Universität Karlsruhe (TH)
Institut für Telematik

TeleMatics Technical Reports

Mobiles Internet
Seminar SS04

Oliver Stanze, Peter Baumung, Tobias Küfner, Christian Vogt,
Jidong Wu, Martina Zitterbart
{stanze,baumung,kuefner,chvogt,wu,zit}@tm.uka.de

August, 26th 2004

TM-2004-7

ISSN 1613-849X

http://doc.tm.uka.de/tr/

Institute of Telematics, University of Karlsruhe
Zirkel 2, D-76128 Karlsruhe, Germany

Vorwort

Das Seminar "Mobiles Internet" wurde im Sommersemester 2004 in Form eines Blockseminars am 26.07.2004 im Institut für Telematik durchgeführt. Die Themen des Seminars stammten vor allem aus den Bereichen "Mobile IP" und "Mobile Ad-hoc-Netze". In diesem Seminarband werden nun die Ausarbeitungen der Studenten in Form eines Internen Berichts zusammengefasst. Folgende Themengebiete werden in diesem Internen Bericht behandelt:

Mobile IP

Protokolle zur Mobilitätsunterstützung, deren bekanntester Vertreter wohl Mobile IP ist, bieten Knoten die Möglichkeit sich in einem beliebigen Netz ans Internet anzuschliessen und dabei weiterhin über die IP-Adresse des Heimatnetzes erreichbar zu sein. Dazu muß der mobile Knoten beim Wechsel des Zugangspunktes zum Internet eine *Konfiguration einer neuen Adresse* durchführen und diese Adresse seinem Heimatagenten mitteilen. Dadurch ist der Heimatagent immer über den aktuellen Aufenthaltort eines mobilen Knotens informiert und kann Datenpakete, die an dessen Heimataddresse adressiert sind, zum mobilen Knoten weiterleiten. Es muß eine *sichere Kommunikation zwischen dem mobilen Knoten und dem Heimatagenten* stattfinden, damit Angreifer nicht in der Lage sind eine Registrierung des mobilen Knotenes vorzutäuschen und damit Datenpakete zu sich umzuleiten. Mobile IPv6 stellt einen Mechanismus zur sogenannten *Route Optimization* zur Verfügung. Dieser Mechanismus ermöglicht es das zwei Knoten direkt miteinander kommunizieren können, ohne daß der Heimagent involviert ist. Dadurch wird die Kommunikation mit mobilen Geräten wesentlich *effizienter*, allerdings treten dadurch auch neue *Sicherheitprobleme* auf, u.a. werden bei der *Route Optimization* spezielle Protokolle für die *Authentifizierung* der mobilen Endgeräte benötigt, z.B. *CAM (Child-Proof Authentication).*

NAT (Network Address Translation) ermöglicht es ganze Netzwerke über nur eine global gültige IP Adresse an das Internet anzubinden. Beim Zusammenspiel von *Mobile IP und NAT* treten jedoch einige Probleme auf, die zusätzliche Protokollmechanismen benötigen.

Mobile Ad-hoc-Netze

Mobile Ad-hoc-Netze sind infrastukturlose Netzwerke, die durch eine Gruppe von drahtlos miteinander kommunizierenden Geräten gebildet werden. In mobilen Ad-hoc-Netzen können zwei Knoten auch dann miteinander kommunizieren, wenn sie sich nicht in direkter Funkreichweite zueinander befinden. Andere im Netz befindliche Knoten fungieren als Router und leiten Datenpakete weiter. Aufgrund der Mobilität der einzelnen Knoten unterscheidet sich das Routing in mobilen Ad-hoc-Netzen erheblich vom Routing im klassischen Internet. Deshalb wurden bereits einige speziell für mobile Ad-hoc-Netze geeignete Routingprotokolle entwickelt. Diese Protokolle führen auf der Vermittlungsschicht das Routing durch. Mit *LUNAR* wurde ein Protokoll entwickelt, welches das Routing unterhalb der Vermittlungsschicht durchführt. Dadurch erscheint das Ad-hoc-Netz auf Vermittlungsschicht wie ein Broadcastsegment.

Da mobile Ad-hoc-Netzwerke meist von Knoten gebiltet werden, die in irgendeiner Form zusammenarbeiten, besteht ein hoher Bedarf an Gruppenkommunikation. Diese kann entweder in Form von Multicast-Routing oder in Form von *Endsystem-basierter Gruppenkommunikation* unterstützt werden.

TCP ist das meist verwendete Transportprotokoll im Internet. Der Einsatz von TCP in mobilen Ad-hoc-Netzwerken ist allerdings problematisch. Der Grund dafür ist die von TCP

verwendete *Staukontrolle*. Diese nimmt bei einem Paketverlust einen Stau als Ursache an und reduziert die Senderate. In mobilen Ad-hoc-Netzwerken sind allerdings Routenbrüche und nicht Stausituationen die Hauptursache für Paketverluste. Dies führt zu einem Fehlverhalten von TCP.

Protokolle für drahtlose Netzwerke werden meist erst im Simulator entwickelt und optimiert, bevor sie in der Realität eingesetzt werden. Um die Performanz eines Protokoll in einem bestimmten Szenario zu überprüfen muß dazu eine *Mobilitäts- und Benutzermodellierung* stattfinden.

Inhaltsverzeichnis

Vorwort i

Abilio Avila Albez:
Sichere Kommunikation mit dem Mobile-IPv6-Heimatagenten 3

Tobias Reisch:
Konfiguration einer neuen Adresse bei einem Mobile-IPv6-Handover 19

Daniel Jungbluth:
Authentifizierungs-Protokolle für Mobile IPv6 Route Optimization 31

Mustafa Yilmaz:
Mobile IP Version 6 Route Optimization: Effizienz vs. Sicherheit 41

Constantin Schimmel:
Child-Proof Authentication for Mobile IPv6 (CAM) 53

Fikri Tekin:
Mobile IPv4 und NAT 61

Maria Maleshkova:
LUNAR – A Lightweight Underlay Network Ad-hoc Routing 75

Inna Löwen:
Mobilitäts- und Benutzermodellierung für drahtlose Kommunikation 89

Andreas Süß:
Ansätze für drahtlose Endsysteme-basierte Gruppenkommunikation 105

Christian Biedermann:
TCP-Staukontrolle in drahtlosen Multi-hop-Netzwerken 117

Sichere Kommunikation mit dem Mobile-IPv6-Heimatagenten

Abilio Avila Albez

Kurzfassung

Gründe für die Verwendung von IPv6 und MIPv6 gibt es reichlich. IPv4 gerät langsam an seine Grenzen, da die vier Milliarden Adressen, die von IPv4 unterstützt werden, dem immer noch fortschreitenden Kommunikationsboom auf Dauer nicht mehr gerecht werden. IPv6 bietet mit 128 Bit langen IP-Adresssen hierfür eine Lösung und verfügt auch über weitere Features, wie z.B. eine vereinfachte Header-Struktur und verbesserte Unterstützung von Streaming-Programmen. Die Begeisterung ist groß. Mit MIPv6 sind allerdings nicht nur ausschließlich Vorteile verbunden. Es bietet auch neue Angriffsflächen und Schwachstellen, die von potentiellen Angreifern ausgenutzt werden können. Vor allem die Absicherung von mobilen Geräten bedarf besonderer Aufmerksamkeit. Hier treten Sicherheitsprotokolle, wie IPSec in Erscheinung, die versuchen diese Sicherheitslöcher zu stopfen. Im Verlauf meines Seminars werde ich zeigen, wie notwendig solche Sicherheitsmechanismen sind, um sicherheitsrelevante Daten gegenüber Angreifern abzusichern. In diesem Rahmen gehe ich speziell auf die Absicherung der Kommunikationsverbindung zwischen mobilen Knoten und deren Heimatagenten ein und werde zeigen, dass eine derartige Absicherung unumgänglich ist.

1 Mobile IPv6 Überblick

Mobile IPv6 ermöglicht mobilen Kommunikationseinheiten ihren Standort zu ändern, ohne, dass die aktuelle Kommunikationssitzung hierzu unterbrochen werden muss. Hierfür besitzt jeder mobile Knoten eine sogenanntes Heimatnetz. Das Heimatnetz ist das Netz seines ursprünglichen Anschlusses zu dem es aufgrund seine IP-Adresse gehört. Über diese IP-Adresse, der Heimatadresse, ist der mobile Knoten zu jeder Zeit erreichbar, unabhängig davon, ob er sich gerade in seinem Heimatnetz oder in einem anderen Netz (Fremdnetz) befindet. Solange sich der mobile Knoten in seinem Heimatnetz befindet, werden Pakete, die an seine (Heimat-)Adresse adressiert sind, unter Verwendung herkömmlicher Routingmechanismen weitergeleitet. Eine Unterstützung durch Mobile IP ist hier noch nicht notwendig. Bewegt er sich allerdings von seinem Heimatnetz in ein anderes Netz, dem Fremdnetz, wird ihm in diesem Fremdnetz eine (weitere) Adresse zugewiesen. Die sogenannte Zustelladresse. Solange sich der mobile Knoten in diesem fremden Netz befindet ist er über diese Adresse direkt erreichbar. (Abbildung 1)

Die Verbindung zwischen dieser Zustelladresse und der Heimatadresse wird als Binding (des mobilen Knoten) bezeichnet.

Befindet sich der mobile Knoten außerhalb seines Heimatnetzes, so muss er seine Zustelladresse bei einem Router in seinem Heimatnetz registrieren. Dieser Router übernimmt spezielle Aufgaben, auf die ich im weiteren Verlauf näher eingehen werde und wird als Heimatagent bezeichnet. Der mobile Knoten führt diese Registrierung durch, in dem er ein Binding Update an den Heimatagenten sendet. Mit diesem Binding Update wird dem Heimatagenten die aktuelle Zustelladresse mitgeteilt. Dieser bestätigt diese Nachricht, indem es ein Binding-Acknowledgement an den mobilen Konten schickt. Die Kommunikationspartner des mobilen

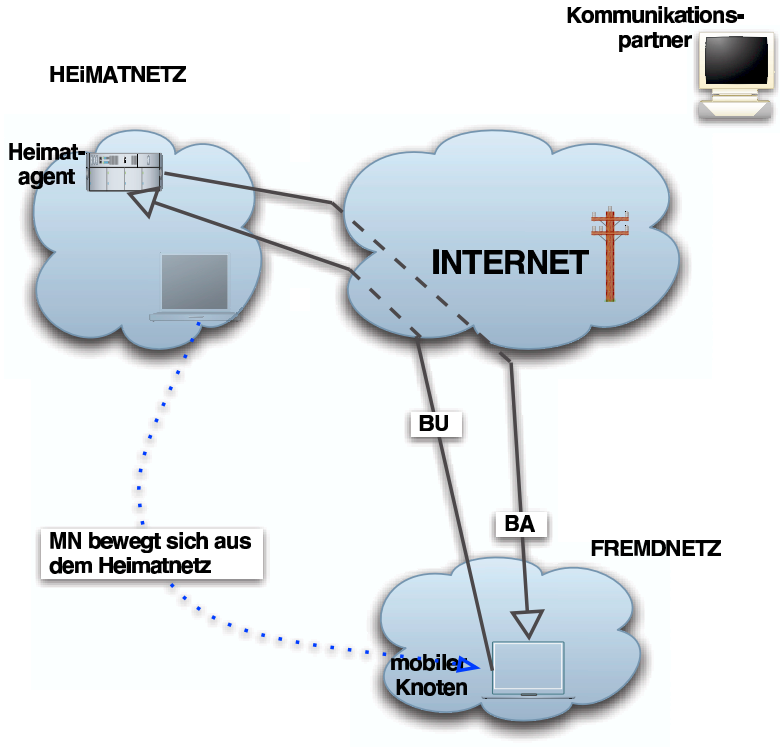


Abbildung 1: Mobiler Knoten bewegt sich aus seinem Heimatnetzt

Knoten (mobile node) werden als correspondent nodes bezeichnet Prinzipiell gibt es zwei Möglichkeiten der Kommunikation zwischen mobilen Knoten und einem Kommunikationspartner:

Beim bidirektionalen Tunneln werden Pakete, die vom Kommunikationspartner an den mobilen Knoten übertragen werden vom, Heimatagenten getunnelt, d.h die Pakete werden nicht direkt an mobilen Knoten geschickt sondern zunnachst an den Heimatagenten der sie dann an den mobilen Knoten weiterleitet. Pakete vom mobilen Knoten zum Kommunikationspartner werden ebenfalls vom Heimatagenten getunnelt (reverse tunneld). (Abbildung 2)

Es besteht auch die Möglichkeit Pakete direkt an die Zustelladresse des mobilen Knotens zu schicken. Man spricht dann von Route Optimierung. Hierzu wird dem Kommunikationspartner die Zustelladresse des mobilen Knotens mitgeteilt, so dass die Kommunikation ohne Umwege über den Heimatagenten erfolgen kann. (Abbildung 3)

Die Kommunikationspartner des mobilen Knotens speichern die erhaltenen Informationen des Binding Updates in ein Cache, dem Binding Cache. Bevor der Kommunikationspartner ein Paket an den mobilen Knoten sendet, überprüft er zunächst, ob sich in diesem Binding Cache ein Eintrag mit der zugehörigen gültigen Care-of-Adresse des mobilen Knoten befindet. Ist dies der Fall kann er die Pakete direkt an den mobilen Knoten senden. Andernfalls werden die Pakete, wie oben beschrieben, vom Heimatagenten abgefangen und zum mobilen Knoten getunnelt.

2 Return Routability Procedure

Die Return Routability Procedure erlaubt es dem Kommunikationspartne zu überprüfen, ob der mobile Knoten, sowohl über seine Heimatadresse, also auch über seine Zustelladresse erreichbar ist. Nur durch diese Überprüfung ist es dem Kommunikationspartner möglich

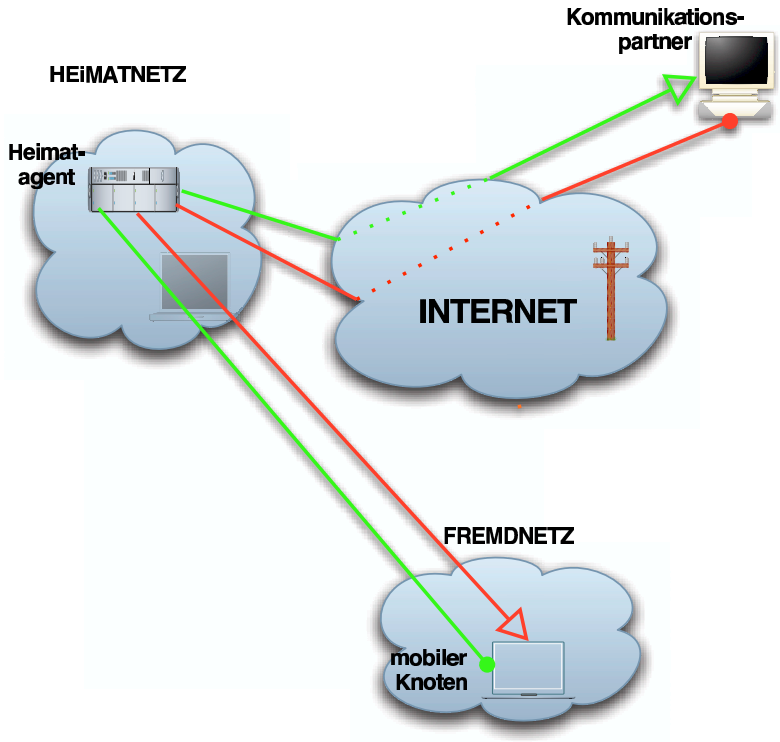


Abbildung 2: bidirektionales Tunneln

sicherzustellen das es sich bei den empfangene Binding Update um ein gültiges handelt. Diese Überprüfung erfolgt, indem getestet wird, ob bestimmte Daten (keygen tokens), die der Kommunikationspartner an den Mobilen Knoten, über beide Adressen sendet , tatsächlich ankommen.

Im ersten Schritt werden zwei Nachrichten, die Heimat-Test-Initiierungsnachricht und die Zustell-Test-Initiierungsnachricht gleichzeitig an den Kommunikationspartner geschickt. Durch diese Nachrichten wird der Kommunikationspartner aufgefordert jeweils einen keygen token an den mobilen Knoten zu senden. Jeder dieser Nachrichten enthält weiterhin ein sogennantes Init Cookie , ein 64-Bit Zufahlswert, der vom Kommunikationspartner an den mobilen Knoten zurückgeschickt wird, um die Identität des Kommunikationspartner zu bestätigen. Die Home-Test-Initiierung wird über den Heimatagenten getunnelt an den Kommunikationspartner geschickt, die Zustell-Test-Initiierung wird direkt an den Kommunikationspartner gesendet.

Im nächsten Schritt werden ebenfalls zwei Nachrichten verschickt, die Heimat-Test-Nachricht und die Zustell-Nachricht. Auch diese werden gleichzeitig verschickt. Dieses mal vom Kommunikationspartner an den Mobilen Knoten. Es handelt sich bei diesen Messages um Antworten auf die Initiierungsnachrichten, welche die bereits erwähnten keygen token enthalten. Diese Messages enthalten weiterhin den Init Cookie und einen nonce index, der dazu dient den für die Generierung des keygen token verwendeten nonce zu identifizieren.

Sobald der mobile Knoten, sowohl Heimat- also auch Zustell-Nachricht erhalten hat, ist die Return Routabilty Procedure beendet. Der mobile Knoten hat nun die notwendigen Daten um ein Binding Update zu versenden. Er erzeugt aus den Heimat-Keygen-Token und dem Zustell-Keygen-Token das sog. Binding Managment Key (KBM) mit dem er ein Binding Updates authorizieren kann. (Abbildung 4)

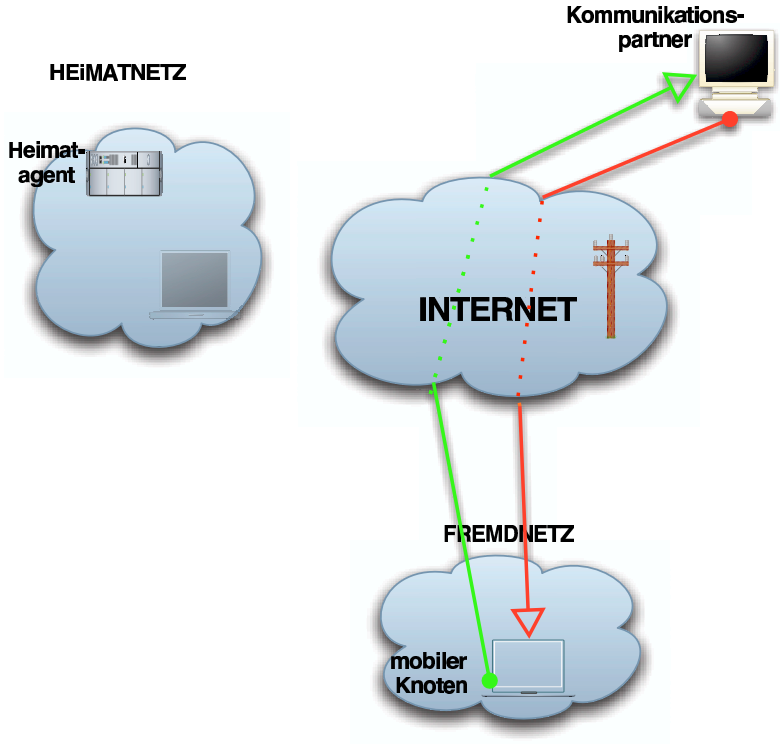


Abbildung 3: Route Optimization

Durch die Verwendung dieses Verfahrens entstehen allerdings auch Sicherheitsprobleme. Um die Return Routability Procedure sicherer zu machen, ist es erforderlich die Kommunikation zwischen mobilen Knoten und Heimatagente mittels IPSec abzusichern.

Bewegt sich der mobile Knoten in ein unsicheres Fremdnetz, z.B. in eine WLAN-Umgebung kann die Kommunikation zwischen dem mobilen Knoten (A) und seinem Heimatagenten leicht abgehört werden.

Im folgenden Angriffsszenario gehen wir von einer bestehenden Kommunikation zwischen dem mobilen Knoten A und einem weiteren Knoten B aus. Der mobile Knoten befindet sich, wie bereits erwähnt ausserhalb seines Heimatnetzes. Ein bösartiger Knoten (C) kann in diesem (unsicheren) Fremdnetz, die Daten die der Knoten A empfängt, abfangen. Der Knoten C kann nun eine Return Routability Procedur initiieren, indem er in der Heimat-Test-Initiierungsnachricht die Heimatadresse des Knoten A angibt und in der Zustell-Test-Initiierungsnachricht seine eigene Adresse als Zustell-Adresse angibt. Der Kommunikationspartner (B) sendet ihm daraufhin die Zustell-Test-Nachricht direkt an seine Zustelladresse. Da er die Kommuniktaion zum Knoten A überwacht, erhält er auch die Heimat-Test-Nachricht, so dass er nun in Besitz der beiden Keygen Token ist und in der Lage ist den Binding Managment Key zu erzeugen. (Abbildung 5)

Der angreifende Knoten C ist somit in der Lage ein gültiges Binding Update an den Kommunikationspartner zu senden, indem er angibt, dass die neue Zustelladresse des Knoten A seine eigene Adresse wäre. Somit hat er die Kommunikationsverbindung des mobilen Knotens übernommen. Man spricht in diesem Zusammenhang auch von Hijacking oder genauer gesagt von Verbindungs-Hijacking.

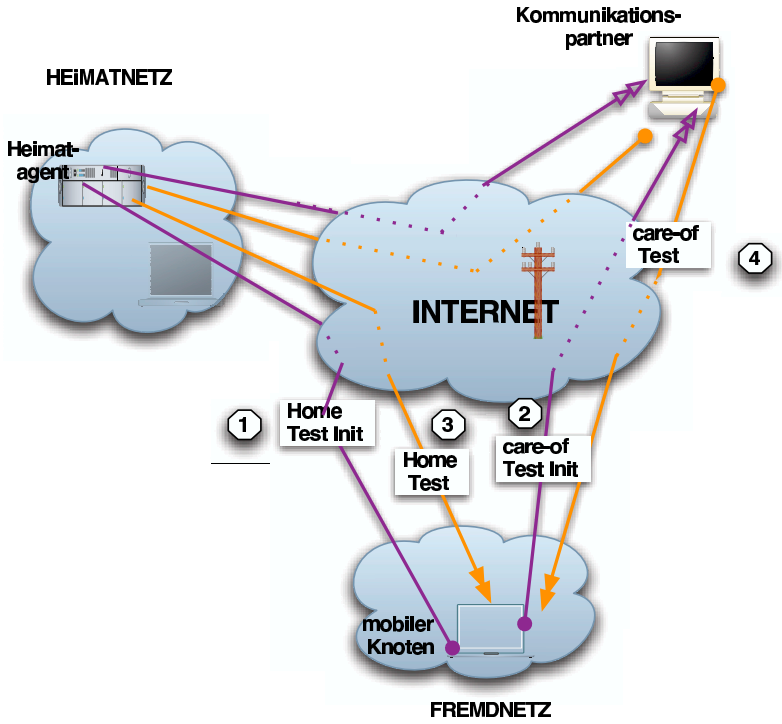


Abbildung 4: Return Routability Procedure

3 Mobile Prefix Discovery

Mobile Prefix Discovery

In Mobile IP werden neue ICMP-Nachrichten eingeführt. Ich möchte in diesem Abschnitt auf die Mobile-Präfix-Aufffforderungsnachricht und auf die Mobile-Präfix-Ankündigungsnachricht eingehen. Diese Nachrichten werden verwendet um Informationen über die Adresspräfixe des Heimatnetzes zwischen den mobilen Knoten und den Heimatagenten auszutauschen.

Der mobile Knoten verschickt die Mobile-Präfix-Aufforderungsnachricht an den Heimatagenten. Zweck dieser Nachricht ist es vom Heimatagenten eine Mobile-Präfix-Ankündigungsnachricht zu erhalten. Diese Nachricht enthält Informationen über die Präfixadresse des Heimatnetzes und kann verwendet werden, um die Heimatadresse entsprechend anzugleichen. Eine Mobile-Präfix-Ankündigungsnachricht kann auch unaufgefordert vom Heimatagenten an den Mobilen Knoten gesendet werden. Ändern sich im Heimatnetz die Präfixe und damit auch die Adresse, muss ein mobiler Knoten von dieser Änderung unterrichtet werden, auch wenn er sich gerade fern vom Heimatnetz befindet. Für diesen Zweck werden die oben erwähnten Nachrichten benötigt. (Abbildung 6)

Auch diese Kommunikation zwischen Mobile Knoten und den Heimatagenten muss geschützt werden. Andernfalls besteht die Gefahr, dass ein Angreifer diese Nachrichten abhört und so unter Umständen wertvolle Informationen über Netzwerktopologie und Präfixlebensdauer erhält. Von der Präfixlebensdauer können z.B. Schlussfolgerungen über die Prefix-Update-Policie geschlossen werden. Dies alles können Informationen sein, die ein Angreifer auf die eine oder andere Art für seine Zwecke verwenden kann.

Ohne einen entsprechenden Schutz wäre es auch möglich gefälschte Mobile Prefix Advertisement einzuschleusen . Ein Angreifer könnte z.B. in einer solchen Nachricht die Lebensdauer des Präfix verringern, so dass der mobile Knoten viel öfter als benötig entsprechende Anfragen stellen muss. Der Angreifer kann so eine grosse Menge von Anfragen abhören und versuchen

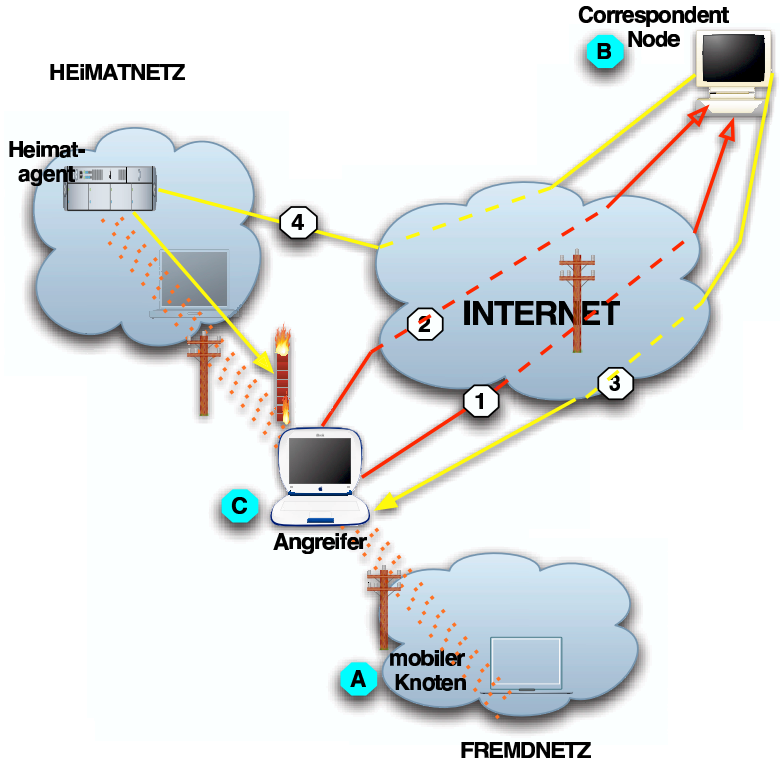


Abbildung 5: Hijacking

anhand dieser Daten auf den Wert des sogenannten Indentifizierungs-Feldes zu kommen. Der Indentifier weist (zukünftig) eintreffende Mobile-Präfix-Aufforderungsnachricht die entsprechende Mobile-Präfix Ankündigungsnachricht zu. Der Angreifer könnte diesen Wert dazu verwende um an den Heimatagenten ein gefälschtes Mobile Prefix Solicitation zu senden und so Informationen zu erhalten.

Ein Angreifer könnte auch versuchen ein Mobile-Präfix Ankündigungsnachrichtan den mobilen Knoten abzufangen. Durch das gezielte Abfangen, wird dem mobilen Knoten Informationen über die neue Präfixadresse und somit über die neue Heimatadresse vorenthalten. Der mobile Knoten steht ohne Möglichkeit da, über seine Heimatadresse angesprochen zu werden bzw. über den Heimatagenten mit anderen Kommunikationspartner Daten auszutauschen. Auf diese Art und Weise ist es einem Angreifer möglich in kurzer Zeit eine Menge von Verbindungen abzubrechen. In vielen Fällen besteht zwischen zwei IP-Adressen sogar eine grosse Anzahl von Verbindungen, die verschiedene Ports ansprechen. (Abbildung 7)

4 Angriffe auf Binding Updates

Vor allem der Austausch von Binding Updates birgt Risiken und bedarf besonderen Schutz. So kann ein bösartiger Knoten versuchen die Kommunikation, an sich oder an einen anderen (Dritten) Knoten umzuleiten.

Wir gehen im folgenden Beispiel von einer bestehenden Kommunikation zwischen den Knoten A und B aus. Der bösartige Knoten C sendet nun ein falsches Binding Update an den Heimatagenten des Knoten A. Er gibt sich also als Knoten A aus und gibt als neue Zustelladresse des vermeintlichen Knotens A, seine eigene Zustelladresse an. Auf diese Weise ist er in der Lage den gesamten Datenverkehr vom Knoten A abzufangen. Durch Verwendung gefälschter

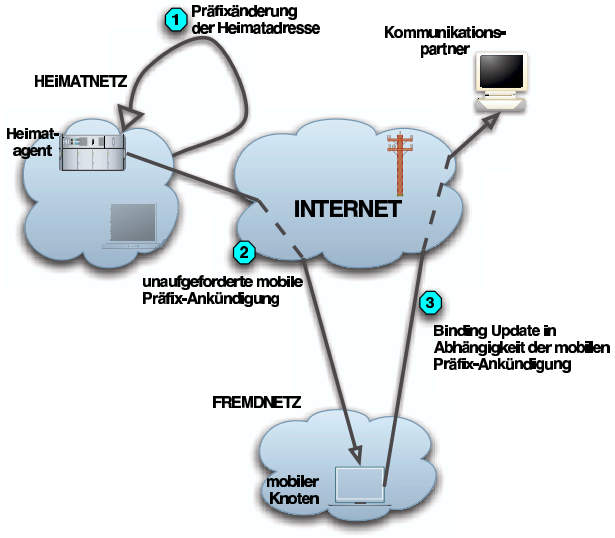


Abbildung 6: Mobile Prefix Discovery

Binding Updates wäre ein Angreifer also in der Lage den Datentransfer zwischen zwei Knoten an eine zufällige oder nicht existierende Adresse umzuleiten. Er kann also die Kommunikation zwischen den beiden Knoten empfindlich stören oder sogar gänzlich unterbinden. Schickt er ein derart gefälschtes Binding Update an den Heimatagenten, so würden die Daten, die an die Heimatadresse des mobilen Knoten gesendet werden, nun nicht mehr beim mobilen Knoten ankommen.

Die Umleitung von Daten könnte auch dazu verwendet werden, einen Dritten derart mit Daten zuzuschütten, dass dessen CPU extrem ausgelastet wird und dieser somit seiner eigentlichen Tätigkeit nicht mehr nachkommen kann. Im schlimmsten Fall kann es auch zu eine totalen Ausfall kommen. Ein solcher Angriff wird als Denial-of-Service bezeichnet. Die Grundidee eines solchen Angriffs besteht also darin, sein Opfer durch das versenden von Unmengen von Daten von der eigentlichen Arbeit abzuhalten und die Kommunikation zu anderen zu verhindern. Ein effektiver Denial-of-Service-Angriff auf einen mobilen Knoten würde also bewirken, dass er keine Kommunikation zu seinem Heimatagenten aufnehmen kann. In der Zwischenzeit schickt der Angreifer ein gefälschtes Binding Update an die Heimatagenten des mobilen Knoten und gibt seine eigene IP-Adresse als neue Zustelladresse des mobilen Knotens an. Er empfängt nun alle Pakete, die für den mobilen Knoten gedacht waren. Die gleiche Prozedur kann er verwenden um den Heimatagenten aussergefecht zu setzen und sich beim mobilen Knoten als Heimatagenten auszugeben. In diesem Fall würde er alle Pakete erhalten die der mobile Knoten versendet.

Wenn ein Angreifer sich geschickt anstellt, kann er es sogar schaffen diese beiden Angriffsszenarien zu kombinieren und sich zwischen die Kommunikation von Heimatagenten und mobilen Knoten einklinken. Nun erfolgt jeglicher Datenaustausch zwischen den beiden nur noch über den angreifenden Knoten (Man-in-the-Middle). (Abbildung 8)

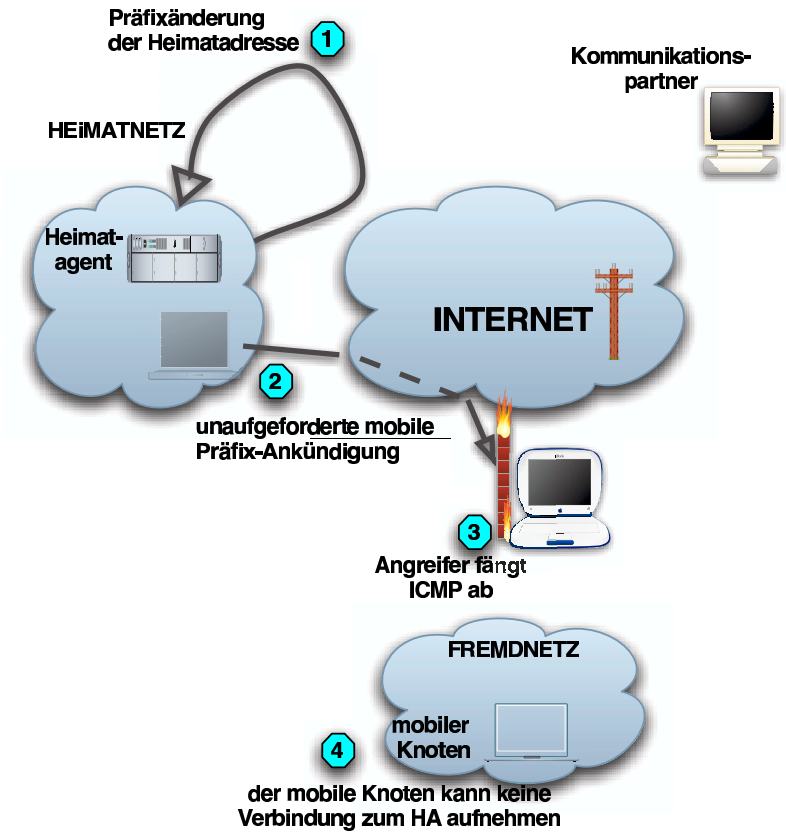


Abbildung 7: ICMP-Angriff

Eine weitere Angriffsform wäre der so genannte Wiederspiegelungs- oder Replay-Angriff. Ein bereits gesendetes Binding-Update wird dabei abgefangen und zu einem späteren Zeitpunkt erneut verschickt. Ein mobiler Knoten befindet sich in einem Fremdnetz und schickt von dort aus ein Binding Update an den Heimatagenten. Diese Kommunikation wird vom Angreifer abgehört und gespeichert. Zu einem späteren Zeitpunkt bewegt sich der mobile Knoten in ein anderes (neues) Fremdnetz und setzt den Heimatagenten wiederum davon in Kenntnis. Nun kann der Angreifer das gespeicherte Binding Update an den Heimatagenten schicken, der dann absofort alle Daten an das alte Fremdnetz schickt.

