicht auf den anderen Komponenten auf. Daher werden an die anderen Komponenten keine weiteren Anforderungen gestellt. Die Pseudonyme sind eine optionale Erweiterung der MANET-IDs.

Aufgabe: Diese Komponente dient dazu, die Identität der Knoten in einem MANET zweifelsfrei festzulegen. Hierzu ist zunächst zu klären, ob sich der Begriff Identität auf ein Gerät oder einen Benutzer bezieht. Es soll einem Knoten nicht möglich sein, sich selbst eine andere Identität zu generieren. Um die Erstellung von Bewegungsprofilen zu verhindern, soll ein Knoten in der Lage sein, für verschiedene Kommunikationsbeziehungen unterschiedliche Pseudonyme nutzen zu können.

Anforderungen: Alle Schritte, die zur Authentisierung und Authentifizierung dienen, müssen im MANET auch ohne Verbindung zu einer zentralen Instanz (z.B. im Internet)

möglich sein. Weiterhin sollen diese Schritte mit Rücksicht auf leistungsschwache mobile Geräte möglichst einfach gehalten sein, unnötige Kommunikation zwischen Knoten ist zu vermeiden. Außerdem ist eine Authentisierung nur durch Fluten, oder während bzw. nach einer Route Discovery möglich, da Knoten anders nicht kommunizieren können.

Sicheres Routing und Schlüsselaustausch (SDSR)

Voraussetzungen: Diese Komponente geht davon aus, dass jeder Knoten im Netz eine eindeutige Identität (z.B. in Form eines RSA-Schlüsselpaares) besitzt, beziehungsweise über eine begrenzte Anzahl von Pseudonymen zu dieser Identität verfügt.

Aufgabe: Diese Komponente definiert ein an DSR angelehntes Routing-Protokoll namens Secure-DSR (SDSR), welches in der Lage ist, vielfältige Modifikationen an den Routing-Nachrichten zu erkennen und somit zu verhindern. Gefälschte Nachrichten werden verworfen, eine Meldung an das *MobIDS* führt gegebenenfalls zum Ausschluss des Verursachers aus dem MANET. Alle an einer Route beteiligten Knoten werden authentifiziert. Eingebettet in den Prozess der Route Discovery wird zwischen dem Urheber einer Route und jedem Knoten im Pfad (inklusive dem Zielknoten) ein gemeinsamer geheimer Schlüssel vereinbart, welcher für die Verschlüsselung der nachfolgenden Datenkommunikation genutzt wird.

Anforderungen: Das im nächsten Abschnitt beschriebene MobIDS geht davon aus, dass zwischen dem Urheber einer Route und allen Knoten im Pfad je ein gemeinsamer geheimer Schlüssel vereinbart wurde. Das Vorhandensein solcher Schlüssel wird für verschiedene Sensoren benötigt. Diese Schlüssel werden im Rahmen der Route Discovery ausgehandelt. Weiterhin setzt das MobIDS voraus, dass die Identität aller an einem Pfad beteiligten Knoten geprüft wurde, was ebenfalls während der Route Discovery geschieht. Außerdem geht das MobIDS davon aus, dass ein Großteil der Topologiemodifikationen aus den Angriffsbäumen vom Routing erkannt und verhindert wird. Somit beschränkt sich das MobIDS darauf, lediglich Angriffe zu erkennen, welche vom Routing-Protokoll nicht verhindert werden können. Alle diese Funktionen müssen mit Rücksicht auf leistungsschwache, mobile Geräte möglichst einfach gehalten sein, unnötige Kommunikation zwischen Knoten oder aufwändige Rechenoperationen sind möglichst zu vermeiden.

Mobile Intrusion Detection System (MobIDS)

Voraussetzungen: Das MobIDS geht davon aus, dass Knoten in einer Route verlässlich identifiziert wurden und Modifikationen in den Routing-Daten anderer Knoten nicht möglich sind. Dies wird durch das SDSR Protokoll gewährleistet. Weiterhin geht das MobIDS davon aus, dass ein Knoten nicht unter einer großen Zahl beliebig wechselnder Identitäten am Netz teilnehmen kann, sonst ist kein wirkungsvoller Ausschluss von Teilnehmern möglich. Dafür ist die Identifizierungskomponente zuständig.

Aufgabe: Das *Mobile Intrusion Detection System* dient der Erkennung und dem Ausschluss von fehlerhaften, egoistischen oder böswilligen Knoten (FEB). Hierzu greift es auf eine Reihe von *Sensoren* zurück, welche Auffälligkeiten im Verhalten eines Knotens bemerken. Die Sensoren liefern Meldungen an den *Bewerter*, welche dieser zu einer

lokalen Bewertung zusammenführt. Anschließend verteilt der *Distributor* diese Information im Netz. Das *Ausschluss-System* sorgt dafür, dass Knoten mit einer eindeutig negativen Bewertung nicht am Netz teilnehmen können.

Anforderungen: Das MobIDS dient der Erkennung von Angriffsformen, welche durch die anderen Mechanismen nicht verhindert werden können. Hier ist primär egoistisches Verhalten zu nennen, wie es in Angriffsbaum A (Tabelle 6.1) dargestellt ist. Vor allem, wenn Knoten nicht am Routing teilnehmen oder keine Datenpakete weiterleiten, versagt das sichere Routing Protokoll, da ja keine Modifikation der Daten stattfindet. Hier muss also das MobIDS erkennen, dass ein Knoten die Kooperation verweigert. Steht dies mit hinreichender Sicherheit fest, so soll der Knoten effektiv an der zukünftigen Nutzung des Netzes gehindert werden. Dabei soll die Erkennungsgenauigkeit der Sensoren möglichst hoch sein, d.h. korrekt funktionierende Knoten sollen nur in seltenen Fällen als FEB erkannt werden, wohingegen möglichst viele FEB Knoten als solche erkannt werden sollen. Gleichzeitig gilt auch hier, dass das MobIDS selbst nur wenig Last im Netz und auf den Knoten erzeugen darf. Insbesondere zur Abschreckung von egoistischen Knoten ist nicht zwangsläufig eine 100 % Erkennungsrate notwendig. Gemäß der von John Nash begründeten Spieletheorie [Nas50] ist ein egoistisches Verhalten für einen Knoten genau dann sinnvoll, wenn der mittlere Gewinn höher liegt, als der mittlere zu erwartende Verlust. Indem man den möglichen Verlust (Ausschluss aus allen MANETs) möglichst hoch ansetzt, genügt bereits eine mittlere Entdeckungswahrscheinlichkeit, um den zu erwartenden Gewinn (Ressourceneinsparung) aufzuwiegen. Der Knoten wird also kooperieren, wenn die Wahrscheinlichkeit einer Entdeckung relativ hoch und die dann folgende Strafe entsprechend schwerwiegend ist.

7.3. Abdeckung der Angriffe

Die verschiedenen Komponenten von SAM decken verschiedene Angriffe aus den Angriffsbäumen in Abschnitt 6.2 ab. Modifikationen an der Route (z.B. A.1.1.3, A.1.2, A.1.3.2 oder B.2.2.4 bis B.2.2.7) soll SDSR durch geeignete Integritätssicherung erkennen. Angriffe, bei denen Daten abgehört werden sollen (Angriffsbaum C) kann SDSR durch Verschlüsselung der Datenverbindung mit dem, zwischen Sender und Empfänger ausgetauschten, geheimen Schlüssel verhindern.

Egoistisches Verhalten (z.B. A.1.1.1, A.1.1.2 oder A.2) soll durch MobIDS erkannt werden, genauso verschiedene Formen von DoS Angriffen (B.2.2).

Die Verwendung von Pseudonymen bei MANET-IDs erschwert Angriffe, welche auf einer Zuordnung der Knotenidentität beruhen (z.B. D.1 und D.2).

Was im Rahmen von SAM nicht betrachtet wird, sind DoS Angriffe gegen das Link-Layer Netz (B.1.1).

7.4. Fazit

Damit sind die Aufgaben und Zuständigkeiten innerhalb von SAM klar umrissen. Die folgenden Kapitel 8, 9, 10 und 11 zeigen nun, wie die Komponenten von SAM ihre jeweiligen Aufgaben erfüllen. Dabei werden die Mechanismen erläutert und es er-

folgt ein Vergleich mit den Arbeiten anderer Forschungsgruppen zu den entsprechenden Themen. Die jeweiligen Vor- und Nachteile werden ausführlich diskutiert. Daran schließt sich Kapitel 12 an, welches die Mechanismen von SAM validiert und analysiert. Insbesondere wird hier nochmals genau überprüft, welche der Angriffe aus den Angriffsbäumen in Abschnitt 6.2 durch welche Komponente verhindert wird und ob eine vollständige Abdeckung der Bäume gegeben ist.

8. Identifizierung

Dieses Kapitel dreht sich um die Identifizierung und Authentisierung in Ad hoc Netzen. Hierzu werden zunächst verwandte Arbeiten vorgestellt und analysiert. Da die bestehenden Arbeiten diese zur unzureichend adressieren, wird dann genauer auf die Fragestellung eingegangen, was eigentlich eine Identität in einem Ad hoc Netzwerk auszeichnet. Schließlich werden die MANET-IDs vorgestellt, welche einige Schwächen der bisherigen Ansätze vermeiden.

8.1. Verwandte Arbeiten

8.1.1. The Resurrecting Duckling

In ihrer Arbeit aus dem Jahr 1999 [SA99] gehen Frank Stajano und Ross Anderson der Frage nach, wie zwei Geräte sicher drahtlos kommunizieren können. Als Beispiel dient ihnen ein drahtloses Fieberthermometer, welches auf Anforderung eines anderen Gerätes, z.B. des PDAs eines Arztes, eine Temperaturmessung vornehmen und an das andere Gerät schicken soll.

Unter Sicherheitsgesichtspunkten sind dabei verschiedene Forderungen zu erfüllen. Das Thermometer wird ausschließlich vom Arzt-PDA gesteuert und der übertragene Messwert darf ausschließlich dem Arzt-PDA zugänglich sein. Manipulationen an den Geräten oder den Messwerten sind zu verhindern. Die Geräte müssen sich also gegenseitig authentisieren, der Zugriff eines Gerätes muss autorisiert sein und die Integrität und Vertraulichkeit der Nachrichten soll sichergestellt werden. Bei all dem darf die Handhabung der Geräte nicht unnötig komplex werden und bei Bedarf soll das Thermometer natürlich auch von einem anderen Arzt wiederverwendet werden können. Auch wird die in einem digitalen Thermometer zur Verfügung stehende Rechenleistung vermutlich sehr beschränkt sein, komplexe Berechnungen sind also zu vermeiden. All diese Anforderungen sind typisch für die drahtlose Kommunikation von Kleingeräten und lassen sich auf viele Szenarien verallgemeinern.

Herkömmliche Lösungen setzen in der Regel eine komplexe Administration und entsprechende Infrastruktur voraus. Im vorliegenden Beispiel ist es aber undenkbar, dass ein Arzt mit seinem PDA und dem Thermometer erst einen IT-Administrator aufsuchen muss, damit dieser die beiden Geräte so einstellt, dass sie sicher miteinander kommunizieren können. Die Herausforderung liegt also darin, zwischen beiden Geräten auf einfache und schnelle Weise eine temporäre, sichere Beziehung aufzubauen, bei der ein Gerät das andere so lange kontrollieren kann, bis die gemeinsame Aufgabe erfüllt und das gesteuerte Geräte wieder für eine neue Verbindung bereit ist.

Die Inspiration für ihre Lösung beziehen Stajano und Anderson aus den Erkenntnissen von Konrad Lorenz. Genauso wie ein Gänseküken durch den ersten optischen Eindruck

auf seine Mutter (oder jedes andere Objekt) geprägt wird und anschließend ausschließlich ihr folgt, soll auch das Thermometer zumindest temporär auf den PDA des Arztes geprägt werden und ausschließlich mit diesem interagieren.

Soll ein ungeprägtes Gerät auf ein Steuergerät wie den erwähnten PDA geprägt werden, so ist hierzu eine kurze physikalische Verbindung (z.B. über elektrische Kontakte an der Geräteaussenseite) notwendig. Hierüber überträgt das „Muttergerät“ einen symmetrischen Schlüssel an das „Kind“. Anschließend reagiert dieses nur noch auf Befehle, welche mit diesem Schlüssel kodiert wurden und verschlüsselt alle seine Daten ebenfalls mit dem gemeinsamen Schlüssel. Diese Prägung bleibt bis zum virtuellen Tod des Kindes bestehen, der entweder nach einer bestimmten Zeitspanne automatisch oder auf Anweisung der Mutter eintritt. Im vorliegenden Fall könnte auch eine bestimmte Aktion wie das Eintauchen des Thermometers in ein Desinfektionsbad für eine Löschung der Prägung sorgen.

Im Gegensatz zu einem echten Lebewesen wird das Thermometer aber nach seinem Ableben wiedergeboren und steht für eine neue Prägung bereit, daher der Titel „Resurrecting Duckling“.

Durch den direkten Kontakt lassen sich viele Probleme der Authentisierung und Authentifizierung elegant lösen. Komplexe asymmetrische Kryptographie wird ganz vermieden, der symmetrische Schlüssel wird einfach im Klartext über eine sichere Verbindung durchgeführt. Auch die Bindung des Schlüssels an das physikalische Gerät ist inhärent gegeben, es sei denn, es würde einem „Man-in-the-Middle“ gelingen, sich in die Kommunikation einzuschleusen. Das ist aber ziemlich unwahrscheinlich, da ja der Benutzer den direkten Kontakt zwischen den Geräten herstellt und die Verbindung für die kurze Zeit des Schlüsselaustauschs überwacht.

In den weiteren Ausführungen beschäftigen sich die Autoren dann noch mit der Sicherung des geheimen Schlüssels in den Geräten. Um diesen wirklich abzusichern, ist sogenannte *tamper-proof hardware* notwendig. Verschiedene Arbeiten legen allerdings den Schluss nahe, dass dies nicht mit vertretbarem Aufwand erreicht werden kann [AK96, AK97, Sch00]. Je nach Einsatzzweck mag allerdings ein simpler physikalischer Schutz der beteiligten Geräte genügen, so könnte man beispielsweise eine Plombe am Thermometer anbringen, die dessen Unversehrtheit garantieren soll.

Der bisherige Ansatz ist insofern recht unflexibel, als er auf einer reinen Zweierbeziehung beruht, welche der Mutter komplette Rechte am Kind-Gerät einräumt und alle anderen Geräte aussperrt. Komplexere technische Systeme bestehen aber aus verschiedenen Komponenten, die miteinander interagieren. So will man beispielsweise eine Stereoanlage möglicherweise mit verschiedenen Geräten steuern können. Deshalb hat Stajano seinen Ansatz im Folgejahr weiterentwickelt [Sta00] und die Möglichkeit geschaffen, dass die Mutter weiteren Geräten explizite Rechte einräumen kann.

Bewertung

Der „Resurrecting Duckling“ basiert im Wesentlichen auf dem geschützten Austausch eines geheimen Schlüssels und der folgenden Beschränkung der Kommunikation auf diese Verbindung. Die Authentisierung der beteiligten Geräte erfolgt durch direkten physikalischen Kontakt, der vom Benutzer überwacht wird. Dies schränkt den Einsatzzweck der Lösung stark ein, dafür funktioniert das vorgestellte Schema komplett ohne

weitere Infrastruktur wie vertrauenswürdige Dritte oder Schlüsselserver. Auch wird ausschließlich symmetrische Kryptographie verwendet, was den effizienten Einsatz der Lösung auch in Kleingeräten erlaubt.

Die Lösung ähnelt sowohl vom Einsatzzweck als auch von der Realisierung den Sicherheitsmechanismen von Bluetooth, wie sie in Abschnitt 4.3.1 vorgestellt wurden.

Obwohl die Autoren in ihrer Arbeit ausführlich den Fragestellungen der Authentisierung und Authentifizierung nachgehen, bleibt das Szenario beschränkt. Sofern mit einem Gerät kein Schlüssel ausgetauscht wurde, kann dessen Identität nicht überprüft werden, eine Verbindung ist nicht möglich. Ein solcher Schlüsselaustausch setzt jedoch eine physikalische Verbindung voraus. Selbst wenn diese gegeben ist, kann die Zahl der notwendigen Schlüssel in einem komplexen Netz mit vielen Teilnehmern stark ansteigen, da bei n Teilnehmern im schlimmsten Fall $\binom{n}{2} = \frac{n(n-1)}{2} = \mathcal{O}(n^2)$ Schlüssel benötigt werden.

Die Autoren gehen auch nur von einem Einsatz in einem Single-Hop Ad hoc Netzwerk aus, für ein Multi-Hop Ad hoc Netz ist die Lösung schlicht ungeeignet. Die zu Grunde liegende Idee eines Master-Slave Modells wird in MANETs kaum vorkommen, die Teilnehmer sind eher autonom und stellen den anderen Knoten bestimmte Dienste zur Verfügung, während sie selbst wieder Dienste der anderen MANET Teilnehmer in Anspruch nehmen.

Ein direkter Kontakt aller Teilnehmer zum gemeinsamen Schlüsselaustausch ist dabei weder gewünscht, noch in größeren Netzen mit entsprechender geographischer Ausdehnung überhaupt machbar.

Trotz der fehlenden Übertragbarkeit auf MANETs zeigt der Ansatz von Stajano und Anderson, dass unter bestimmten Randbedingungen auf asymmetrische Kryptographie durchaus verzichtet werden kann. Vor allem bei Netzen wie den Sensor Networks, in denen Klein- und Kleinstgeräte zum Einsatz kommen, sollte dieses Ziel energisch verfolgt werden.

8.1.2. Zertifizierungsinstanzen in MANETs

Das gängige Verfahren zur Identifizierung von Teilnehmern in einem Netzwerk kommt indes um die Verwendung von asymmetrischer Verschlüsselung nicht herum. Hierbei besitzt jeder Teilnehmer ein Schlüsselpaar, welches durch einen vertrauenswürdigen Dritten (*Trusted Third Party (TTP)*) mit der Identität des Benutzers verknüpft wird. Hierzu überzeugt sich die TTP von der Identität des Teilnehmers und erzeugt dann ein *Zertifikat*, in welchem sie die Beziehung zwischen Identität und öffentlichem Schlüssel bestätigt. Die anderen Teilnehmer verlassen sich dann auf das Urteil der TTP. Die TTP, welche die Zertifikate ausstellt, wird dabei als *Zertifizierungsstelle* oder *Certification Authority (CA)* bezeichnet. Zur Erstellung des Zertifikats kommt das Schlüsselpaar der CA zum Einsatz, es wird vorausgesetzt, dass der öffentliche Schlüssel der CA jedem Teilnehmer in verifizierter Form vorliegt.

Die Aufgabe der CA geht über die Erstellung der Zertifikate hinaus. Um langfristigen Missbrauch auszuschließen, haben Zertifikate in aller Regel eine Gültigkeitsdauer. Nach dieser müssen die Zertifikate durch die CA verlängert werden. Um ein Zertifikat vor Ende der Gültigkeitsdauer ungültig zu machen, führt die CA eine sogenannte

Certificate Revocation List (CRL), welche dem Rückruf von Zertifikaten dient und in bestimmten Abständen an alle Teilnehmer verteilt werden muss. Außerdem dient die CA oft als Verzeichnis für alle existierenden Schlüssel.

Die Idee einer zentralen CA lässt sich analog für MANETs umsetzen, dies ist aber in der Praxis mit einer Reihe von Nachteilen verbunden [LL00, YK02]. Wird die CA als zentraler Knoten im MANET oder außerhalb z.B. im Internet realisiert, so muss dieser für die Erteilung neuer Zertifikate und die Verlängerung oder den Rückruf derselben stets verfügbar sein. Wie wir in Kapitel 5 gesehen haben, kann dies durch die flexible und dynamische Struktur des Ad hoc Netzes aber kaum garantiert werden. Auch können DoS Angriffe auf die CA die Funktionsfähigkeit des Netzes stark in Mitleidenschaft ziehen. Eine Replikation der CA wäre zwar möglich, hätte aber andere negative Auswirkungen.

Erfolgt nämlich ein Angriff auf die CA, bei welchem der CA-Schlüssel kompromittiert wird, können die Angreifer von diesem Zeitpunkt an nach Belieben neue Zertifikate erstellen oder existierende zurückrufen. Das Netz ist dann nicht mehr funktionsfähig. Mit der Zahl der replizierten CAs steigt auch die Gefahr eines erfolgreichen Angriffs.

Ziel muss es also sein, die Verfügbarkeit der CA zu erhöhen, ohne gleichzeitig die Verwundbarkeit mit zu erhöhen, wozu man z.B. die Funktion der Zertifizierungsinstanz im Netz verteilen kann. Hier kommt die bereits in Abschnitt 3.6.3 vorgestellte Schwellwert-Kryptographie ins Spiel. Die nächsten beiden Arbeiten verwenden eine *partiell verteilte Zertifizierungsinstanz*, bei der eine Teilmenge der Knoten im MANET diese Aufgabe übernimmt.

In [ZH99] stellen die Autoren Lidong Zhou und Zygmunt Haas ein Konzept für eine verteilte Certification Authority in MANETs vor. Bei diesem System werden sämtliche Nachrichten mit Hilfe eines Public-Key-Verfahrens gesichert; Nachrichten können hiermit verschlüsselt oder signiert werden. Das Problem der Validierung der Schlüssel der teilnehmenden Netzknoten wird mit Hilfe einer verteilten CA gelöst, wofür ein Schwellwert-Verfahren zum Einsatz kommt. Die Idee ist hierbei, dass es wohl einige kompromittierte Knoten geben kann, diese Anzahl aber immer unter der Schwelle bleibt, die nötig wäre, um den Schlüssel der CA zu rekonstruieren.

Der private Schlüssel S der CA wird dabei auf n Knoten des MANET aufgeteilt, so dass jeder Knoten einen Teil s_i erhält. Soll nun eine Signatur erstellt werden, berechnen mindestens k der n CA Knoten eine partielle Signatur $PS(M, s_i)$. Einer der beteiligten Server kombiniert diese Teile anschließend zur endgültigen Signatur. Fallen einzelne Server aus oder sind sie vorübergehend nicht erreichbar, beeinträchtigt dies das Gesamtsystem nicht, solange jeder Knoten noch mindestens k Server erreichen kann.

Die Verfügbarkeit und Fehlertoleranz des Systems ist somit gegenüber einer einzelnen CA deutlich erhöht, ebenso die Widerstandsfähigkeit gegen gezielte Angriffe. Wie schon im Abschnitt 3.6.3 ausgeführt, empfehlen Zhou und Haas die regelmäßige Erneuerung der Teilschlüssel, um einem langfristigen Angriff standzuhalten[1]. Die genauen Details der Erneuerung werden in der Arbeit aber nicht ausgeführt.

[1] Hierbei wird von einem Angreifer ausgegangen, der nur kurzzeitig in einen Knoten einbricht, dessen Teilschlüssel kopiert und wieder verschwindet (mobile adversary). Würde der Angreifer die Software des Knotens so manipulieren, dass dieser ihm fortlaufend seine jeweils aktuellen Teilgeheimnisse übermittelt, wären die regelmäßigen Schlüsseländerungen nutzlos.

Einige weitere offene Fragestellungen bearbeiten Seung Yi und Robin Kravets in ihrer Arbeit „Key Management for Heterogenous Ad hoc Wireless Networks“ [YK02]. So wird beispielsweise darauf eingegangen, welche Knoten nun als Zertifizierungsserver arbeiten sollen. Die Autoren gehen davon aus, dass in einem MANET sehr verschiedene Geräte mit unterschiedlichem Sicherheitsniveau und unterschiedlicher Rechenleistung zum Einsatz kommen. Anhand dieser Kriterien werden die leistungsfähigsten und sichersten Serverknoten ausgewählt, die hier auch MOCA (*Mobile Certificate Authority*) heißen.

Weiterhin beschäftigen sich Yi und Kravets mit der Kommunikation zwischen den regulären und den Server-Knoten. Muss ein Knoten zur Erlangung einer Signatur k MOCA Knoten erreichen, so sind hierzu im schlimmsten Fall fünf Route-Requests notwendig, die jeweils im gesamten Netz geflutet werden. Um diesen Overhead zu reduzieren führen die Autoren das Konzept des sogenannten β-Multicasts ein. Hat ein Knoten gültige Routen zu $\beta = k + \alpha$ MOCA Knoten, so wird die Zertifizierungsanfrage per Unicast direkt an die MOCA Knoten geschickt. α ist hierbei eine Sicherheitsreserve, so dass trotz eventueller Kommunikationsprobleme trotzdem mindestens k Knoten erreicht werden. Sonst wird statt des Route Requests gleich die Zertifizierungsanfrage im Netz geflutet.

Bei den bisherigen Lösungen wird immer noch zwischen einem oder mehreren CA-Knoten und den sonstigen Knoten im MANET unterschieden. Als konsequente Fortentwicklung beschreiben die Autoren Haiyun Luo und Songwu Lu in [LL00, KZL+01, LZK+02], wie eine *vollständig verteilte Zertifizierungsinstanz* aussehen könnte.

Hierbei ist jeder Knoten im Netz auch Zertifizierungsserver und trägt einen Teil des geteilten Geheimnisses. n ist also gleich der Zahl der Netzknoten. Benötigt ein Knoten nun eine Signatur, so wählt er einfach k beliebige Knoten[2] in seiner Nachbarschaft aus, an welche er seine Zertifizierungsanfrage stellt. Neue Knoten werden in die Zertifizierungsinfrastruktur integriert, indem bei Eintritt ins Netz eine neue Zertifikatsverteilung mit dann $n = n + 1$ Knoten stattfindet.

Yi und Kravets schlagen vor, dass die CA-Knoten eine Zertifizierungsanfrage nur dann bearbeiten, wenn sie den Knoten für vertrauenswürdig halten. Dies soll durch externe Beobachtungen ermöglicht werden, auf die in den Arbeiten nicht weiter eingegangen wird. Da die Zertifikate nur eine bestimmte Gültigkeit haben, muss ein Knoten sein Zertifikat nach einer bestimmten Zeit verlängern lassen, was die CA nur „gutartigen“ Knoten gewährt. Somit werden FEB-Knoten nach einer bestimmten Zeit aus dem Netz entfernt. Wie man zu einer entsprechenden Bewertung eines Knoten gelangen kann, wird in Kapitel 11 ausführlich untersucht.

Bewertung

Inwieweit eignen sich die vorgestellten CA-Konzepte für den Einsatz für das von uns in Kapitel 6 zu Grunde gelegte Szenario eines öffentlichen Mobilen Ad hoc Netzes?

Eine klassische zentrale Zertifizierungsstelle scheint für den Einsatz in MANETs wenig geeignet, da eine dauerhafte Verbindung der Knoten zur CA vom Netz nicht garan-

[2] eigentlich genügen auch $k - 1$ Knoten, da ja der Knoten selbst auch zur Signatur beitragen kann

tiert werden kann. Somit können Verlängerungen oder der Rückruf von Zertifikaten manchmal unmöglich sein.

Zunächst erscheint der Einsatz einer verteilten Zertifizierungsstelle für MANETs diese Probleme zu lösen. Man kommt ohne jede Infrastruktur aus und ein Angreifer kann, bei regelmäßiger Neuverteilung der Schlüssel, nur schwer in den Besitz des gesamten CA-Schlüssels gelangen. Somit sind Fehlertoleranz, Verfügbarkeit und Schutz des geheimen CA-Schlüssels gegenüber der Lösung mit einer CA deutlich erhöht.

Bei genauer Analyse kommt man jedoch zu dem Ergebnis, dass diesen Vorteilen auch gravierende Nachteile gegenüberstehen.

So kommt man keineswegs ohne eine zentrale Instanz aus, zumindest nicht bei der Initialisierung des Netzes. Diese muss die initialen Schlüssel verteilen und vor allem verifizieren. Erst danach funktioniert die verteilte CA. Ein wichtiger Nachteil ist auch die notwendige Rechenleistung. Verglichen mit anderen Krypto-Algorithmen sind asymmetrische und Schwellwert-Kryptographie sehr rechenaufwändig. Insbesondere bei letzterer potenziert sich dieser Nachteil dadurch, dass bei jeder Berechnung mindestens k Knoten beteiligt sind. Auch der Kommunikationsaufwand steigt dadurch stark an. Schließlich sprechen noch weitere Probleme gegen einen allzu leichtfertigen Einsatz der Schwellwertkryptographie.

Eine Frage ist beispielsweise, wie die Parameter k und n gewählt werden sollen. Wählt man k klein, so sinkt die Sicherheit des Systems und einem Angreifer gelingt es leichter, k Knoten unter seine Kontrolle zu bekommen. Je höher man k wählt, desto mehr steigt aber der Rechen- und Kommunikationsoverhead im Netz an. Die Wahl von n beschränkt natürlich zunächst die Größe von k. Ist k nahe bei n, so sinkt die Verfügbarkeit, da dann im Extremfall alle CA-Knoten erreichbar sein müssen. Somit sollte also n deutlich größer als k sein. Ein zu großes n bedeutet aber vor allem beim Neuverteilen der Teilschlüssel wieder einen größeren Overhead. Ist n klein im Vergleich zur Gesamtgröße des Netzes, wird jeder CA-Knoten sehr häufig in Anspruch genommen, was zu einer starken Belastung dieser Knoten führen wird.

Im Idealfall sollten also k und n dynamisch der Netzgröße und den aktuellen Sicherheitsanforderungen angepasst werden. Wie dies zu geschehen hat und wie die Auswirkungen im Sinne von Overhead und Belastung der Knoten sind, ist aber bisher nicht hinreichend untersucht worden.

Vor allem ist die Frage noch unbeantwortet, was im Falle von Netzpartitionierungen und einer späteren Wiedervereinigung zu geschehen hat bzw. wie die Kombination zweier separater MANETs abzulaufen hat. Die Netze haben vielleicht ganz unterschiedliche Schlüssel und Zertifikate, zumindest ist aber die Verteilung unterschiedlich. Die Netze müssten ihre CAs dann „irgendwie“ vereinigen, um die Funktionsfähigkeit des Netzes zu erhalten. Was passiert, wenn sich bei einem Split in keinem der Teilnetze k CA-Knoten mehr befinden? Und wie sieht die Verteilung der Schlüssel in einem kleinen MANET mit nur zwei Knoten aus? In diesem Fall versagen die beschriebenen Ansätze völlig.

All diese Fragen sind noch ungelöst, eine verteilte CA für MANETs ist also noch weit von einer praktischen Realisierbarkeit entfernt.

8.1.3. Selbstorganisierende Infrastruktur (Web of Trust)

Einen anderen Weg zur Zertifizierung von Schlüsseln geht die Verschlüsselungssoftware *Pretty-Good-Privacy (PGP)* [PGP03]. Hier gibt es keine zentrale Instanz, welche die Beziehung zwischen einem öffentlichen Schlüssel und der Identität seines Besitzers zertifiziert. Vielmehr kann jeder Teilnehmer andere Schlüssel signieren. Der Signaturgeber S drückt mit seiner Signatur seinen Glauben aus, dass der öffentliche Schlüssel PK_A tatsächlich zu der behaupteten Identität A gehört.

Obwohl dies technisch das gleiche ist, wie ein Zertifikat, ist die dahinter liegende Aussage eine fundamental andere. In einer PKI gibt es eine zentrale Instanz, deren Urteil alle Teilnehmer vertrauen. Zertifiziert die CA einen Schlüssel, so wird dieser Schlüssel von allen Teilnehmern im Netz als gültig angesehen.

Anders bei PGP: eine Signatur sagt hier lediglich aus, dass S der Meinung ist, dass PK_A zu A gehört. Ob sich ein anderer Knoten X dieser Meinung anschließt, hängt ganz maßgeblich vom Vertrauen ab, das X in die Fähigkeit von S hat, eine solche Aussage gewissenhaft und korrekt zu treffen.

Neben den Signaturen, welche die Authentizität von Schlüsseln ausdrücken, kommt also noch ein weiteres Konzept ins Spiel: Das Vertrauen in einen anderen Knoten, korrekte Signaturen auszustellen. Eine Signatur eines Knotens S, dem ein Knoten A nicht vertraut, ist für A wertlos. Vertraut A jedoch dem Urteil von S, dann wird er einen von S unterschriebenen Schlüssel als authentisch ansehen.

Vertrauen lässt sich auch transitiv übertragen. Vertraut A dem Urteil von B bedingungslos und B vertraut C, so wird A eine Signatur von C unter einem Schlüssel des Knotens D akzeptieren. Trägt man diese Vertrauensbeziehungen als Graph auf, so entsteht das sogenannte „Web of Trust“.

Kritisch anzumerken ist an dieser Stelle, dass bereits ein falsches Urteil zum Bruch von sehr vielen Vertrauensketten führen kann, insbesondere wenn zentrale Knoten im Web of Trust betroffen sind.

Die Arbeitsgruppe von Prof. Hubaux an der EPFL Lausanne hat in mehreren Arbeiten ein Konzept vorgestellt, welches die Idee des *Web of Trust* auf Mobile Ad hoc Netze überträgt [HBČ01, ČBH02]. Eines der Kernprobleme ist die Verteilung und Speicherung der öffentlichen Schlüssel und der Zertifikate. PGP löst dies mit zentralen Schlüsselservern, was für MANETs keine befriedigende Lösung darstellt.

Hubaux et al. schlagen einen anderen Weg vor. Dazu werden die Benutzer und Zertifikate zunächst als gerichteter Graph $G(V, E)$ aufgefasst. Die Knoten V repräsentieren dabei die Benutzer, die Kanten E stellen die Zertifikate dar. Dabei gilt

$$AB \in E \Leftrightarrow \text{A hat Schlüssel von B signiert}$$

Die Kante AB ist also genau dann in E, wenn A ein Zertifikat für den Schlüssel von B ausgestellt hat. Wenn in einem solchen Graphen ein Weg von einem Startknoten S zu einem Zielknoten D existiert, also $S \rightsquigarrow_G D$, dann kann S das Zertifikat von D über die transitiven Vertrauensbeziehungen der Zwischenknoten verifizieren.

Jeder Knoten speichert nun eine begrenzte Auswahl weiterer Zertifikate, wobei die gesamte zu speichernde Datenmenge bei geeigneter Auswahl relativ klein bleibt. Will

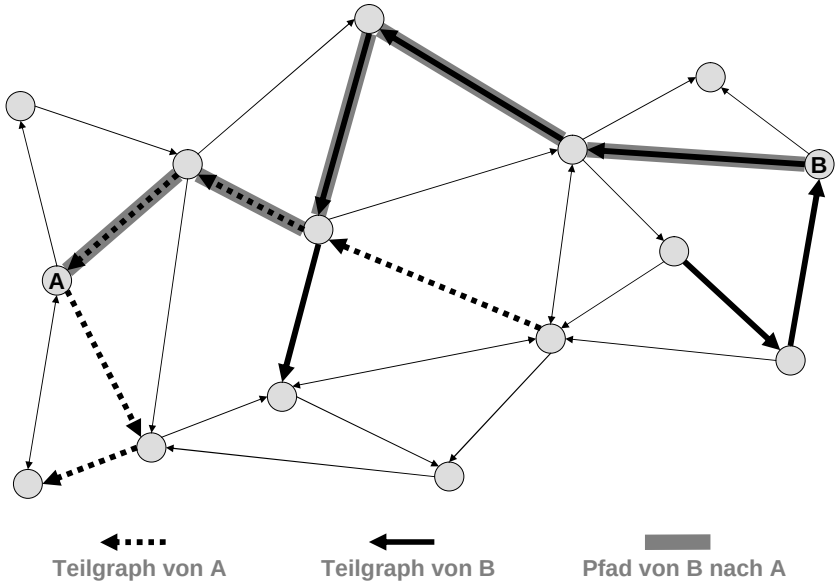

Abbildung 8.1.: Web-of-Trust: Vereinigung der Zertifikatsgraphen (nach [HBČ01])

S mit D kommunizieren, so tauschen beide Knoten in einem ersten Schritt ihre Zertifikatgraphen G_S und G_D aus und bilden jeweils die Vereinigung $G = G_S \cup G_D = G(V_S \cup V_D, E_S \cup E_D)$. Danach suchen beide Knoten je einen Weg $S \rightsquigarrow_G D$ und $D \rightsquigarrow_G S$. Existieren diese Wege, dann haben sich beide Knoten von der Authentizität des öffentlichen Schlüssels des Kommunikationspartners überzeugt. Abbildung 8.1 verdeutlicht dieses Vorgehen.

Die Auswahl der Zertifikate, die ein jeder Knoten speichert, muss also so erfolgen, dass jeder Pfad, der im Gesamtgraphen validiert werden kann, möglichst auch in einer Vereinigung zweier Teilgraphen validiert werden kann. Ein entsprechender Auswahlalgorithmus $\mathcal{A}$ sollte diesbezüglich eine möglichst gute *Performance* p aufweisen, wobei

$$p_{\mathcal{A}}(G) = \frac{\#\{(u,v) \in V \times V : u \rightsquigarrow_{S_{\mathcal{A}}(G,u) \,\cup\, S_{\mathcal{A}}(G,v)} v\}}{\#\{(u,v) \in V \times V : u \rightsquigarrow_G v\}}$$

und $S_{\mathcal{A}}(G,u)$ derjenige Teilgraph von G ist, den der Algorithmus $\mathcal{A}$ für den Knoten u aus G auswählt.

Die Autoren schlagen vor, dass jeder Knoten zunächst alle Zertifikate speichert, die er selbst ausstellt. Somit wird sichergestellt, dass jedes Zertifikat auf jeden Fall einmal gespeichert wird. Zusätzlich werden weitere Zertifikate gespeichert, die von anderen Knoten ausgestellt wurden. In [HBČ01] wird hierzu der sogenannte *Shortcut-Hunter Algorithmus* vorgeschlagen, in einer späteren Veröffentlichung [ČBH02] ist dies der *Maximum Degree Algorithmus*, der sich jedoch nicht wesentlich von Shortcut-Hunter unterscheidet.

Beide nutzen das von Stanley Milgram entdeckte und von Duncan Watts und Steven Strogatz näher untersuchte „Small Worlds-Phänomen“ [Mil67, WS98] aus. Dieses besagt, dass in vielen natürlichen oder künstlichen Netzen, die aus lokal eng verbundenen

Clustern und nur wenigen weiterreichenden Verbindungen bestehen, der kürzeste Pfad zwischen zwei beliebigen Knoten im Durchschnitt trotzdem erstaunlich kurz ist. Verantwortlich sind dafür hauptsächlich die Abkürzungen oder *shortcuts*; als solche werden Kanten bezeichnet, nach deren Entfernen der kürzeste Verbindungspfad zwischen ihren Endpunkten mindestens die Länge drei hat.

Der *Shortcut-Hunter Algorithmus* versucht nun, gezielt diese Abkürzungen zu finden, indem er genau solche Kanten bevorzugt in den Teilgraph eines Knotens aufnimmt.

Ein Problem des Ansatzes von Hubaux et al. stellen, wie schon angedeutet, falsche Zertifikate dar. Wurde in einem Pfad von S nach D ein Zertifikat falsch ausgestellt, so wird die gesamte Verifikation wertlos. Um dem gegenzusteuern, führen die Autoren eine sogenannte Authentifizierungsmetrik ein, die ein Maß dafür darstellt, mit welcher Sicherheit ein Schlüssel mit einem gegebenen Zertifizierungsgraphen authentifiziert werden kann. In diese Metrik können z.B. die Anzahl der Pfade von S nach D, die mittlere Pfadlänge oder die Länge des kürzesten Pfades einfließen.

Bewertung

Der große Vorteil des Web-of-Trust Ansatzes ist der vollkommene Verzicht auf jegliche Infrastruktur oder administrative Organisation. Das System ist komplett selbstorganisierend und kann allein durch die Benutzer aufgebaut werden.

Leider gibt es bei diesem Verfahren keinerlei Garantien, dass zwei Benutzer A und B ihre Schlüssel tatsächlich verifizieren können. Unter Umständen sind schlicht nicht genug Schlüssel signiert und entsprechend existiert gar kein Pfad zwischen den beiden Knoten. Dies dürfte gerade am Anfang eines Ad hoc Netzes eher die Regel denn die Ausnahme sein [Fok02]. Die einzige Lösung besteht im Hinzufügen von zusätzlichen Kanten, d.h. der Signatur weiterer Schlüssel. Dies ist aber immer ein manueller Vorgang, der viel Zeit in Anspruch nehmen kann. Solange könnten dann A und B schlicht nicht miteinander kommunizieren.

Diese manuelle Signatur ist ein weiterer Kritikpunkt. Sollen die Benutzer tatsächlich andauernd neue Schlüssel signieren? Und nach welchen Kritikpunkten sollen sie hier vorgehen? Eigentlich drückt eine Signatur ja nur eine Beziehung zwischen einem öffentlichen Schlüssel und einer Identität aus. Dies könnte ein Netzteilnehmer durch Prüfung des Schlüssels und des Personalausweises einer Person verifizieren. Im System von Hubaux ist die Bedeutung einer Signatur aber in Wahrheit viel weitreichender. Sie bescheinigt nämlich gleichzeitig die Fähigkeit, andere Knoten gewissenhaft zu prüfen und gegebenenfalls zu signieren. Die Konzepte der Schlüsselverifikation und des Vertrauens in einen Schlüsselbesitzer zur korrekten Erstellung weiterer Zertifikate werden hier also vermischt. Bei Systemen wie PGP sind diese Konzepte hingegen aus gutem Grund strikt getrennt. Bei genauer Betrachtung setzt Hubaux also die Ehrlichkeit und Gewissenhaftigkeit aller Benutzer voraus. Dann muss man sich aber die Frage stellen, wieso dann überhaupt noch ein Sicherheitssystem notwendig ist.

Außerdem ist es sehr bedenklich, den Benutzer in so zentraler Weise in das Sicherheitssystem einzubinden. Die meisten Teilnehmer eines MANETs werden schwerlich verstehen, aus welchem Grund sie wann wem einen Schlüssel signieren sollen. Das sind denkbar schlechte Voraussetzungen, wenn es um die gewissenhafte Durchführung einer Identitätsprüfung geht.

Noch ein kritischer Kommentar zum Shortcut-Hunter Algorithmus (siehe auch [Fok02]): zur Bestimmung der Teilgraphen müssen unter anderem die Knoten mit den meisten Abkürzungen in der Nachbarschaft eines Knotens gefunden werden. Hierzu ist Wissen um den Aufbau des Graphen im Umfeld eines Knotens notwendig, welches im Netz kommuniziert werden muss. Dabei kann ein beträchtlicher Overhead entstehen, insbesondere weil die Teilgraphen in regelmäßigen Abständen aktualisiert werden sollten. Die Autoren sind sich dieses Nachteils durchaus bewusst:

> We admit that this initialization phase is relatively expensive (in terms of bandwidth and time), but it must be performed rarely. [...] We should note that the local repositories become obsolete if a large number of certificates are revoked, as then the certificate chains are no longer valid; the same comment applies in the case when the certificate graph changes significantly (e. g., a large number of new users join the system). [ČBH02, Abschnitt 1]

Die Idee, das „Web-of-Trust“ Konzept auf Ad hoc Netze zu übertragen ist sicher interessant und eine eingehende Untersuchung wert. Aus den bisherigen Ergebnissen lässt sich aber der Schluss ziehen, dass signifikante Nachteile einem praktischen Einsatz im Wege stehen. Insbesondere die Frage nach der Bedeutung und der Erstellung von Signaturen ist gänzlich unbeantwortet.

8.1.4. Kryptobasierte Identitäten

In ihrer Arbeit „Bootstrapping Security Associations for Routing in Mobile Ad-Hoc Networks“ [BEGA02] übertragen die Autoren Rakesh Bobba, Laurent Eschenauer, Virgil Gligor und William Arbaugh das Prinzip der *kryptobasierten Identitäten* auf MANETs. Sie bauen dabei auf der Arbeit von Gabriel Montenegro und Claude Castelluccia zu statistisch einmaligen und kryptographisch verifizierbaren (SUCV[3]) Identitäten und Adressen [MC02] auf, die dort zur Absicherung der binding updates bei Mobile IPv6 eingesetzt werden.

Bei MANETs hat man das Problem, dass beim Route-Discovery Prozess eine Authentifizierung der beteiligten Parteien und eine Integritätssicherung der Nachrichten notwendig sind. Hierzu benötigen die teilnehmenden Knoten aber verifizierte Schlüssel. Zur Verteilung der Schlüssel werden aber wiederum funktionsfähige Routen benötigt. Ein klassisches Henne-Ei-Problem. Die Lösung, Schlüssel im Vorfeld offline zu übertragen, kann in Ad hoc Netzen kaum überzeugen. Eine andere Idee, die wir im späteren Verlauf verfolgen, überträgt die Schlüssel im Rahmen des Routingprotokolls.

Die Idee der kryptobasierten Identitäten beruht im Gegensatz dazu darauf, dass man die Identitäten und Adressen eines Netzteilnehmers direkt aus dessen öffentlichem Schlüssel ableitet. Somit wird eine direkte Überprüfung möglich, ob der Schlüssel zur Identität und Adresse des Absenders eines Datenpakets passt. Kein fremder Teilnehmer kann sich unter einer fremden ID oder Adresse im Netz bewegen, da ihm der zu dieser ID/Adresse gehörende private Schlüssel fehlt.

Bobba et al. leiten dazu die Identität und die Adresse eines Knotens mittels einer kryptographischen Hashfunktion aus seinem öffentlichen Schlüssel ab. Der öffentliche

[3] statistically unique and cryptographically verifiable

Schlüssel trägt dabei kein Zertifikat einer CA oder ähnliches. Die *kryptobasierte Identität* (*CBI*) ist dabei ein 128 Bit langer Hashwert des öffentlichen Schlüssels, während sich die *kryptobasierte Adresse* (*CBA*) aus einem 64 Bit langen Netzwerkpräfix und einem 64 Bit langen Hashwert des Schlüssels zusammensetzt. Sie kann somit direkt als IPv6-Adresse [DH98] verwendet werden.

$$\text{CBI} = H_{128}(PK) \qquad \text{bzw.} \qquad \text{CBA} = \text{NetzID}_{64} : H_{64}(PK)$$

Die resultierenden Werte erfüllen die Eigenschaft, dass sie *statistisch einmalig* und *kryptographisch verifizierbar* sind. Die statistische Einmaligkeit sichert, dass zufällige Identitäts- und Adresskollisionen nur mit extrem kleiner Wahrscheinlichkeit auftreten. Durch die kryptographische Verifizierbarkeit kann ein Knoten, der eine bestimmte Identität oder Adresse verwendet, die Rechtmäßigkeit beweisen, indem er beispielsweise mit dem geheimen Schlüssel des Schlüsselpaares eine digitale Signatur erzeugt.

Ein Angreifer, der eine fremde Identität annehmen will (Maskerade), müsste solange Public-Key Schlüsselpaare erzeugen, bis er zufällig auf einen öffentlichen Schlüssel trifft, welcher die gleiche Identität/Adresse erzeugt wie die des Opfers. Man mag einwenden, dass der 64 Bit lange Hashwert der CBA relativ kurz ist und deshalb eine solche Suche nicht lange dauern würde. Hierzu muss man aber berücksichtigen, dass dabei pro Versuch jeweils ein vollständiges Public-Key Schlüsselpaar (z.B. RSA) generiert werden muss, was ein sehr rechenaufwändiger Prozess ist. Dies schränkt die Suchgeschwindigkeit stark ein.

Die Autoren setzen die CBA/CBI zur Absicherung der Route Discovery von DSR ein. Dabei sei CBA_S die Adresse der Quelle S und CBA_D die Adresse des Ziels D eines Route-Requests, ID eine eindeutige ID und PK_S der öffentliche Schlüssel von S. Jetzt schickt S einen RREQ und signiert darin CBA_S, CBA_D und ID. Außerdem wird noch der öffentliche Schlüssel PK_S angefügt.

$$\text{RREQ},\ E_{SK_S}(\text{CBA}_S,\ \text{CBA}_D,\ \text{ID}),\ (),\ PK_S$$

Jeder Knoten, der diese Nachricht empfängt, hängt seine eigene Adresse an die Liste der Zwischenknoten an, so dass D am Ende eine RREQ Nachricht empfängt, die etwa wie folgt aussieht:

$$\text{RREQ},\ E_{SK_S}(\text{CBA}_S,\ \text{CBA}_D,\ \text{ID}),\ (1,2,3),\ PK_S$$

Jetzt kann D zunächst überprüfen, ob die niederwertigen 64 Bit der Absenderadresse gleich $H_{64}(PK_S)$ sind. Wenn nicht, kann das Paket direkt verworfen werden. Als nächstes prüft D die Signatur, um sicherzustellen, dass der Route-Request authentisch ist, d.h. er stammt wirklich von CBA_S, ist wirklich für CBA_D und hat die angegebene ID.

Daraufhin schickt D einen Route Reply zurück, welcher die Absender- und Zieladresse, die ID, und den öffentlichen Schlüssel PK_D enthält.

$$\text{RREP},\ E_{SK_D}(\text{SR}(3,2,1),\ \text{CBA}_S,\ \text{CBA}_D,\ \text{ID},\ (1,2,3)),\ PK_D$$

Bei Empfang dieser Nachricht prüft S wieder, ob die unteren 64 Bit der Adresse CBA_D und $H_{64}(PK_D)$ übereinstimmen und ob die Signatur korrekt ist. Trifft das zu, dann ist die Nachricht authentisch und S kann den Route Reply akzeptieren.

Wie man sieht sind die beiden Parteien also in der Lage, sich gegenseitig zu authentisieren, ohne dabei Zertifikate zu verwenden.

Bewertung

In ihrer Arbeit zu kryptobasierten Identitäten in Ad hoc Netzen zielen die Autoren primär darauf, wie man Knoten und Nachrichten in einem Netz ohne die Übertragung von Schlüsseln über sichere Kanäle authentifizieren kann.

Dieses Ziel wird erreicht; das beschriebene Verfahren ermöglicht zwei Knoten den Aufbau einer authentifizierten Verbindung, ein Knoten kann sich also nicht als ein anderer Knoten ausgeben. Auch kommt das Verfahren gänzlich ohne zentrale oder dezentrale Infrastruktur aus.

Allerdings weiß ein Knoten nach dem Authentifizierungsprozess lediglich, dass er mit einer bestimmten CBA/CBI kommuniziert. Welche Person oder welches Gerät sich hinter CBA_D verbirgt, geht daraus nicht hervor.

Bei genauer Betrachtung dient nämlich der öffentliche Schlüssel gleichzeitig als Identität und als Adresse im Netz, durch die eingesetzten Hashfunktionen wird dieser lediglich etwas gekürzt und man spart sich somit etwas Bandbreite. Dieser öffentliche Schlüssel sagt aber nichts über den Schlüsselerzeuger aus. Aufgrund der Größe des Schlüsselraumes ist der öffentliche Schlüssel statistisch einmalig[4], so dass ein Netzteilnehmer bei wiederholtem Kontakt mit einem Knoten, der mehrfach den gleichen öffentlichen Schlüssel benutzt, sicher sein kann, dass es sich immer um den gleichen Kommunikationspartner handelt.

Jeder Knoten kann sich beliebig viele neue Identitäten generieren, indem er einfach neue RSA-Schlüssel erzeugt. Wie wir jedoch bereits festgestellt haben, ist es insbesondere für den Ausschluss von FEB-Knoten wichtig, dass es pro Knoten eine eindeutige und unveränderbare Identität gibt. Sonst kann jeder Teilnehmer nach dem Ausschluss dem Netz sofort wieder mit einer neuen Identität beitreten.

Neben diesen grundsätzlichen Problemen beim Einsatz kryptobasierter Identitäten, hat die vorgestellte DSR Variante noch eine Reihe weiterer Schwächen. Die Route ist während des Flutens des Route-Requests nicht geschützt, d.h. jeder Zwischenknoten kann sie beliebig manipulieren. Das ermöglicht einige der in den Angriffsbäumen vorgestellten Attacken[5]. Weiterhin müssen diverse Optimierungen des DSR Protokolls (z.B. das Caching von Routen) entfallen.

Nichtsdestotrotz ist die Idee der kryptobasierten Identitäten in MANETs sehr interessant. Bei MANET-IDs werden CBAs zur Generierung von Adressen aus MANET-IDs verwendet.

[4] solange der Besitz des zugehörigen privaten Schlüssels nachgewiesen wird
[5] z.B. Blackhole Routing B.3.1.3

8.1.5. Schlüsselverteilung mit identitätsbasierter Kryptographie

Aus der gleichen Arbeitsgruppe wie die kryptobasierten Identitäten stammt eine Arbeit [KKA03] der Autoren Aram Khalili, Jonathan Katz und William Arbaugh, welche sich mit dem Einsatz von identitätsbasierter Kryptographie in MANETs beschäftigt, um damit eine sichere Schlüsselverteilung zu realisieren. Zusätzlich kommt Schwellwertkryptographie zum Schutz der Schlüsselverteilserver zum Einsatz.

Identitätsbasierte Kryptographie ermöglicht es, einen beliebigen Bitstring als öffentlichen Schlüssel eines Public-Key-Kryptoverfahrens zu verwenden. Im konkreten Fall dient also die Knoten-Identität direkt als Schlüssel. Zertifikate sind damit überflüssig, da sich aus dem Schlüssel auch direkt dessen Besitzer ableiten lässt.

Könnte sich ein Benutzer zu seiner Identität (und damit zu seinem öffentlichen Schlüssel) selbst den passenden geheimen Schlüssel generieren, so könnte das auch jeder Angreifer, der die Identität des Opfers kennt. Um das zu vermeiden, setzen die bekannten Verfahren zur identitätsbasierten Kryptographie einen vertrauenswürdigen Dritten voraus, welcher die geheimen Schlüssel für die Knoten generiert. Dieser wird *private key generation service* oder kurz PKG genannt.

Die Autoren Khalili et al. setzen dabei auf ein relativ neues identitätsbasiertes Verschlüsselungsverfahren von Dan Boneh und Matthew Franklin [BF01]. Hierbei besitzt die PKG ein Public-Key Schlüsselpaar PK_{PKG}/SK_{PKG}. Will ein Knoten eine Nachricht an einen anderen Knoten verschlüsseln, verwendet er hierzu PK_{PKG} in Kombination mit seiner Knoten-Identität ID, also $(PK_{PKG}, ID) \Longrightarrow PK_{ID}$. Die PKG generiert aus SK_{PKG} und ID den geheimen Schlüssel SK_{ID}, welcher zur Entschlüsselung der mit PK_{ID} verschlüsselten Nachrichten verwendet werden kann.

Khalili et al. nutzen nun die Schwellwertkryptographie, um die Funktion der PKG auf alle Knoten im MANET zu verteilen. Kommt ein neuer Knoten A in das MANET, wird auch dieser in die Verteilung der PKG aufgenommen. Zuvor muss er aber erst mindestens k andere Knoten kontaktieren, die dann für ihn seinen persönlichen geheimen Schlüssel generieren und ihm zuschicken.

Da zu diesem Zeitpunkt (mangels Schlüssel) aber weder eine sichere Kommunikation noch ein abgesichertes Routing zur Verfügung stehen, müssen sich die k Knoten alle in Kommunikationsreichweite von A befinden. Alternativ muss sich A so lange durch das Netz bewegen, bis er mindestens k Knoten getroffen und mit diesen kommuniziert hat.

Bewertung

Ziel der Arbeit von Khalili et al. ist die sichere Verteilung von Schlüsseln in einem Ad hoc Netz. Da identitätsbasierte Kryptographie hierzu in aller Regel einen vertrauenswürdigen Dritten benötigt, welcher die geheimen Schlüssel der Knoten generiert, schlagen die Autoren vor, den PKG-Server mittels Schwellwertkryptographie im Netz zu verteilen.

Der Ansatz ist ähnlich der bereits vorgestellten Arbeit von Zhou und Haas [ZH99],

wobei im einen Fall ein verteilte vertrauenswürdige Instanz Zertifikate ausstellt, im anderen Fall geheime Schlüssel generiert[6].

Und genau wie Zhou und Haas keine Angaben darüber machen, wann einem Knoten ein Zertifikat ausgestellt wird, geben auch Khalili et al. keine Auskunft darüber, nach welchen Kriterien ein Knoten authentifiziert werden soll, wenn er einen geheimen Schlüssel anfordert.

Man könnte natürlich versuchen, rein zufällige Identitäten zu vergeben, wobei dann sichergestellt sein muss, dass jede Identität nur genau einmal verwendet wird. Das führt aber bei Verwendung der Schwellwertkryptographie zu Problemen, wenn zwei Knoten parallel eine Anfrage nach einem geheimen Schlüssel zu demselben öffentlichen Schlüssel an zwei disjunkte Teilmengen von je k PKG Knoten schicken. Zufällige Identitäten enthalten aber überhaupt keine Informationen über den Netzteilnehmer. Selbiges gilt für die SUCV-Identitäten, wie sie im vorigen Kapitel vorgestellt wurden.

Somit lösen Khalili et al. zwar das Problem der Verteilung und Authentisierung von Schlüsseln, die Authentisierung von Geräten oder Benutzern gelingt damit aber nicht. Und auch die Problematik von wechselnden Identitäten wird von den Autoren vernachlässigt.

8.2. Identitäten in Ad hoc Netzen

8.2.1. Schwächen bisheriger Ansätze

Unabhängig von technischen Details fällt auf, dass alle vorgestellten Lösungen ein grundlegendes Problem aufweisen. Sie beschäftigen sich nicht oder nur unzureichend mit der Frage, was eine Identität in einem MANET ist. Vielmehr stellen viele der Lösungen lediglich sicher, dass ein öffentlicher Schlüssel oder eine Zufalls-Identität in einem MANET nur einmal verwendet werden kann. Wie aber dieser Schlüssel oder diese Zufallszahl einem realen Benutzer oder einem realen Gerät zugeordnet wird, bleibt völlig offen.

Es erscheint also dringend geboten, zunächst der Frage nachzugehen, was denn eine Identität in einem Ad hoc Netz auszeichnet, bevor man sich die Frage stellt, wie eine solche Identität verifiziert (also authentisiert) werden kann.

Ein Rechnernetz besteht aus einer Anzahl unabhängiger, miteinander kommunizierender Computer [Tan96]. Neben einer Broad- oder Multicast-Kommunikation muss dabei die Möglichkeit gegeben sein, einem bestimmten Knoten gezielt eine Nachricht zuzuschicken. Aus diesem Grund haben Knoten in einem Netzwerk in der Regel eine netzwerkweit eindeutige *Adresse*.

Gelegentlich hat ein Knoten jedoch mehrere Interfaces, mit je einer Adresse pro Interface. Um den Knoten dennoch eindeutig zu bezeichnen, führen Routing-Protokolle hierzu in aller Regel eine *Routing ID* ein [Moy91], die z.B. die kleinste der an die Interfaces vergebenen Adressen sein könnte. In einem anderen Fall bekommt ein Knoten über eine bestimmte Zeit möglicherweise unterschiedliche Adressen dynamisch zugewiesen.

[6] Khalili et al. sehen allerdings keine Auffrischung des Geheimnisses vor, so dass über einen längeren Zeitraum der geheime Schlüssel des PKG kompromittiert werden könnte.

Hierzu kann ein Protokoll wie DHCP[7] zum Einsatz kommen. Auch in Ad hoc Netzen gibt es Vorschläge, IP Adressen dynamisch zuzuweisen [PMW+01, WMP+02]. Auch gibt es den Fall, dass ein Knoten von vornherein mehrere Adressen pro Interface definiert, was spätestens mit der Einführung von IPv6 [DH98] zur Regel wird. Schließlich sind Adressen nicht wirklich sicher, niemand hindert einen Knoten daran, die Adresse eines anderen Knotens anzunehmen oder unter dieser Pakete zu verschicken.

Somit wird klar, dass eine Adresse nicht wirklich geeignet ist, einen Knoten in einem Netzwerk über längere Zeit eindeutig zu identifizieren. Das Konzept der *Identität* muss also anders definiert werden. Die Identität hat im vorliegenden Kontext zwei Aufgaben:

1. Sie ist die Voraussetzung für eine sinnvolle Authentifizierung, denn nur bei Kenntnis der Bedeutung der validierten Identität kann ein Knoten entscheiden, was die Authentifizierung tatsächlich aussagt. Ist der identifizierte Kommunikationspartner ein bestimmter Rechner, eine bestimmte Person oder nur irgendetwas, was einen geheimen Schlüssel kennt?

2. Die Identität dient als unveränderliches Erkennungsmerkmal für einen Knoten, unabhängig von eventuell wechselnden Adressen. Ein solches unveränderliches Erkennungsmerkmal ist von zentraler Bedeutung, sowohl für das sichere Routing-Protokoll SDSR, als auch für das mobile Intrusion Detection System MobIDS.

Es erstaunt, dass dem Konzept der Identität in der Literatur über Authentisierungsprotokolle bisher kaum eine Bedeutung beigemessen wurde. Vielmehr wird meist stillschweigend vorausgesetzt, dass jeder Knoten im Netz über eine dauerhafte und unveränderliche Identität verfügt, ohne genauer auf deren Bedeutung und Eigenschaften einzugehen. Khalili et al. schreiben beispielsweise:

> [...] all principals in the network – including those joining at later times – can now use their *identity* as their public key. [...] Note that we do not specify the nature of the identity to be used [...] [KKA03, Abschnitt 2]

Auch die Arbeiten, die eine Zertifizierungsstelle (CA) zur sicheren Authentisierung von Knoten einsetzen wollen, bleiben sehr wage, wenn es um die Bedeutung dieses Zertifikats geht. Ein Zertifikat bindet einen öffentlichen Schlüssel an weitere Informationen, z.B. die Identität seines Besitzers. Wie diese Identität beschrieben ist und was sie in einem Ad hoc Netz aussagt, bleibt offen. So schreiben Zhou und Haas:

> The *CA* [...] signs certificates binding public keys to nodes. [ZH99, Abschnitt 3]

Ähnliches findet sich in der Arbeit von Luo et al.:

> We assume each networking node obtains its initial certificate through some out-of-band mechanisms and policies that are predefined before the ad hoc network is formed. [KZL+01, Abschnitt 5]

Etwas konkreter werden Hubaux et al.:

> We assume that if a user u believes that a given public key belongs to a given user v, then u issues a public-key certificate to v. [HBČ01, Abschnitt 5]

Hier wird der öffentliche Schlüssel eindeutig einem menschlichen Besitzer zugeordnet.

[7] Dynamic Host Configuration Protokoll [Dro97]

Auf die Thematik von wechselnden Identitäten gehen die Arbeiten auch nicht ein. Deutlich wird dies z.B. bei der Arbeit von Bobba et al. über kryptobasierte Identitäten. Hier kann ein Knoten sich seine Identitäten beliebig selbst erzeugen, indem er einfach ein neues Schlüsselpaar generiert:

> We rely on [...] [uncertified] public-secret key pairs generated by the nodes themselves [...]. [BEGA02, Abschnitt I., Einfügung aus Abschnitt II.B.]

Bei den anderen Arbeiten ist es relativ schwer zu beurteilen, inwieweit ein Knoten seine Identität wechseln kann, da diese – wie erwähnt – erst gar nicht auf die Identität als solche eingehen. In einem Ansatz mit zentraler CA oder verteilter CA hängt dies davon ab, wie und unter welchen Umständen diese ein Zertifikat erteilt und welche Prüfungen zuvor vorgenommen werden. Beim Web-of-Trust Ansatz von Hubaux [HBČ01] muss ein Knoten andere Knoten dazu bringen, ihm einen Schlüssel mit einer neuen Identität zu signieren. In der Praxis dürfte dies wegen der eher Security-unerfahrenen Benutzer in einem MANET leicht gelingen.

Um derartige Schwächen in dieser Arbeit zu vermeiden, sollen daher zunächst folgende Fragen analysiert werden:

1. Was ist eine Identität?
2. Wer wird durch eine Identität identifiziert?
3. Welches Merkmal dient der Identifizierung?
4. Wie werden Identitätsänderungen verhindert?
5. Wie hängen Identität und Adresse zusammen?

8.2.2. Was ist eine Identität?

Ein Knoten in einem Ad hoc Netz muss aus vielerlei Gründen eindeutig zu identifizieren sein. Bereits die Routingprotokolle müssen eindeutige Routing-IDs verwenden, da sonst keine konsistenten Topologien gebildet werden können. Auch dienen die Routing-IDs oft der Erkennung von Schleifen uvm. Hierbei kommen in der Regel *Adressen* zum Einsatz, welche den Knoten entweder dauerhaft oder dynamisch zugewiesen werden. Als Alternative zur manuellen Konfiguration gibt es automatische Konfigurationsprotokolle wie DHCP [Dro97] oder Vorschläge zur automatischen Adresskonfiguration in MANETs [WMP+02][PMW+01].

Aus Sicht des Routingsystems besteht die Anforderung, dass eine Adresse im Netz eindeutig ist, aus Sicht der Netzwerksicherheit genügt diese Adresse aber nicht zur Identifizierung eines Knotens. Ein Knoten kann im Laufe seiner Lebensdauer mitunter verschiedene Adressen verwenden und unter Umständen sogar bewusst Adressen von anderen Knoten annehmen. Eine Adresse liefert also keineswegs eine eindeutige Identifizierung.

Hat ein Knoten aber die Möglichkeit, regelmäßig unter neuen und nicht-korrelierbaren Identitäten im Netz aufzutreten, dann laufen jegliche Schutzmaßnahmen ins Lehre, die auf einen Ausschluss dieses Knotens aus dem Netz abzielen. Ist der Knoten unter einer Identität auffällig geworden und droht ihm ein Ausschluss, so setzt er seine Angriffe einfach unter einer neuen Identität fort. Auch gibt es Schutzsysteme, welche

nur einen bestimmten Anteil böswilliger Knoten am Gesamtnetz tolerieren. Dies wird typischerweise so ausgedrückt, dass maximal k aus n Knoten für einen Angriff kooperieren dürfen. Kann sich ein Knoten selbst beliebige Identitäten generieren, so hindert ihn nichts daran, zusätzlich zu seiner Hauptidentität weitere k virtuelle Identitäten zu erzeugen und diese als eine Art „virtuelles Netz“ in das MANET einzubinden. Mit diesen insgesamt $k+1$ Identitäten ist der Knoten dann in der Lage, einen erfolgreichen Angriff gegen das MANET durchzuführen.

Insbesondere das später vorgestellte MobIDS System ist darauf angewiesen, dass Knoten nicht beliebig ihre Identität wechseln können, da sonst verschiedene der Sensoren unwirksam werden und auch ein Ausschluss aus dem Netz nicht den gewünschten Effekt erzielt. Dies führt uns direkt zu der Frage, was eine *Identität* eigentlich genau ist? Je nach Fachrichtung gebrauchen beispielsweise Psychologie, Philosophie oder auch die Mathematik diesen Begriff sehr unterschiedlich.

Die Soziologie definiert *Identität* z.B. als „das dauernde innere Sich-Selbst-Gleichsein, die Kontinuität des Selbsterlebens eines Individuums [...]“ [FKL+78], ein Lexikon als „das Sich-gleich-Bleiben im Wechsel“ [LIM72]. Für diese Arbeit soll eine etwas technischere Definition verwendet werden.

Definition 8.1 (Identität) *Die Identität eines Objekts ist eine eindeutige und unabänderlich mit diesem Objekt verknüpfte Eigenschaft, welche bei einem Objekt während seiner gesamten Existenz gleich bleibt und auch nicht auf andere Objekte übertragen werden kann.*

Damit erweitern wir die Definition der Identität von Individuen (also Menschen und evtl. auch Tieren) auf beliebige andere Objekte, solange sie eindeutig zu identifizieren sind. Existieren von einer Sache viele identische Versionen, so kann die Identität jeder einzelnen Sache durch Hinzufügen von *Identifikatoren* sicher gestellt werden.

Definition 8.2 (Identifikator) *Ein* Identifikator *ist ein Merkmal (oder eine Gruppe von Merkmalen), welches geeignet ist, ein Objekt zu identifizieren, das heißt, seine Identität zweifelsfrei festzustellen, und welches den Kriterien der Identität (Eindeutigkeit, unveränderliche Verknüpfung, lebenslange Gültigkeit, keine Übertragbarkeit auf andere Objekte) genügt.*

Zur Veranschaulichung sei ein Beispiel herangezogen. Pkws sind heute Massenware. Von einem bestimmten Modell eines bestimmten Herstellers verlassen täglich hunderte Fahrzeuge die Fabrik. Darunter sind, trotz aller Optionen und Farbvarianten, eine Vielzahl von Wagen, die zunächst vollkommen gleich sind. Somit kann man zunächst nicht von einer Identität eines solchen Wagens sprechen. Bei einem so teuren Produkt ist aber eine Identität aus einer Vielzahl von Gründen wünschenswert. Der Hersteller will im Garantiefall genau feststellen können, um welchen Wagen es sich handelt und im Falle eines Diebstahls will die Polizei den Wagen möglichst genau verfolgen können. Deshalb wird bei der Herstellung jedes Wagens eine Fahrgestellnummer fest in das Fahrgestell eingeprägt. Sie wird nur einmal vergeben und erfüllt somit (zusammen mit Hersteller und Modell) die notwendige Eindeutigkeit einer Identität. Auch ist sie unveränderlich und auf seine gesamte Lebensdauer mit dem Wagen (bzw. dessen Fahrgestell) verknüpft. Kleinere Reparaturen am Fahrzeug (z.B. ein Wechsel des Kotflügels) führen zu keiner Änderung seiner Identität. Ein Wechsel der Fahrgestellnummer gelingt höchstens professionellen Autodieben und hinterlässt in der Regel Spuren. Allerdings kann die Identität durch Abschleifen relativ einfach gelöscht werden. Die fehlende Über-

tragbarkeit ist (mit gewissen Einschränkungen) gegeben: Eine existierende Nummer lässt sich nur beim Produktionsprozess auf einen anderen Wagen übertragen, da sonst Spuren der früheren Nummer zurückbleiben würden.

8.2.3. Wer wird identifiziert?

Da nun klar ist, dass in einem Ad hoc Netzwerk in jedem Fall Identitäten benötigt werden und welche Bedingungen diese im Idealfall zu erfüllen haben, stellt sich jetzt noch die Frage, wer oder was mit einer Identität versehen werden soll. Zur Auswahl stehen hier im Wesentlichen drei Möglichkeiten:

1. Der Benutzer
2. Das mobile Gerät
3. Ein Netzwerkanschluss

Vorteil einer Identifizierung auf Benutzerebene ist, dass Menschen bereits aufgrund ihrer Individualität die Identitätsbedingungen recht gut erfüllen. Leider sind diese individuellen Merkmale einem Rechner nur schlecht zugänglich, wie die Probleme beim Einsatz biometrischer Erkennungsverfahren zeigen. Somit müsste ein dem Rechner besser zugänglicher Identifikator eingeführt werden, was beispielsweise in Form eines asymmetrischen Schlüsselpaares geschehen kann. Dieses müsste dann aber wieder eindeutig, unveränderlich und nicht übertragbar mit dem Benutzer verknüpft werden. Weiterhin ist davon auszugehen, dass ein Benutzer manchmal mehrere Geräte in einem MANET einsetzen will. Diese benutzen dann alle den gleichen Identifikator und sind nicht mehr auseinanderzuhalten. Müsste MobIDS einen Knoten im MANET ausschließen, würden somit automatisch alle anderen Geräte eines Benutzers gesperrt. Dies kann als abschreckende Maßnahme gewünscht sein, geht aber im Allgemeinen zu weit. Dann könnte nämlich ein Angreifer ein gestohlenes oder verloren gegangenes Gerät dazu einsetzen, sämtliche anderen Geräte eines Benutzers sperren zu lassen. Ein weiterer Nachteil von benutzerbasierten Identitäten ist, dass manchmal mehrere Menschen immer wieder verschiedene Geräte benutzen und ein bestimmtes Gerät unter Umständen von mehreren Menschen nacheinander genutzt wird. Somit müsste ein Benutzer in der Lage sein, seinen Identifikator z.b. in einer Chipkarte oder einem USB-Token mit sich herumzutragen.

Die Zuordnung einer benutzerunabhängigen Identität pro Gerät erscheint daher naheliegender. Somit kann MobIDS zielgenau einzelne FEB Knoten erkennen und aus dem Netz ausschließen. Auch arbeiten die MANET Routingprotokolle normalerweise auf der Ebene von Knoten, so dass die Integration der Sicherheitsmaßnahmen in den Routingprozess erleichtert wird. Ein böswilliger Benutzer kann dann natürlich sein Angriffspotential durch den Erwerb zusätzlicher Geräte erhöhen. Die Effektivität eines solchen Angriffs scheitert aber in der Regel an den Hardwarekosten.

Einen Nachteil haben gerätebezogene IDs: spätestens auf der Applikationsebene will man in der Regel weniger bestimmte Geräte, sondern vielmehr deren Benutzer ansprechen. Von daher mag es sich, je nach den genutzten Anwendungen, anbieten, zusätzlich auch benutzerbezogene Identitäten einzusetzen.

Die Alternative einer Identität pro Netzwerkkarte bietet im Vergleich zur Zuordnung pro Gerät keine nennenswerte Vorteile. Im Gegenteil muss sich das Gerät dann zumin-

dest für das Routing-Protokoll aus den vorhandenen Identitäten eine Haupt-Identität auswählen, da alle Routingprotokolle von einer einzelnen Knoten-ID ausgehen. Unter Sicherheitsgesichtspunkten bieten mehrere mögliche Identitäten pro Gerät auch mehr Möglichkeiten zum Missbrauch.

Aus den aufgeführten Gründen wird im Folgenden also davon ausgegangen, dass eine Identität im Ad hoc Netzwerk immer für ein Gerät bzw. einen Knoten gilt. Die Benutzer sind aus Netzwerksicht irrelevant, genauso wie die Zahl der Netzwerkschnittstellen pro Gerät. Dabei kann es, wie bereits angemerkt, sinnvoll sein, auf der Anwendungsschicht eine zusätzliche Benutzeridentität einzuführen und zu verwenden. Dies ist für diese Arbeit aber nicht weiter relevant.

8.2.4. Welches Merkmal dient der Identifizierung?

Als nächstes gilt es, einen geeigneten Identifikator für das Gerät zu finden. In Frage kommen hierbei:

Geräteabhängige Identifikatoren: Bei einem geräteabhändigen Identifikator wird ein fest mit dem Gerät verbundenes Merkmal als Identifikator genutzt. Wireless LAN oder Bluetooth Adapter besitzen beispielsweise eine fest eingebaute MAC Adresse[8]. Auch gab es in der Vergangenheit Anstrengungen, eindeutige Seriennummern in Prozessoren und sonstiger Hardware zu verankern [Int99, Bög01]. Allerdings ist man nach technischen Schwierigkeiten und schlechter Akzeptanz auf Kundenseite von diesem Konzept wieder abgekommen [Sti00]. Jüngere Konzepte [TCP] zielen jedoch wieder verstärkt auf eine Identifizierbarkeit der Hardware ab.

Generische Identifikatoren: Alternativ lassen sich auch generische Identifikatoren benutzen, die über keinerlei innere Struktur oder Bedeutung verfügen. Wichtig ist einzig, dass die Eindeutigkeit garantiert ist. Dies könnte durch Verwendung eines ausreichend langen Zufallsbitstrings geschehen, bei dem die Wahrscheinlichkeit einer zufälligen Kollision extrem gering ist. Alternativ könnte auch ein aus der aktuellen Uhrzeit, der geographischen Position und weiteren Daten erzeugter *universally unique identifier (UUID)* zur Anwendung kommen.

Public-Key Identifikatoren: Der öffentliche Teil z.B. eines RSA-Schlüsselpaares kann ebenfalls als generischer Identifikator genutzt werden.

Stand heute erfüllen geräteabhängige Identifikatoren noch nicht alle Anforderungen, wie wir sie in Abschnitt 8.2.2 festgelegt haben: die *Eindeutigkeit* kann zwar vom Hersteller garantiert werden und auch die *lebenslange Gültigkeit* ist in der Regel gegeben. Die weiteren Anforderungen – *keine Übertragbarkeit* auf andere Objekte und *unveränderliche Verknüpfung* mit dem System – werden aber erst die zukünftigen Sicherheitssysteme ermöglichen [TCP].

Bis diese eingeführt und in Geräten verfügbar sind, muss man daher mit generischen Identifikatoren arbeiten. Auch bei diesen kann man durch geeignete Verteilung und Speicherung die *Eindeutigkeit* und *lebenslange Gültigkeit* sicherstellen. Allerdings lässt sich der Identifikator in einem Gerät typischerweise verändern und auch in andere Geräte kopieren.

[8] MAC = Medium Access Control, Teilschicht der Sicherungsschicht im OSI Modell [ISO84]

Etwas besser wird die Situation, wenn man statt generischer Identifikatoren ein Public-Key Schlüsselpaar verwendet. Der öffentliche Teil des Schlüssel erfüllt dabei die Kriterien an SUCV[9] Identitäten [MC02], d.h. er ist statistisch einmalig und außerdem kryptographisch verifizierbar.

Wenn man von sehr unwahrscheinlichen zufälligen Kollisionen absieht, ist ein solcher Schlüssel also *eindeutig* und kann auch *lebenslang* verwendet werden. Wenn der Knoten den geheimen Teil des Schlüsselpaares nicht weiter gibt, ist dieser Identifikator auch *nicht übertragbar*; nur der rechtmäßige Besitzer des Schlüsselpaares kann seine Identität z.B. mit einer Signatur beweisen. Ohne weitere Vorkehrungen kann ein Knoten jedoch beliebig viele dieser Identifikatoren generieren und somit unter neuen Identitäten auftreten. Es stellt sich also noch die Frage, wie man derartige Identitätsänderungen bei Verwendung von Public-Key Identifikatoren verhindert.

8.2.5. Wie werden Identitätsänderungen verhindert?

Behauptung: *Ist die Erzeugung eines Identifikators alleine Sache des mobilen Knotens, so kann eine Identitätsänderung nicht verhindert werden.*

Beweis: Sei $\mathcal{A}(X)$ das Verfahren, welches der Knoten mit Namen X zur Erzeugung seines Identifikators ID_X anwendet. Neben X gehen in $\mathcal{A}$ keine weiteren Informationen von außen ein. Der Algorithmus von $\mathcal{A}$ muss X bekannt sein, um ID_X erzeugen zu können. Folglich kann X den Algorithmus auch mit einem anderen Namen Y ausführen und erhält somit $\mathcal{A}(Y) = \mathrm{ID}_Y$. Wenn ID_X gültig ist, dann ist auch ID_Y gültig.

Somit wird im Umkehrschluss deutlich, dass ein Knoten seine Identität nicht komplett selbst generieren darf. Vielmehr muss eine weitere Partei an diesem Prozess beteiligt werden, die pro Knoten nur genau eine Identität vergibt. Die Netzwerkknoten müssen sich dabei darauf verlassen können, dass diese dritte Partei ihre Aufgabe zuverlässig und gewissenhaft durchführt und nicht doch einem Knoten mehrere Identitäten zukommen lässt bzw. einzelne Knoten des MANETs bevorzugt. Aus diesem Grund bezeichnen wir diese Instanz als *Trusted Third Party (TTP)*. Es lässt sich also feststellen:

Ohne Beteiligung einer Trusted Third Party können keine verlässlichen Identifikatoren generiert werden.

Als TTP kommen entweder die Knoten des MANETs, eine Teilmenge davon oder eine externe Instanz in Frage. Dabei ist eine externe Instanz zu bevorzugen, da diese nicht am MANET teilnimmt und deshalb auch keine eigenen Interessen im Netz verfolgt.

Da im MANET keine ständige Verfügbarkeit der TTP gegeben ist, muss der Identifikator irgendwie im Knoten selbst abgelegt und abgesichert sein. Dies kann entweder auf Basis sicherer Hardware oder mittels Zertifikaten erfolgen.

Sichere Hardware Eine Möglichkeit besteht darin, dass der Identifikator durch die TTP unveränderlich in Hardware abgelegt wird. Dazu kommt sogenannte *tamper-proof hardware* zum Einsatz. Dies kann beispielsweise in Form von Chipkarten oder speziellen Modulen im Gerät geschehen, die gegen Manipulationen von au-

[9] statistically unique and cryptographically verifiable

ßen geschützt sind. Gängige Beispiele sind die SIM-Karte von Mobiltelefonen, verschiedene Pay-TV Karten, die Geldkarte oder das TPM Modul der TCPA [TCP].

Die Kryptanalyse hat mittlerweile jedoch eine Vielzahl von Verfahren entwickelt, mit denen sich z.B. durch Side-Channel-Attacks über den Stromverbrauch oder das Zeitverhalten auch ein ausgefeilter Hardwareschutz überlisten lässt [AK96, AK97, Koc96, KJJ99, Sch00]. Hinreichend sichere Hardware ist zudem sehr teuer, wohingegen in MANETs eher preiswerte, portable Hardware zum Einsatz kommen sollte.

Zertifikate Ein anderer Weg besteht im Einsatz von Zertifikaten. In diesem Fall ist die TTP eine Zertifizierungsstelle (CA). Sie bescheinigt einem Identifikator dessen Gültigkeit. Nur die CA kann mit ihrem geheimen Schlüssel ein Zertifikat erstellen. Gleichzeitig kann jeder MANET Teilnehmer das Zertifikat mit dem öffentlichen Schlüssel der CA überprüfen. Identifikatoren ohne Zertifikat werden von den Knoten des MANET abgelehnt. Dies setzt allerdings voraus, dass der öffentliche Schlüssel der CA auf sicherem Weg jedem Knoten zur Verfügung steht.

Die Aufgabe der CA ist in unserem Fall etwas anders als bei herkömmlichen CAs. Diese verknüpfen die Identität eines Schlüsselbesitzers mit dem öffentlichen Schlüssel. Dazu muss sich der Besitzer bei der CA ausweisen (d.h. einen Identifikator vorweisen) und seinen öffentlichen Schlüssel vorlegen. Anschließend werden der Identifikator „Name“ und der öffentliche Schlüssel durch das Zertifikat miteinander verknüpft.

In unserem Fall ist der öffentliche Schlüssel bereits der (generische) Identifikator. Es ist für unseren Einsatzzweck nicht notwendig, diesen mit einem weiteren Identifikator zu verknüpfen, vielmehr will man lediglich die Gültigkeit eines Identifikators bestätigen. Bindet man zusätzliche Informationen wie Gerätetyp oder Name des Besitzers in ein Zertifikat ein, könnte das mit dem Ziel des Datenschutzes kollidieren (vgl. Kapitel 9). Es genügt also, wenn die CA nur einen öffentlichen Schlüssel signiert.

Vor Ausstellung des Zertifikats muss sich die CA zunächst davon überzeugen, dass das entsprechende Gerät noch kein Zertifikat erhalten hat, da sich sonst ein Knoten ja wieder mehrere öffentliche Schlüssel (und damit Identitäten) signieren lassen kann. Dies setzt aber wieder ein eindeutiges Merkmal des Gerätes voraus, also einen Identifikator. Wir drehen uns im Kreis. Allerdings kann man im Falle der CA davon ausgehen, dass diese tatsächlich die physikalische Einmaligkeit des Gerätes z.B. direkt beim Herstellungsprozess oder über eine eingeprägte Seriennummer prüfen kann.

Letztendlich fallen bei allen zertifikatsbasierten Ansätzen zwei Dinge auf: kein technisches Verfahren kann Vertrauen *erzeugen*. Man kann lediglich bestehendes Vertrauen *übertragen*. Und zweitens: Fakten aus der realen Welt (z.B. Name des Besitzers, physikalische Einmaligkeit des Gerätes etc.) lassen sich mittels Zertifikaten nur durch entsprechende Prüfungen in der realen Welt auf einen virtuellen Identifikator im Rechner übertragen. Deshalb muss eine CA in jedem Fall physikalisch aktiv werden und beispielsweise die physikalische Seriennummer eines Gerätes oder den Ausweis einer Person überprüfen, bevor sie ein Zertifikat ausstellen kann. In MANETs mit vielen unbekannten Knoten stellt dies ein beson-

deres Problem dar, vor allem bei Web-of-Trust basierten Ansätzen. Aber auch herkömmliche CAs können von diesen Problemen betroffen sein [Bag01, ES00].

8.2.6. Wie hängen Identität und Adresse zusammen?

Wenn nun die Identität eines Knotens durch dessen Identifikator (z.B. den öffentlichen Schlüssel) festgelegt ist, so benötigt er trotzdem zu Routingzwecken eine Netzwerkadresse. Bei MANET Routingprotokollen müssen Adresse und Identifikator korrelierbar sein, um bestimmte Manipulationen während des Routingprozesses zu verhindern. Sonst könnte ein FEB Knoten beispielsweise die Adresse eines fremden Knotens in einen Route Request aufnehmen und dies mit seinem Schlüssel signieren.

Es gibt drei Möglichkeiten, diese Verbindung herzustellen. Entweder übernimmt man den Identifikator direkt als Adresse oder man leitet aus dem Identifikator die Adresse mittels einer (Hash-)Funktion ab. Letzteres entspricht den kryptobasierten Adressen aus Abschnitt 8.1.4. Schließlich könnte man auch den Identifikator und die Adresse wieder über ein Zertifikat miteinander koppeln. Dieses Zertifikat kann entweder eine Zertifizierungsstelle oder der Knoten selbst mit seinem eigenen Schlüsselpaar generieren.

Aufbauend auf den Vorüberlegungen und der Analyse der anderen Arbeiten soll nun ein praxistaugliches System zur Identifizierung von Knoten in einem Ad hoc Netzwerk erarbeitet werden, welches später dem SDSR Protokoll und dem MobIDS als Basis dient. Dieses Identifikationsverfahren nennen wir *MANET-IDs*.

8.3. MANET-IDs

8.3.1. Ziele und Voraussetzungen

Folgendes Ziel soll mit *MANET-IDs* erreicht werden:

Gewährleistung von Identitäten: Jeder Knoten soll eine feste und unveränderliche Identität im Sinne der Definition 8.1 erhalten. Diese kann somit als Wiedererkennungsmerkmal, aber auch zum Ausschluss eines Knotens aus einem MANET genutzt werden. Kein Knoten kann mehrere Identitäten annehmen.

Um dieses Ziel zu erreichen, sind mehrere der oben aufgeworfenen Fragen zu klären. Im Rahmen der freien Designparameter haben wir uns als Ergebnis der oben angestellten Überlegungen für folgende Alternativen entschieden:

Wer wird identifiziert? Eine *MANET-ID* identifiziert ein Gerät.

Welches Merkmal dient der Identifizierung? Als Identifikator kommt der öffentliche Teil eines Public-Key Schlüsselpaares zum Einsatz.

Wie werden Identitätsänderungen verhindert? Eine MANET-ID ist nur dann gültig, wenn sie von einer Trusted Third Party bzw. CA signiert wurde.

Wie hängen Identität und Adresse zusammen? Entsprechend der Idee der kryptobasierten Adressen aus [BEGA02] (vgl. Abschnitt 8.1.4) wird aus dem Identifikator eine Adresse generiert. Somit lässt sich stets prüfen, ob eine Adresse zu einer gegebenen Identität gehört und umgekehrt.

Unser System trifft dabei einige Annahmen, welche zur Funktionsfähigkeit erfüllt sein müssen:

Gelegentliche Internetverbindung: Das ursprüngliche Zertifikat wird von der CA bei Herstellung des Gerätes erteilt. Für das Management von Pseudonymen und die Verlängerung- bzw. den Rückruf von Zertifikaten muss ein Knoten jedoch in regelmäßigen Intervallen (in der Größenordnung von Tagen) einen Kontakt zur CA aufbauen. Wenn ein Internet-Gateway vorhanden ist, kann dies über das MANET selbst erfolgen, sonst muss die Verbindung über einen alternativen Weg realisiert werden. In Frage kommen dann ein gelegentlicher Anschluss an ein LAN oder der Kontakt über ein Mobilfunknetz z.B. via GPRS.

Lose Zeitsynchronisation: Um die Gültigkeit von Zertifikaten zu prüfen, müssen die Geräte über lose synchronisierte Uhren verfügen. Hierbei ist eine Genauigkeit von 15 bis 30 Minuten ausreichend. Da die Knoten in definierten Abständen Kontakt mit der CA aufnehmen, lässt sich diese Genauigkeit problemlos erreichen.

Verfügbarkeit einer CA: Aufgrund der in den Abschnitten 8.1.2 und 8.2.1 geschilderten inhärenten Nachteile von ins MANET integrierten und verteilten CA-Lösungen wird hierauf verzichtet. Die CA ist zentral realisiert und wird bereits bei Geräteherstellung tätig. Um Missbrauch zu verhindern, sind Zertifikate nur eingeschränkt gültig und müssen dann verlängert werden. Die CA ist für die Zertifikatsverlängerung verantwortlich. Hierzu muss die CA den Knoten zumindest sporadisch zur Verfügung stehen. Der öffentliche Schlüssel der CA ist allen MANET Knoten bekannt. Die CA ist hinreichend gegen Angriffe geschützt.

Public-Key Operationen: Die Knoten müssen in der Lage sein, zumindest vereinzelte Public-Key Operationen effizient durchzuführen. Nichtsdestotrotz werden diese weitestgehend vermieden, um die Effizienz des Systems zu gewährleisten.

Intrusion Detection System: Die CA gewährleistet nur den langfristigen Ausschluss von FEB Knoten aus dem Netz, indem sie die Verlängerung eines Zertifikates unter gravierenden Umständen verweigert.

8.3.2. Funktionsweise der MANET-IDs

Für jeden Knoten X wird bei der Geräteherstellung ein Public-Key Schlüsselpaar (PK_X, SK_X) generiert. Wir gehen im folgenden davon aus, dass als Algorithmus RSA zum Einsatz kommt, andere Verfahren lassen sich aber genauso verwenden. Der öffentliche Schlüssel PK_X dient dann als Identifikator für X und wird *MANET-ID* genannt. Dieser kann in beliebigen Ad hoc Netzen eingesetzt werden, sofern das SAM Sicherheitssystem unterstützt wird. Die Identität ist also nicht auf ein einzelnes MANET beschränkt, sondern ist überall identisch. Dies hat Vorteile, weil es auch zwischen MANETs nicht zu Doppeldeutigkeiten kommen kann. Auch kann so ein FEB Knoten nicht nur aus einem einzelnen, sondern aus allen MANETs ausgeschlossen werden, die SAM unterstützen.

Eine *MANET-ID* wird erst durch ein Zertifikat der *MANET-CA* gültig. Dieses Zertifikat wird bei der Herstellung des Gerätes erzeugt und im Gerät zusammen mit dem Schlüsselpaar und dem öffentlichen Schlüssel der MANET-CA hinterlegt. Es hat die Form:

$$cert = E_{SK_{CA}}(PK_X, cert\_serial\_number)$$

Wie man sieht, enthält das Zertifikat keinerlei Informationen über das Gerät selbst. Es dient lediglich dazu, einen öffentlichen Schlüssel als gültig zu deklarieren. Dabei wird davon ausgegangen, dass keine zwei öffentlichen Schlüssel identisch sind. Bei hinreichend großer Schlüssellänge ist diese Annahme zulässig, weil die Wahrscheinlichkeit einer zufälligen Kollision astronomisch klein ist[10].

Solange sich MANETs mit dem SAM-System nicht global durchgesetzt haben, wird es natürlich Geräte geben, welche nicht über ein solches Schlüsselpaar plus zugehörigem Zertifikat verfügen. Um diese trotzdem einzubinden, ist bei MANET-IDs die ausnahmsweise spätere Erstellung von Schlüsseln und Zertifikaten vorgesehen. Wie bereits in Abschnitt 8.2.5 ausgeführt, ist hierzu eine physikalische Prüfung des Gerätes unabdingbar. Dazu muss der Anwender sein Gerät bei einer Zertifizierungsstelle[11] vorführen, wo Geräteart, Typen- und Seriennummer in einer Datenbank erfasst werden. Dann wird für dieses Gerät *einmalig* ein Zertifikat erteilt. Dabei sollte die Markierung des Gerätes mit einer Seriennummer fälschungsevident (*tamper evident*) erfolgen, d.h. eine Manipulation sollte dem Prüfer bei der CA auffallen. Hierzu reicht es in aller Regel bereits, die Seriennummer ins Gerät einzugravieren. Der Aufwand ist in jedem Fall deutlich geringer als bei fälschungssicherer (*tamper proof*) Hardware.

8.3.3. Angriffsszenarien gegen MANET-IDs

Hardwaremanipulationen

Ein Angreifer könnte nun versuchen, seine Hardware in irgendeiner Form zu modifizieren, so dass er weitere gültige Zertifikate und somit weitere Identitäten erhält. Das Public-Key Schlüsselpaar und das Zertifikat eines Gerätes werden aber in der Regel bereits bei der Herstellung vergeben, so dass ein Angreifer mit einem der Hersteller kooperieren müsste, um auf diesem Weg eine neue Signatur zu erhalten.

Über den Weg einer nachträglichen Zertifizierung würde man, bei geeigneter Manipulation des Gerätes (z.B. geänderte Seriennummer) und ungenauer Arbeitsweise der Zertifizierungsstelle, vielleicht einige wenige Zertifikate erhalten können. Genauso könnte man durch den Kauf weiterer Geräte zusätzliche Zertifikate erhalten, die man mit etwas Aufwand auf ein einzelnes Gerät übertragen kann. All dies setzt aber eine physikalische Aktivität des Benutzers (Interaktion mit CA oder Händler) sowie u.U. auch finanzielle Aufwendungen (Zertifizierungsgebühr oder Gerätepreis) voraus, so dass der Aufwand zur Beschaffung weiterer Zertifikate relativ hoch ist.

[10] Selbst bei einer konservativen Abschätzung der Wahrscheinlichkeit, dass zwei gleiche Primzahlen ausgewählt werden und wenn jedesmal das gleiche e verwendet wird, ist die Wahrscheinlichkeit für die zufällige Auswahl zweier gleicher öffentlicher Schlüssel sehr viel kleiner als 10^{-100}. Bei geschätzten 10^{80} Atomen im Universum ist der Begriff „astronomisch“ gerechtfertigt.

[11] Dies könnte eine Servicestelle des Herstellers übernehmen.

Betrachtet man demgegenüber den Nutzen, den ein Benutzer aus mehreren Zertifikaten ziehen kann, so hält sich dieser in sehr engen Grenzen. Er ist damit in der Lage, um einige Zeit länger in einem Ad hoc Netzwerk[12] zu verweilen und dort den Betrieb zu stören, bevor schließlich alle seine Identitäten gesperrt werden. Hierzu ein Zitat aus den FAQ der Newsgruppe `de.comp.security.firewall`:

> „Sicher ist ein System genau dann, wenn die Kosten des Angriffs den erzielbaren Nutzen niemals unterschreiten." [D+03, Abschnitt „Was ist eigentlich sicher?"]

Das MANET-ID System kann also als (hinreichend) sicher im Sinne obiger Definition bezeichnet werden. Ein weiterer Angriffsweg bleibt noch zu betrachten. Übernimmt ein Angreifer die Kontrolle über ein Gerät, so könnte er das Schlüsselpaar inklusive geheimem Schlüssel und Zertifikat kopieren und in der Folge einsetzen. Über Würmer ließe sich dieser Angriff u.U. sogar automatisieren und somit eine große Zahl von Zertifikaten besorgen. Um einem solchen Angriff zu begegnen, sollte man den geheimen Schlüssel über ein Passwort (eine sog. *Passphrase*) absichern oder ihn gleich in einer Chipkarte mit Verschlüsselungsfunktion ablegen. Absolut sicher sind die geheimen Schlüssel aber auch dann nicht.

Alternativ könnten sich mehrere Angreifer zusammenschließen und ihre Zertifikate freiwillig austauschen, um sie dann parallel in unterschiedlichen Netzen zu verwenden. Solange eine Identität nicht zweimal im gleichen Ad hoc Netzwerk eingesetzt wird, fällt dies zunächst nicht auf. Man könnte sich sogar eine Art „Identitäten-Tauschbörse" vorstellen, ähnlich den gängigen Filesharing Netzen. Um derartige Aktivitäten unattraktiver zu machen, ist bei MANET-IDs ein *Identitätsrückruf* realisiert. Gibt ein Benutzer also die Identität eines seiner Geräte freiwillig weiter, so riskiert er, dass diese bei Missbrauch gesperrt wird und somit für ihn nicht mehr nutzbar ist.

8.3.4. Identitätsrückruf bei MANET-IDs

Voraussetzungen für einen Identitätsrückruf

Der Identitätsrückruf basiert auf den Resultaten des Intrusion Detection Systems MobIDS. Dieses erkennt Fehlverhalten von Knoten innerhalb eines MANETs. Jeder Knoten in einem MANET führt dabei eine lokale Bewertung über andere Knoten. Diese lokalen Bewertungen werden ausgetauscht und zu einer globalen Bewertung zusammengefasst (vgl. Kapitel 11). Unterschreitet nun die Bewertung eine bestimmte Schranke, so kann ein Knoten eine entsprechende Meldung generieren und diese bei nächster Gelegenheit[13] an eine Meldestelle schicken. Diese wird sinnvollerweise in die CA integriert; hiervon wird im Folgenden ausgegangen.

Um die Meldenachricht zu authentisieren und um Missbrauch vorzubeugen, muss ein Knoten diese Meldung signieren. Somit wird verhindert, dass ein Knoten einen anderen Knoten durch gefälschte Missbrauchsmeldungen diskreditiert und von der MANET Nutzung ausschließt. Will ein Knoten K das Fehlverhalten eines FEB Knotens mit der

[12] welches bei kooperativem Verhalten für jedermann kostenlos und uneingeschränkt zugänglich ist

[13] z.B. bei der nächsten Zertifikatsverlängerung oder wenn ein Internet-Gateway verfügbar ist. Zu diesem Zeitpunkt ist das ursprüngliche MANET u.U. schon nicht mehr existent.

Identität PK_M beim Meldeserver S anzeigen, so geschieht dies wie folgt:

1. $K \rightarrow S: \; E_{SK_K}(K, \; (PK_M, \; date\_of\_incident \; [, \; reason]) \; [, \; (\ldots)])$
2. $S \rightarrow K: \; E_{SK_S}(h((PK_M, \; date\_of\_incident \; [, \; reason]) \; [, \; (\ldots)]))$

In der ersten Nachricht schickt K eine Liste von Meldungen an S. Jedes Element dieser Liste umfasst die Identität des gemeldeten Knotens PK_M, das genaue Datum der Meldung sowie eine optionale Begründung. Letztere wird vom System nicht ausgewertet, kann aber dazu dienen, Vorgänge des Systems für den menschlichen Benutzer transparenter zu machen. Der Server antwortet auf die Nachricht mit einem signierten Hash der Meldungen, um K damit den korrekten Empfang und die korrekte Entschlüsselung der Meldungen anzuzeigen.

Der Knoten K legt dabei dem Server S keinerlei Beweise für die Richtigkeit der Anschuldigungen vor. Das dürfte auch nahezu unmöglich sein, da die Sensoren des MobIDS auf Beobachtungen basiert, die eher Indizien denn harten Beweisen entsprechen. Erst wenn eine genügende Anzahl dieser Indizien zusammen kommt, entscheidet sich ein Knoten für eine Abwertung eines anderen Knotens. Aber selbst wenn ein Knoten stichhaltige Beweise für ein Fehlverhalten hat, dürfte es nahezu unmöglich sein, diese nachvollziehbar und ohne Möglichkeit einer Manipulation auf den Meldeserver S zu übertragen.

Wir gehen daher einen anderen Weg und verzichten von vornherein auf den Beweis von Anschuldigungen. Dafür reagiert der Meldeserver nicht auf einzelne Anschuldigungen, sondern erst, wenn innerhalb eines bestimmten Intervalls T von mindestens k verschiedenen Knoten eine Meldung über eine Identität PK_M eingetroffen sind. Somit können einzelne Knoten oder auch Verschwörungen kleiner Gruppen von Knoten niemals einen anderen Knoten aus dem Netzwerk ausschließen. Da ein solcher Ausschluss generell für jegliche MANETs gilt und erst durch die Erteilung einer neuen Identität durch die CA wieder aufgehoben werden kann, müssen diese Parameter sehr sorgfältig gewählt werden.

Es wird hier davon ausgegangen, dass ein dauerhaftes Fehlverhalten eines Knotens einer großen Zahl von Knoten auffällt und somit eine große Zahl von Meldungen verschiedener Knoten beim Meldeserver eintreffen werden. Wichtig ist, dass das MobIDS System nur eine sehr geringe Rate falsch-positiver Meldungen von Fehlverhalten liefert. Dies wird in Kapitel 12 gezeigt.

Certificate Revocation Lists

Das klassische Mittel zum Rückruf einer Identität sind sogenannte *Certificate Revocation Lists (CRL)* (vgl. Abschnitt 3.6.2). In diesen werden alle ungültigen Zertifikate aufgeführt. Die Knoten müssen diese CRLs in regelmäßigen Abständen vom CA Server holen und vor Benutzung eines Zertifikats prüfen, ob dieses in der jeweils aktuellen CRL enthalten ist.

Damit die CRLs im Laufe der Zeit nicht beliebig lang werden, haben Zertifikate typischerweise eine beschränkte Gültigkeit. Nach Ablauf der Gültigkeit eines Zertifikats kann dieses auch aus den CRLs entfernt werden. Aber auch dann können CRLs sehr schnell sehr groß werden, zu groß für kleine mobile Endgeräte mit begrenztem Speicher.

Die aktuellen CR-Listen von VeriSign[14], einem bekannten kommerziellen CA-Anbieter, sind z.B. deutlich über ein Megabyte groß, obwohl heute nur vergleichsweise wenige Leute über ein solches Zertifikat verfügen. Das kann sich in Zukunft aber ändern. In Deutschland waren Ende 2002 etwa 60 Millionen Mobiltelefone im Einsatz [IZM03]; bei einer ähnlichen Verbreitung MANET-fähiger Geräte und einer angenommenen jährlichen Zertifikatsrückrufrate von 10 % [Mic96], besäße die CRL bis zu 6 Millionen Einträge. Wenn pro zurückgerufenem Zertifikat nur eine 32 Bit lange Seriennummer gespeichert wird, hätte eine solche CRL allein für Deutschland eine Größe von 24 MByte – viel zu viel, um sie sinnvoll auf ressourcenschwächeren Geräten speichern zu können. Die Listen müssen zudem regelmäßig und relativ oft aktualisiert und dann jeweils neu an alle MANET-fähigen Geräte übertragen werden.

Es gibt eine Vielzahl von Rückrufmechanismen [Årn00] mit unterschiedlichen Zielsetzungen, die versuchen, diese Nachteile zu vermeiden. So gibt es den Vorschlag, ganz auf CRLs zu verzichten und bei jeder Zertifikatsprüfung bei der CA den Status des jeweiligen Zertifikats zu erfragen. Mangels ständiger Erreichbarkeit der CA, beschränktem Speicherplatz der Geräte und beschränkter Netzbandbreite sind viele dieser Verfahren für MANETs ungeeignet.

Das Certificate Revocation System von Silvio Micalli

Von Silvio Micalli stammt das *Certificate Revocation System (CRS)* [Mic96], welches ebenfalls ohne die Verteilung von CRLs auskommt und dabei keine ständige Erreichbarkeit des CA Servers voraussetzt. Er baut hierzu zwei weitere Felder Y und N mit jeweils 100 Bit Länge in das Zertifikat ein. Y erzeugt die CA als Hashchain (vgl. Abschnitt 3.2) aus einem initialen Startwert Y_0, auf welchen n mal eine Hashfunktion H angewendet wird. Micalli nimmt eine Zertifikatsgültigkeit von einem Jahr an und setzt $n = 365$, so dass $Y = Y^{365} = H^{365}(Y_0)$.

N berechnet die CA aus einem zufälligen Startwert N_0 als $N = H(N_0)$. Für jeden Tag des folgenden Jahres berechnet und speichert die CA nun pro Zertifikat ein neues Y als $Y = Y^{365-i} = H^{365-i}$, wobei i den i-ten Tag der Gültigkeitsperiode bezeichnet. Wurde das Zertifikat allerdings zurückgerufen und ist somit ungültig, wird stattdessen N_0 gespeichert. Der gespeicherte Wert heißt Verifikator V.

Will ein Knoten jetzt die Gültigkeit eines Zertifikats prüfen, so fragt er bei der CA den Verifikatorwert V ab. V ist nun entweder H^{365-i} oder N_0. Um dies zu prüfen, prüft der Knoten ob $H^i(V) \stackrel{?}{=} Y$ ist[15] – dann ist das Zertifikat gültig. Andernfalls wird geprüft ob $H(V) \stackrel{?}{=} H(N_0) = N$ ist. In diesem Fall hat die CA das Zertifikat zurückgerufen. In jedem anderen Fall wurde das Prüfverfahren manipuliert und der Knoten kann momentan keine Aussage über die Gültigkeit des Zertifikats machen.

Somit findet hierbei eigentlich kein Rückruf eines Zertifikats statt, sondern vielmehr beweist die CA die Gültigkeit eines Zertifikats. Ohne diesen Beweis werden Zertifikate nicht anerkannt.

[14] `http://crl.verisign.com/`

[15] Dies muss gelten, da bei korrektem Ablauf $H^i(V) = H^i(Y^{365-i}) = H^i(H^{365-i}(Y_0)) = H^{365}(Y_0) = Y$ ist.

Dieses Verfahren besitzt einige sehr interessante Eigenschaften. Zum einen genügt ein 100 Bit langer Verifikator V zur Prüfung der Gültigkeit eines Zertifikats. Große CRLs, welcher normalerweise die Identitäten der Schlüsselbesitzer enthalten, entfallen komplett. Zur Prüfung sind weiterhin keine komplexen Public-Key Operationen notwendig, einige einfache Hashberechnungen genügen. Allerdings beschreibt Micalli sein Verfahren als reines Online-Verfahren, so dass der CA Server immer erreichbar sein muss.

8.3.5. MANET-CRS

Wie macht man das Verfahren von Micalli für MANETs nutzbar? In der Literatur findet sich in anderem Kontext die Idee, dass *der Zertifikatsinhaber selbst* die Gültigkeit seines Zertifikats beweisen muss [Koc98]. Diese Idee haben wir mit dem Verfahren von Micalli kombiniert. Hierzu besorgt sich der Knoten in regelmäßigen Abständen – also bspw. täglich – vom CA Server den Verifikator V seines Zertifikats und übermittelt diesen bei Bedarf an seinen Kommunikationspartner. Dieser prüft dann wieder die Gültigkeit des Verifikators identisch zu oben.

Auf die Verwendung von N und N_0 können wir in diesem Fall verzichten. Ist der Verifikator ungültig, so könnte entweder das Zertifikat widerrufen worden sein[16] oder es hat eine Manipulation eines Kommunikationsvorgangs stattgefunden. Im Gegensatz zum Original-CRS können wir bei MANET-CRS nicht zwischen diesen beiden Fällen unterscheiden. Das ist auch nicht notwendig, da für den prüfenden Knoten das Ergebnis in jedem Fall das gleiche ist: er kann die Gültigkeit des Zertifikats nicht prüfen und muss dieses ablehnen.

Die Gültigkeitsdauer T eines Zertifikats wird in n Zeitabschnitte der Dauer t geteilt, so dass $t = T/n$ (vgl. Abbildung 8.2). Bei Erzeugung eines Zertifikats wählt die CA einen zufälligen Startwert Y_0 und berechnet $Y = H^n(Y_0)$. Die Länge von Y und Y_0 entspricht dabei der Ausgabe der Hashfunktion H. Wählt man beispielsweise SHA-1, so ist $|Y| = 160$ *Bit*. Y_0 speichert die CA in ihrer Datenbank und Y sowie das Gültigkeitsende *valid_until* werden mit in das Zertifikat des Knotens K aufgenommen:

$$cert = E_{SK_{CA}}(PK_K,\ cert\_serial\_number,\ Y,\ valid\_until)$$

Sobald ein Knoten Kontakt zum CA-Server aufnehmen kann, holt er sich von diesem den für das aktuelle Zeitintervall i gültigen Verifikator V_i. Das dazu verwendete Protokoll lautet wie folgt:

1. $K \rightarrow CA:\ cert\_serial\_number$
2. $CA \rightarrow K:\ H^{n-i}(Y_0) \quad oder \quad 0,\ E_{SK_{CA}}(cert\_serial\_number\ [,\ reason,\ \ldots])$

Bei Erhalt von (1) prüft die CA zunächst, ob das mit der Seriennummer bezeichnete Zertifikat zurückgerufen wurde oder noch gültig ist. Falls ja, berechnet die CA den für den momentanen Zeitabschnitt i gültigen Verifikator V_i aus dem gespeicherten Startwert Y_0 als $V = H^{n-i}(Y_0)$. V_i kann direkt und ohne weitere Verschlüsselung/Signatur etc. zurückgeschickt werden (2). K prüft dann lediglich, ob der neue Verifikator gültig

[16] Woraufhin die CA dem Knoten keinen gültigen Verifikator mehr ausliefert. Stattdessen wird einfach der Wert 0 zurückgeliefert.

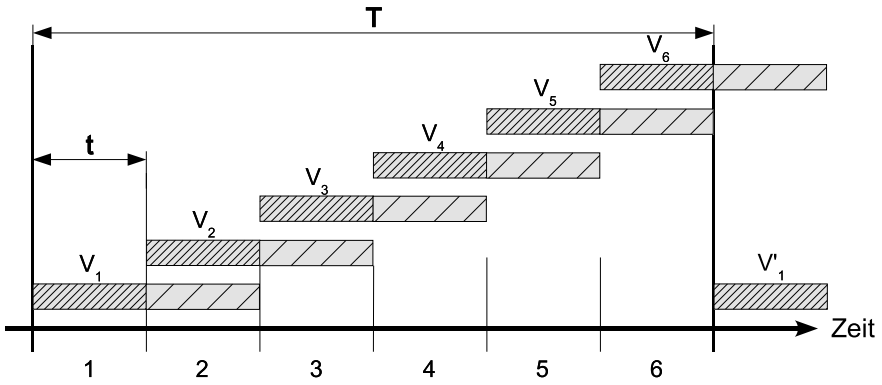

Abbildung 8.2.: Zertifikatsprüfung mit Verifikatoren: Die Zertifikatsgültigkeitsdauer T wurde hier in $n = 6$ Abschnitte unterteilt. Pro Periode gibt die CA einen Verifikator V_i aus, der in der jeweiligen Periode (stark schraffiert) sowie in der Folgeperiode (schwach schraffiert) akzeptiert wird (aus [Spe03]).

ist, also ob $H^i(V_i) \stackrel{?}{=} Y$. Falls nicht, muss er seine Anfrage erneut stellen, da es offensichtlich zu einer Manipulation gekommen ist. Alternativ zu V_i schickt die CA in Nachricht (2) eine Meldung, welche auf die Ungültigkeit des Zertifikats hinweist. In diesem Fall beginnt die Nachricht mit einer 0 anstelle von V_i und liefert dann eine Liste von Gründen für die Ungültigkeit. Diese Daten müssen signiert sein, um eine Fälschung durch Dritte zu verhindern.

Im Rahmen des in Kapitel 10 vorgestellten, sicheren Routing Protokolls SDSR muss nun ein Knoten S zusammen mit seinem öffentlichen Schlüssel und dem zugehörigen Zertifikat immer auch den aktuellen Verifikatorwert V an den Empfänger D übermitteln. Dieser ermittelt aus den festgelegten Werten T, t und n sowie aus der Gültigkeitsdauer des Zertifikats und aus seiner aktuellen Systemzeit das momentan gültige Intervall i. Anschließend prüft er, ob $H^i(V) \stackrel{?}{=} Y$; genau dann ist das Zertifikat noch gültig.

Da wir nur grob synchronisierte Uhren voraussetzen, kann es passieren, dass die Uhr von D etwas *nach*geht und D somit das Intervall i berechnet, obwohl S schon den Verifikator V_{i+1} übermittelt. Somit muss D zusätzlich prüfen, ob $H^{i+1}(V) \stackrel{?}{=} Y$ gilt. Ein anderer Fall kann entstehen, wenn D als gültiges Zeitintervall i ermittelt und S ein V_{i-1} oder noch älter präsentiert. Eine solche Situation entsteht, wenn die Uhr von D *vor*geht oder wenn S trotz neuem Zeitintervall noch keinen aktualisierten Verifikator von der CA geholt hat, da er z.B. noch keine Internetverbindung aufbauen konnte.

Als Lösung bietet sich hier an, dass D auch einen, aus seiner Sicht, veralteten Verifikator V_{i-1} prüft und akzeptiert. Unter Umständen könnte der Knoten sogar noch ältere Verifikatoren $V_{i-2}, V_{i-3}, \ldots$ akzeptieren. Diese Entscheidung bleibt dem jeweiligen Knoten überlassen. Um Probleme mit dem Intrusion Detection System zu vermeiden, gehen wir im folgenden davon aus, dass jeder Knoten genau V_{i-1}, V_i und V_{i+1} prüft.

Somit wird ein Zertifikat unter ungünstigen Umständen spätestens nach 3 Intervallen – also nach 3 Tagen – und frühestens im kommenden Intervall – also am nächsten Tag – gesperrt. Während dieser Zeit kann ein Knoten noch an MANETs teilnehmen. Erkennen die dortigen MobIDS Systeme allerdings ein Fehlverhalten, kommt es unab-

hängig vom Zertifikatsrückruf zu einer lokalen Sperrung des Knotens, so dass sich der Schaden in Grenzen hält.

8.3.6. Verlängerung von Zertifikaten

Da die Hashchain bei MANET-CRS nur eine endliche Länge hat, müssen Zertifikate in gewissen Abständen verlängert werden. Ist beispielsweise in obigem Schema $T = 1$ Jahr, $n = 365$ und somit $t = 24h$, so muss das Zertifikat jährlich erneuert werden. Hierzu nimmt der Knoten K in den letzten Tagen vor Ablauf des Zertifikats Kontakt mit der CA auf und fordert ein neues Zertifikat an. Verpasst ein Knoten den Termin zur Verlängerung, so holt er dies zum nächst möglichen Zeitpunkt nach. Das Protokoll zur Zertifikatsverlängerung lautet wie folgt:

1. $K \rightarrow CA:\ cert\_serial\_number$
2. $CA \rightarrow K:\ E_{PK_K}(cert\_serial\_number,\ N_{CA})$
3. $K \rightarrow CA:\ N_{CA} + 1$
4. $CA \rightarrow K:\ E_{SK_{CA}}(PK_K,\ cert\_serial\_nr + 1,\ Y',\ valid\_until')$

Zunächst fordert K in Schritt (1) eine Verlängerung seines Zertifikats mit der angegebenen Seriennummer an. Die CA überzeugt sich in den nächsten beiden Schritten (2) und (3) von der Authentizität von K, indem dieser eine Nonce N_{CA} entschlüsseln und hochzählen muss. Im letzten Schritt (4) erhält K nun sein neues Zertifikat mit neuer Seriennummer, angepasstem Gültigkeitszeitraum und einem neuen Y'. Selbiges stellt das Ende einer neuen Hashchain dar, welche die CA ausgehend von einem neuen Startwert Y'_0 generiert hat.

Eine erneute physikalische Prüfung des Gerätes wie bei der Erstausstellung des Zertifikats ist in diesem Fall übrigens nicht notwendig, da die CA diese Prüfung ja bereits vollzogen und gespeichert hat und K sich durch die Kenntnis seines geheimen Schlüssels SK_K authentisieren kann.

8.3.7. MANET-IDs und Adressen

Als angenehmer Nebeneffekt lassen sich MANET-IDs auch verwenden, um einfach und effizient Adressen in MANETs zu vergeben. Hierzu bildet man die (IPv6-)Adressen aus einem MANET-spezifischen 64 Bit langen Präfix[17] und einem 64 Bit langen Hostteil. Dieser kann entweder zentral durch die CA zugewiesen und im Zertifikat verankert werden oder die Knoten berechnen ihn analog zu Abschnitt 8.1.4 als CBA durch Anwendung einer Hashfunktion. In letzterem Fall wäre allerdings eine *Duplicate Address Detection* [TN98] notwendig, was in MANETs mit einigem Aufwand verbunden ist.

8.3.8. Authentisierung mit MANET-IDs

Der vorgestellte Mechanismus der MANET-IDs lässt sich jetzt verwenden, um Knoten in MANETs zu authentisieren. Es bietet sich an, diese Authentisierung direkt in

[17] [WMP+02] schlägt hierzu `fec0:0:0:ffff::/64` vor

1. Gehört das Zertifikat $Cert(PK_S)$ zum öffentlichen Schlüssel PK_S?
   (Wenn PK_S Bestandteil von $Cert(PK_S)$ ist, ist dies trivial.)
2. Ist IP_S die legitime IP-Adresse des Absenders?
   (Dies lässt sich entweder als CBA berechnen oder dem Zertifikat entnehmen)
3. Ist die *ID* neu, die Nachricht also nicht nur wiedereingespielt (*replay*)?
4. Wurde PK_S durch das MobIDS System gesperrt?
5. Trägt das Zertifikat eine gültige digitale Signatur der CA?
   (Bei erstmaligem Kontakt mit Zertifikat, Prüfen der Signatur der CA plus Prüfen des Verifikators V_S)
6. Ist die digitale Signatur des Absenders $E_{SK_S}(\ldots)$ korrekt?

Tabelle 8.1.: Tests zur Feststellung der Authentizität eines *Route-Requests*

das Routingprotokoll zu integrieren, da der Route-Request/-Reply den ersten Kontakt zwischen zwei Knoten darstellt und bereits durch Authentisierung abgesichert werden muss. Generiert man im Zuge dieses Ablaufs auch gleich einen geheimen Sitzungsschlüssel zwischen den Knoten, so kann dieser im Folgenden zur Absicherung des regulären Datenverkehrs verwendet werden.

An dieser Stelle werden lediglich die für die Authentisierung wichtigen Abläufe des Route-Discovery Prozesses vorgestellt, eine genaue Beschreibung des gesamten Protokolls erfolgt in Kapitel 10.

Mit dem Route Request verschickt ein Knoten S folgende Informationen:

$$E_{SK_S}(IP_S,\ IP_D,\ ID),\ V_S\ [,\ PK_S,\ Cert(PK_S)]$$

Dabei sind IP_S und IP_D die IP-Adressen von Sender S und Empfänger D, ID ist die Route-Request ID, welche bei jeder neuen Route-Discovery hochgezählt wird. Weiterhin beinhaltet der Route-Request den aktuellen Verifikator von S. Optional können auch der öffentliche Schlüssel von S sowie das zugehörige Zertifikat mit übertragen werden. Alternativ muss sich D diese Informationen auf anderem Wege besorgen (vgl. Kapitel 10).

Der Empfänger des Route-Requests führt nun eine Reihe von Prüfungen durch, welche die Authentizität der übermittelten Daten sicherstellen. Diese werden in einer Reihenfolge durchgeführt, bei der einfache Prüfungen zuerst stattfinden, während komplexere erst später erfolgen. Schlägt eine Prüfung fehl, kann der Prozess abgebrochen werden und der Knoten spart sich so die komplexeren Berechnungen. Tabelle 8.1 zeigt alle notwendigen Überprüfungen.

Konnten alle Tests erfolgreich durchgeführt werden, schickt D einen entsprechenden Route Reply, welcher (unter Anderem) folgende Informationen enthält:

$$E_{SK_D}(IP_S,\ IP_D,\ ID),\ V_D\ [,\ PK_D,\ Cert(PK_D)]$$

S führt nun wieder die Prüfungen nach Tabelle 8.1 durch, nur dass in Schritt 3 geprüft werden muss, ob die *ID* zu einer aktuellen Route Discovery dieses Knotens gehört. Verlaufen alle Tests erfolgreich, so wird der Route-Reply akzeptiert.

Will S zu einem späteren Zeitpunkt[18] eine erneute Route Discovery nach D durchführen, so kommt ein vereinfachtes Authentisierungsverfahren zum Einsatz. Da die Knoten ja bereits einen geheimen Schlüssel K_{SD} ausgetauscht haben, können sie nun diesen zur Authentifizierung ihres Kommunikationspartners verwenden, indem der Schlüssel in eine MAC Funktion eingeht. Der entsprechende RREQ sieht wie folgt aus:

$$\mathrm{IP}_S,\ IP_D,\ ID,\ \underbrace{H_{K_{SD}}(IP_S,\ IP_D,\ ID)}_{MAC},\ V_S$$

Dabei stellt der Verifikator V_S sicher, dass das Zertifikat der Identität noch gültig ist. Die MAC Funktion $H_{K_{SD}}$ garantiert die Authentizität der Daten. Somit ist eine vereinfachte Route Discovery ohne den Einsatz von aufwändiger Public Key Kryptographie möglich.

8.3.9. Speicherung von Schlüsseln und Zertifikaten

Zur Authentifizierung anderer Knoten müssen Knoten in MANETs deren öffentliche Schlüssel und Zertifikate speichern. Auch steigert es die Effizienz, wenn geheime Schlüssel für einen gewissen Zeitraum weiterverwendet werden. Da über das Speichervermögen der Knoten im MANET nichts bekannt ist und dieses in großem Umfang schwanken kann, ist die Speicherung dieser Daten bei Einsatz von MANET-IDs optional.