Die Lösung zum Entschärfen solcher Gefahren besteht in der Verwendung von IPsec, um die Kommunikation zwischen mobilen Knoten und Heimatagenten abzusichern.

5 Sicherung der Verbindung zwischen Heimatagenten und mobilen Knoten durch IPSec

Um die Kommunikation zwischen mobilen Knoten und Heimatagenten zu schützen wird, wie bereits erwähnt, IPsec verwendet. IPSec besteht aus den beiden Protokollen Encapsulated Security Payload (ESP) und Authentication Header (AH), die den eigentlichen Sicherheitsdienst erbringen.

Aufgabe des AH-Protokolls ist es für die Authentizität und Integrität der übertragenden Pakete zu sorgen. Es wird also überprüft, ob die Daten tatsächlich vom Absender stammen und der Inhalt nicht von einem Dritten verändert wurde. Die Hauptaufgabe des ESP-Protokolls

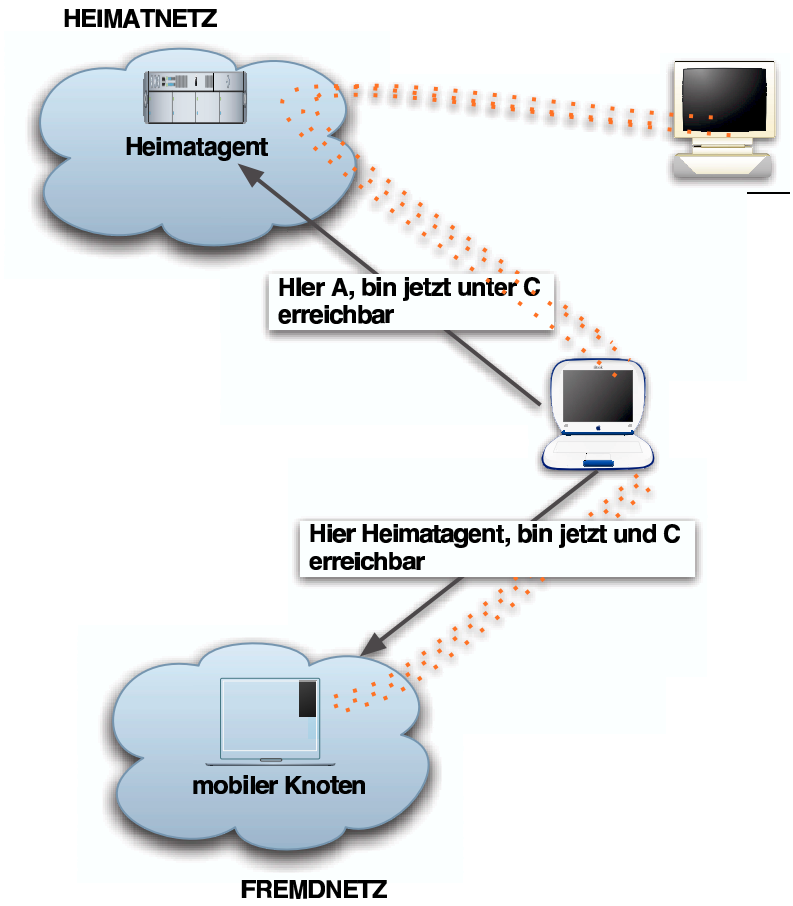


Abbildung 8: Schema eines Man-in-the-Middle-Angriffes

ist es, die Vertraulichkeit der zu übermittelnden Daten zu gewährleisten. Die Daten werden also verschlüsselt übertragen. Diese Verschlüsselung erfolgt mit einem im Vorfeld ausgehandelten symmetrischen Schlüssel. Ich werde im weiteren Verlauf lediglich auf ESP eingehen, da es üblich ist die Kommunikation zwischen Heimatagenten und mobilen Knoten mit diesem Protokoll abzusichern

Abhängig vom Modus, in dem das EPS-Protokoll betrieben wird, bezieht sich die Verschlüsselung entweder nur auf die Nutzdaten oder auf das gesamte IP-Paket.

Man unterscheidet zwischen Transport- und Tunnelmodus.

Transportmodus

Im Transportmodus werden nur die Nutzdaten der zu versendeten IP-Pakete verschlüsselt. Der ursprüngliche IP-Kopf wird dabei beibehalten und es wird zusätzlich ein ESP-Kopf in das Datenpaket eingefügt. Es müssen also lediglich einige wenige Bytes hinzugefügt werden.

Da in diesem Modus der IP-Kopf unverändert bleibt, kann ein Angreifer sehen, zwischen welchen Endknoten eine Kommunikation erfolgt. Er kann also den Datenverkehr analysieren.

Sowohl beim Binding also auch bei der Mobile Präfix-Aufdeckung erfolgt die Verschüsselung im Transportmodus. Grund hierfür ist, dass die Informationen lediglich zwischen zwei Knoten ausgetauscht werden. Hier erfolgt also lediglich die Verschlüsselung der sicherheitsrelevanten Daten. (Abbildung 9 und Abbildung 10)

Tunnelmodus

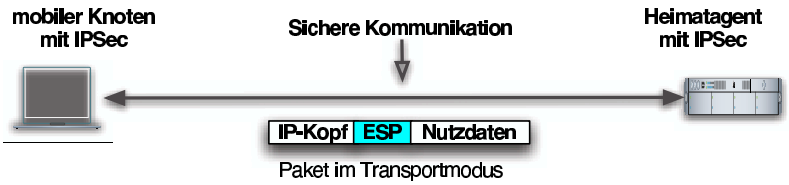


Abbildung 9: Transportmodus-1

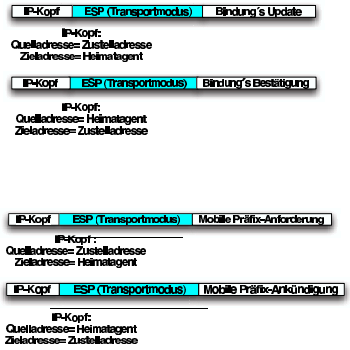


Abbildung 10: Transportmodus-2

Im Tunnelmodus erfolgt eine Verschlüsselung des gesamten IP-Paketes. Auch der Original-IP-Kopf wird mitverschlüsselt. Nach der Verschlüsselung wird ein weiterer (äußerer) IP-Kopf an die so verschlüsselten Daten hinzugefügt. Anhand dieses neuen IP-Kopfes werden die Pakete an das Ende des Tunnels geleitet. Dort werden die Pakete entschlüsselt, so dass die Originaldateien wieder lesbar sind. Nach der Entschlüsselung können diese Daten dann an den inneren IP-Kopf weitergeleitet werden. Auf diese Weise ist es einem Dritten nicht möglich, das Endziel des IP-Paketes zu ermitteln. IP-Köpfe, die nicht für die Öffentlichkeit gedacht sind, sind somit also nicht zugänglich. (Abbildung 11 und Abbildung 12)

Daher erfolgt die Verschlüsselung bei der Return Routability Procedure im Tunnelmodus.

Die Grundidee ist es also, jedes Paket, das sicherheitsrelevante Daten enthält, von einem bestimmten Knoten, mit einem symmetrischen Schlüssel zu verschlüsseln. Diese Daten werden dann von einem dafür vorgesehenen Knoten wieder entschlüsselt, so dass die Originaldaten wieder hergestellt sind. (Abbildung 13)

Um zwischen mobilen Knoten und Heimatagenten Daten verschlüsseln und auch wieder entschlüsseln zu können, muss im Vorfeld natürlich ein entsprechender Schlüssel ausgetauscht werden. Hierzu muss zwischen beiden Kommunikationspartner eine sogenannte Sicherheitsvereinbarung (SV) ausgehandelt werden. In einer solchen SV handeln die beiden Kommu-

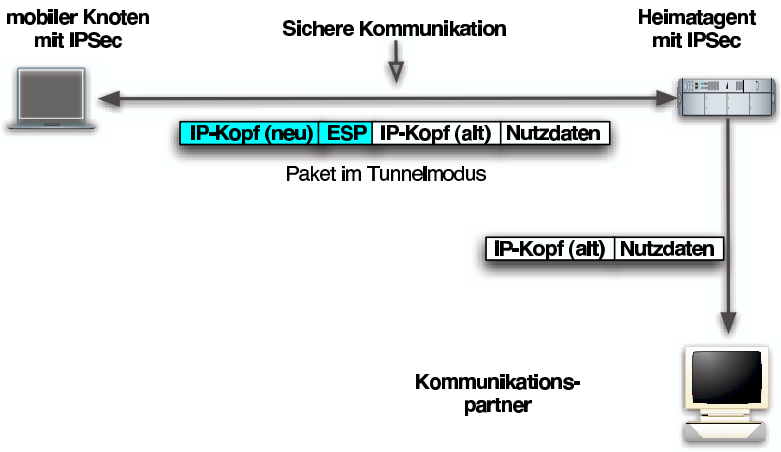


Abbildung 11: Tunnelmodus-1

nikationspartner die Verfahren und Algorithmen, aus die für die Authentifizierung und Verschlüsselung der Daten erforderlich sind. Einmal ausgehandelt definiert eine solche SV exakt wie die Kommunikation mit dem Zielknoten geschützt wird. Jede Sicherheitsvereinbarung ist einmalig und wird im wesentlichen identifiziert durch:

- Security Parameter Index (SPI)
- IP-Zieladresse
- Security Protocol Identifier

Der Security Parameter Index ordnet den empfangenen Daten, die dazugehörige SV zu.

Bei der Zieladresse handelt es sich meistens um eine Unicast-Adresse.

Der Security Protocol Identifier gibt das verwendete Protokoll an: In diesem Fall ESP

Ein weiterer Bestandteil der SV ist der zur Verschlüsselung verwendete Algorithmus, sowie der dazu verwendete Schlüssel. Die Aushandlung einer solchen SV kann manuell erfolgen oder durch ein automatisiertes Schlüsselaustauschprotokoll.

Ich werde hier kurz auf das Schlüsselaustauschprotokoll Internet Key Exchange eingehen. Dieses Protokoll ermöglicht eine automatische Einigung auf einen gemeinsamen Schlüssel und sieht hierfür zwei Phasen vor. (Abbildung 14)

In der ersten Phase wird lediglich ein vergleichsweiser schwach gesicherter, authentifizierter Kanal hergestellt. Der Initiator bietet der Gegenstelle hierzu einige Verschlüsselungsalgorithmen an, von denen sich die Gegenstelle entsprechend seiner Prioritäten eines heraussucht und quittiert. Im Anschluss vereinbaren die beiden Kommunikationspartner mit Hilfe des Diffie-Hellman-Algorithmus eine geheimen symmetrischen Schlüssel. Dieser Schlüssel dient im weiteren Verlauf zur Verschlüsselung der Daten. Nun müssen sich die Gegenstellen noch gegenseitig authentifizieren. Dies kann z.B. durch die Verwendung eines Public-Key-Verfahren erfolgen. Jede Gegenstelle erzeugt hierzu eine Zufallszahl und verschlüsselt sie mit dem öffentlichen Schlüssel des anderen und schickt sie an diesen. Der Empfänger entschlüsselt die Zufallszahl mit seinem privaten Schlüssel und sendet sie wieder an den Absender zurück. Damit ist die Authentifizierung vollzogen. Die Authentizität der Kommunikationspartner, sowie die Vertraulichkeit der Verbindung ist somit erreicht.

Über diesen Kanal wird dann die eigentliche Sicherheitsvereinbarung vereinbart: Phase 2

Abbildung 12: Tunnelmodus-2

In der zweiten Phase wird nun, über die in der ersten Phase erzeugte Sicherheitsbeziehung, die eigentliche SV ausgehandelt. Hierzu gehören Angaben über das zu verwendete Protokoll (ESP), dem Verschlüsselungsalgorithmus und auch die notwendigen Schlüssel werden erzeugt. Nun kann eine sichere Kommunikation zwischen mobilen Knoten und Heimatagenten erfolgen. Es sei noch zu erwähnen, das eine Sicherheitsvereinbarung jeweils nur in eine Kommunikationsrichtung gültig ist. Da zwischen Heimtagenten und mobilen Knoten eine Kommunikation in beide Richtungen erfolgt, werden zwei SV benötigt.

6 Alternatives Authentifizierungsprotokoll für Mobile IPv6

IPsec (ESP) bietet, wie bereits beschrieben, eine starken Schutz gegen eine Vielzahl von Angriffen. Allerdings wir IPsec dennoch nicht in allen Fällen, in dem mobile IPv6 eingesetzt wird, benötigt. Für kleine mobile Geräte, wie z.B. PDAs würde die Verwendung von IPSec einen zu hohen Batterieverbrauch und CPU-Auslastung bedeuten.

Das Verfahren ist eine Light-Mechanismus zur Authentifizierung des mobilen Knotens beim Heimatagenten bzw. beim Heimat-AAA-Server. Es beruht auf der Verwendung einer AAA-Struktur (Authentication Authorization Accounting) in Verbindung mit NAI. Der Heimat-AAA-Server benutzt den Network Access Identifier zur Identifikation des mobilen Knotens. Die Identifikation erfolgt also nicht anhand der IP-Adresse. Dies stellt im Vergleich zu IPsec ein Vorteil da, da es bei IPsec bei einer Änderung der Heimatadresse zu Problemen führen kann. Vorausgesetzt es besteht mit einer Einheit des Heimatnetzes (z.B. ein AAA-Server) eine gültige SV, kann sich der mobile Knoten bei diesem Verfahren bei jedem beliebigen Heimatagenten seines Heimatnetzes anmelden.

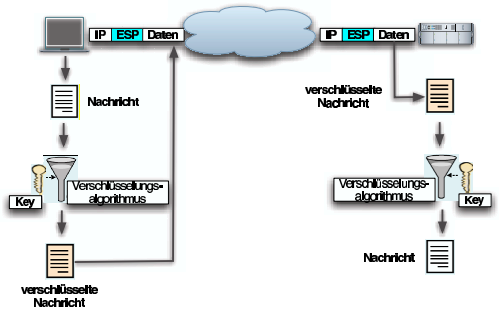


Abbildung 13: Prinzip der Verschlüsselung

Zur Absicherung des Binding Update (BU) und des Binding Acknowledgement (BA) wird eine neue Erweiterung definiert, die sogenannte Mobility Message Authentication Option, die zur Authentifizierung des mobilen Knoten dient. Hierzu wird innerhalb der Mobility Message Authentication Option ein spezielles Feld verwendet. Dieses sogenannte Authenticator-Feld enthält Informationen, die zur Authentifikation des mobilen Knotens dienen. Die Authentifizierung des BU und des BA erfolgt auf Basis einer gemeinsamen SV zwischen mobilen Knoten und Heimatagenten. Der Authenticator besteht aus den ersten 96 Bit des MAC. Die Berechnung des MAC erfolgt aus dem gemeinsamen Schlüssel, der Zustelladresse, der Heimatadresse, sowie aus allen übrigen Feldern des Mobility-Headers

Die Mobility Message Authentication Option, die den Authenticator enthält wird an den Heimatagenten geschickt. Dieser leitet die Authentifizierungs-Information an die AAA-Struktur des HA weiter, die nun entscheidet, ob die Registrierung des mobilen Knotens angenommen wird. Sobald der Heimatagent ein Binding Update mit einer solchen Option erhält, extrahiert er den Authenticator und den SPI (dieser weist einer Nachricht eine SV zu) von der Authentication Option und extrahiert desweiteren die NAI von der NAI-Option und leitet diese Daten dann zur AAA-Struktur, die dann anhand dieser Daten die Authentifizierung übernimmt.

Auch zur Absicherung von Mobile Präfix Anfrage und Mobile Präfix Ankündigung kann dieses Verfahren eingesetzt werden. Zur Authentifizierung des mobilen Knotens wird auch hier die Mobility Message Authentication Option verwendet. Erst nach der Authentifizierung erfolgt auf die Mobile Präfix Anfrage eine Antwort. Diese Antwort erfolgt mit einer Mobile Präfix Ankündigung, der an den mobilen Knoten gesendet wird. Um die Information zu schützen und so mögliche Angriffe zu verhindern, erfolgt eine Verschlüsselung dieser Informationen.

Sobald der Heimatagent die Mobile Präfix Anfrage vom mobilen Knoten erhält, erfolgt die Authentifizierung des mobilen Knotens anhand der oben beschriebenen Mobility Message Authentication Option, die in der Nachricht enthalten ist. Anschliessend versendet der Hei-

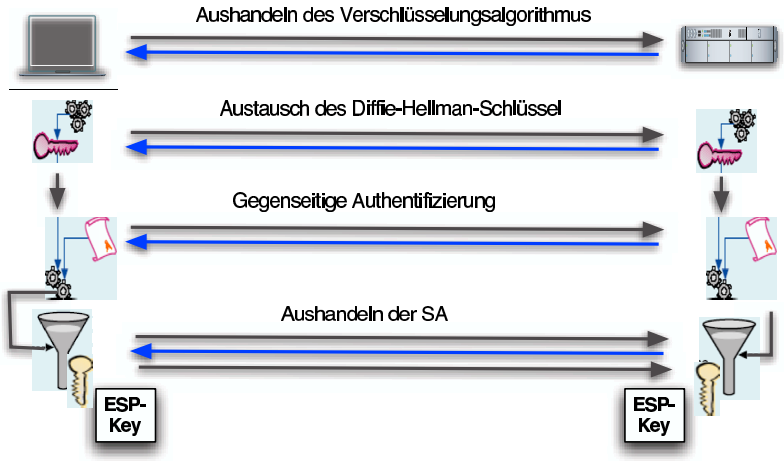


Abbildung 14: Schritte zur Vereinbarung einer Sicherheitsvereinbarung

matagent als Antwort die Mobile Präfix Ankündigung an den mobilen Knoten. Zuvor muss die Präfixinformation allerdings noch mit dem gemeinsamen Schlüssel verschlüsselt werden. Die verschlüsselte Präfix-Information wird dann im Payload-Feld eingefügt. Der mobile Knoten erhält die so verschlüsselten Daten und erkennt anhand der mitgesendeten SPI, welchen Schlüssel er zur Entschlüsselung verwenden soll. Nun stehen ihm die Informationen über die Präfixänderung des Heimatnetzes zur Verfügung.

7 Fazit

MIPv6 bietet viele nützliche und notwendige Verbesserungen, die viele der Nachteile seines Vorgängers, Mipv4, beheben. Mit diesen neuen Features treten allerdings auch viele neue Sicherheitsprobleme in Erscheinung. Ohne entsprechende Schutzmechanismen, wie z.B. IPSec, wiegen diese Gefahren die Vorteile von MIPv6 auf. Wie meine Ausarbeitung zeigt, ist ein Schutz von sicherheitsrelevante Daten zwingend notwendig. Vor allem in mobilen Bereichen, wie z.B. WLan-Umgebungen, ist das Abhören von Verbindungen für Angreifer erheblich erleichtert worden. Solche Umgebungen benötigen daher besondere Aufmerksamkeit.

Literatur

[ArDD03] J. Arkko, V. Devarapalli und F. Dupont. Using IPsec to Protect Mobile IPv6 Signaling between Mobile Nodes and Home Agents. draft-ietf-mobileip-mipv6-ha-ipsec-06.txt, Dezember 2003.

[JoPA03] D. Johnson, C. Perkins und J. Arkko. Mobility Support in IPv6. draft-ietf-mobileip-ipv6-24.txt, Dezember 2003.

[Pate04] A. Patel. Authentication Protocol for Mobile IPv6. draft-patel-mipv6-auth-protocol-01.txt, November 2004.

Abbildungsverzeichnis

1 Mobiler Knoten bewegt sich aus seinem Heimatnetzt 4
2 bidirektionales Tunneln 5
3 Route Optimization 6
4 Return Routability Procedure 7
5 Hijacking 8
6 Mobile Prefix Discovery 9
7 ICMP-Angriff 10
8 Schema eines Man-in-the-Middle-Angriffes 11
9 Transportmodus-1 12
10 Transportmodus-2 12
11 Tunnelmodus-1 13
12 Tunnelmodus-2 14
13 Prinzip der Verschlüsselung 15
14 Schritte zur Vereinbarung einer Sicherheitsvereinbarung 16

Konfiguration einer neuen Adresse bei einem Mobile-IPv6-Handover

Tobias Reisch

Kurzfassung

Diese Seminararbeit beschreibt die Konfiguration einer neuen Adresse bei einem Mobile-IPv6-Handover. Dabei wird erläutert, wie ein mobiler Knoten feststellt, dass er eine neue Adresse braucht, welche Arten der Adresskonfiguration möglich sind und wie dies bewerkstelligt wird.
In Kapitel 1 werden zum besseren Verständnis erst einmal die wichtigsten Bergiffe erklärt und eine kleine Motivation zu diesem Thema gegeben. Kapitel 2 befasst sich mit den verschiedenen Konfigurationsarten und den dazugehörigen Algorithmen und Protokollen. In Kapitel 3 geht es dann um die eigentliche Konfiguration einer neuen Care-of-Adresse und den dazu benötigten Mechanismen. Zuletzt werden in Kapitel 4 Verbesserungen und Alternativen aufgezeigt, welche die Konfiguration beschleunigen sollen. Kapitel 5 gibt dann noch einmal eine Zusammenfassung des kompletten Themas wieder.

1 Einleitung

1.1 Begriffserklärungen

- **Knoten:** Ein Gerät, welches IP implementiert.
- **Mobiler Knoten:** Ein Knoten, der seinen Anschlusspunkt von einem Link zu einem anderen wechseln kann, während er über seine Heimatadresse immer noch erreichbar ist.
- **Heimatadresse:** Eine Adresse, die einem mobilen Knoten zugewiesen wird und als seine permanente Adresse benutzt wird.
- **Heimat-Subnetzpräfix:** Das IP-Subnetzpräfix, welches der Heimatadressse des mobilen Knotens entspricht.
- **Link:** Eine Kommunikationseinrichtung oder Medium, über welche Knoten auf Schicht 2 kommunizieren können.
- **Heimatlink:** Der Link, auf welchem das Heimat-Subnetzpräfix eines mobilen Knotens definiert ist.
- **Heimatagent:** Ein Router auf dem Heimatlink eines mobilen Knotens, bei welchem der mobile Knoten seine derzeitige Care-of-Adresse registriert hat.
- **Fremder Link:** Ein anderer Link als der Heimatlink des mobilen Knotens.
- **Fremdes Subnetzpräfix:** Irgendein anderes IP-Subnetzpräfix als das Heimat-Subnetzpräfix des mobilen Knotens.

- **Care-of-Adresse:** Eine Adresse, die mit dem mobilen Knoten verbunden ist, während er einen fremden Link besucht; das Subnetzpräfix dieser IP-Adresse ist ein Präfix eines fremden Subnetzes. Diejenige, die im Heimatagenten des mobilen Knotens für eine gegebene Heimatadresse eingetragen ist, nennt man primäre Care-of-Adresse.
- **Handover:** Ein Vorgang, bei dem ein mobiler Knoten von einer Schicht-2-Verbindung in eine andere wechselt (L2 Handover).

1.2 Motivation und Überblick

Wenn man mit seinem Notebook heutzutage unterwegs ist, kann es logischerweise dazu kommen, dass man in ein anderes Subnetz gelangt. Damit man nach solch einem Netzwechsel nun weiter mit anderen Rechnern über das Internet kommunizieren kann, ist es notwendig, dass man eine neue IP-Adresse konfiguriert. Das ist auf verschiedene Arten möglich, zum Beispiel durch eine „Zustandslose Autokonfiguration“ (mit Duplicate Address Detection), oder durch eine „Zustandsbehaftete Autokonfiguration“ (mit DHCPv6). Ziel ist es, durch Konfiguration eine neue Care-of-Adresse zu erhalten, deren Präfix im neuen Subnetz gültig ist.
Ein wichtiger Aspekt hierbei ist die sogenannte „Bewegungserkennung“, mit der man feststellen kann, ob ein mobiler Knoten in ein anderes Subnetz gelangt ist. Dies kann über „Router Discovery“ gemacht werden, mit der man benachbarte Router lokalisieren kann. Außerdem dient sie zum Erlernen von Präfixen und Konfigurationsparametern bezüglich der Adress-Autokonfiguration.
In den folgenden Abschnitten wird das Vorgehen zur Konfiguration einer neuen Care-of-Adresse erläutert. Da diese Verfahren (zeitlich) nicht optimal sind, werden zum Schluss noch Verbesserungsmöglichkeiten und Alternativen vorgestellt.

2 Konfigurationsarten

In diesem Kapitel werden nun verschiedene Konfigurationsarten vorgestellt, und es wird darauf eingegangen, wie lange die Konfiguration aufgrund der Duplicate Address Detection (DAD) dauert. In Kapitel 4 werden dann Ansätze vorgestellt, die eine zeitliche Optimierung erbringen sollen.

Der Autokonfigurationsprozess beinhaltet das Kreieren einer Link-Local-Adresse und das Verifizieren ihrer Eindeutigkeit auf einem Link. Außerdem kann mit Hilfe des Autokonfigurationsprozesses ermittelt werden, welche Information autokonfiguriert werden soll (Adressen, andere Information oder beides), und in Bezug auf Adressen, ob sie durch den zustandslosen oder zustandsbehafteten Mechanismus - oder durch beide - beschafft werden sollen. Zustandslose und zustandsbehaftete Adresskonfiguration ergänzen sich gegenseitig: Zum Beispiel kann ein Host zustandslose Adresskonfiguration dazu benutzen, um seine eigenen Adressen zu konfigurieren, aber zustandsbehaftete Adresskonfiguration, um andere Information zu beschaffen. Auf diese zwei Arten der Autokonfiguration wird in den nächsten Abschnitten genauer eingegangen.

2.1 Zustandslose Autokonfiguration (Stateless Autoconfiguration)

Nach [ThNa98] ermöglicht dieses Verfahren es Knoten, sich in einem Netz vollkommen selbstständig zu konfigurieren. Die Knoten generieren Adressen ohne Notwendigkeit eines DHCP-Servers. Die Adressen werden durch das Kombinieren eines Netzwerkpräfixes und eines Interface Identifiers gebildet. Auf Interfaces, welche eingebettete IEEE Identifier beinhalten, wird

der Interface Identifier typischerweise von ihm abgeleitet. Auf anderen Interfacetypen wird der Interface Identifier durch andere Methoden generiert, zum Beispiel durch Generierung einer zufälligen Nummer.
Zustandslose Autokonfiguration benötigt keine manuelle Konfiguration der Hosts, minimale Konfiguration (wenn überhaupt welche) der Router und keine zusätzlichen Server. Der zustandslose Mechanismus erlaubt es einem Host, seine eigenen Adressen zu generieren, indem er eine Kombination aus der lokal verfügbaren Information und der vom Router bekannt gegebenen Information benutzt. Router geben Präfixe bekannt, welche Subnetze, die mit einem Link verknüpft sind, identifizieren, während Hosts einen Interface Identifier generieren, der ein Interface eines Subnetzes eindeutig identifizeirt. Eine Adresse wird durch kombinieren dieser zwei Vorgehensweisen gebildet. In Abwesenheit von Routern kann ein Host nur Link-Local-Adressen generieren. Link-Local-Adressen reichen aber aus, um Kommunikation zwischen Knoten auf dem selben Link zu erlauben. Wie sich Router bekannt machen (durch Router Advertisements/Solicitations) wird näher in Kapitel 3.1.1 („Router Discovery“) betrachtet.
Der Nachteil der zustandslosen Autokonfiguration ist, dass der Administrator die vergebenen IP-Adressen nicht beeinflussen kann. Der zustandslose Ansatz wird benutzt, wenn ein Betrieb sich nicht besonders mit den exakten Adressen, welche Hosts benutzen, befasst, solange sie eindeutig und routfähig sind.

2.1.1 Erkennung einer doppelten Adresse (DAD)

Bevor eine Link-Local-Adresse einem Interface zugewiesen und benutzt werden kann, muss ein Knoten prüfen, ob diese provisorische Adresse[1] nicht schon von einem anderen Knoten auf dem verbundenen Link benutzt wird.
Nach [ThNa98] muss Duplicate Address Detection (DAD) für jede neue Care-of-Adresse ausgeführt werden, wenn sich der mobile Knoten bewegt, und außerdem für die Link-Local-Adresse des mobilen Knotens auf jedem neuen Link. Im Prinzip also bei jedem Subnetzwechsel.
Es sollte speziell geprüft werden, ob es erforderlich ist, DAD jedes Mal für die Link-Local-Adresse erneut anzuwenden, wenn sich der mobile Knoten auf neue Links bewegt.
Duplicate Address Detection muss auf allen Unicast-Adressen durchgeführt werden, egal ob sie durch zustandslose, zustandsbehaftete, oder manuelle Konfiguration bezogen wurden, außer die Adressen haben den selben Interface Identifier.
Die Prozedur zur Erkennung doppelter Adressen benutzt Neighbor Solicitation und Advertising Nachrichten. Wenn eine doppelte Adresse während der Prozedur erkannt wird, kann die Adresse nicht dem Interface zugewiesen werden. Diese Methode ist nicht hundertprozentig zuverlässig und es ist durchaus möglich, dass noch immer Duplikate existieren. Eine Adresse, auf der die DAD Prozedur angewendet wird, bezeichnet man als provisorisch, bis die Prozedur erfolgreich abgeschlossen wurde. Duplicate Address Detection wird in erster Linie dazu verwendet, eine Adresse einem Interface zuzuweisen, um dafür vorzusorgen, dass nicht mehrere Knoten simultan dieselbe Adresse gleichzeitig benutzen. Wenn eine Adresse als eindeutig ermittelt wurde, kann sie dem Interface zugewiesen werden.
Es gibt aber auch Bedingungen, unter denen man voraussagen kann, dass die provisorische Adresse ein Duplikat ist. Eine hierfür ist, wenn eine Neighbor Solicitation für eine provisorische Adresse eher empfangen wird, als dass man eine gesendet hat. Eine andere ist die, dass die aktuelle Anzahl der empfangenen Neighbor Solicitations die Anzahl der zu erwartenden überschreitet. In diesen beiden Fällen ist die provisorische Adresse nicht eindeutig. Eine provisorische Adresse, die als Duplikat identifiziert wurde (wie eben beschrieben), darf einem

[1] provisorisch heißt, ihre Eindeutigkeit auf einem Link wird überprüft, bevor sie einem Interface zugewiesen werden

Interface nicht zugewiesen werden und der Knoten sollte einen System Management Fehler protokollieren.
Das DAD-Zeitintervall hat eine erwartete Dauer von mindestens 1000 ms, währenddessen die ganze existierende Kommunikation mit dem mobilen Knoten pausiert. Erst nachdem das DAD-Zeitintervall beendet wurde, kann der mobile Knoten mit dem Heimat-Registrierungsprozess beginnen. Da die Zeit, die ein mobiler Knoten braucht, um wieder im Subnetz kommunizieren zu können, durch die DAD-Intervallzeit signifikant beeinflusst werden kann, werden alternative Mechanismen gebraucht, um dieses Zeitintervall zu minimieren. Auf diese Mechanismen wird in Kapitel 4 eingegangen.

2.2 Zustandsbehaftete Autokonfiguration (Stateful Autoconfiguration)

Im zustandsbehafteten Autokonfigurationsmodell erhalten Rechner IP-Adressen und/oder Konfigurationsinformationen und Parameter von einem Server. Server erhalten eine Datenbank, die im Auge behält, welche Adressen welchen Hosts zugewiesen wurden. Der zustandsbehaftete Ansatz wird benutzt, wenn ein Betrieb eine schärfere Kontrolle über exakte Adresszuweisungen benötigt. Zustandslose und zustandsbehaftete Autokonfiguration müssen nacheinander verwendet werden [DCBV+03].

2.2.1 DHCPv6

DHCPv6 ist ein Client/Server-Protokoll, welches verwaltende Konfiguration von Geräten bereitstellt. Es befähigt den DHCP-Server, Konfigurationsparameter (wie z.B. IPv6 Netzwerkadressen) an IPv6-Knoten zu übergeben. Es besitzt die Fähigkeit der automatischen Zuordnung von wiederverwendbaren Netzwerkadressen und stellt außerdem zusätzliche Konfigurationsflexibilität bereit. Es ist sozusagen das zustandsbehaftete Pendant zur „IPv6 zustandslosen Autokonfiguration“ [DCBV+03].
Der DHCP-Nachrichtenaustausch von Clients und Servern erfolgt per UDP. Der Client benutzt eine durch zustandslose Autokonfiguration und DAD erhaltene Link-Local-Adresse oder eine durch andere Mechanismen ermittelte Adresse, um DHCP-Nachrichten zu übertragen bzw. zu empfangen.
DHCP-Server empfangen Nachrichten von Clients, indem sie eine reservierte, linkbegrenzte Multicast-Adresse benutzen. Ein DHCP-Client überträgt die meisten Nachrichten zu dieser reservierten Multicast-Adresse, so dass der Client nicht mit der Adresse oder den Adressen der DHCP-Server konfiguriert werden muss. Der DHCP-Server generiert nun aus der Link-Local-Adresse eine globale Adresse.
In Bezug auf die Erkennung doppelter Adressen sollte der Client auf jeder Adresse in jeglichen Identify-Associations[2], die er in der Antwortnachricht empfängt, DAD anwenden, bevor er sie zum Verkehr benutzt. Wenn irgend eine Adresse gefunden wird, die in Gebrauch eines anderen Knotens auf diesem Link ist, sendet der Client eine Ablehnungsnachricht an den Server, um ihn darüber zu informieren, dass diese Adresse verdächtig ist. Der Server sollte die vom Client abgelehnten Adresssen kennzeichnen, damit diese Adressen nicht anderen Clients zugewiesen werden.
DHCP ist eine Möglichkeit, zustandsbehaftete Autokonfiguration zu verrichten. Kompatibilität mit der zustandslosen Adressautokonfiguration ist eine Designanforderung von DHCP. Es kann die dynamischen Updates zu DNS benutzen, um Adressen und Namensräume zu integrieren, was nicht nur die Autokonfiguration, sondern auch die Autoregistrierung in IPv6 unterstützt.

[2] ein Konstrukt, durch welches ein Server und ein Client eine Menge von verwandten IPv6-Adressen identifizieren, verwalten und gruppieren können

3 Konfiguration einer neuen Care-of-Adresse

Nachdem jetzt die Konfigurationsarten besser bekannt sein sollten, geht es nun darum, wie ein mobiler Knoten feststellt, dass ein L2-Handover stattgefunden hat bzw. dass er dadurch eine neue Adresse braucht. Außerdem soll betrachtet werden, was nach dem Erkennen eines Handovers passiert und wie letztendlich eine neue Care-of-Adresse gebildet wird.

3.1 Bewegungserkennung (Movement Detection)

Das primäre Ziel der Bewegungserkennung ist das Erkennen von Subnetzwechsel. Sie benutzt dazu die Nachbarauffindung (Neighbor Discovery), welche die Routerauffindung (Router Discovery) und die Nachbarunerreichbarkeitserkennung (Neighbor Unreachability Detection, NUD) beinhaltet. Nach [JoPA04] wird Nachbarunerreichbarkeitserkennung von der Bewegungerkennung insofern benutzt, dass sie registriert, wenn der Default Router nicht länger bidirektional erreichbar ist. Daraufhin muss der mobile Knoten einen neuen Default Router (normalerweise auf einem neuen Link) aufsuchen. Wenn nämlich Router Advertisements seines aktuellen Routers den mobilen Knoten nicht nach einer gewissen Zeitspanne erreichen, kann er davon ausgehen, dass mindestens eine Advertisement dieses Routers nicht angekommen ist. Der mobile Knoten kann seine eigene Strategie implementieren, um zu entscheiden, wie viele verlorene Advertisements einen Handoverhinweis ausmachen. Wenn er dann die Situation als Handover erkannt hat, folgt die Routerauffindung [JoPA04].

3.1.1 Routerauffindung (Router Discovery)

Gemäß [NaNS98] wird die Nachbarauffindung (Neighbor Discovery) von IPv6-Knoten verwendet, um gegenseitig ihre Anwesenheit ausfindig zu machen, um gegenseitig ihre Link-Layer-Adressen zu bestimmen (mit Neighbor Solicitations/Advertisements) und um Router zu finden.
Routerauffindung wird angewendet, um benachbarte Router (die sich auf einem festem Link befinden) zu lokalisieren und um Präfixe und Konfigurationsparameter in Bezug auf die Adresskonfiguration in Erfahrung zu bringen.
Ein Router muss jede empfangene Router-Solicitation-Nachricht (zu deutsch: Router-Aufforderungs-Nachricht) verwerfen, die eine der verschiedenen Validitätsprüfungen nicht besteht. Zu diesen Prüfungen gehört zum Beispiel, dass die Prüfsumme gültig ist oder dass alle miteinbezogenen Optionen eine Länge haben, die größer als Null ist. Das gleiche gilt für einen Knoten, wenn er Router-Advertisement-Nachrichten (zu deutsch: Router-Bekanntmachungs-Nachrichten) empfängt. Auch diese werden verworfen, wenn sie verschiedenen Validitätsprüfungen nicht genügen.
Ein Router darf außerdem Router Advertisements nicht von einem Interface aus verschicken, wenn dieses kein Advertising Interface[3] ist. Ein Router muss sich der „All-Router-Multicast-Address“ auf einem Advertising Interface anschließen. Router antworten auf Router Solicitations, die zu dieser Adresse gesendet wurden und überprüfen die Beschaffenheit von Router Advertisements, die von benachbarten Routern gesendet wurden. Neben dem Senden von periodischen, unaufgeforderten Advertisements (von seinem Advertising Interface aus), um zu gewährleisten, dass sich die Hosts selbst konfigurieren können, verschickt ein Router auch Advertisements in Antwort auf gültige Solicitations, die er auf einem Advertising Interface empfängt. Solicitations sendet der Knoten, wenn er ein neues Interface konfigurieren will, ohne auf die automatischen (unaufgeforderten) Advertisements warten zu müssen.

[3] ein funktionsfähiges und aktives Multicast Interface, welches mindestens eine ihm zugewiesene Unicast-IP-Adresse hat

Die Link-Local-Adresse auf einem Router sollte, wenn überhaupt, nur selten wechseln. Knoten, die Nachbarauffindungsnachrichten empfangen, benutzen die Quelladresse, um den Sender zu identifizieren. Wenn mehrere Pakete vom selben Router verschiedene Quelladressen beinhalten, werden die Knoten annehmen, dass sie von verschiedenen Routern kommen, was zu unerwünschtem Verhalten führt. Das Benutzen der Link-Local-Adresse, um Router eindeutig auf dem Link identifizieren zu können, hat den Vorteil, dass die Adresse, unter welcher der Router bekannt ist, nicht wechseln sollte, wenn ein Netzwerk umnummeriert.
Wenn ein Router die Link-Local-Adresse für einen seiner Interfaces wechselt, sollte er Hosts über diesen Wechsel informieren. Der Router sollte ein paar Router Advertisements von der alten Link-Local-Adresse, mit dem auf Null gesetzten Router-Lebenszeit-Feld, multicasten und ebenso ein paar Router Advertisements der neuen Link-Local-Adresse. Die Routerauffindung wird nach dem Subnetzwechsel ausgeführt. Mit dem neuen Router wird dann die Präfixauffindung gemacht.

3.2 Care-of-Address Konfiguration

Kommen wir nun zum eigentlichen Kernpunkt, der Care-of-Address Konfiguration. Voraussetzungen für diese sind die Duplicate Address Detection und die Routerauffindung, um eine neue Care-of-Adresse nach einem Handover zu bilden. Ist dies geschehen, so macht der mobile Knoten die neue primäre Care-of-Adresse dem Heimatagenten bewusst. Wegen der Paketflussunterbrechung und des Signalisierungsoverhead sollte der mobile Knoten allerdings die Ausführung eines Care-of-Adresswechsel vermeiden, bis er absolut notwendig ist. Eine Care-of-Adresse ist eine mit einem mobilen Knoten verbundene IP-Adresse, welche das Subnetzpräfix eines bestimmten fremden Links hat.
Der mobile Knoten kann seine Care-of-Adresse durch konventionelle IPv6-Mechanismen erhalten, wie zum Beispiel durch zustandslose und zustandsbehaftete Autokonfiguration. Wenn zustandsbehaftete Autokonfiguration nach der zustanslosen ausgeführt werden soll, muss in der vom mobilen Knoten empfangenen Router Advertisement das M-flag (Managed flag) gesetzt sein. Solange der mobile Knoten an diesem Ort bleibt, werden Pakete, die an diese Care-of-Adresse adressiert sind, zu dem mobilen Knoten geroutet. Eine neue primäre Care-of-Adresse wird generiert, nachdem der mobile Knoten erkannt hat, dass er sich in ein anderes Subnetz bewegt hat. Dies sollte auch gemacht werden, falls die aktuelle primäre Care-of-Adresse abgelehnt wird. Er darf jederzeit eine neue primäre Care-of-Adresse bilden, darf aber ein Binding Update über eine neue Care-of-Adresse nicht öfter zu seinem Heimatagenten schicken, als die maximale Update Rate innerhalb einer Sekunde beträgt.
Ein „Binding“ für einen mobilen Knoten bezeichnet die Verbindung zwischen seiner Heimatadresse und seiner Care-of-Adresse. Währernd er von daheim weg ist, registriert der mobile Knoten seine primäre Care-of-Adresse bei einem Router auf seinem Heimatlink, mit der Bitte an diesen Router, als Heimatagent für ihn zu agieren, damit er auch weiterhin unter seiner Heimatadresse erreichbar ist. Der mobile Knoten vollzieht diese Binding-Registrierung, indem er eine „Binding-Update“-Nachricht zum Heimatagenten (dem Router) verschickt. Diese enthält die zur Zeit benutzte Care-of-Adresse. Der Heimatagent antwortet dem mobilen Knoten, indem er eine „Binding-Acknowledgement“-Nachricht zurück gibt.
Versucht nun ein ferner Rechner den mobilen Knoten zu erreichen, wird dieses Paket vom Heimatrouter erkannt und an die Care-of-Adresse getunnelt. Der mobile Knoten hat genau eine primäre Care-of-Adresse, welche bei seinem Heimatagenten registriert ist, kann aber eine zusätzliche Care-of-Adresse für irgendeine oder alle Präfixe auf dem aktuellen Link haben.
Der mobile Knoten sollte die Duplicate Address Detection nicht verzögern, weil es dadurch zu einer signifikanten Verzögerung beim Bilden der Care-of-Adresse kommen kann, wenn sich der mobile Knoten auf einen neuen Link bewegt. Um bei reibungslosen Handovers behilflich zu sein, sollte der mobile Knoten seine vorherige primäre Care-of-Adresse als (nicht-primäre)

Care-of-Adresse beibehalten und sollte immer noch an diese Adresse gerichtete Pakete akzeptieren, auch wenn er seine neue primäre Care-of-Adresse schon beim Heimatagenten registriert hat. Das macht auch Sinn, weil der mobile Knoten nur Pakete auf seiner vorherigen primären Care-of-Adresse empfangen kann, wenn er immer noch mit diesem Link verbunden ist.
Wann immer ein mobiler Knoten ermittelt, dass er durch einen gegebenen Link nicht länger erreichbar ist, sollte er alle Care-of-Adressen ungültig machen, die mit Adresspräfixen des unerreichbaren Links verbunden sind, außer sie sind in der aktuellen Adresspräfixmenge enthalten, die vom (vielleicht neuen) aktuellen Default Router angekündigt werden.

Ein mobiler Knoten erkennt durch den Bewegungserkennungsalgorithmus, dass er wieder auf seinem Heimatlink zurückgekehrt ist, wenn er bemerkt, dass sein Heimatsubnetzpräfix wieder on-link ist. Der mobile Knoten sollte dann ein Binding Update an den Heimatagenten senden, um seinen Heimatagenten darüber aufzuklären, nicht länger Pakete für ihn abzufangen und zu tunneln. Durch das Verarbeiten dieses Binding Updates wird der Heimatagent das Verteidigen der Heimatadresse des mobilen Knotens für Duplicate Address Detection beenden, und er wird nicht länger auf Neighbor Solicitations für die Heimatadresse des mobilen Knotens antworten. Wenn der mobile Knoten heimkehrt, nachdem die Bindings für alle seiner Care-of-Adressen abgelaufen sind, sollte er DAD ausführen.
Nachdem das Binding Acknowledgement für das Binding Update zu seinem Heimatagenten empfangen wurde, muss der mobile Knoten eine Neighbor Advertisement auf dem Heimatlink multicasten, um die eigene Link-Layer-Adresse zu seiner eigenen Heimatadresse bekannt zu machen. Die Zieladresse in dieser Neighbor Advertisement muss auf die Heimatadresse des mobilen Knotens gesetzt werden.
Der mobile Knoten muss solch eine Neighbor Advertisement für jede seiner Heimatadressen multicasten, welche durch die aktuellen on-link Präfixe festgelegt werden, die seine Link-Local-Adresse und seine Site-Local-Adresse enthalten.
Weil Multicasting auf einem lokalen Link (wie z.B. Ethernet) typischerweise nicht absolut verlässlich ist, kann der mobile Knoten diese Neighbor Advertisements bis zu einer maximalen Neighbor-Advertisment-Anzahl wieder und wieder übertragen, um die Zuverlässigkeit zu erhöhen. Es ist aber immer noch möglich, dass einige Knoten auf dem Heimatlink keine dieser Neighbor Advertisements empfangen haben, aber diese Knoten kann man eventuell durch das Benutzen von Nachbarunerreichbarkeitserkennung entdecken [JoPA04].

4 Verbesserungen und Alternativen

Aufgrund der Tatsache, dass die Duplicate Address Detection eine gewisse Zeit, nämlich mindestens 1000 ms in Anspruch nimmt und währenddessen die ganze existierende Kommunikation mit dem mobilen Knoten pausiert, versucht man nun mit anderen Methoden, wie zum Beispiel „Optimistic DAD“ oder „UnDAD“, die Geschwindigkeit von DAD zu verbessern. Im Folgenden werden diese beiden Methoden nun genauer betrachtet.

4.1 Optimistic DAD

Wie in [Moor04] beschrieben wird, ist Optimistic DAD schneller und vorausschauend. Es ist eine interoperable Modifikation der existierenden IPv6-Nachbarauffindung und der zustandslosen Adressautokonfiguration. Ziel ist es, Adresskonfigurationsverzögerungen im erfolgreichen Fall zu minimieren und (zeitliche) Störungen - so weit wie möglich - im fehlgeschlagenen Fall (Kollision) zu reduzieren. Optimistic DAD ist eine nützliche Optimierung, weil DAD für gut verteilte zufällige Adressen öfter von Erfolg gekrönt ist, als dass es fehlschlägt.
Störungen werden dadurch minimiert, dass man die Beteiligung der Knoten bei der Nachbarauffindung begrenzt, während ihre Adressen immer noch provisorisch sind. Die optimierte

DAD-Methode muss außerdem gewährleisten, dass die Wahrscheinlichkeit einer Adresskollision nicht ansteigt und sie muss die Auflösungsmechanismen für Adresskollisionen verbessern. Ein optimistischer Knoten geht davon aus, dass DAD erfolgreich sein wird und erlaubt höherschichtige Kommunikation auf einer Adresse, sogar während diese noch provisorisch ist. Dies macht das Verfahren um mindestens 1000 ms schneller als Standard-DAD.
Optimistic DAD ist nur dann eine nützliche Optimierung, wenn die Kollisioswahrscheinlichkeit sehr gering ist, weil eine Kollision zu einer zeitlichen Störung führen würde und der kollisionsverursachende Knoten sich rekonfigurieren müsste. Darum sollte der „Optimistische Algorithmus“ nicht für manuell zugewiesene Adressen benutzt werden, wo die Kollisionswahrscheinlichkeit aufgrund menschlicher Fehler voraussichtlich viel höher ist, als die für zufällige Adressen.
Modifikationen sind nur für optimistische Knoten erforderlich, denn diese interoperieren mit Standardknoten ohne einen signifikanten Nachteil oder Inkompatibilität. Für einen Nachbarn, zum Beispiel den Router, wäre es wünschenswert, schnell eine Kommunikation mit dem neu konfigurierten optimistischen Knoten aufzubauen. Um das zu bewerkstelligen, muss der Nachbar so bald wie möglich die Ankunft des optimistischen Knotens in Erfahrung bringen. Um zu vermeiden, auf Nachbarauffindung warten zu müssen, dürfte sich der optimistische Knoten wünschen, nicht angeforderte Neighbor Advertisements zu senden. Um dies aber effektiv bewerkstelligen zu können, müsste der Nachbar noch zwei Voraussetzungen erfüllen. Zum einen müsste er der Ankunft des optimistischen Knotens entgegensehen, und zum anderen müsste er gewillt sein, nicht angeforderte Neighbor Advertisements zu cachen.
Weil Optimistic DAD es Knoten erlaubt zu kommunizieren, selbst wenn sie provisorisch sind, kann man die erneute Übertragungszeit ohne signifikanten Nachteil bei der Default-Zeit von 1000 ms bewenden lassen. Es ist außerdem möglich, die DAD-Übertragungsanzahl zu erhöhen und auf diese Weise die Wahrscheinlichkeit einer unerkannten Adresskollision aufgrund Paketverlustes zu reduzieren.
Folgende Modifikationen bezüglich der Nachbarauffindung und der zustandslosen Adressautokonfiguration sind gegeben:

- **Nachbarauffindung:**
  - Ein Knoten darf eine Router Solicitation mit einer SLLAO (Source Link Layer Address Option) nicht von einer provisorischen Adresse senden. Router Solicitations sollten von einer nicht-provisorischen oder unspezifizierten Adresse gesendet werden, jedoch dürfen sie auch von einer provisorischen Adresse gesendet werden, solange die SLLAO nicht miteinbezogen ist.
  - Ein Knoten darf eine provisorische Adresse nicht als Quelladresse einer Neighbor Solicitation benutzen.
  - Wenn ein Knoten ein Unicast-Paket von einer provisorischen Adresse zu einem Nachbar senden muss, weiß aber dessen Link-Layer-Adresse nicht, darf er keine Nachbarauffindung ausführen und sollte stattdessen das Paket an den Router dieses Netzwerks verschicken.
  - Der optimistische Knoten darf eine unaufgeforderte Neighbor Advertisement zu allen Knoten senden, wenn er anfangs eine Adresse konfiguriert. Das Override Flag in dieser Advertisement muss gelöscht (auf 0 gesetzt) werden.
  - Der optimistische Knoten darf eine unaufgeforderte Neighbor Advertisement zu allen Knoten senden, wenn er DAD abgeschlossen hat. Das Override Flag in dieser Advertisement sollte (auf 1) gesetzt werden.
- **Zustandslose Adressautokonfiguration:**

- Wenn ein optimistischer Knoten sich dazu entscheidet, eine Adresse zu konfigurieren, hängt er ein Suffix an ein Präfix an, welches er von der Router Advertisement erhält.
- Sobald die initiale Neighbor Solicitation (und optional die unaufgeforderte Neighbor Advertisement) gesendet wird, wird die Adresse auf einem Interface konfiguriert und ist sofort zur Benutzung verfügbar.
- Ein Knoten muss von der unspezifizierten Adresse einer Neighbor Solicitation (für seine Adresse) antworten. Dies muss er mit einer Neighbor Advertisement zu der Adresse aller Knoten bewerkstelligen. Wenn die Solicitation für eine noch provisorische Adresse ist, muss die Antwort das Override Flag gelöscht haben.
- Ein Knoten muss von einer Unicast-Adresse auf eine Neighbor Solicitation (für seine Adresse) antworten, sogar während die Adresse provisorisch ist, aber die Antwort muss das Override Flag gelöscht haben.
- Eine provisorische Adresse, die als Duplikat enttarnt wird, muss sofort dekonfiguriert werden.
- Die DAD-Übertragungsanzahl sollte, wenn eine signifikante Wahrscheinlichkeit eines Paketverlustes besteht, erhöht werden.

Die folgende Abbildung zeigt den Aufbau einer Neighbor Advertisement, um zu verdeutlichen, wo die einzelnen Flags angesiedelt sind. Ab dem „Option(=2)“-Feld beginnt die SSLAO:

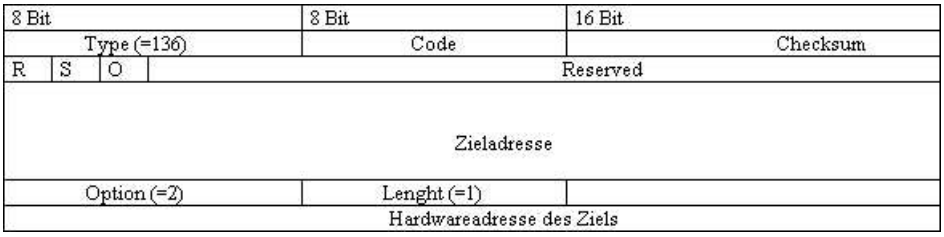


Abbildung 1: Neighbor Advertisement. Quelle: http://www.ipv6-net.org

Die Operation läuft nun folgendermaßen ab: Der optimistische Knoten wird sofort ein Neighbor Solicitation senden, um festzustellen, ob seine neue Adresse schon in Gebrauch ist, und eine Neighbor Advertisement für die Adresse. Dieses Neighbor Advertisement erlaubt es, dass die Kommunikation mit Nachbarn sofort beginnen kann. Es werden nun vier Fälle unterschieden, die alle einen optimistischen Knoten, der ein Router Advertisement empfängt beschreiben, welches ein neues Präfix besitzt, und sich entscheidet, eine neue Adresse auf diesem Präfix zu autokonfigurieren.

- **Einfacher Fall:**
  - Beim kollisionsfreien Fall ist die vom neuen Knoten konfigurierte Adresse unbenutzt und im Nachbarcache seiner Nachbarn nicht präsent. Deshalb wird es auf seine Neighbor Solicitation keine Antwort geben, und die Neighbor Advertisement mit dem Override Flag gleich 0 wird ausreichen, um Nachbarcache-Einträge in interessierten Knoten zu kreieren.
- **Kollisionsfälle:**
  - Im einfachsten Kollisionsfall ist die vom neuen Knoten konfigurierte Adresse schon in Gebrauch eines anderen Knotens und in den Nachbarcaches von den Nachbarn

präsent, welche mit diesem Knoten kommunizieren. Weil bei der optimistischen Advertisement das Override Flag auf 0 gesetzt ist, wird es existierende Nachbarcache-Einträge nicht nichtig machen. Ein Neighbor Advertisement, bei dem das Solicited Flag gleich 0 ist, bewirkt, wenn zusätzlich eine SLLAO vorhanden ist, dass der Nachbarcache-Eintrag auf STALE (abgestanden) gesetzt wird, woraufhin Nachbarunerreichbarkeitserkennung auf der Adresse ausgeführt wird.
Knoten, die kein Interesse an einer Kommunikation mit den neuen Knoten haben, sollten die Neighbor Advertisement lautlos verwerfen und werden deswegen voraussichtlich störungsfrei sein.

- **Interoperationsfälle:**
  - Wenn ein optimistischer Knoten DAD einmal beendet hat, agiert er genau wie ein Standardknoten und deshalb treten Interoperationsfälle nur dann auf, wenn der optimistische Knoten provisorisch ist.
    Falls ein optimistischer Knoten versucht, eine zurzeit provisorisch zu einem Standardknoten zugewiesene Adresse zu konfigurieren, wird der Standardknoten die Neighbor Solicitation sehen und seine Adresse dekonfigurieren. Umgekehrt, wenn der Knoten versucht, eine zurzeit provisorisch zu einem optimistischen Knoten zugewiesene Adresse zu konfigurieren, wird der optimistische Knoten seine Adresse nicht dekonfigurieren und sich stattdessen mit einer Neighbor Advertisement verteidigen, welche den Neuankömmling dazu veranlasst, sich zu rekonfigurieren. Das gibt dem optimistischen Knoten einen leichten Vorteil gegenüber Standardknoten, was aber dadurch gerechtfertigt wird, dass der optimistische Knoten schon Verbindungen aufgebaut haben könnte, während er provisorisch war.

- **Pathologische Fälle:**
  - Diese Version von Optimistic DAD hängt von Details des Routerverhaltens ab, zum Beispiel ob SLLAOs in Router Advertisements vorhanden sind und ob er gewillt ist, Verkehr für den optimistischen Knoten weiterzuleiten. Falls der Router sich nicht auf diese Art und Weise verhält, geht das Verhalten von Optimistic DAD auf das von Standard DAD zurück.

4.2 UnDAD

Die Duplicate Address Detection (DAD) verlangt von den Knoten, dass sie ein gewisses Zeitintervall warten, während die konfigurierte Adresse in dem verbundenen Link auf Duplikate überprüft wird. Diese Verzögerung beeinträchtigt die normale Kommunikation in einem mobilen Knoten während des DAD-Zeitintervalls und behindert dadurch den schnellen Handover. Die in großem Ausmaß erwartete Benutzung vom Mobile IP und das Bedürfnis einer minimalen Zeitverzögerung während des Weiterreichprozesses erfordert Ersatzmechanismen, um das DAD-Zeitintervall zu überwinden.
Bei der UnDAD-Operation wird der Nachbarcache-Eintrag des Zugangsrouters benutzt, um die Verzögerung - verbunden mit der DAD-Prozedur - zu minimieren! Zugangsrouter sind erforderlich zur Ausführung von zusätzlichen Funktionalitäten, um die ganze Liste von Knoten, die im Subnetz anwesend sind, in ihrem Nachbarcache beizubehalten. Mobile Knoten, die die Bewegung in ein neues Subnetz erkennen, erkundigen sich beim Zugangsrouter über die Eindeutigkeit der Care-of-Adresse. Der Zugangsrouter überprüft den Nachbarcache auf passende Informationen und beantwortet die Anfrage des mobilen Knotens. Diese Methode des Benutzens des Nachbarcaches des Zugangsrouters kann schnelles Weiterreichen durch Reduzierung der DAD-Verzögerungszeit unterstüzen.

Nun zum detaillierten Ablauf der Operation: Wenn ein mobiler Knoten sich in ein fremdes Netzwerk bewegt, kreiert er eine neue Care-of-Adresse mit dem durch den Zugangsrouter angezeigten Präfix. Dann muss der mobile Knoten die modifizierte DAD-Neighbor-Solicitation-Nachricht zu der Multicast-Adresse (für die DAD-Prozedur) des angeforderten Knotens senden, bevor er sie dem Interface zuweist. Die modifizierte Neighbor-Solicitation-Nachricht wird mit einem gesetzten Bit in einem dafür reservierten Feld gesendet, und der mobile Knoten wartet auf eine Neighbor-Advertisement-Nachricht, die entweder vom Zugangsrouter kommt oder von einem anderen Knoten im selben Netzwerk, welcher mit derselben Adresse konfiguriert wird. Wenn der mobile Knoten keine Neighbor-Advertisement-Nachricht empfängt, weder vom Zugangsrouter noch von einem anderen existierenden Knoten, dann muss er mindestens 1000 ms warten, bis er die Adresse zu seinem Interface konfigurieren kann.
Der Zugangsrouter ist auch zur Bereitstellung zusätzlicher Funktionalität erforderlich, bezüglich der Aufrechterhaltung der Adressen aller verbundenen Knoten im Link. Er erreicht das, indem er alle DAD-Pakete im Netzwerk abhört. Sobald der Zugangrouter eine DAD-Neighbor-Solicitation-Nachricht empfangen hat, in der das entsprechende Bit (A-Bit) gesetzt ist, schaut er in seinem Nachbarcache nach, um zu ermitteln, ob schon ein Eintrag vorhanden ist. Wenn kein Eintrag vorhanden ist, sendet er eine Neighbor-Advertisement-Nachricht mit dem gesetzten A-Bit hinaus. Er kreiert außerdem einen neuen Eintrag in seinem Nachbarcache mit der Zieladresse der Neighbor-Solicitation-Nachricht [KRHK03].

5 Zusammenfassung

Es gibt zwei verschiedene Arten der Autokonfiguration, die zum Kreieren einer neuen Care-of-Adresse gebraucht werden. Zum einen die zustandslose Adresskonfiguration, bei der sich die Knoten selbst konfigurieren und somit eine Link-Local-Adresse ermitteln, und zum anderen die zustandsbehaftete Adresskonfiguration, bei der die globale Adresse mittels eines DHCP-Servers generiert wird. Beide Konfigurationsarten müssen simultan verwendet werden.
Nach einem Handoverhinweis bei der Bewegungserkennung muss Duplicate Address Detection (DAD) und Router Discovery ausgeführt werden, um eine neue Adresse (Care-of-Adresse), bestehend aus einem linkspezifischen Subnetzpräfix und einem Interface Identifier, zu konfigurieren und somit zu erhalten. Diese Adresse muss eindeutig sein.
Da das DAD-Zeitintervall mindestens 1000 ms beträgt und währenddessen die ganze Kommunikation mit den Knoten pausiert, kann man mit der Opimistic-DAD-Methode diese 1000 ms einsparen, indem man dem Knoten zu kommunizieren erlaubt, während die Adresse noch provisorisch ist. Beim UnDAD-Ansatz wird der Nachbarcache-Eintrag eines Zugangsrouters benutzt, um die Verzögerung durch DAD zu minimieren.

Literatur

[DCBV+03] R. Droms, Ed. Cisco, J. Bound, B. Volz, T. Lemon, C. Perkins und M. Carney. Dynamic Host Configuration Protocol for IPv6 (DHCPv6). RFC 3315 (Standards Track), Juli 2003.

[JoPA04] D. Johnson, C. Perkins und J. Arkko. Mobility Support in IPv6. RFC 3775 (Standards Track), Juni 2004.

[KRHK03] JH. Kim, P. Rajendran, Y. Han und J. Kim. Using Neighbor Cache Entry for Duplicate Address Detection. Internet Draft, Oktober 2003.

[Moor04] N. Moore. Optimistic Duplicate Address Detection for IPv6. Internet Draft (In Arbeit), Juni 2004.

[NaNS98] T. Narten, E. Nordmark und W. Simpson. Neighbor Discovery for IP Version 6 (IPv6). RFC 2461 (Standards Track), Dezember 1998.

[ThNa98] S. Thomson und T. Narten. IPv6 Stateless Address Autoconfiguration. RFC 2462 (Standards Track), Dezember 1998.

Abbildungsverzeichnis

1 Neighbor Advertisement. Quelle: http://www.ipv6-net.org 27

Authentifizierungs-Protokolle für Mobile IPv6 Route Optimization

Daniel Jungbluth

Kurzfassung

Dieses Dokument beschreibt Protokolle aus der Mobilitätsunterstützung von IPv6, die dafür eingesetzt werden, Bindungsaktualisierungen zu authentifizieren. Bindungen werden bei Mobile IPv6 eingesetzt, um die Kommunikation von mobilen Knoten mit anderen Knoten im Internet zu optimieren. Man spricht von Route Optimization. Aus Sicherheitsgründen sollten nur authentifizierte Bindungen akzeptiert werden. Die Protokolle Shared Key, Binding Key Establishment Key Version 2 und Return Routability stellen dazu für mehr oder weniger spezielle Umbegungen Mittel bereit.

1 Einleitung

1.1 Übersicht

Dieses Dokument beschreibt Authentifizierungs-Protokolle für Mobile IPv6 Route Optimization. Abschnitt 2 erklärt das Shared Key-Protokoll. Eine Erweiterung dieses Protokolls ist das Binding Authentication Key Establishment Version 2-Protokoll, das in Abschnitt 3 beschrieben wird. Beide Protokolle sind in [Roe02] zu finden. In Abschnitt 4 wird nach [JoPA03] das Return Routability-Protokoll behandelt.

Die folgenden Unterabschnitte geben eine grobe Grundlage für das Verständnis der hier behandelten Thematik. Dabei versucht Abschnitt 1.2 Vorteile von Route Optimization aufzuführen, wohingegen Abschnitt 1.3 kurz erklärt, warum Authentifizierungs-Protokolle in diesem Kontext sinnvoll sind. Abschnitt 1.4 definiert grundlegende Begriffe, die in diesem Dokument verwendet werden.

1.2 Route Optimization

Grundsätzlich werden alle Pakete an einen mobilen Knoten durch dessen Heimatagenten weitergeleitet werden. Ein typischer Ablauf, bei dem ein Kommunikationspartner Daten an einen mobilen Knoten senden will, sieht so aus, dass der Kommunikationspartner die Pakete an die Heimatadresse des mobilen Knotens sendet. Befindet sich der mobile Knoten nicht in seinem Heimatnetz leitet der Heimatagent die Pakete an die Zustelladresse des mobilen Knotens weiter. Zusammen mit der Route vom mobilen Knoten zum Kommunikationspartner zurück bilden diese Paketweiterleitungen ein Dreieck. Abbildung 1 illustriert die sogenannte Dreieckspaketweiterleitung (triangular routing). Man erkennt, dass bei diesem Verfahren ein unnötiger Mehraufwand für das Netz zwischen Kommunikationspartner und Heimatagent, aber auch für das Netz zwischen Heimatagenten und Zustelladresse des mobilen Knotens entsteht.

Eine mögliche Optimierung besteht nun darin, dass der Kommunikationspartner über den aktuellen Aufenthaltsort des mobilen Knotens informiert wird. Das heißt, der Kommunikationspartner speichert bei sich temporär die Zustelladresse des mobilen Knotens. Pakete an den mobilen Knoten werden nun direkt an die Zustelladresse gesendet und nehmen nicht den Umweg über den Heimatagenten. Diese Optimierung wird mit Route Optimization bezeichnet.

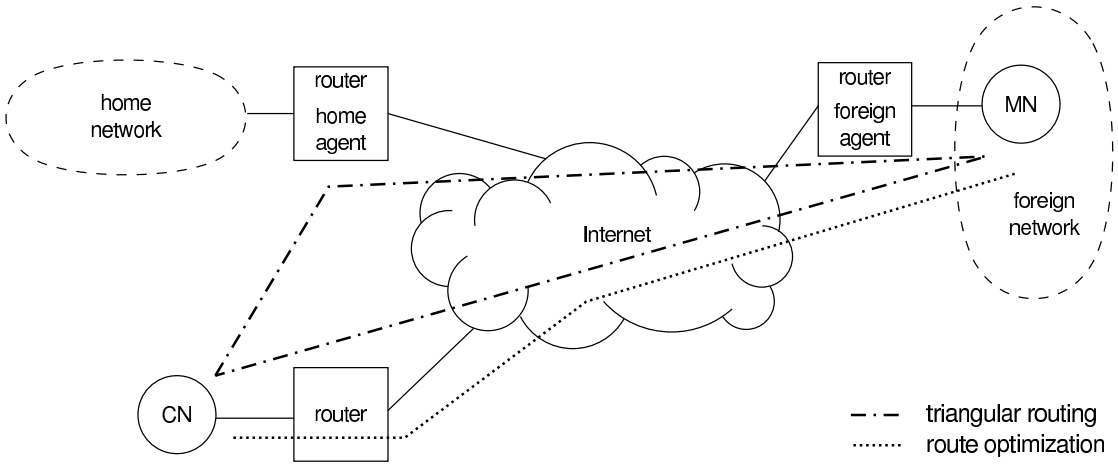


Abbildung 1: triangular routing vs. route optimization

1.3 Authentifizierung

Wenn ein Knoten Bindungsaktualisierungen ohne Authentifizierung akzeptiert, ist er verwundbar durch mehrere Angriffe, bei denen ein Angreifer gefälschte Bindungsaktualisierungen für andere Knoten sendet. Diese Angriffe beinhalten Denial of Service-Attacken, wie zum Beispiel der Fall, dass ein bösartiger, mobiler Knoten einen Dienst, der eine hohe Bandbreite benötigt (z.B. Video Stream), in Anspruch nimmt und eine Bindungsaktualisierung mit der Adresse des Opfers als gefälschten Zustelladresse durchführt. Eine andere denkbare Angriffsart über Denial of Service-Attacken hinaus ist, dass ein Angreifer über eine gefälschte Bindungsaktualisierung den Datenverkehr eines Opfers zu seiner eigenen Adresse umleitet.