Besitzt ein Knoten einen entsprechend großen Hintergrundspeicher, so wird er eine umfangreiche Sammlung von öffentlichen Schlüsseln anlegen. Da diese nicht ungültig werden, können sie unbegrenzt gespeichert werden. Auch Zertifikate haben eine vergleichsweise lange Gültigkeitsdauer und entsprechend lohnenswert ist deren Speicherung. Geheime Schlüssel mit anderen Knoten können ebenfalls für eine begrenzte Zeit weiterverwendet werden, um bei einer späteren Route Discovery ein vereinfachtes Authentisierungsverfahren nutzen zu können. Ein Knoten sollte allerdings die Lebensdauer der Sitzungsschlüssel begrenzen und von Zeit zu Zeit eine vollständige Authentisierung durchführen.

Knoten mit weniger Speicherplatz werden die gespeicherten Informationen irgendwann wieder löschen, um Platz für neue Daten zu erhalten. Bei erneuter Kommunikation müssen dann die öffentlichen Schlüssel und Zertifikate wieder übermittelt werden und es wird ein neuer Sitzungsschlüssel vereinbart. Wie sich die Knoten auf die zu übertragenden Schlüsselinformationen einigen, wird in Kapitel 10 erläutert.

8.4. Fazit

In diesem Kapitel wurde ausführlich auf die Fragestellungen rund um Identitäten in Mobilen Ad hoc Netzen eingegangen. Es hat sich gezeigt, dass die bisherigen Arbeiten auf diesem Gebiet diverse Schwächen aufweisen und insbesondere den Begriff der Identität nur unzureichend definieren.

Im Gegensatz dazu ist die Identität eines Knotens bei MANET-IDs präzise beschrieben. MANET-IDs zeichnen sich dadurch aus, dass mit ihnen ein zuverlässiges und effizientes

[18] z.B. nach einem Route Error

Identitätsmanagement in Ad hoc Netzen möglich ist. Public Key Kryptographie wird nur sparsam eingesetzt, was die mobilen Knoten entlastet. Mit MANET-CRS ist ein effizienter Rückrufmechanismus realisiert, welcher auch ohne ständigen Kontakt zur CA funktioniert.

Zur Erlangung dieser Vorteile macht das MANET-ID System einige Annahmen, wie den regelmäßigen Kontakt zur CA im Abstand von einigen Tagen oder die initiale Zertifizierung von Geräten durch den Hersteller. Betrachtet man den Sicherheitsgewinn, erscheinen diese allerdings vertretbar. Insbesondere die initiale Zertifizierung ist – wie gezeigt – unabdingbar, um unveränderliche Identitäten zu gewährleisten.

Ein Nachteil des Systems wurde noch nicht betrachtet. Durch die unveränderlichen Identitäten lassen sich problemlos genaue Bewegungsprofile der Knoten generieren (vgl. Abschnitt 6.2.2). Um dies zu verhindern, werden wir das MANET-ID System im folgenden Kapitel um die Unterstützung von Pseudonymen erweitern.

Die späteren Kapitel zu sicherem Routing und zu Mobile Intrusion Detection setzen eine korrekte Identifizierung von Knoten durch MANET-IDs voraus. Kapitel 12 schließlich beinhaltet eine Analyse des Berechnungs-, Speicher- und Kommunikationsaufwands von MANET-IDs.

9. Pseudonyme

Während der Diskussion verschiedener Angriffsformen wurde in Abschnitt 6.2.2 in Angriffsbaum D gezeigt, dass auch die *Erstellung von Bewegungsprofilen* (engl. *Location Tracking*) ein lohnendes Ziel für einen Angreifer in einem MANET sein kann. In diesem Kapitel wird das Problem nochmals aufgegriffen und die MANET-IDs werden so weiterentwickelt, dass Location Tracking zumindest stark erschwert wird. Doch zunächst soll die Problemstellung nochmals deutlich herausgearbeitet werden.

9.1. Datenschutz und Privatsphäre in mobilen Kommunikationssystemen

In Deutschland und vielen anderen Ländern besteht eine gesetzliche Grundlage, welche z.B. die Betreiber von Kommunikationseinrichtungen zum Schutz der privaten Daten der Teilnehmer verpflichtet. Im Bundesdatenschutzgesetz ist zu lesen:

> „ § 1: Zweck und Anwendungsbereich des Gesetzes
> (1) Zweck dieses Gesetzes ist es, den Einzelnen davor zu schützen, dass er durch den Umgang mit seinen personenbezogenen Daten in seinem Persönlichkeitsrecht beeinträchtigt wird.“ [Bun90]

In Deutschland hat das Bundesverfassungsgericht dem Gesetzgeber im Rahmen des sog. „Volkszählungsurteils“ [Bun83] auferlegt, Gesetze zum Schutz der „informationellen Selbstbestimmung“ zu entwickeln. Dieses Recht leitet sich aus der Menschenwürde (Artikel 1 I GG) und der allgemeinen Handlungsfreiheit (Artikel 2 I GG [Bun45]) her. So sollte jeder deutsche Bürger über die Verwendung von Daten, welche er weiter gibt, in größtmöglichem Umfang selbst bestimmen können [Sta].

Neben diesem gesetzlichen Anspruch hat natürlich jeder Bürger auch ein intuitiv einsichtiges Recht auf den Schutz seiner Privatsphäre. Dieses Recht kollidiert bisweilen mit dem Wunsch von Unternehmen, möglichst viele Informationen ihrer (potentiellen) Kunden zu erhalten, um damit zusätzlichen Umsatz zu generieren. Würde ein Pharmakonzern beispielsweise herausfinden, dass ein Arzt bei seinem Patienten eine bestimmte Krankheit diagnostiziert hat, so könnte das Unternehmen diesem zielgerichtet Werbung zu passenden Medikamenten zuschicken. Dass nicht jeder Patient möchte, dass seine Patientenakte allgemein zugänglich ist, liegt auf der Hand.

Die Betreiber von Kommunikationssystemen sind also aus datenschutzrechtlicher Sicht gehalten, die Kommunikationsdaten ihrer Kunden vor unbefugtem Zugriff zu schützen. Bei drahtlosen Kommunikationssystemen kommt hier ein weiterer Aspekt hinzu. Insbesondere zellenbasierte Systeme erfassen geradezu zwangsweise auch Positionsdaten der Benutzer, da sich relativ leicht feststellen lässt, an welcher Zelle ein Benutzer angemeldet ist.

9.2. Bewegungsprofile

Wertet man solche Positionsbestimmungen zeitabhängig aus, so ergeben sich die sogenannten *Bewegungsprofile*[1]. Damit lassen sich genaue Bewegungsmuster ableiten, die eine Vielzahl von Missbrauchsmöglichkeiten eröffnen. Im Supermarkt könnte der Betreiber beispielsweise auswerten, auf welchen Wegen sich die Kunden durch den Laden bewegen und dort die hochpreisigen Produkte platzieren. Oder es lässt sich auf größerer Skala überwachen, wo sich der Polizei bekannte Demonstranten aufhalten, um gegen mögliche, ungenehmigte Veranstaltungen präventiv vorgehen zu können. Vielleicht findet ja auch jemand per Funkpeilung heraus, dass sein Chef sich in den örtlichen Etablissements herumtreibt und nutzt das zu einer kleinen Gehaltserhöhung. Diese Liste ließe sich endlos fortsetzen und zeigt, dass ein Zugriff auf Bewegungsprofile aus vielerlei Gründen interessant sein kann.

9.2.1. Beispiele für Bewegungsprofile

Im zellbasierten GSM Funknetz werden schon heute sogenannte Location-based Services angeboten [Han03]. Typische Anwendungen sind Suchfunktionen nach Hotels oder Tankstellen und Navigations- oder Hilfssysteme, welche automatisch die aktuelle Position des Nutzers an den Diensteanbieter übermitteln [Opi02, Özk03]. Während heute die Positionsbestimmung noch relativ ungenau ist und sich lediglich aus der Zelle ableitet, in welche der Benutzer eingebucht ist, wollen die Netzbetreiber zukünftig eine deutlich genauere Erkennung anbieten, bei der die Signalstärke an mehreren Basisstationen gemessen und die vermutliche Position trianguliert wird.

Auch bei Wireless LANs lassen sich Positionen durch Ermittlung von Access Point und Signalstärken messen [Kar01]. Da jedoch die Basisstationen längst nicht so flächendeckend verteilt sind, wie beim GSM Netz und da viele verschiedene Betreiber beteiligt sind, ist hier die durchgängige Erstellung von Bewegungsprofilen deutlich schwieriger.

In beiden Fällen braucht man zur Lokalisierung von Knoten die absoluten Koordinaten der Basisstationen. Im einfachsten Fall nimmt man für einen Knoten die Position der Basisstation an und gibt als möglichen Messfehler die Ausdehnung der Zelle an. Alle weiteren Maßnahmen dienen lediglich dazu, diesen Fehler zu verringern und die vermutete Position des Knotens genauer einzugrenzen.

9.3. Positionsbestimmung in MANETs

9.3.1. Informationsquellen

In MANETs ist die Positionsbestimmung deutlich komplizierter. Mangels Basisstationen fehlen zunächst die Referenzpunkte, an denen man sich orientieren kann. Zur Positionsbestimmung kann man folgende Informationsquellen heranziehen:

Routingdaten: Aus den Routingdaten lassen sich bruchstückhaft Informationen zur logischen Netzwerktopologie gewinnen. Je nach Protokolltyp sind unterschied-

[1] Im Englischen wird dieser Vorgang als *location tracking* bezeichnet.

lich viele Daten verfügbar. Bei *link state* Protokollen wie OLSR (siehe Abschnitt 5.3.1) wird die gesamte Topologie an alle Knoten verteilt und steht somit jedem automatisch zur Verfügung. Bei *distance vector* Protokollen wie AODV (siehe Abschnitt 5.3.2) sieht der Knoten jeweils nur einen kleinen Ausschnitt seiner unmittelbaren Umgebung, was die Gewinnung von Informationen über entferntere Knoten merklich erschwert. *Source-Routing* Protokolle wie DSR (siehe Abschnitt 5.3.3) nehmen hier eine Zwischenstellung ein, da jeweils nur Topologieinformationen über die bekannten Routen verteilt werden. Hier kann durch Routensuche gezielt Topologie-Information abgefragt werden. Manche Routingprotokolle verteilen zusätzlich weitere Informationen wie Koordinaten oder Daten zur Verbindungsstärke (siehe Abschnitt 5.3.5), was die Lokalisierung der Knoten beliebig vereinfachen kann. Weiterhin ist zwischen *proaktiven* und *reaktiven* Protokollen zu unterscheiden. Proaktive Protokolle verteilen ihre Daten automatisch, bei reaktiven Protokollen müssen die Knoten diese explizit anfordern.

Overhearing von Routingdaten: Neben den regulären Routingdaten lassen sich unter Umständen auch Daten anderer Knoten im Promiscuous Modus abhören (siehe Abschnitt 11.5.1). Hiermit lässt sich im Wesentlichen wieder die Informationsmenge vergrößern. Auch durch Kooperation mehrerer Knoten gelangt ein Knoten an mehr Informationen.

Overhearing von regulären Daten: Natürlich können auch normale Datenpakete Informationen zur Position eines MANET Knotens beinhalten. Tätigt ein Benutzer beispielsweise eine Online-Bezahlung in einem Geschäft über sein MANET Gerät, so lässt sich daraus auf den Standort schließen. Da jedoch bei SAM alle Kommunikation grundsätzlich verschlüsselt abläuft, soll dieser Fall hier nicht weiter betrachtet werden.

Absolute Koordinaten: Haben einzelne Knoten Zugriff auf ihre absolute Position[2], so lässt sich diese verwenden, um relative Positionen im MANET zu abzuschätzen.

9.3.2. Verfahren zur Positionsbestimmung in MANETs

Bisher gibt es noch relativ wenig Arbeiten zur Positionsbestimmung in MANETs. Andere Verfahren zur Lokalisierung mobiler Knoten wie das „Cricket location-support system“ [PCB00] oder RADAR [BP00, BBP00] gehen immer von einer festen Infrastruktur aus. Lediglich Srđan Čapkun et al. stellen in „GPS-free Positioning in Mobile Ad-Hoc Networks“ [ČHH02] ein Verfahren vor, bei welchem mehrere Knoten in einem MANET kooperieren, um ihre gegenseitigen Positionen in einem relativen Koordinatensystem ohne Hilfe externer Informationsquellen festzulegen.

Hierzu fragt zunächst jeder Knoten seine Nachbarn ab und bestimmt die Entfernung zu ihnen. Als mögliche Verfahren geben die Autoren die Verwendung von „Hello“ Nachrichten und die „Time of Arrival“ [CS98] Methode an. Diese Informationen werden im nahen Umkreis[3] des Knotens verteilt. Nun setzt der Knoten I sich selbst an den Ursprung eines *lokalen Koordinatensystems* und wählt zwei andere Knoten P und Q[4], von

[2] z.B. durch GPS Empfänger, oder weil sie an festen Positionen aufgestellt sind
[3] bis zu zwei Hops Entfernung
[4] wobei I, P und Q nicht auf einer Linie liegen dürfen, also $d_{IQ} + d_{QP} \neq d_{IP}$

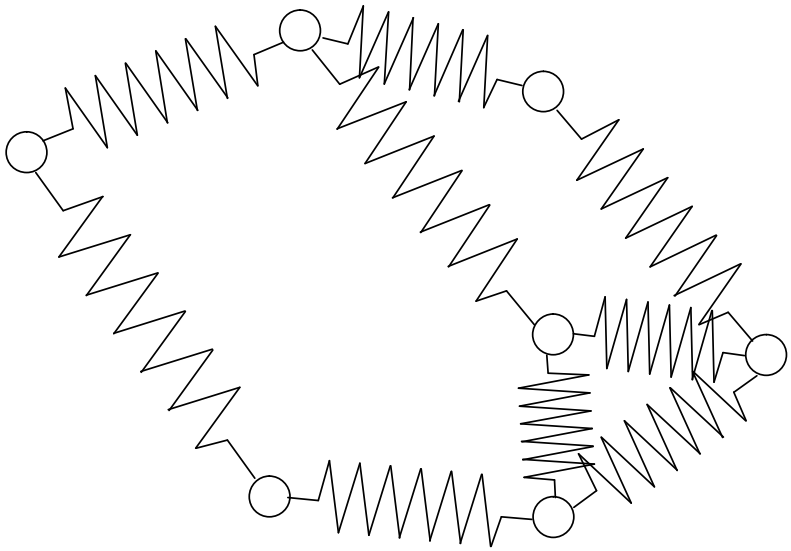

Abbildung 9.1.: Feder Modell zur Bestimmung von Knotenpositionen in MANETs

denen der Abstand d_{PQ} bekannt ist, als Achspunkte. Anschließend werden mittels Triangulation alle weiteren bekannten Nachbarn in dieses Koordinatensystem eingetragen. Schließlich werden die lokalen Koordinatensysteme zwischen den Knoten ausgetauscht und mittels affiner Operationen so transformiert, dass sich ein netzweit eindeutiges Koordinatensystem ergibt.

Für einen Angreifer besteht das Problem dieser Methode darin, dass er auf die Kooperation praktisch aller Knoten im MANET angewiesen ist, um zu verwertbaren Ergebnissen zu gelangen. Dies dürfte im Normalfall nicht gegeben sein.

Deshalb soll an dieser Stelle kurz ein anderes Verfahren skizziert werden, mit welchem man aus den gegebenen, bruchstückhaften Informationen ungefähre Positionen von Knoten ermitteln kann. Dabei geht man von einem physikalischen Modell mit Federn aus, bei dem jede Kante des Netzes durch eine Feder repräsentiert wird (siehe Abbildung 9.1). In einem Knoten sind die zugehörigen Federn fest verbunden. Jeder Knoten A ermittelt für die Verbindung k_{AB} zu seinem Nachbarn B eine wahrscheinlichste Länge l_{AB} sowie einen Fehlerwert e_{AB}. In einem Wireless LAN nach IEEE 802.11b kann eine Verbindung eine Länge zwischen 0 m und 300 m haben. Ohne zusätzliche Informationen setzt ein Knoten also $l_{AB} = 150\,m$ und $e_{AB} = 150\,m$. Sind zusätzliche Verbindungsparameter, wie die empfangene Signalstärke, bekannt, so können diese Werte entsprechend angepasst werden.

Nun werden diese Daten auf das Federmodell übertragen. Dabei wird l_{AB} als Federlänge im entspannten Zustand interpretiert, e_{AB} ergibt die Federhärte. Mittels einer einfachen Simulation [Kar01, Suna] lässt sich nun der Zustand finden, bei dem die gesamte Federspannung im System minimal wird. Dieser spiegelt dann die bei den gegebenen Informationen wahrscheinlichsten Positionen der Knoten wieder. Kennt man nun noch einige wenige Positionen absolut, so lässt sich damit vermutlich relativ genau

die Position aller Knoten ermitteln. Eine experimentelle Bestätigung dieser Vermutung steht allerdings noch aus.

Nachdem nun hinreichend gezeigt wurde, dass man mit vertretbarem Aufwand die Positionen von Knoten im MANET ermitteln und verfolgen kann und welche Auswirkungen daraus entstehen können, stellt sich die Frage, wie man dies verhindern kann. Damit beschäftigt sich der Rest dieses Kapitels.

9.4. Schutzmechanismen

Die Aussage „Ein Knoten befindet sich an Position x,y“ ist typischerweise für jemanden, der an der Erstellung und Auswertung von Bewegungsprofilen interessiert ist, wertlos. Erst im Zusammenhang – wenn also der Knoten sich konkret einer Person zuordnen oder er sich eine längere Zeit in seiner Bewegung verfolgen lässt – lassen sich verwertbare Informationen z.B. zum Kaufverhalten oder zum Aufenthaltsort einer bestimmten Person gewinnen.

Hier setzen nun die Schutzmechanismen an, indem sie zum Einen die Verknüpfung einer Adresse oder ID im Netzwerk mit einem konkreten Knoten oder einer Person verhindern und es zum anderen einem Knoten erlauben, parallel mehrere solcher Adressen oder IDs zu verwenden.

Den einfachsten Mechanismus stellt die Anonymität dar.

Definition 9.1 *A|no|ny|mi|tät <gr.-nlat.> die; -: das Nichtbekanntsein, Nichtgenanntsein; Namenlosigkeit. (aus [Wis90])*

Kann keine Verbindung zwischen einem handelnden Subjekt und seiner Identität hergestellt werden, so handelt dieses Subjekt anonym. Dies kann von Vorteil sein, wenn beispielsweise eine Straftat anonym bei der Polizei angezeigt wird, aber auch von Nachteil, wenn z.B. anonym im Internet verbotene Medien wie NS-Propaganda verteilt werden.

Im Internet gibt es verschiedene Systeme zur Realisierung von Anonymität. Viele bauen auf dem MIX Konzept von David Chaum auf [Cha81]. Dabei werden die Daten durch eine Reihe von Proxies, die sogenannte MIX Kaskade geleitet. In jedem Schritt werden die Daten mit dem öffentlichen Schlüssel des jeweils nächsten Proxies verschlüsselt und mit anderen Daten gemischt. Ein Angreifer, welcher die Kommunikation zwischen MIX Proxies abhören kann, ist nicht in der Lage, aus den Daten irgendwelche verwertbaren Informationen abzuleiten. Eine Vielzahl von weiteren Veröffentlichungen beschäftigen sich mit Anonymität im Kontext von Internet Kommunikation [SGR97], E-Commerce [SSG97], dem World Wide Web [RR98, ANO], Email [GT96] oder E-Cash [Way97].

In einem MANET ließe sich Anonymität relativ einfach realisieren, wenn es Knoten gestattet wird, jederzeit beliebige, zufällige Adressen zu verwenden. Ein Rückschluss auf das tatsächliche Gerät bzw. den Benutzer ist dann höchstens noch über die Analyse der übertragenen Daten möglich, wenn dort beispielsweise Benutzernamen übertragen werden. Dies kann durch Verschlüsselung der Datenkanäle verhindert werden. Schließlich verhindert die gleichzeitige Verwendung von mehreren Identitäten und Adressen durch einen Knoten, dass das Verhalten eines Knotens als eine generische Identität analysiert werden kann, um zum Beispiel verschiedene Einkäufe zu korrelieren.

Vollständige Anonymität kollidiert allerdings oft mit anderen Sicherheitszielen. Im Rahmen von SAM könnten Knoten in diesem Fall nicht authentifiziert werden und auch der Ausschluss von FEB Knoten durch MobIDS hätte keine Wirkung. Zwar könnte eine Adresse gesperrt werden, der Knoten kann jedoch im selben Moment mit einer beliebigen Anzahl von Adressen weiter im Netz aktiv bleiben.

Deshalb ist eine abgeschwächte Form von Anonymität notwendig, welche als Pseudonymität bezeichnet wird. Hierbei arbeiten die Knoten mit zusätzlichen Identitäten, den sogenannten Pseudonymen. Für unsere Zwecke seien Pseudonyme wie folgt definiert.

Definition 9.2 *Ein* Pseudonym *ist eine Identität, welche ein Knoten zusätzlich zur Hauptidentität annehmen kann. Ein trivialer Rückschluss vom Pseudonym auf die Hauptidentität ist nicht möglich. Im Gegensatz zur Anonymität kann ein Pseudonym aber unter genau definierten Umständen zur Hauptidentität aufgelöst werden.*

Das Wort Pseudonym selbst stammt vom griechischem *pseudonymos* ab und bedeutet soviel wie „unter falschem Namen“. Pseudonyme wurden in der Geschichte oft von Schriftstellern verwendet, um Werke verbotenen Inhalts zu publizieren.

9.5. Pseudonymisierungsverfahren in MANETs

Arbeiten zu Pseudonymen in MANETs finden sich bisher noch keine. Deshalb werden im Folgenden verschiedene eigene Ansätze entwickelt und miteinander verglichen. Folgende Anforderungen stellt SAM an das Pseudonymsystem:

Gleichwertigkeit: Es ist für einen Knoten im MANET nicht ersichtlich, ob es sich bei einer MANET-ID um eine Hauptidentität oder um ein Pseudonym handelt. Ein Pseudonym ist also, wie eine MANET-ID, ein Public-Key Schlüsselpaar. Auch müssen die Pseudonyme alle weiteren Eigenschaften der MANET-IDs tragen, u.a. eine gültige Signatur der MANET-CA.

Beschränkte Gültigkeit: Die Gültigkeit eines Pseudonyms soll beschränkt sein. Die Kontrolle der Gültigkeit soll dabei analog zu dem Mechanismus bei MANET-IDs erfolgen. Um die *Unverknüpfbarkeit* zu gewährleisten, sollen mehrere Pseudonyme eines Knotens unabhängige Gültigkeitsintervalle besitzen.

Unverknüpfbarkeit: Es soll einem anderen MANET Knoten nicht möglich sein, zwei Pseudonyme P_A^1 und P_A^2 eines Knotens A einander oder A zuzuordnen. Aus Sicht des anderen Knotens erscheinen P_A^1, P_A^2 und A als drei getrennte Identitäten.

Auflösung und Sperrung der Pseudonyme: Stellt MobIDS bei einem Knoten massives Fehlverhalten einer Hauptidentität oder eines Pseudonyms fest, so soll es möglich sein, die Hauptidentität sowie alle zugehörigen Pseudonyme dauerhaft zu sperren. Hierzu muss das Pseudonym gegebenenfalls zur Hauptidentität aufgelöst werden können. Diese Auflösung darf bei normalem Betrieb natürlich nicht möglich sein[5].

[5] Es sei denn, der Knoten veranlasst explizit eine Auflösung. Schickt ein Knoten eine sowohl mit der Hauptidentität, als auch dem Pseudonym signierte Nachricht, so hat er damit bewiesen, dass er sowohl die Hauptidentität als auch das Pseudonym kontrolliert.

Unabhängigkeit: Solange ein Pseudonym gültig ist, kann der Knoten es unabhängig von anderen Pseudonymen und insbesondere auch parallel zu diesen einsetzen.

Bereich: Die Pseudonymunterstützung beschränkt sich primär auf die Schicht des MANET Routing. Auswirkungen der Anwendungsschicht werden nicht untersucht.

Bei den Verfahren zur Erzeugung von Pseudonymen für MANET Knoten gibt es drei verschiedene Möglichkeiten:

- Im erstem Fall erzeugt der Knoten seine Pseudonyme einfach selbst, indem er beliebige zusätzliche Identitäten oder Adressen generiert. Die Auflösung eines Pseudonyms zur Hauptidentität kann dann natürlich nur vom Knoten selbst geleistet werden. Die Nachteile dieses Ansatzes sind vielfältig. Zeigt ein Knoten unter einem Pseudonym bewusst ein Fehlverhalten, wird er kaum bereit sein, seine wahre Identität öffentlich zu machen. Weiterhin kann kaum kontrolliert werden, dass ein Knoten nicht Pseudonyme in beliebiger Anzahl generiert. Ein Ausschluss eines Pseudonyms hätte somit keine Wirkung. Schlimmer noch könnte ein Knoten das Pseudonym eines anderen Knotens übernehmen. Später wäre dann nicht mehr zu unterscheiden, welcher der beiden Knoten unter diesem Pseudonym aufgetreten ist. Im Endeffekt entspricht diese Form der Pseudonymität weitestgehend der Anonymität und ist aus den dort aufgeführten Gründen für den Einsatz in SAM nicht geeignet.

- Im zweiten Fall werden die Pseudonyme von einer zentralen Instanz erstellt und an die Knoten ausgegeben. Die zentrale Instanz fungiert in diesem Fall als Trusted Third Party. Wird aus einem wichtigen Grund die Auflösung eines Pseudonyms notwendig, so kann die zentrale Instanz einfach die zu dem Pseudonym gehörende Hauptidentität liefern. Hierzu speichert sie für jedes ausgegebene Pseudonym die zugehörige Hauptidentität. Die zentrale Instanz kann auch die Menge der an einen Knoten ausgegebenen Pseudonyme kontrollieren und im Falle eines Missbrauchs die weitere Ausgabe neuer Pseudonyme ganz einstellen. Hauptnachteil dieser Lösung ist die zentrale Instanz, deren ständige Erreichbarkeit im MANET nicht sichergestellt werden kann.

- Schließlich gibt es auch die Möglichkeit einer Zwischenlösung, bei der eine Gruppe von Knoten im MANET die Ausgabe neuer Pseudonyme kontrolliert. Ein solcher Ansatz wäre ähnlich den Zertifizierungsverfahren mit Schwellwertkryptographie, wie sie in Abschnitt 8.1.2 und 3.6.3 vorgestellt wurden und hätte auch ähnliche Nachteile.

Im Folgenden wird zunächst ein Verfahren vorgestellt, bei dem der Knoten selbst Pseudonyme erstellt. Nachdem die Vor- und Nachteile dargelegt wurden, werden verschiedene Varianten eines zentralen Verfahrens für MANETs diskutiert. Es wird insbesondere untersucht, inwieweit sich das jeweilige Verfahren als optionale Komponente in die MANET-IDs integrieren lässt.

9.6. Abgeleitete Pseudonyme

Die Idee bei abgeleiteten Pseudonymen ist, dass ein Knoten aus einer von der MANET-CA zertifizierten MANET-ID neue Pseudonyme ableiten kann, ohne dass hierzu eine

neue Interaktion mit der CA notwendig ist. Die MANET-ID wird zunächst auf ein von der CA signiertes RSA-Schlüsselpaar reduziert. Aus diesem wird dann ein anderes RSA-Schlüssel abgeleitet, welches als Pseudonym verwendet wird. Die Signatur der CA soll dabei gültig bleiben.

Sei (PK_A, SK_A) das RSA-Schlüsselpaar des Benutzers A. Dabei ist SK_A der öffentliche Teil des Schlüssels, A_d wird geheim gehalten. Sei (PK_{CA}, SK_{CA}) das RSA-Schlüsselpaar der CA. Die CA bildet jetzt eine Signatur über PK_A der Form

$$S = ({PK_A}^{SK_{CA}} \bmod n)$$

Für die Signatur kommt dabei *keine* Hashfunktion zum Einsatz, die Signatur ist also genauso lang wie ein Schlüsselteil selbst. Jetzt können neue Pseudonyme für A gebildet werden:

$$P_A^x = ({PK_A}^x \bmod \varphi(n), SK'_A)$$

SK'_A muss hierbei gemäß dem erweiterten Euklidischen Algorithmus aus ${PK_A}^x$ (bei bekanntem p, q und n) berechnet werden. Dabei gilt weiterhin

$$m^{{PK_A}^x} \bmod n = c \Rightarrow c^{SK'_A} \bmod n = m$$

d.h. das neue Pseudonymschlüsselpaar kann für normale RSA Operationen wie Ver-/Entschlüsselung oder Signatur verwendet werden. Als nächstes soll gezeigt werden, dass sich die Signatur S von CA auf einfache Weise in S^x überführen lässt, so dass S^x auch für das Pseudonym P_A^x gültig bleibt. Sei

$$S^x = ({PK_A}^{SK_{CA}} \bmod n)^x \bmod n \tag{9.1}$$

Es bleibt jetzt zu zeigen, dass sich diese Signatur mit PK_{CA} prüfen lässt, also

$$(S^x)^{PK_{CA}} \bmod n \stackrel{?}{=} {PK_A}^x$$

Beweis 9.1

$$\begin{aligned} S^x &= ({PK_A}^{SK_{CA}} \bmod n)^x \bmod n \\ &= ({PK_A}^{SK_{CA}})^x \bmod n \\ &= ({PK_A}^x)^{SK_{CA}} \bmod n \\ \Rightarrow & \\ (S^x)^{PK_{CA}} \bmod n &= (({PK_A}^x)^{SK_{CA}} \bmod n)^{PK_{CA}} \bmod n \\ &= {PK_A}^x \bmod n \\ q.e.d. & \end{aligned}$$

Benutzt Knoten A sein Pseudonym P_A^x und will einem anderen Knoten B seine Identität beweisen, so kann er dies einfach tun, indem er x an B schickt. B kann dann analog

zur Erstellung des Pseudonyms prüfen, ob der öffentliche Schlüssel ${PK_A}^x$ tatsächlich dem öffentlichen Pseudonymschlüssel entspricht.

Ähnlich wie beim Diffie-Hellmann Verfahren (siehe Abschnitt 3.4.1) beruht die Sicherheit hier darauf, dass kein Knoten bei Kenntnis von ${PK_A}^x$ effizient den diskreten Logarithmus x berechnen kann.

Kennt ein Knoten B allerdings einmal x, so kann er problemlos anderen Knoten den Zusammenhang zwischen dem Knoten A und seinem Pseudonym beweisen, indem er einfach x weiter gibt.

Jeder Knoten kann bei einem Pseudonym problemlos testen, ob dieses eine gültige Signatur der CA aufweist und somit aus einem gültigen Schlüssel abgeleitet wurde. Somit kann sich kein externer Knoten unter einem gefälschten Pseudonym in das System einschleichen.

Bewertung

Das System liefert ein einfaches Verfahren zur Erzeugung von Pseudonymen. Besonders hervorzuheben ist die Tatsache, dass die Signatur der CA vom Knoten so angepasst werden kann, dass sie weiterhin gültig ist. Ein Knoten kann sehr einfach den Besitz eines Pseudonyms beweisen, indem er das zugehörige x veröffentlicht.

Es bleibt zu untersuchen, inwieweit das Verfahren die oben aufgestellten Anforderungen erfüllt:

Gleichwertigkeit: Ein beliebiger Knoten im Netz kann nicht entscheiden, ob es sich bei einem vorgelegten öffentlichen Schlüssel um ein Pseudonym oder um eine Hauptidentität handelt. Gleichwertigkeit ist somit gegeben.

Beschränkte Gültigkeit: Um die Gültigkeit eines Pseudonyms wirksam zu beschränken, muss die CA weitere Informationen – insbes. ein Ablaufdatum – in den Schlüssel integrieren können. Beim vorliegenden Ansatz ist dies aber nicht möglich, da diese Informationen in der Signatur bei der Transformation in Formel 9.1 unweigerlich zerstört werden. Somit kann dieser Punkt nicht erfüllt werden.

Unverknüpfbarkeit: Ist in vollem Umfang gegeben. Verschiedene Schlüssel können nicht korreliert werden.

Unabhängigkeit: Ist in vollem Umfang gegeben. Die verschiedenen Schlüssel können beliebig eingesetzt werden.

Auflösung und Sperrung der Pseudonyme: Da außer dem Knoten niemand den Wert x kennt, kann ihm auch niemand ein Pseudonym zuordnen. Lediglich er selbst ist hierzu in der Lage.

Insbesondere die Punkte *beschränkte Gültigkeit* und *Auflösung und Sperrung der Pseudonyme* verhindern einen Einsatz dieser Lösung im Rahmen von SAM. Den letzten Punkt könnte man dadurch lösen, dass nicht der Knoten die Pseudonyme generiert, sondern die CA ausgehend von einer nur ihm bekannten Master-Identität die Pseudonyme an die Knoten ausgibt. Kommt es bei einem Pseudonym zu einer Beschwerde, so kann die CA dies anhand gespeicherter Daten zurückverfolgen und alle zu diesem Masterkey gehörenden Pseudonyme zurückrufen. Werden die x-Werte durch eine

Hashchain oder einen Pseudozufallszahlengenerator erzeugt, so lassen sich auch größere Zahlen von Pseudonymen effizient speichern und zurückrufen.

Es bleibt trotzdem das Problem, dass sich kein Ablaufdatum oder sonstige Information im Zertifikat unterbringen lässt. Sollen verschiedene Pseudonyme unterschiedliche Ablaufdaten erhalten, so muss die CA jeden Schlüssel einzeln signieren. Der Vorteil der abgeleiteten Pseudonyme entfällt. Deshalb werden im weiteren Lösungen untersucht, bei der die CA Pseudonyme signiert.

Trotz der Nachteile stellen abgeleitete Pseudonyme ein sehr effizientes Verfahren mit wertvollen Eigenschaften dar. Hierzu sind Anwendungsszenarien ohne die problematischen Einschränkungen notwendig. Denkbar ist beispielsweise ein mobiles Online-Shopping Szenario mit Verhandlungskomponente und beschränktem Nutzerkreis. Ein mobiler Knoten führt unter selbst abgeleiteten Pseudonymen Verhandlungen mit einer Vielzahl von Händlern. Die Händler können sich anhand der Signatur bereits davon überzeugen, dass sie mit einem gültigen Teilnehmer kommunizieren. Gültige Teilnehmer können sich dabei z.B. auf Kunden eines Kreditkartenunternehmens o.Ä. beschränken. Kommt es bei einem Händler nach längerer Verhandlung zu einer Einigung, so legt der Kunde sein Pseudonym offen und beweist damit, dass er tatsächlich Anspruch auf den ausgehandelten Preis hat.

9.7. CA-signierte Pseudonyme

Letztlich wird eine von der CA unabhängige Lösung zur Erzeugung von Pseudonymen nicht funktionieren. Ein Pseudonym benötigt eine eingeschränkte Gültigkeitsdauer. Diese muss aber in der Signatur verankert werden, da sie der Knoten sonst beliebig modifizieren kann. Da verschiedene Pseudonyme unterschiedliche Gültigkeitsdauern benötigen[6], muss also die CA jedes Pseudonym getrennt signieren.

Die Pseudonyme werden dabei vollkommen analog den MANET-IDs aus dem letzten Kapitel aufgebaut. Dabei werden Pseudonyme als *Pseudo-MANET-IDs* bezeichnet, wohingegen die Hauptidentitäten der Knoten weiterhin *MANET-IDs* heißen. Genauso wie bei den MANET-IDs wird vorausgesetzt, dass der Client sporadischen Kontakt mit der CA hat, da nun auch die Pseudonymschlüssel verlängert und aktualisiert werden müssen. Die verbleibende Frage ist lediglich, ob der Client oder die CA die Pseudonymschlüssel generiert[7].

9.7.1. Klient generiert Schlüsselpaar

Hierbei generiert sich der Knoten A zunächst eine beliebige Anzahl von Schlüsselpaaren (PK_A^i, SK_A^i), welche jeweils das Pseudonym P_A^i ergeben. Hat er im Rahmen der normalen Identitätsverlängerung Kontakt mit der CA, so schickt er ihr die öffentlichen Schlüssel signiert und verschlüsselt.

$$A \rightarrow CA \; : \; E_{PK_{CA}} \left(E_{SK_A} \left(PK_A^i,\, PK_A^j,\, PK_A^k,\, \ldots \right) \right)$$

[6] sonst könnten sie korreliert werden

[7] womit die CA faktisch zum KDC wird. Es ist weiterhin trotzdem von CA die Rede

Die CA prüft zunächst die Gültigkeit der Nachricht anhand der Signatur und erzeugt aus den öffentlichen Schlüsseln vollwertige Pseudo-MANET-IDs, indem Gültigkeitsdaten und Verifikatoren hinzugefügt werden. Die Gültigkeitsdaten werden dabei zufällig in einem geeigneten Zeitraum verteilt, so dass keine Korrelation der Pseudonyme möglich ist. Dass dadurch manche der Pseudonyme eine kürzere Gültigkeit haben, ist kein tatsächliches Problem, da ein Knoten jederzeit ein altes Pseudonym durch ein komplett neues ersetzen kann. Details dazu folgen später.

Anschließend werden die Pseudo-MANET-IDs von der CA signiert und die Zertifikate an den Client zurück geschickt.

$$CA \rightarrow A \; : \; E_{PK_A}\left(E_{SK_CA}\left(cert_A^i,\, cert_A^j,\, cert_A^k,\, \ldots\right)\right)$$

Der Client kann nun die Pseudonyme wie reguläre MANET-IDs verwenden. Die Anforderungen an Pseudonyme sind gänzlich erfüllt. Insbesondere kann die CA auch Pseudonyme sperren. Hierzu muss sie aber alle ausgestellten und noch aktiven Pseudo-MANET-IDs eines Knotens speichern, um im Falle eines Missbrauchs jederzeit die zugehörige MANET-ID sowie weitere zum Knoten gehörende Pseudo-MANET-IDs ebenfalls zu sperren. Im Analysekapitel wird gezeigt, dass der hierzu notwendige Aufwand[8] im Rahmen der heutigen Technologie durchaus handhabbar bleibt.

9.7.2. Klient und KDC generieren Schlüsselpaar

Durch eine Modifikation bei der Schlüsselgenerierung könnte der Speicheraufwand reduziert werden. Hierbei wird das RSA Schlüsselpaar (PK_A^i, SK_A^i) analog wie bei den abgeleiteten Pseudonymen generiert:

1. A wendet sich an die CA und beantragt n neue Pseudonyme.
2. Die CA generiert daraufhin eine Hashchain, ausgehend von einem zufälligen Startwert y_0, wobei $y_i = h(y_{i-1})$. y_0 wird an A geschickt.
3. A berechnet nun die Pseudonymschlüssel P_A^i als $P_A^{y_i} = ({PK_A}^{y_i}, SK'_A)$. SK'_A wird wieder mittels erweitertem euklidischem Algorithmus aus ${PK_A}^{y_i}$ berechnet.
4. Nun schickt A die öffentlichen Schlüssel ${PK_A}^{y_i}$ an die CA.
5. Die CA prüft, ob die Schlüssel tatsächlich der vorgegebenen Hashchain entsprechen. Wenn ja, werden die Zertifikate analog dem Vorschlag bei Client generierten Schlüsseln mit einem Ablaufdatum und Verifikator versehen und signiert.
6. Die CA verschickt die Zertifikate an A.
7. A kann daraufhin die Pseudo-MANET-IDs wie reguläre MANET-IDs nutzen.

Die Kommunikation erfolgt auch hier verschlüsselt und signiert. Der Vorteil dieser Lösung ist, dass die CA nun nicht mehr alle ausgegebenen Pseudonymschlüssel speichern muss und trotzdem die geheimen Schlüssel nur dem Klienten bekannt sind. Es genügt die Speicherung des Startwertes der Hashchain y_0 und der Anzahl der generierten Pseudonyme, um daraus alle Pseudonymschlüssel zu regenerieren. Nachteil ist

[8] die zu verwaltende Datenmenge liegt im Bereich von mehreren hundert Gigabyte

der hohe Rechenaufwand, den diese Lösung dafür benötigt. Wird eine MANET-ID als auffällig gemeldet und soll gesperrt werden, so muss die CA der Reihe nach die ausgegebenen ${PK_X}^i$ für alle bekannten Knoten X anhand der Hashchains wiederherstellen und prüfen, ob das Ergebnis mit dem gesuchten öffentlichen Schlüssel übereinstimmt.

9.7.3. CA generiert Schlüsselpaar

Vertraut man der CA auch die geheimen Pseudonymschlüssel an, so kann diese natürlich die Aufgaben des Clients sowohl beim Verfahren mit Client generiertem Schlüsselpaar als auch beim gemeinsamen Verfahren übernehmen. Der Knoten A bekommt dann lediglich neben den Zertifikaten auch noch die Schlüsselpaare für die Pseudonyme geschickt. Allerdings muss man hier dann noch Frischeinformationen in die Nachrichten zwischen dem Knoten und der CA packen, da sonst Replay Angriffe möglich werden.

9.7.4. Bewertung

Letztlich ist die Entscheidung, ob Schlüssel vom Klienten oder gemeinsam generiert werden, eine Abwägung zwischen notwendigem Speicherplatz und Rechenaufwand. Wobei auch im zweiten Fall die CA natürlich die generierten Schlüssel speichern kann und somit nicht über die Hashchain rekonstruieren muss. Beide Verfahren erfüllen alle Anforderungen an das Pseudonymsystem und können je nach Randbedingungen alternativ eingesetzt werden. Vorteil der Lösung mit Hashchains: der Klient hat keine vollständige Kontrolle über die Schlüssel, vielmehr gibt diese der Server vor. Trotzdem kennt dieser die geheimen Schlüssel nicht. Bei der ersten Lösung könnte der Klient von der CA alte Pseudonyme erneut signieren lassen, ohne dass diese das bemerkt. Damit wäre das gleiche Pseudonym mit unterschiedlichen Verfallsdaten im Netz aktiv, was zu einiger Verwirrung führen könnte.

9.7.5. Einsatz von Pseudonymen in SAM

Einige Aspekte im Zusammenhang mit dem Einsatz eines Pseudonymsystems in SAM bleiben noch offen. Zum einen müssen Pseudonyme, wie die MANET-IDs auch, verlängert werden. Eine Verlängerung der Zertifikatslebensdauer ist dabei nicht vorgesehen. Statt dessen fordert der Knoten ein neues Pseudonym an. Was vorgesehen ist, ist die Validierung von Pseudonymen mittels Verifikatoren. Das Protokoll zur Generierung neuer Verifikatoren ist dabei vollkommen analog zu den MANET-IDs.

Die Zahl der Pseudonyme eines Knotens muss begrenzt bleiben. Hätte ein Knoten beliebig viele Pseudonyme zur Auswahl, so könnte er die Sensoren des Intrusion Detection Systems MobIDS effektiv unterlaufen, indem er unter einem Pseudonym immer nur soviel Schaden anrichtet, dass er unter den Erkennungsschwellwerten bleibt. Dann wechselt er zum nächsten Pseudonym. Indem die Zahl der Pseudonyme begrenzt wird, ist auch der mögliche Schaden kontrollierbar. Wir gehen bei unseren weiteren Überlegungen davon aus, dass einem Knoten nur eine kleine Zahl von Pseudonymen (ca. 5) zur Verfügung steht. Dies kann die CA kontrollieren, indem sie dem Knoten nicht mehr Zertifikate mit überlappender Gültigkeitsdauer zur Verfügung stellt.

Bei fünf Pseudonymen muss ein Knoten allerdings Pseudonyme unter Umständen öfters einsetzen, was wiederum die Qualität möglicher Bewegungsprofile erhöht. Deshalb kann ein Knoten neue Pseudonyme anfordern, indem er alte „zurückgibt“. Statt einen neuen Verifikator anzufordern, verlangt der Knoten stattdessen ein neues Pseudonym. Dieses liefert die CA aber erst aus, wenn der alte Verifikator abgelaufen ist, also frühestens nach einem Tag.

Generell gilt es, eine Abwägung zwischen dem Schutz der Privatsphäre und der Sicherheit des Netzes zu finden. Je größer die Zahl der Pseudonyme, die ein Knoten gleichzeitig nutzen kann, desto besser kann er sich vor der Erstellung von Bewegungsprofilen schützen. Gleichzeitig wird es für ihn damit aber auch leichter, den Sensoren von MobIDS zu entgehen. Egoistisches oder böswilliges Verhalten wird damit unter Umständen erst später erkannt.

9.8. Fazit

In diesem Kapitel wurde das MANET-ID System um die Unterstützung von Pseudonymen erweitert. Diese Komponente ist optional und schützt die Knoten davor, dass ihre Privatsphäre durch die Erstellung von Bewegungsprofilen verletzt wird. Es wurden mehrere alternative Systeme zur Erstellung von Pseudonymen vorgestellt und diskutiert. Wie schon bei den MANET-IDs, ist auch hier eine zentrale Lösung über die CA wegen der besseren Funktionalität vorzuziehen. Die schon bei der MANET-ID Lösung entstehenden Nachteile (gelegentliche Internetverbindung) werden durch die Pseudonymunterstützung nicht weiter vergrößert. Die funktionalen Vorteile wiegen die Nachteile deutlich auf.

Im nächsten Kapitel wird nun das sichere Routingprotokoll SDSR vorgestellt. Dieses setzt auf den MANET-IDs zur Identifizierung von Knoten auf. Die Verwendung von Pseudonymen macht hierbei keinerlei Unterschied. Erst beim Intrusion Detection System MobIDS spielt der Einsatz von Pseudonymen eine Rolle.

10. Secure Dynamic Source Routing - SDSR

Wie in den Abschnitten 6.2 und 6.5 gezeigt, bieten die Routingprotokolle selbst zunächst keine Sicherheitsmechanismen und sind anfällig für eine Vielzahl von Angriffen, welche die Routinginformationen verändern und somit zu „falschen“ Routen führen. Dies kann ein Angreifer ausnutzen, um Routen gezielt um sich herum- oder zu sich hinzulenken, um andere Knoten zu überlasten oder um schlicht das Netz so durcheinander zu bringen, dass keine sinnvolle Kommunikation mehr möglich ist. Sichere Routingprotokolle versuchen dies zu verhindern.

Im Verlauf dieses Kapitels werden zunächst verschiedene Arbeiten anderer Forscher zum Thema „Sicheres Routing in MANETs“ vorgestellt. Es folgt eine Beschreibung eines eigenen Protokolls namens *Secure Dynamic Source Routing* (*SDSR*).

Die Ziele eines sicheren Routing Protokolls lassen sich relativ klar umreißen:

1. Die am Routing beteiligten Knoten werden authentifiziert, *Spoofing von Nachrichten ist nicht möglich.*
2. Außenstehende können *keine generierten Routingnachrichten in das Netz einschleusen.*
3. Die *Unveränderlichkeit der Routingnachrichten* muss sicher gestellt sein, d.h. kein Knoten darf in der Lage sein, Elemente der Nachricht zu löschen oder auszutauschen, es sei denn, dies geschieht im Rahmen der korrekten Protokollfunktionalität.
4. Ein FEB Knoten kann *keine Routingschleifen oder unnötig lange Routen* erzeugen.
5. Die *Authentizität und Aktualität der Route* muss sichergestellt sein, d.h. die Source Route muss auch tatsächlich die beim aktuellen Route-Request/-Reply beteiligten Knoten widerspiegeln.
6. *Unautorisierte Knoten* sollen beim Routing nicht berücksichtigt werden.
7. Wie üblich bei MANETs, soll der durch das Routing-Protokoll erzeugte *Overhead minimiert* werden. Insbesondere sollten aufwändige Public Key Operationen soweit möglich vermieden werden.

10.1. Verwandte Arbeiten

Es gibt bereits einige Arbeiten, die sich mit sicherem Routing in Ad hoc Netzen beschäftigen. Diese werden im Folgenden vorgestellt.

10.1.1. SAODV

Secure AODV [Gue02a, GA02, Gue02b] wurde von Mitarbeitern von Nokia Research entwickelt, welche auch an der Entwicklung von AODV (siehe Abschnitt 5.3.2) beteiligt waren. Es ist eine direkte Weiterentwicklung von AODV mit dem Ziel, verschiedene Angriffe gegen das Routing Protokoll zu verhindern.

Hierzu geht SAODV davon aus, dass es ein funktionsfähiges Key Management System gibt, welches die Public Keys der Knoten verwaltet und sicher an die Teilnehmer des Netzes verteilt. Die statischen Anteile eines AODV Route Request/Reply Paketes werden durch eine *klassische Signatur* mit dem privaten Schlüssel des Absenders vor Veränderungen geschützt. Der einzig variable Teil im Paket, der Hopcount, ist von dieser Signatur ausgenommen.

Hier sorgt eine Hashchain dafür, dass jeder Knoten den Hopcount lediglich vergrößern, aber niemals verkleinern kann. Der Absender erzeugt dazu ausgehend von einem Startwert *seed* eine Hashchain wobei $TopHash = h^{max\_hop\_count}(seed)$.

Dieser Wert wird im signierten Teil des Paketes mitgeschickt. Außerdem setzt der Absender das *Hash*-Feld des Paketes auf *TopHash*. Jeder Zwischenknoten prüft nun die Signatur über die statischen Daten und ob $TopHash \stackrel{?}{=} h^{max\_hop\_count - HopCount}(Hash)$. Wenn ja, entspricht der aktuelle Hopcount also genau der momentanen Position in der Hashchain. Vor der Weiterleitung erhöht der Knoten nun den *HopCount* um eins und setzt $Hash = h(Hash)$, entfernt also ein Element aus der Hashchain.

Mit diesen Mechanismen lassen sich die normalen Route Requests und Replies absichern. AODV erlaubt es einem Zwischenknoten, einen Route Request direkt zu beantworten, wenn er eine gültige Route zum Ziel kennt. Dieses sog. *Route Caching* bereitet in sicheren Routing Protokollen einige Schwierigkeiten. Da hier der Zielknoten nicht am Route Reply beteiligt ist, kann der Route Reply auch nicht von ihm signiert werden. SAODV verwendet hierzu die sogenannte *Double Signature Extension*. Dabei verschickt der Zwischenknoten im Route Reply zum einen die Originalsignatur des Zielknotens, die er beim Erhalt der Route bekommen hat, zum anderen fügt er eine neue Gültigkeitsdauer für die Route hinzu, welche er mit seinem eigenen Schlüssel signiert. Der Empfänger des RREP kann dann einerseits sicherstellen, dass die ursprüngliche Route tatsächlich vom Zielknoten stammt, andererseits kann der Zwischenknoten die Gültigkeitsdauer der Route anpassen.

Weitere Teile der Spezifikation von SAODV beschäftigen sich mit der Behandlung von Route Errors, dem Verhalten beim Reboot von Knoten usw. Dies soll hier aber nicht weiter ausgeführt werden.

Bewertung

SAODV ist eine relativ einfache und klar strukturierte Erweiterung von AODV, die aufwändige Public-Key Operationen zum Teil vermeidet. Allerdings zeigt die Arbeit auch eine Reihe von Schwächen.

Zum einen wird ein komplexes Key Management zwar postuliert, aber nicht weiter ausgeführt. Insbesondere bleibt unklar, wie die Knoten an die Schlüssel kommen sollen, wenn noch gar keine Routen existieren.

Weiterhin beschäftigt sich SAODV ausschließlich mit der Absicherung der Routingdaten. Wie dann der weitere Datenverkehr zu sichern ist, bleibt vollkommen offen. Dies ist insbesondere deshalb ein Nachteil, weil sich der Austausch von Sitzungsschlüsseln – wie bei SDSR gezeigt – sehr gut in den Route Request/Reply integrieren lässt.

Während der Route Discovery muss jeder Knoten, der einen RREQ erhält, die Signatur des Absenders prüfen. Dies ist ein relativ aufwändiger Vorgang, der bei SDSR erst auf die Route-Reply Phase verschoben wird, wo ihn nicht mehr das gesamte MANET ausführen muss.

Auch die Verwendung gecachter Routen ist kritisch zu sehen. Obwohl es aus Performancesicht wünschenswert wäre, erlauben gecachte Routen einem Angreifer zumindest, veraltete Topologieinformationen im Netz zu verbreiten. Weiß der Angreifer[1], dass eine Route nicht mehr gültig ist, so kann er trotzdem mittels des veralteten Route Replies Verkehr auf diese Route lenken.

Schließlich bietet auch SAODV noch eine Reihe von weiteren möglichen Angriffspunkten. Der Hopcount ist durch die Hashchain zwar davor geschützt, dass ihn ein Zwischenknoten erniedrigt, d.h. dass er eine Route *attraktiver*, weil kürzer, gestaltet und somit Verkehr anziehen kann[2]. Ein Zwischenknoten kann den Hopcount jedoch beliebig hochsetzen, d.h. die Route wird länger und somit unattraktiver. Gegen egoistisches Verhalten hilft SAODV also nicht.

Zusammenfassend lässt sich sagen, dass SAODV nicht gegen alle möglichen Angriffe schützt und verschiedene Fragestellungen überhaupt nicht betrachtet. Die Verwendung herkömmlicher Signaturen zum Schutz der Protokollpakete entspricht einem Standardvorgehen. Lediglich die Verwendung der Hashchains zum Schutz des TTL Feldes ist wirklich neu.

10.1.2. Ariadne

Ähnlich wie SAODV arbeitet auch *Ariadne* [HPJ02] mit Hashchains zur Integritätssicherung der Routingnachrichten. Ariadne arbeitet dabei mit Source Routen und lehnt sich eng an DSR an (siehe Abschnitt 5.3.3). Eine andere Arbeit der gleichen Autoren überträgt einige der Ideen von Ariadne auf Distance Vector Routing [HJP02]. Dies soll aber hier nicht weiter interessieren.

Ariadne kennt drei Betriebsarten:

- **Ariadne mit symmetrischer Verschlüsselung:** in diesem Fall geht Ariadne davon aus, dass zwischen allen Knoten paarweise geheime Schlüssel vorhanden sind. Hierzu schlagen die Autoren einige Verfahren vor, die jedoch bis auf SPINS [PSW+01] allesamt nicht für den Einsatz in Ad hoc Netzen ausgelegt sind.
- **Ariadne mit digitalen Signaturen:** hier wird vorausgesetzt, dass authentische Public Keys aller Knoten im Netz verfügbar sind.
- **Ariadne mit TESLA:** hier verwenden die Knoten das TESLA System zur Absicherung von Broadcast-Nachrichten

[1] z.B. durch einen Route Error
[2] z.B. für Blackhole Angriffe

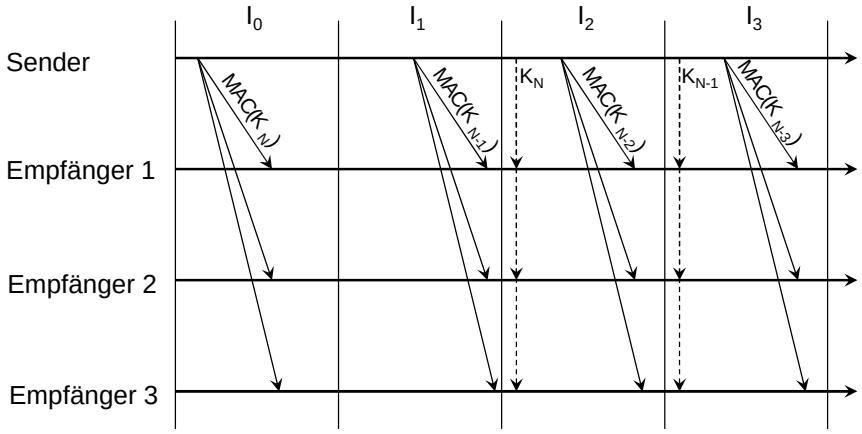

Abbildung 10.1.: Hashchain Authentisierung bei TESLA

TESLA

Die Variante mit TESLA [PCTS00, PCST01] ist die leistungsfähigste und interessanteste dieser drei Variante und wird im Folgenden beschrieben. Ziel von TESLA ist es, Nachrichten in einem Broad- oder Multicast-Szenario zu authentisieren. Dazu berechnet der Sender über die Nachricht einen MAC, in den ein Schlüssel K_i eingeht. Ausgehend von einem Initialschlüssel K_N baut ein Sender eine Hashchain auf mit $K_i = h(K_{i-1}) = h^{N-i}(K_N)$. Wie in Abbildung 10.1 gezeigt, wird nun die Zeit in Intervalle I_i eingeteilt. In jedem Intervall ist der Schlüssel K_{N-i} gültig, den der Sender zum Verschlüsseln der Nachrichten verwendet. Diesen Schlüssel verschickt der Knoten in einem späteren Intervall $I_{i+\delta}$ an alle Teilnehmer, die daraufhin in der Lage sind, die Authentizität der Nachricht zu verifizieren.

Die Sicherheit von TESLA beruht nun darauf, dass bei Veröffentlichung eines Schlüssels bereits alle damit verschlüsselten Nachrichten die Empfänger erreicht haben müssen. Kommen Nachrichten verspätet an, so können sie nicht mehr authentifiziert werden und werden verworfen. Die Gültigkeit eines Schlüssels kann aus dem Schlüssel des vorhergehenden Intervalls abgeleitet werden. Damit TESLA funktioniert, müssen die Knoten über hinreichend genau synchronisierte Uhren verfügen.

Ariadne

Die wesentlichen Funktionen von Ariadne sind:

Ziel authentifiziert Sender von Route Request: In der Version mit TESLA geht Ariadne nun lediglich davon aus, dass Quellknoten S und Zielknoten D über einen gemeinsamen geheimen Schlüssel K_{SD} verfügen. S bildet mit diesem Schlüssel einen MAC über den Route Request, wodurch der Zielknoten D die Authentizität des Route Requests verifizieren kann.

Sender authentifiziert Zwischenknoten: Bevor die Zwischenknoten einen Route Request weiterleiten, fügen sie sich analog wie bei DSR zur Source-Route hinzu und berechnen dann mit ihrem aktuell gültigen TESLA-Schlüssel K_i einen MAC.

Während des Route Replies geben die Zwischenknoten ihre verwendeten TESLA-Schlüssel frei, womit S diese am Ende authentifiziert. Alternativ zu TESLA kann die Authentifizierung der Zwischenknoten auch über gemeinsame geheime Schlüssel oder Signaturen mit Public Key Kryptographie erfolgen.

Per-hop Hashing: Um sicherzustellen, dass keine Knoten aus der Liste ausgelassen werden, verwendet Ariadne Per-Hop Hashing. Hierzu besitzt das Route Request Paket ein *Hash*-Feld, welches der Absender auf einen aus K_{SD} berechneten Wert setzt. Jeder Zwischenknoten A berechnet nun $Hash = H(A, Hash)$. Am Ende kann D durch simple Wiederholung der Berechnung prüfen, ob alle Zwischenknoten auch tatsächlich in der Source Route und in der Hashchain auftauchen. Somit kann kein Knoten frühere Knoten aus der Source-Route hinauswerfen.

Weitere Aspekte von Ariadne sind die Routenpflege mit Route Error Paketen und ein einfacher Schutz vor DoS durch Fluten des Netzes mit gefälschten Route Request Paketen.

Bewertung

Ariadne stellt ein deutlich fortgeschrittenes Sicherheitsprotokoll für Routing in Ad hoc Netzen dar. Insbesondere der weitgehende Verzicht auf Public Key Kryptographie liefert gute Performancewerte. Die Grundidee geht etwas weiter als bei SAODV, da hier auch die Zwischenknoten explizit authentifiziert werden[3]. Bei SAM eröffnet dies zusätzliche Möglichkeiten zur Erkennung von FEB Knoten durch MobIDS.

Ariadne hat auch eine Reihe von Schwächen, die allesamt darauf beruhen, dass die Autoren nur einen kleinen Ausschnitt einer umfassenden Sicherheitsinfrastruktur betrachten. Die Autoren erläutern nicht, woher notwendige gemeinsame geheime Schlüssel kommen oder wie diese authentifiziert werden. Solche Schlüssel sind aber notwendig, egal welches Verfahren zur Authentifizierung der Zwischenknoten genutzt wird. Bei Verwendung von TESLA bleibt unklar, wie die Anfänge der TESLA Hashchains verteilt werden. Erschwerend kommt hinzu, dass synchronisierte Uhren benötigt werden. Die Autoren geben zwar an, dass eine lose Synchronisation genügt, je größer aber die Zeitdrift zwischen den Knoten, desto länger muss für die Veröffentlichung eines Schlüssels gewartet werden. Dies kann dazu führen, dass ein Zielknoten zunächst lange warten muss, ehe er einen Route Reply verschickt. Dies führt zu unnötig langen Delays bei der Routensuche. Generell kann darüber diskutiert werden, inwieweit eine Uhrensynchronisation zwischen MANET Knoten überhaupt möglich und sinnvoll ist. MANET-IDs benötigen zwar auch eine Synchronisation, die notwendige Genauigkeit liegt aber im Bereich von Stunden, wohingegen sie bei TESLA eher im Sekundenbereich liegt.