Man erkennt, dass es für Knoten im Internet empfehlenswert ist, Authentifizierungs-Protokolle bei Route Optimization einzusetzen.

1.4 Terminologie

Im Folgenden werden mehrere Begriffe definiert, die für das Verständnis von Mobile IP benötigt werden. Diese werden in [JoPA03] aufgeführt. Deutsche Übersetzungen der Begriffe sind teilweise aus [Schi03] übernommen worden.

Mobiler Knoten (mobile node, MN) — Ein mobiler Knoten kann seinen Zugangspunkt zum Internet wechseln. Dabei ist er stets über seine Heimatadresse erreichbar.

Heimatadresse (home address, HoA) — Die Heimatadresse ist die IP-Adresse, die der mobile Knoten stets behält.

Heimatnetz (home network) — Das Subnetz, in das der mobile Knoten auf Grund seiner IP-Adresse gehört, ist das Heimatnetz.

Heimatagent (home agent, HA) — Der Heimatagent bietet vielfältige Dienste für den mobilen Knoten an. Er leitet Pakete, die an die Heimatadresse des mobilen Knotens adressiert sind, an die aktuelle Zustelladresse weiter, sofern sich der mobile Knoten nicht in seinem Heimatnetz aufhält.

Fremdnetz (foreign network) — Das Fremdnetz ist das aktuelle Subnetz, in dem sich der mobile Knoten aufhält und das nicht das Heimatnetz ist.

Zustelladresse (care-of address, CoA) — Die Zustelladresse ist die IP-Adresse des mobilen Knotens im aktuellen Fremdnetz.

Kommunikationspartner (correspondent node, CN) — Mit Kommunikationspartner wird der Knoten bezeichnet, mit dem der mobile Knoten kommuniziert.

Bindungsaktualisierung (binding update, BU) — Die Bindungsaktualisierung aktualisiert die Bindung, also die Zuordnung der Heimatadresse eines mobilen Knotens mit einer Zustelladresse.

2 Shared Key

2.1 Übersicht

Das Shared Key-Protokoll wird eingesetzt, um Bindungsaktualisierungen zwischen dem mobilen Knoten und dem Kommunikationspartner zu authentifizieren. Der mobile Knoten und der Kommunikationspartner teilen sich hierfür einen symmetrischen Schlüssel K_h. Es gibt verschiedene Möglichkeiten wie sich beide auf den gemeinsamen Schlüssel einigen. In Abschnitt 2.4 wird darauf eingegangen.

2.2 Protokoll

Das Shared Key-Protokoll unterteilt sich in drei Nachrichten, die zwischen dem mobilen Knoten und dem Kommunikationspartner ausgetauscht werden.

Zunächst informiert der mobile Knoten den Kommunikationspartner darüber, dass er mobil ist und überträgt in der ersten Nachricht als Parameter die Heimat- und Zustelladresse.

Der Kommunikationspartner antwortet auf diese Nachricht mit einem *binding request*. Diese Nachricht wird an die Zustelladresse des mobilen Knotens gesendet und enthält als Parameter einen *nonce*-Index und einen Message Authentication Code, der mit einem privaten Schlüssel K_{cn} des Kommunikationspartners über den Daten Zustelladresse, einer nonce (Zufallszahl) und einem angehängten Byte mit dem Wert Eins erstellt wird. Der Message Authentication Code stellt eine Challenge für den mobilen Knoten dar. Die Berechnung wird in Abbildung 2 illustriert. Der nonce-Index identifiziert die in der Berechnung der Challenge verwendeten nonce in einer internen Tabelle des Kommunikationspartner. Der Kommunikationspartner muss

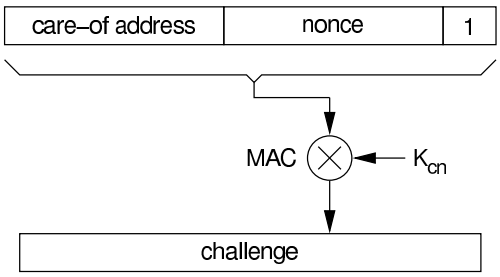


Abbildung 2: Generierung der Challenge

keinen Zustand, der mit dem mobilen Knoten assoziert ist, speichern. Aus diesem Grund werden durch Flooding von gefälschten Nachrichten eines Angreifers, der sich für einen mobilen Knoten ausgibt, beim Kommunikationspartner keine Ressourcen verbraucht. Der Kommunikationspartner wird durch dieses Protokoll daher keiner höheren Gefahr von dieser Art von Denial of Service-Attacken ausgesetzt. Er kann die Challenge über die nonce, die er durch den nonce-Index auffindet, rekonstruieren. Weil der private Schlüssel K_{cn} eingesetzt wird, kann der Kommunikationspartner zu einem späteren Zeitpunkt sicherstellen, dass er diese Nachricht erzeugt hat.

Der mobile Knoten sendet abschließend eine Bindungsaktualisierung an den Kommunikationspartner. Diese Nachricht enthält die Heimatadresse, die Zustelladresse, eine Sequenznummer und den nonce-Index. Sie wird mit dem *session key* K_{BU} authentifiziert. Der mobile Knoten erhält K_{BU}, indem er die Challenge und den gemeinsamen Schlüssel K_h hasht. Weil K_h nur dem mobilen Knoten und dem Kommunikationspartner bekannt ist, können durch einen Angreifer Spoofing-Attacken ausgeschlossen werden. Der Kommunikationspartner verifiziert diesen Message Authentication Code und legt bei Erfolg einen Eintrag für die Zustelladresse des mobilen Knotens an. Die Sequenznummer wird benötigt, um mehrere Bindungsaktualisierungen während des Lebenszyklus einer nonce zu unterscheiden. Ausserdem werden Replay-Attacken, also das Wiederholen dieser Nachricht durch einen Angreifer, unterbunden.

2.3 Änderung der Zustelladresse

Eine Variation dieses Protokolls wird eingesetzt, wenn der mobile Knoten sich bei einem Kommunikationspartner bereits authentifiziert und eine Bindungsaktualisierung durchgeführt hat. Sollte der mobile Knoten bei einem Wechsel des Fremdnetzes nicht mehr in der Lage sein, Nachrichten, die an seine alte Zustelladresse gesendet werden, zu empfangen, kann er den Kommunikationspartner vorher durch den folgend vorgestellten Ablauf benachrichtigen.

Im ersten Schritt sendet der mobile Knoten eine Bindungsaktualisierung mit seiner neuen Zustelladresse an den Kommunikationspartner. Diese wird mit dem aus dem obigen Verfahren bereits gewonnenen session key K_{BU} authentifiziert. Weil der Kommunikationspartner den mobilen Knoten bereits kennt und dessen alte Zustelladresse gespeichert hat, kann er den session key K_{BU} erneut verifizieren. Der Kommunikationspartner wird nun den Eintrag für den mobilen Knoten mit der alten Zustelladresse in seiner internen Wegewahltabelle als ungültig markieren.

Im nächsten Schritt sendet der Kommunikationspartner erneut eine Challenge an den mobilen Knoten. Diese Challenge r_c wird wie aus dem obigen Verfahren mit der neuen Zustelladesse erstellt und entsprechend an die neue Zustelladresse des mobilen Knoten gesendet. Diese Nachricht wird auch gesendet werden, wenn der Message Authentication Code aus der Bindungsaktualisierung nicht verifiziert werden kann. Somit erlaubt das Protokoll auch ei-

ne erneute Synchronisation zwischen mobilem Knoten und Kommunikationspartner, wenn Letzterer den Zustand zum mobilen Knoten bereits verloren hat.

Abschließend kann der mobile Knoten nun aus der neuen Challenge und dem gemeinsamen Schlüssel K_h einen neuen session key K_{BU} erzeugen. Dieser wird für die Bindungsaktualisierung der neuen Zustelladresse eingesetzt. Erhält der Kommunikationspartner diese Nachricht, kann er einen neuen bzw. aktualisierten Eintrag für den mobilen Knoten anlegen.

2.4 Sicherheit

Ein Knoten muss den gemeinsamen Schlüssel K_h kennen, um Bindungsaktualisierungen zu erzeugen, die vom Kommunikationspartner akzeptiert werden. Ein Problem hierbei besteht allerdings darin, wie sich die Beteiligten auf K_h einigen. Das Shared Key-Protokoll eignet sich daher für Verfahren, bei denen der Schlüssel manuell festgelegt wird. Dies ist beispielsweise möglich zwischen dem mobilen Knoten und dem Heimatagenten.

3 Binding Authentication Key Establishment Version 2

3.1 Übersicht

Das Binding Authentication Key Establishment Version 2-Protokoll, kurz Bake/2, erweitert das aus Abschnitt 2 beschriebene Shared Key-Protokoll. Es ist nun möglich den gemeinsamen Schlüssel dynamisch zu erzeugen. Allerdings kann Bake/2 nur in Umgebungen eingesetzt werden, in der die Kommunikation zumindest zwischen dem mobilen Knoten und seinem für ihn zuständigen Heimatagenten vor Lauschangriffen geschützt ist. Beispielsweise stellt IPsec ESP (Encapsulating Security Payload) Mittel hierfür zur Verfügung [KeAt98].

3.2 Protokoll

Bei Bake/2 wird - analog zum Shared Key-Protokoll - in der ersten Nachricht, die vom mobilen Knoten an den Kommunikationspartner gesendet wird, die Heimat- und Zustelladresse übermittelt.

Im zweiten Schritt generiert der Kommunikationspartner den Schlüssel K_h, der als gemeinsamer Schlüssel zwischen mobilem Knoten und Kommunikationspartner dienen wird. Hierzu wird der Message Authentication Code mit dem geheimen Schlüssel des Kommunikationspartners K_{cn} über der Heimatadresse, einer *nonce* und einem angehängten Byte mit dem Wert Null ausgeführt. K_h und der Index, der die nonce identifiziert, wird an die Heimatadresse des mobilen Knotens gesendet. Weil diese Nachricht an die Heimatadresse gesendet wird, dient diese Nachricht ausserdem als Challenge dafür, dass überprüft wird, ob der mobile Knoten überhaupt in der Lage ist, Nachrichten über die Heimatadresse zu empfangen. Ein Angreifer kann sich daher nur für einen mobilen Knoten ausgeben, wenn er Nachrichten, die über den Heimatagenten weitergeleitet werden, abfangen kann.

Gleichzeitig sendet der Kommunikationspartner an die Zustelladresse des mobilen Knotens eine Challenge. Diese setzt sich genauso wie die in Abschnitt 2.2 beschriebene zweite Nachricht des Shared Key-Protokolls zusammen.

Abschließend kann der mobile Knoten den *session key* K_{BU} aus dem Schlüssel K_h und der Challenge bilden. Mit diesem Schlüssel wird die Bindungsaktualisierung authentifiziert.

3.3 Sicherheit

Dieses Protokoll schützt den Kommunikationspartner vor Denial of Service-Attacken, bei denen dieser Knoten mit vielen gefälschten Nachrichten überflutet wird. Der Knoten muss keine Zustandsinformationen von Quellen, die nicht authentifiziert sind, speichern und die Bearbeitungszeit der Nachrichten ist gering, daher sind Attacken, die auf Ressourcenmißbrauch basieren, für den Angreifer weniger erfolgsversprechend.

Ausserdem schützt dieses Protokoll vor Angriffen, bei denen ein mobiler Knoten das Opfer ist. Der Angreifer gibt dabei vor, mobil zu sein und benutzt die Adresse des Opfers als Zustelladresse in einer gefälschten Bindungsaktualisierung. Dies führt dazu, dass der Kommunikationspartner Datenverkehr zum Opfer umleitet, den dieser aber nicht verlangt hat. Weil das Bake/2-Protokoll erst vollständig ist, wenn der mobile Knoten auf eine an seine Zustelladresse gesendete Challenge antwortet, kann diese Denial of Service-Attacke nicht greifen. Der Kommunikationspartner legt keine Bindung zu dem vermeintlichen Opfer an.

4 Return Routability

4.1 Übersicht

Der Kommunikationspartner benötigt die Gewissheit, dass der mobile Knoten tatsächlich über seine Heimat- und Zustelladresse erreichbar ist. Das Return Routability-Protokoll kann diese Gewissheit zusichern, damit der Kommunikationspartner Bindungsaktualisierungen von dem mobilen Knoten akzeptiert. Der Datenverkehr kann dann zu der von dem mobilen Knoten benutzten Zustelladresse geleitet werden.

Bei Return Routability werden Pakete an die beiden vom mobilen Knoten in Anspruch genommenen Adressen gesendet. Nur wenn der mobile Knoten diese Pakete erhält, kann er deren kombinierte Daten für eine Bindungsaktualisierung verwenden. Da der Kommunikationspartner die Pakete erzeugt, kennt er die zu kombinierenden Daten, sogenannte *keygen tokens*.

Abbildung 3 zeigt den Austausch der erforderlichen Nachrichten beim Return Routability-Protokoll. Die Nachrichten *Home Test Init* (HoTI) und *Care-of Test Init* (CoTI) werden gleichzeitig gesendet. Da die Bearbeitung dieser Initialisierungsnachrichten beim Kommunikationspartner wenig Zeit benötigt, können die Nachrichten *Home Test* (HoT) und *Care-of Test* (CoT) schnell generiert und zurückgesendet werden.

4.2 Protokoll

Im folgenden werden die vier Nachrichten, die zusammengenommen das Return Routability-Protokoll bilden, beschrieben.

4.2.1 Home Test Init

Die Nachricht Home Test Init wird über den Heimatagenten zu dem Kommunikationspartner gesendet. Der mobile Knoten zeigt damit an, dass er den *home keygen token* erhalten möchte. Ein keygen token ist eine Zahl, die vom Kommunikationspartner bereitgestellt wird und die der mobile Knoten zur Berechnung eines Schlüssels zur Authentifizierung einer Bindungsaktualisierung benötigt. In Abschnitt 4.2.3 wird beschrieben, wie der Kommunikationspartner den home keygen token generiert.

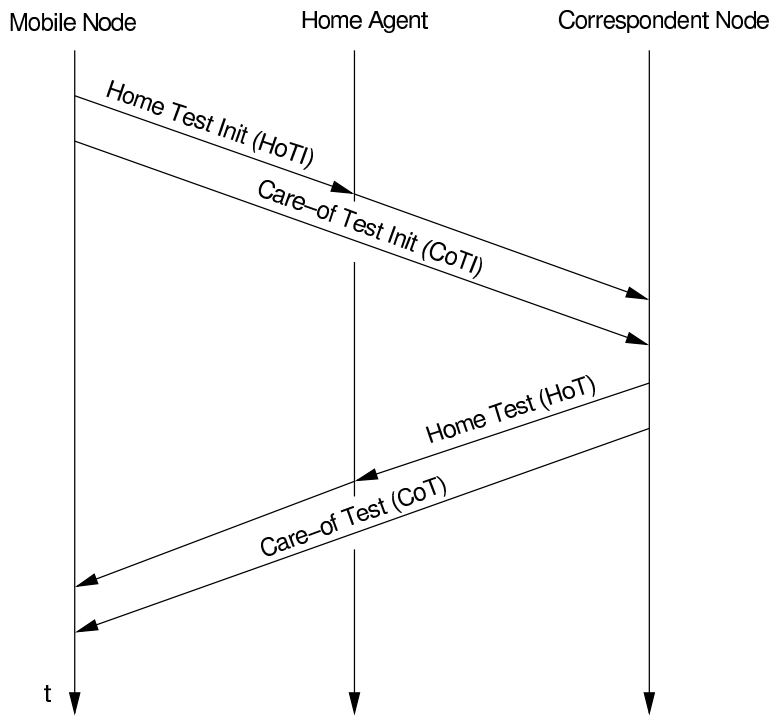


Abbildung 3: Return Routability

Als Zieladresse der Nachricht Home Test Init wird die Adresse des Kommunikationspartners eingetragen. Desweiteren wird die Heimatadresse als Quelladresse in der Nachricht vermerkt, um dem Kommunikationspartner die Heimatadresse des mobilen Knotens bekannt zu geben. Als Parameter enthält die Nachricht den *home init cookie*, dessen Wert der mobile Knoten speichern muss. Dieser cookie ist eine Zufallszahl. Der Kommunikationspartner sendet den home init cookie in einer späteren Nachricht zurück. Somit kann der mobile Knoten sicherstellen, dass seine Nachrichten von dem richtigen, beabsichtigten Kommunikationspartner bearbeitet worden sind. Ein betrügerischer Knoten kann also Antworten nicht vortäuschen, solange er den cookie nicht kennt.

4.2.2 Care-of Test Init

Ohne Umweg über den Heimatagenten wird die Nachricht Care-of Init Test direkt an den Kommunikationspartner gesendet, um den *care-of keygen token* anzufordern. Dieser fließt wie der home keygen token in die Berechnung des Authentifizierungsschlüssels mit ein. Die Berechnung des care-of keygen token wird in Abschnitt 4.2.4 beschrieben.

Die Quelladresse der Nachricht wird auf die Zustelladresse des mobilen Knotens gesetzt. Analog zum Home Test Init wird als Zieladresse die Adresse des Kommunikationspartners eingetragen. Ebenfalls analog zum Home Test Init enthält die Nachricht Care-of Test Init als Paramter den *care-of init cookie*. Dieser wird vom Kommunikationspartner in einer späteren Nachricht zurückgesendet und stellt damit sicher, dass diese Antwort nicht von Knoten vorgetäuscht werden kann, die nicht auf der Route zwischen mobilem Knoten und Kommunikationspartner liegen.

4.2.3 Home Test

Die Nachricht Home Test wird vom Kommunikationspartner als Antwort auf die Nachricht Home Init Test gesendet. Da im Home Init Test als Quelladresse die Heimatadresse des mobilen Knoten eingetragen ist, wird die Nachricht Home Test durch den Heimatagenten zum mobilen Knoten weitergeleitet. Als Parameter werden der home init cookie, der home keygen token und der *home nonce index* mitgesendet.

Das in der Nachricht Home Init Test gesendete home init cookie wird in dieser Nachricht zurückgesendet. Dadurch wird sichergestellt, dass der Home Test nur von einem Knoten stammt, der auf der Route zwischen dem Heimatagenten des mobilen Knotens und dem Kommunikationspartner liegt.

Abbildung 4 (links) illustriert die Generierung des home keygen tokens. Aus der durch die Nachricht Home Init Test dem Kommunikationspartner bekannt gegebenen Heimatadresse, einer *nonce* und einem angehängten Byte mit dem Wert Null wird der Message Authentication Code mit der Hashfunktion SHA1 (Secure Hash Algorithm Version 1, [EaJo01]) berechnet. Die ersten 64 Bit dieses Wertes bilden den home keygen token. Der Schlüssel K_{cn} ist nur dem Kommunikationspartner bekannt und dient dazu, später sicherzustellen, dass der Kommunikationspartner den home keygen token erzeugt hat. Es ist nicht erforderlich eine Liste aller herausgegebenen home keygen tokens zu halten.

Der home nonce index identifiziert die im home keygen token verwendete nonce. Dadurch kann der Kommunikationspartner bei einer späteren Bindungsaktualisierung durch den mobilen Knoten den Wert dieser nonce über den Index effizient in internen Tabellen auffinden. Dies ermöglicht es dem Kommunikationspartner, den home keygen token später zu rekonstruieren.

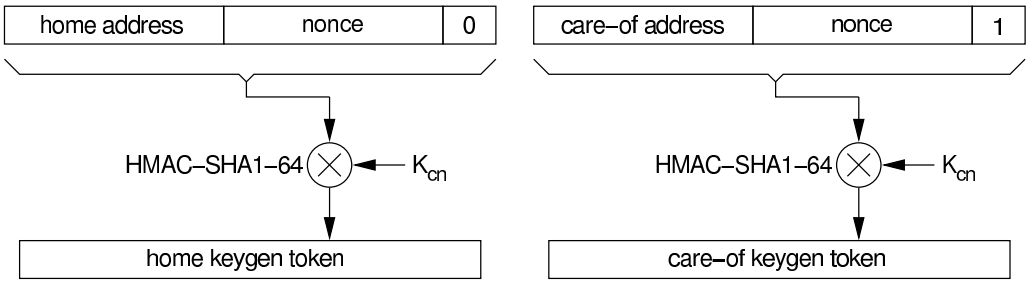


Abbildung 4: Generierung der keygen tokens bei Return Routability

4.2.4 Care-of Test

Die Nachricht Care-of Test wird im Gegensatz zur Nachricht Home Test nicht über den Heimatagenten des mobilen Knotens gesendet, sondern direkt an die Zustelladresse des mobilen Knoten übertragen. Diese Antwort auf die Nachricht Care-of Init Test enthält als Parameter den care-of init cookie, den care-of keygen token und den *care-of nonce index*.

Ebenso wie bei der Nachricht Home Test wird der init cookie - diesmal der care-of init cookie - zurückgesendet, um sicherzustellen, dass die Nachricht Care-of Test nicht von Knoten stammt, die auch nicht auf der Route zum Kommunikationspartner liegen.

Der care-of keygen token berechnet sich aus der Zustelladresse des mobilen Knotens, einer nonce und einem Byte mit dem Wert Eins. Abbildung 4 (rechts) zeigt, dass der Secure Hash Algorithm in der Version 1 als Hashfunktion für den Message Authentication Code eingesetzt

wird. Ausserdem werden wie beim home keygen token nur die ersten 64 Bit für den care-of keygen token verwendet. Ein Nachteil, der dadurch entsteht, dass nur die ersten 64 Bit der eigentlich 160 Bit langen Ausgabe verwendet werden, besteht darin, dass ein potentieller Angreifer weniger Bits vorhersagen muss. Die Vorteile dieser Vorgehensweise werden in [PrvO95] beschrieben.

Der care-of nonce index identifiziert die nonce, die zur Generierung des care-of keygen tokens benutzt wird. Der home nonce index und der care-of nonce index dürfen gleich sein.

4.3 Bindungsaktualisierung

Wenn der mobile Knoten die beiden Nachrichten Home Test und Care-of Test erhalten hat, ist das Return Routability-Protokoll beendet. Der mobile Knoten besitzt nun in Form der beiden keygen tokens die Daten, die er benötigt um eine verifizierbare Bindungsaktualisierung an den Kommunikationspartner zu senden. Hierzu führt der mobile Knoten den Secure Hash Algorithm 1 über den beiden keygen tokens aus und erhält den *binding management key* K_{bm}. Mit diesem Schlüssel wird der Message Authentication Code einer Bindungsaktualisierung erzeugt.

Eine Bindungsaktualisierung enthält als Parameter die Heimatadresse, eine Sequenznummer, den home nonce index, den care-of nonce index und einen Message Authentication Code, der auf 96 Bit gekürzt ist. Diesen erhält man mit dem Schlüssel K_{bm} über den Daten Zustelladresse, Adresse des Kommunikationspartners und Inhalt der selbigen Bindungsaktualisierung.

Sobald der Kommunikationspartner den Message Authentication Code verifiziert hat, kann er einen Eintrag für die Bindung anlegen. Der mobile Knoten hat sich somit authentifiziert, indem er bewiesen hat, dass er über seine beiden benutzten Adressen (Heimat- und Zustelladresse) tatsächlich erreichbar ist.

4.4 Sicherheit

Das Return Routability-Protokoll benötigt keine Zustandsinformationen, daher ist es nicht anfällig für Attacken, die versuchen, Speicherplatz der beteiligten Knoten zu verbrauchen. Dies trifft ebenso auf Prozessorleistung zu, weil keine rechenintensiven Algorithmen eingesetzt werden.

Ein Angreifer kann sich für einen bestimmten mobilen Knoten nicht ausgeben, weil er dazu unter beiden Adressen (Heimat- und Zustelladresse) erreichbar sein muss.

Die Denial of Service-Attacke, bei der der Verkehr eines hohen Bandbreite benötigenden Dienstes auf die gefälschte Zustelladresse eines Opfers geleitet wird, kann durch die Return Routability-Prozedur insofern ausgeschlossen werden, als dass das Protokoll erst abgeschlossen ist, wenn das vermeintliche Opfer auf eine Nachricht, die an seine Zustelladresse gesendet wird, antwortet.

Literatur

[EaJo01] D. Eastlake und P. Jones. US Secure Hash Algorithm 1. RFC 3174, Network Working Group, September 2001.

[JoPA03] D. Johnson, C. Perkins und J. Arkko. Mobility Support in IPv6. Internet Draft, IETF Mobile IP Working Group, Juni 2003.

[KeAt98] S. Kent und R. Atkinson. IP Encapsulating Security Payload. RFC 2406, Network Working Group, November 1998.

[PrvO95] B. Preneel und P. van Oorschot. *Building fast MACs from hash functions*. Springer-Verlag. 1995.

[Roe02] M. Roe. Authentication of Mobile IPv6 Binding Updates and Acknowledgments. Internet Draft, IETF Mobile IP Working Group, Februar 2002.

[Schi03] Jochen Schiller. *Mobilkommunikation*. Pearson Studium. 2003.

Abbildungsverzeichnis

1 triangular routing vs. route optimization 32

2 Generierung der Challenge 34

3 Return Routability 37

4 Generierung der keygen tokens bei Return Routability 38

Mobile IP Version 6 Route Optimization: Effizienz vs. Sicherheit

Mustafa Yilmaz

Kurzfassung

Das Mobile IPv6 (MIPv6) Protokoll ermöglicht die direkte Kommunikation zwischen zwei mobilen Endgeräten, ohne Umweg über den Heimat Agenten (HA). Dieser als Route Optimization bezeichnete Modus schränkt die Mobilität der Endgeräte nicht ein, so dass ein IP Wechsel möglich ist. Mit dem somit verbesserten Routing sind jedoch auch Sicherheitsprobleme verbunden. Jedes Endgerät, dass mit einem anderen Endgerät kommunizieren will, muss dessen aktuelle IP über seinen Heimatagenten anfragen. Da eine Authentifizierung nicht zwingend ist, kann die Antwort auf diese Anfrage abgefangen bzw. verfälscht werden. In diesem Dokument sollen die Sicherheitsbedrohungen und Lösungsansätze betrachtet werden.

1 Einleitung

"Vertraue Allah, aber binde dein Kamel an"
Ägyptisches Sprichwort

Die steigende Anzahl mobiler Endgeräte erfordert die Bewältigung der mit ihr entstehenden Probleme und die Entwicklung neuer Technologien. Ständige Erreichbarkeit und Verfügbarkeit sowie der nahtlose Wechsel zwischen verschiedenen Netzen sind die Anforderungen an die Mobilkommunikation. Mobile IPv6 wurde entwickelt, um den wachsenden Anforderungen der mobilen Welt entgegenzukommen. MIPv6 ermöglicht mobilen Knoten erreichbar zu bleiben, während sie zwischen verschiedenen Netzen wechseln, um somit Mobilität zu gewährleisten. Mit der Einführung dieses neuen Protokolls wurden zahlreiche Verbesserungen gegenüber Mobile IPv4 erzielt. Erfahrungen aus der Entwicklung von MIPv4, sowie die Möglichkeiten von IPv6 wurden bei der Entwicklung dieses Protokolls berücksichtigt und haben zu folgenden Neuerungen geführt:

- Es besteht kein Bedarf mehr an Fremdagenten wie bei MIPv4, die für die Kommunikation zwischen mobilen Knoten nötig waren.
- Durch Nutzen des *"IPv6 Router Advertisement"*s können mobile Knoten ihren aktuellen Aufenthaltsort bestimmen (*Movement Detection*).
- Mobile Knoten erhalten ihre IP Adresse mittels IPv6-Autokonfiguration.
- Die Unterstützung von Authentifizierungsprotokollen wurde in das Protokoll aufgenommen.

Ein weiterer bedeutender Bestandteil ist die Route Optimization (RO), auf die später noch genauer eingegangen wird. Kurzgefasst kann man sagen, dass RO eine effizientere Kommunikation durch Verkürzung der Route ermöglicht. Dabei wird bei der Kommunikation, der Umweg um der Heimatagenten gemieden. Stattdessen können mobile Knoten eine direkte Verbindung mit ihren Kommunikationspartnern aufbauen. Wie später verdeutlicht wird, ergeben sich jedoch durch den Einsatz von RO neue Risiken und Angriffsmöglichkeiten. In dieser Arbeit sollen Sicherheitsprobleme, sowie passende Lösungsansätze im Hinblick auf effizientes Routing erörtert werden.

1.1 Terminologie

Im Folgenden sollen einige Begriffe, die zum Verständnis der Funktionsweise von Mobile IPv6 und dieser Arbeit nötig sind, definiert werden:

Heimatadresse (*home address, HoA*)	Dauerhafte Adresse, unter der ein mobiler Knoten erreichbar ist. Sie befindet sich im Heimatnetz des mobilen Knotens.
Heimatagent (*home agent, HA*)	Ein Router im Heimatnetz des mobilen Knotens, bei dem der mobile Knoten seine aktuelle IP Adresse, die sogenannte care-of-Adresse (CoA), hinterlässt
Zustelladresse (*care-of-address, CoA*)	Adresse, die ein mobiler Knoten in einem fremden Netz erhält
Mobiler Knoten (*mobile node, MN*)	Mobiles Endgerät, welches MIPv6 unterstützt
Kommunikationspartner (*corresponding node, CN*)	Partner, der mit einem mobilen Knoten kommuniziert.
Binding	Assoziation zwischen Heimatadresse eines mobilen Knotens und seiner CoA mit zugehöriger Gültigkeitsdauer
Binding Update (BU)	Vorgang, bei dem ein mobiler Knoten seine aktuelle CoA seinem zugehörigen Heimatagenten oder einem Kommunikationspartner mitteilt.
Binding Cache	Cache, in dem Bindings gespeichert werden, bis deren Gültigkeitsdauer abgelaufen ist

Tabelle 1: Begriffsdefinitionen für MIPv6

1.2 Mobile IPv6

Wie bereits erwähnt, ermöglicht Mobile IPv6 (MIPv6) mobilen Knoten erreichbar zu bleiben, während sie sich im IPv6-Internet bewegen. Dabei sind sie immer unter ihrer Heimatadresse

(HoA) erreichbar, unabhängig von ihrem Standort. Wenn ein mobiler Knoten sich außerhalb seines Heimatnetzes befindet, ist er zusätzlich unter seiner sogenannten care-of-Adresse (CoA) erreichbar. Jeder mobile Knoten, der sich in einem Fremdnetz befindet, ermittelt seine CoA durch die in IPv6 implementierte Autokonfiguration. Der mobile Knoten kann also gleichzeitig und unter zwei verschiedenen Adressen angesprochen werden. Durch die Existenz dieser beiden Adressierungsmöglichkeiten ergeben sich zwei mögliche Szenarien, um zu kommunizieren. So unterstützt MIPv6 auch zwei Modi: "Bidirektionales Tunneling" und "Route Optimization". Auf diese soll in den folgenden zwei Abschnitten genauer eingegangen werden.

1.2.1 Kommunikation ohne Route Optimization

Um die Mobilität bei MIPv6 zu gewährleisten besitzt jeder mobile Knoten einen sogenannten Heimatagenten (HA). Dieser nimmt als eine Art Proxy Pakete, die für den mobilen Knoten bestimmt sind, entgegen und leitet sie weiter an diesen, falls er sich nicht im Heimatnetz befindet. In diesem Fall leitet der Heimatagent die entgegengenommenen Pakete mittels eines Tunnels zu der aktuellen Adresse des mobilen Knotens, der care-of-Adresse (CoA). Dieser Modus wird daher auch als "Bidirektionales Tunneling" bezeichnet. Er wird verwendet, wenn ein mobiler Knoten mit einem Kommunikationspartner kommunzieren will, jedoch nur seine HoA kennt. Diese Lösung ist zwar einfach, erfordert aber eine Umleitung der Pakete über den Heimatagenten und den Aufbau eines Tunnels. Sie verhindert somit die direkte Kommunikation zwischen zwei Knoten. Route Optimization soll hierbei Abhilfe schaffen.

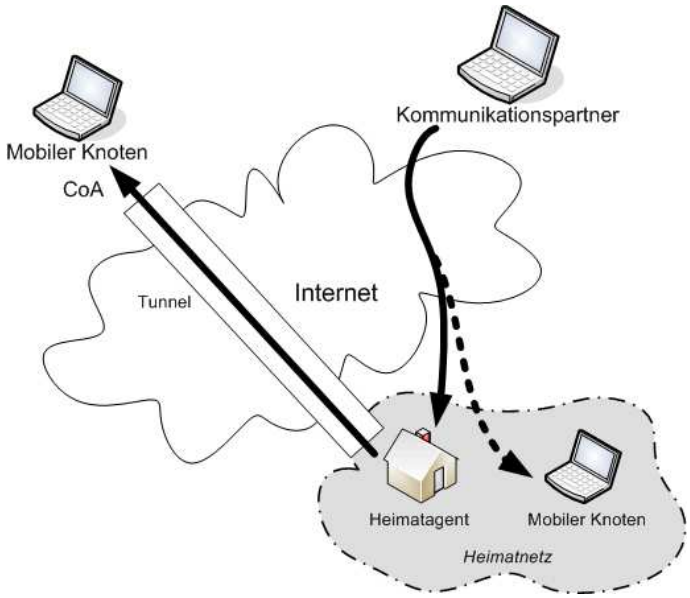


Abbildung 1: Kommunikation ohne Route Optimization

1.2.2 Route Optimization

Der in MIPv6 implementierte Modus Route Optimization ermöglicht die direkte Kommunikation zwischen zwei Knoten. Dabei teilt der mobile Knoten seine CoA nicht nur seinem Heimatagenten, sondern auch dem Kommunikationspartner mit. Dazu sendet der mobile Knoten seine aktuelle CoA dem Kommunikationspartner mittels einer Binding-Update-Nachricht. Jeder Knoten verwaltet eine Liste mit Assoziationen zwischen Heimatadresse und CoA aller Knoten, mit denen kommuniziert werden wird. Die genannten Assoziationen werden als Bindings bezeichnet. Die Liste der Bindings bildet den Inhalt des Binding Cache (auch Routing-Tabelle

genannt). Um Aktualität im Binding Cache zu gewährleisten, können Kommunikationspartner auch aufgefordert werden wiederholt Binding Updates zu verschicken. Dazu reicht es eine Binding Request-Nachricht an den Knoten zu schicken.

1.3 Effizienz versus Sicherheit

Mit der Verwendung von Route Optimization sind zahlreiche Vorteile verbunden. Einer der wichtigsten Vorteile ist die Verkürzung der verwendeten Route. Anstatt den Umweg über den Heimatagenten zu gehen, wird der viel kürzere und somit effizientere, direkte Weg zum mobilen Knoten gewählt. Zudem wird der Heimatagent entlastet, da er nicht mehr als Proxy für sämtlichen Verkehr zum mobilen Knoten dienen muss. Die Verwendung eines Tunnels und der mit ihm verbundene Overhead, der durch die Kapselung der Pakete entsteht, ist ebenfalls nicht mehr notwendig. Jedoch ergeben sich mit der Verwendung von RO auch neue Sicherheitsfragen. So muss ein mobiler Knoten seine aktuelle IP Adresse seinem Kommunikationspartner preisgeben. Dieser kann somit durch das Netzpräfix der IP Adresse eine Art Lokalisierung durchführen und auf den aktuellen Aufenthaltsort des mobilen Knotens schliessen. Ein anderes Problem entsteht dadurch, dass der Kommunikationspartner sich in einem beliebigen fremden Netz aufhalten und somit eine nahezu beliebige IP Adresse haben kann. Um eine sichere Kommunikation zu gewährleisten, muss deswegen die Authentizität eines Binding Updates überprüft werden können. Dadurch kann verhindert werden, dass sich unbeteiligte Dritte als Kommunikationspartner ausgeben. Weiterhin muss hinterfragt werden, ob unsinnige Binding Updates dazu genutzt werden können, Kommunikation zwischen zwei Knoten zu stören. Insbesondere muss hierbei das Fluten mit Binding Update Nachrichten betrachtet werden. Im Folgenden sollen die oben genannten Probleme genauer betrachtet und nach Lösungen gesucht werden.

2 Angriffe auf Route Optimization

Im Folgenden sollen typische Angriffe auf Route Optimization betrachtet werden. Alle Angriffe verbindet die Gemeinsamkeit gefälschte Binding Update Nachrichten zu benutzen, um ihr Ziel zu erreichen. Die dadurch entstehenden Einträge im Binding Cache der mobilen Knoten, haben verschiedenste Auswirkungen und verändern das Verhalten der beteiligten Knoten. Die dadurch möglichen Angriffe lassen sich in drei Kategorien unterteilen:

- Angriffe auf die Vertraulichkeit (Abhören von Kommunikation)
- Angriffe auf die Integrität (Verändern von Nachrichten)
- Angriffe auf die Verfügbarkeit (DoS-Angriffe, Umleiten von Verkehr)

Bei der Implementierung von Protokollen, die zur Authentifizierung von BUs dienen, müssen diese Angriffe berücksichtigt und Mechanismen zum Schutz entworfen werden.

2.1 Authentifizierungsproblem

MIPv6 erlaubt es, das zu verwendende Authentifizierungsprotokoll frei zu wählen. Diese Protokolle sollten gewisse Vorraussetzungen erfüllen, um eine sichere Kommunikation zu ermöglichen. So gibt es zahlreiche Gefahren und Angriffstypen, die bei der Entwicklung eines solchen Protokolls berücksichtigt werden müssen. Beispielsweise kann sich ein Angreifer als Kommunikationspartner ausgeben oder den Datenverkehr zwischen zwei Knoten abfangen, verändern und mithören. Im Folgenden sollen zwei konkrete Angriffe auf RO betrachtet werden, die sich dieses Problem zu Nutze machen.

2.1.1 Spoofing von Binding Updates

Falls keinerlei Authentifizierung eingesetzt wird, kann ein Angreifer gefälschte Binding Updates an sein Opfer schicken. Alle mobilen Knoten sind von diesem Angriff betroffen, da sie alle Bindings Updates ungeprüft in ihren Binding Cache aufnehmen. Anhand der Absenderadresse ist es nicht möglich festzustellen, ob der Absender wirklich der Kommunikationspartner ist. Angreifer können durch diese Methode Verkehr zwischen zwei Knoten zu einer dritten beliebigen Adresse umleiten. Wenn A und B zwei mobile Knoten sind, so kann ein Angreifer dem mobilen Knoten A ein Binding Update mit der Information, dass als Heimatadresse B und als care-of-Adresse C verwendet werden soll, senden. Nach Empfang dieses Binding Updates wird der Knoten A alle Pakete, die für B bestimmt sind, an den Knoten C schicken, unter der Annahme, dass sich B unter diese Adresse befindet. Wenn der Angreifer seine eigene Adresse als CoA angibt, kann er den gesamten Verkehr zwischen A und B abhören oder sogar die Verbindung übernehmen (*session highjacking*). Da Einträge im Binding Cache nur begrenzte Zeit gültig sind, muss der Angreifer das Binding Update in bestimmten Abständen wiederholen, um den Eintrag im Binding Cache des Opfers zu aktualisieren. Um den auch als *"man-in-the-middle"* bekannten Angriff durchzuführen, müssen dem Angreifer sowohl die IP des Senders, als auch die IP des Empfängers bekannt sein. Daher gestaltet es sich schwer, den gesamten Verkehr zu bzw. von einem mobilen Knoten abzuhören. Ohne den Einsatz von RO gestaltet sich ein derartiger Angriff viel schwieriger, denn der Angreifer muss direkt auf der Route zwischen zwei Kommunikationspartnern liegen, um die Kommunikation abhören zu können.

2.1.2 Wiederholen bzw. Abfangen von Binding Updates

Wiederholungsangriffe (*replay attacks*) können eine Gefahr für Protokolle darstellen, die ein Binding Update auf Authenizität überprüfen sollen. So werden bei manchen Protokollen beispielsweise Zeitstempel verwendet um die Aktualität eines Binding Updates zu prüfen, um sicherzugehen, dass durch veraltete Einträge keine Daten zu einer nicht mehr gültigen Adresse geschickt werden. Dabei wird nach Ablauf der Gültigkeitsdauer ein Binding Update angefordert. Wechselt jedoch ein mobiler Knoten in kurzer Zeit das Netz, nämlich bevor die Gültigkeitsdauer eines Bindings abgelaufen ist, so ist es möglich mittels eines Wiederholungsangriffes, die für ihn bestimmten Daten in das Netz, in dem er sich zuvor befunden hat, zu schicken. Der Einsatz von Sequenznummern kann diese Art von Angriffen verhindern. Wird ein Paket abgefangen und erneut geschickt um einen Angriff zu starten, so wird das Paket nicht akzeptiert, da seine Sequenznummer bereits verwendet wurde und eine höhere Sequenznummer erwartet wird. Ein weiterer Angriff besteht darin, die Binding Update Nachrichten eines Opfers nach vollzogenem Netzwechsel abzufangen. Wechselt ein mobiler Knoten in ein neues Netz, muss ein Angreifer sämtliche Binding Update Nachrichten dieses mobilen Knotens stören. So kann sich der Angreifer in dem Netz, in dem sich der mobile Knoten zuvor befunden hat, als sein Opfer ausgeben und somit die Verbindung seines Opfers übernehmen. Beide aufgeführten Angriffe stellen keine große Gefahr da. Denn um die Angriffe ausführen zu können, muss der Angreifer sich in den selben Netzen aufhalten, in dem sich das Opfer aufgehalten hat. Ein Angriff von einem beliebigen Netz aus, wie bei den anderen Angriffen, ist nicht möglich.

2.2 Anwesenheitsproblem

Einfache DoS-Angriffe (Denial of Service)

Wie bereits erläutert, ist es Angreifern möglich Verkehr zwischen zwei Knoten zu einer beliebigen Adresse umzuleiten. Auf diese Weise kann Kommunikation zwischen Knoten gestört

oder auch unterbrochen werden. Es genügt, den Verkehr zu einer unbeteiligten dritten Partei oder gar zu einer nicht existenten Adresse umzuleiten. Besteht beispielsweise eine Verbindung zwischen zwei mobilen Knoten A und B, so kann ein Angreifer dem Knoten A ein gefälschtes Binding Update senden. Im Binding Update gibt sich der Angreifer als B aus und nennt als seine aktuelle CoA eine nicht existente IP Adresse. Dieser Vorgang muß regelmäßig wiederholt werden, um den Binding Cache von Knoten A aktuell zu halten. Während dem Angriff ist der Knoten B von Knoten A aus nicht mehr erreichbar und Knoten A erhält ICMP Fehlernachrichten. Diese Angriffsmöglichkeit ist besonders bedrohlich, da nicht nur mobile Knoten, sondern jeder beliebige Knoten im Internet potentiell gefährdet ist.

2.2.1 Bombing Angriffe

Wie bereits festgestellt, können Angreifer den Datenverkehr zwischen zwei Knoten zu einer beliebigen Adresse umleiten, falls keine Sicherheitsmechanismen verwendet werden. Dies kann dazu genutzt werden einen Knoten mit einer großen Menge an Daten zu bombardieren (*bombing attack*). Es können durch Umleitung des Verkehrs zu mehreren Adressen auch gesamte Netze angegriffen werden. Zum Durchführen dieses Angriffes ist es nötig eine Verbindung zwischen zwei Knoten A und B zu finden, bei der große Datenmengen versendet werden. Durch Umleiten dieser Daten zu einem dritten Knoten C kann dieser angegriffen werden. Dieser einfache Angriff ist aber nicht sehr effektiv, denn der Knoten A würde schon bald das Versenden der Daten stoppen, da er keine Empfangsbestätigung mehr von B erhält. Angreifer können dies verhindern, in dem sie sich als Knoten B ausgeben. Um beispielsweise einen Multimedia-Stream, der kontinuierlich Daten liefert, als Quelle für Angriffe zu mißbrauchen müssten sie im einzelnen

- einen Datenstrom von Quelle A anfordern und eventuell beim Verbindungsaufbau auftretende Sequenznummern in Erfahrung bringen,
- den Datenstrom an das Opfer, Knoten C, umleiten und schließlich
- regelmäßig Empfangsbestätigungen an die Quelle, Knoten A senden, um die Verbindung aufrechtzuhalten.

Dabei kommt beim Versenden der Empfangsbestätigungen, je nach verwendetem Protokoll, die im ersten Schritt gewonnene Sequenznummer zum Einsatz. Zusätzlich muss der Angreifer regelmäßig Binding Updates an die Quelle A schicken, um die Einträge im Binding Cache aktuell zu halten. Bombing Angriffe sind besonders bedrohlich, denn jeder Knoten im Internet ist potentiell betroffen. Die Wirkung des Angriffes kann verstärkt werden, indem mehrere Datenquellen gleichzeitig als Quelle benutzt werden. Dieser unter dem Namen DDoS (*Distributed Denial-of-Service*) bekannte Angriff bündelt die Datenströme unterschiedlicher Quellen und leitet sie an ein Angriffsziel um. Im Gegensatz zu DDoS-Angriffen im heutigen Internet sind keine Viren bzw. Würmer notwendig, um den nötigen Datenstrom zu erzeugen. Ein einzelner Angreifer muss nur wenig Daten zum Einleiten und Durchführen des Angriffs erzeugen, da der eigentliche Datenstrom von unabhängigen Dritten erzeugt wird. Um die genannten Angriffe durchführen zu können, muss der Angreifer die IP Adresse seines Angriffszieles kennen und eine oder mehrere Quellen finden, die keine Authorisierung beim Verbindungsaufbau verlangen. Eine raffinierte Variante des Bombing Angriffs besteht darin, ein Binding Update an eine Quelle zu senden, in dem als Heimatagent das Angriffsziel und als Empfänger die eigene IP Adresse angegeben wird. Wurde eine Verbindung erfogreich aufgebaut, sendet der Angreifer dem Quellknoten eine Aufforderung, das zuvor gesendete Binding Update aus seinem Binding Cache zu entfernen (*binding update cancellation message*) oder wartet darauf, dass die Gültigkeitsdauer des Bindings abläuft. Der Knoten, der als Quelle dient, sendet von

nun an die Daten nicht mehr an den Angreifer, sondern an das Angriffsziel, das er für den Heimatagenten des Angreifers hält.

2.3 Resourcenmißbrauch

Verkehr, der zum Verbindungsaufbau dient, sowie möglicherweise eingesetzte Authentifizierung beanspruchen die Ressourcen eines mobilen Knotens. Auch wenn dieser Ressourcenanspruch verhältnismäßig gering ist, kann er eine Bedrohung darstellen, wenn ein Knoten innerhalb kurzer Zeit zahlreiche Anfragen erhält. Im Folgenden sollen Angriffe betrachtet werden, die sich diese Eigenschaft von mobilen Knoten zu Nutze machen und die Verfügbarkeit beeinträchtigen.

2.3.1 Flooding mit unnötigen Binding Updates

Sobald ein mobiler Knoten ein IP-Paket von einem neuen Kommunikationspartner mittels seines Heimatagenten erhält, sendet er automatisch ein Binding Update an seinen neuen Kommunikationspartner. Angreifer können dieses Verhalten ausnutzen, in dem sie Pakete mit gefälschter Absenderadresse an ihr Opfer schicken. Dazu reichen einfache Ping- oder TCP-SYN-Pakete aus. Durch das Fluten mit Paketen unterschiedlicher Absenderadressen ist der mobile Knoten damit beschäftigt, Binding Update-Nachrichten zu versenden, und verbraucht damit einen großen Teil seiner Ressourcen. Binding Updates können auch dazu genutzt werden, beliebige Knoten im Internet anzugreifen. Dazu schickt der Angreifer einer möglichst großen Anzahl von mobilen Knoten ein IP-Paket, das als Absenderadresse die Adresse seines Opfers enthält. Alle mobilen Knoten, die nun diese Pakete erhalten haben, schicken daraufhin Binding Update-Nachrichten an das Opfer. Das Opfer ist mit der Bearbeitung der eintreffenden Nachrichten beschäftigt und verbraucht einen beträchtlichen Teil seiner Ressourcen.

2.3.2 Ressourcenmißbrauch bei Authorisierungsverfahren

Um einen Schutz gegen einige der oben genannten Sicherheitsprobleme zu gewährleisten, werden Authentifizierungsprotokolle eingesetzt. Jedoch bringt der Einsatz dieser auch neue Angriffsmöglichkeiten mit sich. Nehmen wir an, dass ein Authorisierungsverfahren verwendet wird, das Kryptographie verwendet. Wenn nun ein mobiler Knoten eine Verbindung zu einem Kommunikationspartner aufbaut, ist dieser gezwungen aufwändige kryptographische Berechnungen durchzuführen, um die Authentizität des Absenders zu prüfen. Durch das Fluten eines Knotens mit unterschiedlich signierten Nachrichten, kann man ihn dazu zwingen seine Ressourcen für die Prüfung dieser Nachrichten zu verbrauchen und bremst oder unterbricht somit seine sonstige Kommunikation.

2.3.3 Erzwingen von nicht optimiertem Routing

Ein mobiler Knoten muss in seinem Binding Cache die aktuelle CoA seines Kommunikationspartners vorliegen haben, damit er mit diesem mittels Route Optimization kommunizieren kann. Angreifer können dieses Verhalten ausnutzen, in dem sie den Binding Cache des Opfers durch Senden von Binding Updates mit gefälschten unnötigen Einträgen füllen. Das Opfer ist gezwungen Daten, die er seinem Kommunikationspartner senden will, zu dem entsprechenden Heimatagenten zu schicken und kann somit kein Gebrauch von optimiertem Routing machen. Selbst wenn eine Authentifizierung bei Binding Updates gefordert wird, können Angreifer authentische, aber sinnlose Binding Updates erzeugen.

3 Authentifizierung von Binding Updates

Wie wir gesehen haben, werden alle Angriffe durch das Manipulieren von Routing Caches durch gefälschte Binding Updates verursacht. Eine naheliegende Lösung ist daher der Einsatz von Authentifizierungsverfahren bei Binding Updates. Die folgenden Abschnitte stellen einen kleinen Überblick über verschiedene Verfahren dar. Im Vergleich soll deutlich werden, welche Verfahren einen Schutz gegen oben genannte Angriffe bieten und welche Schwächen sie besitzen. Keines der Verfahren kann einen vollständigen Schutz gegen alle Angriffstypen bieten und kann sogar neue Schwachstellen hervorbringen. Den besten Schutz bieten Verfahren, die auf Kryptographie oder auf Prüfung der Routen zum Kommunikationspartner basieren.

3.1 Public Key Verfahren

Eine naheliegende Lösung ist die Nutzung von Public Key Authentifizierung. Bereits bekannte und bewährte Verfahren wie IKE oder PKI könnten hierzu genutzt werden. Sie bieten einen hohen Grad an Sicherheit durch die eingesetzten Verschlüsselungsverfahren. Jedoch gibt es zwei entscheidende Gründe die gegen einen Einsatz sprechen. Einer dieser Gründe ist, dass bei der Entwicklung dieser Verfahren mobile Endgeräte nicht berücksichtigt worden sind. Es wurde davon ausgegangen, dass außreichend Rechenkapazität wie bei einem Arbeitsplatzrechner zur Verfügung steht. Dies hat den Nachteil, dass mobile Knoten, wie beispielsweise Handys oder PDAs, den für die kryptographischen Berechnungen benötigten Rechenaufwand eventuell nicht bereitstellen können. Ausserdem wurden diese Verfahren für den Einsatz auf Anwendungsebene entworfen. Der Einsatz auf der Vermittlungsschicht, der mit dem Authentifizieren von Binding Updates notwendig ist, würde viel mehr Overhead verursachen, da hierbei mit häufigerem Einsatz der Authentifizierung gerechnet werden muss. Als zusätzliche Hürde, die gegen eine Verwendung spricht, ist die Notwendigkeit eine globale Public Key Infrastruktur aufzubauen, welche mit hohem Aufwand verbunden ist. Jedoch gibt es auch Szenarien, bei denen eine Verwendung dieser Verfahren sinnvoll ist. In geschlossenen Benutzergruppen, Intranetzen und Umgebungen, die hohe Sicherheitsanforderungen verlangen, kann der Aufbau einer PKI eine realisierbare Lösung darstellen.

3.2 Kryptographisch generierte Adressen

Kryptographisch generierte Adressen wurden erstmals in dem als CAM (*child-proof Authentication for MIPv6*) bezeichneten Authentifzierungsprotokoll verwendet. Die grundlegende Idee dabei ist, die letzten Bits der 64 Bit langen IP Adresse (*Interface Identifier*) durch eine Hashfunktion zu berechnen. Dazu wird die gewählte Hashfunktion auf den öffentlichen Schlüssel des mobilen Knotens angewendet. Binding Updates können somit mittels dieses Schlüssels signiert werden. Eine Hashfunktion erschwert es Angreifern einen Schlüssel zu finden, der zu einer gegebenen Adresse passt. Dieser könnte verwendet werden um signierte BUs zu generieren. Ein großer Vorteil dieses Verfahrens ist, dass er auf öffentlichen Schlüsseln basiert und somit kein Aufbau einer Infrastruktur wie etwa bei der PKI nötig ist. Leider hat auch dieses Verfahren einen Schwachpunkt. Da nur weniger Bits als für den Hashwert verwendet werden können, ist ein Erraten dieser Bits durch einen Brute Force Angriff möglich.

3.3 Return Routability

Ein Verfahren, das nicht auf Kryptographie basiert, ist Return Routability (RR), auf das im Rahmen dieser Arbeit nicht detailliert eingegangen werden kann. Vereinfacht kann man sagen, dass bei der RR Prozedur ein mobiler Knoten zwei Pakete, nämlich eine an die HoA und

Lösung	Vorteile	Nachteile
Public Key Verfahren	- Starke Verschlüsselung	- erfordert hohe Rechenleistung - Aufbau einer Infrastruktur notwendig
Kryptographisch generierte Adressen	- Keine spezielle Infrastruktur nötig - Einfach zu implementieren	- Anfällig gegen Brute Force Angriffe - Bombing Angriffe sind möglich
Return Routability	- Schutz gegen Bombing Angriffe	- Angreifbar mit erhöhtem Aufwand

Tabelle 2: Übersicht der Authentifizierungsverfahren

eine an die CoA seines Kommunikationspartners, schickt. Binding Updates vom Kommunikationspartner werden nur dann akzeptiert, wenn dieser beide Pakete erhalten hat. Somit soll gewährleistet werden, dass kein Angreifer die Route zwischen den Knoten manipuliert hat. Angriffe sind nur dann einfach durchführbar, wenn sich der Angreifer im Netz des Kommunikationspartners aufhält und beide Pakete abfangen kann. Auch dieses Verfahren hat seine Schwächen und ist angreifbar. Wie aus Tabelle 2 zu entnehmen ist, bietet er einen Schutz gegen Bombing Angriffe, deren Ziel die CoA des mobilen Knotens ist.

4 Lösungen gegen Ressourcenmißbrauch

Alle bisher betrachteten Authentifizierungsverfahren bieten Schutz gegen Angriffe auf Authentizität und Integrität. Um zusätzlich die Verfügbarkeit zu gewährleisten, werden weitere Anforderungen an das Authentifizierungsprotokoll gestellt. Die besprochenen Verfahren bieten einen Schutz gegen einfache DoS-Angriffe durch Unterbrechen oder Umleiten von Datenverkehr um die Verfügbarkeit zu erhalten. Andererseits erleichtern sie Angriffe mittels DDoS-Angriffen (*Distributed Denial of Service*) oder Bombing Angriffe. Denn jede Authentifizierung bei den besprochenen Verfahren benötigt zusätzliche Rechenleistung, um beispielsweise kryptographische Berechnungen durchzuführen. So können Angriffe, bei denen in kurzer Zeit zahlreiche Authentifizierungsvorgänge erzwungen werden, die Verfügbarkeit eines Systems beinträchtigen. Im Folgenden sollen einige Verfahren betrachtet werden, die bei der Entwicklung von Authentifizierungsprotokollen berücksichtigt werden sollten.

4.1 Einsatz von Verzögerung

Eine Möglichkeit um ein System gegen DoS-Angriffe zu schützen ist, die Zusicherung von Ressourcen hinauszuzögern, bis davon ausgegangen werden kann, dass es sich bei dem Kommunikationspartner um keinen Angreifer handelt. Beispielsweise könnte ein Knoten, der einen Verbindungsaufbau initiiert hat, dazu aufgefordert werden ein mathematisches Problem zu lösen oder sich zu authentifizieren, bevor Ressourcen für die Anwendung des verwendeten Protokolls reserviert werden. Eine weiteres Verfahren besteht darin, beim Verbindungsaufbau eine schwache Authentifizierung vor eine aufwendige kryptographische Authentifizierung zu schalten. Diese bildet eine weitere Hürde für den Angreifer und schont die Ressourcen des Angegriffenen. Eine weitere effektive Möglichkeit zum Schutz vor DoS-Angriffen besteht darin, die Authentifizierung zustandslos zu gestalten, um während der Authentifizierungsphase keinen Arbeitsspeicher reservieren zu müssen. Dieses Verfahren findet beispielsweise bei der "Return Routability" Verwendung.

4.2 Einschränkung der Ressourcen

Um sich gegen Ressourcenmißbrauch zu schützen, kann ein mobiler Knoten die Ressourcen, die er für Authentifzierung verwenden möchte, einschränken. So kann er gewährleisten auch bei hoher Last stabil arbeiten zu können. Dazu wird die Prozessorzeit, der Speicher und die Bandbreite, die für das Authentifizieren von Binding Updates zur Verfügung stehen, eingeschränkt. Liegen zu viele Anfragen vor, so dass die Kapazität aufgebraucht ist, werden alle weiteren Authentifizierungsvorgänge gestoppt. Einträge, die bereits im Binding Cache vorhanden sind, können bei Bedarf aktualisiert werden. So wird die Route Optimization bei alten Verbindungen weiterhin verwendet. Neue Verbindungen können hingegen keinen Gebrauch von ihr machen. Nimmt die Anzahl der Authentifierzungsanfragen ab, so kann der mobile Knoten davon ausgehen, dass der Angriff vorbei ist und zu regulärem Betrieb zurückkehren.

4.3 Bevorzugen bekannter Knoten

Ein mobiler Knoten kann, um sich vor Ressourcenmißbrauch zu schützen, eine Liste von bevorzugten Kommunikationspartnern führen, um ihnen bei einem Ansturm von Authentifizierungsanfragen Vorrang zu gewähren. So könnten beispielsweise Kommunikationspartner, mit denen in der Vergangenheit problemlos kommuniziert wurde, eine höhere Priorität erhalten. Ein erfolgreiche Verbindung auf einer höheren Protokollschicht könnte ebenfalls als Kriterium dienen eine höhere Priorität zu vergeben.

5 Fazit

Die Vorteile von Route Optimization hinsichtlich effizientem Routing sind enorm. Sie garantiert kürzere Routen und die Entlastung des Heimatagenten, so dass ein Einsatz unverzichtbar ist. Mit dem Einsatz von RO ist die Verwendung eines geeigneten Authentizierungsprotokolls zu empfehlen, um sich gegen die Angriffsmöglichkeiten, die RO eröffnet zu schützen. Dabei sollte die Auswahl des Protokolls anhand von Anforderungen der Benutzer getroffen werden. Kleine Benutzergruppen oder firmeninterne Intranetze ermöglichen es aufwändigere Protokolle zu nutzen, um etwa eine PKI aufzubauen. Im globalen Internet sind deswegen Verfahren, die keine spezielle Infrastruktur benötigen, sicherlich besser geeignet. Bei allen eingesetzten Verfahren sollte nicht vergessen werden, dass keines der oben genannten Authentifizierungsverfahren immun gegen sämtliche Angriffe ist. Während Angriffe auf die Integrität und die Authentizität verhindert werden können, sind es vor allem Angriffe gegen die Verfügbarkeit eines mobilen Knotens, die schwer abzuwehren sind.

Literatur

[AuAr02] T. Aura und J. Arkko. *MIPv6 BU Attacks and Defenses*. Internet Draft. abgelaufen im August 2002.

[fTel04] Institut für Telematik. *Vorlesung Mobilkommunikation (Kapitel 6.4)*. Universität Karlsruhe (TU). Sommersemester 2004.

[JoPA03] D. Johnson, C. Perkins und J. Arkko. *Mobility Support in IPv6*. Internet Draft. abgelaufen im Dezember 2003.

[Schi03] Jochen Schiller. *Mobilkommunikation*. Pearson. 2003.

[Solo98] J. D. Solomon. *Mobile IP, The Internet Unplugged*. Prentice Hall. 1998.

Abbildungsverzeichnis

1 Kommunikation ohne Route Optimization 43

Tabellenverzeichnis

1 Begriffsdefinitionen für MIPv6 42

2 Übersicht der Authentifizierungsverfahren 49

Child-Proof Authentication for Mobile IPv6 (CAM)

Constantin Schimmel

Kurzfassung

Das Child-proof Authentication Protokoll basiert auf einer kryptographisch generierten Heimatadresse in Mobile IPv6. Der mobile Knoten authentifiziert seine Heimatadresse über die Kenntnis eines geheimen, asymetrischen Schlüssels. Das Verfahren ist schnell, effizient und einfach zu implementieren. Es genügt, eine einzige Nachricht an den Kommunikationspartner zu senden. Dieser überprüft, ob die Heimatadresse mit dem übermittelten, öffentlichen Schlüssel verbunden ist. Anschliessend wird getestet, ob der mobile Knoten auch im Besitz des geheimen Schlüssels ist.

1 Einleitung

Mobilität ist in der heutigen Zeit von großer Bedeutung. Den Benutzern stehen immer mehr Kommunikations-Dienstleistungen auf kleinen, portablen Geräten zur Verfügung. Diese können sich an beliebigen Stellen in fremde Netze einbinden und weiterhin erreichbar sein. Ihre gemeinsame Schnittstelle ist das Internetprotokoll IP. Leider konnte während der Entwicklung des Internets keiner den aktuellen Boom mobiler Endgeräte vorausahnen. IP verlangt eine topologisch korrekte Adresse. Aus diesem Grund wird in IP standardmäßig keine Mobilität unterstützt. Ein neues Protokoll, das mit diesen Anforderungen zurechtkommt, musste geschaffen werden.

Mobilität bedeutet, dass zu jeder Zeit, auch beim Übergang von einem Subnetz ins andere, die Verbindung eines mobilen Teilnehmers zum Netz aufrecht erhalten wird. Will ein mobiler Teilnehmer im Netz erreichbar sein, so darf sich seine IP Adresse bei einem Ortswechsel nicht ändern. Aus diesen Anforderungen entstand Mobile IP. Die neuste Version des Protokolls ist Mobile IPv6 (MIPv6).

Eine MIPv6 Adresse ist immernoch topologisch korrekt. Dennoch kann ein mobiler Knoten (Mobile Node, MN) bei Mobile IPv6 in andere Subnetze wechseln, ohne die Verbindung zu verlieren. Er bleibt ununterbrochen über seine Heimatadresse in seinem Heimatnetz (Home Network) erreichbar. Wenn sich der mobile Knoten nicht in seinem Heimatnetz befindet, dann tunnelt ein Heimatagent (Home Agent, HA) im Heimatnetz alle an ihn gerichteten Nachrichtenpakete an die Zustelladresse (Care-of Address, COA). Die Zustelladresse bezieht sich immer auf den aktuellen Aufenthaltsort des mobilen Knoten. Tunneln bedeutet, dass der Stellvertreter eine empfangene Nachricht komplett in den Nutzanteil eines neuen IP-Paketes kapselt und anschliessend dieses Datenpaket an den mobilen Knoten weiterleitet.

Dieser Ansatz hat ein schwerwiegendes Problem. Wenn sich der mobile Knoten in einem fremden Netz befindet, werden die Nachrichtenpakete immer über den Heimatagenten getunnelt. Das kann unter Umständen einen weiten Umweg bedeuten. Abhilfe dagegen schafft Route Optimization. Hier versuchen die Kommunikationspartener direkt miteinander zu kommunizieren, auch wenn sie sich nicht in ihrem Heimatnetz befinden. Bei Route Optimization teilt der mobile Knoten seinem Heimatagenten die Heimatadresse, die

Zustelladresse und die Gültigkeitsdauer über eine Binding Update Nachricht mit. Der Heimatagent merkt sich die Binding Updates, damit er Nachrichtenpakete, die an den mobilen Knoten gerichtet sind, weiterleiten kann. Erhält der mobile Knoten von einem Kommunikationspartner (Correspondent Node, CN) Nachrichtenpakete, die über den Heimatagenten getunnelt wurden, so kann er auch eine Binding Update Nachricht an den Kommunikationspartner schicken. Dieser kann nun Nachrichtenpakete direkt an die Zustelladresse schicken. Die Kommunikationsverbindung ist dadurch schneller und der Heimatagent wird entlastet. Der Kommunikationspartner und der Heimatagent können den mobilen Knoten über Binding Request Nachrichten auffordern, die Binding Updates zu erneuern. Wurde eine Binding Update Nachricht erhalten, so wird diese mit einem Binding Acknowledgement beantwortet. Abbildung 1 zeigt den Aufbau einer Routing Optimization Verbindung.

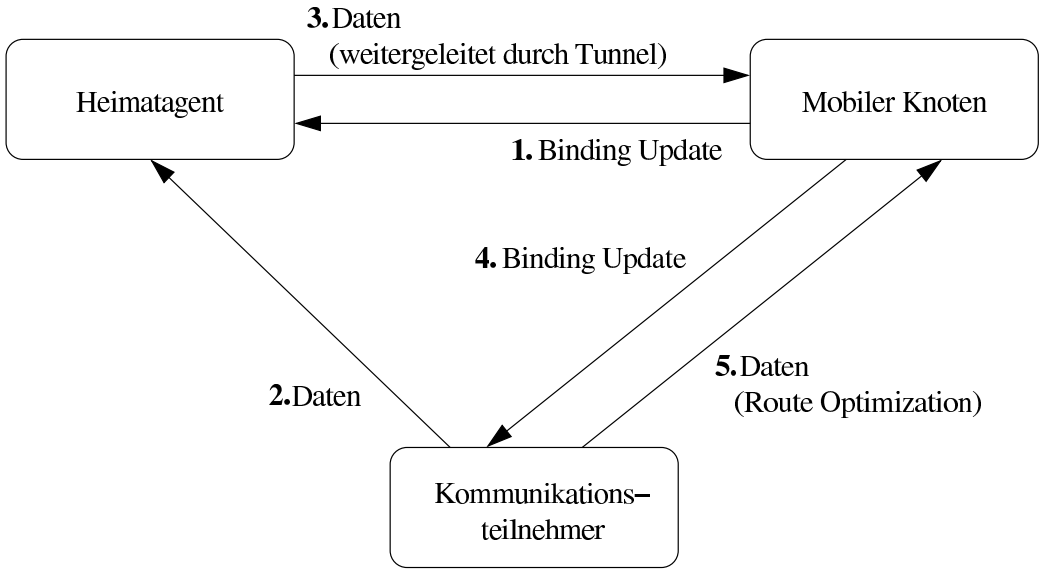


Abbildung 1: Ablauf von Route Optimization

Ein Angreifer kann diese Binding Updates ausnutzen, um durch diverse Attacken anderen Netzwerkteilnehmern zu schaden. Der Einsatz von Authentifizierungsprotokollen soll diese Attacken unterbinden. Sie sollen überprüfen, ob die Angaben des Kommunikationspartners glaubwürdig sind. Das Child-proof Authentication Protokoll sieht seine Aufgabe darin, die Binding Updates auf eine einfache, schnelle und zugleich sichere Weise gegen Missbrauch abzusichern.