10.1.3. ARAN

Das ARAN Protokoll („*Authenticated Routing for Ad hoc Networks*“) wird in [SDL+02] beschrieben. Vom Prinzip her arbeitet ARAN wie ein on-demand link state Protokoll und ist eng mit AODV verwandt. Um seine Ziele – Authentisierung, Integrität und Verbindlichkeit – zu erreichen, werden jedoch grundsätzlich alle Nachrichten signiert.

[3] Dies ist bei einem Distance-Vector Protokoll wie SAODV prinzipbedingt nicht sinnvoll, da hier jeweils nur Informationen zum nächsten Hop verarbeitet werden.

ARAN setzt hierzu die Existenz einer Zertifizierungsstelle voraus, welche die öffentlichen Schlüssel der Knoten vor Teilnahme am MANET signiert. Pro Knoten darf dabei nur ein Schlüssel zertifiziert werden. Ein Zertifikat in ARAN hat die Form:

$$CA \rightarrow A \ : \ cert_A = [IP_A,\ PK_A,\ t,\ e]\ SK_{CA}$$

Es enthält also die IP Adresse des Knoten, seinen öffentlichen Schlüssel, sowie Zeitstempel für die Gültigkeitsdauer und ist von der CA signiert.

Knoten A kann nun ein *Route Discovery Paket* (RDP) an seine Nachbarn verschicken:

$$A \rightarrow brdcast \ : \ [RDP,\ IP_X,\ cert_A,\ N_A, t]\ SK_A$$

Das Paket beinhaltet einen Pakettyp (RDP), die Adresse des Zielknotens X, das Zertifikat von A und eine Nonce und einen Timestamp gegen Replays. Das gesamte Paket ist von A signiert.

Ein Zwischenknoten B prüft zunächst die Signatur und sendet das Paket dann in folgender Form weiter.

$$B \rightarrow brdcast \ : \ [[RDP,\ IP_X,\ cert_A,\ N_A, t]\ SK_A]\ SK_B$$

Er zertifiziert also das Paket seinerseits. Ein nächster Knoten C entfernt nun die Signatur von B und hängt seine eigene an. Dies setzt sich bis zum Zielknoten fort. Beim Route Reply Paket läuft der Prozess analog in umgekehrter Richtung ab. Schließlich wird noch eine Route Error Nachricht spezifiziert. Diese wird vom Initiator des Route Errors signiert und beim Transport nicht weiter verändert.

Bewertung

Die Autoren zeigen in ihrer Arbeit, dass sich mit ARAN eine Vielzahl von Modifikationen verhindern oder zumindest erkennen lassen. Indem sie auch den Aspekt der notwendigen CA kurz beleuchten, geht ihr Ansatz bereits weiter, als dies bei den meisten anderen der Fall ist.

Indem ARAN eine CA postuliert, welche im Vorfeld die öffentlichen Schlüssel der Knoten signiert, ergeben sich gewisse Parallelen zu den MANET-IDs. Allerdings ist der Zertifikatsrückruf bei ARAN nur rudimentär über einfache Broadcastnachrichten mit Revocation Lists realisiert.

Ein Problem bei ARAN ist, dass es pro weitergeleitetem RDP Paket einiges an Zustand speichern muss, insbesondere die Nonce und den Timestamp sowie die öffentlichen Schlüssel der beteiligten Knoten. Dies führt zu einigem Speicherbedarf. Weiter reagiert auch ARAN nicht auf egoistisches Verhalten, ein Knoten kann eine Route zum Beispiel beliebig verlängern.

Der größte Nachteil von ARAN besteht jedoch in der Vielzahl von digitalen Signaturen, welche während der Route Discovery verifiziert und erstellt werden müssen. Da ARAN keine Einschränkung der Suchreichtweite über ein TTL Feld kennt, wird das RDP

Paket immer im gesamten Netz geflutet. Jeder Knoten muss pro empfangenem RDP Paket zwei Signaturen prüfen sowie eine neue Signatur generieren, bevor er den Route Request weiterleiten kann. Entsprechend haben die Autoren als Voraussetzung für ihre Simulationen auch die Leistung eines Notebooks mit einem Intel Pentium III 750MHz Prozessor angenommen.

10.1.4. SRP

Als letzter Vertreter sicherer Routing Protokolle soll das *Secure Routing Protocoll* SRP vorgestellt werden [PH02b, PH02c, PH02d, PHS02, PH03].

Dieses recht einfache Protokoll geht davon aus, dass zwischen Sender S und Empfänger D eine initiale *Security Association* besteht und insbesondere ein gemeinsamer geheimer Schlüssel K_{SD} vorhanden ist. Das Protokoll selbst orientiert sich stark an DSR. Der Sender schickt einen Route Request, den er mittels eines MAC unter Benutzung des gemeinsamen Schlüssels gegen Veränderungen absichert. Der Route Request wird im Netz geflutet, wobei im Paket eine Source Route aufgebaut wird. Zwischenknoten nehmen sonst keine weiteren Veränderungen am Paket vor. Erreicht ein RREQ Paket schließlich D, so prüft dieser zunächst den MAC Code und schickt dann ein Route Reply Paket zurück, welches die Source Route enthält und ebenfalls mittels eines MAC geschützt wird. Durch den Einsatz einer Sequenznummer und eines zufälligen *Query Identifiers* schützt sich das System vor Replays bzw. Routingschleifen.

Weitere Teile der Arbeiten beschäftigen sich mit der Behandlung von Route Errors und gecachten Routen, die teilweise unterstützt werden[4]. Die Autoren diskutieren außerdem verschiedene Angriffsszenarien und zeigen, dass bereits dieses einfache Protokoll eine Vielzahl von möglichen Attacken verhindern kann.

Bewertung

SRP ist vermutlich eine der schlankesten Lösungen zum Thema sicheres Routing in MANETs. Es kommen ausschließlich MACs zum Einsatz und diese werden auch nur in den Endknoten berechnet. Auch der Overhead in den Routingpaketen ist minimal. Damit erreicht SRP eine sehr gute Performance und kommt fast an ein ungesichertes DSR heran [PH02a].

Diese hohe Geschwindigkeit erkauft sich SRP mit einer Reihe von Nachteilen. Die Zwischenknoten werden in keinster Weise authentisiert und auch eine beliebige Verlängerung oder Umleitung der Route ist bei SRP möglich. Damit ist insbesondere Blackhole Angriffen Tür und Tor geöffnet. Auch kann ein Knoten die Identität eines anderen Knotens in die Route einbauen. Die Autoren belegen zwar, dass dies im Rahmen der SRP Operationen keine Probleme bereitet. Integriert man SRP aber in ein Sicherheitsframework, welches z.B. über ein Intrusion Detection System verfügt, entstehen doch gewaltige Probleme.

Viele mögliche Probleme und Fragestellungen zur Authentisierung werden insbesondere durch die Forderung nach einem gemeinsamen geheimen Schlüssel zwischen den

[4] dann müssen aber zusätzlich Gruppenschlüssel vorhanden sein

Endknoten verdeckt. Wie dieser (ohne bestehende Routen) ausgetauscht werden soll, bleibt offen.

10.2. SDSR

Nachdem nun verschiedene sichere Routingprotokolle vorgestellt wurden, wird im Folgenden ein eigener Vorschlag mit Namen *Secure Dynamic Source Routing* oder kurz *SDSR* entwickelt. Wie der Name schon andeutet, orientiert sich auch SDSR an DSR, erweitert dieses aber um mächtige Sicherheitsfunktionen.

Die Entscheidung über die Grundlage für ein sicheres Routingprotokoll fiel zu Gunsten von DSR aus, da das Konzept der Source Route einige Vorteile für die Erkennung von egoistischem Verhalten bietet. Wir werden im nächsten Kapitel sehen, dass einige Sensoren von MobIDS von dieser Möglichkeit Gebrauch machen.

Auch für sicheres Routing bietet die Kenntnis der genauen Route Vorteile, da dann sämtliche Zwischenknoten authentifiziert werden können. Dies wäre bei einem Distance-Vector Protokoll nicht ohne Weiteres der Fall. Umgekehrt würde ein Link-State Protokoll, bei dem die komplette Topologie verteilt wird, noch weitergehende Möglichkeiten bieten. Da der Overhead einer kompletten Topologie-Verteilung aber nicht zu unterschätzen ist, fiel die Wahl auf DSR.

Schließlich bietet DSR den Vorteil, dass die Zwischenknoten nach Weiterleitung eines Route Requests im Gegensatz zu anderen Protokollen wie AODV keinen Zustand speichern müssen. Diese Eigenschaft von DSR soll im Rahmen von SDSR erhalten bleiben.

10.2.1. Aufgaben und Einschränkungen

Wie schon in Kapitel 7 geschildert, hat SDSR einige klar umrissene Aufgaben. Diese werden nun noch weiter detailliert.

Authentifizierung der an Route beteiligten Knoten: Nach erfolgreicher Route Discovery soll für den Initiator sicher feststehen, dass alle Knoten entlang der Route auch tatsächlich authentisch sind. Ein Spoofing eines anderen Knotens oder die Verwendung zufälliger IDs soll nicht möglich sein.

Sicherung der Integrität der Route: Manipulationen an den Routing Paketen sollen verhindert werden. Dazu sind folgende Absicherungen notwendig:

- Verhindern von Änderungen der Source Route oder sonstiger Daten im Route Request.
- Verhindern von Änderungen der Source Route oder sonstiger Daten im Route Reply.

Sicherung der Aktualität der Route: Es soll auch kein Replay älterer und damit möglicherweise veralteter Routeninformation möglich sein. Dieses Ziel kollidiert mit Optimierungsstrategien wie dem *Route Caching*, also dem Generieren von Route Replies durch Zwischenknoten. SDSR beinhaltet dafür andere Optimierungen, welche nicht mit der Sicherheit des Netzes kollidieren.

Austausch geheimer Schlüssel: Im Rahmen der Routensuche soll zwischen Quelle und Ziel ein geheimer Sitzungsschlüssel vereinbart werden, der danach zur Verschlüsselung des Datenverkehrs genutzt wird. Weiterhin benötigen sowohl Quelle als auch Ziel je einen geheimen Schlüssel mit jedem der Zwischenknoten. Diese Schlüssel werden im Rahmen von MobIDS z.B. vom Probing Sensor genutzt, können aber auch zur regulären Kommunikation mit einem Zwischenknoten verwendet werden. Zur Erstellung der Schlüssel kommt bei SDSR ein abgewandeltes Diffie-Hellmann Verfahren zum Einsatz (siehe auch Abschnitt 3.4.1).

Zusätzlich gibt es eine Reihe von Einschränkungen und Randbedingungen:

Keine zentralen Komponenten: wie üblich soll das System komplett ohne zentrale oder besonders hervorgehobene Knoten auskommen. Vielmehr ist ein komplett dezentraler Betrieb notwendig.

Keine/wenige aufwändige Krypto-Operationen: Public Key Operationen sollen soweit möglich vermieden werden. Dies gilt insbesondere für die Route Request Phase, da hier alle Knoten belastet werden. Demgegenüber betreffen Operationen während der Route Reply Phase nur einen relativ kleinen Teil der Knoten. SDSR verschiebt daher viele Prüfungen und Berechnungen in die Route Reply Phase.

Zustandslose Operation: Leitet ein Zwischenknoten ein RREQ Paket weiter, so muss er dazu keinen Zustand speichern. Ausgenommen hiervon ist das Caching von Public Keys und Zertifikaten anderer Knoten.

Keine speziellen Annahmen: Viele Systeme haben Voraussetzungen, die in der Praxis so nur selten gegeben sind. Dazu zählen bspw. streng synchronisierte Uhren, Tamper-proof Hardware usw. SDSR soll ohne derartige Voraussetzungen auskommen.

Byzantinisches Verhalten der Knoten: Das Verhalten eines FEB Knotens wird nicht eingeschränkt. Er verhält sich komplett byzantinisch, wie von Lamport in [LSP82] beschrieben. FEB-Knoten können also beliebige Routingpakete fälschen, modifizieren oder löschen.

Bidirektionale Links: Zur Vereinfachung gehen wir von bidirektionalen Verbindungen aus, so dass Hin- und Rückroute identisch sind. Dies ist im Rahmen der gängigen Funknetze praktisch immer gegeben.

Im Folgenden wird die Route Discovery bei SDSR vorgestellt. Da dieser Vorgang etwas komplizierter ist, wird zunächst eine einfachere Variante vorgestellt. In der vereinfachten Version von SDSR vereinbart lediglich der Quellknoten S geheime Schlüssel mit D und den Zwischenknoten der Route. Am Ende hat also D noch keine geheimen Schlüssel mit den Zwischenknoten ausgemacht. Später wird dann eine Erweiterung beschrieben, welche auch diese Schlüssel vereinbart.

10.2.2. SDSR Route Discovery

Ausgehend von den obigen Anforderungen und Annahmen wurde das SDSR Protokoll entwickelt. SDSR ist ein *reaktives* Protokoll, welches mit *Source-Routen* arbeitet. Will ein Knoten S ein Datenpaket P an einen anderen Knoten D schicken und besitzt

keine gültige Route zu D, so initiiert er eine sogenannte *Route Discovery*. Die Route Discovery gliedert sich in zwei Phasen. In der ersten Phase wird ein *Route Request* im Netz geflutet. Erreicht ein solches *RREQ Paket* den Zielknoten, so schickt dieser ein *Route Reply (RREP) Paket* über die reverse Source Route zurück.

Route Request Phase

Das Route Request Paket, welches D als Broadcast verschickt, hat einen Aufbau wie in Abbildung 10.2 – RREQ Schritt 1 gezeigt. Das erste Feld ist ein Typbezeichner und kennzeichnet das Paket als RREQ. Dann folgen Absender- und Ziel-ID (S bzw. D) sowie eine Route Request ID, welche der Absender eindeutig vergibt. Hinzu kommt mit $DHPK_S$ ein öffentlicher Diffie-Hellmann Schlüssel, den S zufällig wählt. Dabei werden analog zu Abschnitt 3.4.1 die Werte n und z fest von SAM vorgegeben, so dass S lediglich ein a zufällig wählt und daraus das zugehörige A berechnet, welches dann als $DHPK_S$ fungiert.

Wie durch die rote Umrandung angedeutet ist, signiert S alle diese Felder mit seinem MANET-ID Schlüssel und hängt diese Signatur an den RREQ an. Zusätzlich fügt S noch eine zufällige Nonce N_1 sowie eine Sourceroute mit sich selbst als einzigem Eintrag hinzu. a und N_1 muss sich S bis zum Eintreffen der Route Replies merken.

Jeder Zwischenknoten K_i, der einen solchen RREQ empfängt, prüft zunächst, ob er bereits einen RREQ mit gleichem Absender und gleicher Request ID weitergeleitet hat. In diesem Fall verwirft das Paket.

Ansonsten flutet er den RREQ weiter, wie in Abbildung 10.2 – RREQ Schritt 2 zu sehen ist. Vorher fügt er sich noch der Source-Route hinzu und berechnet eine neue Nonce. Dabei gilt $N_{i+1} = \{N_i\}_{k_i}$. Die neue Nonce N_{i+1} entsteht also, indem die alte Nonce N_i durch ein symmetrisches Verschlüsselungsverfahren wie AES mit dem Schlüssel k_i verschlüsselt wird. Den zufällig gewählten Schlüssel k_i kennt nur K_i und merkt ihn sich für den Route Reply.

Erreicht der RREQ das Ziel D, endet die Route Request Phase. Bevor D einen Route Reply generiert, prüft er die Signatur von S, was beweist, dass S den RREQ selbst geschickt hat. Stimmt die Signatur nicht, verwirft D den RREQ.

Route Reply Phase

Stimmt die Signatur, generiert D einen Route Reply. Zur Begrenzung der Zahl von Antworten kann dabei ein Schwellwert vorgegeben werden, wieviele Replies maximal auf einen Request mit einer bestimmten *RREQID* generiert werden. In unseren späteren Simulationen generieren wir maximal drei Antworten pro Request.

Im Route Reply signiert D mit seiner MANET-ID die Source Route und die Signatur von S (siehe Abbildung 10.2 – RREP Schritt 1). Damit wird erreicht, dass sich die Source Route ab diesem Zeitpunkt nicht mehr ändern kann und dass der Reply eindeutig dem Request zugeordnet wird. Schließlich fügt D noch seinen öffentlichen DH-Schlüssel $DHPK_D$ hinzu. Dieser wird, wie schon $DHPK_S$ zufällig generiert. D ist zu diesem Zeitpunkt schon in der Lage k_{SD} gemäß Diffie-Hellmann zu berechnen.

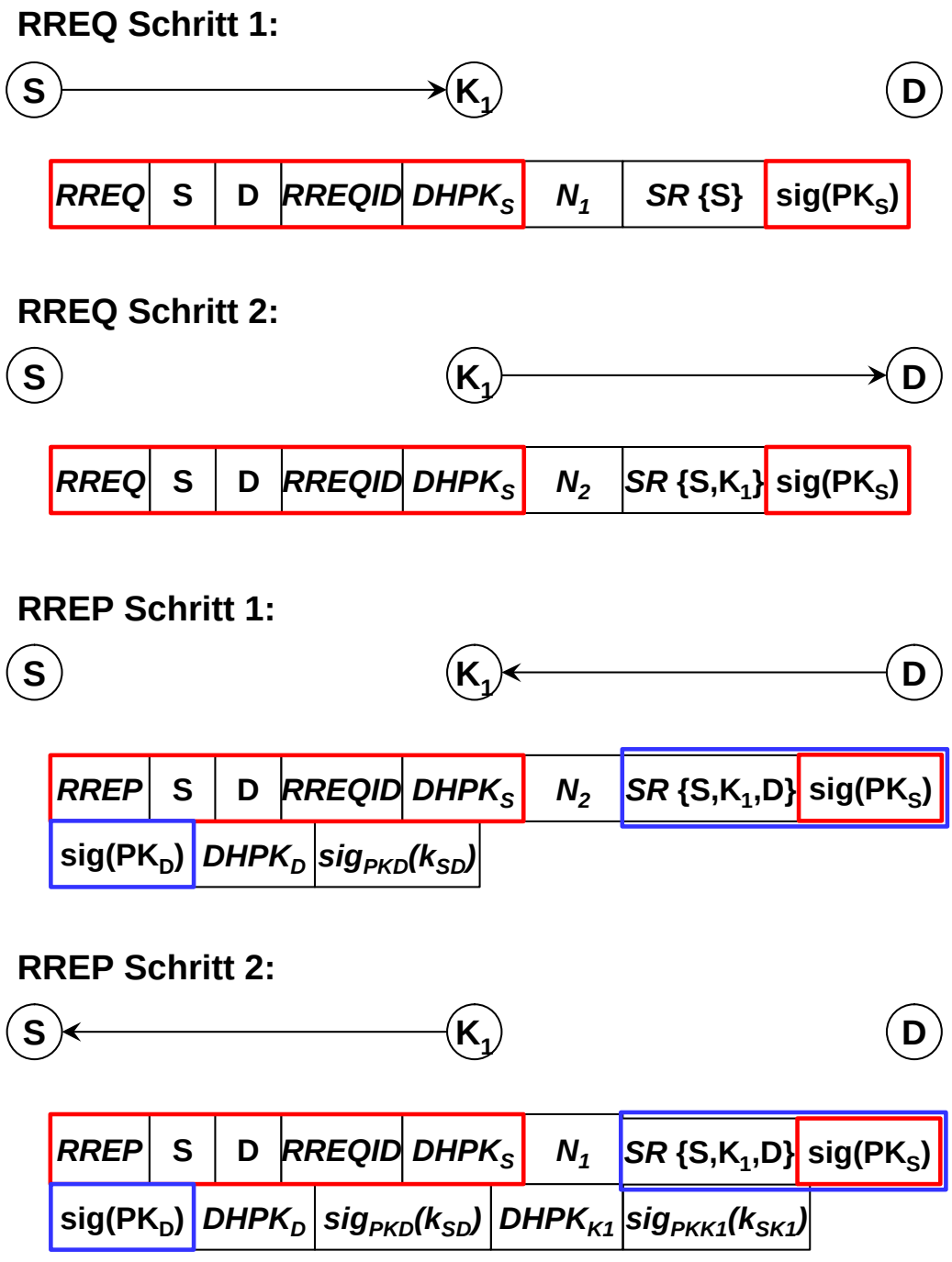

Abbildung 10.2.: Route Discovery bei SDSR

Zu diesem Schlüssel berechnet D nun

$$sig_{PK_D}(k_{SD}) = \{h(k_{SD})\}_{SK_D}$$

Es wird also der Hashwert des gemeinsamen Schlüssels mit dem geheimen RSA Schlüssel von D verschlüsselt. Später kann S anhand von $DHPK_D$ ebenfalls k_{SD} berechnen. Nun kann er die Signatur mit PK_D entschlüsseln und die Hashwerte vergleichen. Stimmen die Werte überein, ist sich S sicher, dass er den gleichen Schlüssel wie D berechnet hat und dass der Wert tatsächlich von D kommt. Die restlichen Felder übernehmen die Werte aus dem RREQ unverändert.

In Abbildung 10.2 – Schritt 2 ist gezeigt, wie der Zwischenknoten K_1 den RREP entlang der Sourceroute weiterleitet. Zunächst prüft K_1 die Signaturen von S und D, um deren Authentizität sicherzustellen. Dann berechnet er ebenfalls einen zufälligen öffentlichen Diffie-Hellmann Schlüssel $DHPK_{K_1}$ und trägt diesen im Paket ein. Außerdem bestimmt er den gemeinsamen geheimen Schlüssel k_{SK_1} und fügt analog D einen signierten Hashwert dieses Schlüssels dem Paket hinzu. Damit kann S später den gemeinsamen Schlüssel berechnen und verifizieren.

Schließlich muss der Zwischenknoten die empfangene Nonce N_{i+1} mit seinem geheimen Schlüssel k_i *entschlüsseln* und somit N_i wieder herstellen. Da nur K_i den Schlüssel k_i kennt, wird damit sichergestellt, dass der Route Reply den gleichen Weg nimmt wie zuvor der Route Request.

Erreicht der RREP schließlich S, so prüft dieser zunächst, ob die Signatur von D stimmt. Damit weiß S, dass D den RREP geschickt hat und dass der RREP sich auf den eigenen RREQ bezieht. Außerdem steht damit fest, dass die Source Route auf dem Rückweg nicht mehr verändert wurde.

Als nächstes prüft S, ob die empfangene Nonce N_1 der abgeschickten Nonce entspricht. Falls ja, steht damit fest, dass Route Request und Route Reply die gleichen Sequenz von Zwischenknoten durchlaufen haben.

Jetzt muss lediglich noch geprüft werden, ob die Sequenz der durchlaufenen Knoten auch der in der Source Route angegebenen Abfolge entspricht. Dies kann S prüfen, indem er gemäß Diffie Hellmann die gemeinsamen geheimen Schlüssel k_{SK_i} und k_{SD} berechnet und dann die Ergebnisse mit den signierten Hashwerten vergleicht. Durch die Signaturen werden die Knoten zuverlässig authentifiziert.

Als Ergebnis der Route Discovery lässt sich festhalten:

- S kennt eine oder mehrere Routen zu D.
- S hat die Authentizität aller anderen Knoten geprüft.
- S ist sich sicher, dass die Source Route unterwegs nicht manipuliert wurde.
- S hat gemeinsame geheime Schlüssel mit jedem K_i und D vereinbart.

Diese Aussagen wurden in obigem Text informell erläutert. Im Analysekapitel wird deren Korrektheit nochmals mittels BAN Logik verifiziert.

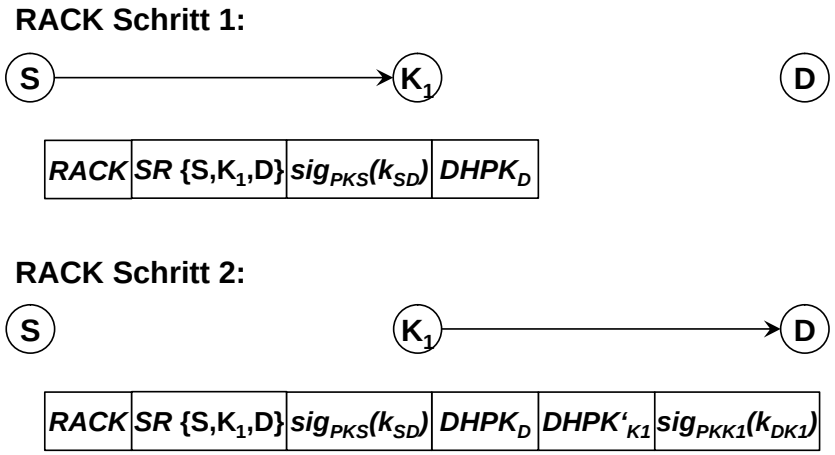

Abbildung 10.3.: Route Acknowledgment bei SDSR

Erweiterte Schlüsselvereinbarung

Die vereinfachte Version des SDSR Protokolls erfüllt zwei Anforderungen noch nicht. Zum einen hat D danach mit den Zwischenknoten K_i keine gemeinsamen geheimen Schlüssel k_{DK_i} vereinbart, zum anderen hat D auch die Zwischenknoten nicht authentifiziert.

Der letzte Punkt ist nicht sehr schwerwiegend, da S schon eine Authentifizierung vornimmt. Schlägt diese fehl, so wird S die Route verwerfen. D wird die Route erst dann als gültig markieren, wenn er über sie ein reguläres Datenpaket von S erhalten hat. Das Problem der fehlenden geheimen Schlüssel betrifft vor allem MobIDS. Ohne diese Schlüssel kann D kein Probing durchführen (siehe Abschnitt 11.5.4). Da S jedoch durch sein Probing die Durchgängigkeit der Route in beide Richtung verifiziert, ist dieser Nachteil nicht gravierend.

Soll D dennoch Schlüssel vereinbaren und die Zwischenknoten authentifizieren, so kommt ein erweitertes Protokoll zum Einsatz. Nach Ablauf von RREQ und RREP schickt S, wie in Abbildung 10.3 gezeigt, nochmals ein sogenanntes *Route Acknowledgment (RACK)* Paket als Unicast an D. Mit diesem Paket wird zweierlei erreicht. Zum einen kann D durch $sig_{PK_S}(k_{SD})$ verifizieren, dass S den Route Reply erhalten hat. Zum anderen können die Zwischenknoten K_i nun mit $DHPK_D$ und einem zufällig gebildeten $DHPK'_{K_i}$ einen weiteren Schlüssel k_{DK_i} berechnen und signiert dem Paket hinzufügen. D berechnet am Ende ebenfalls k_{DK_i} und kann über $sig_{PK_{K_i}}$ die Authentizität von K_i feststellen.

Da D jetzt über gemeinsame geheime Schlüssel mit K_i verfügt und auch die Zwischenknoten authentifiziert hat, verfügen S und D über die gleiche Menge an Informationen, die Route kann also vollkommen symmetrisch genutzt werden.

Schlüsselverteilung

Ein offener Punkt ist die Schlüsselverteilung. Die Adresse eines Knotens K_i wird als CBA aus dem öffentlichen Schlüssel eines Knotens errechnet (siehe Abschnitt 8.3.7). Somit kann jeder Knoten direkt ermitteln, ob eine Adresse zu einem angegebenen öffentlichen Schlüssel gehört. Über die Signatur und den Verifikator kann ferner ermittelt werden, ob es sich um eine gültige MANET-ID handelt.

Unter diesen Voraussetzungen können öffentliche MANET-ID Schlüssel ungesichert im Netz übertragen werden. Solange genügend Speicherplatz vorhanden ist, cachen Knoten die öffentlichen Schlüssel anderer Knoten im Rahmen von deren Gültigkeit. Ist ein neuer Verifikator notwendig, so kann dieser direkt vom Besitzer des Schlüssels per Unicast angefordert werden.

Im Übrigen werden öffentliche Schlüssel im Rahmen der Route Discovery verschickt. Im einfachsten Fall werden alle notwendigen öffentlichen Schlüssel an die RREQ und RREP Pakete angehängt. Der Nachteil dieser Lösung: die Pakete werden unter Umständen sehr groß und Schlüssel werden oft umsonst übertragen, da die Knoten sie möglicherweise schon gespeichert haben. Deshalb wird im folgenden eine optimierte Alternative vorgestellt, welche die übertragene Datenmenge reduziert, dafür aber unter Umständen die Zeit zum Verbindungsaufbau verlängert.

Abbildung 10.4 zeigt ein Beispiel für den Ablauf der Schlüsselverteilung. Zunächst schickt S seinen RREQ wie gewohnt. Er fügt jedoch einen kurzen Bitvektor hinzu, welcher aussagt, ob die Schlüssel von S oder D benötigt werden. Dieser Bitvektor wird *Initialvektor* genannt. Im Beispiel setzt S den Bitvektor auf 00, da er den Schlüssel von D schon kennt. Weiterhin schicken alle Knoten grundsätzlich ihre aktuellen Verifikatoren mit.

Jeder Zwischenknoten kann durch Setzen von Bits öffentliche Schlüssel anfordern. K_1 setzt im Beispiel im RREQ ein Bit, um das Zertifikat von D $cert_D$ anzufordern. Die Zertifikate beinhalten immer auch die Schlüssel selbst. Schließlich setzt K_2 ein Bit um $cert_S$ anzufordern.

Da die spätere Route im RREQ noch nicht bekannt ist, beinhaltet der Initialvektor nur S und D. Die Länge des sogenannten *Upstreamvektors* im RREP entspricht der Länge der Route $l = |SR_{SD}|$. Hier setzt wiederum jeder Knoten die Bits der Schlüssel, die er noch nicht kennt, allerdings nur für Schlüssel, welche „*upstream*“, d.h. in Richtung Quelle S liegen. Die Schlüssel werden dann im RACK zu den Knoten geliefert. Im Beispiel hat der Upstreamvektor am Ende den Wert 1010, d.h. die Schlüssel von S und K_2 werden angefordert. Da das Bit für D im Initialvektor gesetzt war, fügt D auch sein Zertifikat $cert_D$ zum RREP hinzu.

In einem nächsten Schritt wird der Upstreamvektor im RACK Paket dazu benutzt, Zertifikate „*downstream*“ in Richtung Ziel D zu transportieren. Hierzu prüft jeder Knoten K_i, ob seine Position im Vektor gesetzt ist. Wenn ja, fügt er sein $cert_{K_i}$ zum Paket hinzu. Im Beispiel schicken also S und K_2 ihre Zertifikate.

Ebenfalls im RACK Paket wird der *Downstreamvektor* aufgebaut. In diesem setzen alle Knoten die Bits der Zertifikate, welche sie von Knoten „*downstream*“, also in Richtung D, benötigen. Im Beispiel ist der Downstreamvektor 0100, weshalb das Zertifikat von

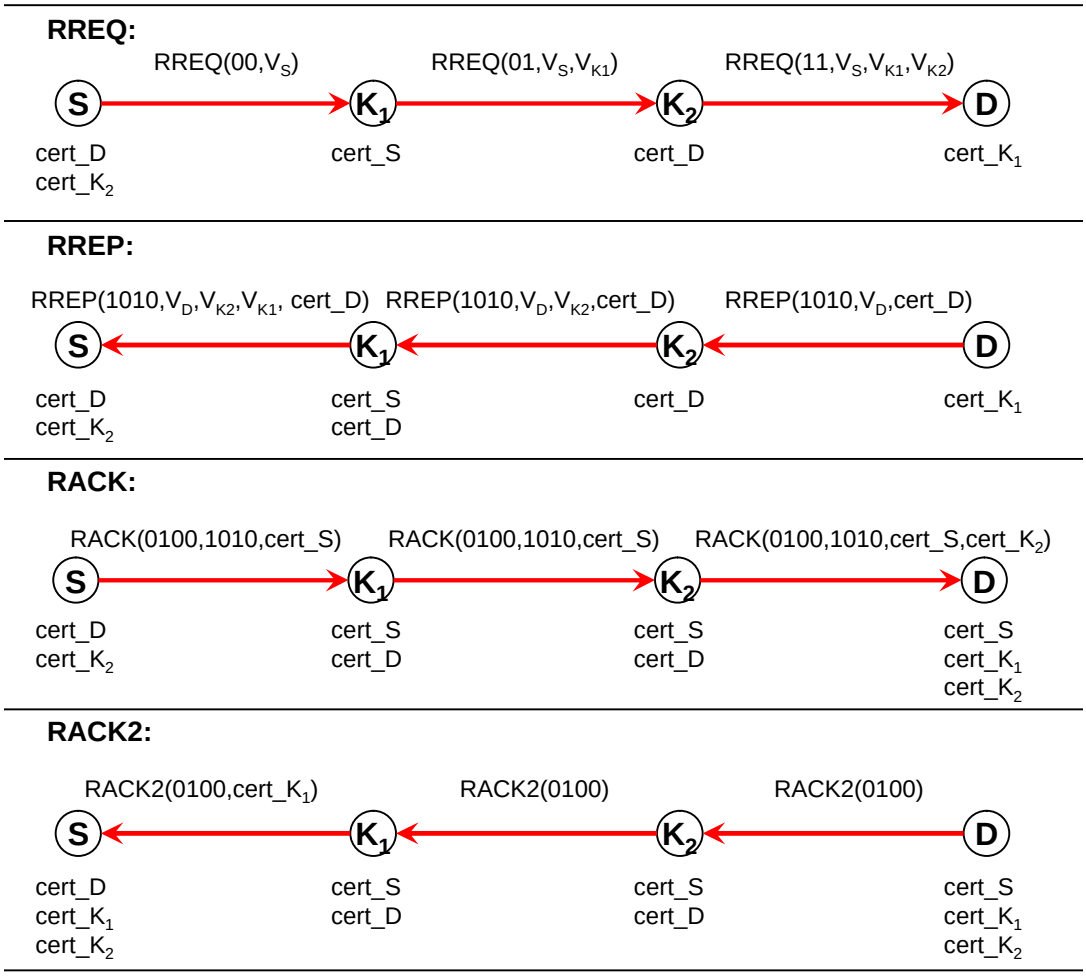

Abbildung 10.4.: Schlüsselverteilung bei SDSR

$K1$ im abschließenden RACK2 Paket geliefert wird. Natürlich können Knoten jederzeit Zertifikate aus den Paketen cachen, auch wenn sie diese nicht angefordert haben.

Fehlt einem Knoten ein öffentlicher Schlüssel, so wird die Verifikation der entsprechenden Signaturen solange hinausgezögert, bis der Schlüssel vorliegt. In dieser Zeit ist die Source Route als *temporär* zu kennzeichnen und darf nur zum Abschluss der Routenfindung und des Schlüsselaustauschs, nicht jedoch für eigentlichen Datenverkehr genutzt werden. Ist der Downstreamvektor bei D komplett 0, so fehlen keine weiteren Schlüssel mehr und RACK2 kann entfallen.

Wie man sieht, sind für den optimierten Schlüsselaustausch die Pakete RACK und RACK2 notwendig, was den Aufbau der Route verzögert. So ist es letztlich eine Abwägung zwischen Geschwindigkeit und Bandbreite, ob man die optimierte Schlüsselverteilung verwendet oder grundsätzlich alle Zertifikate an das Paket anhängt.

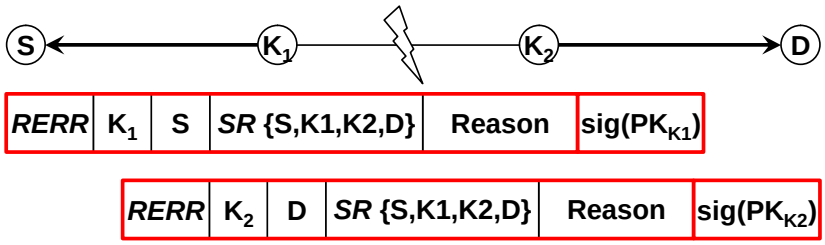

Abbildung 10.5.: Route Error bei SDSR

10.2.3. SDSR Route Maintenance

Kommt es bei einer etablierten Route zu Fehlern, so wird dies, wie bei allen gängigen Protokollen, durch ein *Route Error* Paket gemeldet. Der Aufbau des Paketes ist in Abbildung 10.5 gezeigt. Der meldende Knoten schickt ein signiertes Paket, welches die fehlerhafte Route sowie einen Grund für den Fehler anzeigt. Neben den bei bisherigen Protokollen üblichen Fehlern kann bei SDSR auch eine fehlerhafte Signatur zum Auslösen eines RERR führen.

Verschickt ein Knoten unmotiviert größere Mengen von RERRs, so kann dies ein MobIDS Sensor erkennen und den Knoten gegebenenfalls vom Netz ausschließen.

Empfängt ein Knoten einen Route Error und benötigt er die Route noch für weitere Kommunikation, so muss er eine neue Route Discovery initiieren. Da bereits gemeinsame, geheime Schlüssel vereinbart wurden, kann unter Umständen eine optimierte Version zum Einsatz kommen.

10.2.4. Details und Optimierungen

Neben den beschriebenen Mechanismen gehören noch eine Vielzahl von Details zu einer funktionierenden Implementierung von SDSR, wie wir sie im Rahmen der ns2 Analyse entwickelt haben. So muss beispielsweise die Weiterleitung von Route Requests zufällig verzögert werden, um zu verhindern, dass es zu Resonanzphänomenen kommt und während einer Route Discovery kein normaler Verkehr mehr transportiert werden kann. Diese Details stehen dabei nicht in Bezug zur Sicherheitsfunktion von SDSR und sind überdies für DSR bereits hinreichend gut untersucht und im DSR Draft dokumentiert [JMHJ03]. Auf sie wird deshalb an dieser Stelle nicht weiter eingegangen.

Der DSR Draft beschreibt ebenfalls eine Reihe von möglichen Optimierungen. Das Route Caching zielt darauf ab, dass Zwischenknoten einen Route Request mit ihnen bekannten Informationen direkt beantworten, ohne den RREQ bis zum Zielknoten weiterzuleiten. Bei einer anderen Optimierung arbeiten die Knoten im Promiscuous Modus und bauen die Routen, welche sie in zufällig aufgefangenen Paketen mithören, in ihre Topologiedatenbank ein.

Beide Optimierungen sind in SDSR nicht möglich, da dabei die Authentizität der Knoten nicht mehr sicher gestellt werden kann und auch die Vereinbarung von gemeinsamen geheimen Schlüsseln wird damit gestört. Um diesen Nachteil auszugleichen, verfügt SDSR über eine Reihe anderer Optimierungsmöglichkeiten, welche unnötigen

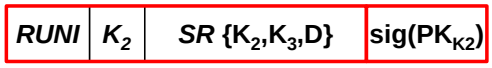

Abbildung 10.6.: RREQ Unicast Option bei SDSR

Verkehr während der Route Discovery vermeiden können. Diese werden im Folgenden kurz umrissen.

Piggybacking

Unter Piggybacking ist zu verstehen, dass Knoten Daten des Routingprotokolls und normale Daten gemeinsam verschicken können und somit Pakete eingespart werden. Generell werden alle SDSR Daten als IPv6-Optionen im Paketheader kodiert (siehe [DH98, Hui00]). Somit steht es einem Knoten frei, sonstige Informationen im Datenteil mitzuschicken.

Im RREQ Paket ist dies nicht sinnvoll, weil diese Daten dann im gesamten Netz geflutet werden müssen. Lediglich ein TCP-SYN Paket zum Aufbau einer TCP Verbindung wäre hier sinnvoll. Bei späteren Paketen (RREQ, RACK, RACK2) können jedoch sinnvoll reguläre Daten eingebaut werden. Dies setzt allerdings voraus, dass S und D bereits über einen gemeinsamen geheimen Schlüssel verfügen. Wenn dies nicht schon vorher gegeben ist, so können reguläre Daten also frühestens im RACK Paket eingebaut werden.

Route Request Unicasting

Empfängt ein Zwischenknoten K_i einen Route Request vom Knoten S und kennt er bereits eine gültige Route zum Ziel D, so ist es nicht mehr notwendig, dass der Route Request per Broadcast durch das Netz läuft. Vielmehr kann K_i eine Option zum Paket hinzufügen, welche dafür sorgt, dass der RREQ den restlichen Weg als Unicast durchläuft. Die sonstigen Berechnungen zur Vereinbarung von Schlüsseln und Authentifizierung der Knoten laufen aber wie bisher ab. Abbildung 10.6 zeigt den Aufbau dieser Option.

Der Knoten K_2 zeigt damit an, dass er eine Route via K_3 nach D kennt und diese für die Weiterleitung des RREQs verwendet werden soll. Diese Optimierung hat jedoch ein Problem: K_2 kann bewusst eine suboptimale oder gar falsche Route angeben. Man könnte verlangen, dass K_2 die Signatur der Source Route, welche er im Route Reply erhalten hat, in der Option mitliefert. Damit wird dann zumindest verhindert, dass der Zwischenknoten beliebige Phantasierouten ausliefert. Auch sollte man für die signierte Route eine gewisse Haltbarkeit definieren, damit keine veralteten Informationen verwendet werden können. Schließlich wird man darauf hoffen, dass bei verlängerten Routen andere RREQs auf alternativen Wegen das Ziel erreichen und die problematische Route somit nicht zum Tragen kommt.

Abbildung 10.7.: Reuse Secret Key Request und Reuse Secret Key Answer Optionen bei SDSR

Generell ist es in jedem Fall problematisch, wenn Zwischenknoten basierend auf partiellen und teilweise veralteten Informationen das Fluten des Route Requests umgehen. Der Einsatz derartiger Optimierungen sollte also in jedem Fall gut überlegt werden.

Schlüssel-Wiederverwendung

Hat ein Knoten S bereits einmal einen geheimen Schlüssel k_{SD} mit einem anderen Knoten D vereinbart, so ist es nicht unbedingt notwendig, dass er bei einer neuen Route Discovery einen neuen Schlüssel vereinbart. Auch die Authentifizierung lässt sich mittels k_{SD} erledigen. Hierzu schickt S eine verschlüsselte Nonce $\{N\}_{k_{SD}}$ an D. Dieser entschlüsselt N und schickt dann $\{N-1\}_{k_{SD}}$ an S zurück.

Hierzu setzt ein Knoten K_i in einem entsprechenden Paket sein $DHPK_{K_i}$ und die zugehörige Key Signatur auf 0 und fügt stattdessen eine *Reuse Secret Key Request* Option wie in Abbildung 10.7 ins Paket ein. Hat der Zielknoten den entsprechenden Schlüssel noch gespeichert, antwortet er mit einer *Reuse Secret Key Answer* Option. Danach kann der entsprechende Schlüssel einfach weiter genutzt werden.

10.2.5. Bewertung

Damit ist SDSR beschrieben. Es erfüllt alle in Abschnitt 10.2.1 gestellten Anforderungen, wie insbesondere die Analyse in Abschnitt 12.4 zeigen wird. Alle an der Route beteiligten Knoten werden zuverlässig authentifiziert. Änderungen an der Route sind unmöglich bzw. werden erkannt dies gilt auch für Verlängerungen. Auch wird durch die Route Request ID sichergestellt, dass immer nur aktuelle Routen gefunden werden. Dies geht allerdings zu Lasten von Optimierungen, die bei SDSR nur eingeschränkt möglich sind. Quell- und Zielknoten einer Route Discovery haben nach deren Ablauf geheime Schlüssel miteinander und mit den Zwischenknoten vereinbart, die im weiteren zur Absicherung der Verbindung und von MobIDS verwendet werden können. Das Routing Protokoll verwendet (mit Ausnahme der MANET-CA) keine zentralen Komponenten. Aufwändigere Kryptooperationen, insbesondere asymmetrische Kryptographie, sind in der Zahl beschränkt und in die Route Reply Phase verlagert, wo sie weniger Knoten belasten. Die Zwischenknoten müssen praktisch keinen Zustand während einer Route Discovery speichern; lediglich die eigenen geheimen Schlüssel und Zertifikate sind dauerhaft aufzubewahren. Spezielle Annahmen zu synchronisierten Uhren o.Ä. trifft SDSR nicht. Die Schadwirkung von FEB Knoten beschränkt sich darauf, dass ein solcher Knoten Pakete nicht weiterleiten oder falsch signieren kann. Jede Form von Manipulation wird im Laufe der Route Discovery erkannt und die entsprechende Route wird verworfen. Stehen keine alternativen Routen zur Verfügung,

Funktion	SAODV	Ariadne	ARAN	SRP	SDSR
Schlüsselverteilung	vorausgesetzt	vorausgesetzt	integriert	vorausgesetzt	integriert
Authentifizierung der Knoten	nur Endknoten	alle	alle	nur Endknoten	alle
Sicherung Route im Request	ja (Verl. möglich)	ja	ja (Verl. möglich)	nein	ja
Sicherung Route im Reply	ja	ja	ja	ja	ja
Sicherung Aktualität Route	ja	ja	ja	ja	ja
Austausch geheimer Schlüssel	nein	nein	nein	nein	ja
Verwendung gecachter Routen	ja	nein	nein	nein	nein
Performance	$\oplus$	$\bigcirc$	$\ominus$	$\oplus\oplus$	$\bigcirc$
weitere Voraussetzungen	keine	synch. Uhren	synch. Uhren	keine	keine

Tabelle 10.1.: Vergleich verschiedener sicherer MANET Routing Protokolle

so führt dies im schlimmsten Fall zu einem DoS, da dann nicht kommuniziert werden kann.

Tabelle 10.1 vergleicht nochmals die vorgestellten Routing Protokolle hinsichtlich ihrer Eigenschaften. Wie man sieht liefert SDSR den höchsten Schutz von allen. Dafür fallen einige Optimierungsmöglichkeiten weg. Verglichen mit Protokollen wie SRP ist auch ein etwas erhöhter (Rechen-)Aufwand notwendig. Zwar fallen auch die Nachrichten etwas länger aus als bei der Konkurrenz, dies ist aber vor allem auf die integrierte Verteilung von Schlüsseln zurückzuführen, was die anderen Protokolle implizit voraussetzen.

10.3. Fazit

Mit SDSR wurde ein sicheres Routingprotokoll für MANETs vorgestellt, was den Konkurrenten wie SAODV, Ariadne, ARAN oder SRP je nach Zielsetzung mindestens ebenbürtig, wenn nicht überlegen ist. Insbesondere schützt SDSR vor einer größeren Bandbreite von Fehlverhalten, als die anderen Systeme. Durch die Integration ins SAM Framework werden auch Fragen zu Schlüsselverwaltung, Erzeugung von Identitäten oder Bestrafung und Ausschluss von FEB Knoten beantwortet.

Was SDSR noch nicht bemerkt sind Knoten, die am Routing überhaupt nicht teilnehmen, also einen Route Request z.B. nicht weiterleiten. Um diese letzte Art von Fehlverhalten abzudecken, verfügt SAM über ein spezielles Intrusion Detection System, welches im folgenden Kapitel vorgestellt wird.

11. Mobile Intrusion Detection System - MobIDS

11.1. Grundlagen der Intrusion Detection

Wie bereits in Abschnitt 7.2 erläutert, besteht die Aufgabe des MobIDS Systems primär darin, ein Fehlverhalten, das sich nicht im Rahmen von SDSR sicher verhindern lässt, zu erkennen und den FEB Knoten gegebenenfalls aus dem MANET auszuschließen.

In diesem Abschnitt werden zunächst einige Grundlagen von Intrusion Detection Systemen (IDS) vorgestellt. Ursprünglich wurden IDS für den Einsatz auf Mainframes entwickelt. Hier dienten sie beispielsweise AT&T dazu, Unregelmäßigkeiten in Telefonabrechnungen aufzudecken, die auf den Einsatz von sog. Blueboxing zurückzuführen waren [And80]. Ein weiteres frühes System war IDES von Denning und Neumann [DN85]. Neben dem kommerziellen Sektor kamen diese frühen Systeme vor allem im militärischen Bereich zum Einsatz. Zu einer größeren Verbreitung kam es allerdings erst mit der Ausbreitung des Internet.

In Erweiterung der Definition aus Abschnitt 2.2.1 ist zunächst zu klären, was aus Sicht eines IDS unter einer „*Intrusion*“ zu verstehen ist. Anderson definiert eine Intrusion als [And80]:

> „The potential possibility of a deliberate unauthorized attempt to
>
> - access information,
> - manipulate information, or
> - render a system unreliable or unusable.“

Eine ähnliche Definition stammt von Heberlein et al. aus dem Jahr 1991 [HML91]:

> „. . . eine Menge von Handlungen, deren Ziel es ist die Integrität, die Verfügbarkeit oder die Vertraulichkeit eines Betriebsmittels zu kompromittieren.“

Das sogenannte *Common Intrusion Detection Framework (CIDF)* [SCTS98, Tun99] unterteilt den Aufbau eines IDS in folgende Komponenten (siehe Abbildung 11.1):

Event Box: registriert verschiedene Ereignisse aus dem Netzwerk, vom Betriebssystem oder von Applikationen[1] und bringt diese in ein einheitliches Format, welches die A-Box verarbeiten kann.

Analysis Box: analysiert und bewertet die von der E-Box gelieferten Ereignisse.

[1] Dies können z.B. Netzwerkpakete, Log-Ausgaben von Serverprozessen, Kernelmeldungen uvm. sein, die an verschiedenen Stellen im System gewonnen werden. Diese Informationen werden auch *Audit Daten* genannt

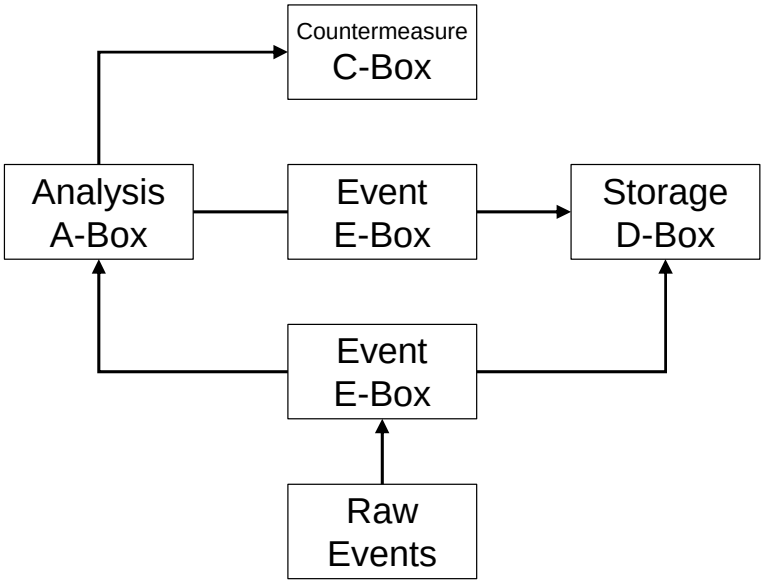

Abbildung 11.1.: Common Intrusion Detection Framework

(Data-)Storage Box: speichert die Ausgaben der A-Box und gegebenenfalls auch direkt die Ereignisse der E-Box zur späteren Verwendung. So kann die A-Box mehrere Ereignisse korrelieren, um komplexere Angriffe zu erkennen. Auch kann ein Administrator den Angriff zu einem späteren Zeitpunkt manuell nachvollziehen.

Countermeasure Box: führt bei einem erkannten Angriff entsprechende Gegenmaßnahmen durch. Dies besteht im einfachsten Fall aus einer Protokollierung und Benachrichtigung eines Administrators. Weiterhin kann die C-Box beispielsweise versuchen, zusätzliche Informationen über den Angreifer zu erlangen, diesen aus dem eigenen Netz auszusperren oder sogar einen Gegenangriff zu starten. Letzteres wird von Sicherheitsfachleuten in der Regel abgelehnt.

Gemäß obiger Aufteilung unterscheidet man auch zwischen dem eigentlichen *Intrusion Detection System (IDS)*, welches lediglich für das Erkennen eines Angriffs zuständig ist, und dem *Intrusion Response System (IRS)*, welches daraus entsprechende Maßnahmen ableitet. Bei der Erkennung gibt es *hostbasierte* und *netzwerkbasierte* IDS, je nachdem, ob die Erkennung nur aus Sicht eines einzelnen Knotens oder netzwerkweit erfolgt.

Für die eigentliche Analyse wurden verschiedene Ansätze entwickelt. Die *Signaturerkennung* arbeitet ähnlich einem Virenscanner. Dem IDS sind typische Muster von bestimmten Angriffen vorher bekannt[2]. Entdeckt das IDS ein solches Muster, wird dies als Angriff gewertet. Komplexere Angriffssignaturen, die bspw. aus mehreren Schritten bestehen, können durch Zustandsautomaten oder Expertensysteme modelliert werden.

Ein anderer Ansatz ist die *Anomalieerkennung*. Hier wird davon ausgegangen, dass ein regulärer Nutzer eines Systems ein statistisch erfassbares und regelmäßiges Verhalten zeigt. Somit ergibt sich ein typisches Nutzerprofil, welches bspw. Nutzungszeiten, Häufigkeit der Nutzung oder Ressourcenverbrauch beinhalten kann. Eine signifikan-

[2] z.B. eine bestimmte Bytefolge, die in Netzwerkpaketen beim Angriff auf einen Webserver auftritt

te Abweichung von diesem Profil wird als Angriff gewertet. Das Nutzungsprofil kann entweder fest programmiert sein oder über vorgegebene Parameter eingestellt werden. Noch flexibler sind selbstlernende Systeme. Diese erfassen während einer Lernphase das normale Verhalten der Benutzer und prüfen in der Betriebsphase, ob die Benutzer sich weiterhin entsprechend dem erlernten Profil verhalten. Ein Beispiel für ein solches System liefert [LX01].

Ein weiterer Ansatz ist die *spezifikationsbasierte Erkennung*, wie sie in [Ko96, KRL97] erstmals erwähnt wurde. Hier wird davon ausgegangen, dass sich das korrekte Verhalten eines Programms in einer formalen Spezifikation ausdrücken lässt. Dies kann bspw. mit einfachen, kontextfreien Grammatiken o.Ä. geschehen. Weicht ein Programm von dieser Spezifikation ab, so wird dies als Einbruch gewertet.

Diese drei Ansätze haben jeweils verschiedene Vor- und Nachteile. Die signaturbasierte Erkennung findet bekannte Angriffe mit sehr hoher Zuverlässigkeit und Präzision und produziert praktisch keine Fehlalarme. Sie ist einfach und effizient zu implementieren. Unbekannte Angriffe werden allerdings nicht erkannt und oft genügen schon kleine Abwandlungen eines bestehenden Angriffs, um ein signaturbasiertes IDS zu überlisten.

Da bei der spezifikationsbasierten Erkennung keine Angriffe, sondern vielmehr das korrekte Verhalten eines Programms erkannt wird, können auch unbekannte Angriffe erkannt werden. Hauptnachteil ist hier, dass es oftmals sehr schwierig ist, ein korrektes Verhalten formal anzugeben.

Auch die Anomalieerkennung basiert auf dem Prinzip eines korrekten oder regulären Zustandes und kann somit unbekannte Angriffe erkennen. Hierzu muss aber zunächst der Normalzustand spezifiziert werden, was mit ähnlichen Problemen wie beim spezifikationsbasierten Ansatz einher geht. Verwendet man ein selbstlernendes System so muss sichergestellt sein, dass die Trainingsdaten auch wirklich ein typisches Verhalten des Systems widerspiegeln.

Kommt es zu einer Abweichung von der Spezifikation bzw. vom Normalzustand, so ist es bei spezifikationsbasierter und vor allem bei der Anomalieerkennung meist sehr schwer, auf den Angriff bzw. den Angreifer zurückzuschließen. Auch müssen derartige Systeme so eingestellt werden, dass möglichst wenige falsche Angriffe erkannt werden, ohne dass dabei tatsächliche Angriffe übersehen werden. Aus diesen Gründen sind heute ähnlich wie bei den Virenscannern hauptsächlich signaturbasierte Systeme im produktiven Einsatz. Die andere Ansätze sind hauptsächlich im Forschungsbereich zu finden.

11.2. IDS für MANETs

Will man Intrusion Detection Systeme in MANETs einsetzen, sieht man sich zunächst mit einigen speziellen Problemen konfrontiert.

- In einem klassischen, netzbasierten IDS wählt der Administrator, welchem die Verantwortung über die Sicherheit des Netzes obliegt, eine kleine Anzahl von vertrauenswürdigen und besonders geschützten Knoten aus, welche den Netzverkehr aufzeichnen und analysieren. In einem öffentlichen, subscription-less MANET ist kein zentrales System und auch kein verantwortlicher Administrator verfügbar.

Zentrale Knoten könnten auch schnell zu einem Flaschenhals oder Opfer von gezielten Attacken werden. Bedingt durch die dynamische und stark vermaschte Netzwerktopologie sieht ein zentraler IDS Knoten auch immer nur einen kleinen Bruchteil des gesamten Netzwerkverkehrs.

- In einem MANET ist es sehr einfach, Datenpakete abzuhören oder neue Pakete ins Netz einzuschleusen. Auch sind Kommunikationsvorgänge sehr störanfällig. Entsprechend kann nie sichergestellt werden, dass das IDS alle Daten korrekt empfängt.

- Die Knoten verfügen nur über begrenzte Ressourcen. Wären nur wenige Knoten für das IDS zuständig, so würden diese durch die Zahl der zu empfangenden und zu analysierenden Pakete vermutlich schnell überlastet.

- Im Gegensatz zu bezahlten Administratoren in einem Firmennetz sind die Benutzer von MANET Knoten in der Regel keine Sicherheitsspezialisten. Entsprechend wird es ihnen schwer fallen, die Meldungen des IDS über bestimmte Angriffe korrekt zu bewerten und sinnvolle Gegenmaßnahmen einzuleiten.

- Neben klassischen Angriffen muss ein IDS für MANETs auch egoistisches Verhalten erkennen. Die Schwierigkeit ist hier, dass ein egoistischer Knoten meist schlicht nichts tut, d.h. beispielsweise keine Pakete weiterleitet. Dies zu erkennen ist wesentlich schwieriger, als eine Signatur eines Angriffs in einem Datenpaket zu finden.

Berücksichtigt man diese Probleme und die Aufgaben von MobIDS, wie sie in Abschnitt 7.2 dargestellt wurden, so lassen sich daraus einige konkrete Anforderungen für MobIDS ableiten:

- Primäre Aufgabe von MobIDS ist die *Erkennung von egoistischen und böswilligen Knoten*. Durch den drohenden Ausschluss werden diese *zur Kooperation motiviert*.

- Zur Vermeidung der dargestellten Probleme, ist MobIDS komplett dezentral aufzubauen.

- Es kann nicht vom Vorhandensein von initialen Vertrauensbeziehungen ausgegangen werden, da sich die Teilnehmer vorher nicht kennen.

- Die vorhandenen Ressourcen sind möglichst sparsam zu nutzen.

- Kennt ein Angreifer die Methoden, mit denen MobIDS nach FEB Knoten sucht, so wird er unter Umständen Wege finde, einer Erkennung zu entgehen. Somit sollte MobIDS modular aufgebaut sein, um jederzeit eine Erkennung für neue Angriffsmethoden hinzufügen zu können.

- Der Benutzer eines MANET Gerätes sollte möglichst nicht mit MobIDS in Kontakt kommen. Das System sollte vollautomatisch und ohne Benutzereingriff arbeiten können.

11.3. Verwandte Arbeiten

Bevor im Anschluss Aufbau und Funktionsweise von MobIDS beschrieben werden, sollen nun zunächst andere Arbeiten mit ähnlicher Zielsetzung vorgestellt werden. Dabei wird auch analysiert, inwieweit diese den gestellten Anforderungen gerecht werden.

11.3.1. Watchdog und Pathrater

Die erste Arbeit zur Intrusion Erkennung in Ad hoc Netzen publizierten Sergio Marti, T. J. Giuli, Kevin Lai und Mary Baker im Jahr 2000 unter dem Titel „Mitigating routing misbehavior in mobile ad hoc networks“ [MGLB00]. Darin beschreiben sie ein auf dem DSR Protokoll beruhendes System, welches aus den beiden Komponenten *Watchdog* und *Pathrater* besteht. Ziel ist es, den Schaden zu begrenzen, den egoistische Knoten im Ad hoc Netz anrichten, wenn sie Pakete nicht weiterleiten.