Kapitel 2 beschreibt kurz die Funktionsweise und den Aufbau von kryptographisch generierten Adressen. Kapitel 3 befasst sich mit dem Child-Proof Authentication Protokoll. Zuerst wird die verwendete Notation eingeführt. Anschließend wird der Aufbau einer Heimatadresse beschrieben. Darauf folgt der Ablauf des Child-proof Authentication Protokolls. Danach werden der gebotene Schutz und die möglichen Attacken aufgeführt. Einige Vorteile des Child-proof Authentication Protokolls werden in Kapitel 4 zusammengefasst. Das 5. Kapitel fasst die Eigenschaften des Protokolls nochmal in wenigen Worten zusammen.

2 Cryptographically Generated Addresses (CGA)

Das Child-proof Authentication Protokoll basiert auf einer kryptographisch generierten Heimatadresse. An dieser Stelle soll daher kurz auf den Aufbau und das Prinzip einer kryptographisch generierten Adresse eingegangen werden.

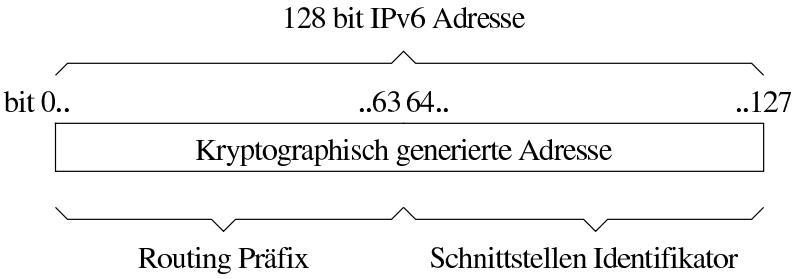


Abbildung 2: Aufbau einer kryptographisch generierten Adresse

Abbildung 2 zeigt das Adressformat einer kryptographisch generierten Adresse. Die ersten 64 bit ergeben sich aus dem Routing-Präfix. Die letzten 64 bit ergeben den Schnittstellen-Identifikator. Für mehr Informationen sei der Leser auf [Aura03] verwiesen.

Kryptographisch generierte Adressen sind IPv6 Adressen, deren Schnittstellen-Identifikatoren aus einem nicht invertierbaren Haschwert eines öffentlichen Schlüssels des Adress-Inhabers und des Routing-Präfix der Adresse gebildet werden. Eine nicht invertierbare Haschfunktion bildet eine beliebig lange Bitfolge auf eine Bitfolge konstanter Länge ab. Dabei ist es nicht mehr möglich, aus dem Haschwert wieder die Ursprüngliche Bitfolge zurückzugewinnen. Der Adress-Inhaber kann mit Hilfe des zugehörigen geheimen Schlüssels die Korrektheit der Adresse beweisen ("Proof of Ownership"). Kryptographisch generierte Adressen sind somit an das asymetrische Schlüsselpaar gebunden. Dieses Schlüsselpaar kann später für die Signierung von Binding Updates verwendet werden. Abbildung 3 zeigt das Schema einer Signierung mittels asymetrischer Verschlüsselung.

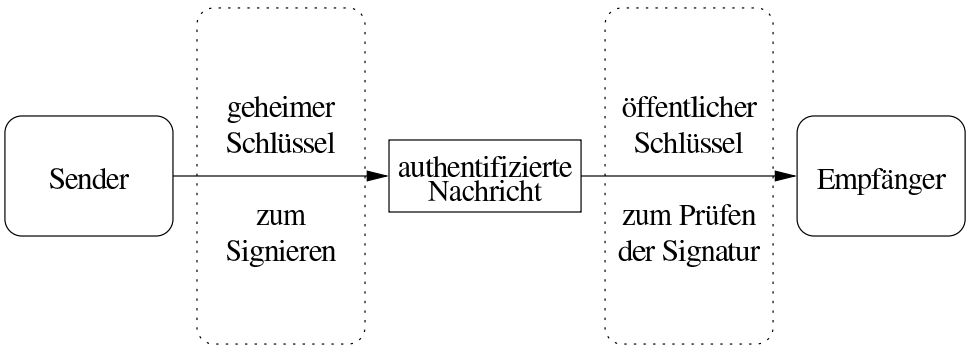


Abbildung 3: Signierung von Nachrichten

Dabei wird der Haschwert einer zu authentifizierenden Nachricht wird mit dem geheimen Schlüssel eines asymetrischen Schlüsselpaars verschlüsselt und mit der unverschlüsselten Nachricht mitgeschickt. Der Empfänger kann nun selbst den Haschwert über die Nachricht bilden und mit der Signatur, die er mit dem öffentlichen Schlüssel entschlüsseln kann, vergleichen. Bei Gleichheit ist die Nachricht authentifiziert. Asymetrische Verschlüsselung hat den Vorteil, dass sich der Sender und der Empfanger der verschlüsselten Nachricht auf kein gemeinsames Geheimnis einigen müssen. Ein gesicherter Austausch von Schlüsseln ist dadurch nicht mehr notwendig. Nur der Sender kennt den geheimen Schlüssel, während der öffentliche Schlüssel allen zur Verfügung steht. Für die Authentifizierungen von Binding Updates bei kryptographisch generierten Adressen werden keine weiteren Sicherheitsvorkehrungen wie IPSEC benötigt.

3 CAM Protokoll

Der Inhaber einer zu authentifizierenden Adresse erzeugt ein Schlüsselpaar für asymetrische Verschlüsselung. Anschließend wird der Schnittstellen Indentifikator der Heimatadresse aus einem Haschwert des öffentlichen Schlüssels und einem Modifizierer bestimmt. Beim Child-proof Authentifizierungs Protokoll beweist der mobile Knoten die Korrektheit seiner Heimatadresse, indem er zeigt, dass er Kenntnis über den geheimen Schlüssel hat. Eine detailiertere Beschreibung folgt in den nächsten Abschnitten.

3.1 Notation

Für eine vereinfachte Darstellung wird folgende Notation eingeführt:

- M: Mobiler Knoten
- C: Kommunikationspartner
- A'_m: Zustelladresse von M
- A_c: Adresse des Komunikationspartners
- (PK_m, SK_m): (öffentlich, geheim) Schlüsselpaar von M
- i: Modifizierer
- $H(x)$: Haschwert von x
- T_m: Zeitstempel von M
- $\{x\}SK_m$: x, verschlüsselt mit geheimen Schlüssel von M
- A_m: Heimatadresse von M
- R: Routing-Präfix von A_m

3.2 Home Address

Wie oben schon erwähnt wurde, ist die Heimatadresse des mobilen Knoten beim Child-proof Authentifizierungs Protokoll eine kryptographisch generierte Adresse. Die ersten 64 bit bilden den Routing-Präfix. Für den Schnittstellen-Identifikator wird ein nicht invertierbarer Haschwert aus dem öffentlichen Schlüssel und dem Modifizierer gebildet.

$A_m = R|H(PK_m, i)$

Für die Haschfunktion wird der SHA1 Algorithmus angewendet. Nur die ersten 62 bit des Haschwertes werden in den Schnittstellen-Identifikator aufgenommen. Zwei Bits werden auf null gesetzt, damit keine Verwechselungen mit anderen Schnittstellen-Identifikatoren auftreten[O'Ro03].

Bei diesem Verfahren ist es theoretisch möglich, dass eine Adresse mehrfach belegt wird. Das ist aber sehr unwahrscheinlich. Dafür müssten zwei mobile Knoten im selben Subnetz den gleichen Schlüssel verwenden oder die Haschfunktion bildet zufällig beide Schlüssel auf den selben Haschwert ab. Aus diesem Grund wurde der Modifizierer in die Haschfunktion eingebaut. Sollte eine Adresse einmal belegt sein, so ändert man einfach den Modifizierer und bekommt über den veränderten Haschwert eine neue Adresse zugewiesen. Auf diese Weise muss man kein neues Schlüsselpaar erzeugen. Der Modifizierer sollte nicht größer als ein oder zwei bits sein. Das reicht mit großer Wahrscheinlichkeit aus, um keine erneute Kollision der Adressen zu erzeugen. Ein zu großer Modifizierer würde das Protokoll schwächen. Je mehr Werte der Modifizierer annehmen kann, desto häufiger kann ein möglicher Angreifer jedes eigene Schlüsselpaar auf die Erzeugung von Kollisionen mit fremden Adressen testen.

3.3 Ablauf

Das besondere am Child-proof Authentication Protokoll ist, dass die Authentifizierung nur in eine Richtung abläuft. Der mobile Knoten sendet nur eine einzige Nachricht an den Kommunikationspartner. Dieser prüft die Nachricht und bewertet daraufhin das Binding Update als authentisch oder lehnt es ab. Man muss keine manuellen Einstellungen vornehmen. Trotzdem ist das Verfahren schnell und sicher. Wie das funktioniert sehen wir gleich.

Der mobile Knoten sendet folgende Nachricht an den Kommunikationspartner:

$$\mathbf{M \rightarrow C : A'_m, A_c, A_m, PK_m, i, T_m, \{H(A'_m, A_c, A_m, T_m)\}SK_m}$$

- C vergleicht den Zeitstempel mit seiner Uhr.
- C stellt sicher, dass $A_m = R|H(PK_m, i)$.
- C entschlüsselt $\{H(A'_m, A_c, A_m, T_m)\}SK_m$ mit PK_m.
- C hascht A'_m, A_c, A_m, T_m.
- C vergleicht beide Ergebnisse.
- Sind die Ergebnisse unterschiedlich verwirft er die Nachricht.
- Ansonsten gilt die Heimatadresse als authetifiziert.

Der Kommunikationspartner kennt eventuell den öffentlichen Schlüssel und den Modifizierer noch nicht. Aus diesem Grund werden sie mit der Nachricht mitgeschickt. Es steht dem Kommunikationspartner offen, welche Zeitunterschiede er zwischen dem Zeitstempel der Nachricht und der eigenen Uhr toleriert. Zu alte Nachrichten werden verworfen. Die Überprüfung des Haschwertes über den öffentlichen Schlüssel und den Modifizierer stellt sicher, dass die angegebene Adresse mit dem öffentlichen Schlüssel generiert wurde. Der Vergleich der Haschwertbildung über die gesendeten Parameter und der entschlüsselten Signatur zeigt bei Gleichheit, dass der mobile Knoten über den geheimen Schlüssel verfügen muss. Die Kenntnis des mobilen Knoten über den passenden geheimen Schlüssel ist der Beweis, dass die Heimatadresse authentisch ist.

3.4 Schutz

Um einen erfolgreichen Angriff durchzuführen, muss der Angreifer ein Schlüsselpaar finden, das auf dieselbe Heimatadresse gehascht wird. Da das Routing-Präfix mit in den Haschwert, der den Schnittstellen-Identifikator bildet, eingeht, muss ein Angreifer seine Schlüssel für jedes Netz auf Kollisionen testen. Das Auffinden eines solchen Schlüsselpaars ist sehr unwahrscheinlich. Würde eine Angreifer an den geheimen Schlüssel kommen, dann wäre der Schutz aufgehoben. Zeitstempel bieten einen Schutz gegen Replay Attacken. Eine Authentifizierung durch Senden alter, authentifizierter Binding Updates kann vom Kommunikationspartner durch eine kurze Gültigkeitsdauer der Nachrichten eingedämmt werden.

3.5 Attacken

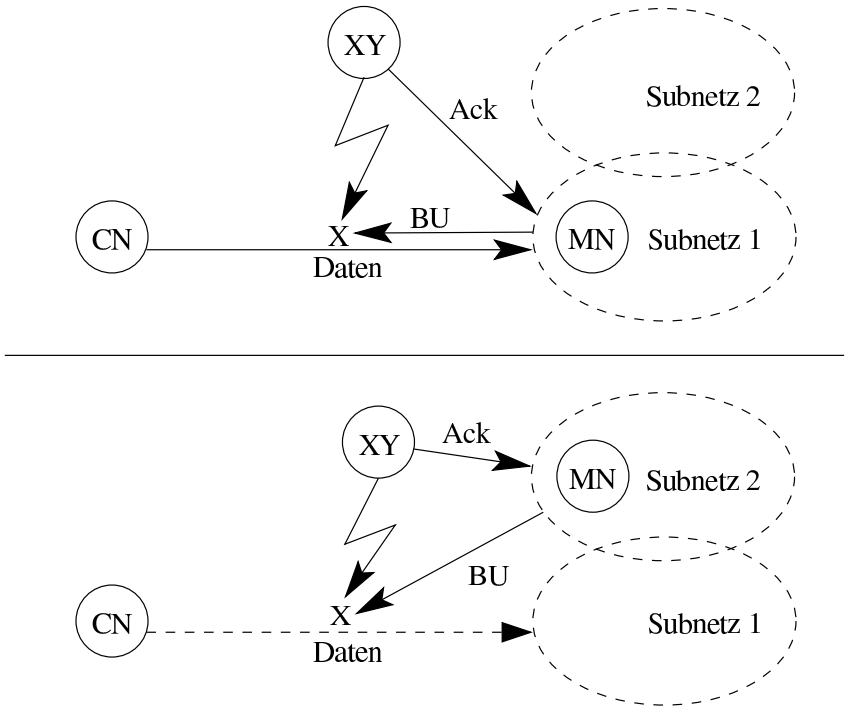


Abbildung 4: Ein möglicher Angriff auf CAM.

Eine mögliche Attacke ist es, die Zustelladresse zu manipulieren, während man die Heimatadresse regelkonform authentifizieren läßt. Die Zustelladresse wird im Child-proof Authentication Protokoll nicht authentifiziert. Eine Möglichkeit sich gegen diesen Angriff abzusichern, wäre den CoA-Test aus dem Return Routability Protokoll zu verwenden. Die Eigenschaften von CAM, dass die Authentifizierung nur in einer Richtung abläuft und dass nur eine Nachricht benötigt wird, gingen verloren. Kann man im Fremdnetz eine bestimmte Adresse beantragen, ist es im Prinzip kein Problem, die Zustelladresse auf dieselbe Art und Weise wie die Heimatadresse authentifizieren zu lassen. Es stellt sich die Frage, warum die Entwickler von CAM das nicht umgesetzt haben.

Denial of service Attacken mittels einer Unmenge an versendeten Binding Updates können nicht verhindert werden. In diesem Fall sollten ander Protokolle, die solche Attacken verhindern können, zum Einsatz kommen.

Wie Abbildung 4 zeigt, könnte ein Angreifer Binding Updates abfangen und falsche Binding Acknowledgements, die nicht authentifiziert werden, an den mobilen Knoten zurücksenden.

Ein Kommunikationspartner könnte einen Wechsel des mobilen Knoten nicht mehr feststellen und würde weiterhin Pakete an das alte Netzwerk schicken. Ein Protokoll kann allerdings keine Angriffe abwehren, wenn ein Angreifer in der Lage ist Nachrichten abzufangen.

4 Vorteile

Vorteil des Child-proof Authentifizierungs Protokolls ist, dass kein gemeinsames Geheimnis verwendet wird. Ein sicherer Austausch wird somit nicht benötigt. Des weiteren muss bei CAM nur eine einzige Nachricht versand werden. Das Netz wird dadurch weniger durch Binding Updates belastet. Positiv macht sich auch bemerkbar, dass keine sicheren Verbindungen zwischen den Teilnehmern vorausgesetzt werden. Es ist einfach zu implementieren und ohne großen Aufwand anzuwenden. Manuelle Einstellungen sind nicht erfordert. Der Einsatz bietet sich vor allem an, wenn IPSEC nicht zur Verfügung steht und dennoch eine ausreichende Sicherheit erfordert wird. Eine einzige Nachricht genügt um die Heimatadresse des mobilen Knoten zu authentifizieren.

5 Fazit

Das Child-proof Authentication Protokoll besticht durch seine Kompaktheit, die dennoch eine sehr gute Absicherung gegen Vortäuschung von Adressen bietet. Mit sehr wenig Aufwand wird eine ausreichende Sicherheit geboten. Die Einfachheit sollte das Child-proof Authentication Protokoll auch für die praktische Anwendung interessant machen.

Literatur

[Aura03] Tuomas Aura. Cryptopgraphically Generated Addresses (CGA). Internet draft, Februar 2003.

[O'Ro03] Greg O'Shea und Michael Roe. Child-proof Authentication for MIPv6 (CAM). *ACM SIGCOMM Computer Communication Review* 31(2), April 2003.

Abbildungsverzeichnis

1 Ablauf von Route Optimization 54

2 Aufbau einer kryptographisch generierten Adresse 55

3 Signierung von Nachrichten 55

4 Ein möglicher Angriff auf CAM. 58

Mobile IPv4 und NAT

Fikri Tekin

Kurzfassung

Mobile IPv4 und NAT bilden zwei wesentliche Grundbausteine in der Welt des Internets. Mobile IPv4 ist eine Erweiterung des Internetprotokolls (IP), die die Voraussetzung für eine unterbrechungsfreie Paketweitervermittlung zu portablen Geräten schafft; NAT ist eine Technik, um private Netzwerke unter Verwendung möglichst weniger IP-Adressen an das Internet anzukoppeln. Bei der Zusammenarbeit zwischen diesen beiden Konzepten kommt es jedoch zu einigen Schwierigkeiten. Ziel dieser Arbeit ist die Erläuterung der dabei entstehenden Probleme und die Vorstellung von Lösungsansätzen. Darauf aufbauend wird ein Ansatz vorgestellt, der es mobilen Endgeräten ermöglicht, Mobile IP zu unterstützen und in privaten Netzwerken, die mit NAT-Geräten vom Internet getrennt werden, zu funktionieren.

1 Einleitung

Im heutigen Internet ist Mobilität ein wichtiger Aspekt geworden. Dies ist zurückzuführen auf die weite Verbreitung und ständig wachsende Anzahl von portablen Computern wie z.B. Notebooks oder PDAs. Die Anforderungen, die die Mobilität der Endgeräte mit sich bringt, wurden bei der Entwicklung des IPv4-Protokolls allerdings nicht berücksichtigt. Dieses bedingt nämlich, dass mobile Geräte bei jedem Anschluss an ein fremdes Netzwerk auch dessen Netzwerk-Präfix der IP-Adresse annehmen, um die topologische Korrektheit der IP-Adressen zu gewährleisten. Denn sobald man sein „Heimatnetz“, in dem das mobile Gerät normalerweise arbeitet und für das es konfiguriert wurde, verlässt und sich an einem anderen Ort mit dem Internet verbindet, würde es zu einer Unterbrechung der aktuellen Verbindung kommen. Folglich müsste dem mobilen Endgerät eine neue IP-Adresse zugewiesen werden, die im fremden Netzwerk gültig ist. Um sicherzustellen, dass alle von einem beliebigen Absender an die alte IP-Adresse gesendeten Pakete den neuen Zielort erreichen, wäre es nun nötig, jeden einzelnen Router (weltweit) über diese neue IP-Adresse in Kenntnis zu setzen. Dies ist jedoch nicht mit vertretbarem Aufwand an Datenverkehr und Zeit möglich. So ist es mit IP nicht sinnvoll möglich, Mobilität von Endgeräten zu implementieren. Deshalb wurde mit Mobile IP eine Erweiterung der bestehenden Mechanismen von IPv4 vorgenommen.

Eng verbunden mit dem IP-Protokoll ist der Begriff des NAT, das in der Welt des Internets ebenfalls eine bedeutende Rolle spielt. NAT *(Network Address Translation)* ist in Computernetzwerken ein Verfahren, bei dem private IP-Adressen auf öffentliche IP-Adressen abgebildet werden. Es wird aus verschiedenen Gründen verwendet. Hauptsächlich ist NAT notwendig, weil öffentliche IP-Adressen immer knapper werden und man deshalb private IP-Adressen einsetzen muss. Zum Anderen kann es der Datensicherheit dienen, weil die interne Struktur eines Netzwerkes nach außen hin verborgen wird.

Im weiteren Verlauf der Arbeit werden Mobile IPv4 und NAT näher erläutert und die dazugehörigen Komponenten und Funktionsweisen beschrieben. Anschließend werden Probleme, die beim Zusammenspiel zwischen den beiden Konzepten entstehen, beschrieben und grundlegende Lösungsansätze vorgestellt.

2 Mobile IPv4

Das Routing im Internet basiert auf der Zieladresse, das im Paketkopf des jeweiligen IP-Pakets steht. Die Zieladresse deutet nicht nur auf den Empfänger hin, sondern auch auf sein physikalisches Subnetz. IP-Pakete werden im Internet von Router zu Router weitergeleitet, wobei jeder Router die Zieladresse eingehender Pakete betrachtet und dann die Pakete anhand einer Abbildungsfunktion von Zieladresse auf einen Ausgang des Routers weiterleitet. Damit die Tabellen, welche die Abbildung von Zieladresse auf einen Ausgang speichern, nicht zu groß werden, speichern Router nur die Präfixe und leiten mit Hilfe dieser das Paket weiter an das zugehörige Subnetz. Erst im Heimatnetz wird die IP-Adresse dem richtigen Rechner zugewiesen.
Befindet sich nun ein mobiles Gerät in einem fremden Netz, so können an ihn adressierte Pakete nicht ausgeliefert werden, da sie aufgrund des Netzwerk-Präfixes der Zieladresse immer in Richtung des Heimatnetzes geroutet werden. Die Kommunikation mit anderen Rechnern ist somit für einen mobiles Gerät nicht möglich, wenn er ein fremdes Netz besucht. Aus diesen Gründen gibt es eine Reihe von Anforderungen, auf die bei der Entwicklung von Mobile IP [Schi03] [Perk98] geachtet wurde und die im nachfolgenden Abschnitt beschrieben werden.

2.1 Anforderungen

Folgende Anforderungen wurden an Mobile IP beim Entwurf gestellt:

- Kompatibilität:
  Es sollen die gleichen Protokolle unterstützt werden wie von IP, so dass keine Änderungen an bisherigen Endsystemen und Routern nötig werden. Ferner sollen mobile Endsysteme mit festen und umgekehrt kommunizieren können.
- Transparenz:
  Mobile Endsysteme können ihre IP-Adresse behalten. Die Mobilität ist für andere Protokolle nicht sichtbar, d.h. die Fortsetzung der Kommunikation ist auch nach einer physikalischen Unterbrechung durchführbar. Zudem kann der Anschlusspunkt an das Netz gewechselt werden.
- Effizienz:
  Da das mobile Endsystem eventuell nur über eine schmalbandige Anbindung zu erreichen ist, sollte der Overhead so gering wie möglich sein. Weiterhin soll internetweit eine sehr große Anzahl mobiler Endsysteme unterstützt werden können.
- Sicherheit:
  Eine Authentisierung aller Registrierungsnachrichten, d.h. Nachrichten zur Integration des mobilen Endsystems ins Netz, ist erforderlich. Das IP-Protokoll kann in seiner jetzigen Version (IPv4) nur sicherstellen, dass die Zieladresse richtig ist.

2.2 Komponenten

Einige Begriffe tauchen im Zusammenhang mit Mobile IP immer wieder auf, die zugehörigen Mechanismen werden nun geschildert:

- Mobiler Knoten (Mobile Node, MN):
  Ein MN ist ein Host oder ein Router, der seinen Zugangspunkt zum Internet wechseln kann. Dabei behält er stets seine eigene IP-Adresse und führt seine Kommunikation mit anderen Partnern im Internet ununterbrochen weiter.

- Kommunikationspartner (Correspondent Node, CN):
  Zumindest ein Partner ist für eine Kommunikation notwendig. Ein CN stellt diesen Partner dar und kann sowohl mobil als auch stationär sein.

- Heimatagent (Home Agent, HA):
  Als ein HA wird ein Router im Heimatnetz vom MN bezeichnet, der an den MN adressierte Datenpakete abfängt und an dessen gegenwärtige Zustelladresse weiterschickt.

- Fremdagent (Foreign Agent, FA):
  Unter dem FA versteht man einen Router des Fremdnetzes, in dem sich der MN gerade aufhält. Er stellt einen Tunnelendpunkt dar und signalisiert die Erkennung von Mobilität.

- Zustelladresse (Care-of Address, COA):
  Die COA kennzeichnet die Zustelladresse, unter der der MN für den HA im Internet erreichbar ist. Falls der MN vorübergehend noch eine vom FA verschiedene IP-Adresse erhalten hat, spricht man von co-located COA.

2.3 Funktionsweise

In Abbildung 1 wird der Ablauf des Datenverkehrs zwischen einem CN und einem MN dargestellt. Aufgrund der Transparenz von Mobile IP muss der sendende CN den aktuellen Aufenthaltsort des empfangenden MN nicht kennen und sendet das Datenpaket an die ihm bekannte IP-Adresse des MN (Schritt 1). Das Internet, für das die Mobilität ebenfalls transparent ist, leitet das Datenpaket wie gewohnt zum Router weiter, bis der HA das Paket erhält und es abfängt. Falls sich derzeit der MN im Heimatnetz aufhält, leitet der HA das Paket dorthin weiter. Wenn der MN nicht im Heimatnetz ist, wird das Datenpaket gekapselt und in einem Tunnel an die COA des MN weitergeleitet (Schritt 2). Diese Kapselung geschieht dadurch, dass über den ursprünglichen Paketkopf einfach noch ein zweiter gesetzt wird, der den HA als Quelladresse und die COA als Zieladresse enthält. Die Mechanismen eines Tunnels und der Kapselung werden in Abschnitt 2.3.3 näher erläutert. Der Tunnelendpunkt und der FA liegen im Router für das aktuelle Fremdnetz des MN. Der FA entkapselt das eingehende Datenpaket, entfernt also den zusätzlichen Paketkopf, und leitet das Originalpaket, mit CN als Quelladresse und MN als Zieladresse, an den MN weiter (Schritt 3).
Der MN sendet ein Paket wie gewöhnlich mit seiner eigenen festen IP-Adresse als Absender und der Adresse des CN als Ziel (Schritt 4). Der Router mit dem FA arbeitet hierbei als Standard-Router für den MN und leitet das Paket wiederum in das Internet weiter.

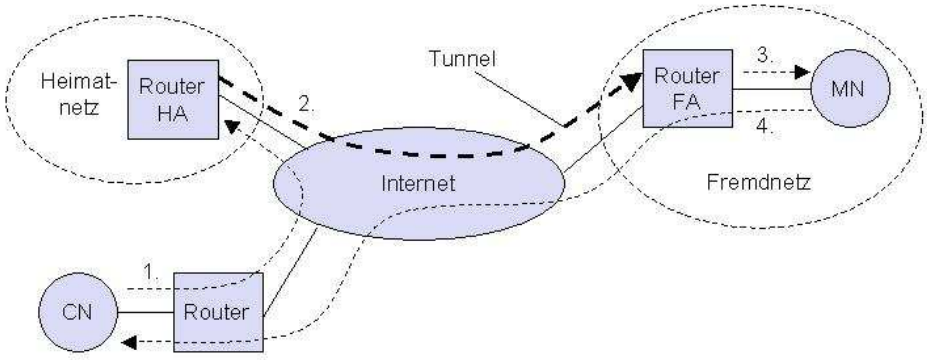


Abbildung 1: Paketweiterleitung zum/vom MN

2.3.1 Auffinden eines Agenten

Sobald ein MN sich in ein neues Netz hineinbewegt, besteht das erste Problem darin, einen FA zu finden, falls die Datenpakete hierüber weitergeleitet werden. Daher müssen Heimat- und Fremdagenten ihre Verfügbarkeit für einen MN sichtbar machen, damit dieser darüber informiert werden kann, ob er sich in seinem Heimatnetz oder in einem Fremdnetz befindet. Auch müssen etwaige Netzwechsel erkannt werden und der MN über seine derzeitige COA informiert werden.
Dazu gibt es zum einen die Methode des *Agent Advertisement*. Hierbei senden Heimat- und Fremdagenten in regelmäßigen Abständen Pakete in das Subnetz, die sogenannten *Agent Advertisement Messages*, die ihre Anwesenheit signalisieren und ihre Dienste zur Verfügung stellen. Ein MN in einem Subnetz empfängt diese *Agent Advertisements* entweder vom Heimatagenten oder vom Fremdagenten. Anhand von diesen Nachrichten ist ein MN somit in der Lage, seinen aktuellen Aufenthaltsort festzustellen. Ist ein MN an einem FA registriert, so muss er weiter auf die *Advertisements* horchen, um erkennen zu können, ob der aktuelle FA noch erreichbar ist. Außerdem muss er dem HA in seinem Heimatnetz seine COA mitteilen, unter der er im fremden Netz zu erreichen ist. Die COA ist meist die IP-Adresse desjenigen FA, der die eingehenden Pakete an den MN weiterreicht.
Werden keine Ankündigungen von Agenten vom MN empfangen bzw. gehen diese Ankündigungen zu selten ein, so muss der MN aktiv Agenten suchen. Dies geschieht über *Agent Solicitation Messages*, welche ein MN ins Netz aussendet, wenn zur Zeit keine COA verfügbar ist. Der MN erhält daraufhin die *Agent Advertisements* aller Agenten und kann dabei ermitteln, ob er sich in seinem Heimatnetz oder in einem Fremdnetz befindet.

Nach dem Auffinden eines geeigneten Agenten kann der MN eine geeignete COA erhalten, entweder ist dies eine COA für den FA oder eine co-located COA für den MN. Im Falle eines Aufenthalts in einem Fremdnetz besteht der nächste Schritt für den MN in der Registrierung beim HA, die im folgenden Abschnitt beschrieben wird.

2.3.2 Registrierung

Unter Registrierung soll hier der Prozess verstanden werden, durch den der MN dem HA seinen momentanen Aufenthaltsort bekannt gibt, indem er seine aktuelle COA mitteilt. Dadurch können Pakete vom HA korrekt an den MN weitergeleitet werden. Für den Registrierungsprozess werden zwei Nachrichten zwischen dem MN und dem HA ausgetauscht, nämlich die Registrierungsanfrage und die Registrierungsantwort. Diese Nachrichten werden entweder direkt zwischen MN und HA versendet oder laufen über einen FA. Welcher Weg gewählt wird, hängt davon ab, ob die COA des MN beim FA ist oder es sich um eine co-loctaed COA handelt. Ersterer wird in Abbildung 2 veranschaulicht. Entdeckt ein MN auf der Suche nach einem Agenten einen FA, so sendet er eine Registrierungsanfrage. Der FA leitet diese nach Prüfung der Registrierungsbedingungen, z.B. der Gültigkeitsdauer der Registrierung, an den HA weiter. Nach Prüfung der

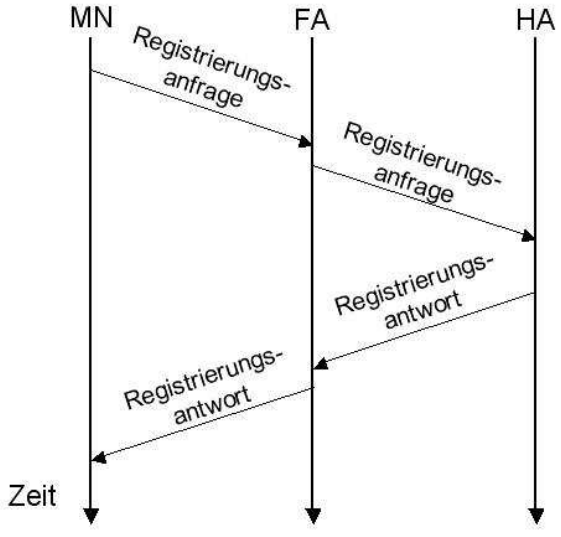


Abbildung 2: Registrierung über den FA

Authentizität des Pakets sendet dieser dann eine Registrierungsantwort an den FA zurück, der die Nachricht an den MN überliefert. Verfügt der MN allerdings noch über eine co-located COA, wirkt der FA im Registrierungsvorgang nicht mit, so dass Registrierungsanfrage und Registrierungsantwort direkt zwischen dem MN und dem HA erfolgen. Da die Registrierung automatisch nach Ablauf der Gültigkeitsdauer erlischt, sollte sich ein MN regelmäßig vor dem Ablauf erneut registrieren.
Für die Registrierungsnachrichten werden UDP-Pakete eingesetzt, deren Anwendung in Abschnitt 4.1 noch genauer erklärt wird. Die IP-Quelladresse der Pakete wird auf die Adresse des MN gesetzt, die IP-Zieladresse ist je nach Lage der COA die des FA oder HA. Die für die Registrierung notwendigen Angaben stellt der Nutzdatenteil des UDP-Pakets dar.

2.3.3 Tunneln von Paketen

Im Folgenden werden die Mechanismen beschrieben, die zur Weiterleitung von Paketen zwischen dem HA und der COA eingesetzt werden, also der zweite Schritt aus Abbildung 1. Ein *Tunnel* stellt logisch betrachtet einen Kanal für Datenpakete zwischen einem Tunneleingang und einem Tunnelausgang dar. Alle Pakete, die in den Tunnel hineingeschickt wurden, sollten unverändert durch den Tunnel bis zu dessen Endpunkt weitergeleitet werden und den Tunnel genau so wieder verlassen, wie sie am Anfangspunkt abgeschickt wurden. Das Tunneln, also das Senden eines Pakets durch einen Tunnel, wird mit Hilfe der Kapselung von Paketen durchgeführt. Dieser Vorgang wird bei Mobile IP eingesetzt, um die Pakete, die ursprünglich an den MN adressiert sind, vom HA zum FA zu schicken. Der FA kann hinterher das originale Paket an den MN ausliefern.

Mit Kapselung ist der Prozess gemeint, dass ein Paket, bestehend aus Nutzdatenteil und Paketkopf, zum Nutzdatenteil eines neuen Pakets wird. Die Kapselung findet am Tunneleingang statt. Dabei nimmt der HA das Originalpaket mit dem MN als Ziel, fügt es als Datenteil zu einem neuen Paket hinzu und setzt die Zieladresse des neuen Pakets so, dass es zur COA weitergeleitet wird, wobei der neue Paketkopf auch als äußerer Paketkopf bezeichnet wird. Weiterhin gibt es einen inneren Paketkopf, der mit dem Originalkopf identisch sein kann, wie dies bei der IP-in-IP-Kapselung (siehe Abbildung 3) nach RFC 2003 [Perk96a] der Fall ist. Er enthält folglich insbesondere den originalen Absender CN und den Empfänger MN des Pakets. Die umgekehrte Operation, welche ein Paket aus dem Nutzdatenteil eines anderen Pakets wieder entfernt, wird im übrigen Entkapselung genannt.

Abbildung 3: IP-in-IP-Kapselung

Da jedoch bei der IP-in-IP-Kapselung mehrere Informationsfelder in den Paketköpfen redundant sind, wurde die minimale Kapselung nach RFC 2004 [Perk96b] als optionale Kapselungsmethode für Mobile IP definiert. Diese Variante unterscheidet sich von der oben beschriebenen dadurch, dass nicht ein vollständiger Paketkopf um das zu tunnelnde Paket gesetzt wird, sondern der ursprüngliche Paketkopf verändert wird. Die benötigten Daten werden im speziellen und wesentlich kleineren Paketfeld *Minimal Forwarding Header*

gespeichert. Somit wird der Overhead bei der Übertragung des gekapselten Pakets minimiert. Während IP-in-IP-Kapselung und die minimale Kapselung nur für IP funktionieren, unterstützt *Generic Routing Encapsulation* (GRE) (siehe Abbildung 4) die Kapselung von Paketen eines Protokolls in den Nutzdatenteil von Paketen eines anderen Protokolls, wie sie in RFC 1701 [HLFT94] spezifiziert wurde. Entscheidend dabei ist, dass dem Originalpaket eines Protokolls zusätzlich ein neuer GRE-Kopf vorangestellt wird. GRE bietet unter anderem den Vorteil, die Anzahl von Kapselungen desselben Originalpakets einzugrenzen. Sobald also ein Paket an einem neuen Tunneleingang zum Kapseln ansteht, wird zunächst überprüft, ob es schon gekapselt wurde oder noch eine weitere Kapselung erlaubt ist.

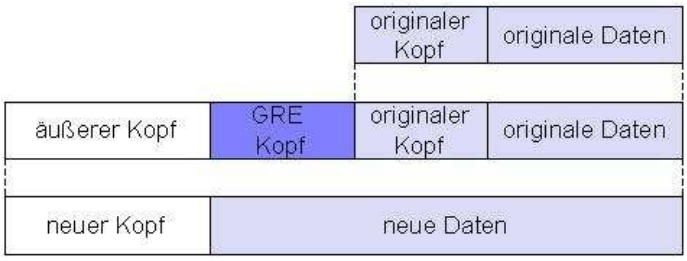


Abbildung 4: Generic Routing Encapsulation

3 NAT

Befasst man sich mit dem geteilten Internetzugang aus einem privaten Netzwerk mit mehreren Rechnern, so stößt man nach kurzer Zeit auf den Begriff NAT (*Network Address Translation*) [EgFr94] [SrHo99], welches soviel bedeutet wie „Übersetzung von Netzwerkadressen“. Entstanden ist diese Technik 1993. Bereits damals war absehbar, dass der durch IPv4 verfügbare Adresseraum in Zukunft nicht mehr ausreichen wird, um alle Rechner an das Internet anschließen zu können. NAT wurde damals als eine Übergangslösung bis zur Entwicklung eines neuen Protokolls (IPv6) bezeichnet. Es hat sich jedoch bis heute zu einem wichtigen Standard entwickelt. Fast jeder in Kleinbetrieben und im Heimbereich verfügbare Router beherrscht diese Funktionalität. Auch Geräte für den „gehobenen“ Bedarf bieten dies als Option an.

3.1 Funktionsprinzip

Voraussetzung für die Nutzung der NAT-Technologie war die Unterteilung des Adressraums von Mobile IPv4 in zwei Teile: den wiederverwendbaren (privaten) und global eindeutigen (öffentlichen) Bereich. Adressen aus dem privaten Adressraum können somit jederzeit und von jedem benutzt werden, um ein größeres Netzwerk aufzubauen und dieses mittels eines NAT-Geräts mit dem Internet zu verbinden.

Um nun aus einem lokalen Netzwerk mit mehreren Rechnern über einen zentralen Router oder einen Gateway auf das Internet zuzugreifen, ist es nötig, die unterschiedlichen IP-Adressen des privaten Netzwerkes in öffentlich registrierte IP-Adressen umzuwandeln. Ebenso muss die Zieladresse des Antwortpakets aus dem Internet wieder in eine nur im lokalen Netz gültige zurückübersetzt werden. Dabei muss der Router sich aber auch merken können, von welchem Host eine Anfrage kam, um genau diesem auch die dazugehörige Antwort mitteilen zu können. Der Router muss hierzu alle IP-Adressen des internen Netzes kennen und über den Provider öffentliche IP-Adressen beziehen. Die IP-Adressen interner Rechner, die eine Kommunikation mit Zielen im Internet aufbauen müssen, erhalten in dem Router, der zwischen dem ISP

(*Internet Service Provider*) und dem privaten Netzwerk steht, einen Tabelleneintrag. Da die Anforderungen von Webseiten sowie der Datentransport über den Router läuft, kann dieser für eingehende Daten den Zielrechner bestimmen und über die Tabelle die IP des Rechners in dessen lokale IP-Adresse zurückübersetzen. Somit sind diese Rechner durch diese Eins-zu-Eins-Zuordnung nicht nur in der Lage, eine Verbindung zu Zielen im Internet aufzubauen, sondern sie sind auch aus dem Internet erreichbar. Die Adressumsetzung hat den Vorteil, daß für Rechner, die innerhalb eines privaten Netzes miteinander kommunizieren müssen, keine öffentlichen IP-Adressen benötigt werden.
Das Ziel von NAT besteht zum einen darin, sowohl die interne Struktur des Netzwerkes als auch die internen Host-Adressen nach außen hin zu verbergen, indem die IP-Adressen auf öffentliche abgebildet werden. Des weiteren ist eine Kontrolle der Daten von Extern nach Intern möglich. Hierzu werden zwar nach außen alle Datenpakete durchgelassen, nach intern allerdings nur mit expliziter Freigabe.

3.2 IP-Masquerading

Eine besondere Form von NAT ist IP-Masquerading, was auch als PAT (*Port and Address Translation*), NAPT (*Network Address and Port Translation*) oder 1-to-n-NAT bezeichnet wird. Hier werden sämtliche Adressen eines privaten Netzwerkes auf eine einzelne öffentliche (dynamische) IP-Adresse gemappt bzw. maskiert[1]. Dies geschieht dadurch, dass bei einer existierenden Verbindung zusätzlich zu den Adressen auch die Portnummern ausgetauscht werden. Auf diese Weise benötigt ein gesamtes privates Netz nur eine einzige registrierte öffentliche IP-Adresse, welche dann gegenüber anderen Servern und Clients im Internet als Quelladresse erscheint. Dieser Zusammenhang wird in Abbildung 5 veranschaulicht. Nachteil dieser Lösung ist der, dass die Rechner im privaten Netzwerk aus dem Internet nicht angewählt werden können. Diese Methode eignet sich daher besonders dazu, zwei und mehr Rechner eines privaten Anschlusses über eine gemeinsame IP-Adresse per DFÜ-Netzwerk an das Internet zu koppeln.
Schickt beispielsweise ein Rechner aus dem Internet ein Datenpaket an das Notebook im Netzwerk, so sieht der externe Rechner nur die Adresse des Routers. Dieser wiederum ersetzt anhand seiner internen Tabelle die Zieladresse des Pakets durch die IP-Adresse des Notebooks, so dass das Paket am richtigen Rechner ankommt. Das heißt also, anhand der Portnummer des Dienstes wird die Adresse des richtigen internen Rechners gefunden, wobei im Internet nur eine einzige IP benutzt wird.

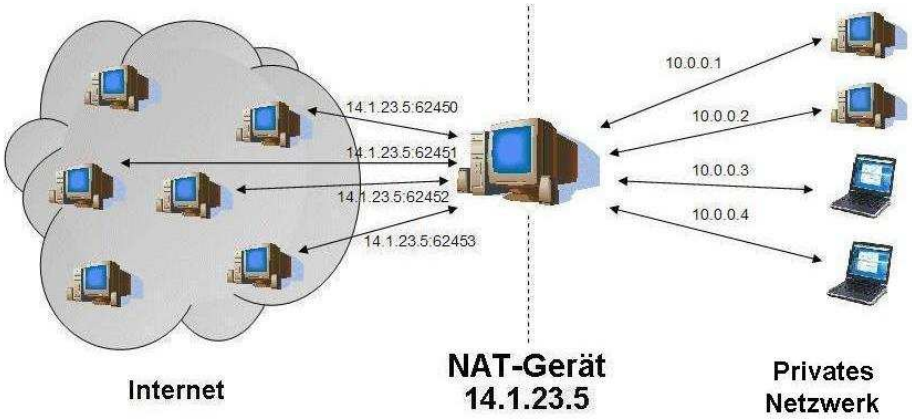


Abbildung 5: NAPT

[1]Maskieren und Masquerading haben dieselbe Bedeutung

4 Zusammenwirken von Mobile IPv4 und NAT

Wie schon in Kapitel 2 beschrieben, wurde Mobile IPv4 entwickelt, um die Adressierung eines mobilen Hosts über seine feste Heimatadresse zu ermöglichen, und zwar unabhängig vom Subnetz, in dem er sich gerade aufhält. Dadurch ist ein MN in der Lage, sein Subnetz zu wechseln, während dies für höhere Schichten transparent ist und die eventuell bestehende Verbindung zwischen dem MN und seinem zugehörigen CN nicht unterbrochen wird. Voraussetzung für das Mobile-IP-Protokoll ist, dass sowohl der MN als auch der FA jeweils durch eine weltweit erreichbare IP-Adresse eindeutig identifizierbar sind. Ferner sei nochmals erwähnt, dass Mobile IP darauf basiert, Daten vom Heimatnetz aus an den MN bzw. den FA mit Hilfe der IP-in-IP-Kapselung zu versenden.
Geht nun ein MN in ein Fremdnetz über, das vom Internet durch ein NAT-Gerät getrennt wird, wird dem MN eine IP-Adresse aus dem privaten Netzwerk zugewiesen. Um über das Internet kommunizieren zu können, wird diese neue Adresse des MN in einem Tabelleneintrag vom NAT-Gerät gespeichert und einer öffentlich gültigen Adresse zugeordnet. Dadurch wird sichergestellt, dass einkommende Antwortpakete die richtige IP-Adresse im lokalen Netzwerk als Ziel haben. Erst danach können Datenpakete des MN aus dem privaten Netzwerk durch das NAT-Gerät ins Internet weitergeleitet werden.
Um für den erwünschten Datentransfer vom HA einen direkten Tunnel zwischen diesem und dem zugehörigen FA aufzubauen, sendet der MN an seinen HA eine Registrierungsanfrage. Diese Anfrage enthält als Quelladresse die neu erlangte private IP-Adresse. Aufgrund der Tatsache, dass die Mobilitätsfunktion auf routing-fähigen öffentlichen IP-Adressen basiert, kann somit ein MN mit einer privaten IP-Adresse die Dienste von Mobile IP nicht nutzen, da die private IP-Adresse „hinter“ einem NAT-Gerät im Internet nicht routing-fähig ist. Daher kann der HA keine Registrierungsantwort an den MN senden. Folglich ist es nicht möglich, für den Datenaustausch zwischen dem HA und dem FA einen Tunnel aufzubauen, um die Pakete vom HA zum MN weiterzuleiten. Es kommt also wegen der Unerreichbarkeit des MN keine Kommunikation über das Internet zustande. Abbildung 6 veranschaulicht das Problem.

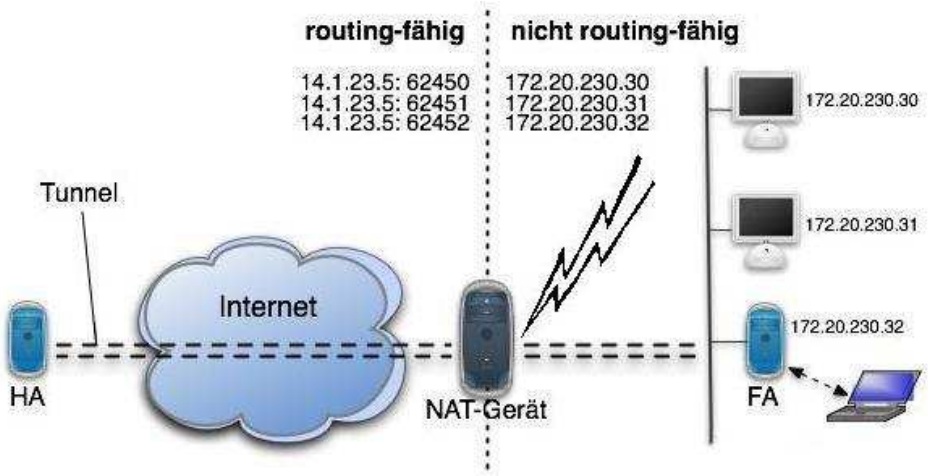


Abbildung 6: Problem der Routing-Fähigkeit

Ein weiterer Grund für die Inkompatibilität zwischen Mobile IPv4 und NAT ist, dass das NAT-Gerät nicht in der Lage ist, eingehende mittels IP-in-IP-Kapselung gekapselte Datenpakete der korrekten Zieladresse bzw. dem MN weiterzuleiten. Das NAT-Gerät führt die Adressenumsetzung anhand der IP-Adressen und der Portnummern durch. Nach der Kapselung der Pakete, kann das NAT-Gerät diese IP-Adressen und Portnummern nicht mehr finden und letztlich nicht zurückersetzen. Der Grund liegt darin, dass die IP-in-IP-Kapselung nicht genügend Informationen enthält, um eine eindeutige Umsetzung der

gemeinsam benutzten öffentlichen Adresse(n) in die einzelnen Zustelladressen eines MN oder FA zu realisieren.
Ferner gilt es den Fall in Betracht zu ziehen, dass zwei oder mehr MNs beim gleichen HA registriert sind und sich zudem in jeweils verschiedenen privaten Netzwerken befinden. Handelt es sich bei den COAs der MNs nicht um co-located COAs, benutzen sie in den Fremdnetzen die jeweilige IP-Adresse des FA als ihre COA. Dabei ist es möglich, dass die IP-Adressen der FAs miteinander identisch und in deren privaten Netzwerken dennoch weiterhin gültig sind. Daher kommt es bei diesem Sachverhalt ebenfalls zu Komplikationen bei der Paketweiterleitung. Denn Datenpakete, die an diese MNs adressiert sind und gemeinsam die IP-Adresse des HA als Quelladresse haben, können vom NAT-Gerät nicht problemlos an die richtigen FAs in den privaten Netzwerken überliefert werden.

Um das Problem der Inkompatibilität zwischen Mobile IPv4 und NAT zu lösen, wird also zum einem eine Alternative zum Mechanismus des Tunnelns von Paketen benötigt, die dem NAT-Gerät die Mittel für eine eindeutig bestimmte Adressenübersetzung bereitstellt, und zum anderen ein dazu kompatibler Registrierungsmechanismus [LeVa03].

4.1 UDP-Tunnelling

Als Lösung für die oben genannten Probleme wird UDP-Tunnelling nach RFC 768 [Post80] vorgestellt. UDP (*User Datagram Protocol*) ist ein einfaches Transportprotokoll, das wie TCP ebenfalls auf IP aufsetzt, aber deutlich weniger Overhead erzeugt. Es bietet einen ungesicherten, verbindungslosen Dienst, dessen Hauptfunktion der Austausch von Nachrichten ist. Da kein Mechanismus zur Bestätigung empfangener Pakete vorgesehen ist, ist es im Gegensatz zu TCP möglich, dass Pakete nicht in der Reihenfolge des Abschickens eintreffen bzw. verlorene Pakete nicht bemerkt werden. Der Sinn von UDP-Tunnelling ist, dass ein MN mit einer privaten IP-Adresse für seine co-located COA oder ein FA mit einer privaten IP-Adresse für die gewöhnlich benutzte COA nun in der Lage ist, einen Tunnel zur Paketweiterleitung aufzubauen und somit Datenpakete mit dem HA auszutauschen. Dieser Tunnel wird auch UDP-Tunnel genannt, die Datenpakete, die über den UDP-Tunnel versendet werden, dementsprechend UDP-Pakete. Im Folgenden soll das Funktionsprinzip des Tunnelns von UDP-Paketen und der dafür nötige Aufbau eines UDP-Tunnels durch eine Registrierung des MN erläutert werden.

Der Unterschied zum gewöhnlichen Tunneln von Paketen besteht bei UDP-Tunnelling in der Erweiterung der Registrierungsnachrichten um eine bestimmte zusätzliche Information. Diese Information spiegelt die Bekanntgabe in den Registrierungsnachrichten wider, dass der jeweilige MN, FA oder HA, von dem jeweils die Nachricht stammt, imstande ist, UDP-Tunnelling zu unterstützen. Denn erst dadurch kann ein UDP-Tunnel aufgebaut werden, der der Weiterleitung von Paketen zwischen dem MN und dem HA dient. So zeigt in einer Registrierungsanfrage der MN dem HA mit Hilfe der Erweiterung an, dass er im Falle eines Aufenthaltes „hinter“ einem NAT-Gerät UDP-Pakete sowohl versenden als auch empfangen kann (siehe Abbildung 7). Diese Registrierungsanfrage wird durch das NAT-Gerät weitergeleitet. Nach Empfang der Nach-

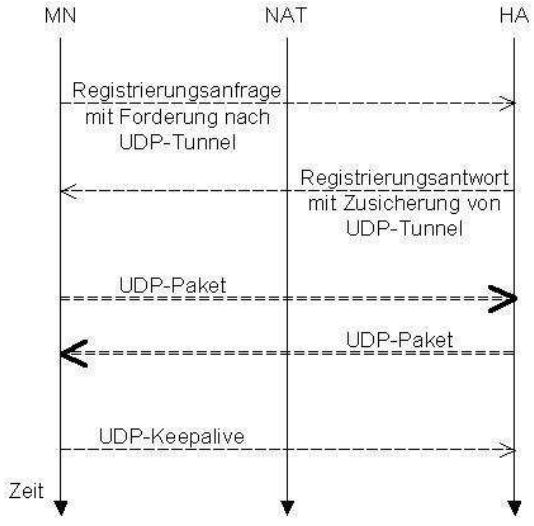


Abbildung 7: Ablauf der UDP-Nachrichten

richt erkennt der HA die Unterstützung von UDP-Tunnelling seitens des MN und sendet anschließend eine Registrierungsantwort zurück, in der er ebenfalls anzeigt, dass er in der Lage ist, UDP-Pakete zu tunneln. Dadurch wird zwischen der festen Heimatadresse und dem FA ein UDP-Tunnel zur Paketweiterleitung aufgebaut. Sollte der HA jedoch UDP-Tunnelling nicht unterstützen können, wird die Registrierungsanfrage abgelehnt, und es kann kein UDP-Tunnel aufgebaut werden. Nach dem Aufbau eines UDP-Tunnels kann nun das Tunneln zwischen MN und HA stattfinden, das mit Hilfe der im nächsten Abschnitt geschilderten IP-in-UDP-Kapselung geschieht. Des weiteren gibt es bei UDP-Tunnelling die Keepalive-Nachricht, auf die aber später noch eingegangen wird.

4.1.1 IP-in-UDP-Kapselung

Die Kapselung bei UDP-Tunnelling nennt man IP-in-UDP-Kapselung. Darüber hinaus kann der MN auch eine besondere UDP-Tunnelling unterstützende Kapselungsart anfordern, indem er das in seiner Registrierungsanfrage bekannt gibt. Hierfür ergeben sich drei Kapselunsgarten:

1. IP in UDP
2. Minimal Encapsulation in UDP
3. GRE in UDP

Das Kapseln von UDP-Paketen erfolgt nach dem gleichen Schema wie die IP-in-IP-Kapselung bei Mobile IP. Der einzige Unterschied ist, dass zwischen dem inneren und äußeren IP-Paketkopf ein UDP-Paketkopf und ein *MIP Tunnel Data Message Header* eingefügt werden, wie dies in Abbildung 8 dargestellt wird. Dieser hinzugefügte UDP-Paketkopf ermöglicht die Paketweiterleitung an ein MN durch ein NAT-Gerät und gewährleistet Kompatibilität mit NAT. Die Zieladressen der Datenpakete werden wie gewohnt übersetzt. Die minimale Kapselung und GRE laufen ebenso analog zu den Kapselungsweisen ohne UDP-Tunnelling ab.
Die im äußeren IP-Paketkopf befindlichen Informationen über die Quell- und Zielportnummern kennzeichnen die Tunnelendpunkte. Mit der Quell- und Zieladresse im inneren IP-Paketkopf werden der ursprüngliche Sender und Empfänger des Datenpakets bezeichnet. Während dem ganzen Tunnelvorgang werden im inneren Paketbereich keine Veränderungen vorgenommen. Vor der Entkapselung des Datenpakets muss allerdings auf jeden Fall überprüft werden, ob der äußere IP-Paketkopf und die UDP-Portnummern mit den für den Tunnel festgelegten Werten übereinstimmen. Der Grund für diese Bedingung ist, dass die Quell- und Zielportnummer eines vom MN verschickten UDP-Pakets jeweils identisch mit denen sein müssen, die auch bei der Registrierungsanfrage angegeben wurden. Die Quellportnummer wird solange nicht verändert, bis der MN sich erneut registriert.

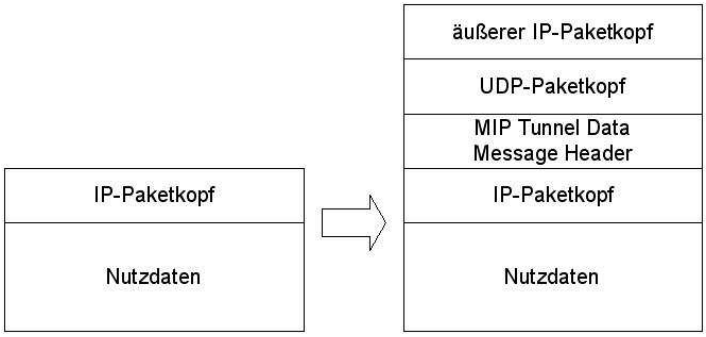


Abbildung 8: IP-in-UDP-Kapselung

4.1.2 Gültigkeit des Tunnels

Bleibt der Tunnel zwischen dem MN und HA eine bestimmte Mindestzeit lang unbenutzt, in der keine Datenpakete versendet werden, kann die Verbindung vom NAT-Gerät für beendet erklärt werden. Aus diesem Grund ist es sehr sinnvoll und von Vorteil, Nachrichten einzuführen, mit denen mitgeteilt werden kann, dass der Tunnel für den Datenverkehr noch gebraucht wird und deswegen weiterhin bestehen soll. Diese Nachrichten sind die sogenannten *Keepalives*. Sie erhalten die Gültigkeit eines Tunnels aufrecht. Daher wäre es angebracht, diese „Gültigkeitsnachrichten" öfter als Unterbrechungssignale wie Timeouts auszusenden, bevor das NAT-Gerät die Verbindung unterbricht. Sonst müsste nämlich durch eine erneute Registrierung wieder ein Tunnel aufgebaut werden. Aufgrund der Variation der Unterbrechungsintervalle ist die Möglichkeit gegeben, vor dem Aufbau des Tunnels in der Registrierungsantwort den Wert für den Zeitabstand zwischen den regelmäßig zu verschickenden *Keepalives* festzulegen, um dem vorgegebenen Standardwert entgegenzuwirken. In jeder Verbindung der festen IP-Adresse des MN mit seiner aktuellen COA gilt es für den FA, den erforderlichen Gebrauch des Tunnels mitzuteilen, falls der HA im vergangenen Intervall kein Paket versendet hat.
Das Tunneln von Paketen geschieht über den Quellport, an dem auch der Austausch der Registrierungsnachrichten durchgenommen wurde. Der MN bzw. FA gibt seine Gültigkeitsnachricht gleich nach dem Senden seiner Registrierungsanfrage von sich. Aber das muss zwangsweise über den Quellport abgeschickt werden, über den auch die Registrierung und der Datentransfer erfolgte.

4.2 Konflikte beim Zusammenspiel über UDP-Tunneling

Trotz UDP-Tunnelling können beim Zusammenspiel von Mobile IPv4 und NAT weitere kritische Punkte auftreten, die im Folgenden vorgestellt und erläutert werden.

4.2.1 Verlust von Adresszuordnungen

Dieser Abschnitt beschreibt das Auffinden und Ausgleichen des Verlusts von Adressenzuordnungen. Dieser Verlust entsteht bei der Umwandlung der Netzwerkadressen, die in Tabellen gespeichert sind, und wird vom NAT-Gerät verursacht.
Es muss stets damit gerechnet werden, dass eine Adresse, die vom NAT-Gerät umgesetzt und in der Tabelle gespeichert wird, verloren geht. Die Ursache kann der Ablauf der Gültigkeitsdauer des Tunnels sein. Der entscheidende Punkt hierbei ist nicht der Verlust der Adressenzuordnung an sich, sondern dass der HA schnellstens wieder seine Datenversendung an den Port fortsetzen kann, von wo aus die Daten weiter an die Adresse in der aktuellen Verbindung übertragen werden. Falls der MN trotz mehrerer wiederholter Versendungen eines Datenpakets keine Antwort auf seine Nachrichten bekommen sollte, und sein FA zudem durch einen UDP-Tunnel immer noch mit dem gleichen Router verbunden ist, über den kommuniziert wurde, sollte der MN sich durch Senden einer Registrierungsanfrage erneut beim HA registrieren. Dadurch erfährt der HA von der neuen Adressenzuordnung, so dass die Verbindungsmöglichkeit zwischen dem MN und dem HA wiederhergestellt wird. Nachdem eine neue Verbindung mit dem HA aufgebaut wurde, kann der MN eine kürzere Gültigkeitsdauer verwenden als vor dem Verlust der Adressenzuweisung. Insbesondere kann der MN von der vom HA bestimmten Gültigkeitsdauer abweichen. Tritt der Abbildungsverlust weiterhin auf, kann der MN bei jedem erneuten Registrieren den Wert dafür optimieren, um ein Intervall zu erhalten, das letztendlich eine ausreichende Gültigkeit garantiert. Wichtig hierbei ist, dass der HA keinesfalls versuchen sollte, verloren gegangene Adressenzuordnungen aufzuspüren

und hinterher die Lebensdauer in seiner Registrierungsantwort zu ändern. Denn sonst können Datenpakete an die falsche Adresse verschickt werden. Der HA hat dazu nicht genügend Information, um das zuverlässig durchzuführen, weil der MN sich in einer weitaus besseren Lage befindet, um Verluste zu erkennen. Der MN ist außerdem berechtigt zu entscheiden, dass die Adresszuordnung verfällt, falls er die dazugehörige Verbindung nicht mehr braucht.

4.2.2 Address Aliasing

Damit die Zusammenarbeit von Mobile IP und NAT-Geräten gewährleistet wird, wurde der Adressraum, in dem der MN tätig ist, erweitert. Dies führt zum Problem des sogenannten *Address Aliasing*, was im übrigen nicht allzu häufig vorkommt.
Seitdem es in privaten Netzwerken zur Überschneidung von Adressräumen kommen kann, können diese Adressräume manchmal miteinander verwechselt werden. Daher ist es notwendig, dass ein MN *Address Aliasing* unterstützt, um für die weitere Paketvermittlung den Überblick über die aktuelle Adresse eines Gateways zu bewahren. Eine Änderung der Adresse bedeutet dabei, dass wahrscheinlich das Netzwerk gewechselt wurde, obwohl er danach eine neue Adresse erhält und die alte IP-Adresse dennoch weiterhin existiert. In seltenen Fällen ist der MN nicht in der Lage, die von ihm zu benutzende Adresse des Gateways zu ermitteln, und verwechselt dadurch den Anschlusspunkt zu diesem Gateway mit einer anderen Gateway-Adresse. Der potentielle Ausfall des Datenverkehrs, zu dem die Situation schließlich führt, kann abgeschwächt werden, indem sichergestellt wird, dass die Verbindungsdauer kurz genug ist. Denn erneutes Registrieren nach Ablauf der Verbindung behebt das Problem.

4.3 Sicherheit

Für die Arbeitsweise des Zusammenspiels sind die Sicherheitsmechanismen des üblichen Mobile IP relevant und werden hier auch eingesetzt. Allerdings gibt es einen wesentlichen Unterschied: Beim Zusammenspiel muss sich der HA vollkommen auf die authentifizierten Adressen und Ports verlassen, an denen möglicherweise vom NAT-Gerät Änderungen vorgenommen wurden.
Wenn eine Verbindung aufgebaut oder aktualisiert wird, führt das Vertrauen zu unauthentifizierten Adressinformationen zu verschiedenen Anfälligkeiten gegen Umleitungsangriffe. Dabei führt ein Angreifer im wesentlichen genau das aus, was auch bei der NAT passiert. Adressen und Ports können verändert werden, um Datenpakete an eine andere beliebige Adresse weiterzuleiten. Findet keine NAT statt, wird genau genommen die COA in der Registrierungsanfrage direkt vom HA benutzt, um Daten zurück an den MN bzw. FA weiterzuleiten. Die COA ist dabei durch ihre Authentizität geschützt. Bei einer Kommunikation mit NAT dagegen ist die eigentliche COA aus Sicht der HA scheinbar die gleichwertige Adresse des NAT-Geräts. Nach Empfang von Paketen, die vom MN geschickt wurden, ist es zwar daraufhin möglich, nach der Entkapselung des Pakets die eigentliche Quelladresse zu erfassen, jedoch gibt es bei UDP-Tunneln keine Garantie, eine mögliche Modifizierung der COA des MN durch einen Angreifer zu erkennen. Einige Umleitungsangriffe können in gewissem Maße durch einfaches Kontrollieren des UDP-Tunnels abgeschwächt werden. Da diese Methode aber nicht vor allen Angriffen schützt, sollte bei UDP-Tunneln stets auf eine kurze Verbindungsdauer geachtet werden.

4.3.1 Umleitung des Datenverkehrs

Ein Angreifer, der sich auf der Route zwischen dem MN oder FA und dem HA befindet, kann Verbindungen zu einer gewünschten Adresse umleiten, indem er einfach die IP- und UDP-

Paketköpfe der Registrierungsanfragen verändert. Nach der Modifizerung der Registrierung braucht der Angreifer nicht mehr den Datenverkehr abzuhören bzw. zu manipulieren, da die Umleitung gezwungenermaßen erfolgt, bis die Verbindung abläuft oder der MN sich erneut registriert. Diese Anfälligkeit des UDP-Tunnels wird von einem Angreifer ausgenutzt, um Verkehr abzuhören, der an den MN gerichtet ist, und an ihn Daten zu senden. Die Angreifbarkeit kann auch für das Umleiten von Daten des „Opfers“ ausgenutzt werden, um Teile von seinen Diensten zu blockieren. Die einzige Abwehr dagegen sind kurze Zeitabstände zwischen den Registrierungsvorgängen, was die Dauer des Umleitungsvorgangs nach der beendeten Modifikation der Registrierungsnachrichten eingrenzen soll.

4.3.2 Gefälschte Gültigkeitsnachrichten

Durch die Registrierung eines MN an einem HA über ein FA entsteht eine andere Sicherheitslücke. Nachdem die Registrierungsnachrichten zwischen dem HA und dem FA ausgetauscht wurden, erwartet der HA vom MN, dass dieser eine erste Keepalive-Nachricht von sich gibt und die momentane Benutzung des Tunnels mitteilt, um somit den endgültigen Tunnelaufbau fertigzustellen. Die Verbindung ist nämlich in gewisser Hinsicht in einem „halbaufgebauten“ Zustand, bis die Gültigkeitsnachricht eingetroffen ist. Nachdem nun ein Angreifer den Versand der Registrierungsnachrichten festgestellt hat, nimmt er an, dass die Verbindung im „halbaufgebauten“ Zustand ist, und sendet dann eine gefälschte Gültigkeitsnachricht aus. Dabei muss der Angreifer nicht unbedingt fähig dazu sein, Pakete, die sich noch im Umlauf befinden, modifizieren zu können, sondern es reicht aus, den Austausch der Registrierungsnachrichten mitbekommen zu haben. Aus diesem Grund darf der HA auf keinen Fall eine Gültigkeitsnachricht akzeptieren, die von einer anderen IP-Adresse als von der der Registrierungsanfrage stammt. Diese Forderung grenzt das Ausmaß des Angriffs ein, Datenpakete an einen falschen UDP-Port weiterzuleiten.
Dieser Angriff kann folgendermaßen abgewehrt werden. Entweder werden kurze Zeitabstände zwischen den Registriervorgängen eingesetzt, mit der Absicht, nach dem Versand der gefälschten Nachricht die Dauer des Umleitungsvorgangs zu begrenzen, oder die Verweildauer im „halbaufgebauten“ Zustand wird minimiert, was dadurch geschieht, dass der MN unmittelbar nach Erhalt der Registrierungsantwort die erste Gültigkeitsnachricht verschickt.

5 Fazit

In dieser Arbeit wurden nun sowohl Methoden und Mechanismen zur Verständigung zwischen Mobile IPv4 und NAT als auch damit verbundene Konflikte und entsprechende Gegenmaßnahmen vorgestellt und beschrieben. Diese werden allerdings mit dem Einsatz und der Standardisierung von Mobile IPv6 nicht mehr nötig sein. Das Ziel von Mobile IPv6 entspricht dem von Mobile IPv4: Es soll für mobile Endgeräte permanente Erreichbarkeit gewährleistet werden, insbesondere bei der Trennung eines privaten Netzwerks vom Internet durch ein NAT-Gerät. Mobile IPv6 ist ausgereifter und vereinfacht einige Mechanismen beträchtlich. Mehrere Mechanismen, die bei Mobile IPv4 speziell für die Unterstützung der Mobilität und für die Verstärkung der Sicherheit integriert werden mussten, sind bereits ein fester Bestandteil. Beispielsweise ist in Mobile IPv6 die Sicherheit von Registrierungsnachrichten garantiert, da keine besonderen Maßnahmen während der Registrierung mehr getroffen werden müssen; auch ist der Mechanismus zur Erlangung einer COA entbehrlich geworden, da die COA über eine automatische Konfiguration erhalten werden kann. Abschließend ist zu sagen, dass durch die Standardisierung von Mobile IPv6 grundlegende Probleme von Mobile IPv4 behoben werden können, insbesondere die Inkompatibilität zwischen Mobile IPv4 und NAT.