Schickt ein Knoten A ein Paket an einen Folgeknoten B, der dieses laut Sourceroute an C weiterschicken soll, so prüft A mittels des Watchdogs, ob B diese Weiterleitung tatsächlich durchführt (siehe Abbildung 11.2). Hierzu hört er im *Promiscuous Mode* die von B verschickten Datenpakete ab. Hört er nach einer gewissen Zeitspanne nicht, dass das Paket weitergeleitet wurde, so geht er davon aus, dass B nicht kooperiert. A benachrichtigt nun die Quelle des Pakets von seiner Beobachtung. In der Folge wird die Pathrater Komponente in den Knoten Pfade, welche B beinhalten, abwerten und somit möglichst meiden.

Bewertung

Die Arbeit von Marti et al. zählt zweifellos zu den Grundlagen der MANET-Security und wird entsprechend oft zitiert. Und in der Tat kann das System die Auswirkungen von einfachem egoistischem Verhalten in Ad hoc Netzen begrenzen. Die Autoren haben Simulationen durchgeführt, bei denen von 50 Knoten in einem MANET 20 keine Pakete weiterleiten. Ohne Schutzmaßnahmen erreichen dabei nur 64% der Pakete ihr Ziel. Bei Einsatz des Watchdogs und Pathraters sind dies hingegen 85%.

Das System weist allerdings einige gravierende Schwächen auf. Zunächst lässt sich feststellen, dass egoistisches Verhalten nicht verhindert, sondern im Gegenteil noch gefördert wird. Ein Knoten, der keinen Verkehr weiterleitet, wird auch in Zukunft

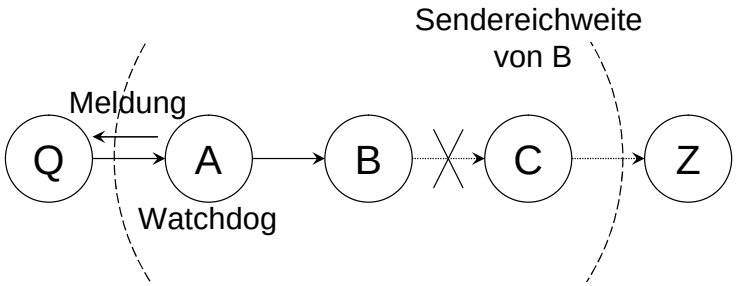

Abbildung 11.2.: Watchdog mit Overhearing

nicht mehr mit Paketen anderer Knoten „belästigt“. Nachteile entstehen ihm dadurch keine.

Weiterhin ist, wie die Autoren selbst feststellen, die Watchdog Erkennung nur sehr bedingt zuverlässig. Es gibt eine Vielzahl von Situationen, bei denen ein Knoten ein Paket korrekt weiterleitet und ein Watchdog in einem vorherigen Knoten dies nicht erkennt. Umgekehrt kann ein Zwischenknoten dem Watchdog relativ leicht vormachen, er hätte ein Paket weitergeleitet, obwohl der Empfänger dieses nie erhalten hat. In Abschnitt 11.5.1 werden diese Probleme genauer untersucht.

11.3.2. CONFIDANT

Sonja Buchegger und Jean-Yves Le Boudec gehen in [BB01, BB02a] genauer auf die Auswirkungen von egoistischem Verhalten in MANETs ein. Sie leiten aus der Biologie ihre Motivation zur Bestrafung von egoistischen oder böswilligen Knoten ab. Sie argumentieren, dass ohne Gegenmaßnahmen egoistische Knoten überhand nehmen werden und somit die korrekte Funktionsfähigkeit des Netzes nicht mehr gewährleistet ist.

Daraus leiten sie das Grudger Protokoll und später das CONFIDANT System [BB02b] ab, welches dem gegensteuern soll. Die Grundidee beruht darauf, dass sich Knoten, analog zum Watchdog Ansatz, gegenseitig durch Overhearing von Nachrichten kontrollieren. Damit soll Fehlverhalten wie das Nicht-Weiterleiten oder die Modifikation von Paketen erkannt werden. Knoten kommunizieren ihre Beobachtungen in sogenannten Alarm Nachrichten. Diese werden an befreundete Knoten geschickt, die daraufhin dem gemeldeten Knoten ebenfalls misstrauen.

Im einzelnen besteht das System aus folgenden Komponenten (siehe Abbildung 11.3):

Monitor: erfasst Pakete im Promiscuous Mode und versucht bösartiges oder egoistisches Verhalten zu erkennen. Er erzeugt daraufhin ein Ereignis für das Reputation System.

Reputation System: ordnet den einzelnen Knoten im MANET Ratings zu. Diese basieren auf eigenen Beobachtungen und Informationen aus Alarm Nachrichten von anderen Knoten. Je nach Vertrauenswürdigkeit des Knotens werden diese externen Informationen geringer gewichtet als eigene Beobachtungen. Unterschreitet das Rating eines Knoten eine bestimmte Schwelle wird der Path Manager benachrichtigt.

Path Manager: vermeidet Pfade, die durch böswillige oder egoistische Knoten gehen. Weiterhin wird für solche Knoten die Weiterleitung von Paketen verweigert.

Trust Manager: verschickt und empfängt Alarm Nachrichten mit Ratings über andere Knoten.

Bewertung

Das CONFIDANT System entwickelt die Ideen aus dem vorherigen Abschnitt konsequent weiter. Insbesondere die Verwaltung von Bewertungen über andere Knoten und deren Verteilung sind wesentliche Ergänzungen. Auch werden nun die egoistischen oder böswilligen Knoten insofern bestraft, als andere Knoten deren Nachrichten bei

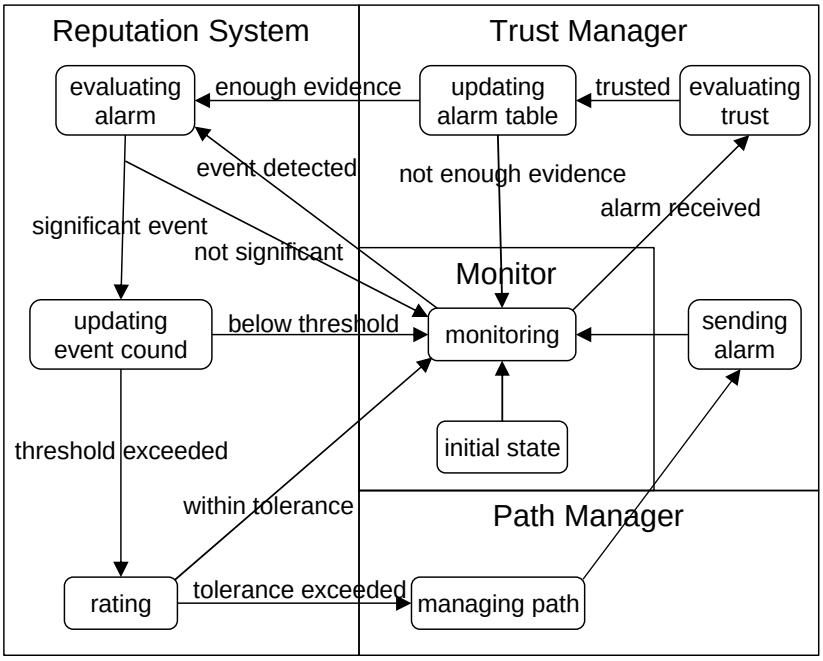

Abbildung 11.3.: Komponenten des CONFIDANT Systems (nach [BB01])

Vorliegen von negativen Bewertungen nicht mehr weiterleiten. In [BB02b] zeigen die Autoren die Effizienz ihres Systems.

Doch auch dieses System hat einige Schwächen. Als Detektor für egoistisches Verhalten dient wiederum nur ein einfaches Overhearing von Nachrichten im Promiscuous Mode. Somit können viele Ereignisse nicht erkannt oder nur schlecht eingeschätzt werden. Auf viele weitere Fragestellungen gehen die Autoren nicht ein. So wird nicht untersucht, inwieweit das System selbst für Angriffe anfällig ist, wenn beispielsweise negative Anschuldigungen gefälscht und verteilt werden. Auch eine Absicherung der Identitäten der Netzteilnehmer ist nicht vorgesehen.

11.3.3. Mobiles Intrusion Detection System nach Zhang/Lee

Ebenfalls modular aufgebaut ist das IDS, welches Wenke Lee, Yongguang Zhang et al. in [ZL00, ZLH03] beschreiben. Die Autoren versuchen darin, ihr für klassische Netze entwickeltes IDS auf MANETs zu übertragen. Wie in Abbildung 11.4 zu sehen, besteht deren System aus insgesamt sechs Komponenten:

Lokale Datensammlung: erfasst lokale Aktivitäten des Benutzers und Kommunikation aus Sicht des Knotens.

Sichere Kommunikation: tauscht Bewertungen mit anderen Knoten aus.

Lokale Erkennung: wertet die Daten der *lokalen Datensammlung* aus. Wird aus diesen mit hoher Wahrscheinlichkeit eine Anomalie erkannt, so erfolgt eine lokale Reaktion, sonst wird die *globale Erkennung* genutzt.

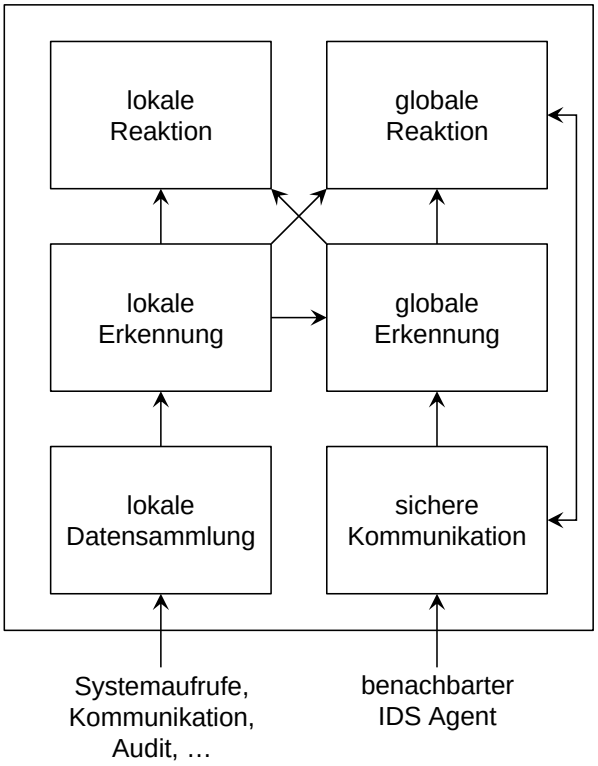

Abbildung 11.4.: Aufbau des IDS von Zhang, Lee et al. (nach [ZL00])

Globale Erkennung: nutzt auch die Daten von anderen vertrauenswürdigen Knoten für die Bewertung eines potentiell bösartigen Knotens.

Lokale Reaktion: kann zum Beispiel dadurch erfolgen, dass der Benutzer informiert wird.

Globale Reaktion: erfolgt gemeinsam durch alle Knoten im Netzwerk. Zum Beispiel könnten die restlichen Knoten neue Schlüssel vereinbaren und den bösartigen Knoten so aus dem Netz ausschließen.

Neben diesem grundsätzlichen Aufbau eines IDS widmen sich Wenke Lee et al. in [ZL00, LX01, ZLH03] der Fragestellung, wie Anomalien in herkömmlichen und in Ad hoc Netzen erkannt werden können. Hierzu setzen sie insbesondere auf Methoden der Informationstheorie. Relevant sind hierbei die *Entropie* und die *relative Entropie*:

Definition 11.1 (Entropie) *Für Daten X, bei denen jedes Element $x \in C_X$ ist, seien C_X die Klasse der Daten und P(x) die Wahrscheinlichkeit des Auftretens von x in X. Dann bezeichnet*

$$H(X) = \sum_{x \in C_X} P(X)\, log \frac{1}{P(x)}$$

den Informationsgehalt *oder die* Entropie von X.

Definition 11.2 (relative Entropie)

$$relEntropie(\, p|q\,) = \sum_{x \in C_X} p(x)\, log \frac{p(x)}{q(x)}$$

Lee et al. übertragen diese Begriffe auf IDSs. Bei einem IDS bezeichnet dabei die Entropie die „Gleichmäßigkeit“ oder „Regelmäßigkeit“ der Audit-Daten. Die relative Entropie vergleicht nun zwei Wahrscheinlichkeitsverteilungen miteinander. Dadurch kann ein Trainingsdatensatz mit den aktuell beobachteten Daten verglichen werden. Je kleiner die relative Entropie, desto ähnlicher sind sich die Datensätze. Definiert man nun noch zusätzlich die relative konditionale Entropie, dann lassen sich auch Sequenzen von Ereignissen vergleichen.

Definition 11.3 (relative konditionale Entropie)

$$relKondEntropie(\,p|q\,) = \sum_{x \in C_X,\, y \in C_Y} p(x|y)\; log \frac{p(x|y)}{q(x|y)}$$

Schließlich lässt sich noch der sogenannte Informationsgewinn (Information Gain) in Bezug auf ein Attribut A im Datensatz X berechnen.

Definition 11.4 (Informationsgewinn)

$$Gain(X, A) = H(X) - \sum_{v \in Values(A)} \frac{|X_v|}{|X|} H(X)$$

$Values(A)$ gibt alle möglichen Werte von A an. X_v ist eine Teilmenge von X, in der A den Wert x hat. Mit diesem Maß kann nun auf der Datenmenge bestimmt werden, wie hoch der Informationsgewinn bei einer bestimmten Aufspaltung der Datenmenge ist. Je höher er für diese Aufspaltung ausfällt, desto schärfer können die Daten charakterisiert werden.

Als primäre Informationsquellen nutzt das IDS von Lee et al. nun Audit Daten aus dem Routing Protokoll[3] sowie Positionsinformationen durch in den Knoten integrierte GPS Empfänger. Zunächst muss das IDS aber mittels Trainings- und Testdaten geeignet eingestellt werden. Hierzu werden die Trainings- und Testdaten in mehreren Phasen verarbeitet:

Partitionierung der Daten: die Daten werden derart partitioniert, dass sie eine möglichst kleine *konditionale Entropie* haben. Der Informationsgewinn wird dabei in mehreren Durchläufen mit Testdaten ermittelt und es werden Partitionierungen gewählt, bei denen er einen bestimmten Schwellwert übersteigt.

Transformation der Daten: um die erkannten Partitionen mit hohem Informationsgewinn zu erschließen. Hierbei setzen die Autoren die Tools RIPPER und SVM Light ein, um aus den Ergebnissen des ersten Schrittes entsprechende Transformationsregeln zu definieren und Auswertungsfunktionen aufzustellen.

Funktionen auf Trainingsdaten anwenden: hier werden passende Schwellwerte festgelegt, oberhalb derer später ein abnormales Verhalten erkannt wird.

Funktionen auf Testdaten anwenden: um die Qualität der Erkennung zu verifizieren. Gegebenenfalls müssen sonst die Trainingsdaten modifiziert werden. Auch hier kommt das Tool RIPPER zum Einsatz.

[3] z.B. gecachte Routen, Route-Request und Route-Reply Pakete etc.

Entstehende Alarm Nachrichten auswerten: hier werden Ausschnitte der Audit Daten betrachtet. Wenn in einem Abschnitt mehr Funktionen ein abnormales Verhalten erkennen, als ein normales, so wird dieser Abschnitt als abnormal gewertet. Zusammenhängende abnormale Abschnitte werden schließlich als ein abnormales Ereignis gewertet. Werden abnormale Ereignisse nicht korrekt erkannt, so müssen gegebenenfalls wieder die Trainings-, die Testdaten oder die Schwellwerte angepasst werden.

Der nun entstehende Satz von Testfunktionen und Schwellwerten kann jetzt auf Geräte übertragen werden. Die von den Autoren verwendeten Angriffe bestanden aus einer Modifikation der Source-Route und dem Verwerfen von Paketen. Unter Verwendung des RIPPER Tools konnte das System so eingestellt werden, dass ca. 90 % der Angriffe korrekt erkannt wurden bei einer Fehlerrate von 15 %. SVM Light lieferte demgegenüber eine Erkennungsrate von 99 % bei 2 % Fehlerrate.

Bewertung

Das IDS von Wenke Lee et al. liefert wichtige Impulse für den Einsatz von IDS Systemen in MANETs. Der modulare Aufbau ist ähnlich wie bei Buchegger. Die besondere Stärke dieser Arbeit liegt in der sehr tiefgehenden Analyse des Erkennungsprozesses. Hier wird erstmals der Ansatz der allgemeinen Anomalieerkennung auch auf Ad hoc Netze übertragen.

Allerdings lassen verschiedene Faktoren Zweifel an der Praxistauglichkeit des Systems aufkommen. Zunächst ist der Analyseabschnitt zur Bestimmung der Funktionen und Schwellwerte sehr aufwändig und kann sicher nicht im Ad hoc Netz selbst geleistet werden. Da jedoch, wie in Kapitel 5 festgestellt, die Charakteristik eines solchen Netzwerkes sehr starken Schwankungen unterliegen kann, dürfte es faktisch unmöglich sein, im Vorfeld wirklich repräsentative Test- und Trainingsdaten zu generieren, die den späteren Einsatz der Geräte in einer Vielzahl unterschiedlicher Ad hoc Netze möglich macht.

Auch die von den Autoren durchgeführten Versuche bauen auf einem sehr statischen Benutzerverhalten und nur einfachen, statischen Angriffsvarianten auf. Entsprechend vorsichtig sind die guten Erkennungsleistungen zu bewerten. Außerdem gingen die Simulationen immer von einer sehr langen Laufzeit (10.000 Sekunden) aus, bei der die böswilligen Knoten ihr abnormales Verhalten über die gesamte Laufzeit unverändert gezeigt haben. Dies erleichtert eine korrekte Erkennung natürlich ungemein.

Schließlich hat auch das IDS von Wenke Lee et al. mit dem typischen Problem der Anomalieerkennung zu kämpfen. Die Erkennung von abnormalem Verhalten liefert keinerlei Hinweis auf den Verursacher dieses Verhaltens. Entsprechend kann das IDS keine sinnvolle Reaktion wie den Ausschluss des böswilligen Knotens durch ein IRS veranlassen.

Eine weitere Schwäche der Arbeiten ist die enge Fokussierung auf ein IDS System. Andere Aspekte wie die Authentisierung der Knoten und Absicherung der Nachrichten zwischen den IDS Partner werden nicht erwähnt. Doch gerade in diesem Zusammenspiel mehrerer Komponenten liegt eine der Hauptschwierigkeiten bei der Absicherung von MANETs.

11.3.4. CORE

Das nächste IDS für MANETs, welches an dieser Stelle vorgestellt werden soll, ist der sogenannte „*Collaborative Reputation Mechanism*“ oder kurz *CORE*. Die Autoren Pietro Michiardi und Refik Molva stellen CORE in [MM, MM02, MM03] vor. Auch bei diesem System bewerten sich die Knoten gegenseitig und tauschen auch Bewertungen aus. Erhält ein Knoten eine negative Bewertung, erfolgt eine Bestrafung. Dadurch soll die Kooperation der Knoten erzwungen werden.

Die Autoren gehen dabei von einem allgemeinen System aus, bei dem ein Knoten A einen anderen Knoten B bei der Ausführung einer Funktion $f()$ überwacht. Dies kann z.B. die Weiterleitung eines Paketes sein. Wie bereits in Abschnitt 11.3.1 vorgestellt, könnte A dies z.B. durch Overhearing im Promiscuous Modus überprüfen. Der Funktionswert fließt dann in die Berechnung der Bewertung eines Knotens mit ein.

Michiardi und Molva definieren drei Komponenten innerhalb von CORE:

Requestor: der Requestor verlangt die Ausführung einer Funktion $f()$ durch einen Provider und erhält daraufhin ein Ergebnis.

Provider: der Provider führt die Funktion $f()$ aus und liefert ein Ergebnis zurück.

Validierungsmechanismen: Die Knoten im System prüfen gegenseitig die korrekte Funktion durch die sogenannte Peer Validation. Diese geschieht auf drei verschiedene Arten. Zunächst prüft jeder Requestor die korrekte Ausführung der Funktion durch den Provider und passt die lokale Bewertung entsprechend an. Die Autoren schlagen vor, die Überprüfung der Weiterleitung analog dem Watchdog-Ansatz durchzuführen. Zweitens gibt es auch eine mittelbare Validation. Schickt eine Quelle ein Datenpaket entlang einer Route, so ist er der Requestor der Funktion „Datenpaket an Empfänger zustellen“. Jeder Knoten in der Route ist gleichzeitig Provider, indem er das Paket einen Hop weiter schickt, als auch Requestor, indem er den nächsten Knoten dazu auffordert, das Paket einen Hop weiterzusenden. In dieser Funktion validiert er den Provider, indem er die korrekte Weiterleitung überprüft. Ist die Weiterleitung erfolgreich, meldet er dies an den ursprünglichen Requestor, den Absender, indem er diesem eine Acknowledgment Nachricht (ACK) schickt. Jeder Knoten, der die ACK Nachricht empfängt, passt daraufhin seine indirekte Bewertung entsprechend an. Schließlich gibt es noch einen dritten Validierungsmechanismus, die sogenannte Peer-Validation. Hat der Requestor eine negative Bewertung, so verweigert der Provider die Ausführung der Funktion und schickt statt dessen eine *DoS Nachricht* an alle Nachbarn. Diese prüfen nun, ob ihr Rating des Providers mit der DoS Nachricht im Einklang steht. Wenn ja, wird die DoS Nachricht akzeptiert, ansonsten wird davon ausgegangen, dass der Provider die DoS Nachricht gefälscht hat und der Provider wird abgewertet.

Jeder Knoten berechnet also zwei Werte, welche das Ansehen eines anderen Knotens ausdrücken. Einmal die sogenannte subjektive Reputation und zum anderen die indirekte Reputation. In die subjektive Reputation fließen dabei lediglich eigene Beobachtungen ein, wohingegen die indirekte Reputation auch die Meldungen anderer Knoten berücksichtigt.

Definition 11.5 (Subjektive Reputation)

$$r^t_{s_i}(s_j|f) = \sum \rho(t, t_k) \cdot \sigma_k$$

$r^t_{s_i}(s_j|f)$ drückt die subjektive Reputation aus, die ein Knoten s_i zum Zeitpunkt t über Knoten s_j bezüglich der Ausführung der Funktion f berechnet hat. $\rho(t, t_k)$ ist eine Funktion, welche länger vergangene Ereignisse stärker bewertet als aktuelle, wobei $0 \leq \rho(t, t_k) \leq 1$. σ_k drückt die Bewertung der k-ten Beobachtung aus und kann Werte von -1 bis +1 annehmen. Somit ist auch $r^t_{s_i}(s_j|f) \in [-1; 1]$. Eine subjektive Reputation wird nur über direkte Nachbarn gebildet.

Die indirekte Reputation $ir^t_{s_i}(s_j|f)$, die ein Knoten s_i zum Zeitpunkt t bzgl. eines Knoten s_j und der Funktion f bildet, bestimmt sich anhand der ACK Meldungen anderer Knoten. Dieser Wert kann ausschließlich positive Werte $\in [0; 1]$ annehmen. Beide Werte werden nun zur globalen Reputation verrechnet.

Definition 11.6 (Globale Reputation)

$$r^t_{s_i}(s_j) = \sum_k w_k \cdot \{r^t_{s_i}(s_j|f_k) \; + \; ir^t_{s_i}(s_j|f_k)\}$$

Dabei gehen verschiedene Funktionen f_k mit unterschiedlichen Gewichten w_k in die Berechnung ein. Basierend auf diesem Wert entscheidet nun jeder Knoten, ob er gegenüber einem Requestor als Provider auftritt oder eine Ausführung mit einer DoS Nachricht ablehnt.

Die Autoren untersuchen in ihren Veröffentlichungen theoretisch, wie sich das gegenseitige Ansehen von Knoten bei verschiedenen Angriffsformen entwickelt.

Bewertung

Auch CORE liefert einen wertvollen Beitrag zur Diskussion um IDS für MANETs. Insbesondere wird hier ein komplettes Rahmenwerk betrachtet und die Berechnung des gegenseitigen Ansehens genauer analysiert. Wie bei den anderen Arbeiten werden jedoch auch signifikante Schwachstellen sichtbar.

Zunächst geht auch hier die eigentliche Erkennung eines egoistischen oder böswilligen Verhaltens nicht über einen Hinweis auf den Watchdog Mechanismus hinaus. Zwar sprechen die Autoren davon, dass die Knoten gegenseitig die korrekte Routing Funktion überwachen sollen, wie dies aber geschehen soll, bleibt offen.

Auch wurde das CORE System noch nicht implementiert und in Simulationen getestet, so dass nicht klar ist, ob die Annahmen der Autoren in der Praxis tatsächlich gelten. So gehen die Autoren beispielsweise bei der Peer Validation davon aus, dass die Mehrzahl der Knoten ein konsistentes Rating eines Providers hat, um entscheiden zu können, ob die DoS Nachricht eines Requestors berechtigt ist oder nicht. Unsere folgenden Simulationen haben jedoch gezeigt, dass meist nur ein kleiner Kreis von unmittelbar benachbarten Knoten Kenntnis von der Böswilligkeit eines Knotens erlangen. Somit würde also oft der Sender einer (korrekten) DoS Nachricht abgewertet, anstatt den eigentlichen schuldigen, böswilligen Provider zu bestrafen.

Schließlich gilt auch für CORE, dass es das Thema MANET-IDS isoliert von anderen Sicherheitsmechanismen betrachtet und daher viele der Zusammenhänge unterschlägt.

11.3.5. Nuglets

Als letzte der verwandten Arbeiten soll nun noch ein anderer Ansatz vorgestellt werden, der nicht auf einer Erkennung von böswilligem oder egoistischem Verhalten mit anschließender Bestrafung basiert, sondern vielmehr den Anreiz zur Motivation erhöhen will.

Im Rahmen des Terminode Projektes entwickelten Hubaux und seine Mitarbeiter das Nuglets System [BH01b, BH03]. Nuglets sind eine virtuelle Währung, die in MANETs zur Verrechnung gegenseitiger Dienste verwendet wird. Leitet ein Knoten Daten für einen anderen Knoten weiter, so erhält er Nuglets, erzeugt er selbst Verkehr, so muss er Nuglets ausgeben.

Wie in Abschnitt 6.1.2 und Abbildung 6.1 gezeigt, kann ein Knoten nun selbst bewerten, wie nützlich es für ihn ist, Verkehr weiterzuleiten, um somit Nuglets für die eigenen Datenpakete zu besitzen. Die Autoren diskutieren auch mehrere Alternativen, bei denen man statt für den Versand eines Datenpaketes (Packet Purse Model) für den Empfang bezahlen muss (Packet Trade Model).

Um Manipulationen an der virtuellen Währung zu vermeiden, setzen die Autoren den Einsatz von manipulationssicherer Hardware voraus, die so gestaltet sein muss, dass zwar das Nuglets System, nicht jedoch der Benutzer die Kontostände verändern kann.

Bewertung

Das Nuglets System stellt eine interessante Alternative zur Erkennung von Fehlverhalten dar. Die Knoten werden hier motiviert, sich möglichst intensiv an der Weiterleitung von Paketen zu beteiligen. Auch hier bleiben jedoch einige kritische Fragen unbeantwortet.

Zunächst setzt das System den Einsatz von manipulationssicherer Hardware voraus. Verschiedene Arbeiten [AK96, AK97] legen aber nahe, dass dies nur sehr schwer oder gar unmöglich zu realisieren ist. Auch die Entwickler der Nuglets äußern sich nicht weiter zu diesem Thema.

Ein Problem beim Packet Trade Modell ist, dass ein Angreifer einem Knoten durch Zusenden vieler Pakete alle Nuglets entziehen kann. Somit ist er nicht mehr in der Lage, reguläre Pakete zu bezahlen und zu empfangen. Auch beim Packet Purse Modell kann es zu Problemen kommen, wenn ein Knoten in ungünstiger Lage – z.B. am Rand des Netzwerks – kaum Pakete weiterleiten muss. Da er keine Nuglets verdienen kann, ist er auch nicht in der Lage, selbst Datenverkehr zu generieren. Auch Knoten, die überwiegend große Datenmengen versenden[4], dürften Probleme haben, die notwendigen Nuglets zu erwerben.

Schließlich mag das Nuglets System zwar egoistisches Verhalten zu einem gewissen Grad verhindern, eine Erkennung für böswilliges Verhalten bietet es allerdings nicht. Auch die anderen Sicherheitsfragestellungen wie die Authentisierung der Knoten oder der Schutz der Pakete vor Manipulationen[5] werden nicht adressiert.

[4] z.B. ein Streaming Server
[5] z.B. Löschung der im Paket enthaltenen Nuglets

11.3.6. Vergleich

Wie gezeigt beschränken sich die beschriebenen Systeme bisher auf sehr enge Fragestellungen. Insbesondere die Erkennung von Fehlverhalten ist bisher nicht wesentlich über die Arbeiten zum Watchdog hinaus weiter entwickelt worden. Auch setzen die Systeme meist implizit ein umfangreiches Sicherheitsrahmenwerk für Authentisierung und zur Verhinderung von Manipulationen voraus, ohne dieses oder nur die Anforderungen an dieses genauer zu beschreiben. Solange aber der Absender eines Pakets nicht zweifelsfrei festgestellt werden und somit der Angriff keinem Knoten sicher zugeordnet werden kann, ist jede Reaktion sinnlos. Auch bleibt bei den Systemen unklar, welche möglichen Angriffe das Rahmenwerk abfangen muss und für welche das IDS zuständig ist. Selbst die möglichen Angriffe sind in den Publikationen nicht hinreichend systematisch untersucht und kategorisiert, wie dies im Rahmen dieser Arbeit durch die Angriffsbäume in Abschnitt 6.2 geschehen ist. Im Folgenden wird nun MobIDS als Intrusion Detection System für SAM beschrieben.

11.4. Design von MobIDS

Wie bereits in Kapitel 7 ausgeführt wurde, geht das MobIDS System von einer Reihe von Voraussetzungen aus und liefert einige Voraussetzungen für andere Komponenten. Dabei ist MobIDS eng mit dem SDSR Protokoll verknüpft. Dieses ist für die Sicherheit des Routings zuständig, MobIDS für die Erkennung von Verhalten, das in diesem Rahmen nicht auffällt.

Beide Systeme arbeiten aber auch zusammen. Entdeckt SDSR beispielsweise Manipulationen an den Routing Daten und kann diese eindeutig einem Knoten zuordnen, so wird dies an MobIDS gemeldet und fließt mit in die Bewertungen ein. Umgekehrt liefert MobIDS dem SDSR Protokoll Informationen über gesperrte Knoten, die dann nicht in der Routenfindung berücksichtigt werden.

Die *Architektur von MobIDS* ist ähnlich dem CONFIDANT System oder dem IDS von Zhang/Lee. Wie in Abbildung 11.5 zu sehen, unterstützt MobIDS eine beliebige Anzahl von sogenannten *Sensoren*. Diese werten Audit Daten aus und leiten daraus *Beobachtungen* ab. Der modulare Aufbau gestattet die flexible Erweiterung um zusätzliche Sensoren, falls sich herausstellen sollte, dass ein Fehlverhalten nicht durch die bisherigen Sensoren erkannt wird.

Die *Beobachtungen* werden anschließend zu einer *lokalen Bewertung* verrechnet. Dabei können verschiedene Sensoren je nach ihrer Genauigkeit und Signifikanz unterschiedlich gewichtet werden. Die *Verteilungs-Komponente* ist dafür zuständig, die lokale Bewertung an die anderen Knoten im MANET zu verteilen. Aus der lokalen Bewertung und den Bewertungen der anderen Knoten wird nun eine *globale Bewertung* erstellt.

Unterschreitet die globale Bewertung eines Knotens einen gewissen Schwellwert, so wird der Knoten im lokalen MANET ausgeschlossen, d.h. ab sofort werden sämtliche Route-Requests von ihm ignoriert. Der Knoten kann keine neuen Routen mehr aufbauen. Da auch die gemeinsamen Schlüssel mit anderen Knoten im MANET invalidiert werden, ist auch keine Kommunikation über bereits bestehende Routen mehr möglich.

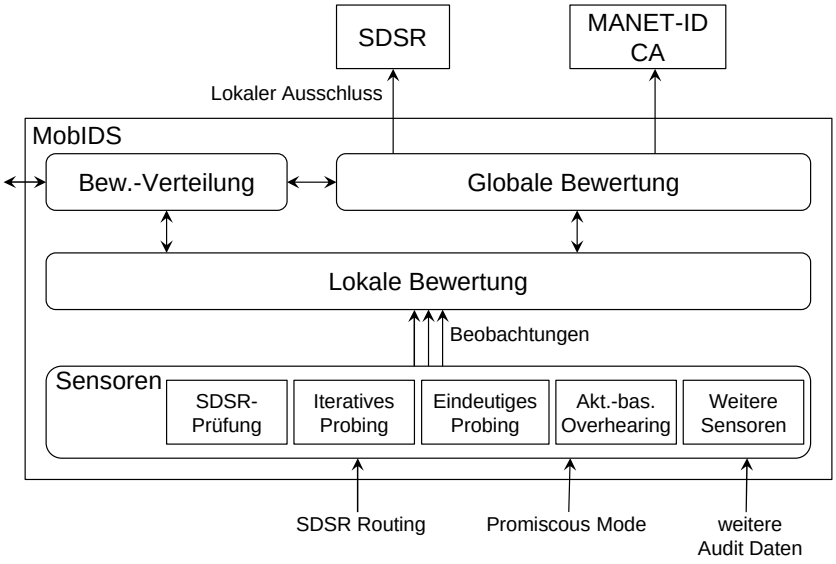

Abbildung 11.5.: Aufbau des Mobile Intrusion Detection Systems MobIDS

Hat ein Knoten über einen anderen Knoten eine lokale Bewertung, welche diesen eindeutig als böswillig ausweist und erhält er von anderen Knoten ebensolche Informationen, so dass sich zusätzlich eine eindeutig negative globale Bewertung ergibt, so wird zusätzlich eine Anschuldigung an die MANET-ID CA vorbereitet und bei nächster Gelegenheit verschickt. Kommen genügend derartige Anschuldigungen bei der MANET-CA zusammen, so wird das Zertifikat des FEB Knotens, wie in Abschnitt 8.3.4 dargestellt, nicht weiter verlängert. Der Knoten kann somit bis auf weiteres an keinen MANETs mehr teilnehmen, welche SAM als Sicherheitssystem verwenden. Dies ist dann die maximale Reaktion, mit der SAM auf einen FEB Knoten reagiert.

Nach diesem groben Überblick sollen nun die einzelnen Komponenten detaillierter betrachtet werden.

11.5. Sensoren

Aufgabe der Sensoren ist es, ein Fehlverhalten eines Knotens im MANET festzustellen. Da es eine große Bandbreite an Fehlverhalten oder Attacken gibt, sind auch unterschiedliche Sensoren notwendig, um dies festzustellen. Im Rahmen dieser Arbeit werden einige neue Sensoren vorgestellt.

11.5.1. Promiscuous Overhearing

Bisher konzentrierten sich die Vorschläge primär auf das Mithören von Netzwerkverkehr im sogenannten Promiscuous Mode (siehe Abschnitt 11.3). Dies soll im folgenden als *Promiscuous Overhearing* bezeichnet werden.

Normalerweise empfängt eine Netzwerkkarte nur Pakete, welche direkt an diese adressiert oder an die Broadcastadresse gerichtet sind. Pakete an andere Adressen werden

bereits in der MAC Schicht ausgefiltert. Im sogenannten *Promiscuous Mode* sind diese Filter deaktiviert und alle Pakete werden empfangen und an den Rechner weitergeleitet.

Dies nutzt der Watchdog und Pathrater Ansatz (siehe Abschnitt 11.3.1, [MGLB00]) um die korrekte Weiterleitung eines Pakets „mitzuhören“. Diese Idee wurde von den nachfolgenden Arbeiten praktisch unverändert übernommen. Dabei haben bereits die ursprünglichen Autoren eine Reihe von Problemen festgestellt.

Probleme bei Promiscuous Overhearing

Zunächst ist festzuhalten, dass ein Netzwerkinterface im Promiscuous Mode signifikant mehr Energie verbraucht als im Standardmodus. In [LF01] wird gezeigt, dass eine Wireless LAN Karte bereits im Ruhezustand etwa 800 mW verbraucht und dieser Wert beim Senden auf 1400 mW ansteigt. Der Empfang benötigt immerhin noch 1000 mW. Entsprechend führt beim dauernden Empfang von Paketen im Promiscuous Modus der erhöhte Energieverbrauch zu einer kürzeren Gesamtlaufzeit der beteiligten Geräte.

Der Einsatz des Promiscuous Modus sollte also wohl überlegt erfolgen und die Resultate den Mehraufwand rechtfertigen. Ein einfaches Overhearing, wie von Marti et al. vorgeschlagen, hat jedoch diverse Probleme, die im Folgenden erläutert werden. Dabei bezeichnet analog zu Grafik 11.2 A den Knoten, der im Promiscuous Mode die korrekte Paketweiterleitung verifiziert, und B den Knoten, der ein Paket P an Knoten C weiterleiten soll. Dann heißt A der *Relayingsensor bezüglich* B.

1. **Empfangsprobleme beim Relayingsensor:** Aus verschiedenen Ursachen erkennt A die korrekte Paketweiterleitung nicht. Während der Übertragung von B zu C kann es bspw. bei A zu einer Kollision mit einem anderen Sender kommen. Ähnlich können Interferenzen wirken. Oder B bewegt sich vor der Übertragung außer Empfangsreichweite von A. Als Folge erkennt A die korrekte Übertragung von P nicht und geht fälschlich von einem Fehlverhalten aus.

2. **Kollision beim Empfänger:** Obwohl A das Paket P korrekt empfängt, kann es bei C zu einer Kollision kommen, die A nicht registriert. B unterlässt nun eine notwendige Wiederholung der Übertragung. Somit geht A fälschlich davon aus, dass das Paket korrekt übertragen wurde.

3. **Anpassung der Sendeleistung:** Unter Umständen kann ein böswilliger Knoten B seine Sendeleistung so anpassen, dass das Paket zwar A erreicht, nicht jedoch C. Wieder geht A fälschlicherweise von einer korrekten Übertragung aus.

4. **Kooperative Angriffe:** Arbeiten mehrere Knoten zusammen, so versagt der Watchdog Ansatz ebenfalls. Angenommen, C sollte das Paket gemäß DSR Source-Route an D weiterleiten und kooperiert mit B. B leitet das Paket an C weiter, was A registriert und als korrektes Verhalten einstuft. C verwirft nun das Paket, B löst daraufhin aber keine Reaktion aus. Ein Fehlverhalten wird also nicht erkannt.

Hierbei ist Fall 1 der gefährlichste, da hier falsch-positive Erkennungen generiert werden. Einem korrekt funktionierenden Knoten wird dabei fälschlicherweise ein Fehlver-

halten zugeschrieben. Als Reaktion kann im Extremfall[6] ein funktionierender Knoten aus dem Netz ausgeschlossen werden. Demgegenüber liefern die anderen drei Fälle lediglich falsch-negative Erkennungen, d.h. ein FEB Knoten wird nicht erkannt. Es wird klar, dass zum einen für Fall 1 die Rate der falsch-positiven Erkennungen gesenkt werden muss und dass sich MobIDS zum anderen nicht allein auf Promiscuous Overhearing zur Erkennung von FEB Knoten verlassen darf, sondern zusätzliche Sensoren zur Ergänzung notwendig sind.

11.5.2. Aktivitätsbasiertes Overhearing

Im Gegensatz zu den zitierten Arbeiten soll das Konzept des Promiscuous Overhearing in dieser Arbeit nicht lediglich übernommen, sondern weiterentwickelt und verbessert werden. Ein Ansatz ist das sogenannte aktivitätsbasierte Overhearing.

A führt beim aktivitätsbasierten Overhearing nun ein Aktivitätsprotokoll seiner Nachbarn. Darin wird vermerkt, wann ein Paket von einem der Nachbarn empfangen wurde. Registriert der Sensor die korrekte Weiterleitung eines Paketes durch B, so führt dieses kooperative Verhalten zu einer positiven Bewertung von B.

Wird die Weiterleitung nicht innerhalb einer definierten Frist detektiert, so wird zunächst überprüft, ob der Knoten in dieser Zeitspanne oder kurze Zeit davor irgendwelche sonstige Aktivität gezeigt hat. Ist dies nicht der Fall, so geht der Relayingsensor davon aus, dass es möglicherweise Empfangsprobleme gibt und A deshalb die Pakete von B generell nicht empfangen kann. Die Bewertung von B bleibt dann unverändert. Hat jedoch A in seinem Logfile Hinweise auf kürzliche Aktivitäten von B, insbesondere auch Pakete, die B selbst abgeschickt hat, so geht A davon aus, dass B die Weiterleitung nicht durchgeführt hat und wertet ihn entsprechend ab.

Später wird im Analyseteil der Arbeit gezeigt, dass hiermit eine bessere Erkennung und höhere Trennschärfe zwischen korrekten und FEB Knoten ermöglicht wird. Bei diesen Untersuchungen stellte sich heraus, dass es noch eine weitere Möglichkeit des Overhearings mit nochmals leicht verbesserten Erkennungsraten gibt.

11.5.3. Kombiniertes Overhearing

Hierbei dient die Aktivitätserkennung lediglich zur Bewertung der Beobachtungen. Wird eine fehlende Weiterleitung detektiert, so hängt der Grad der Abwertung des Knotens vom Aktivitätslog ab. Hat der Knoten kürzlich Aktivitäten gezeigt, so erfolgt eine starke Abwertung, sonst eine schwächere. Unsere Analysen haben gezeigt, dass sich damit die Erkennungsleistung im Vergleich zum rein aktivitätsbasierten Overhearing nochmals leicht steigern lässt.

11.5.4. Probing

Aufgrund der genannten Probleme beim Overhearing kann ein hinreichend intelligenter Angreifer einer Erkennung durch Overhearing mit einigem Aufwand entgehen. Deshalb

[6] Dazu müssen bei MobIDS aber immer mehrere Fehlerkennungen auf unterschiedlichen Knoten zusammen kommen.

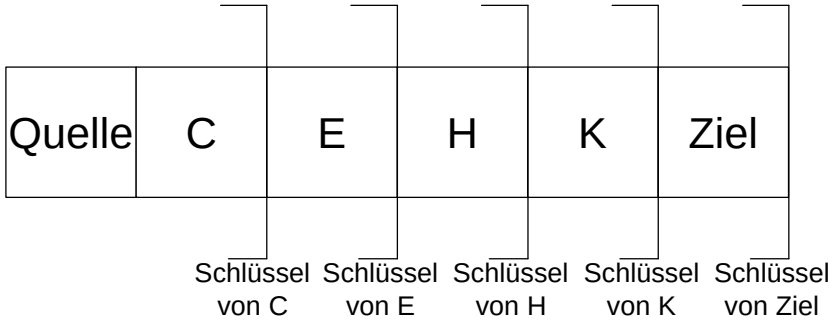

Abbildung 11.6.: Aufbau der Probe List nach Awerbuch

bietet es sich an, verschiedene Erkennungsmethoden zu kombinieren, um so eine Umgehung aller Sensoren zu erschweren. Ein denkbarer Ansatz ist das sogenannte Probing. Hierbei wird eine bestehende Verbindung gezielt auf bösartiges Verhalten hin untersucht.

Probing bei Awerbuch

Ein solches Konzept stellen Awerbuch et al. in [AHNRR02] vor. Dabei werden gezielte Testpakete, sogenannte *Probes*, entlang einer Route geschickt, um darin befindliche FEB Knoten zu finden. Auch Awerbuch geht von einem reaktiven Routingprotokoll ähnlich DSR aus, welches mit einer festgelegten Source-Route arbeitet. Er setzt ferner voraus, dass die Knoten entlang einer Route analog zu den Voraussetzungen für MobIDS authentisiert sind und paarweise geheime Schlüssel vereinbart wurden (siehe Abschnitt 7.2).

Ein Paket kann dann eine in Abbildung 11.6 gezeigte *Probe List* enthalten, welche nach dem Zwiebelschalenmodell [SGR97] mehrfach verschlüsselt ist. Jeder Knoten versucht, die Liste zu entschlüsseln und prüft dann, ob er selbst an vorderster Stelle der Liste steht. Wenn ja, muss der Knoten eine Bestätigung (ACK) an den Empfänger zurückschicken.

Im Normalfall schickt lediglich das Ziel ein ACK an die Quelle zurück. Bleiben die ACKs für eine gewisse Zeit aus, initiiert die Quelle den Probing Prozess. Awerbuch et al. schlagen dabei eine binäre Suche nach dem fehlerhaften Link vor (siehe Abbildung 11.7). Die Probe an das Ziel kommt nicht an, weil sie vom FEB Knoten B verworfen werden. Auch die Probe an B selbst bleibt unbeantwortet. Erst die Probe an A wird beantwortet. Somit schließt die Quelle, dass der Link zwischen A und B fehlerhaft ist.

Bewertung

Der Ansatz von Awerbuch hat eine Reihe von kleinen Nachteilen. Zunächst ist der Verschlüsselungsaufwand beim Aufbau einer Probe List bedingt durch das Zwiebelschalenmodell relativ hoch. Auch muss jedes Paket vom Ziel durch ein explizites ACK

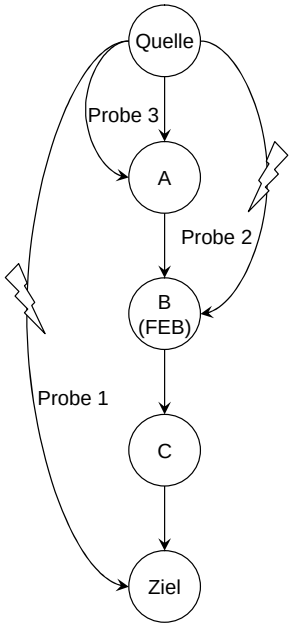

Abbildung 11.7.: Binäres Probing

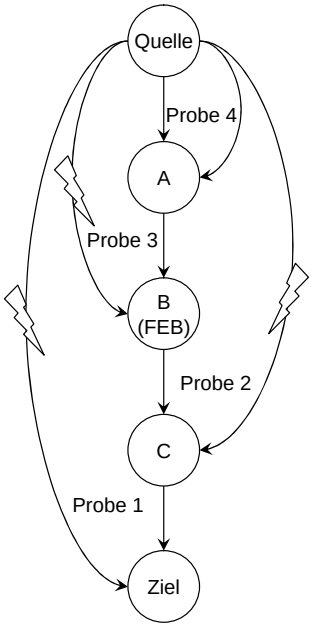

Abbildung 11.8.: Iteratives Probing

Paket bestätigt werden, was möglicherweise unnötigen Aufwand bedeutet, wenn sowieso Antwortpakete generiert werden. Wirklich schwerwiegend ist allerdings, dass es keine Möglichkeit zur sicheren Erkennung von bösartigen Knoten bietet. Awerbuch et al. schreiben auch explizit, dass deren System keine Erkennung von bösartigen Knoten, sondern lediglich von defekten Links bietet. Entsprechend werden auch bösartige Knoten nicht bestraft, sondern lediglich in folgenden Routen vermieden (analog zum Watchdog/Pathrater Ansatz), was genau der Intention von egoistischen Knoten entspricht.

Das Probing Dilemma

Ein Knoten, der Pakete verwirft, hat grundsätzlich die Wahl, ob er auf Probes antwortet oder nicht. Angenommen B *antwortet nicht* auf Probes (Abbildung 11.10). Somit könnte die Quelle schließen, dass B der FEB Knoten ist. Genauso gut könnte die Quelle aber annehmen, dass der FEB Knoten antwortet. Also könnte auch A der FEB Knoten sein. Verdächtig sind in diesem Fall also die Knoten A und B. Wenn B auf Probes *antwortet* (Abbildung 11.9), dann sind analog die Knoten B und C verdächtig. Awerbuch umgeht das Problem, indem er nicht einen Knoten beschuldigt, sondern den Link zwischen den Knoten als fehlerhaft deklariert. Im Hinblick auf eine Bestrafung und den Ausschluss von FEB Knoten hilft dies nicht wirklich weiter.

Zusätzlich kann Knoten B, wenn er selbst eine Probe Nachricht erhält, beschließen, dass er für eine beschränkte Zeit – während des Probing Durchlaufs – alle Datenpakete weiterleitet. Somit wird er überhaupt nicht auffindbar. Kennt B das Probing Schema, ist er unter Umständen sogar in der Lage, selektiv einen anderen Knoten zum Opfer des Probing Mechanismus zu machen. Hierzu muss er nur die ACK Pakete des Knotens

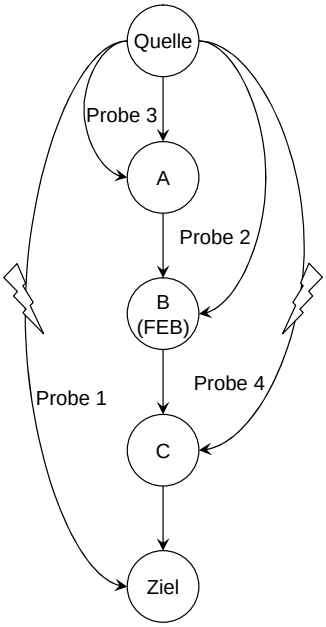

Abbildung 11.9.: B antwortet: potentiell FEB sind $\{B,C\}$

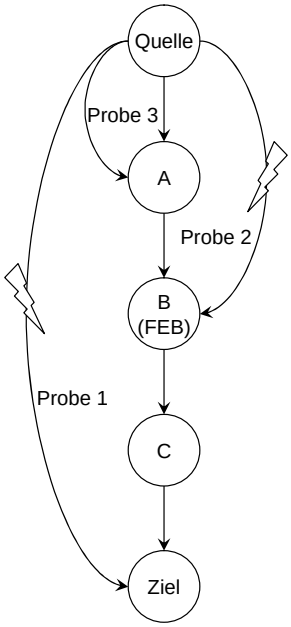

Abbildung 11.10.: B antwortet nicht: potentiell FEB sind $\{A,B\}$

bzw. die Probe Pakete an diesen Knoten abfangen, so dass dessen Antwort die Quelle nicht erreicht.

11.5.5. Iteratives Probing

Zur Vermeidung des letzten Problems verwendet MobIDS das sogenannte iterative Probing. Hier werden die Probe Pakete nicht analog einer binären Suche verschickt, vielmehr werden diese zunächst ans Ziel und dann an den jeweils vorherigen Knoten in der Sourceroute geschickt (siehe Abbildung 11.8).

Die Aufforderung zum Senden eines ACKs drückt die Quelle in einem Kommandofeld PK im Paketheader aus. Soll keine Probe ausgelöst werden, so setzt die Quelle dieses auf einen zufälligen Wert, sonst verschlüsselt sie die Adresse des zu prüfenden Knotens (konkateniert mit einem Zufallswert) mit dem während der Route Discovery vereinbarten geheimen Schlüssel und setzt PK auf den resultierenden Wert. Will also A eine Probe-Aufforderung an B schicken, so setzt er $PK = E_{K_{AB}}(B \cdot random(x))$.

Jeder Zwischenknoten und das Ziel entschlüsseln PK und prüfen, ob das Ergebnis der eigenen Adresse entspricht. Wenn ja, schickt der Knoten ein (verschlüsseltes) ACK Paket zurück an die Quelle, sonst ignoriert er das PK Feld. Anschließend wird das Paket unverändert weitergeschickt.

Probe Pakete werden unter folgenden Umständen geschickt:

1. *Beim Aufbau der Route* nach Erhalt eines Route-Replies schickt die Quelle eine Probe an das Ziel, um die Funktionsfähigkeit der Route zu überprüfen. Sind in

der Route fehlerhafte Knoten enthalten, welche Pakete verwerfen, so wird dies erkannt und es wird ein iteratives Probing gestartet, um diese zu identifizieren.

2. *Nach Ablauf eines zufällig bestimmten Intervalls* schickt die Quelle eine Probe zum Ziel, um die weitere Funktionsfähigkeit der bestehenden Route zu überprüfen. Geht die Probe schief, wird wiederum ein iteratives Probing initiiert.

3. Schließlich werden Probe Pakete *im Rahmen des iterativen Probings* verschickt, um gezielt die Funktionsfähigkeit einzelner Knoten zu überprüfen.

Durch dieses Vorgehen werden verschiedene Ziele erreicht:

- Ein FEB Knoten besitzt keine Informationen darüber, ob das Kommandofeld eines Pakets eine Probe-Aufforderung enthält oder nicht, sofern diese Probe nicht an ihn selbst gerichtet ist.

- Da eine Probe-Aufforderung an einen bestimmten Knoten durch den konkatenierten Zufallswert jedes mal unterschiedlich aussieht, wird verhindert, dass ein FEB Knoten aus wiederholt auftretenden PK Werten auf eine Probe-Aufforderung schließen kann.

- Wegen des iterativen Probings erkennt ein FEB Knoten erst dann, dass die Quelle ein Probing durchführt, wenn er selbst an der Reihe ist. Daraufhin ist es zu spät sein Verhalten noch zu ändern, da die hinter ihm liegenden Knoten bereits überprüft wurden. Dies steht im Gegensatz zur binären Suche von Awerbuch, bei der ein FEB Knoten beim Erhalt einer Probe durchaus noch Probes an hinter ihm liegende Knoten beeinflussen kann.

- Durch die zufällig gewählten Intervalle zwischen den Probings zur regelmäßigen Prüfung einer Route kann ein FEB Knoten diese auch nicht vorhersehen.

Will ein Knoten also nicht vom Probing detektiert werden, so bleibt ihm nichts anderes übrig, als alle Pakete weiterzuleiten. Allerdings kann er sich bei Erhalt einer Probe noch entscheiden, ob er diese beantwortet oder nicht. Analog zur Argumentation am Beginn dieses Abschnitts kann der Sensor dann nicht eindeutig entscheiden und muss zwei Knoten verdächtigen. Um dennoch zu eindeutigen Ergebnissen zu kommen, ist ein eindeutiges iteratives Probing notwendig.

11.5.6. Eindeutiges iteratives Probing

Eindeutiges iteratives Probing kombiniert das iterative Probing mit dem Overhearing Sensor. In einem zusätzlichen (verschlüsselten) Headerfeld $OH = ID$ weist die Quelle einen geprobten Knoten X an, in den Audit Daten des Overhearing Sensors zu prüfen, ob dieser die Weiterleitung des vorletzten Probepakets mit der Paketnummer ID durch den folgenden Knoten Y bemerkt hat. Dieser meldet in seinem ACK Paket nun entsprechend $OH = ID, \{ACK|NACK\}$.

Die Abbildungen 11.11 und 11.12 verdeutlichen dieses Vorgehen. In Abbildung 11.11 wird davon ausgegangen, dass der FEB Knoten B auf Probes antwortet. Die Probe Pakete mit den IDs 1 und 2 fordern noch kein Overhearing an. Da B beide Pakete verwirft, werden keine ACK Pakete generiert und die Quelle schickt iterative Probes zu den vorherigen Knoten. Probe 3 ist an Knoten B gerichtet und enthält die Aufforderung, den Transport von Probe 1 durch Knoten C zu überprüfen. B antwortet

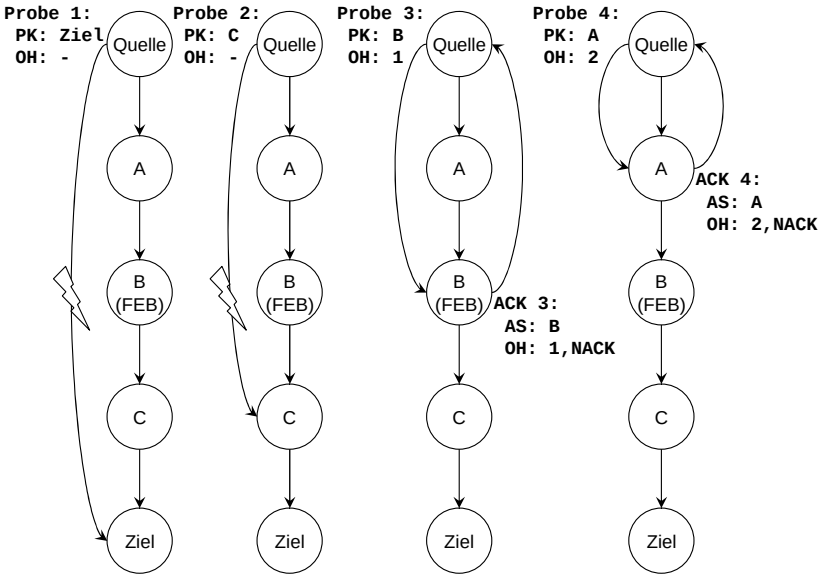

Abbildung 11.11.: Eindeutiges Probing 1 (FEB Knoten antwortet auf Probe)

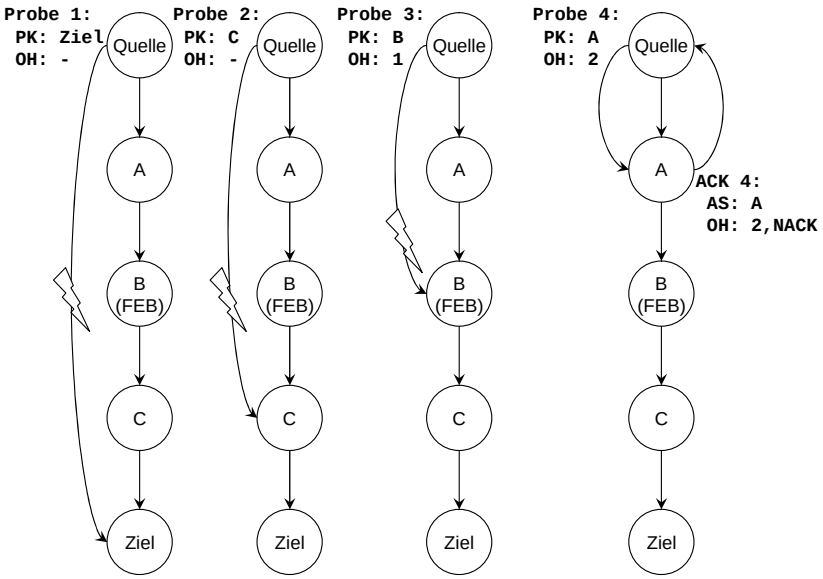

Abbildung 11.12.: Eindeutiges Probing 2 (FEB Knoten antwortet nicht auf Probe)

mit einem ACK und sagt im OH Feld, dass Probe 1 von C nicht weitergeleitet wurde (NACK). C wird also fälschlicherweise von B beschuldigt, nicht kooperiert zu haben. Um Klarheit über die Situation zu erlangen, muss die Quelle noch ein weiteres Probe Paket an Knoten A schicken. Dieser wird angewiesen, die Weiterleitung von Probe 2 zu überprüfen. Da er keine Weiterleitung von Probe 2 feststellen konnte, meldet er ein NACK. Daraus kann die Quelle schließen, dass B der FEB Knoten ist.

Ähnlich ist der Ablauf, wenn der FEB Knoten B nicht auf Probes antwortet (Abbildung 11.12). Das erste ACK empfängt die Quelle von Knoten A. Dieses sagt aus, dass Knoten B das zweite Probe Paket nicht weitergeleitet hat. Bei einer längeren Route wäre nun noch eine weitere Probe notwendig, so kann die Quelle selbst in ihren Audit Daten nachsehen, ob sie die Weiterleitung von Probe 3 durch A verifizieren kann.

Natürlich besteht die Möglichkeit, dass der Overhearing Sensor ein korrektes Weiterleiten nicht erkennt. In diesem Fall würde der falsche Knoten als FEB erkannt. Die Wahrscheinlichkeit hierfür ist allerdings relativ gering, da ja zusätzlich erst die Probe verloren gehen muss.

11.5.7. Route-Request Scanning

Die vorgestellten Sensoren erkennen primär FEB Knoten, die sich weigern, Verkehr weiterzuleiten. Was noch nicht erkannt wird, sind Knoten, welche Route Request Pakete nicht weiterleiten und deshalb nicht Bestandteil einer Route werden. Dieser spezielle Fall wurde bisher in der Literatur noch nicht näher betrachtet.

Das *Route-Request Scanning* prüft wiederum mittels Promiscuous Overhearing, ob ein Knoten einen empfangenen Route-Request weiterleitet. Leitet ein Knoten einen Route-Request weiter, so prüft er in der Folgezeit, ob jeder Nachbarknoten diesen Request ebenfalls weiterleitet. Ausgenommen ist natürlich der Knoten, von dem der Request empfangen wurde.

Voraussetzung dafür ist, dass ein Route Request in jedem Fall weitergeleitet werden muss. Caching von Routen und die Erzeugung von Route Replies durch Zwischenknoten sind dann nicht mehr möglich. Da jedoch das SDSR Protokoll wegen der notwendigen Schlüsselgenerierung sowieso keine derartigen Optimierungen vorsieht, ist dies keine echte Einschränkung. Erreicht der Time-To-Live Wert eines Route Request Pakets den Null-Wert, wird natürlich keine Weiterleitung vom nächsten Knoten erwartet.

Weiterhin ist natürlich die Kenntnis der Nachbarschaftsknoten Voraussetzung für eine korrekte Prüfung. Da SDSR genau wie DSR die Verwaltung einer Nachbarschaftsliste zunächst nicht vorsieht, muss dies durch einen separaten Mechanismus, die *Nachbarschaftserkennung* geleistet werden. Da die Knoten sowieso im Promiscuous Modus arbeiten, müssen sie lediglich alle aufgefangenen Pakete analysieren und die jeweiligen Knoten in die Nachbarschaftsliste eintragen. Hier besteht allerdings die Gefahr, dass Knoten die Anwesenheit von anderen Knoten durch gefälschte Pakete vortäuschen und so diese Knoten in Misskredit bringen. Entsprechend sollten nur Pakete berücksichtigt werden, welche durch eine digitale Signatur eindeutig zuzuordnen sind. Dies ist z.B. bei Route-Request und -Reply Paketen der Fall.

Ergänzt werden kann dieses System durch ein explizites *Nachbarschaftserkennungsprotokoll*, bei welchem sich Knoten durch signierte *Hello-Pakete* in regelmäßigen Ab-

ständen melden müssen. Bleibt diese Meldung über einen längeren Zeitraum aus und erkennt ein Knoten trotzdem Aktivitäten eines Nachbarn (z.B. Route-Request Pakete), so wird dies als unkooperatives Verhalten gewertet und führt entsprechend zu einer Abwertung.

Erkennt das Route-Request Scanning eine korrekte Weiterleitung durch einen Nachbarknoten, so wird die lokale Bewertung über diesen Knoten erhöht, im anderen Fall erniedrigt.

11.5.8. Weitere Sensoren

Neben den vorgestellten Varianten sind eine Vielzahl weiterer Sensoren denkbar, welche die Erkennungsleistung des MobIDS Systems weiter steigern können. So sollte z.B. analog dem Route-Request Scanning auch die korrekte Rückübermittlung des Route Replies geprüft werden. Dies kann aber bereits der aktivitätsbasierte Overhearing Sensor leisten, da es sich hier ja um eine reguläre Unicast-Übertragung handelt.

Komplexere Sensoren könnten versuchen, die gesammelten Topologieinformationen auf ihre Konsistenz zu prüfen, um auf den Verursacher von Topologiemanipulationen zurückzuschließen[7]. Auch sind Sensoren denkbar, welche klassische Attacken wie DoS erkennen und in die Bewertung einfließen lassen. Insbesondere kann auch das SDSR Protokoll als Sensor dienen. Erkennt dieses Manipulationsversuche, die sich eindeutig einem Knoten zuordnen lassen, so kann dies zu einer sofortigen deutlichen Abwertung führen.

Dabei sollte man in der Praxis die Balance zwischen genügender Erkennungsleistung und dem zusätzlichen Overhead durch viele Sensoren bewahren. Wie bereits in Abschnitt 7.2 geschildert, genügt es im Kontext von SAM, wenn die Möglichkeit einer Erkennung samt anschließendem Ausschluss die Teilnehmer so stark abschreckt, dass sie ein kooperatives Verhalten zeigen. Wie gut hier die Erkennung tatsächlich sein muss, können erst längere Erfahrungen im realen Praxiseinsatz zeigen.

11.6. Lokale Bewertung

Aus den Meldungen der Sensoren bildet MobIDS zunächst eine lokale Bewertung, die das subjektive Ansehen ausdrückt, welches ein anderer Knoten beim lokalen Knoten genießt. Das Verfahren zur Bildung des lokalen Ansehen ist dabei ähnlich aufgebaut wie bei CORE, welches in Abschnitt 11.3.4 vorgestellt wurde. Es unterscheidet sich aber in wesentlichen Details von CORE, z.B. bei der Abbaurate von Bewertungen und in der Verteilung von Bewertungen.

Die Sensoren liefern die Eingabe für die lokale Bewertung in Form von *Beobachtungen*. Eine Beobachtung σ_n^s drückt die n-te Beobachtung des Sensors s aus. Dabei ist $\sigma_n^s \in [-1;1]$, wobei positive Werte eine positive Beobachtung ausdrücken, wohingegen eine negative Bewertung ein beobachtetes Fehlverhalten repräsentieren. Bezüglich eines Sensors s wird daraus nun eine *Sensorbewertung* ermittelt.

[7] im Rahmen von SAM ist dies nicht notwendig, da SDSR derartige Manipulationen komplett ausschließt

$$r_{k_i}^t(k_j|s) = \left(\sum_{\forall n} \rho(t, t_n) \cdot \sigma_n \right) / n$$

Zur Bestimmung der Sensorbewertung des Knotens k_i über den Knoten k_j zum Zeitpunkt t bezüglich eines Sensors s werden also alle verfügbaren Sensorbewertungen σ_n mittels einer Funktion $\rho(t, t_n)$ gewichtet und aufsummiert. Der resultierende Wert wird wieder auf den Bereich $[-1; 1]$ normalisiert. Dabei ist

$$\rho(t, t_n) = 1 - \left(\frac{t - t_n}{T} \right)^x$$

t_n ist dabei der Zeitpunkt der Beobachtung σ_n. $\rho(t, t_n)$ liefert eine von 1 auf 0 abfallende Funktion über das Zeitintervall $[t-T; t]$, welche dafür sorgt, dass ältere Beobachtungen nach und nach unwichtiger werden. Über den Parameter x kann der Grad des Abfalls bestimmt werden. T ist dabei die maximale Gültigkeitsdauer einer Beobachtung. Nach Ablauf von T Zeiteinheiten werden alte Beobachtungen gelöscht und gehen nicht mehr in die Rechnung ein.

CORE geht hier den umgekehrten Weg und wertet ältere Beobachtungen stärker als aktuelle. Dieses Vorgehen ist aus meiner Sicht nicht nachvollziehbar und wird auch in den Publikationen von Michiardi und Molva nicht hinreichend begründet. Es erscheint mir naheliegender, aktuelle Beobachtungen etwas höher zu gewichten als ältere, wobei aber ein einzelnes kurzfristiges Fehlverhalten durch ein auf längere Sicht positives Verhalten aufgewogen werden kann. Alternativ zu einer vorgegebenen Funktion könnte die Erstellung einer idealen Bewertungsfunktion auch aus Testdaten ermittelt werden, z.B. durch Anpassen eines neuronalen Netzes.

Anschließend berechnet der Knoten k_i aus den verschiedenen Sensorbewertungen seine *lokale Bewertung* über den Knoten k_j zum Zeitpunkt t als

$$r_{k_i}^t(k_j) = \sum_{\forall s} w_s \cdot r_{k_i}^t(k_j|s)$$

Jede Sensorbewertung wird also nochmals mit einem Faktor w_s gewichtet. Dies erlaubt es, die Bewertungen von zuverlässigen Sensoren stärker zu berücksichtigen, als die von unzuverlässigen. Es gilt $w_s \in [0, 1]$.

Die Wertebereiche von $r_{k_i}^t(k_j|s)$ und $r_{k_i}^t(k_j)$ liegen, wie die σ_n, ebenfalls im Intervall von $[-1, 1]$, d.h. ein gutartiger Knoten kann maximal eine Bewertung von 1 erhalten, ein FEB Knoten maximal mit -1 bewertet werden.

Es fällt auf, dass hier keine Aussage über die konkreten Werte von σ_n und w_s getroffen wird. Tatsächlich hängt aber die Erkennungsleistung von MobIDS maßgeblich von der korrekten Einstellung dieser Werte ab. Zum jetzigen Zeitpunkt werden die passenden Werte durch Simulationen manuell ermittelt. Denkbar wäre natürlich auch, die Werte dynamisch an die jeweiligen Gegebenheiten anzupassen. Hierzu müsste ein Feedback zwischen Erkennung und Sensor realisiert werden, in welches auch Informationen über

Dichte, Mobilität etc. des MANETs einfließen können. Diese Idee wird im letzten Kapitel nochmals aufgegriffen.

Die bisher vorgestellten Sensoren sind meist so ausgelegt, dass ein Knoten vor allem lokale Bewertungen über seine direkten Nachbarn sammelt. Lediglich der Probing Sensor sammelt auch Bewertungen über entferntere Knoten. Somit ist davon auszugehen, dass die lokalen Bewertungen über Knoten im MANET stark differieren können und die meisten Knoten vermutlich mangels direkter Beobachtungen gar keine lokale Bewertung bilden können.

Ein Ausschluss eines Knotens als Reaktion auf ein Fehlverhalten muss aber im MANET abgestimmt werden. Würde sonst ein Knoten A bspw. keine Pakete eines FEB Knotens B mehr annehmen, dann könnte der Fall auftreten, dass A durch den Probing Sensor selbst als FEB erkannt wird. Um dies zu vermeiden, stimmen bei MobIDS die Knoten zunächst ihre lokalen Bewertungen ab und bilden daraus eine globale Bewertung, welche als Entscheidungsgrundlage für einen Ausschluss dient.

11.7. Verteilung und Globale Bewertung

Ein Knoten k_i erstellt seine lokalen Bewertungen $r^t_{k_i}(k_j)$ über Nachbarn in regelmäßigen Abständen Δt und *verteilt diese an seine Nachbarn*. Dabei werden die lokalen Bewertungen in einer Liste gesammelt, diese wird mit einer eindeutigen Seriennummer ID und dem Zeitstempel t versehen und von k_i digital signiert, um die Urheberschaft zweifelsfrei nachweisen zu können (siehe Abbildung 11.13). Hat ein Knoten mangels eigener Beobachtungen keine lokale Bewertung über einen bestimmten Knoten k_j, so verschickt er natürlich auch keine Bewertung. Mittels eines TTL Wertes läst sich die Reichweite einer solchen Liste kontrollieren.

Empfängt ein anderer Knoten eine solche Liste, so wird zunächst geprüft, ob diese Liste schon früher empfangen wurde. Wenn ja, wird die Nachricht ignoriert. Sonst wird zunächst die Signatur geprüft und bei Erfolg die Liste solange gespeichert, bis sie durch eine Liste mit höherer ID ersetzt wird, oder ein Mehrfaches des Intervalls Δt verstrichen ist. In letzterem Fall wird davon ausgegangen, dass sich der Knoten k_i außer Reichweite bewegt hat und keine weiteren Bewertungen von ihm empfangen

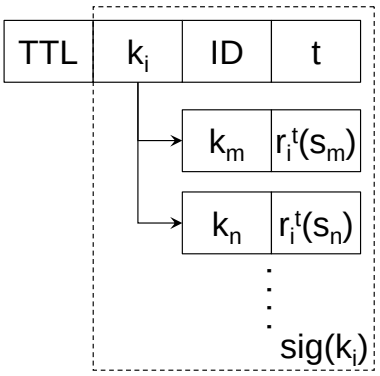

Abbildung 11.13.: Aufbau der Bewertungsliste

mehr werden. Wenn es der herabgezählte TTL erlaubt, wird die Liste an die Nachbarn geflutet. Dabei sorgt eine Ratenkontrolle dafür, dass das Netz nicht mit solchen Listen überflutet werden kann.

Für jeden Knoten k_j in der Liste berechnet der Empfänger nun eine neue *globale Bewertung*. Hierzu wird die durchschnittliche Bewertung gemäß den vorliegenden lokalen Bewertungen berechnet. Allerdings wird diese nur als gültig angesehen, wenn von mindestens n Knoten eine lokale Bewertung vorliegt. Hiermit wird verhindert, dass einzelne Knoten oder kleine Gruppen einen anderen Knoten durch sehr negative Bewertungen diskreditieren. Auch haben dann Fehlbewertungen einzelner Knoten, die sich z.B. aus einer ungünstigen Bewegungsrichtung ergeben können, keine Auswirkungen.

11.8. Lokaler Ausschluss

Erreicht ein Knoten eine hinreichend schlechte globale Bewertung, so erfolgt als lokale Reaktion ein Ausschluss des Knotens aus dem lokalen MANET. Dabei ist jedoch ein Problem zu beachten.

Bedingt durch den Verteilungsprozess und die beschränkte Reichweite der Verteilung können Knoten natürlich immer noch zu einer unterschiedlichen globalen Bewertung kommen. Unterschiedliche Bewertungen können zu folgendem Szenario führen: Knoten A hat über Knoten B eine schlechte globale Bewertung gebildet und will diesen ausschließen. Knoten C kommt aufgrund anderer Daten jedoch zu einer besseren Bewertung und erkennt B nicht als FEB. Wie oben bereits angeführt, könnten dann bestimmte Sensoren bei C ein FEB Verhalten von A erkennen, weil dieser die Kommunikation mit B verweigert. Im Extremfall wird A aus dem Netz ausgeschlossen.

In CORE wird dies durch das verschicken expliziter DoS Nachrichten gelöst (siehe Abschnitt 11.3.4). Doch auch hier gibt es Probleme. Wie soll ein Knoten reagieren, wenn er selbst eine positive Bewertung von C hat, jedoch eine DoS Nachricht von A erhält?

Bei MobIDS wird das Problem durch die Einführung einer Hysterese-Funktion gelöst. Diese kennt, wie in Abbildung 11.14 gezeigt, drei Schwellwerte. Unterschreitet die globale Bewertung eines Knotens die *Reaktionsschwelle*, so wird dieser Knoten als FEB erkannt und dann so lange ignoriert, bis die schlechten Bewertungen verfallen und die globale Bewertungen irgendwann wieder die *Rehabilitationsschwelle* übersteigt. Unterschreitet die Bewertung die *Toleranzschwelle*, so wird der Knoten andere Knoten nicht dafür abwerten, wenn sie den FEB Knoten ignorieren. Durch geeignete Wahl der Schwellwerte und der Verteilungszeiträume kann sichergestellt werden, dass die globalen Bewertungen verschiedener Knoten in der Nachbarschaft eines FEB Knotens zumindest hinreichend ähnlich sind und somit Knoten nicht dafür bestraft werden, dass sie einen FEB Knoten ignorieren.

Der Ausschluss selbst erfolgt durch ein schlichtes Ignorieren eines Knotens. Route-Requests und Datenpakete, die von diesem Knoten ausgehen, werden verworfen. Routen, welche diesen Knoten beinhalten, werden invalidiert und gegebenenfalls werden RERR Pakete verschickt. Route-Replies, welche durch diesen Knoten laufen, werden ebenfalls durch entsprechende RERR Pakete ersetzt.

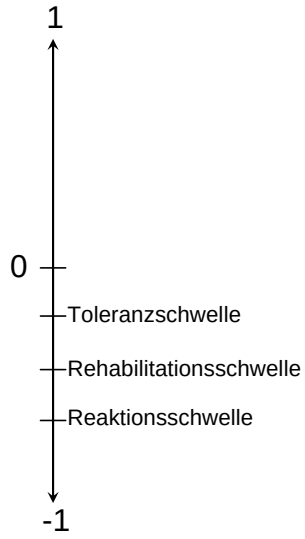

Abbildung 11.14.: Toleranz- und Reaktionsschwelle bei MobIDS

Ist sich die Nachbarschaft eines Knotens in der negativen globalen Bewertung eines FEB Knotens einig, so ist dieser effektiv isoliert. Er kann selbst keine Route-Discovery mehr initiieren und wird auch in anderen Route-Discoveries nicht mehr berücksichtigt.

Allerdings hat er noch die Möglichkeit, nach Ablauf einer gewissen Zeit und Erholung seiner globalen Bewertung wieder kooperativ am Netz teilzunehmen und seine Bewertung somit zu verbessern. Dagegen ist der globale Ausschluss über die CA zunächst irreversibel.

11.9. Globaler Ausschluss

Verweigert ein Knoten dauerhaft die Kooperation im MANET oder verhält sich ständig bösartig, so ist der lokale Ausschluss unter Umständen kein geeignetes Mittel mehr. Der Knoten kann bis zur Erkennung durch MobIDS ungehindert den Netzbetrieb stören. Auch wird sich seine Bewertung langsam wieder erholen, so dass er nach einiger Zeit wieder aktiv werden kann.

Um gegen solche Knoten vorzugehen, sieht SAM den globalen Ausschluss vor. Wie in Abschnitt 8.3.4 ausgeführt, kann die CA jederzeit die Verlängerung von Identitätszertifikaten verweigern, indem keine neuen Verifikatoren mehr ausgegeben werden.

Hierzu schickt das MobIDS System eines Knotens eine Meldung an die CA, sobald die lokale Bewertung einen bestimmten Schwellwert unterschreitet. Kommen über einen längeren Zeitraum hinweg immer wieder Beschwerden verschiedener Knoten über einen FEB Knoten, so sieht die CA dies als Beweis eines dauerhaften Fehlverhaltens und sperrt die Identität dauerhaft. Eine Entsperrung ist mit einem manuellen Antrag bei einem Vertreter der CA verbunden.

Name des Systems	Watchdog Pathrater	CONFIDANT	CORE	Wenke-Lee	SAM & MobIDS	Nuglets
Sicherheit durch	Erkennung					Vermeidung
Abgedecktes Verhalten						
fehlerhaft	ja	ja	ja	ja	ja	nein
egoistisch	ja	ja	ja	ja	ja	ja
böswillig	nein	nein	nein	ja	ja	nein
Informationsquelle						
Routing	nein	nein	nein	?	ja	nein
Promiscuous Mode	ja	ja	ja	?	ja	nein
IDS Nachrichten	nein	ja	ja	ja	ja	nein
Bewertung des Angriffs						
Lokal	ja	ja	ja	ja	ja	nein
Global	nein	ja	ja	ja	ja	nein
Reaktion						
Bestrafung	nein	ja	ja	ja	ja	nein
Belohnung	nein	nein	nein	nein	nein	ja
Vermeidung	ja	ja	ja	ja	ja	nein
Vertrauen	sich selbst	befreundete Knoten	positiv bew. Knoten	Mehrheit	pos. bew. Kn. m. gült. Zert.	sichere Hardware
Informationsaustausch durch	nein	ja	ja	ja	ja	nein
Denunzierungen	nein	ja	nein	ja	nein	nein
Bewertungen	nein	ja	ja (pos.)	ja	ja	nein
Missbrauch möglich	nein	ja	ja	ja	nein	nein
Schutz						
Nutzdaten	nein	nein	nein	nein	ja	nein
IDS Nachrichten	nein	ja	ja	nein	ja	ja (Nuglets)
Lokales System	nein	nein	nein	nein	nein	nein
Architektur						
Zentral	nein	nein	nein	nein	nein	nein
Regelbasiert	ja	ja	ja	nein	ja	nein
Anomalieerk.	nein	nein	nein	ja	nein	nein

Tabelle 11.1.: Vergleich von MANET IDS

Diese Ultima Ratio sollte natürlich nur in gravierenden Fällen angewendet werden. Entsprechend konservativ sind hier die Schwellwerte zu wählen.

11.10. Fazit

In diesem Kapitel wurden Intrusion Detection Systeme für MANETs vorgestellt. Tabelle 11.1 gibt nochmals einen tabellarischen Überblick über die behandelten Systeme und erlaubt einen Vergleich der Funktionalität.

Nachdem auch MobIDS ausführlich beschrieben wurden, sind damit sämtliche Komponenten von SAM und auch die Zusammenhänge zwischen den Teilbereichen bekannt. Während sich die letzten vier Kapitel auf eine reine Beschreibung beschränkten, soll im letzten Kapitel nun der Versuch unternommen werden, die Korrektheit und Effizienz der Mechanismen nachzuweisen.

12. Analyse

Ziel dieses Kapitels ist es einerseits, die Funktionsfähigkeit der Komponenten von SAM zu belegen, andererseits Abschätzungen zu Performance und Overhead zu geben. Dabei kommen teilweise formale Methoden wie die BAN Logik (siehe Abschnitt 3.7) zum Einsatz, andererseits werden Komponenten implementiert und im Simulator getestet. Zunächst soll aber untersucht werden, ob die Komponenten von SAM auch tatsächlich die verschiedenen in Abschnitt 6.2 vorgestellten Angriffe komplett abdecken.

12.1. Abdeckung der Angriffsbäume

Angriffsbaum A

In Angriffsbaum A (Tabelle 6.1) wird Fall A.1.1 („Keine Weiterleitung von Routing-Daten“) entweder durch Sensoren von MobIDS erkannt (A.1.1.1 und A.1.1.2) oder durch das SDSR verhindert (A.1.1.3). Erkennt MobIDS eine Manipulation, wird der betreffende Knoten schlechter bewertet und nach einiger Zeit aus dem Netz ausgeschlossen.

Fall A.1.2 („Routing Daten/Topologie modifizieren“) wird durch SDSR verhindert. Im Fall von A.1.3 („Aus aktiver Route aussteigen“) kann der FEB Knoten A.1.3.2 nicht durchführen, da es, analog oben, dem MobIDS oder SDSR auffallen würde, wenn er nicht an der Route Discovery teilnimmt.

Im Fall von A.2 („Keine Weiterleitung von Datenpakete“) wird ebenfalls das MobIDS aktiv.

Damit kann das Ziel von Baum A nicht erreicht werden, weil alle Teilbäume verhindert werden. Egoistisches Verhalten ist somit nicht mehr möglich oder wird durch das MobIDS zumindest auf ein unschädliches Maß eingedämmt.

Angriffsbaum B

Angriffsbaum B (Tabellen 6.2 und 6.3) enthält mit B.1.1 („Angriff auf die Funkschnittstelle“) eine Form des Angriffs, welche SAM nicht abdeckt. Generell lassen sich Angriffe auf die physikalische Übertragungsschicht innerhalb eines MANETs nur schwer abfangen. Hier muss vielmehr die Störfestigkeit der verwendeten Trägertechnologie (z.B. IEEE 802.11b) entsprechend verbessert werden. Einen Ansatz dazu liefert das Ultra Wide Band Verfahren [CS02]. Dies liegt aber außerhalb des Bereichs dieser Arbeit und soll hier nicht weiter vertieft werden.

Ein weiterer Aspekt ist das Überlasten von Knoten (B.2). Das Überlasten direkter Nachbarn durch eine Flut beliebiger Pakete (B.2.1) lässt sich auch kaum verhindern,

da ein Knoten ein Paket zunächst empfangen muss, bevor er darauf reagieren kann. Bereits das Empfangen kann aber zu einer Überlast führen. Auch hier sind natürlich Modifikationen im Radioempfänger denkbar, die beispielsweise nur stochastisch einzelne Pakete empfangen und auf einen Angriff hin auswerten. Auch dies wird hier jedoch nicht weiter betrachtet.

Relevant für diese Arbeit sind wieder Angriffe, welche sich die Netzwerkstruktur zu nutze machen, um andere Knoten zu überlasten (B.2.2). Sendet ein Knoten eine Vielzahl von Paketen ins Netz (egal ob Routing-Protokoll (B.2.2.1, B.2.2.2) oder Datenpakete (B.2.2.3)), so kann dies durch Identifizierung der Knoten eindeutig einem Verursacher zugeordnet werden. Dieser kann dann durch das MobIDS ausgeschlossen werden.

Punkt B.2.2.2.3 geht davon aus, dass ein Knoten eine große Menge normaler Datenpakete generiert, welche zwar nicht einen Nachbarknoten über- (B.2.1), aber das Netz als solches belasten. Ein solcher Angriff kann kaum vom normalen Verhalten eines Knotens unterschieden werden, da der Datenstrom ja auch eine sinnvolle Übertragung z.B. von Videodaten darstellen kann und das Netz dieser Menge nicht gewachsen ist. Hier wäre es Aufgabe entsprechender QoS Mechanismen, ein Traffic Shaping durchzuführen und gezielt Pakete zu verwerfen.

Die Punkte B.2.2.4 bis B.2.2.7 setzen eine Veränderung der korrekten Topologie voraus. Diese kann durch das SDSR Protokoll erkannt und verhindert werden. Ähnliches gilt für B.3 („Korrekte Routingfunktion stören“). SDSR verhindert, dass unter B.3.1.1 ein Knoten Pakete im Namen eines anderen Knotens verschickt. Das Verwerfen von RERR Meldungen (B.3.1.2) genau wie das Verwerfen von Datenpaketen (B.3.1.3.2) kann das MobIDS erkennen. Auch kann ein Knoten keine Routen über sich Umlenken (B.3.1.3.1), wenn er nicht wirklich im optimalen Pfad sitzt. Eine Verkürzung der Route verhindert SDSR. Die Punkte B.3.1.4, B.3.1.5 und B.3.1.6 gehen von Veränderungen in den Routing-Paketen aus, was wiederum SDSR verhindert.

Ein Problem stellt B.3.2.1 dar. Hier nehmen Knoten Informationen in ihre Routingtabellen auf, die sie aus der Kommunikation zwischen anderen Knoten zufällig mitgehört haben (sog. „Overhearing“). Da eine saubere Authentifizierung dieser Daten nur schwer machbar ist, verzichtet SDSR auf diese Optimierung. Die Funktionsfähigkeit des Routing Protokolls wird dadurch nicht beeinträchtigt.

Mit Ausnahme von direkten Angriffen auf die physikalische Funkschnittstelle und von Überlast-Denial-of-Service Attacken auf benachbarte Knoten deckt das Framework somit auch alle Angriffsformen aus Baum B ab. Ein Angriff auf die Funktionsfähigkeit des Netzes ist somit kaum noch möglich.

Angriffsbaum C

Angriffsbaum C beschäftigt sich mit dem Abhören von Datenpaketen. Da SDSR zwischen Quell- und Zielknoten einen gemeinsamen geheimen Schlüssel vereinbart und alle nachfolgende Kommunikation grundsätzlich verschlüsselt wird, sind abgehörte Datenpakete für einen Angreifer wertlos, da er deren Inhalt ohne Kenntnis des Schlüssels nicht rekonstruieren kann. Somit kann das Angriffsziel nicht erreicht werden.

Auch die Umleitung von Datenpaketen in den eigenen Knoten (C.2) verhindert SDSR, da es die dazu notwendigen Fälschungen verhindert. Lediglich der Wurmloch Angriff

(C.2.2.2), manchmal auch als Tunnel Angriff bezeichnet, bei dem zwei kooperierende Knoten Daten durch einen Tunnel austauschen und somit eine direkte Verbindung vortäuschen, wird nicht entdeckt. Wie die wenigen Arbeiten zu diesem Thema zeigen, ist dies ein grundsätzliches Problem, was vermutlich nicht zufriedenstellend gelöst werden kann [HPJ03]. Der Schadeffekt einer solchen Tunnel-Attacke dürfte sich aber (insbesondere vor dem Hintergrund verschlüsselter Verbindungen) in Grenzen halten.

Angriffsbaum D

Das Ziel bei Angriffsbaum D ist die Gewinnung von Informationen über einen Benutzer. Das kann zum Beispiel die Erstellung eines Bewegungsprofils sein (D.1). Hierzu muss die Bewegung eines Knotens erfasst werden (D.1.1) **und** es muss die Identität des Knotens erfasst werden (D.1.2 bzw. daraus folgend D.2). Da unser System die Verwendung mehrerer wechselnder Pseudonyme unterstützt, ist D.2 deutlich erschwert. Zumindest auf Ebene der Routingprotokolle ist keine Zuordnung möglich.

Alle weiteren Daten, welche zur Identifizierung eines Knotens geeignet sind (bspw. Anmeldedaten bei einem Diensteserver) werden grundsätzlich nur verschlüsselt übertragen und stehen einem Angreifer somit nicht zur Verfügung. Durch die Verschlüsselung werden D.2.1 und D.3 verhindert. Ohne eine Zuordnung der Pseudonyme zu einem Knoten bleiben die erstellten Bewegungsprofile wertlos. Einziger Ansatzpunkt ist D.2.2, hier ist noch ein Angriff auf den CA-Server denkbar. Dieser Server (der selbst nicht Bestandteil des MANETs ist) muss also hinreichend gesichert sein.

Somit können auch die Angriffsmöglichkeiten aus Baum D ausgeschlossen werden.

12.2. MANET-IDs

Es folgt die Analyse der einzelnen Komponenten von SAM. Das Vorgehen ist dabei immer gleich. Zunächst wird untersucht, ob die Komponente ihre Funktionen fehlerfrei erfüllt. Als nächstes ist zu prüfen, inwieweit verbleibende Angriffsmöglichkeiten das System gefährden. Schließlich werden Aufwand und Effizienz der Lösung abgeschätzt.

12.2.1. Funktionsfähigkeit

Der Grundgedanke des MANET-ID Systems basiert auf einer traditionellen PKI, bei der ein CA Server Zertifikate erstellt. Dieses Konzept ist lange etabliert, so dass sich ein formaler Nachweis seiner Funktionsfähigkeit erübrigt. Da bei MANET-IDs keine Namen an Schlüssel gebunden werden, sondern lediglich die Gültigkeit eines Schlüssel bestätigt wird, entfallen zudem alle Fragestellungen, die sich mit der Identifizierung von Personen beschäftigen.

Bei MANET-IDs wurde das PKI Konzept für den Einsatz in Mobilen Ad hoc Netzen angepasst. Die grundlegenden Sicherheitseigenschaften blieben dabei aber erhalten. Lediglich der Zertifikatsrückruf über Verifikatoren stellt in diesem Kontext ein neuartiges Konzept dar.

Deren Funktionsfähigkeit ist intuitiv klar. Die grundlegende Eigenschaft einer Hashchain ist es, dass man bei Kenntnis eines gegebenen Elements $H^n(x)$ zwar jederzeit spätere Elemente $H^{n+i}(x)$ berechnen, jedoch keinesfalls auf ein früheres Element $H^{n-i}(x)$ schließen kann.

Bei den MANET-IDs wird die Verifikator-Hashchain vom Ende zum Start durchlaufen. Entsprechend können Knoten bei Kenntnis eines Verifikators V_t zwar alle früheren Verifikatoren V_{t-i} berechnen, den nächsten Verifikator V_{t+1} kann aber ausschließlich die Zertifizierungsstelle ausgeben, da nur sie den Startwert Y_0 und somit den Anfang der Hashchain kennt. Damit ist das Ziel der Verifikatoren bereits erreicht: indem die CA keine neuen Verifikatoren mehr ausgibt, kann ein Zertifikat innerhalb weniger Intervalle als ungültig markiert werden. Entsprechende Überlegungen finden sich auch in [Mic96].

Im Rahmen von [Spe03] wurde die Funktionsfähigkeit des Konzepts durch eine konkrete Implementierung unter Verwendung von Java [Sunc] und der *Java Cryptography Extension* (JCE) [Sunb] verifiziert.

12.2.2. Mögliche Angriffe

Das System hat zwei mögliche Schwachstellen, an denen ein erfolgreicher Angriff ansetzen kann. Gelingt ein Einbruch in den CA Server, dann ist – wie bei jedem PKI System – die Sicherheit nicht mehr gewährleistet, da ein Angreifer dann beliebige Verifikatoren und Zertifikate erzeugen kann. Der CA Server ist also hinreichend abzusichern.

Alternativ kann ein Angreifer auch die Zeitsynchronisation attackieren. Gelingt es einem Angreifer, die Uhren von MANET Teilnehmern in die Vergangenheit zu stellen, so werden diese auch veraltete Verifikatoren und Zertifikate akzeptieren. Umgekehrt hätten Uhren, welche in die Zukunft gestellt werden, den Effekt, dass diese Knoten aktuell gültige Zertifikate und Verifikatoren nicht akzeptieren. Die Synchronisierung der Uhren soll daher bei Kontakt mit der CA in einer authentisierten und integritätsgesicherten Verbindung erfolgen, was wegen der vorhandenen Public-Keys kein Problem darstellt.

Bei gewissenhafter Umsetzung der Implementierung von MANET-IDs erscheinen beide Angriffe hinreichend unwahrscheinlich. Weitere theoretische Attacken setzen das Versagen eines kryptographischen Mechanismus voraus[1] und sind somit ebenfalls höchst unwahrscheinlich[2].

Es bleibt noch zu untersuchen, inwieweit das System für Denial of Service Angriffe anfällig ist. Zunächst könnte ein Angreifer im Netz versuchen, die Kommunikation mit der CA zu stören. Läuft diese Kommunikation nicht über das MANET, sondern *out-of-band* z.B. über eine alternative Internet-Verbindung, dann dürfte eine solche Störung recht schwierig zu realisieren sein. Läuft die Kommunikation hingegen über das MANET, kann ein Angreifer Datenpakete an die CA verwerfen oder modifizieren. Dem stellt SAM eine Reihe von Hindernissen in den Weg. Zunächst muss ein Angreifer dafür sorgen, dass keine Pakete an die CA über alternative Routen transportiert werden. Ein Umleiten des Verkehrs zum Angreifer verhindert aber SDSR. Ferner kann ein MobIDS Sensor das Verwerfen oder die Modifikation von Paketen bemerken. In der

[1] z.B. inverse Berechnung von Hashfunktionen, Berechnung eines geheimen RSA-Schlüssels etc.
[2] korrekte Implementierung vorausgesetzt

Folge würde der Angreifer aus dem MANET ausgeschlossen und könnte seinen Angriff nicht fortsetzen.

Ein weiterer DoS Angriff besteht darin, dass ein Knoten Anschuldigungen an die CA schickt. Da diese signiert sind, kann er keine gefälschten Meldungen verschicken. Da erst eine hinreichend große Anzahl von Meldungen *verschiedener* Knoten zur Sperrung einer MANET-ID führt, bleibt also auch dieser Angriff ohne Wirkung.

Schließlich ist ein expliziter DoS auf den CA Server denkbar. Da dieser über genügend Ressourcen und eine leistungsstarke Internet-Anbindung verfügen sollte, ist ein erfolgreicher DoS Angriff zumindest stark erschwert.

Entsprechend dieser Überlegungen kann das MANET-ID System als zumindest hinreichend sicher bezeichnet werden.

12.2.3. Aufwand und Effizienz

Als letztes soll die Frage geklärt werden, welchen Overhead die MANET-IDs erzeugen. Zunächst folgen einige Abschätzungen über die entstehende Datenmenge. Die folgende Tabelle zeigt die einzelnen Felder eines Zertifikats sowie eine Abschätzung ihrer jeweiligen Größe.

PK_K	1024 Bit	Öffentlicher Schlüssel, z. B. RSA oder ElGamal
Y	160 Bit	Letzter Hashwert (SHA-1) der Verifikator-Hashchain
cert_serial_number	32 Bit	Seriennummer des Zertifikats
valid_until	16 Bit	Gültigkeitsende des Zertifikats, z. B. #Tage seit 1.1.2003
Signatur	1024 Bit	Digitale Signatur der SHA-1-gehashten Felder von oben
gesamt	2256 Bit	$\widehat{=}$ 282 Byte

Wie man sieht, hat das Zertifikat eine Länge von etwa 282 Byte, wobei diese maßgeblich von der Schlüssellänge des RSA Schlüssels abhängt. Welche Datenmenge muss nun der CA Server speichern?

$cert_K$	2256 Bit	Zertifikat von Knoten K
Y_0	160 Bit	Start der Verifikator-Hashchain
incident_report	80 Bit	Meldung eines Fehlverhaltens (beinhaltet Sender und Datum)
$\times 100$	800 Bit	Kapazität für 100 Meldungen pro Knoten
pro Knoten	402 Byte	Datenmenge pro Knoten
$\times$ 80 Mio. Geräte	$\approx$ 30 GByte	

Bei geschätzten 80 Mio. Geräten (eines pro Bundesbürger) ist also eine Datenbank von 30 GByte notwendig. Für heutige Serversysteme sollte dies kein Problem darstellen, selbst wenn die Zahl der Geräte noch deutlich darüber wächst.

Interessanter ist vielleicht die Frage, wieviele Daten der MANET Knoten speichern muss? Zunächst das eigene Zertifikat plus den privaten Schlüssel und den aktuellen Verifikator. Weiterhin pro Kommunikationspartner ein Zertifikat plus den aktuellen

Verifikator und einen geheimen Sitzungsschlüssel. Nachfolgende Tabelle geht von etwa 100 Kommunikationspartner in einem MANET aus.

$cert_K$	2256 Bit	Eigenes Zertifikat
SK_K	1024 Bit	Geheimer Schlüssel zu PK_K
V_K	160 Bit	Eigener Verifikator
$cert_{K_i}$	2256 Bit	Zertifikat eines fremden Knotens
V_{K_i}	160 Bit	Verifikator eines fremden Knotens
k_{KK_i}	128 Bit	gemeinsamer geheimer Schlüssel
$\times$ 100 Knoten	254400 Bit	
gesamt	$\approx$ 31 kByte	Datenmenge pro Knoten

Speichert ein mobiler Knoten also seine eigenen Schlüsseldaten sowie die von 100 anderen Knoten, so benötigt er dafür etwa 31 kByte. Abhängig von der Art der eingesetzten Geräte kann man darüber diskutieren, ob dies viel ist oder nicht. Für Notebooks mit Speicher im Mega- und Gigabyte Bereich sicher nicht. Auch PDAs haben heute typischerweise zwischen 32 und 64 MByte Speicher. Selbst bei Mobiltelefonen sind heute bereits 1 MByte üblich[3]. Die heute üblichen Geräte sind also durchweg für SAM geeignet. Lediglich im Bereich der Sensor Networks könnten die Anforderungen von SAM noch zu groß sein.

Als nächstes soll der Kommunikationsoverhead bei der Kommunikation mit der CA untersucht werden. Die folgende Tabelle beinhaltet alle auftretenden Kommunikationen.

Verifikator abfragen	senden	32 Bit	S.nr. des Zertifikats
	empfangen	160 Bit	1056 Bit bei Rückruf
	gesamt	192 Bit	$\widehat{=}$ 24 Byte
Melden von FEB-Knoten	senden	$64 + 1040nBit$	n gemeldete Knoten
	empfangen	64 + 1040n Bit	n gemeldete Knoten
	gesamt	$128 + 2080n$ Bit	$\widehat{=}$ 276 Byte bei einem gemeldeten Knoten
Zertifikat erneuern	senden	96 Bit	Zert.seriennr., 64 Bit Nonce
	empfangen	3280 Bit	Signierte Nachricht und Zertifikat
	gesamt	3376 Bit	$\widehat{=}$ 422 Byte

Der Verifikatorabruf ist unkritisch, wenn man bedenkt, dass pro Tag lediglich ein neuer Verifikator nötig wird. Die Meldung eines FEB Knotens an die CA ist zwar etwas umfangreicher, allerdings sollte das Netz auch dies verkraften. Bedenkt man, dass die Erneuerung eines Zertifikats einmal pro Jahr anfällt, sind auch 422 Byte nicht wirklich viel. Selbst über eine schmalbandig GPRS Verbindung lassen sich diese Datenmengen problemlos austauschen. Folglich stellt die Kommunikation mit der CA kein Hindernis für den Einsatz von MANET-IDs dar.

[3] z.B. Siemens S55: 1 MByte; Nokia 7650: 3,6 MByte; Sony Ericsson P800: 12 MByte

12.3. Pseudonyme

Die Pseudonym-Komponente ist eine Erweiterung der MANET-IDs. Entsprechend werden hier lediglich die Unterschiede zum vorigen Abschnitt betrachtet.

12.3.1. Funktionsfähigkeit

Der Korrektheitsbeweis der abgeleiteten Pseudonyme wurde bereits in Abschnitt 9.6 geführt. An dieser Stelle werden die CA-signierten Pseudonyme betrachtet, wie in Abschnitt 9.7 vorgestellt. Die Pseudonyme werden als vollwertige MANET-IDs generiert. Entsprechend ist die Funktionsfähigkeit im MANET gesichert.

Die spezifische Anforderung an Pseudonyme besteht darin, dass ein Pseudonym nicht der Hauptidentität bzw. anderen Pseudonymen derselben MANET-ID zugeordnet werden kann. Bei einem Klienten-generierten Schlüsselpaar sind die Pseudonyme eigenständige RSA Schlüsselpaare und somit besteht grundsätzlich keine Möglichkeit der Zuordnung.

Bei gemeinsam generierten Schlüsseln wird der öffentliche Pseudonymschlüssel PK_A^i berechnet, indem $A_e^i \bmod \varphi(n)$ gebildet wird. Der private Schlüssel SK'_A wird dann passend zum öffentlichen Schlüssel berechnet. Um zu testen, ob ein Pseudonymschlüssel PK_A^i zu einer Hauptidentität gehört, müsste man den diskreten Logarithmus in $\mathbb{Z}_n$ berechnen können, um zu testen, ob $\log_A(PK_A^i) = x$ für eine natürliche Zahl x erfüllt ist. Weiß man zudem nicht, zu welcher Identität PK_x^i gehört, muss man dies für alle existierenden Identitäten durchführen. Außer dem CA Server gibt es aber nirgends eine Verzeichnis der bestehenden Identitäten. Ein Rückschluss von einem Pseudonym auf einen Schlüssel erscheint also nahezu unmöglich.

12.3.2. Mögliche Angriffe

Wie beschrieben ist der Versuch, mittels mathematischer Methoden aus einem Pseudonym eine Identität abzuleiten, wohl zum Scheitern verurteilt. Nichtsdestotrotz besteht auch hier die Gefahr, dass ein Angreifer erfolgreich die CA attackiert und über die dort hinterlegten Informationen Pseudonyme zuordnet. Die CA ist also entsprechend gut abzusichern, um solche Attacken auszuschließen.

Eine weitere Gefahr besteht darin, dass ein Angreifer verschiedene Pseudonyme über ihr Verhalten im MANET korreliert. Liegen dem Angreifer hinreichend viele Routeninformationen vor, so kann er unter Umständen schließen, dass zwei oder mehr Pseudo-MANET-IDs immer die gleichen Bewegungen ausführen und somit vermutlich zu einem Knoten gehören.

Derartige Vermutungen werden sich nie ganz ausschließen lassen. Durch geschickte Verwendung der Pseudonyme kann aber die Wahrscheinlichkeit einer richtigen Zuordnung reduziert werden. So kann ein Knoten für die reine Weiterleitung von fremdem Datenverkehr alternierend zwei unterschiedliche Pseudonyme verwenden, für die eigene Kommunikation mit anderen Knoten ein anderes Pseudonym (oder mehrere). Somit lassen sich Informationen, welche sich aus der Weiterleitung von Verkehr ergeben, im-

mer nur in kurzen Perioden korrelieren. Beim Wechsel des Pseudonyms entsteht dann ein Bruch, den ein Angreifer nur schwer überbrücken kann.

Die eigentliche Gefahr bei der Verwendung von Pseudonymen liegt weniger in einem erfolgreichen Angriff gegen das Pseudonymsystem. Wie schon ausgeführt, können Pseudonyme die Effizienz der MobIDS Sensoren beeinträchtigen. Indem ein Knoten mehrere Identitäten verwendet, kann er mit jeder Identität gerade soviel Schaden anrichten, dass der Sensor nicht anspricht. Die Zahl gleichzeitig aktiver Pseudonyme muss also sorgfältig gegen die notwendige Erkennungsleistung von MobIDS abgewogen werden. Da diese maßgeblich von der Abschreckungswirkung auf die Benutzer abhängt, können hier erst ausführliche Erfahrungen im realen Einsatz zu einer sinnvollen Wahl der Designparameter führen. In jedem Fall erlaubt SAM die flexible Einstellung einer Balance zwischen Schutz der Privatsphäre auf der einen und Schutz des MANETs auf der anderen Seite.

12.3.3. Aufwand und Effizienz

Analog zum Abschnitt über MANET-IDs soll auch hier eine Aufwandsabschätzung für Pseudonyme erfolgen. Zunächst gilt auch hier, dass sich die Pseudonyme im Einsatz analog verhalten wie MANET-IDs. Die dort aufgeführten Zahlen gelten also allesamt weiter, lediglich werden statt einer einzelnen Identität nun zusätzlich n Pseudo-MANET-IDs ausgegeben. Da n, wie oben beschrieben, relativ klein gehalten werden muss[4], gelten die Aussagen für MANET-IDs im Wesentlichen unverändert weiter.

Zusätzlicher Aufwand entsteht vor allem beim Speichern der Pseudonyme. Statt ca. 30 kByte können so beim MANET Knoten etwa 120 kByte Platz zum Cachen der öffentlichen Schlüssel nötig werden. Folgen die Knoten der Empfehlung, lediglich ein oder zwei MANET-IDs für das Routing und die reine Weiterleitung zu verwenden, reduziert sich der Aufwand wieder etwas.

Problematisch ist, dass sich beim Einsatz vieler Pseudonyme zwangsläufig auch die Zahl der Route Discoveries erhöhen muss. Dies wird im nächsten Abschnitt untersucht.

12.4. SDSR

Die Funktionsfähigkeit des SDSR Protokolls wird zunächst formal analysiert. Hierzu kommt die in Abschnitt 3.7 vorgestellte BAN Logik zum Einsatz. Die Untersuchungen beschränken sich dabei aus Gründen der Übersichtlichkeit auf die einfache Version des Protokolls. Da die erweiterte Schlüsselvereinbarung die gleichen Mechanismen nur in umgekehrter Richtung nutzt, lassen sich die Ausführungen zur einfachen Version analog auf das erweiterte Protokoll anwenden. Aus Platzgründen ist dies hier nicht ausgeführt.

[4]Sinnvoll erscheinen etwa 3–4 Pseudonyme gleichzeitig.

12.4.1. Funktionsfähigkeit

Beim Routingprotokoll SDSR funktioniert die Routenfindung vollkommen analog zu DSR, auch wenn diverse Optimierungen wegfallen mussten. Dass SDSR als Routingprotokoll korrekt funktioniert, wird durch die später folgenden Simulationen gezeigt. Insbesondere fällt die Performance nicht wesentlich hinter DSR zurück.

Im Folgenden soll mit Hilfe der in Kapitel 3.7 beschriebenen BAN Logik analysiert werden, ob die folgenden Sicherheitsziele von SDSR erreicht werden:

1. Wenn eine Route erfolgreich zu Stande kommt, muss die Identität aller Knoten in der Source-Route authentifiziert sein.
2. Nach erfolgreicher Route Discovery besitzt der Initiator je einen gemeinsamen geheimen Schlüssel mit jedem anderen Knoten in der Source Route.

Dabei ist die Notation teilweise erweitert worden, da BAN ursprünglich nicht für die Kommunikation einer beliebigen Anzahl von Knoten innerhalb eines Protokoll-Durchlaufs entwickelt wurde. Außerdem mussten verschiedene Berechnungen – insbesondere von Hashfunktionen – in die Notation aufgenommen werden, um den Sinn der Abläufe zu erhalten. Die Formeln sind ausreichend kommentiert, so dass diese Änderungen verständlich werden.

Formalisierung von SDSR

Analog zum Vorgehen in Abschnitt 3.7.3 müssen zunächst die wesentlichen Bestandteile des SDSR Protokolls (siehe Kapitel 10) entsprechend der BAN Logik formalisiert werden. Seien hierzu $H_1, \ldots, H_n$ die Knoten entlang der Route. H_1 ist dabei der Initiator der Route Discovery, H_n deren Ziel. N_1 ist eine zufällige Nonce, generiert von H_1, die anderen $N_i \,|\, i = 2 \ldots n$ werden über eine Verschlüsselungsfunktion berechnet, wie im Protokoll beschrieben. ID ist eine eindeutige Route-Discovery ID. Y_i bezeichnet den öffentlichen Schlüssel von H_i im Diffie-Hellmann Protokoll. PK_i ist der öffentliche Schlüssel von H_i, PK_i^{-1} entsprechend der private Schlüssel. Weiterhin verfügt jeder Knoten noch über einen geheimen Schlüssel k_i, den nur er kennt. k_i^{-1} dient zur Entschlüsselung von mit k_i signierten Nachrichten.

Die Protokollschritte sind dann wie folgt:

1. $H_1 \rightarrow H_2 \quad : \quad N_1,\ \{ID,\ Y_1\}_{PK_1^{-1}}$
2. $H_i \rightarrow H_{i+1} \quad : \quad N_i = \{N_{i-1}\}_{k_i},\ \{ID,\ Y_1\}_{PK_1^{-1}}$
   $\forall\, i = 2 \ldots (n-1)$
3. $H_n \rightarrow H_{n-1} \quad : \quad \left\{(H_1, \ldots, H_n),\ \{ID,\ Y_1\}_{PK_1^{-1}}\right\}_{PK_n^{-1}},\ N_{n-1},\ (Y_n),$
   $\left(\left\{h(k_{(1,n)})\right\}_{PK_n^{-1}}\right)$
4. $H_i \rightarrow H_{i-1} \quad : \quad \left\{(H_1, \ldots, H_n),\ \{ID,\ Y_1\}_{PK_1^{-1}}\right\}_{PK_n^{-1}},\ N_{i-1} = \{N_i\}_{k_i^{-1}},$
   $\forall\, i = (n-1) \ldots 2 \qquad (Y_n, \ldots, Y_i),\ \left(\left\{h(k_{(1,j)})\right\}_{PK_j^{-1}}\ \forall\, j = n \ldots i\right)$

Schritt 1 beschreibt den Versand des Route Requests von Knoten H_1 an Knoten H_2.

Schritt 2 beschreibt das sukzessive Weiterleiten des Route Requests von H_2 an H_3, H_3 an H_4 usw. bis zu H_n. Jeder Knoten i generiert dabei vor dem Versand der Nachricht

eine neue Nonce N_i, indem er die Nonce seines Vorgängers mit seinem Schlüssel k_i verschlüsselt. Der Rest der Nachricht wird unverändert weitergeleitet.

Schritt 3 beschreibt den Versand des Route Replies von H_n an H_{n-1}. H_n sichert dabei die Source-Route $(H_1, \ldots, H_n)$, ID und Y_1 durch seine Signatur ab. Weiterhin enthält die Nachricht seinen öffentlichen Schlüssel im Diffie-Hellmann Protokoll Y_n und einen Hashwert des zukünftigen gemeinsamen Schlüssels $k_{(1,n)}$, welcher von H_n signiert wurde.

In Schritt 4 wird nun diese Nachricht entlang der Source Route zu Knoten 1 zurück geschickt. Dabei nimmt jeder Zwischenknoten H_i eine Reihe von Modifikationen vor. Zunächst macht er seine Verschlüsselung der Nonce rückgängig, generiert also N_{i-1}. Außerdem fügt er seinen öffentlichen Diffie-Hellmann Schlüssel Y_i und den signierten Hashwert des gemeinsamen Schlüssels $k_{(1,i)}$ zu den jeweiligen Listen hinzu.

Damit ist die Formalisierung des Protokolls abgeschlossen. Im nächsten Schritt müssen nun die Anfangsbedingungen definiert werden.

Anfangsbedingungen

Es wird von folgenden Anfangsbedingungen ausgegangen:

$H_1 \models\Rightarrow N_1$	:	H_1 erzeugt die Nonce N_1 in jedem Ablauf neu und zufällig.
$H_1 \models\Rightarrow ID$	:	H_1 erzeugt die RREQ ID in jedem Ablauf neu.
$\overset{Y_i}{\mapsto} H_i$	:	Jeder Knoten H_i besitzt ein Schlüsselpaar für das
$\forall\, i = 1 \ldots n$		DH-Protokoll (geheim: y_i, öffentlich: Y_i)
$H_i \models \sharp(y_i, Y_i)$	:	Jeder Knoten H_i erzeugt das DH-Schlüsselpaar für jeden
$\forall\, i = 1 \ldots n$		Protokolldurchlauf neu und zufällig.
$H_i \models H_i \overset{k_i}{\leftrightarrow} H_i$	:	Jeder Zwischenknoten besitzt einen geheimen
$\forall\, i = 2 \ldots (n-1)$		Schlüssel k_i, den nur er kennt.
$\overset{PK_i}{\mapsto} H_i$	:	Jeder Knoten verfügt über ein MANET-ID Schlüsselpaar.
$\forall\, i = 1 \ldots n$		
$H_i \models \overset{PK_j}{\mapsto} H_j$	:	Jeder Knoten kennt die Public Keys der anderen Knoten
$\forall\, i, j = 1 \ldots n$		und kann diese verifizieren.

Zur letzten Anfangsbedingung noch ein kurzer Hinweis: SDSR verteilt die Public-Key Schlüssel der Teilnehmer zwar erst im Laufe des Protokolls, im Sinne einer Idealisierung und Vereinfachung für BAN wird jedoch davon ausgegangen, dass die öffentlichen Schlüssel bereits vor dem Protokollablauf zur Verfügung stehen. Die Korrektheit der öffentlichen Schlüssel können die Knoten über die enthaltenen Zertifikate sicherstellen. Details hierzu siehe Kapitel 8.

Gemäß Formel 3.10 folgt aus den Anfangsbedingungen unmittelbar:

$H_1 \models \sharp N_1$	:	Da H_1 die Nonce erzeugt, ist er auch von deren Frische überzeugt.
$H_1 \models \sharp ID$	:	Da H_1 ID erzeugt, ist er auch von deren Frische überzeugt.

Nachbedingungen

Aus jedem der Protokollschritte ergeben sich bestimmte Folgerungen:

$$\begin{aligned}
&\text{Schritt 1:} && H_2 \lhd N_1,\ \{ID,\ Y_1\}_{PK_1^{-1}} \overset{(3.1)}{\Rightarrow} H_2 \models H_1 \mid\sim ID, Y_1 \\
&\text{Schritt 2:} && H_{i+1} \lhd N_i,\ \{ID,\ Y_1\}_{PK_1^{-1}} \overset{(3.1)}{\Rightarrow} H_{i+1} \models H_1 \mid\sim ID, Y_1 \\
& && \forall i = 2 \ldots (n-1) \\
&\text{Schritt 3:} && H_{n-1} \lhd \left\{ (H_1, \ldots, H_n),\ \{ID,\ Y_1\}_{PK_1^{-1}} \right\}_{PK_n^{-1}}, \\
& && \qquad N_{n-1},\ (Y_n),\ (\{h(k_{(1,n)})\}_{PK_n^{-1}}) \\
& && \overset{(3.1)}{\Rightarrow} H_{n-1} \models H_n \mid\sim (H_1, \ldots, H_n),\ \{ID,\ Y_1\}_{PK_1^{-1}} \quad \text{sowie} \\
& && \overset{(3.1)}{\Rightarrow} H_{n-1} \models H_1 \mid\sim ID,\ Y_1 \quad \text{sowie} \\
& && \overset{(3.1)}{\Rightarrow} H_{n-1} \models H_n \mid\sim h(k_{(1,n)}) \\
&\text{Schritt 4:} && H_{i-1} \lhd \left\{ (H_1, \ldots, H_n),\ \{ID,\ Y_1\}_{PK_1^{-1}} \right\}_{PK_n^{-1}},\ N_{i-1},\ (Y_n, \ldots, Y_i), \\
& && \quad (\{h(k_{(1,j)})\}_{PK_j^{-1}}\ \forall j = n \ldots i) \quad \forall i = (n-1) \ldots 2 \\
& && \overset{(3.1)}{\Rightarrow} H_{i-1} \models H_n \mid\sim (H_1, \ldots, H_n),\ \{ID,\ Y_1\}_{PK_1^{-1}} \quad \text{sowie} \\
& && \overset{(3.1)}{\Rightarrow} H_{i-1} \models H_1 \mid\sim ID,\ Y_1 \quad \text{sowie} \\
& && \overset{(3.1)}{\Rightarrow} H_{i-1} \models H_j \mid\sim h(k_{(1,j)})\ \forall j = n \ldots i
\end{aligned}$$

Schlussregeln anwenden

Jetzt können aus den Anfangs- und Nachbedingungen durch Anwendung der Schlussregeln entsprechende Folgerungen gezogen werden:

$$H_1 \models \sharp(ID) \wedge H_1 \models H_n \mid\sim (H_1, \ldots, H_n),\ \{ID,\ Y_1\}_{PK_1^{-1}} \overset{(3.5)(3.2)}{\Longrightarrow} H_1 \models H_n \models (H_1, \ldots, H_n)$$

Nach Ablauf des Protokolls ist sich H_1 also sicher, dass H_n seinen aktuellen Route-Request tatsächlich mit der Source-Route $(H_1, \ldots, H_n)$ beantwortet hat. Allerdings bleibt noch die Gefahr, dass ein Zwischenknoten die Route auf dem Hinweg zu H_n manipuliert hat. Im Folgenden wird gezeigt, dass H_1 auch dies ausschließen kann.

$$H_1 \lhd N_1 \wedge H_1 \models \sharp(N_1) \wedge H_i \models H_i \overset{k_i}{\leftrightarrow} H_i \Rightarrow \left\{ \left\{ \left\{ \left\{ \{\{N_1\}_{k_2}\}_{k_3} \ldots \right\}_{k_{n-1}} \right\}_{k_{n-1}^{-1}} \ldots \right\}_{k_3^{-1}} \right\}_{k_2^{-1}}$$

Wenn H_1 am Ende der Route Discovery wieder die Nonce N_1 empfängt, dann heißt das, dass die Ver- und Entschlüsselungen mit den geheimen Schlüsseln k_i der Zwischenknoten genau symmetrisch erfolgt ist. Solange jeder Zwischenknoten seinen Schlüssel geheim hält, lässt sich hieraus folgern, dass auf Hin- und Rückweg die Zwischenknoten in exakt der gleichen Reihenfolge durchschritten wurden. Diese Reihenfolge ist in der im Route Reply angegebenen Source-Route festgehalten.

Allerdings lässt sich noch nichts über die Identität der Zwischenknoten aussagen, da noch keine verlässliche Authentifizierung stattgefunden hat. Diese führt H_1 durch,

indem er prüft, ob die aus dem DH-Schlüsselaustausch gebildeten Schlüssel dem von H_i signierten Hashwert entsprechen. Sei y_i der geheime Schlüssel im Diffie-Hellmann Protokoll (passend zum öffentlichen Teil Y_i). Dann gilt $\forall\, i = 2 \ldots (n-1)$:

$\mathbf{H_1}$: $\mathbf{H_i}$:

$$H_1 \models \sharp(y_1) \ \wedge \ H_1 \models H_i \mid\sim h(k_{(1,i)}) \qquad H_i \models \sharp(y_i) \ \wedge \ H_i \models H_1 \mid\sim Y_1$$

$$\overset{DH,(3.11)}{\Rightarrow} H_1 \models \sharp(k_{(1,i)}) \qquad \overset{DH}{\Rightarrow} H_i \models \sharp(k_{(1,i)})$$

$$\Rightarrow H_1 \models H_1 \overset{k_{(1,i)}}{\leftrightarrow} H_i \ \wedge \ H_i \models H_1 \overset{k_{(1,i)}}{\leftrightarrow} H_i$$

Da H_1 mit y_1 einen frischen Teil zum Schlüssel beiträgt, kann er sicher sein, dass dieser im Falle eines korrekten Diffie-Hellmann-Schlüsselaustauschs auch frisch ist. Sobald H_1 den gemeinsamen Schlüssel $k_{(1,i)}$ berechnet hat, kann er ferner über den signierten Hashwert prüfen, ob der Partner im DH-Protokoll tatsächlich H_i war und ob dieser den gleichen Schlüsselwert berechnet hat, wie er selbst.

Auch H_i trägt mit y_i einen frischen Teil zum Schlüssel bei, entsprechend ist auch er sicher, dass $k_{(1,i)}$ frisch sein wird. Weiter kann er sich sicher sein, im aktuellen Protokolldurchlauf tatsächlich ein – nicht notwendigerweise frisches – Y_1 von Knoten H_1 erhalten zu haben. Wäre Y_1 nicht frisch (also ein Replay einer alten Nachricht), so werden H_1 und H_i unterschiedliche $k_{(1,i)}$ berechnen, was H_1 bei der Schlüsselüberprüfung auffällt. Somit wird – wie bei allen sonstigen Manipulationen –H_1 den Schlüssel verwerfen, eine neue Route Discovery wird notwendig.

Es lässt sich also festhalten, dass sich beide Ziele des SDSR Protokolls mittels BAN Logik verifizieren lassen. H_1 hat mit jedem Knoten H_i ($\forall\, i = 2 \ldots n$) einen geheimen Schlüssel vereinbart und die Parteien haben sich gegenseitig authentifiziert. H_1 kann die Sourceroute und die geheimen Schlüssel verwenden, um sicher mit den anderen Knoten zu kommunizieren.

12.4.2. Mögliche Angriffe

Die Sicherheit des Protokolls ist mit dem vorigen Abschnitt nachgewiesen. Verbleibende Angriffe sind im Wesentlichen DoS Attacken. Die Möglichkeiten eines Angreifers beschränken sich auf einen der folgenden Fälle:

Modifikationen statischer Teil RREQ: Der statische Teil des RREQ ist durch eine Signatur des Senders S abgesichert. Wird hier eine beliebige Modifikation vorgenommen, so wird dies vom Empfänger D erkannt und dieser RREQ wird verworfen.

Modifikation der Source-Route im RREQ: Modifiziert ein Knoten die Source Route, so weichen Hinweg und Rückweg des Pakets voneinander ab. Dies wird S nach Erhalt des RREP beim Prüfen der Nonce feststellen und den RREP verwerfen.

Modifikation der Nonce im RREQ: Wird die Nonce manipuliert, so stimmen die Ver- und Entschlüsselungsvorgänge nicht mehr überein. S wird am Ende nicht mehr N_1 im RREP empfangen und das Paket verwerfen.

Verwerfen des RREQ: Verwirft ein Knoten einen RREQ anstatt in weiterzuleiten, so wird durch ihn keine Route zustande kommen. MobIDS kann dies mittels des *Route Request Scanning* Sensors erkennen und den FEB Knoten schlechter bewerten bzw. ausschließen.

Modifikation RREP, Verwerfen durch Zwischenknoten: Manipuliert ein Knoten den signierten Teil des RREP, so wird dies der nachfolgende Knoten feststellen und das Paket verwerfen bzw. einen RERR an S schicken. In jedem Fall wird diese Route nicht genutzt.

Modifikation RREP, Verwerfen durch Endknoten: Manipuliert einer der Zwischenknoten einen nicht-signierten Bestandteil des RREP[5], wird dies bei S auffallen, da entweder nicht N_1 ankommt oder einer der nach Diffie-Hellmann generierten Schlüssel nicht zur Signatur passt. Auch hier wird der RREP nicht akzeptiert und die Route kommt in dieser Form nicht zu Stande.

Wie man sieht beschränken sich alle Angriffe darauf, dass Pakete als modifiziert erkannt und/oder verworfen werden. Da die betreffende Route offensichtlich einen oder mehrere FEB Knoten beinhaltet, kann diese Route aber in keinem Fall verwendet werden. Ein Verwerfen der Route ist daher die angemessene Reaktion. Solange im MANET alternative Wege bestehen, werden diese automatisch verwendet. Ansonsten kann keine Verbindung über einen zuverlässigen Pfad aufgebaut werden.

12.4.3. Aufwand und Effizienz

Ziel der Effizienzabschätzung ist es primär, einen Vergleich zum DSR Protokoll zu ziehen, da dieses ja die Vorlage für SDSR war. Zu diesem Zweck wurde SDSR für den Netzwerksimulator ns2 implementiert[6]. Die Implementierung entspricht der Variante mit einfacher Schlüsselverteilung und ohne die genannten Optimierungen.

Interessant ist der Vergleich vor allem deshalb, weil man hier direkt sieht, welchen Overhead die Sicherheitsmechanismen verursachen. Vergleichswerte finden sich auch in [JLH+99, BMJ+98].

Das Simulationsszenario ist analog zu Abschnitt 6.3, die wichtigsten Einstellungen sind nochmals in der folgenden Tabelle zusammengefasst, die darauf folgenden Grafiken zeigen die Simulationsergebnisse. Pro Wert wurden jeweils zehn Simulationen mit unterschiedlichen Szenarien durchgeführt. Die Fehlerbalken zeigen jeweils die gemessene Standardabweichung der Werte.

[5] also die Nonce oder ein DH-Schlüsselelement
[6] Details zur Implementierung siehe [Gei03]

Parameter	Wert
Anzahl Knoten	50
Raumgröße X (m)	1500
Raumgröße Y (m)	300
Verkehrsmodell	cbr
Senderate (packets/s)	1.0
Zufalls-Initialisierung	1
Max. Zahl von Verbindungen	20
Paketgröße (byte)	512
Simulationszeit (s)	900

Abbildung 12.1 zeigt die Empfangsrate beider Protokolle unter verschiedenen Bedingungen, also wie viel Prozent der abgeschickten Pakete auch tatsächlich ankommen. Wie man sieht, schlägt sich SDSR nur geringfügig schlechter als DSR. Der Grund liegt vermutlich in der ausgefeilteren und besser eingestellten Implementierung von DSR. Diverse Timer und Verzögerungswerte sowie Puffergrößen beeinflussen das Ergebnis. Trotzdem ist die Aussage zulässig, dass DSR in der Performance SDSR entspricht. Da das Routingverfahren an sich nicht geändert wurde, verwundert dies nicht.

Die nächsten beiden Graphen in Abbildung 12.2 zeigen den Protokolloverhead oder genauer die Zahl der verschickten Protokollpakete[7]. Dabei wird hier jedes Routingpaket nur einmal gezählt, egal über wieviele Zwischenstationen es weitergeleitet wurde. Es überrascht, dass vor allem bei größerer Mobilität SDSR dem DSR Protokoll deutlich überlegen ist. Die Erklärung liegt in der Expanding Ring Search, einer Optimierung von DSR, welche wir bei SDSR (noch) nicht implementiert haben. Um ein Fluten des ganzen MANET mit RREQ Paketen zu verhindern, wird hierbei zunächst ein RREQ mit einem TTL Wert von 1 verschickt, welcher nur die unmittelbaren Nachbarn erreicht. Erfolgt hierauf keine Antwort, so wird die Routensuche mit den TTL Werten 2, 4, 8, 16 usw. fortgeführt, bis eine Antwort beim Sender ankommt. Dass hierbei mehr RREQ Pakete generiert werden, ist einsichtig.

Wie man in der Grafik deutlich sieht, hat eine höhere Mobilität vor allem bei DSR einen starken Anstieg an Route Requests zur Folge. Auch zeigen die Fehlerbalken, dass eine sehr starke Abhängigkeit vom Szenario File besteht. Ungünstige Bewegungsmuster können die Zahl der für das Routing benötigten Pakete massiv beeinflussen.

Allerdings sagt die absolute Zahl der generierten Protokollpakete nur bedingt etwas über die Netzbelastung aus. Vor allem während des Flutens des Netzes mit einem RREQ spielt es natürlich eine Rolle, wie oft die Pakete weitergeleitet wurden. Daher zählen die Grafiken in Abbildung 12.3 jedes Protokollpaket pro Hop. Wird also ein Route Reply über 3 Zwischenstationen weitergeleitet, so werden hier insgesamt 4 Pakete gezählt, während bei den vorherigen Graphen lediglich ein Paket registriert worden ist.

Hier sieht die Situation deutlich anders aus. Die Werte von SDSR liegen zunächst deutlich über denen von SDSR. Wie ist das zu erklären? Eine Analyse der Daten hat ergeben, dass bei den gemessenen Werten die Zahl der RREQs bei Weitem überwiegt. Durch die Expanding Ring Search kann also DSR die Zahl der weitergeleiteten Pakete wohl erfolgreich eindämmen. Allerdings genügt diese Erklärung alleine nicht, um den

[7] also RREQ, RREP und RERR Pakete

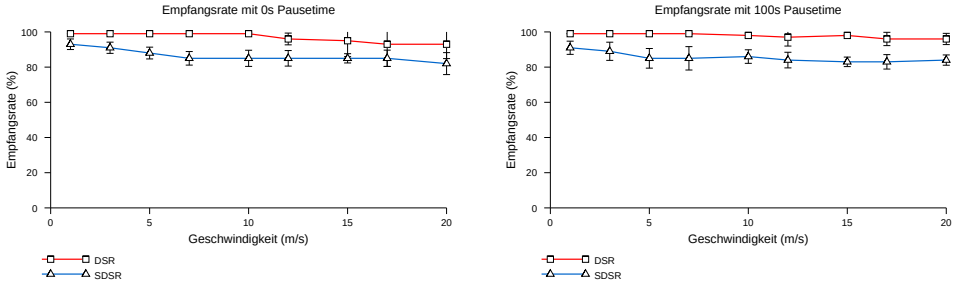

Abbildung 12.1.: Vergleich Empfangsrate DSR vs. SDSR

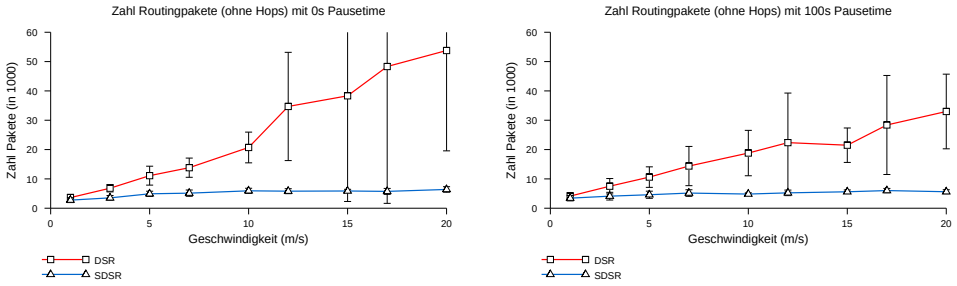

Abbildung 12.2.: Vergleich Overhead DSR vs. SDSR (jedes Routingpaket nur einfach gezählt)

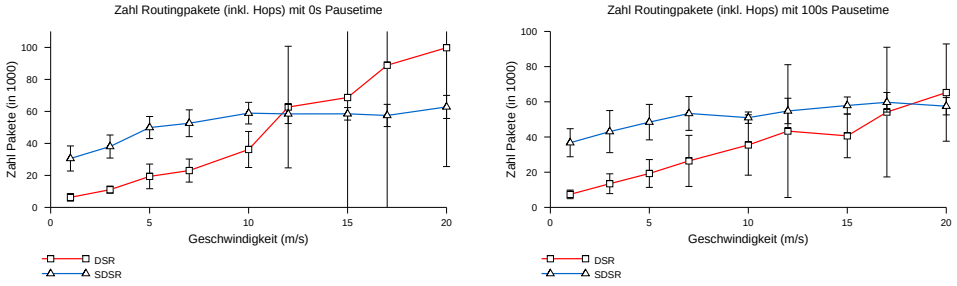

Abbildung 12.3.: Vergleich Overhead DSR vs. SDSR (jedes Routingpaket pro Hop gezählt)

großen Unterschied zu erklären. Dieser kommt erst durch eine weitere Optimierung zu Stande. Durch Route Caching können Zwischenknoten einen RREQ direkt beantworten, wenn sie bereits eine Route zum Ziel kennen. Der RREQ muss in diesem Fall nicht mehr weiter geflutet werden. Offensichtlich sind bei DSR nach einer gewissen Zeit genügend Cacheinformationen vorhanden, dass praktisch jeder RREQ bereits nach ein oder zwei Hops beantwortet werden kann.

Bei höherer Mobilität wird dieser Vorteil schnell zu einem Nachteil. Dann sind die gecachten Routinginformationen oft veraltet, wenn sie beim Sender des RREQ ankommen. Dieser benutzt die Route trotzdem, nur um sofort einen RERR zu erhalten. Daraufhin muss er wieder einen RREQ anstoßen. So werden im Endeffekt mehr Pakete generiert als bei SDSR ohne diese Optimierungen.

Vom Standpunkt der Sicherheit aus ist Route Caching generell sehr kritisch zu bewerten. Wie bereits in den Abschnitten 10.1.1 und 10.2.4 diskutiert wurde, eröffnen sich hier für einen Angreifer Möglichkeiten, veränderte oder veraltete Routen-Information in das Netz einzuspeisen. Deshalb verwendet SDSR vorläufig keine derartige Optimierung. In Kapitel 13 wird aber nochmals auf diesen Punkt eingegangen.

Neben der Zahl der Pakete ist auch deren Größe von Interesse. Im Folgenden werden die typischen Paketgrößen von RREQ, RREP und RERR bei DSR und SDSR dargestellt, allerdings ohne Berücksichtigung der Schlüsselverteilung.

Pakettyp	**DSR**	**SDSR**
RREQ		
– stat. Teil	$4 + L$ *Byte*	$268 + L$ *Byte*
– var. Teil	$n * L$ *Byte*	$n * L$ *Byte*
RREP		
– stat. Teil	4 *Byte*	268 *Byte*
– var. Teil	$n * L$ *Byte*	$n * (L + 256)$ *Byte*
RERR	$4 + 2L + x$ *Byte*	$132 + 2L + x$ *Byte*
Source Route H.	$n * L$ *Byte*	$n * L$ *Byte*

Die Werte für DSR sind direkt aus dem aktuellen Draft [JMHJ03] entnommen, die Werte für SDSR aus den Beschreibungen in Abschnitt 10.2 abgeleitet. Wo sinnvoll, wurde dabei zwischen dem statischen Teil eines Pakets und dem variablen Teil unterschieden, der mit der Zahl der Hops in der Source Route skaliert. L bezeichnet die Länge einer Adresse bzw. einer Identität. Während der DSR Draft hier noch mit IPv4 Adressen und somit 4 Bytes rechnet, gehen wir von IPv6 Adressen aus, bei denen (bei statischem Präfix) immer noch 8 Byte zu übertragen sind. n ist die Zahl der Zwischenknoten in einer Route.

Wie man sieht, ist der variable Teil des RREQ bei SDSR genauso groß wie bei DSR, weshalb die Skalierbarkeit auf große Netze der von DSR entspricht. Dagegen ist der statische Teil mit $268 + L$ Byte deutlich größer. Der zusätzliche Platz wird für den Diffie-Hellmann Key, die Nonce und die Signatur benötigt.

Im RREP ist der statische Platzbedarf jeweils identisch zum RREQ, wobei die Adressen wegfallen, die bereits im IP Header enthalten sind. Beim variablen Platzbedarf kommt bei SDSR allerdings pro Zwischenknoten noch jeweils eine Signatur von 256 Byte Länge hinzu. Ein RREP mit 5 Hops hat bereits eine Länge von ca. 1800 Byte und

überschreitet damit die MTU von IEEE 802.11. Das Paket muss also auf IP Ebene fragmentiert werden. Werden noch öffentliche MANET-ID Schlüssel beigefügt, wächst die Länge der Pakete entsprechend weiter an. Immerhin werden diese großen Paket nicht mehr im Netz geflutet, sondern als Unicast direkt von D zu S transportiert.

Wird mit Wiederverwertung von geheimen Schlüsseln gearbeitet, kann der variable Teil des RREP unter günstigen Umständen deutlich reduziert werden. Da dann keine DH-Schlüssel plus Signatur mehr übertragen werden müssen, ergibt sich ein Wert von $n * (L + 8)$ Byte, wenn von einer 8 Byte langen Nonce ausgegangen wird.

Die Größe eines RERR bzw. eines Source Routing Headers bei SDSR entspricht wieder (bis auf einen konstanten Wert) dem von DSR.

Es lässt sich also festhalten, dass SDSR signifikant mehr Overhead erzeugt als DSR. Dies liegt zum Einen an fehlenden Optimierungen während der RREQ Phase, was die Zahl der Pakete erhöht, zum anderen auch an den kryptographischen Daten, die zu transportieren sind. Wird mit 1024 Bit langen RSA Schlüsseln gearbeitet, so ist jeder signierte Wert ebenfalls 1024 Bit lang. Auch für die Diffie-Hellmann Schlüssel sind 1024 Bit große Werte heute üblich. Um eine sichere Authentifizierung und einen entsprechenden Schlüsselaustausch zu erreichen, sind diese Informationen zwingend erforderlich. Trotzdem bleibt festzuhalten, dass es SDSR gelingt, den Großteil des Overheads in den RREP zu verlagern, der nur noch als Unicast gesendet wird. Somit schlagen die hohen Paketzahlen im RREQ nicht so gravierend durch. Trotzdem wird das Thema Overhead von SDSR im Ausblick nochmals aufgegriffen, da hier weitere Optimierungen sinnvoll sind.

12.5. MobIDS

Eine formale Untersuchung des Intrusion Detection Systems ist aus nahe liegenden Gründen relativ schwierig. Da insbesondere die Sensoren eher den Charakter einer Heuristik haben, kann man deren Erfolg eigentlich nur im realen Einsatz messen oder mittels einer Simulation nachstellen. Im Rahmen dieser Arbeit wurde der zweite Weg beschritten. Die Simulation zeigt damit die Funktionsfähigkeit der Sensoren und von MobIDS und liefert gleichzeitig Aussagen über die Effizient des Systems.

12.5.1. Funktionsfähigkeit

Die Implementierung erfolgte, ebenso wie die SDSR Implementierung, für den ns2 [NS2] Simulator. Details zur Implementierung finden sich in [Kle03]. Als Simulationsszenario kamen wieder die Parameter zum Einsatz, wie sie in Tabelle 6.6 dargestellt sind. Pro Wert wurden wieder zehn Simulationen mit unterschiedlichen Szenarien durchgeführt.

Die Simulationen wurden mit DSR als Routingprotokoll durchgeführt, da dieses die stabilere und verlässlichere Basis verglichen mit SDSR darstellt. Der Prozess der Routenfindung ist bei beiden Protokollen prinzipiell identisch und die modellierten Angriffe sind so gewählt, dass SDSR sie auch nicht erkannt hätte. Somit sind die Sensorergebnisse auch auf SDSR übertragbar.

Sensoren

In Abbildung 12.4 (links) ist die Erkennungsleistung der verschiedenen *Overhearing Sensoren* aus Abschnitt 11.5.1 bei einer Knotengeschwindigkeit von 1 m/s gezeigt. Die x-Achse zeigt die absolute Zahl der FEB Knoten im Netz. Das Verhalten der Knoten entspricht dem Typ Egoistisch-2 wie in Abschnitt 6.3.1 beschrieben. Die y-Achse zeigt, wie oft jeder egoistische Knoten im Durchschnitt von Overhearing Sensoren anderer Knoten als egoistisch erkannt wurde. Damit stellt der absolute Wert ein vergleichbares Maß für die Erkennungsleistung dar. Dabei muss ein Sensor eines Knotens mehrfach anschlagen, bis tatsächlich eine Erkennung festgestellt wird.

Diese Metrik soll anhand eines Beispiels verdeutlicht werden. Bei zwei egoistischen Knoten erkennt das herkömmliche Promiscuous Overhearing jeden egoistischen Knoten im Schnitt auf 1,1 anderen Knoten. Insgesamt erkennen also 2,2 Knoten einen FEB Knoten. Wie man sieht ist das aktivitätsbasierte Overhearing bei wenigen egoistischen Knoten deutlich überlegen. So wird bei zwei egoistischen Knoten im MANET jeder dieser Knoten im Schnitt von 6,2 Knoten erkannt, insgesamt gibt es also 12,4 Erkennungen.

Wächst die Zahl der egoistischen Knoten, so sinkt die Erkennungsleistung merklich, d.h. jeder egoistische Knoten wird im Schnitt seltener erkannt. Dies liegt vor allem daran, dass in unserem Modell egoistische Knoten selbst nicht als Sensoren fungieren und somit nicht zur Erkennung beitragen. Diese Annahme erscheint sinnvoll, da ein egoistischer Knoten auch kein Interesse daran hat, Ressourcen in die Erkennung anderer Knoten zu investieren.

Wie man sieht ist bei 5 bis 10 egoistischen Knoten das herkömmliche Promiscuous Overhearing dem aktivitätsbasierten Overhearing leicht überlegen. Durch kombiniertes Overhearing kann man die Vorzüge beider Verfahren nutzen. Die rechte Seite von Abbildung 12.4 zeigt die falsch-positiven Erkennungen, d.h. wie oft wurde ein regulärer Knoten von einem anderen Knoten als egoistisch eingestuft. Diese Rate fällt von maximal 3 Fehlerkennungen schnell unter 1. Da jedoch für einen Ausschluss immer mehrere Alarme unterschiedlicher Knoten notwendig sind, sind einzelne Fehlalarm unproblematisch und wirken sich nicht aus.

Erhöht man die Bewegungsgeschwindigkeit auf 20 m/s, wie in Abbildung 12.5 gezeigt, so steigt die Erkennungsleistung des aktivitätsbasierten Overhearings deutlich an. Dies ist darauf zurückzuführen, dass mehr Route Discoveries durchgeführt werden und der Aktivitätssensor somit mehr Pakete von anderen Knoten empfängt. Damit kann er genauer feststellen, ob er einen anderen Knoten noch empfangen kann, was sich in der Erkennungsleistung niederschlägt. Indem man die Schwellwerte für eine Erkennung noch höher ansetzten kann, geht zudem die Zahl der falsch-positiven Erkennungen deutlich zurück.

Abbildung 12.6 zeigt links die Erkennungsleistung für das *Iterative Probing*. Der Graph ist dabei analog zu den Overhearing Graphen aufgebaut. Auch hier fällt die Erkennungsleistung mit der Zahl egoistischer Knoten ab. Bemerkenswert ist noch, dass die Erkennungsleistung bei höherer Mobilität steigt. Dies ist damit zu erklären, dass dann auch mehr Pfade gebildet werden. Somit ist die Wahrscheinlichkeit erhöht, dass ein egoistischer Knoten Bestandteil vieler Routen und somit öfter erkannt wird. Die Zahl

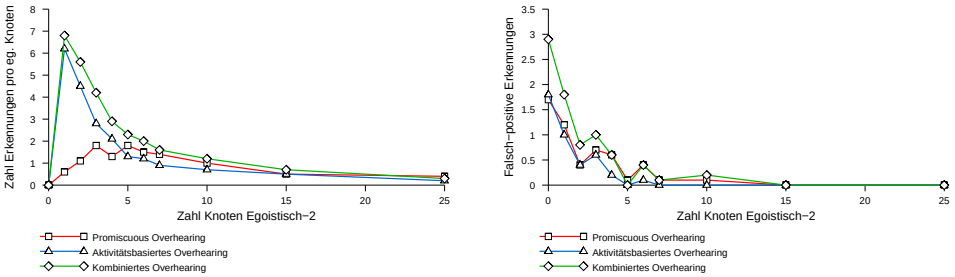

Abbildung 12.4.: Vergleich der Erkennungsleistung verschiedener Overhearing Sensoren bei 1 m/s

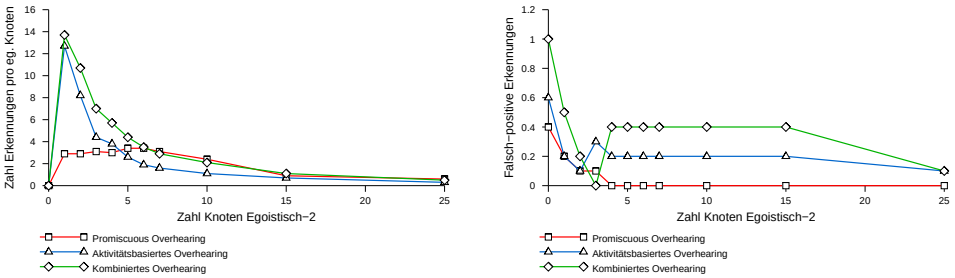

Abbildung 12.5.: Vergleich der Erkennungsleistung verschiedener Overhearing Sensoren bei 20 m/s

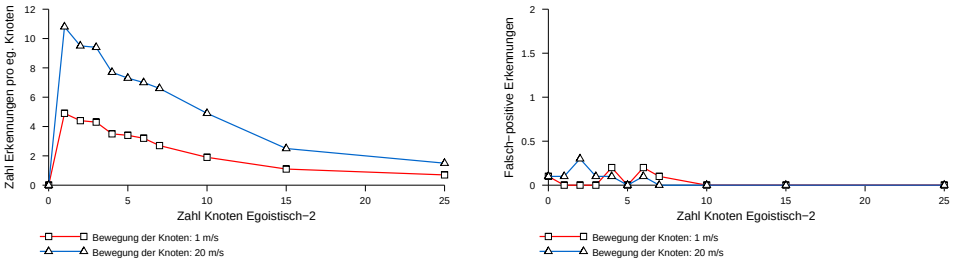

Abbildung 12.6.: Erkennungsleistung von iterativem Probing

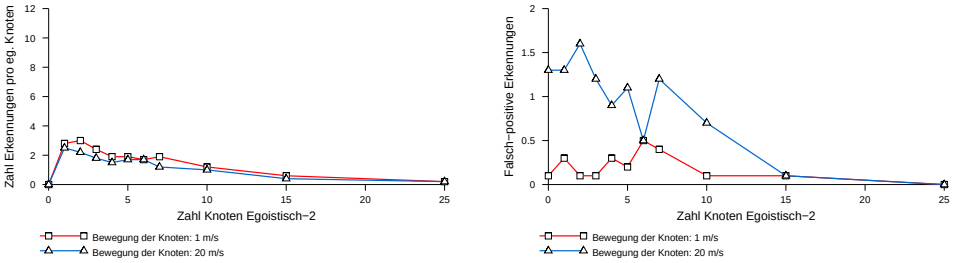

Abbildung 12.7.: Erkennungsleistung von eindeutigem Probing

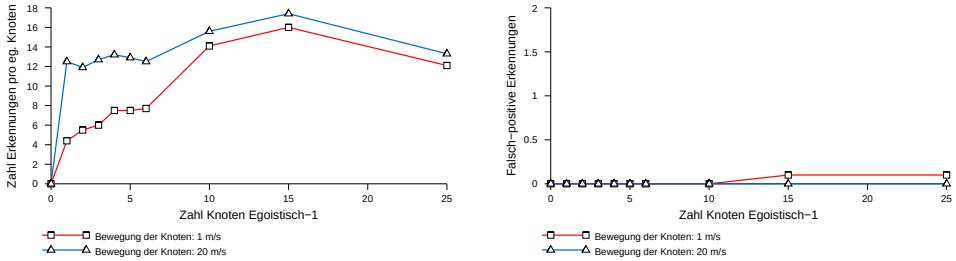

Abbildung 12.8.: Erkennungsleistung von Route Request Scanning

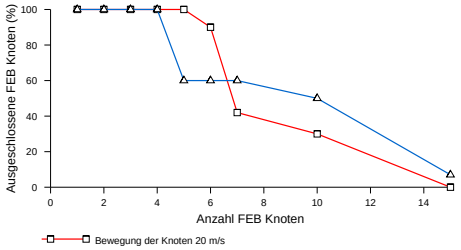

Abbildung 12.9.: Durch MobIDS ausgeschlossene FEB Knoten

der falsch-positiven Erkennungen liegt konstant unter 0,3 und ist somit zu vernachlässigen.

Bei *eindeutigem Probing* (siehe Abbildung 12.7) fällt die Erkennungsleistung deutlich geringer aus. Bei niedriger Geschwindigkeit ist immerhin die falsch-positiv Rate noch relativ gering, so dass immer noch mit einiger Trennschärfe zwischen egoistischen und normalen Knoten unterschieden werden kann. Demgegenüber nähern sich bei höherer Geschwindigkeit korrekte Erkennungen und Fehlerkennungen stark an. Im schlimmsten Fall sind bei 20 m/s und 7 egoistischen Knoten die Erkennungsrate eines Knotens und die falsch-positiv Rate mit jeweils 1,2 identisch. Somit kann aus einer Erkennung keine Information mehr gewonnen werden, ob es sich um eine Erkennung oder einen Fehlalarm handelt.

Allerdings wird sich im nächsten Abschnitt zeigen, dass MobIDS durch seine redundante Architektur mit mehreren kooperierenden Sensoren auch in dieser Situation keine Knoten zu Unrecht aus dem Netz ausschließt. Da es sehr unwahrscheinlich ist, dass ein Knoten von mehreren Sensoren und mehreren Knoten fälschlich als egoistisch eingestuft wird, wirkt sich eine einzelne falsche Beschuldigung nicht negativ aus.

Als letzter Sensor wurde das *Route Request Scanning* implementiert. Die egoistischen Knoten sind hier vom Typ Egoistisch-1, d.h. sie leiten keine Route Requests weiter (vergleiche Abschnitt 6.3.1). Abbildung 12.8 zeigt die Simulationsergebnisse. Wie man sieht, ist die Erkennungsrate sehr hoch. Was vor allem auffällt: die Zahl der Erkennungen pro egoistischem Knoten steigt mit der Zahl egoistischer Knoten sogar noch an. Eine Erklärung hierfür ist, dass die durchschnittliche Routenlänge mit der Zahl der egoistischen Knoten immer länger wird. Zum einen stehen immer weniger Knoten zur Weiterleitung des RREQ zur Verfügung, zum anderen ist auch immer weniger gecachte Routeninformation vorhanden, so dass der RREQ in einem größeren Teil des Netzes geflutet werden muss. Somit werden auch immer mehr Anfragen zu egoistischen Knoten

geschickt, welche von diesen nicht weitergeleitet werden. Es besteht somit eine größere Chance, ein Fehlverhalten zu erkennen, was letztlich zu einer häufigeren Erkennung führt.

Bei höherer Mobilität ist die Erkennungsrate ebenfalls erhöht. Dies liegt daran, dass auch hier mehr Route Requests initiiert werden und somit auch mehr Erkennungen möglich sind. Die Zahl der falsch-positiven Fehlerkennungen ist vernachlässigbar gering.

Es lässt sich also feststellen, dass alle Sensoren eine mindestens befriedigende, meist jedoch gute bis sehr gute Erkennungsleistung liefern. Wichtig ist in diesem Fall vor allem die niedrige falsch-positiv Rate, da dadurch das Vertrauen in die Richtigkeit einer erfolgte Erkennung steigt.

Ein kritischer Punkt sei an dieser Stelle noch angemerkt: eine Erkennung eines Sensors wird in den obigen Messungen immer erst dann registriert, wenn der Sensor in einem Knoten mehrfach ein Fehlverhalten festgestellt hat, also z.B. ein Overhearing Sensor dreimal keine Paketweiterleitung aufzeichnen konnte. Die Wahl dieser Schwellwerte erfolgt bei allen Messungen manuell, d.h. es wurden für die Messungen bewusst Werte gesucht, bei welchen die positive Erkennungsrate hoch, die falsch-positiv Rate jedoch gleichzeitig möglichst niedrig lag. Ziel zukünftiger Arbeiten könnte es sein, hier eine automatische Schwellwertanpassung zum Beispiel durch selbstlernende neuronale Netze zu erzielen. Mehr hierzu im Ausblick am Ende der Arbeit.

Bewertung und Ausschluss

Die Ergebnisse der Sensoren müssen nun zu lokalen und globalen Bewertungen zusammengefasst werden. Auf Basis dieser Werte kann dann eine Entscheidung über einen Ausschluss erfolgen. Die Ergebnisse der obigen Simulationen wurden hierzu dem in Abschnitt 11.7 beschriebenen Bewertungsverfahren unterzogen[8]. Abbildung 12.9 zeigt die Ergebnisse.

Bis 4 bzw. 5 FEB Knoten werden alle fehlerhaften Knoten sicher erkannt und ausgeschlossen. Auch bei 6 – 7 Knoten sind die Ergebnisse noch hinreichend gut, vor allem, wenn man bedenkt, dass eine mögliche Erkennung immer noch einen großen abschreckenden Effekt hat, da ja der MANET Ausschluss droht. Erst wenn sich eine größere Zahl von Knoten egoistisch oder böswillig verhält, beginnt MobIDS zu versagen. Dies kommt vor allem daher, weil dann nicht mehr genügend Sensoren im MANET zur Verfügung stehen, welche Daten liefern.

12.5.2. Mögliche Angriffe

Ein FEB Knoten hat zwei Möglichkeiten, wie der das MobIDS System angreifen kann. Zum einen kann er versuchen, trotz des egoistischen oder böswilligen Verhaltens seiner Entdeckung bzw. Bestrafung zu entgehen. Zum anderen kann das IDS missbraucht werden, um andere Knoten ungerechtfertigt aus dem MANET auszuschließen.

[8] Diese Analyse wurde dabei nicht innerhalb der Simulation vollzogen, sondern separat mit Hilfe einer Tabellenkalkulation durchgerechnet. Auch hier wurden die Gewichtungen der Sensoren und die Schwellwerte manuell gesetzt.

Inwieweit die Täuschung eines Sensors möglich ist, muss jeweils pro Sensor untersucht werden. Einige der Probleme von Overhearing Sensoren wurden bereits in Abschnitt 11.5.1 besprochen. Es ist bspw. eine Anpassung der Sendeleistung möglich, so dass der Sensor fehlschlägt. Allerdings treten viele dieser Probleme eher zufällig und abhängig von den Knotenpositionen auf, so dass es für einen Angreifer schwer ist, sicher einer Erkennung zu entgehen.

Das iterative Probing kann man nur schwer täuschen, wie schon in Abschnitt 11.5.4 diskutiert wurde. Entsprechend wird eine fehlende Weiterleitung hier mit hoher Wahrscheinlichkeit erkannt. Auch ist das iterative Probing, selbst bei einer intelligenten Steuerung der Weiterleitung durch den FEB Knoten, deutlich schwerer zu täuschen als das Probing mit binärer Suche von Awerbuch.

Route Request Scanning basiert wiederum auf dem Overhearing Sensor und es gilt das dort Gesagte. Da die Sensoren kooperativ arbeiten, genügt es nicht, wenn ein Knoten nur einen einzelnen Knoten täuscht, vielmehr muss er einer Entdeckung durch möglichst alle Sensoren entgehen. Je mehr Sensoren zusammenwirken, desto schwerer wird dies.

Als nächstes kann ein FEB Knoten versuchen, die Verteilung der lokalen Bewertungen zu sabotieren, um einen Ausschluss zu verhindern. Da die Bewertungen signiert sind, kann er allerdings lediglich Nachrichten verwerfen. Ist das Netz hinreichend eng vermascht, können Bewertungen allerdings an ihm vorbei transportiert werden.

Will ein Angreifer das MobIDS System gegen einen anderen Knoten einsetzen, so könnte er wiederum versuchen, einen der Sensoren so zu täuschen, dass dieser ein Fehlverhalten eines anderen Knotens zu erkennen glaubt und diesen folglich bestraft. Da Knoten in einer Route jedoch authentifiziert sind und die Kommunikation mit Sitzungsschlüsseln abgesichert ist, bleibt hierfür relativ wenig Spielraum. Außerdem müsste dann ja verhindert werden, dass der Sensor das korrekte Verhalten des Knotens registriert. Durch gezielte Störmaßnahmen könnte dies zwar realisiert werden, der Aufwand wäre aber relativ hoch.

Effizienter ist es für einen Angreifer, schlechte Bewertungen über einen Knoten im MANET zu verteilen oder Beschwerden über Fehlverhalten an die CA zu schicken. Da diese Nachrichten signiert werden, kann er allerdings keine Bewertungen anderer Knoten fälschen. Da ein Fehlverhalten aber immer von mehreren Knoten erkannt werden muss, wenn es zu einer Reaktion kommen soll, müssten also hinreichend viele Knoten kooperieren, um zum Ziel zu gelangen. Verteilt ein Knoten aber zu viele falsche Anschuldigungen, so könnte dieser wiederum von einem MobIDS Sensor erkannt werden, wodurch ein solcher Knoten auch bestraft werden kann.

Letztlich bedarf der Einsatz MobIDS einer sorgfältig abgewogenen Balance der verschiedenen Einstellungen, um einerseits keine ehrlichen Knoten zu bestrafen und andererseits egoistische und böswillige mit so großer Wahrscheinlichkeit zu entdecken, damit Benutzer es nicht riskieren, dass ihre Knoten aus dem MANET ausgeschlossen werden.

12.5.3. Aufwand und Effizienz

MobIDS erzeugt im Netz einen nicht unerheblichen Aufwand. Der Betrieb von MobIDS auf einem Knoten verbraucht Energie und Speicherplatz, vor allem der Promiscuous Modus belastet die Knoten (siehe Abschnitt 11.5.1). Zusätzliche Kommunikation entsteht durch Overhearing allerdings nicht.

Demgegenüber erzeugt Probing zusätzlichen Netzwerkverkehr durch die Probing Pakete. Eine Probe wird durch ein zusätzliches Kommandofeld im Header erzeugt, welches 128 Bit lang ist. Dieses Feld muss in jedem (Daten-)Paket vorhanden sein. Acknowledgments werden entweder in den normalen Datenverkehr integriert oder in separaten Paketen übertragen. Das Probing wird im Normalfall nur beim Aufbau einer Route und danach zyklisch in größeren Abständen verwendet. Auch werden hier Probes zunächst nur zum Zielknoten geschickt, Zwischenknoten werden nicht explizit getestet. Erst wenn es bei einem solchen Probing zu einem Fehler kommt, wird das iterative Probing aktiviert, bei dem jeder Zwischenknoten untersucht wird. Die Zahl der dann notwendigen ACK Pakete entspricht der Länge der Route. Somit ist das Probing bei MobIDS deutlich ressourcenschonender als der Ansatz von Awerbuch [AHNRR02], bei dem jedes Paket explizit bestätigt werden muss.

Das Route Request Scanning erzeugt neben den Kosten des Promiscuous Modes hauptsächlich Speicherlast, weil jeder Knoten die Route Requests für eine gewisse Zeit zwischenspeichern muss.

Die Verteilung der lokalen Bewertungen erzeugt wiederum einiges an Daten, die im Netz geflutet werden müssen. Wie man in Abbildung 11.13 sieht, bestehen die statischen Daten des Pakets aus insgesamt 13 Byte[9]. Hinzu kommen pro bewertetem Knoten nochmals 10 Byte[10]. Ein solches Paket muss dann noch signiert werden, was weitere 128 Byte benötigt. Hat ein Knoten also lokale Bewertungen über 10 verschiedene Knoten, so hat ein Bewertungspaket die Größe 241 Byte. In einem MANET mit 100 Teilnehmern müssen dann 100 dieser Pakete zyklisch verteilt werden.

12.6. Fazit

Die Analyse der verschiedenen Komponenten von SAM hat gezeigt, dass zum einen viele Angriffe effektiv verhindert werden. Andererseits bleiben bei allen Komponenten gewisse Restrisiken bestehen, die jedoch einem erfolgreichen Einsatz von SAM nicht unbedingt im Wege stehen.

Wie sich gezeigt hat, sind MANET-IDs eine relativ schlanke Authentifizierungslösung, welche kaum Overhead erzeugt und nur schwer angegriffen werden kann. Beim Einsatz von Pseudonymen besteht ein grundsätzlicher Konflikt zwischen Schutz der Privatsphäre und der sicheren Erkennung und Identifizierung egoistischer böswilliger Knoten. Hier muss beim konkreten Einsatz von SAM eine entsprechende Abwägung getroffen werden.

Es konnte durch Einsatz formaler Methoden gezeigt werden, dass SDSR die gestellten Anforderungen erfüllt. Dies sollte aber nicht mit der Aussage verwechselt werden, dass

[9] TTL (1 Byte); k_i (8 Byte); ID (2 Byte); t (2 Byte)
[10] k_n (8 Byte); r (2 Byte)

keine Angriffe gegen SDSR möglich sind. Viele als sicher geltende Protokolle wurden in der Vergangenheit durch neue Formen von Angriffen bedroht. Durch Simulationen und Abschätzungen der Paketgrößen konnte ferner gezeigt werden, dass sich der Overhead durch SDSR in gewissen Grenzen hält. Da bei SDSR, bedingt durch die stark erweiterte Funktionalität, deutlich mehr Daten transportiert werden müssen, als bei DSR, verwundert dies nicht.

Schließlich konnten weitere Simulationen die Effektivität der verschiedenen MobIDS Sensoren belegen. Zusammengenommen ließen sich so nahezu alle FEB Knoten aus einem Netz ausschließen, ohne dass es zu Fehlerkennungen von normalen Knoten kam. Der entstehende Overhead im Netz bewegt sich ebenfalls innerhalb erträglicher Grenzen.

Trotzdem wirft die Analyse einige Fragestellungen auf und gibt Hinweise auf zukünftige Optimierungsmöglichkeiten. Das folgende Kapitel fasst nochmals die wesentlichen Aspekte von SAM zusammen und gibt einen Ausblick auf diese zukünftigen Entwicklungen.

13. Zusammenfassung und Ausblick

Dieses Kapitel schließt die Arbeit ab. Es folgt zunächst eine Zusammenfassung der vorangegangenen Kapitel. Daran schließt sich eine Aufstellung der wesentlichen Beiträge an. Schließlich folgt ein Ausblick auf noch offene Fragestellungen und es wird diskutiert, wie diese in zukünftigen Arbeiten gelöst werden könnten. Den Abschluss bildet eine Bewertung und Einschätzung der zukünftigen Entwicklung.

13.1. Zusammenfassung

Nach den einleitenden Kapiteln, welche die Grundlagen von Sicherheit, Kryptographie und mobiler Datenkommunikation darlegten, folgte eine umfassende Sicherheitsanalyse für Mobile Ad hoc Netze. Insbesondere der Fall der öffentlichen, „subscription-less“ Netze wurde diskutiert. Zunächst wurden hier die Ziele einer Sicherheitslösung vorgestellt, anschließend wurde eine große Zahl möglicher Angriffe in Form sogenannter Angriffsbäume aufbereitet, was insbesondere die Analyse der Abhängigkeiten verschiedener Attacken vereinfacht. Für zwei Formen von Egoismus wurden exemplarisch die Auswirkungen auf ein MANET durch Simulationen untersucht. Es hat sich gezeigt, dass solche Knoten die Leistungsfähigkeit eines Netzes massiv beeinträchtigen können. Wie sich herausgestellt hat, sehen die bestehenden Routingprotokolle gegen solche Angriffe keinerlei Sicherheitsmechanismen vor.

In Kapitel 7 wurde als Lösung die Sicherheitsinfrastruktur SAM („Security Architecture for Mobile Ad hoc Networks“) vorgeschlagen, welche aus den Komponenten „Identifizierung und Pseudonyme“, „Sicheres Routing und Schlüsselaustausch“ und einem „Mobilen Intrusion Detection System“ besteht. Dabei wurden die gegenseitigen Abhängigkeiten und Anforderungen untersucht.

Die folgenden Kapitel widmeten sich jeweils einer Komponente im Detail. Kapitel 8 stellte zunächst verwandte Arbeiten zur Authentifizierung von Knoten in MANETs vor. Wie sich herausstellte, war der Identitätsbegriff nur unzureichend definiert, so dass zunächst erörtert wurde, was unter einer Identität in einem Ad hoc Netzwerk zu verstehen ist. Anschließend wurde das MANET-ID System beschrieben, welches SAM zur Identifizierung von Knoten in MANETs verwendet. Zum Ausschluss von Knoten verfügen MANET-IDs mit dem MANET-CRS über ein speziell auf die Belange von MANETs zugeschnittenes Rückrufsystem.

Als nächste Komponente von SAM wurde in Kapitel 10 das „Secure Dynamic Source Routing“ Protokoll (SDSR) vorgestellt. Auch hier wurden zunächst andere Arbeiten zu sicherem Routing analysiert. Aufbauend auf diesen Erkenntnissen wurde dann SDSR entwickelt. SDSR erfüllt dabei zusätzlich zur Absicherung der Routingdaten weitere Aufgaben, wie Authentifizierung der Knoten und Austausch von Sitzungsschlüsseln, und ist hierin den anderen Protokollen überlegen.

Als letzten Bestandteil von SAM stellte Kapitel 11 das „Mobile Intrusion Detection System“ (MobIDS) vor. Wie einige der anderen beschriebenen Systeme, verfügt MobIDS auch über Sensoren, welche unerwünschtes Verhalten registrieren. Die Sensoren von MobIDS sind deutlich vielfältiger und weiter entwickelt, als die der anderen Systeme. Die Beobachtungen der Sensoren werden zu einer lokalen Bewertung zusammengefasst. Nach Austausch der lokalen Bewertungen entscheiden dann die Knoten kooperativ über einen Ausschluss eines Knotens aus dem MANET. Zusätzlich besteht die Möglichkeit, die MANET-ID eines Knotens bei der MANET-CA global zu sperren und ihm somit die weitere Teilnahme an MANETs zu verwehren.

Den funktionalen Beschreibungen schloss sich ein Analysekapitel an, in welchem die Komponenten von SAM auf ihre Funktionsfähigkeit, auf verbleibende Angriffsmöglichkeiten und auf ihre Effizienz hin untersucht wurden. Dabei kamen neben grundlegenden Überlegungen zu Paketgrößen und -anzahlen vor allem Simulationen zum Einsatz. Ferner wurde das SDSR Protokoll mittels BAN Logik formal analysiert. Die Ergebnisse zeigen die große Funktionalität von SAM, die mit moderatem Aufwand zu erreichen ist. Allerdings wurden auch Grenzen entdeckt, vor allem was die Effizienz einzelner MobIDS Sensoren in bestimmten Situationen betrifft.

13.2. Geleistete Beiträge

Welches sind nun die wesentlichen Beiträge dieser Arbeit zum Forschungsgebiet?

1. Nach meinem Kenntnisstand ist dies die erste Arbeit zu Sicherheit in Mobilen Ad hoc Netzen, welche umfassend alle beteiligten Aspekte untersucht und Lösungen vorschlägt. Andere Arbeiten befassen sich nur mit Teilgebieten und übersehen meist bestehende Abhängigkeiten.

2. Im Bereich der Angriffsanalyse wurde insbesondere das DSR Protokoll sehr detailliert untersucht. Die dargestellten Angriffsbäume sind die erste strukturierte und umfassende Analyse dieser Art.

3. Der Begriff der Identität in MANETs wurde hier erstmals genauer definiert. Bisher blieb meist unklar, ob Geräte oder Personen authentifiziert wurden und was eine solche Authentifizierung aussagt.

4. Das MANET-ID System ist ein eigenständiges Authentifizierungssystem für MANETs, welches sich von anderen Arbeiten insbesondere durch die klar definierten Anfangsbedingungen unterscheidet. Auch der Rückruf von Zertifikaten und die globale Sperrung von Knoten sind Alleinstellungsmerkmale.

5. Das SDSR Protokoll ist ein hochfunktionales und sicheres Routingprotokoll für MANETs. Authentifizierung und die Integration eines Schlüsselaustauschs für Sitzungsschlüssel sind hier integriert. Auch die Verteilung von öffentlichen Schlüssel, die bei vielen anderen Protokollen vorausgesetzt wird, ist bei SDSR explizit beschrieben. Besonders ist zu erwähnen, dass SDSR, im Gegensatz zu vielen anderen Protokollen, mittels BAN Logik formal untersucht wurde, was das Vertrauen in seine Sicherheit stärkt.

6. MobIDS entwickelt mit dem aktivitätsbasierten und dem kombinierten Overhearing, mit iterativem und eindeutigem Probing und dem Route Request Scanning

vor allem die Sensoren deutlich weiter, verglichen mit den bisherigen Arbeiten anderer Autoren. Aber auch der Ausschluss von Knoten ist hier präziser beschrieben als bei anderen Lösungen.

13.3. Zukünftige Arbeiten

Bedingt durch die Breite der Arbeit konnten nicht alle Fragestellungen in beliebiger Tiefe behandelt werden, ohne den Umfang vollends zu sprengen. Es ging mir primär darum, ein komplettes Sicherheitsframework ohne Lücken zu schaffen, welches in der Praxis so eingesetzt werden kann. Entsprechend stellt diese Dissertation nicht das Ende der Forschungsanstrengungen dar, vielmehr ergeben sich aus ihr eine Reihe von weiteren, interessanten Fragestellungen, welche in der Zukunft in Form von Diplomarbeiten oder weiteren Dissertationen bearbeitet werden können.

Eines der Ergebnisse der Analyse von SDSR war, dass der Overhead mitunter relativ hoch ist. Dies ist vor allem darauf zurückzuführen, dass Optimierungen wie das Route Caching aus Sicherheitsgründen nicht eingesetzt wurden.

Es ist denkbar, ähnlich wie bei SAODV, Route Caching trotzdem einzusetzen und die Signatur des ursprünglichen Absenders mit einzubauen. Dies wird zu großen Problemen beim Schlüsselaustausch führen, weil dann der Diffie-Hellmann Prozess zeitlich stark auseinander gezogen wird und die Knoten ihre DH Schlüssel längere Zeit speichern müssen. Trotzdem sollte dieser Weg untersucht und analysiert werden. Kombiniert mit einer Expanding Ring Search ergibt sich eventuell eine Möglichkeit, einiges an Overhead einzusparen.

Auch die Größe der SDSR Pakete ist noch nicht wirklich zufriedenstellend. Maßgeblich hierfür sind die Signaturen und öffentlichen Schlüssel, welche in den Paketen transportiert werden. Auch hier ist zu prüfen, inwieweit es noch Einsparpotential gibt.

Bei MobIDS gibt es noch ein weites Betätigungsfeld. Bisher wurden die Gewichte und Schwellwerte für die Sensoren und die Bewertungserstellung manuell festgelegt. Da die optimalen Werte jedoch je nach Einsatzszenario schwanken können, empfiehlt sich eigentlich eine dynamische Anpassung, die im Betrieb selbst stattfinden muss. Neben der klassischen Anpassung der Gewichte anhand gemessener Werte[1], lassen sich hier möglicherweise auch neuronale Netze oder Fuzzy Logik sinnvoll einsetzen.

Ein weiterer Punkt, der noch weiter untersucht werden kann: wie wirkt sich die Zahl der verfügbaren Pseudonyme auf die Erkennungsleistung von MobIDS aus? Um dies zu untersuchen ist ein intelligenter Angreifer notwendig, welcher geschickt in der Lage ist, das jeweilig passende Pseudonym zu wählen, um einer Erkennung zu entgehen und unter Umständen sogar mehrere Pseudonyme für einen Angriff kooperieren lässt.

Damit sind wir auch schon beim Hauptproblem. Die durchgeführten Simulationen repräsentieren immer nur ein festgelegtes und beschränktes Szenario, nie die tatsächliche Wirklichkeit. Es wird also notwendig sein, die Effektivität von SAM in realen Feldversuchen mit realen Benutzern und realen Anwendungen zu untersuchen. Im Idealfall

[1] z.B. durchschnittliche Routenlänge, durchschnittliche Haltbarkeit von Routen, Zahl der Nachbarn usw.

steht dabei für die Benutzer die Anwendung im Vordergrund während das Sicherheitsframework unbemerkt im Hintergrund mitläuft. Motiviert durch beschränkte Energieressourcen oder auch aus Spieltrieb, werden Benutzer irgendwann versuchen, die Sicherheitsmechanismen zu umgehen. Dann wird sich zeigen, wie stabil SAM wirklich ist.

Die Erkenntnisse, die bei der Arbeit mit SAM gewonnen wurden, sind nicht unbedingt auf das MANET Szenario beschränkt. Bei der Vernetzung von IT-Systemen geht die Tendenz immer mehr zu dezentralen und unorganisierten Verbünden von Knoten, die unter der Kontrolle einer Vielzahl unterschiedlicher Benutzer stehen, und die gemeinsam eine bestimmte Aufgabe vollbringen. Ein aktuelles Beispiel sind die populären Peer-to-Peer Netze, die heute hauptsächlich für Filesharing genutzt werden, aber auch für verteiltes Rechnen, verteiltes Publizieren uvm. eingesetzt werden können.

Ohne eine klare Trennung von Infrastruktur und Benutzer und bedingt durch die große Zahl von Teilnehmern, die sich nie vorher kennen gelernt haben, stehen diese Netze vor ähnlichen Problemen, wie MANETs. Vertrauen muss aufgebaut werden und das Netz muss sich kooperativ vor egoistischen oder böswilligen Teilnehmern schützen. Hier können viele Verfahren aus SAM übertragen werden.

Und auch das Internet selbst steht mit zunehmendem Wachstum vor dem Problem, dass eine zentrale Kontrolle durch wenige Backbone-ISPs nicht mehr funktioniert. Heute tragen tausende von Netzen zum Betrieb des Internets bei. Prinzipiell könnte jedes einzelne davon durch die Verbreitung falscher BGP Routen großen Schaden anrichten und die globale Konnektivität gefährden. Heute werden solche Katastrophen meist noch durch die Erfahrung der Admins und manuelle Filterlisten verhindert. In dem Masse, wie sich neue Teilnehmer anschließen, steigt aber auch die Dynamik des Internet und somit die Gefahr, dass es zu Problemen kommt. Schon heute geht der Trend zu Multihoming, d.h. dass sich Netze aus Redundanzgründen mit möglichst vielen anderen Netzen verbinden. In diesem Sinne wird das Internet vielleicht einmal wie ein großes MANET funktionieren. Dann werden Sicherheitsmechanismen wie SAM möglicherweise einmal auch im Internet eingesetzt werden.

13.4. Schlusswort

Ob eine Sicherheitsarchitektur wie SAM für MANETs wirklich notwendig ist, hängt natürlich primär vom Erfolg und dem tatsächlichen Einsatzzweck der MANETs selbst ab. Obwohl es eine Vielzahl von Forschungsvorhaben auf diesem Gebiet gibt, sind wir von einem breiten, produktiven MANET Einsatz noch eine Weile entfernt. Auch ist unklar, welche Anwendungsszenarien sich tatsächlich durchsetzen werden. Doch egal, in welcher Form MANETs schließlich zum Einsatz kommen, eines ist klar: die Benutzer werden keine Systeme einsetzen wollen, bei denen ihr Nachbar ihre Nachrichten mitlesen oder das Netz beliebig außer Funktion setzen kann.

> Vielleicht wird es in der Zukunft keine MANETs und keine MANET-Sicherheit geben, aber ganz sicher wird es keine MANETs ohne MANET-Sicherheit geben.

A. Abbildungsverzeichnis

B. Tabellenverzeichnis

C. Literaturverzeichnis

[Abr70] Abramson, N.: The ALOHA system - another alternative for computer communications. In: *Fall Joint Computer Conference, AFIPS Conference Proceedings* Bd. 37, 1970, S. 281–285

[ACJ+03] Adjih, Cedric ; Clausen, Thomas ; Jacquet, Philippe ; Laouiti, Anis ; Minet, Pascale ; Muhlethaler, Paul ; Qayyum, Amir ; Viennot, Laurent. *Optimized Link State Routing Protocol*. http://www.ietf.org/internet-drafts/draft-ietf-manet-olsr-08.txt. März 2003

[AG00] Asokan, N. ; Ginzboorg, Philip: Key agreement in ad hoc networks. In: *Computer Communications* 23 (2000), S. 1627–1637

[AHNRR02] Awerbuch, Baruch ; Holmer, David ; Nita-Rotaru, Cristina ; Rubens, Herbert: An On-Demand Secure Routing Protocol Resilient to Byzantine Failures. In: *ACM Workshop on Wireless Security (WiSe)*. Atlanta, Georgia, September 2002. – auch verfügbar unter http://citeseer.nj.nec.com/article/awerbuch02demand.html

[AIR] *Airsnort Homepage*. http://airsnort.shmoo.com/

[AK96] Anderson, R. ; Kuhn, M.: Tamper Resistance - a Cautionary Note. In: *Proceedings of the Second Usenix Workshop on Electronic Commerce*, 1996, S. 1–11. – auch verfügbar unter http://citeseer.nj.nec.com/article/anderson96tamper.html

[AK97] Anderson, Ross ; Kuhn, Markus: Low Cost Attacks on Tamper Resistant Devices. In: *IWSP: International Workshop on Security Protocols, LNCS*, 1997. – auch verfügbar unter http://citeseer.nj.nec.com/anderson97low.html

[Alt99] Althouse, E.: Extending the Littoral Battlespace (ELB). In: *Advanced Concept Technology Demonstration (ACTD), NATO Information Systems Technology Panel Symposium on Tactical Mobile Communications*, 1999

[And80] Anderson, J. P.: Computer Security Threat Monitoring and Surveillance / James P Anderson Co. Fort Washington, PA, April 1980. – Forschungsbericht

[ANO] *The Anonymizer*. http://www.anonymizer.com/

[ANS] *IRTF RRG Ad hoc Network Scaling Research Subgroup*. http://www.flarion.com/ans-research/

[Arb01] Arbaugh, W.A.: An inductive chosen plaintext attack against WEP/WEP2 / IEEE. 2001 (802.11-01/230). – Forschungs-

bericht. http://grouper.ieee.org/groups/802/11/Documents/DocumentHolder/1-230.zip

[Årn00] Årnes, André: *Public Key Certificate Revocation Schemes*, Queen's University, Kingston, Canada, Thesis, Januar 2000. – auch verfügbar unter http://www.pvv.ntnu.no/~andrearn/certrev/

[ASW01] Arbaugh, W.A. ; Shankar, N. ; Wan, Y.J. *Your 802.11 wireless network has no clothes*. http://www.cs.umd.edu/~waa/wireless.pdf. März 2001

[AT91] Abadi, Martín ; Tuttle, Mark R.: A Semantics for a Logic of Authentication. In: *Proceedings of the Tenth Annual ACM Symposium on Principles of Distributed Computing*, 1991, S. 201–216. – auch verfügbar unter http://citeseer.nj.nec.com/article/abadi91semantics.html

[AT02] Arsenault, Alfred ; Turner, Sean. *Internet X.509 Public Key Infrastructure: Roadmap*. PKIX Working Group Internet Draft, http://www.ietf.org/internet-drafts/draft-ietf-pkix-roadmap-09.txt. Juli 2002

[Bag01] Bager, Jo. *Heise News-Ticker: Microsoft warnt vor Cracker-Zertifikat*. http://www.heise.de/newsticker/data/jo-24.03.01-001/. 2001

[Bak97] Baker, D.: Data/Voice Communication over a Multihop, Mobile, High-Frequency Network. In: *Proceedings of the IEEE Military Communications Conference (MILCOM '97)*, 1997

[BAN90a] Burrows, Michael ; Abadi, Martín ; Needham, Roger M.: A Logic of Authentication. In: *ACM Transactions on Computer Systems* 8 (1990), Nr. 1, S. 18–36

[BAN90b] Burrows, Michael ; Abadi, Martín ; Needham, Roger M.: Rejoinder to Nessett. In: *ACM Operating Systems Review* 24 (1990), Nr. 2, S. 39–40

[Bau00] Bauer, Friedrich L.: *Entzifferte Geheimnisse*. 2nd. Springer, 2000

[BB01] Buchegger, Sonja ; Boudec, Jean-Yves L.: Nodes Bearing Grudges: Towards Routing Security, Fairness, and Robustness in Mobile Ad Hoc Networks. In: *Proceedings of IEEE/ACM Workshop on Mobile Ad Hoc Networking and Computing (MobiHOC)*. Lausanne, CH, 2001

[BB02a] Buchegger, Sonja ; Boudec, Jean-Yves L.: Nodes Bearing Grudges: Towards Routing Security, Fairness, and Robustness in Mobile Ad Hoc Networks. In: *Proceedings of the Tenth Euromicro Workshop on Parallel, Distributed and Network-based Processing*. Canary Islands, Spain : IEEE Computer Society, Januar 2002, S. 403–410. – http://citeseer.nj.nec.com/article/buchegger02nodes.html

[BB02b] Buchegger, Sonja ; Boudec, Jean-Yves L.: Performance Analysis of the CONFIDANT Protocol: Cooperation Of Nodes - Fairness in Distributed Ad-hoc Networks. In: *Proceedings of IEEE/ACM Workshop on Mobile Ad Hoc Networking and Computing (MobiHOC)*. Lausanne, CH, Juni 2002

[BBP00] Bahl, P. ; Balachandran, A. ; Padmanabhan, V.: Enhancements to the RADAR User Location and Tracking System / Microsoft Research. 2000. – Forschungsbericht. auch verfügbar unter http://citeseer.nj.nec.com/bahl00enhancements.html

[BEGA02] Bobba, Rekesh B. ; Eschenauer, Laurent ; Gligor, Virgil ; Arbaugh, William A.: Bootstrapping Security Associations for Routing in Mobile Ad-Hoc Networks / Institute for Systems Research, UMd. 2002 (TR 2002-44). – Forschungsbericht. auch verfügbar unter http://citeseer.nj.nec.com/bobba02bootstrapping.html

[Bel58] Bellman, R.: On a routing problem. In: *Quart. Appl. Math.* 16 (1958), S. 87–90

[BF01] Boneh, Dan ; Franklin, Matt: Identity-Based Encryption from the Weil Pairing. In: *Lecture Notes in Computer Science* 2139 (2001), S. 213–229

[Bög01] Bögeholz, Harald: Kopierschutz kommt – Festplatten bewachen Inhalte. In: *c't Magazin für Computertechnik* (2001), Nr. 9, S. 50. – Heise Zeitschriften Verlag, Hannover

[BGW01] Borisov, Nikita ; Goldberg, Ian ; Wagner, David: Intercepting mobile communications: the insecurity of 802.11. In: *Proceedings of the seventh annual international conference on Mobile computing and networking*, 2001, S. 180–189

[BH01a] Buttyan, Levente ; Hubaux, Jean-Pierre: Rational Exchange – A Formal Model Based on Game Theory. In: *Proceedings of 2nd International Workshop on Electronic Commerce (WELCOM 2001)*. Heidelberg, Germany, November 2001

[BH01b] Buttyán, Levente ; Hubaux, Jean-Pierre: Nuglets: a Virtual Currency to Stimulate Cooperation in Self-Organized Mobile Ad Hoc Networks / EPFL-DI-ICA. 2001 (DSC/2001/001). – Forschungsbericht

[BH03] Buttyán, Levente ; Hubaux, Jean-Pierre: Stimulating Cooperation in Self-Organizing Mobile Ad Hoc Networks. In: *ACM/Kluwer Mobile Networks and Applications* 8 (2003), Oktober, Nr. 5

[BHČ02] Buttyán, Levente ; Hubaux, Jean-Pierre ; Čapkun, Srđan: A Formal Analysis of Syverson's Rational Exchange Protocol. In: *Proceedings of IEEE Computer Security Foundations Workshop*. Cape Breton, Nova Scotia, Canada, Juni 2002

[BHGV87] Binder, R. ; Huffman, S. ; Gurantz, I. ; Vena, P.: Crosslink Architectures for a Multiple Satellite System. In: *Proceedings of the IEEE (Special Issue, Packet Radio Networks)* 75 (1987), Januar, Nr. 1, S. 75–82

[Bla79] Blakley, G. R.: Safeguarding Cryptographic Keys. In: *Proceedings of the National Computer Conference* 48 (1979), S. 242–268

[Blu01a] Bluetooth SIG. *Bluetooth Specifications Version 1.1*. https://www.bluetooth.org/foundry/specification/document/spec_v1_1/en/2/spec_v1_1.html. 2001

[Blu01b] Bluetooth SIG: *Specification of the Bluetooth System - Volume 1 Core - Version 1.1*, Februar 2001. – https://www.bluetooth.org/foundry/specification/document/Bluetooth_V1.1_Core_Specifications

[Blu01c] Bluetooth SIG: *Specification of the Bluetooth System - Volume 2 Profiles - Version 1.1*, Februar 2001. – https://www.bluetooth.org/foundry/specification/document/Bluetooth_11_Profiles_Book

[BMJ+98] Broch, Josh ; Maltz, David A. ; Johnson, David B. ; Hu, Yih-Chun ; Jetcheva, Jorjeta: A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols. In: *Mobile Computing and Networking*, 1998, S. 85–97. – auch verfügbar unter http://citeseer.nj.nec.com/broch98performance.html/

[BP00] Bahl, Paramvir ; Padmanabhan, Venkata N.: RADAR: An In-Building RF-Based User Location and Tracking System. In: *INFOCOM (2)*, 2000, S. 775–784. – auch verfügbar unter http://citeseer.nj.nec.com/bahl00radar.html

[BRK95] Bartholomé, A. ; Rung, J. ; Kern, H.: *Zahlentheorie für Einsteiger*. Vieweg, 1995

[BRSP01] Belding-Royer, Elizabeth M. ; Sun, Yuan ; Perkins, Charles E. *Global Connectivity for IPv4 Mobile Ad hoc Networks*. http://www.cs.ucsb.edu/~ebelding/txt/globalv4.txt. 2001

[BS01] Bray, Jennifer ; Sturman, Charles: *Bluetooth: Connect Without Cables*. Prentice Hall PTR, 2001

[BSI94] BSI. *E-Government Handbuch, Kapitel IV B, Authentisierung im E-Government*. BSI, 1994, http://www.bsi.de/fachthem/egov/download/4_Authen.pdf. 1994

[Bun45] Bundesrepublik Deutschland. *Grundgesetz (Fassung vom 29.11.2000)*. Mai 1945

[Bun83] Bundesverfassungsgericht Deutschland. *Volkszählungsurteil*. Az.: 1 BvR 209/83; NJW 84, 419. Dezember 1983

[Bun90] Bundesrepublik Deutschland. *Bundesdatenschutzgesetz (BDSG) (Fassung vom 14. Januar 2003)*. http://www.brandenburg.de/land/lfdbbg/gesetze/bdsg.htm. Dezember 1990

[Cal03] Callaway, Edgar H.: *Wireless Sensor Networks: Architectures and Protocols*. CRC Press, 2003

[Cas03] Castelluccia, Claude: Hash-Based Dynamic Source Routing (HBDSR) / Institut National de Recherche en Informatique et en Automatique (INRIA). 2003 (4784). – Forschungsbericht. – 18 S. auch verfügbar unter http://www.inrialpes.fr/planete/people/ccastel/RT4784.ps

[CB95] Cheswick, William R. ; Bellovin, Steven M.: *Firewall and Internet Security : Repelling the Wily Hacker*. Addison-Wesley, 1995

[ČBH02] Čapkun, Srđan ; Buttyan, Levente ; Hubaux, Jean-Pierre: Self-Organized Public-Key Management for Mobile Ad Hoc Networks / Swiss Federal Institute of Technology Lausanne (EPFL). 2002. – Forschungsbericht

[CCI87] *The Directory – Authentication Framework*. CCITT Draft Recommendation X.509. 1987

[Cha81] Chaum, David: Untraceable electronic mail, return addresses, and digital pseudonyms. In: *Communications of the ACM* 4 (1981), Februar, Nr. 2

[ČHH02] Čapkun, Srđan ; Hamdi, Maher ; Hubaux, Jean-Pierre: GPS-free Positioning in Mobile Ad-Hoc Networks. In: *Cluster Computing* 5 (2002), April, Nr. 2

[CJL+01] Clausen, T. ; Jacquet, P. ; Laouiti, A. ; Muhlethaler, P. ; Qayyum, A. ; Viennot, L.: Optimized Link State Routing Protocol. In: *IEEE INMIC*. Pakistan, 2001. – auch verfügbar unter http://hipercom.inria.fr/olsr/inmic2001.ps

[CLR92] Cormen, T.H. ; Leiserson, C.E. ; Rivest, R.L.: *Introduction to Algorithms*. MIT Press, 1992

[CM99] Corson, S. ; Macker, J. *RFC 2501: Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations*. http://www.ietf.org/rfc/rfc2501.txt. Januar 1999

[CP02] Courtois, Nicolas T. ; Pieprzyk, Josef: Cryptanalysis of Block Ciphers with Overdefined Systems of Equations. In: *Proceedings of Asiacrypt 2002*, Springer, 2002 (Lecture Notes in Computer Science). – auch verfügbar als http://citeseer.nj.nec.com/courtois02cryptanalysis.html

[CS98] Caffery, J. ; Stuber, G.: Overview of Radiolocation in CDMA Cellular Systems. In: *IEEE Communications Magazine* 36 (1998), April, Nr. 4, S. 38–45. – auch verfügbar unter http://citeseer.nj.nec.com/caffery98overview.html

[CS02] Choi, John D. ; Stark, Wayne E.: Performance of Ultra-Wideband Communications With Suboptimal Receivers in Multipath Channels. In: *IEEE Journal on Selected Areas in Communications* 20 (2002), Dezember, Nr. 9

[CZ95] Chapman, D. B. ; Zwicky, Elizabeth D.: *Building Internet Firewalls*. O'Reilly, 1995

[D+03] Donnerhacke, Lutz [u. a.]. *de.comp.security.firewall FAQ*. http://www.iks-jena.de/mitarb/lutz/usenet/Firewall.html. 2003

[DF89] Desmedt, Yvo ; Frankel, Yair: Threshold cryptosystems. In: *Lecture Notes in Computer Science* 435 (1989), S. 307–315

[DFJ+94] Deering, S. ; Farinacci, D. ; Jacobson, V. ; Lui, C. ; Wei, L.: An Architecture For Wide Area Multicast Routing. In: *Proceedings of ACM SIGCOMM '94*, 1994

[DH76] Diffie, W. ; Hellmann, M.E.: New Directions in Cryptography. In: *IEEE Transactions on Information Theory IT* 22 (1976), Nr. 6, S. 644–654

[DH98] Deering, Stephen E. ; Hinden, Robert M. *RFC2460: Internet Protocol, Version 6 (IPv6) Specification*. http://www.ietf.org/rfc/rfc2460.txt. 1998

[Dij59] Dijkstra, E.W.: A note on two problems in connexion with graphs. In: *Numerische Mathematik* 1 (1959), S. 269–271

[DN85] Denning, Dorothy E. ; Neumann, Peter G.: Requirements and Model for IDES – A Real-Time In5 trusion Detection System / Computer Science Laboratory, SRI International. 1985 (83F83-01-00). – Forschungsbericht

[DR99] Daemen, J. ; Rijmen, V. *AES Proposal: Rijndael*. NIST AES Homepage: http://csrc.nist.gov/encryption/aes/round2/r2algs.htm. 1999

[DR00] Daemen, J. ; Rijmen, V.: The Block Cipher Rijndael. In: *Smart Card Research and Applications* Bd. 1820. Springer-Verlag, 2000, S. 288–296

[Dro97] Droms, R. *RFC 2131: Dynamic Host Configuration Protocol*. http://www.ietf.org/rfc/rfc2131.txt. 1997

[DRWT97] Dube, R. ; Rais, C. ; Wang, K.-Y. ; Tripathi, S.K.: Signal Stability-Based Adaptive Routing (SSA) for Ad Hoc Mobile Networks. In: *IEEE Personal Communications* 4 (1997), Februar, Nr. 1, S. 36–45

[DS81] Denning, Dorothy E. ; Sacco, Giovanni M.: Timestamps in Key Distribution Protocols. In: *Communications of the ACM* 24 (1981), Nr. 8, S. 533–536

[DSS94] *Digital Signature Standard*. NIST FIPS PUB 186, National Institute of Standards and Technology, 1994, http://grouper.ieee.org/groups/802/11/Documents/DocumentHolder/0-362.zip. 1994

[EF01] Espenschied, Dragan ; Freude, Alvar C. *insert_coin*. http://odem.org/insert\protect\T1\textunderscorecoin/. 2001

[Ert01] Ertel, Wolfgang: *Angewandte Kryptographie*. 2nd. Fachbuchverlag Leipzig im Carl Hanser Verlag, 2001

[ES00] Ellison, Carl ; Schneier, Bruce: Ten Risks of PKI: What You're not Being Told about Public Key Infrastructure. In: *Computer Security Journal* 16 (2000), Nr. 1, S. 1–7. – auch verfügbar unter http://www.counterpane.com/pki-risks.html

[EVB01] Eberspächer, Jörg ; Vögel, Hans-Jörg ; Bettstetter, Christian: *GSM Global System for Mobile Communication*. Teubner Verlag, 2001

[EWB87] Ephremides, A. ; Wieselthier, J.E. ; Baker, D.J.: A Design Concept for Reliable Mobile Radio Networks with Frequency Hopping Signaling. In: *Proceedings of the IEEE (Special Issue, Packet Radio Networks)* 75 (1987), Januar, Nr. 1, S. 56–63

[FCA+87] Fischer, J. ; Cafarella, J. ; Arsenault, D. ; Flynn, G. ; Bouman, C.: Wide-Band Packet Radio Technology. In: *Proceedings of the IEEE (Special Issue, Packet Radio Networks)* 75 (1987), Januar, Nr. 1, S. 100–115

[FD92] Frankel, Yair ; Desmedt, Yvo G.: Parallel Reliable Threshold Multisignature / Dept. of EECS, University of Wisconsin-Milwaukee. 1992 (TR-92-04-02). – Forschungsbericht

[FF62] Ford, L.R. ; Fulkerson, D.R. *Flows in Networks*. Princeton University Press, Princeton, NJ. 1962

[FKL+78] Fuchs, Werner (Hrsg.) ; Klima, Rolf (Hrsg.) ; Lautmann, Rüdiger (Hrsg.) ; Rammstedt, Otthein (Hrsg.) ; Wienold, Hanns (Hrsg.): *Lexikon zur Soziologie*. Überarb. und erweit. 2. Auflage. Opladen : Westdeutscher Verlag, 1978

[Fle] *Fleetnet*. http://www.fleetnet.de/

[FLS02] Frey, Hannes ; Lehnert, Johannes K. ; Sturm, Peter: UbiBay: An auction system for mobile multihop adhoc networks. In: *Workshop on Ad hoc Communications and Collaboration in Ubiquitous Computing Environments at ACM 2002 Conference on Computer Supported Cooperative Work (CSCW2002)*. New Orleans, USA, November 2002

[FM02] Fuller, Joanne ; Millan, William. *On Linear Redundancy in the AES S-Box*. Preprint, verfügbar als http://citeseer.nj.nec.com/fuller02linear.html. 2002

[FMS01] Fluhrer, Scott ; Mantin, Itsik ; Shamir, Adi: Weaknesses in the Key Scheduling Algorithm of RC4. In: *Proceedings of Eigth Annual Workshop on Selected Areas in Cryptography*, 2001. – auch verfügbar unter http://citeseer.nj.nec.com/fluhrer01weaknesses.html

[Fok02] Fokine, Klas: *Key Management in Ad Hoc Networks*, Linköping University, Schweden, Master's Thesis, September 2002. – auch verfügbar unter http://www.ep.liu.se/exjobb/isy/2002/3322/exjobb.pdf

[Fra86] Frankel, M.: Tactical C3 for the Ground Forces. In: *Data Distribution in a Tactical Environment*. Washington, D.C. : AFCEA International Press, 1986

[GA02] Guerrero Zapata, Manel ; Asokan, N.: Securing Ad hoc Routing Protocols. In: *Proceedings of the 2002 ACM Workshop on Wireless Security (WiSe 2002)*, 2002, S. 1–10. – auch verfügbar unter http://doi.acm.org/10.1145/570681.570682

[Gei03] Geiss, Alfred: *Sicheres Routing in Mobilen Ad-Hoc Netzwerken*. Ulm, Germany, University of Ulm, Diplomarbeit, 2003

[GJKR96] Gennaro, Rosario ; Jarecki, Stanislav ; Krawczyk, Hugo ; Rabin, Tal: Robust and Efficient Sharing of RSA Functions. In: *Advances in Cryptology – CRYPTO '96*, 1996, S. 157–172

[GNY90] Gong, Li ; Needham, Roger M. ; Yahalom, Raphael: Reasoning About Belief in Cryptographic Protocols. In: Cooper, Deborah (Hrsg.) ; Lunt, Teresa (Hrsg.): *Proceedings of the IEEE 1990 Symposium on Research in Security and Privacy*, IEEE Computer Society, 1990, S. 234–248. – auch verfügbar unter http://citeseer.nj.nec.com/gong90reasoning.html

[GS96] Garfinkel, Simson ; Spafford, Gene: *Practical Unix & Internet Security*. 2nd. O'Reilly, 1996

[GT96] Gülcü, C. ; Tsudik, G.: Mixing Email with Babel. In: *1996 Symposium on Network and Distributed System Security*. San Diego, Februar 1996

[Gue02a] Guerrero Zapata, Manel: Secure Ad hoc On-Demand Distance Vector Routing. In: *ACM Mobile Computing and Communications Review (MC2R)* 6 (2002), Juli, Nr. 3, S. 106–107. – auch verfügbar unter http://doi.acm.org/10.1145/581291.581312

[Gue02b] Guerrero Zapata, Manel. *Secure Ad hoc On-Demand Distance Vector (SAODV) Routing*. http://ant.eupvg.upc.es/~tarom/draft-guerrero-manet-saodv-00.txt. August 2002

[Han03] Hansmann, Markus. *Location-based Services, Ausarbeitung im Rahmen des Proseminars Drahtlose Netze, SS 2003, Abt. Medieninformatik, Universität Ulm*. http://medien.informatik.uni-ulm.de/lehre/current/prosemdrahtlos/. 2003

[HBČ01] Hubaux, Jean-Pierre ; Buttyán, Levente ; Čapkun, Srđan: The Quest for Security in Mobile Ad Hoc Networks. In: *Proceeding of the ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHOC)*, 2001. – auch verfügbar unter http://citeseer.nj.nec.com/493788.html

[Hed88] Hedrick, C. *RFC 1058: Routing Information Protocol*. http://www.ietf.org/rfc/rfc1058.txt. Juni 1988

[HGBV01] Hubaux, J. P. ; Gross, Th. ; Boudec, J. Y. L. ; Vetterli, M.: Towards self-organized mobile ad hoc networks: the Terminodes project. In: *IEEE Communications Magazine* (2001), Januar

[HIP] *HiperLAN/2 Global Forum*. http://www.hiperlan2.com/

[HJKY95] Herzberg, Amir ; Jarecki, Stanislaw ; Krawczyk, Hugo ; Yung, Moti: PROACTIVE SECRET SHARING Or: How to Cope With Perpetual Leakage. In: *Lecture Notes in Computer Science* 963 (1995), S. 339ff.

[HJP02] Hu, Yih-Chun ; Johnson, David B. ; Perrig, Adrian: SEAD: Secure Efficient Distance Vector Routing for Mobile Wireless Ad Hoc Networks. In: *Proceedings of the 4th IEEE Workshop on Mobile Computing Systems and Applications (WMCSA 2002)*. Calicoon, NY, Juni 2002, S. 3–13

[HML91] Heberlein, T.L ; Mukhejee, B. ; Levitt, K.N.: A method to detect intrusive activity in a networked environment. In: *Proceedings of the Fourteenth National Computer Security Conference*. Washington, D.C., Oktober 1991, S. 362–371

[Hoa69] Hoare, C.A.R.: An Axiomatic Basis for Computer Programming. In: *Communications of the ACM* 12 (1969), Oktober, Nr. 10, S. 576–583

[HP98] Haas, Z.J. ; Pearlman, M.R.: Providing Ad-Hoc Connectivity with Reconfigurable Wireless Networks. In: *ACM SIGCOMM '98* (1998), September

[HPJ02] Hu, Yih-Chun ; Perrig, Adrian ; Johnson, David B.: Ariadne: A secure On-Demand Routing Protocol for Ad hoc Networks. In: *Proceedings of MobiCom 2002*. Atlanta, Georgia, USA, September 2002

[HPJ03] Hu, Yih-Chun ; Perrig, Adrian ; Johnson, David B.: Packet Leashes: A Defense against Wormhole Attacks in Wireless Ad Hoc Networks. In: *Proceedings of the Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2003)*. San Francisco, CA, April 2003. – to appear

[HPS02a] Haas, Z.J. ; Pearlman, M.R. ; Samar, P. *Bordercast Resolution Protocol (BRP)*. IETF Internet Draft, http://www.ietf.org/internet-drafts/draft-ietf-manet-brp-02.txt. Juli 2002

[HPS02b] Haas, Z.J. ; Pearlman, M.R. ; Samar, P. *Interzone Routing Protocol (IERP)*. IETF Internet Draft, http://www.ietf.org/internet-drafts/draft-ietf-manet-ierp-02.txt. Juli 2002

[HPS02c] Haas, Z.J. ; Pearlman, M.R. ; Samar, P. *Intrazone Routing Protocol (IARP)*. IETF Internet Draft, http://www.ietf.org/internet-drafts/draft-ietf-manet-iarp-02.txt. Juli 2002

[HPS02d] Haas, Z.J. ; Pearlman, M.R. ; Samar, P. *The Zone Routing Protocol (ZRP)*. IETF Internet Draft, http://www.ietf.org/internet-drafts/draft-ietf-manet-zone-zrp-04.txt. Juli 2002

[Hui00] Huitema, Christian: *Routing in the Internet*. 2nd. Prentice Hall, 2000

[ICP+99] Iwata, A. ; Chiang, C.-C. ; Pei, G. ; Gerla, M. ; Chen, T.-W.: Scalable Routing Strategies for Ad Hoc Wireless Networks. In: *IEEE Journal on Selected Areas of Communications* 17 (1999), August, Nr. 8, S. 1369–1379

[IEEa] *Institute of Electrical and Electronics Engineers, Inc.* http://www.ieee.org/

[IEEb] *IEEE 802 LAN/MAN Standards Committee*. http://grouper.ieee.org/groups/802/index.html/

[IEEc] *IEEE 802.11 Wireless Local Area Networks*. http://grouper.ieee.org/groups/802/11/index.html

[IEEd] *IEEE 802.11 - IEEE Standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications*. http://standards.ieee.org/getieee802/download/802.11-1999.pdf

[IEEe] *IEEE 802.11 - IEEE Standard for Information Technology - Telecommunications and information exchange between systems - Local and Metropolitan Area networks - Specific requirements - Part 11: Wireless*

LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications. http://standards.ieee.org/getieee802/download/802.11-1999.pdf

[IEEf] *IEEE 802.11a - IEEE Standard for Telecommunications and Information Exchange Between Systems - LAN/MAN Specific Requirements - Part 11: Wireless Medium Access Control (MAC) and physical layer (PHY) specifications: High Speed Physical Layer in the 5 GHz band.* http://standards.ieee.org/getieee802/download/802.11a-1999.pdf

[IEEg] *IEEE 802.11b - Corrigenda to IEEE 802.11b-1999, Information Technology - Telecommunications and Information Exchange Between Systems - Local and Metropolitan Networks - Specific Requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Higher Speed Physical Layer (PHY) Extension in the 2.4 GHz Band.* http://standards.ieee.org/cgi-bin/status?wireless

[IEEh] *IEEE 802.11b - IEEE Standard for Information Technology - Telecommunications and information exchange between systems - Local and Metropolitan networks - Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: Higher speed Physical Layer (PHY) extension in the 2.4 Ghz band.* http://standards.ieee.org/getieee802/download/802.11b-1999.pdf

[IEEi] *IEEE 802.11e - Amendment to STANDARD for Information Technology-Telecommunications and information exchange between systems-Local and metropolitan area networks-Specific requirements-Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: Medium Access Method (MAC) Quality of Service Enhancements.* http://standards.ieee.org/cgi-bin/status?wireless

[IEEj] *IEEE 802.11f - Trial-Use Recommended Practice for Multi-Vendor Access Point Interoperability Via an Inter-Access Point Protocol Across Distribution Systems Supporting IEEE 802.11 Operation.* New standard project, see http://standards.ieee.org/cgi-bin/status?wireless

[IEEk] *IEEE 802.11g - Supplement to STANDARD FOR Information Technology-Telecommunications and information exchange between systems-Local and Metropolitan networks- Specific requirements-Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: Further Higher data rate extension in the 2.4GHz band.* New standard project, see http://standards.ieee.org/cgi-bin/status?wireless

[IEEl] *IEEE 802.11h - STANDARD FOR Information Technology - Telecommunications and information exchange between systems - Local and Metropolitan networks - Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications, for Spectrum and Transmit Power Management extensions in the 5 GHz band in Europe.* http://standards.ieee.org/cgi-bin/status?wireless

[IEEm] *IEEE 802.11i - Amendment to STANDARD for Information Technology-Telecommunications and information exchange between systems-Local*

and metropolitan area networks-Specific requirements-Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: Medium Access Method (MAC) Security Enhancements. http://standards.ieee.org/cgi-bin/status?wireless

[Int99] *Pentium(R) III Processors - Serial Number*. Intel Corporation, URL: http://www.intel.com/support/processors/pentiumiii/psu.htm. 1999

[IPo] *IPonAir*. http://www.iponair.de/

[ISO84] Norm ISO 7498 1984. Basic Reference Model for Open Systems Interconnection

[ITU] *International Telecommunication Union*. http://www.itu.int/home/index.html

[IZM03] *IZMF | Mobilfunknutzung Deutschland 3*. Informationszentrum Mobilfunk. URL: http://www.izmf.de/html/de/1403.html. 2003

[JBH78] Jacobs, I. ; Binder, R. ; Hoversten, E.: General Purpose Packet Satellite Networks. In: *Proceedings of the IEEE* 66 (1978), November, Nr. 11, S. 1448–1467

[JDHC97] John D. Howard (CERT/CC, USA): *An Analysis of Security Incidents on the Internet 1988 - 1995*. Pittsburgh, PA, Carnegie Mellon University, Diss., 1997

[JLH+99] Johansson, Per ; Larsson, Tony ; Hedman, Nicklas ; Mielczarek, Bartosz ; Degermark, Mikael: Scenario-based performance analysis of routing protocols for mobile ad-hoc networks. In: *Proceedings of the fifth annual ACM/IEEE international conference on Mobile computing and networking*, ACM Press, 1999, S. 195–206. – auch verfügbar unter http://doi.acm.org/10.1145/313451.313535

[JMB01] Johnson, David B. ; Maltz, David A. ; Broch, Josh: DSR: The Dynamic Source Routing Protocol for Multi-Hop Wireless Ad Hoc Networks. In: Perkins, Charles E. (Hrsg.): *Ad Hoc Networking*. Addison-Wesley, 2001, Kapitel 5, S. 139–172

[JMHJ03] Johnson, David B. ; Maltz, David A. ; Hu, Yih-Chun ; Jetcheva, Jorjeta G. *The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR)*. http://www.ietf.org/internet-drafts/draft-ietf-manet-dsr-09.txt. April 2003

[JTA97] *Joint Technical Architecture, Version 5.0*. September 1997

[JW01] Jakobsson, Markus ; Wetzel, Susanne: Security Weaknesses in Bluetooth. In: *Lecture Notes in Computer Science* 2020 (2001), S. 176+. – auch verfügbar unter http://citeseer.nj.nec.com/jakobsson01security.html

[K+78] Kahn, R. [u. a.]: Advances in Packet Radio Technology. In: *Proceedings of the IEEE* 66 (1978), November, Nr. 11, S. 1468–496

[Kah91] Kahn, D.: *Seizing the Enigma*. Houghton Miffin Co., 1991

[Kar01] Kargl, Frank. *Praktikum Wireless Networks and Mobile Services, WS 2002/2003, Abt. Medieninformatik, Universität Ulm.* http://medien.informatik.uni-ulm.de/lehre/courses/ws0102/wireless/. 2001

[KBC97] Krawczyk, H. ; Bellare, M. ; Canetti, R. *RFC2104: HMAC: Keyed-Hashing for Message Authentication.* http://www.ietf.org/rfc/rfc2104. Februar 1997

[KDIW02] Kargl, Frank ; Dong, Bin ; Illmann, Torsten ; Weber, Michael: SmartReminder - Personal Assistance in a Mobile Computing Environment. In: *Workshop on Ad hoc Communications and Collaboration in Ubiquitous Computing Environments at ACM 2002 Conference on Computer Supported Cooperative Work (CSCW2002).* New Orleans, USA, November 2002

[KID+02] Kargl, Frank ; Illmann, Torsten ; Dong, Bin ; Geiss, Alfred ; Zeile, Matthias. *SmartReminder - Personal Assistance in a Mobile Computing Environment.* Demonstration at Pervasive2002, verfügbar als http://medien.informatik.uni-ulm.de/~frank/research/pervasive2002.pdf. August 2002

[KJJ99] Kocher, Paul C. ; Jaffe, Joshua ; Jun, Benjamin: Differential Power Analysis. In: *Lecture Notes in Computer Science* 1666 (1999), S. 388–397. – auch verfügbar unter http://citeseer.nj.nec.com/kocher99differential.html

[KKA03] Khalili, Aram ; Katz, Jonathan ; Arbaugh, William A.: Toward Secure Key Distribution in Truly Ad-Hoc Networks. In: *Symposium on Applications and the Internet Workshops (SAINT 2003)*, IEEE Computer Society, 2003, S. 342–346

[Kle03] Klenk, Andreas: *Mobile Intrusion Detection in Mobilen Ad-Hoc Netzwerken.* Ulm, Germany, University of Ulm, Diplomarbeit, 2003

[KMM94] Kemmerer, Richard ; Meadows, Catherine ; Millen, Jonathan: Three systems for cryptographic protocol analysis. In: *Journal of Cryptology* 7 (1994), Nr. 2, S. 79–130

[KMSW01] Kargl, F. ; Maier, J. ; Schlott, S. ; Weber, M.: Protecting Web Servers from Distributed Denial of Service Attacks. In: *Proceedings of WWW'01.* Hongkong, China, Mai 2001

[Ko96] Ko, Calvin: *Execution Monitoring of security-critical Programs in a Distributed System: A Specification-Based Approach*, UC Davis, Diss., 1996

[Koc96] Kocher, Paul C.: Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In: *Lecture Notes in Computer Science* 1109 (1996), S. 104–113. – auch verfügbar unter http://citeseer.nj.nec.com/kocher96timing.html

[Koc98] Kocher, Paul C.: On Certificate Revocation and Validation. In: *Proc. International Conference on Financial Cryptography* Bd. 1465, Springer-Verlag, 1998, S. 172–177

[KRL97] Ko, Calvin ; Ruschitzka, M. ; Levitt, K.: Execution Monitoring of Security-critical Programs in Distributed Systems: A Specification-based Approach. In: *Proceedings of the 1997 IEEE Symposium on Security and Privacy* Bd. 9. Oakland, CA : IEEE Computer Society Press, Mai 1997, S. 175–187

[KRSW03] Kargl, Frank ; Ribhegge, Stefan ; Schlott, Stefan ; Weber, Michael: Bluetooth-based Ad-Hoc Networks for Voice Transmission. In: *Proceedings of the 36th Hawaii International Conference on System Sciences (HICSS-36)*. Hilton Waikoloa Village, HA, Januar 2003

[KV98] Ko, Y. ; Vaidya, N.H.: Location-Aided Routing (LAR) in Mobile Ad Hoc Networks. In: *Proceedings of the Fifth annual ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM '98)*, 1998

[KZL+01] Kong, Jiejun ; Zerfos, Petros ; Luo, Haiyun ; Lu, Songwu ; Zhang, Lixia: Providing Robust and Ubiquitous Security Support for Mobile Ad-hoc Networks. In: *Ninth International Conference on Network Protocols (ICNP)*, 2001, S. 251–260. – auch verfügbar unter http://citeseer.nj.nec.com/kong01providing.html

[L+97] Leiner, B. [u. a.]: The Past and Future History of the Internet. In: *Communications of the ACM* 40 (1997), Februar, Nr. 2, S. 102–08

[Lam81] Lamport, Leslie: Password Authentication with Insecure Communication. In: *Communications of the ACM* 24 (1981), November, Nr. 11, S. 770–772

[Lau86] Lauer, G.S.: Hierarchical Routing Design for SURAN. In: *Proceedings of IEEE ICC '86*, 1986, S. 93–102

[LF01] Laura Feeney, Martin N.: Investigating the Energy Consumption of a Wireless Network Interface in an Ad Hoc Networking Environment. In: *INFOCOM 2001*. Swedish Institute of Computer Science, 2001

[LG97] Lin, C.R. ; Gerla, M.: Adaptive Clustering for Mobile Wireless Networks. In: *IEEE Journal on Selected Areas of Communications* 15 (1997), September, Nr. 7, S. 1265–1275

[LIM72] Lexikon-Institut, Bertelsmann (Hrsg.) ; Müller, Dr. Hans F. (Hrsg.): *Das moderne Lexikon*. Bd. 8. Gütersloh : Bertelsmann Lexikon-Verlag, 1972

[LKG86] Leiner, B. ; Klein, T. ; Graff, B.: Tactical C3 for the Ground Forces. In: *Data Distribution in a Tactical Environment*. Washington, D.C. : AFCEA International Press, 1986

[LL00] Luo, Haiyun ; Lu, Songwu: Ubiquitous and Robust Authentication Services for Ad Hoc Wireless Networks / Department of Computer Science, University of California at Los Angeles. 2000 (TR-200030). – Forschungsbericht

[LM90] Lai, X. ; Massey, J.: A proposal for a new block encryption standard. In: *Advances in Cryptography, EUROCRYPT '90*, Springer-Verlag, 1990, S. 389–404

[LNT87a] Leiner, B. ; Nielson, D. ; Tobagi, F.: Issues in Packet Radio Network Design. In: *Proceedings of the IEEE (Special Issue, Packet Radio Networks)* 75 (1987), Januar, Nr. 1, S. 6–20

[LNT87b] Leiner, B. ; Nielson, D. ; Tobagi, F.: Scanning the Issue. In: *Proceedings of the IEEE (Special Issue, Packet Radio Networks)* 75 (1987), Januar, Nr. 1, S. 3–5

[Low96] Lowe, G.: Breaking and fixing the Needham-Schroeder public-key protocol using FDR. In: *Tools and Algorithms for the Construction and Analysis of Systems* Bd. 1055, Springer, 1996

[LRS96] Leiner, B. ; Ruth, R. ; Sastry, A.R.: Goals and Challenges of the DARPA GloMo Program. In: *IEEE Personal Communications* (1996), Dezember, S. 34–3

[LSP82] Lamport, L. ; Shostak, R. ; Peace, M.: The Byzantine Generals Problem. In: *ACM Transactions on Programming Languages* 4 (1982), Juli, Nr. 3, S. 382–401

[LX01] Lee, Wenke ; Xiang, Dong: Information-theoretic measures for anomaly detection. In: *Proc. of the 2001 IEEE Symposium on Security and Privacy*, 2001, S. 130–143. – auch verfügbar unter http://citeseer.nj.nec.com/lee01informationtheoretic.html

[LZK+02] Luo, Haiyun ; Zefros, Petros ; Kong, Jiejun ; Lu, Songwu ; Zhang, Lixia: Self-securing Ad Hoc Wireless Networks. In: *Seventh IEEE Symposium on Computers and Communications (ISCC)*, 2002. – auch verfügbar unter http://citeseer.nj.nec.com/507663.html

[MAN] *Internet Engineering Task Force - Mobile Ad-hoc Networks (manet) Working Group.* http://www.ietf.org/html-charters/manet-charter.html

[Mar02] Marshall, Trevor. *Achieving the Best Compromise Between Workspace Coverage and Signal Security.* WLAN Security '02 Conference. Juni 2002

[MC02] Montenegro, Gabriel ; Castelluccia, Claude: *Statistically Unique and Cryptographically Verifiable (SUCV) Identifiers and Addresses.* Network and Distributed System Security Symposium (NDSS). Februar 2002. – auch verfügbar unter http://citeseer.nj.nec.com/montenegro02statistically.html

[MCF87] Millen, Jonathan K. ; Clark, Sidney C. ; Freedman, Sheryl B.: The Interrogator: Protocol Security Analysis. In: *IEEE Transactions on Software Engineering* 13 (1987), Nr. 2, S. 274–288

[Mea96] Meadows, Catherine A.: The NRL Protocol Analyzer: An Overview. In: *Journal of Logic Programming* 26 (1996), Nr. 2, S. 113–131. – auch verfügbar unter http://citeseer.nj.nec.com/meadows96nrl.html

[MGLB00] Marti, Sergio ; Giuli, T. J. ; Lai, Kevin ; Baker, Mary: Mitigating routing misbehavior in mobile ad hoc networks. In: *Mobile Computing and Networking*, 2000, S. 255–265. – auch verfügbar unter http://citeseer.nj.nec.com/marti00mitigating.html

[Mic96] Micali, Silvio: Efficient Certificate Revocation / MIT Laboratory for Computer Science. 1996 (TM-542b). – Forschungsbericht. – 10 S. auch verfügbar unter http://citeseer.nj.nec.com/article/micali96efficient.html

[Mil67] Milgram, Stanley: The small world problem. In: *Psychology Today* 1 (1967), S. 60–67

[MKK98] Massey, J.L. ; Khachatrian, G.H. ; Kuregian, M.K.: SAFER+ Candidate Algorithm for AES - Submission Document / Cylink Corp. 1998. – Forschungsbericht

[Mül] Müller, Burkhard. *Funknetze im Überblick*. http://www.tecchannel.de/special/1047/index.html

[MM] Michiardi, Pietro ; Molva, Refik. *Prevention of Denial of Service attacks and Selfishness in Mobile Ad Hoc Networks*. http://www.eurecom.fr/ michiard/pub/michiardi_adhoc_dos.ps

[MM02] Michiardi, Pietro ; Molva, Refik: A Collaborative Reputation mechanism to enforce node cooperation in Mobile Ad Hoc Networks. In: *Proceedings of the 6th IFIP Communication and Multimedia Security Conference*. Portorosz, Slovenia, September 2002

[MM03] Michiardi, Pietro ; Molva, Refik: A Game Theoretical Approach to Evaluate Cooperation Enforcement Mechanisms in Mobile Ad hoc Networks. In: *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt'03)*. Sophia-Antipolis, France, März 2003

[Moy91] Moy, J. *RFC 1247: OSPF Version 2*. http://www.ietf.org/rfc/rfc1247.txt. Juli 1991

[MR02] Murphy, S. ; Robshaw, M.J.B.: Essential Algebraic Structure within the AES. In: *Advances in Cryptology - Proceedings of the 22nd Annual International Cryptology Conference (CRYPTO 2002)*, Springer, 2002 (Lecture Notes in Computer Science 2442), S. 1–16

[MRR80] McQuillan, J.M. ; Richer, I. ; Rosen, E.C.: The New Routing Algorithm for the ARPANET. In: *IEEE Transactions on Communications* 28 (1980), Mai, Nr. 5, S. 711–719

[MW77] McQuillan, J.M. ; Walden, D.C.: The ARPA Network Design Decisions. In: *Computer Networks* 1 (1977), August, S. 243–289

[MWH01] Mauve, M. ; Widmer, J. ; Hartenstein, H.: A survey on position-based routing in mobile ad hoc networks. In: *IEEE Network Magazine* 15 (2001), November, Nr. 6, S. 30–39. – auch verfügbar unter http://citeseer.nj.nec.com/article/mauve01survey.html

[Nas50] Nash, John: Equilibrium points in n-person games. In: *Proceedings of the National Academy of Sciences* Bd. 36, 1950, S. 48–49

[Nes90] Nessett, Dan M.: A Critique of the Burrows, Abadi and Needham Logic. In: *ACM Operating Systems Review* 24 (1990), Nr. 2, S. 35–38

[New01] Newsham, Tim. *Cracking WEP Keys*. Talk at Black Hat Briefings. Verfügbar als http://www.lava.net/~newsham/wlan/WEP_password_cracker.ppt. Juni 2001

[NN98] Naor, Moni ; Nissim, Kobbi: Certificate Revocation and Certificate Update. In: *Proceedings of the 7th USENIX Security Symposium (San Antonio, Texas)*, 1998. – auch verfügbar unter http://citeseer.nj.nec.com/naor98certificate.html

[NS78] Needham, R.M. ; Schroeder, M.D.: Using Encryption for Authentication in Large Networks of Computers. In: *Communications of the ACM* 21 (1978)

[NS2] *The Network Simulator - ns-2*. http://www.isi.edu/nsnam/ns/

[NT94] Neuman, B. C. ; Ts'o, Theodore: Kerberos: An Authentication Service for Computer Networks. In: *IEEE Communications Magazine* 32 (1994), Nr. 9, S. 33–38

[Nyq24] Nyquist, H.: Certain factors affecting telegraph speed. In: *Bell System Technical Journal* 3 (1924), S. 324–346

[Opi02] Opitz, Rudolf: Location Based Services mit Sprachsteuerung für Smartphones. In: *heise news* (2002), Oktober

[OR87] Otway, D. ; Rees, O.: Efficient and timely mutual authentication. In: *Operating system review* 21 (1987), Januar

[PB94] Perkins, Charles E. ; Bhagwat, Pravin: Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers. In: *ACM SIGCOMM* 24 (1994), Oktober, Nr. 4. – auch verfügbar unter http://www.iprg.nokia.com/charliep/txt/sigcomm94/paper.ps

[PCB00] Priyantha, Nissanka B. ; Chakraborty, Anit ; Balakrishnan, Hari: The Cricket location-support system. In: *Mobile Computing and Networking*, 2000, S. 32–43. – auch verfügbar unter http://citeseer.nj.nec.com/priyantha00cricket.html

[PCST01] Perrig, Adrian ; Canetti, Ran ; Song, Dawn ; Tygar, Doug: Efficient and secure source authentication for multicast. In: *Network and Distributed System Security Symposium, NDSS 01*, 2001

[PCTS00] Perrig, Adrian ; Canetti, Ran ; Tygar, J.D. ; Song, Dawn X.: Efcient authentication and signing of multicast streams over lossy channels. In: *IEEE Symposium on Security and Privacy*, 2000

[PCTS02] Perrig, Adrian ; Canetti, Ran ; Tygar, J.D. ; Song, Dawn: The TESLA Broadcast Authentication Protocol. In: *RSA CryptoBytes* 5 (Summer) (2002)

[Per01] Perkins, Charles E. (Hrsg.): *Ad Hoc Networking*. Addison-Wesley, 2001

[PGP03] PGP Corporation. *Whitepaper: An Introduction to Cryptography*. http://pgp.com/products/whitepapers/pgp_introtocryptography.pdf. 2003

[PH02a] Papadimitratos, P. ; Haas, Z.J.: Performance Evaluation of Secure Routing Protocol for Mobile Ad Hoc Networks. In: *First ACM Workshop on Wireless Security (WiSe), Poster Session, in conjunction with ACM MobiCom 2002*, 2002. – auch verfügbar unter http://people.cornell.edu/pages/pp59/Docs/wise02.pdf

[PH02b] Papadimitratos, Panagiotis ; Haas, Zygmunt J.: Secure Routing for Mobile Ad hoc Networks. In: *SCS Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS 2002)*. San Antonio, TX, Januar 2002. – auch verfügbar unter http://wnl.ece.cornell.edu/Publications/cnds02.pdf

[PH02c] Papadimitratos, Panagiotis ; Haas, Zygmunt J. *Secure Routing for Mobile Ad Hoc Networks*. Working Session on Security in Wireless Ad Hoc Networks, EPFL, (published in Mobile Computing and Communications Review, vol.6, no.4). Juni 2002

[PH02d] Papadimitratos, Panagiotis ; Haas, Zygmunt J.: Securing Mobile Ad Hoc Networks. In: Ilyas, M. (Hrsg.): *Handbook of Ad Hoc Wireless Networks*. CRC Press, 2002

[PH03] Papadimitratos, Panagiotis ; Haas, Zygmunt J.: Secure Link State Routing for Mobile Ad Hoc Networks. In: *IEEE Workshop on Security and Assurance in Ad hoc Networks, in conjunction with the 2003 International Symposium on Applications and the Internet*. Orlando, FL, Januar 2003

[PHS02] Papadimitratos, Panagiotis ; Haas, Zygmunt J. ; Samar, P. *The Secure Routing Protocol (SRP) for Ad Hoc Networks*. draft-papadimitratos-secure-routing-protocol-00.txt. Dezember 2002

[PMW+01] Perkins, Charles E. ; Malinen, Jari T. ; Wakikawa, Ryuji ; Belding-Royer, Elizabeth M. ; Sun, Yuan. *IP Address Autoconfiguration for Ad Hoc Networks*. http://www.watersprings.org/pub/id/draft-perkins-manet-autoconf-01.txt. 2001

[PR99] Perkins, Charles E. ; Royer, Elizabeth M.: Ad hoc On-Demand Distance Vector Routing. In: *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*. New Orleans, LA, Februar 1999. – auch verfügbar unter http://www.cs.ucsb.edu/~ebelding/txt/aodv.ps

[PRD03] Perkins, Charles E. ; Royer, Elizabeth M. ; Das, S. *RFC 3561: Ad Hoc On Demand Distance Vector (AODV) Routing*. http://www.ietf.org/rfc/rfc3561. Juli 2003

[PSW+01] Perrig, Adrian ; Szewczyk, Robert ; Wen, Victor ; Culler, David E. ; Tygar, J. D.: SPINS: security protocols for sensor networks. In: *Mobile Computing and Networking*, 2001, S. 189–199. – auch verfügbar unter http://citeseer.nj.nec.com/perrig01spins.html/

[Qua00] Quayam, Amir: *Analysis and evaluation of channel access schemes and routing protocols for wireless networks*. Orsay, France, University of Paris-Sud, Diss., November 2000

[RBS87] Ramamoorthy, C. ; Bhide, A. ; Srivastava, J.: Reliable Clustering Techniques for Large, Mobile Packet Radio Networks. In: *Proceedings of IEEE INFOCOM '87*, 1987, S. 218–226

[Rib02] Ribhegge, Stefan: *Sprachübertragung in Bluetooth-basierten Ad-Hoc Netzwerken*. Ulm, Germany, University of Ulm, Diplomarbeit, 2001/2002

[Riv92a] Rivest, R.L. *The RC4 Encryption Algorithm*. RSA Data Security Inc. (proprietary). März 1992

[Riv92b] Rivest, Ron L. *RFC1321: The MD5 Message Digest Algorithm*. http://www.ietf.org/rfc/rfc1321.txt. April 1992

[RL03] Ramaswamy, L. ; Liu, L.: Free Riding: A New Challenge to Peer-to-Peer File Sharing Systems. In: *Proceedings of the Thirty-Sixth Annual Hawaii International Conference on System Sciences (HICSS-36)*, 2003

[RP] Royer, Elizabeth M. ; Perkins, Charles E. *Multicast Ad hoc On-Demand Distance Vector (MAODV) Routing*. http://www.watersprings.org/pub/id/draft-ietf-manet-maodv-00.txt

[RR98] Reiter, Michael K. ; Rubin, Aviel D.: Crowds: anonymity for Web transactions. In: *ACM Transactions on Information and System Security* 1 (1998), Nr. 1, S. 66–92. – auch verfügbar unter http://citeseer.nj.nec.com/284739.html

[RS98] Ramanathan, R. ; Streenstrup, M.: Hierarchically Organized, Multihop Mobile Wireless Networks for Quality-of-Service Support. In: *ACM/Baltzer Mobile Networks and Applications Journal* 3 (1998), Januar, Nr. 1, S. 101–119

[RSA78] Rivest, R.L. ; Shamir, A. ; Adleman, L.M.: A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. In: *Communications of the ACM* 21 (1978), Nr. 2, S. 120–126

[RT99] Royer, E.M. ; Toh, C.-K.: A Review of Current Routing Protocols for Ad-Hoc Mobile Networks. In: *IEEE Personal Communications* 6 (1999), April, Nr. 2, S. 46–55

[Röt00] Rötzer, Florian: Echelon gerät unter Beschuss. In: *Telepolis - Magazin der Netzkultur* (2000), Februar. – http://www.heise.de/tp/deutsch/special/ech/6637/1.html

[Rut98] Ruth, R. *Global Mobile Information Systems Program Overview*. Juli 1998

[SA99] Stajano, Frank ; Anderson, Ross: The Resurrecting Duckling: Security Issues for Ad-hoc Wireless Networks. In: Christianson, B. (Hrsg.) ; Crispo, B. (Hrsg.) ; Roe, M. (Hrsg.): *Security Protocols, 7th International Workshop Proceedings*, 1999, S. 172–194

[Sas99] Sass, P.: Communications Networks for the FORCE XXI Digitized Battlefield. In: *ACM/Baltzer Mobile Networks and Applications Journal (Special Issue, Mobile Ad Hoc Networking)* 4 (1999), Oktober

[Sch96] Schneier, Bruce W.: *Applied Cryptography*. Wiley, 1996

[Sch99] Schneier, Bruce: Modeling security threats. In: *Dr Dobb's Journal* (1999), Dezember. – auch verfügbar unter http://www.ddj.com/documents/s=896/ddj9912a/9912a.htm

[Sch00] Schneier, Bruce: *Secrets & Lies*. Wiley, 2000

[Sch02] Schäfer, Günter. *Network Security & IEEE 802.11 Wireless LANs*. Tutorial at WLAN Security '02 Conference. Juni 2002

[SCTS98] Staniford-Chen, S. ; Tung, B. ; Schnackenberg, D.: The common intrusion detection framework (CIDF). In: *Information Survivability Workshop*. Orlando, FL, Oktober 1998

[SDL+02] Sanzgiri, Kimaya ; Dahill, Bridget ; Levine, Brian N. ; Shields, Clay ; Belding-Royer, Elizabeth M.: A Secure Routing Protocol for Ad Hoc Networks. In: *Proceedings of 2002 IEEE International Conference on Network Protocols (ICNP)*, 2002. – auch verfügbar unter http://signl.cs.umass.edu/pubs/aran.icnp02.ps

[SGR97] Syverson, P.J. ; Goldschlag, D.M. ; Reed, M.G.: Anonymous connections and onion routing. In: *IEEE Symposium on Security and Privacy*, 1997

[Sha48] Shannon, C.E.: A mathematical theory of communication. In: *Bell Sys. Tech. Journal* 27 (1948), S. 379–423,623–656

[Sha49] Shannon, C. E.: Communication theory of secrecy systems. In: *Bell System Technical Journal* 28 (1949), S. 656–715

[Sha79] Shamir, A.: How to share a secret. In: *Communications of the ACM* 24 (1979), Nr. 11, S. 612–613

[Sha84a] Shacham, N.: Organization of Dynamic Radio Networks by Overlapping Clusters: Architecure Considerations and Optimizations. In: *Performance '84*, 1984, S. 435–447

[Sha84b] Shamir, Adi: Identity-based cryptosystems and signature schemes. In: *Proceedings of CRYPTO'84*, 1984, S. 47–53

[Sha85] Shacham, N.: Hierarchical Routing in Large, Dynamic Ground Radio Networks. In: *Proceedings of the Eighteenth Hawaii International Conference on System Sciences*, 1985, S. 292–301

[Sha96] Sharony, J.: An Architecture for Mobile Radio Networks with Dynamically Changing Topology Using Virtual Subnets. In: *ACM Mobile Networks and Applications (MONET)* 1 (1996), August, Nr. 1, S. 75–86

[Sho97] Shoup, Victor: Lower Bounds for Discrete Logarithms and Related Problems. In: *Lecture Notes in Computer Science* 1233 (1997). – http://citeseer.nj.nec.com/shoup97lower.html

[Sik] Sikora, A. *DECT - Die Alternative zu Bluetooth*. http://www.tecchannel.de/hardware/511/index.html

[Sik01] Sikora, Axel: *Wireless LAN - Protokolle und Anwendungen*. Addison-Wesley Verlag, 2001

[Sil00] Silverman, R. *A Cost-Based Security Analysis of Symmetric and Assymetric Key Lengths*. RSA Bulletin 13: http://www.rsasecurity.com/rsalabs/bulletings/bulletin13.html. 2000

[Sim85] Simmons, Gustavus J.: How to (Selectively) Broadcast a Secret. In: *Proceedings of the 1985 IEEE Symposium on Security and Privacy*, 1985, S. 108–113

[SIR01] Stubblefield, A. ; Ioannidis, J. ; Rubin, A.: Using the Fluhrer, Mantin, and Shamir Attack to Break WEP / ATT Labs. 2001 (TD4ZCPZZ, Revision 2). – Forschungsbericht. auch verfügbar unter http://citeseer.nj.nec.com/stubblefield01using.html

[SPCK02] Shamir, Adi ; Patarin, Jacques ; Courtois, Nicolas ; Klimov, Alexander: Efficient Algorithms for solving Overdefined Systems of Multivariate Polynomial Equations. In: *Proceedings of Eurocrypt 2000*, Springer, 2002 (Lecture Notes in Computer Science)

[Spe03] Specht, Raimund: *Authentifikation und Schlüsselaustausch in Mobilen Ad-Hoc Netzwerken*. Ulm, Germany, University of Ulm, Diplomarbeit, 2003

[SS80] Schwartz, M. ; Stern, T.: Routing Techniques Used in Computer Communication Networks. In: *IEEE Transactions on Communications* 28 (1980), April, S. 539–552

[SSG97] Syverson, Paul F. ; Stubblebine, Stuart G. ; Goldschlag, David M.: Unlinkable Serial Transactions. In: *Financial Cryptography*, 1997, S. 39–56. – auch verfügbar unter http://citeseer.nj.nec.com/syverson97unlinkable.html

[SSS02] Singh, H. ; Saxena, S. ; Singh, S.: Comparison of ECN-ELFN and SACK on TCP's performance for ad hoc networks. In: *Proceedings of the 5th ACM international workshop on Modeling analysis and simulation of wireless and mobile systems*, ACM Press, 2002, S. 38–45. – auch verfügbar unter http://doi.acm.org/10.1145/570758.570766

[Sta] Stadler, Thomas. *Das Recht auf informationelle Selbstbestimmung*. http://www.afs-rechtsanwaelte.de/volkszaehlung.htm

[Sta77] of Standards, National B.: Data encryption standard / Federal Information Processing Standard, U.S. Department of Commerce. Washington, D.C., 1977 (46). – FIPS PUB

[Sta94] of Standards, National I. *Digital Signature Standard*. NIST FIPS PUB 186. Mai 1994

[Sta97] of Standards, National B.: Announcing Request for Candidate Algorithm Nominations for the Advanced Encryption Standard (AES) / U.S. Department of Commerce. Washington, D.C., 1997 (0693-ZA16). – RIN

[Sta00] Stajano, Frank: The Resurrecting Duckling - What Next? In: Christianson, B. (Hrsg.) ; Crispo, B. (Hrsg.) ; Roe, M. (Hrsg.): *Security Protocols, 8th International Workshop Proceedings*, 2000, S. 204ff.

[Sta03] Stallings, William: *Cryptography and Network Security*. 3rd. Prentice Hall, 2003

[Sti00] Stiller, Andreas: Bei Lichte betrachtet – Die Architektur des Pentium 4 im Vergleich zu Pentium III und Athlon. In: *c't Magazin für Computertechnik* (2000), Nr. 24, S. 134ff.. – Heise Zeitschriften Verlag, Hannover

[Suna] Sun Microsystems. *Graph Layout Demonstration Applet*. http://java.sun.com/applets/jdk/1.4/demo/applets/GraphLayout/Graph.java

[Sunb] Sun Microsystems. *Java Cryptography Extension (JCE)*. http://java.sun.com/products/jce/

[Sunc] Sun Microsystems. *Java Homepage*. http://java.sun.com/

[SW87] Schacham, N. ; Westcott, J.: Future Directions in Packet Radio Architectures and Protocolls. In: *Proceedings of the IEEE (Special Issue, Packet Radio Networks)* 75 (1987), Januar, Nr. 1, S. 83–9

[Tan96] Tannenbaum, Andrew S.: *Computer Networks*. Prentice Hall PTR, 1996

[TCP] *Trusted Computing Platform Alliance*. http://www.trustedcomputing.org/tcpaasp4/index.asp

[TN98] Thomson, Susan ; Narten, Thomas. *RFC2462: IPv6 Stateless Address Autoconfiguration*. http://www.ietf.org/rfc/rfc2462.txt. 1998

[Toh97a] Toh, C.-K.: Associativity-Based Routing for Ad-Hoc Mobile Networks. In: *Journal on Wireless Personal Communication* 4 (1997), März, Nr. 2

[Toh97b] Toh, C.-K.: *Wireless ATM and Ad Hoc Networks: Protocols and Architectures*. Norwood, Mass. : Kluwer Academic Publishers, 1997

[Toh02] Toh, C.-K.: *Ad Hoc Mobile Wireless Networks: Protocols and Systems*. Prentice Hall PTR, 2002

[Tun99] Tung, Brian. *CIDF Homepage*. http://www.isi.edu/gost/cidf/. 1999

[USA02] USA Today. *Military network technology passing into civilian use*. http://www.usatoday.com/tech/news/techinnovations/2002-12-10-mesh-network_x.htm. 2002

[Wal00a] Walke, B.: *Mobilfunknetze und ihre Protokolle, Band 1 und 2*. Teubner, 2000

[Wal00b] Walker, Jesse: Unsafe at any key size; An Analysis fo the WEP encapsulation / IEEE. 2000 (802.11-00/362). – Forschungsbericht. http://grouper.ieee.org/groups/802/11/Documents/DocumentHolder/0-362.zip

[Way97] Wayner, Peter: *Digital Cash: Commerce on the Net*. 2nd. Morgan Kaufmann, 1997

[Wis90] Wissenschaftlicher Rat der Dudenredaktion: *Duden Fremdwörterbuch*. 5. Dudenverlag, 1990

[WMP+02] Wakikawa, Ryuji ; Malinen, Jari T. ; Perkins, Charles E. ; Nilsson, Anders ; Tuominen, Antti J. *Global connectivity for IPv6 Mobile Ad Hoc Networks*. http://www.wakikawa.net/Research/drafts/draft-wakikawa-manet-globalv6-02.txt. 2002

[WS98] Watts, Duncan J. ; Strogatz, Steven H.: Collective dynamics of small-world networks. In: *Nature* 393 (1998), S. 440–442

[Wät02] Wätjen, Dietmar. *Kryptologie*. Vorlesungsskript. Technische Universität Braunschweig, Institut für Theoretische Informatik. http://www.iti.cs.tu-bs.de/~waetjen/krypto.ps. Oktober 2002

[YK02] Yi, Seung ; Kravets, Robin: Key Management for Heterogeneous Ad Hoc Wireless Networks / University of Illinois. 2002 (UIUCDCS-R-2002-2290, UILU-ENG-2002-1734). – Forschungsbericht

[ZH99] Zhou, Lidong ; Haas, Zygmunt J.: Securing Ad Hoc Networks. In: *IEEE Network* 13 (1999), Nr. 6, S. 24–30. – auch verfügbar unter http://citeseer.nj.nec.com/zhou99securing.html

[Zhu] Zhu, Feng. *Paper list: Security for Ad hoc Networks*. http://www.ccs.neu.edu/home/zhufeng/security_manet.html

[Özk03] Özkilic, Murat: Auskunfts- und Suchdienste fürs Handy. In: *heise news* (2003), März

[ZL00] Zhang, Yongguang ; Lee, Wenke: Intrusion detection in wireless ad-hoc networks. In: *Mobile Computing and Networking*, 2000, S. 275–283. – auch verfügbar unter http://citeseer.nj.nec.com/zhang00intrusion.html

[ZLH03] Zhang, Yongguang ; Lee, Wenke ; Huang, Yi-An: Intrusion Detection Techniques for Mobile Wireless Networks. In: *to appear in ACM Wireless Networks (WINET)* 9 (2003). – auch verfügbar unter http://www.wins.hrl.com/people/ygz/papers/winet03.pdf

D. Danksagung

Eine Arbeit wie diese entsteht nicht im leeren Raum, vielmehr bedarf es eines entsprechenden Umfeldes, in dem Ideen wachsen und gedeihen können. Und so möchte ich mich an dieser Stelle bei einer Vielzahl von Leuten bedanken, die alle auf ihre Weise zum Gelingen dieser Dissertation beigetragen haben.

Zunächst gilt mein Dank dem Betreuer meiner Arbeit, Prof. Michael Weber. Unter seiner Anleitung habe ich im Verlauf meiner Tätigkeit in den Abteilungen Verteilte Systeme und Medieninformatik die Freude an der wissenschaftlichen Forschung und Lehre entdeckt, die mich hoffentlich noch viele Jahre begleiten wird. Ebenso gilt mein Dank dem zweiten Gutachter meiner Arbeit, Prof. Jörg Kaiser.

Ich möchte mich auch bei meinen Kollegen in der Abteilung Medieninformatik bedanken, insbesondere bei Torsten Illmann und Stefan Schlott, die mir immer für Diskussionen zur Verfügung standen.

Weiterhin gilt mein Dank auch den vielen Studenten, mit denen ich in den letzten Jahren zusammengearbeitet habe. Vor allem sind hier Alfred Geiß, Joachim Karrer, Andreas Klenk und Raimund Specht zu nennen. Die Ergebnisse ihrer Diplomarbeiten zu verschiedenen Themen der Sicherheit von Ad hoc Netzen sind an vielen Stellen in diese Arbeit eingeflossen. Am Wertvollsten empfand ich aber unsere wöchentlichen Diskussionsrunden, in denen viele neue Ideen entstanden, verfeinert oder wieder verworfen wurden.

Schließlich gilt mein größter Dank meiner Familie. Meinen Eltern, die mir den Weg dahin geebnet haben, wo ich heute stehe. Meiner Frau Birgit, die während unseres Hausbaus und Umzugs unglaublich hart gearbeitet hat, damit ich den Freiraum zur Fertigstellung meiner Dissertation hatte. Auch ihre Anmerkungen zu meiner Arbeit waren immer sehr hilfreich. Und letztlich möchte ich mich bei meinem Sohn Julian bedanken, der mir mit seiner Liebe und Zuneigung jeden Tag ein bisschen schöner macht.

E. Curriculum vitae

Name:	Frank Kargl
Eltern:	Günther Kargl, Versandleiter
	Sigrid Kargl, geb. Dülk, Bankfachwirtin
Geboren:	11. Februar 1972 in Werneck bei Schweinfurt
Sept. 1978 bis Juli 1982	Grundschule Dittelbrunn
Sept. 1982 bis Juni 1991	Alexander von Humboldt Gymnasium, Schweinfurt
	Abschluß: Abitur
	Leistungskurse: Mathematik und Biologie
Okt. 1991 bis Juni 1997	Studium der Informatik
	an der Universität Ulm
	Abschluß: Diplom am 11. Juni 1997
Juli 1997 bis Okt. 1998	Wissenschaftlicher Mitarbeiter
	am Universitätsrechenzentrum
	der Universität Ulm
Nov. 1998 bis Juni 2000	Wissenschaftlicher Mitarbeiter
	in der Abteilung Verteilte Systeme
	an der Universität Ulm
Juli 2000 bis Okt. 2003	Wissenschaftlicher Mitarbeiter
	in der Abteilung Medieninformatik
	an der Universität Ulm
Aug. 2002 bis Okt. 2003	Promotionsstudent
	an der Universität Ulm
21. Oktober 2003	Promotion zum Dr.rer.nat.
	an der Fakultät für Informatik
	der Universität Ulm
Seit Nov. 2003	Wissenschaftlicher Assistent
	in der Abteilung Medieninformatik
	an der Universität Ulm