Literatur

[EgFr94] K. Egevang und P. Francis. The IP Network Address Translator (NAT). RFC 1631, Mai 1994.

[HLFT94] S. Hanks, T. Li, D. Farinacci und P. Traina. Generic Routing Encapsulation (GRE). RFC 1701, Oktober 1994.

[LeVa03] H. Levkowetz und S. Vaarala. Mobile IP Traversal of Network Address Translation (NAT) Devices. RFC 3519, April 2003.

[Perk96a] C. Perkins. IP Encapsulation within IP. RFC 2003, Oktober 1996.

[Perk96b] C. Perkins. Minimal Encapsulation within IP. RFC 2004, Oktober 1996.

[Perk98] Charles E. Perkins. *Mobile IP: design principles and practices*. Addison-Wesley. 1998.

[Post80] J. Postel. User Datagram Protocol. RFC 768, August 1980.

[Schi03] Jochen Schiller. *Mobilkommunikation, 2. Auflage*. Addison-Wesley. 2003.

[SrHo99] P. Srisuresh und M. Holdrege. IP Network Address Translator (NAT) Terminology and Considerations. RFC 2663, August 1999.

Abbildungsverzeichnis

1 Paketweiterleitung zum/vom MN 63
2 Registrierung über den FA 64
3 IP-in-IP-Kapselung 65
4 Generic Routing Encapsulation 66
5 NAPT 67
6 Problem der Routing-Fähigkeit 68
7 Ablauf der UDP-Nachrichten 69
8 IP-in-UDP-Kapselung 70

LUNAR – A Lightweight Underlay Network Ad-hoc Routing

Maria Maleshkova

Kurzfassung

LUNAR is a compact and efficient routing protocol for wireless ad hoc networks. Its design and implementation solves the routing problem for the „small common case“, addressing spontaneously formed networks with 10 to 15 nodes and a diameter of maximum of 3 hops. LUNAR does not provide route repair mechanisms, route caching or packets salvation. However, with the method of forced route rebuilding controlled by a give timeout, it shows comparable performance with other ad-hoc routing protocols. The routing strategy combines both on-demand and pro-active routing in order to be able to create routes only when necessary but still have the possibility to actively rebuild them when the timer expires. There already exists a few different implementations of LUNAR, all having the main function of auto-configuration of the IP network addresses and support for both unicast and broadcast communication.

1 Introduction

LUNAR, Lightweight Underlay Network Ad-hoc Routing is designed, as its name says, to perform routing in ad-hoc networks, which are composed of a number of nodes connected by wireless links. The main challenge that has to be solved in such networks is caused by the problems, resulting form the flat structure of the network and from the topological network changes, because in an ad-hoc network typically every node is an endpoint and every node is a router and therefore, there is no node hierarchy. This challenge was taken by a number of research groups and as a result a few designs for protocols with a variety of approaches were produced.

A number of designs for protocols meant to effectively perform routing in ad-hoc networks started by only describing the core functionalities needed for establishing paths between nodes. However, soon the necessity of additional mechanisms such as route repair and maintenance or packet salvation were added because of the need to provide a reliable service. As a result, these first designs went through a number of reviews; the three protocols described later, that are used for comparisons, went through 10 to 13 revision cycles. Finally, protocols that started with slim and specific designs ended up having many more lines of code that initially planed and being partially really sophisticated to configure. Therefore, it could not be said that there is an ad-hoc protocol, whose implementation provides stability and ease of configuration.

This considerations leads logically to the need to design and implement a new protocol that fills the protocol gap of ad-hoc networks. LUNAR, in contrast to the continuous sophistication of ad hoc routing specification, aims to reduce the complexity of both routing algorithm and implementation by solving the common case for networks with a dozen of nodes reachable within 3 hops. The simplicity of operation and ease of configuration are also leading points that are taken into consideration in the design and implementation of the LUNAR protocol.

1.1 LUNAR Main concept

The LUNAR design and implementation is based on a few main decisions. First, it is important that the complexity of LUNAR is low because this enables it to be easily implemented in other environments. A lean protocol with small implementation can always be extended and made more efficient by adding elements, for example for self-configuration. However, it is necessary that the main functionalities are precisely designed and tightly implemented. That is why it is targeted that the core of LUNAR is kept small and clear, offering the possibilities for different extensions and extra functionalities.

It is important to mention about the LUNAR concept is that it adopts a hybrid routing style. This is so, because it uses both reactive ad hoc routing approach, since it discovers routes only when necessary, and proactive routing, in the sense that it actively rebuilds routes every 3 sec dependent on timeout and not on route failure. This is an important feature of LUNAR because it removes the need of additional mechanisms for route maintenance and link repair and in this way keeps the protocol complexity low.

LUNAR is also different from other ad hoc routing protocols because it is located below IP. In this way it appears to the IP stack that the mobile node is connected to a LAN and inside LUNAR there is an underlay at layer 2.5 that emulates LAN behavior by enabling point-to-point multihop routes. This underlay design methods saves the extra effort resulting from the interaction with the IP routing tables and enables the adding of self-configuration mechanisms in an uncomplicated way.

It is also necessary to be pointed out that LUNAR uses separation of delivery routes. This means that each pair of hosts has its own route and the originating node is responsible for keeping the routes active. This way of dealing with routes makes the implementation easier because nodes situated between two communicating ones only need to forward messages without actively participating. The main design and implementation decisions described above are made in accordance with the goals meant to achieve with LUNAR and based on the assumptions about ad hoc networks.

1.2 Basic Assumptions and Goals

LUNAR is especially created and designed to meet particular goals with its implementation. There already exist many other ad-hoc protocols, however, they are inefficient in achieving the main targets of LUNAR mentioned below:

1. LUNAR aims to solving the routing problem for the small common case. This means, it is designed for small, spontaneously formed networks with 10 to 15 nodes and a diameter of not more than 3 hops.

2. LUNAR should enable many common cases in parallel. This means that it is differentiated between logical and physical networks and logically independent networks can be setup and still able to physically overlap. In this way it is ensured that several ad hoc clouds are able to coexist, even if physically they have common nodes.

3. LUNAR should be easy to operate and integrate into existing IPv4 networking. Ëasymmeans that ideally, in order to start the LUNAR software, no parameters should be required.

4. LUNAR should have an actual working implementation, design close. The implementation code size should be small and the algorithms should not have many subtleties.

In order to be able to fulfill these goals, LUNAR makes a number of adequate assumptions. First, is it assumed that there is a certain *ad-hoc horizon* beyond which it is not economic to handle topology changes as they occur in mobile wireless networks. This is the reason for the limiting of LUNAR's route's length to 3 hops exactly, since more hops would be really hard to handle and are using up a lot of the resources anyway. This is so because wireless cards have it really hard to handle when multihop routing is used and this results in hight neighbor set fluctuation, i.e. position changes of moving objects have drastic effects on the neighbor set. Another reason for the existence and need of a *ad-hoc horizon* is the fact that routing information decays rapidly with the number of hops and only results in the necessity of route repair mechanisms and the need to buffer packets and ways to recognize packet reordering. It is really hard to determine the limit of the network when ad hoc routing is used because it is dependent on a lot of factors such as technology and network structure. However, in the case of LUNAR it is assumed that it is adequate to set this *ad-hoc horizon* to 3 hops in order to be able to keep the implementation simple and not to involve a lot of additional mechanisms to secure routes and package buffering.

Another basic assumption is that it is necessary that the protocol be simple to operate. It is not only important that a protocol manages the routing in an efficient way but it is also needed that it is not too sophisticated when in comes to configuring. The best ad hoc routing protocol is not going to be very widely used if it is really hard to tune and to configure. That is why LUNAR configuration is profile- based. In this way it offers the option to predefine settings and to self-configure. A user can enter a standard network using a default profile, or if all network participants agree on a profile they can build a individualized ad-hoc network.

2 Similar Protocols

The reason why LUNAR was designed is not because there are no other functioning ad-hoc routing protocol. There already exist a number of protocols such as DSR[7], AODV[8] and OLSR[9] that could all theoretically provide the functionality of LUNAR, however, are not especially designed for solving the problem for the small common case and providing simple implementation and integration into existing IPv4 networking.

DSR[7], for example, is an Ad-Hoc on-demand routing protocol. In order to communicate with another node, a node searches first its own route cache if it already has a route that can be used or if it needs to discover on. If there is no valid route, the node floods the network with Route Requests (RREQ). It then waits to receive a Route Reply (RREP) packet, which can be sent by any node that has a route to the requested destination node. The source routing and the path establishment are facilitated by affixing of addresses. The answering node adds its own address to the header of the RREP packets in order to enable the source routing and the node looking for a new route would use the same technique when adding the pre-hop path to the destination addresses, build with the help of the reply, to the packages sent. In order not to put too much extra additional weight on the network, DSR uses aggressive caching and overheard routes. In this way a node can get additional information about the network without having to send extra packets, by simply overhearing the other nodes that are communicating.

DSR needs mechanisms for route repair and maintenance because of the way it functions, which is superfluous in the case of LUNAR since routes are invalid after 3 seconds and have to be rebuild anyway.

Another on-demand protocol is AODV[8]. It is very similar to DSR because it uses the same technique for building routes. It uses flooding for discovering routes. However, it removes the source-routing overhead and instead uses a distance vector technique. In this way, the

nodes in the network exchange vector values with each other that give information about the distance between these nodes.

AODV would not be efficient to solve the small case problem, with 10 to 15 nodes and a maximum of 3 hops because the distance-information transferred with the help of the Distance Vectors is not so important and does not assist to achieve the goals targeted by LUNAR. The exchange of vectors would be necessary if the distance between nodes were of crucial importance, however, by LUNAR it is targeted to solve the network routing for routes with a small number of hops and therefore, additional information about the exact distance relations between nodes it not really needed.

There is also the class of the link state protocols and OLSR[9] is such a protocol. To reduce the routing traffic as much a possible, in OLSR only a number of nodes perform routing activities. These selected nodes are called mmulti point relayänd only they exchange link information with each other. In this way a OLSR uses a node hierarchy and the MPRs (Multi Point Relay) are the ones mainly responsible for doing the routing work. Besides the node hierarchy, another special feature of OLSR is the fact that it only maintains information about a subset of its neighbors which is also a difference to classical Links State protocols.

OLSR implementation would not be efficient for networks with a lager number of nodes, where a typical routing, without reducing the number of communicating nodes would increase the routing traffic and would use up too large amount of the network capacity. However, in the case which LUNAR is meant to solve, where only up to 15 nodes communicate with each other, it is unnecessary to implement a hierarchy of nodes and to reduce the information maintained about neighbors. The efficiently of routing traffic that OLSR is meant to achieve is only possible for larger networks and in the case of less than 15 nodes it would only add complexity to the network.

3 LUNAR

As already described, a number of other routing protocols exists that could perform routing in smaller networks. However, LUNAR is especially designed and structured to address the four major goals, already mentioned.

As it can be seen in figure 1, LUNAR is towards the IP stack as a network interface card that connects to a local area network, since it is placed directly over the wireless connection. As a result of this positioning, all nodes from LUNAR's network will appear as being one hop away, while actually inside this is realized by forwarding packets over multiple hops.

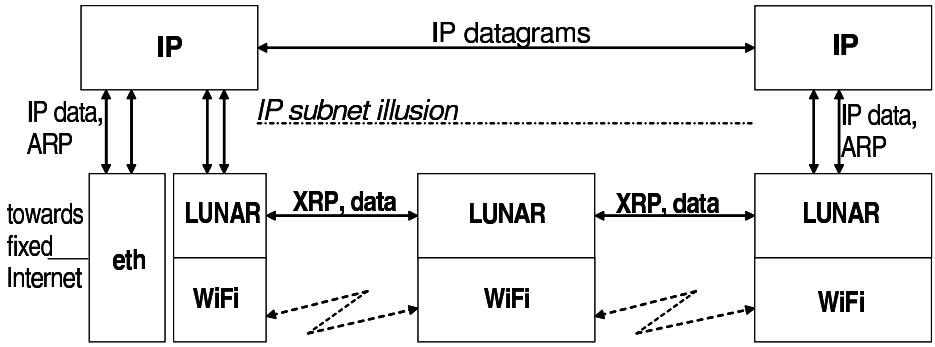


Abbildung 1: LUNAR as an underly to the IP layer

A typical use case of LUNAR would be the following:

Two nodes in a LUNAR network want to communicate with each other. The first one is the source node and it wishes to send an IP packet to the second one, the destination node. This usually happens through an ARP request, emitted internally by the sender. This ARP request is caught my the LUNAR routing layer and is then mapped to a LUNAR Route Request (RREQ) message, that is transmitted through the network. When the destination node receives the RREQ it answers with a RREP (Route Reply) and an unicast route is built. After the RREP is received by the source, it is mapped back to a ARP reply message by the source node, allowing the IP stack to send IP datagrams to the destination node.

In this way LUNAR transmits the ARP requests and ARP replies between the nodes and build the routes, an absolutely transparent for the IP layer, in the sense that it does not participate in actively negotiating the connection like most other routing protocols. LUNAR routes, however, have a limited lifetime set to 3 seconds, after which they have to be rebuild. That is why a particular route is in use only 3 seconds.

The restricted lifetime of the LUNAR routes is tightly connected with the lack of route repair, route caching, and packets salvation functions. Instead of facilitating a reliable transfer of data, LUNAR builds a new route between the source and the receiver every 3 seconds, eliminating the need of repair. This fact should be seen as a positive feature of LUNAR because its routing and forwarding strategies are simpler than those of other MANET routing protocols, but test show that the performance is just as good.

The following sections describe the four main building blocks of LUNAR:

3.1 IP adaption and steering layer (ASL)

The main function of the IP adaption and steering layer (ASL) is to translate events. As figure 2 shows, it communicates between the IP stack and the LUNAR application level, mapping ARP and DHCP requests to the corresponding LUNAR ones. As already mentioned, the ASL maps, for example, the ARP request to LUNAR Route Request (RREQ) and then maps RREP back to an ARP reply.

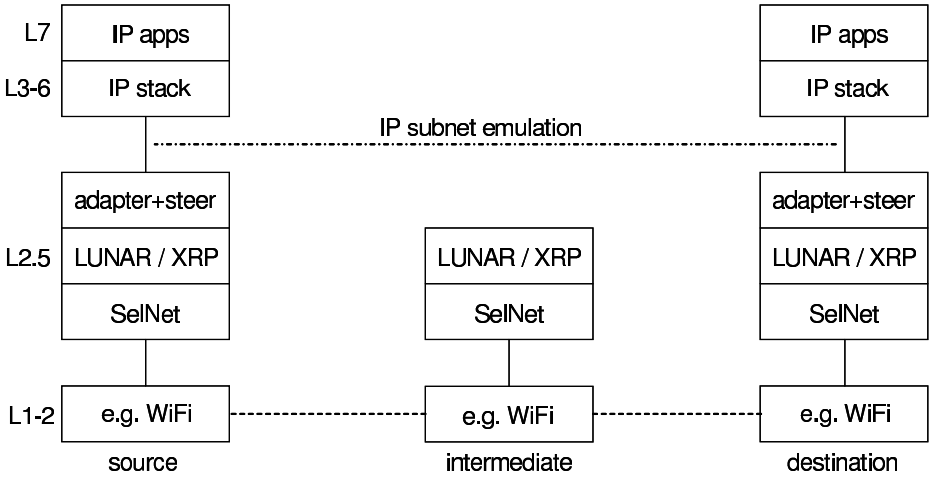


Abbildung 2: LUNAR building blocks

The main functions of ASL include: to start the route discovery, to refresh routes and to prove IP addresses allocation support, in other words, LUNAR's IP Adaption and Steering Layer (ASL) is responsible for steering LUNAR's main operations. This can be achieved because ASL is an emulation that creates a IP subnet illusion, using the LUNAR services and therefore, make LUNAR appear to the IP stack as a low level Ethernet device. It this way all mobile

nodes of LUNAR can be assigned to the same IP subnet, which in the standard profile is defined by using 192.168.42.0/24 as the default IPv4 prefix. This feature of the nodes, that they all have to belong to the same IP subnet, is freely chosen but also necessary because LUNAR could be implemented directly on the WiFi interface card itself. This would mean that LUNAR would have the same features as a Ethernet LAN, which can be formulates as one IP subnet.

As already mentioned, the LUNAR protocol uses as standard IPv4 prefix 192.168.42.0/24. However, it still has to be considered how the IP stack obtains a dynamic IPv4 address. This happens with the use of DHCP protocol. ASL actually implements a DHCP server by intercepting DHCP queries of the nodes and answering them. In this way the standard functionality of the server is emulated and can be used for IP address allocation. ASL receives a DHCP_DISCOVER query form a node and the address allocation request can be outside the LUNAR prefix, inside the prefix, or without an IP address. In the first case, where the DHCP_DISCOVER query requests an address outside the specified prefix, a DHCP_NAK is send. If the allocation is inside the LUNAR prefix, it has to be tested for collisions before being assigned. This happens by trying to create a LUNAR route for the given address. If the attempt fails, this means that the IP address is not already used and a DHCP_ACK message is returned. If a route for the given address is discovered, the address already exists in the LUNAR network and therefore, a DHCP_NAK is returned. The third option, that a DHCP_DISCOVER without IP number is received, handled by the measure that a random host id within the LUNAR IPv4 prefix is chosen and than the same testing process as if the address was initially send within the prefix is used.

In this way the ASL is responsible for setting up an IP subnet and allocating IPv4 addresses by answering the DHCP queries. However, it is also responsible for route discovery steering and forced rediscovery of routes. The whole route discovery process is managed with the help of IP-to-Selector tables, where IP destination addresses are mapped to selectors. ASL maintains such table in every node, where the selectors identify entries in the forwarding information base (FIB) pointing to SelNet route pointer. The route discovery process is initiated when the IP stack wants to sent a datagram to a host in the LUNAR network for the first time. This happens by the IP stack issuing an ARP request (to map IP number to Ethernet address) which is handled by the ASL. There are two options for the host address for the ARP request: it is either in the IP-to-Select table or not. If the address is in the table, an ARP reply is sent, containing the target's Ethernet address, which is composed of the low 48 bits of the associated selector, and in this way the IP address to Ethernet address mapping is completed. If the address is not in the table, this means that the ASL has to start a route discovery process. LUNAR has an implemented routing algorithm that is started in this case, and if a new route within the given limit of hops if found, it is added to the IP-to-Selector table and the ARP reply can be sent, just as it was done when the host was found in the table. However, the routing algorithm can also fail to find a route, then simply more discovery attempts will be made with a break between each one. If after a limited number of attempts there is still no route found, the request is silently dropped. The same happens if requests are received for hosts that are already being processed.

The last significant function of the Adaption and Steering layer is to perform forced rediscovery of routes. In this way LUNAR does not need any additional mechanisms for route maintenance, since they are actively rebuild after a given time period. In order to always have as many active and functioning routes as possible, LUNAR uses a timer for each entry in the IP-to-Selector table. After the time given by the timer is up, the ASL launches a LUNAR discovery for the given IP addres, however, only if the SelNet route corresponding to the entry was used within the given time. If this is not the case the table entry is deleted. When the route discovery is launched, the old entry is kept until the routing is successfully completed. Than it is updated and an ARP reply is send to IP stack with the new Ethernet address.

If the route discovery fails, the entry is deleted. This way of route management removes the need of garbage collection for old routes, and as already said, ensures that routes are active and usable.

In this way the IP Adaption and Steering Layer manages to perform the functions of route maintaining, route building, and IPv4 address allocation and these functionalities provide the necessary communication between the IP stack and LUNAR and give the base for operation of the LUNAR protocol engine.

3.2 LUNAR protocol engine

The LUNAR protocol engine is responsible for the the routes detection between the source and the destination node and for the generation of LUNAR RREP and RREQ messages after they were mapped by the ASL. The LUNAR protocol engine implements the core routing algorithm and this is achieved by performing two basic functions: creation of unicast SelNet routes and creation of SelNet broadcast delivery trees.

By the creation of unicast SelNet routes, a target IP address is matched to a selector, identifying the first network pointer of the SelNet route. LUNAR discovers routes by a flooding algorithm and they are established only on demand. The source node broadcasts a RREQ with the following fields (only the required fields are listed in table 1, there are also four optional ones). The RREQ_TTL gives the number of remaining permitted hops, out of initially 3, the RREQ_SERIES indentifies the type of route request, the target or destination node is given by RREQ_TARGET and finally, the address where replies are to be send is given by RREQ_REPLY.

RREQ Fields		
Field Name	Data Type	Comment
RREQ_TTL	int	remaining permitted hops
RREQ_SERIES	selector	identifies the RREQ
RREQ_TARGET	IPv4 or IPv6	identifies the target
RREQ_REPLYTO	sel/eth pair	where to send back replies

Tabelle 1: RREQ

Route Reply (RREP) messages have the following fields listed in table 2(only the required fields are listed, there is also one optional). The RREP_HOPS gives the distance to the target node in hops and the RREP_FWPTR is the forwarding pointer that shows where data has to be send to, in the form of a selector-ethernet address pair.

RREP Fields		
Field Name	Data Type	Comment
RREP_HOPS	int	distance to target
RREP_FWDPTR	sel/eth pair	where to send data to

Tabelle 2: RREP

A simplified description of the LUNAR route discovery algorithm would be the following. First, broadcast search with range of one hop is performed. If the destination is within this range, the communication details are unicast to it. If the destination node is not a direct neighbor, a maximum hop (for example range is set to 3 hops) search is performed by requesting neighbors to forward a search request. If the routing is successful, addressing

information is send by unicast, the same way as by the one-hop broadcast. An entry is than made in the node's IP-to-Selector table and the timer for it is started, so that after 3 sec the route has to be rebuilt. The discovered path is uni-directional (IP to Ethernet), therefore, if the destination node wishes to send packets in the other direction it has to build its own path.

The broadcasting is enabled in LUNAR with the use of delivery trees that are used for subsequent broadcast. The creation of such a tree is initiated by the source and it broadcasts, while neighbors rebroadcast a message. For joining a tree, neighbors can reply and if only one of them replies a unicast branch is added and if more, broadcast is used for information transfer.

In this way the LUNAR protocol uses unicast or broadcast to deliver packets and to build routes using the discovery algorithm. How the communication between LUNAR engines is facilitated is described in the next section.

3.3 XRP

XRP stands for eXtensible Resolution Protocol and is responsible for facilitating the interaction between LUNAR engines. The XRP defines the format for LUNAR's messages, where each XRP message is a sequence of XRP commands. The communication between nodes is enabled on LUNAR protocol level by the use of request and reply messages. The XRP main function is the resolve commands, for example it is used to resolve the address of a neighbor to which a node wants to send data, or the resolve a given IP address to a multihop path. All XRP messages have the same structure, starting with a header, then followed by a container for various parameters and closed by a nnullöbject. The exact structure of the XRP message is shown below.

version	ttl	flags	reserved
lenght (bytes)		class	class-type
...content...			
0			

Tabelle 3: XRP Message

The header (the first line) is 4-byte and it deals with the version, the time to live and the flags associated with the header. It is followed by command parameters that are always padded to a 32-bit boundary. The end of the message is marked by a zero-block. A typical request would contain the following parameters:

1. Request series (len=12, class=request series, ctype=sel) is used to prevent broadcast conflicts.
2. Address to resolve (len=8, class=target, ctype=IPv4), which indicates that an IPv4 address is to be resolved.
3. Requested resolution(len=4, class=reqstyle, ctype=sel/eth), which determines that the resolved address should be from selector/ethernet type.
4. Reply address (len=20, class=reply addr, ctype=sel/eth) is the location where the result of the request is to be received.

By using the four parameters above a LUNAR route request can be described. The route reply is much simpler in form and it has only one parameter, indicating a selector/ethernet address pair: Forward address (len=4, class=reqstyle, ctype=sel/eth). In this way the requesting node knows where to forward data traffic to. However, the XRP message exchange would not be possible without the underlaying SelNet and its forwarding function.

The following pseudo code is an example for multi-hop resolution composed of a sequence of XRP commands.

jumponexistence(REQUESTID)
resolve(scheme=sapf, id=REQUESTID, style=selector, target=0);
resolve(scheme=sapf/eht, id=MYSEL/MYETH, style=selector, target=TMP);
resolve(scheme=ipv4, id=IPNUMBER, style=sapf/eth, target=TMP, series=REQUESTID, depth=2);

The first command is used to determine if a selector has a corresponding handler or not. If the handler for the selector exists the complete packet will continue its execution at this place and if it does not exist, the following XRP command in the packet will be executed. In this case the second command redirect the whole packet to location 0, which is the garbage bin. In this way only the first instance of a query packet at one node will be executed because only it will receive a handler. All other ones will not find a handler and therefore, be with the second command automatically redirected and deleted.

The third command creates the delivery path that is used for returning the result of the resolution in command 4. The last command performs the actual resolution of the IPNUMBER and the corresponding address is then returned by using the TMP selector and the delivery path created by the command before.

The depth parameter plays a really important role because it gives the allowed number of re-issuing of resolution queries and forwarding the resolution request with a number of necessary modifications. Depth=2 means that the resolution request can be forwarded two times before being discarded.

3.4 SelNet virtual link layer

The Selector Network is a layer, working directly on top of the link layer (in LUNAR‘s case WiFi). It is independent from IP addresses and IP routing and provides the building of unreliable routes for datagrams. It functions by trapping IP data and control traffic and translating it to an representation that can be handled and processed by SelNet nodes; for example ARP packets are trapped and rewritten to XRP commands. A SelNet-over Ethernet frame has the following format shown in table 4 ,where the selector and the data fields form the SelNet packet format, which is used for XRP requests and replies and also for pure data traffic.

dst (48)	src (48)	typ (16)	selector (64)	data

Tabelle 4: SelNet Frame over Ethernet

The data field is of variable length and therefore, the total packet's length has to be extra provided. The selector, on the other hand, is fixed to 64 bit and it permits to demultiplex received packets to suitable handlers, with the help of FIB (Forwarding Information Base), which is the table that contains the entry pairs of selectors and packet handlers. Such a handler could be a forwarding procedure or a procedure that delivers the data to the IP

stack after removing the selector header. In this way every received packets is assigned a corresponding handler that performs the necessary actions.

However, in order to enable multihop forwarding, a handler that forwards the received packet to one or more neighbors, after having replaced the selector field with a new value that appropriate for the new node, is not enough. That is why LUNAR uses NNetwork Pointersön intermediate nodes. In this way packets are forwarded from pointer to pointer until they reach the end of the SelNet route, while ethernet address and selector are rewritten every time they change a node. This is how with the help of forwarding of data packets and the use of network pointers, the transmission of packets along multihop routes in SelNet is possible.

4 Implementation

4.1 Software Architecture

The first implementation of LUNAR was a Linux user space programm. As it can be seen in figure 3, LUNAR places itself between the IP stack and the networking hardware drive (wireless card drive). In this way NETLINK provides information about pending ARP requests.

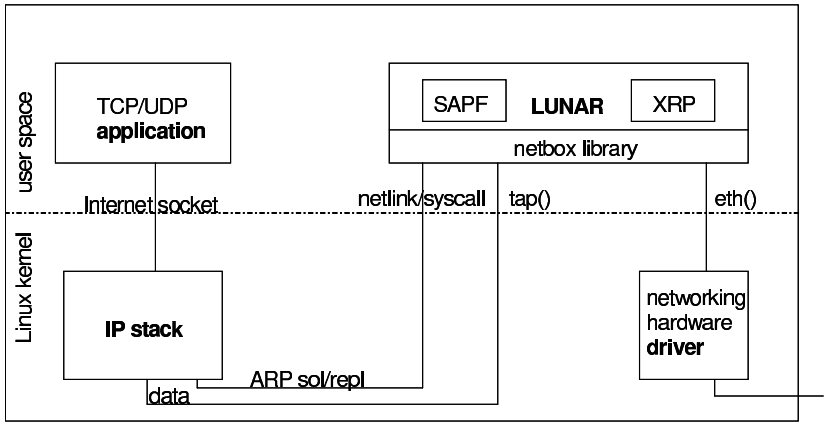


Abbildung 3: LUNAR node architecture

ARP packets can also be captured when the IP stack sends them out. However, even though ARP packets can be easily captured, it is necessary that the ARP cash is directly accessed. When the LUNAR delivery path times out, the IP stack is noticed that it has to resend an ARP request before continuing with the data transmission so that no packets are lost. In this way, LUNAR routing is implemented and a new route is created after 3 seconds in parallel to the existing one and than switches silently to the new one.

There is also a newer version of LUNAR that is fully located in the kernel. It does not have the TUN/TAP and NETLINK dependency, since otherwise it would be necessary that users recompile their kernel for TUN/TAP and NETLINK support. The new kernel version of LUNAR also creates an ethernet device which can be used by the IP stack like usual subnet.

Both implementations are rather small. Table 5 shows the Linux implementation of LUNAR code line count. In comparison, the other ad hoc routing protocol implementations are in the rang of 2500 to 8000 lines of C code, do not have any self-configuring logic.

#of C lines	Module	Content
550	lunar.c	main() program and self-config logic
420	netbox.c	OS glue and timer support
330	sapf.c	SAPF engine
500	xrp.c	XRP and resolution forwardinglogic
160	xrp pkt.c	XRP encoding/parsing
1960	Total	

Tabelle 5: LUNAR Lines of Code

4.2 Implementation Examples

As already seen, the size of the implementation is very small and it is possible that LUNAR and a Linux system are saved on a single floppy disk. Since gatewaying into the Internet is achieved without explicit nomination of a default gateway and a NAT module will map the 192.168.42.x LUNAR subnet address to a topologically correct IP number, it is possible only with this floppy disk to implement a self-configuring access point that enables Internet connection for mobile nodes using ad hoc routing.

4.2.1 LUNAR for Embedded Systems

The LUNAR protocol was also implemented in the very restricted and tight environment of a LEGO Mindstorm robot, which is only possible because of the simplicity of the routing protocol and the small size of the implementation. As an application scenario, the the robot was steered over IP and then the communicated information was reflected in a html page displaying the actual position and status. Because of the constraint environment the SelNet layer was made to cover layer 2 and address fields were reduced to 16 bits.

4.2.2 LUNAR for Bluetooth

There exists a version of LUNAR that enables the UDP and TCP over multiple Bluetooth piconets. This is complicated to implement because of the structure and the functionality of piconets. In such a Bluetooth net there is always one master node to which up to seven slave nodes are connected. All communication in a piconets passes through the master node and new nodes can join a net only after a long inquiry process and after that become slave nodes. Connected piconets form a scatternet, by having a node that acts as a master node in its own piconet and as a slave node in the other network to which it is connected. This complexity of the Bluetooth networks is hard to cover with LUNAR since in piconets there have no broadcast option and all communication is connection oriented.

In the case of LUNAR these problems are solved by letting idle slaves temporarily disconnect from the master and look for new piconets to bridge to and than feed this information back as if it was gathered by broadcasting. This is a necessary way of solving the problem because a source node, which in LUNAR is responsible for establishing the delivery tunnel and sending the data, would otherwise be disconnecting and reconnecting the whole time only because it looks for nodes in potential neighbor piconets.

4.2.3 LUNAR for Windows

Since a protocol is really applicable only when it has implementation for different platforms, a LUNAR version for Windows XP/2000 is also available. It is based on the kernal Linux

version and that is why the core logic of LUNAR stays unchanged, while the modules containing networking and platform functionalities are modified. LUNAR for Windows is implemented as NDIS (Network Driver Interface Specification) because in this way incoming or outgoing ethernet packets can be intercepted and replaced by new packets, which is the main functionality necessary for the implementation of LUNAR.

5 Performance

LUNAR was tested in a variety of different ways, focusing on diverse criteria. Such test include pings, running of TCP and UDP applications, flood pinging and broadcast pings in parallel, both for the one-hop and multi-hop cases. For the case where LUNAR's performance was tested with ping, a linear network of three stationary nodes and one mobile node, that roams along this network, were used. In this test LUNAR showed better ping performance (96,5%) then OLSR and AODV. It can be argued that the route rebuilding every 3 seconds is inefficient, however, it has to be taken into consideration that AODV needs 2 seconds to conclude that HELLO messages are lost or that the topology of the networks has changed and this before it has started its route repair or route rediscovery. Another very important fact connected with testing LUNAR it that there exists the danger of looping packets, since there is no TTL field for data packets. However, the use of forwarding credits, which limit the number of package passes, and the use of formal validation show that LUNAR operates correctly.

It must be pointed out that most test result were made in comparison to OLSR, since it has a stable Linux implementation. Nevertheless, LUNAR showed good and comparable ping performance results for routing and it has the positive feature of being simple, without fancy optimizations, and has a very small implementation size, including the functions of broadcasting, address auto-configuration and automatic IP gatewaying.

6 Conclusion

It can be concluded that with its design and implementation LUNAR successfully fulfills the goals set for its development. It traps ARP request, rewrites them to XRP requests, which are then rebroadcasted by transit nodes to the destination node by setting up a multihop route. This mechanism of rewriting messages is only possible because LUNAR is placed below IP and above Ethernet and in this way it can control and manage the data exchange between the two layers. With its forced rediscovery of routes, LUNAR removes the need for extra mechanisms for route maintenance and with the help of additional assumptions about the network, provides a slim and focused implementation. It is important also to point out that by solving the routing problem for the common small case of networks with up to 15 nodes and with diameter of 3 hops, LUNAR shows comparable performance with other ad-hoc routing protocols and even has the advantage of being really easy to configure and with small implementation size.

I personally believe that it is unsatisfying to base the whole design and implementation only on IPv4, without taking into consideration the introduction of the IPv6 standard. The whole self-configuration mechanism and the IP to ethernet address pairing is exclusively based on IPv4 and this means that LUNAR is practically unadaptable to a new IP standard. Another problem is that there are no security considerations about the way the protocols is implemented and in this way it is vulnerable to practically to all attacks. If any security measures are included the LUNAR implementation will then be no longer so slim and will louse its positive feature in comparison to other ad hoc protocols.

Literatur

[CLI] DSR Click Router Implementation homepage. http://pecolab.colorado.edu/DSR.html.

[E4A] CA.Stemper E4A. Ad Hoc Implementationen. www.ist.lu/users/thomas.engel/2002-adhoc-workshop/ws8.ppt.

[LHP] LUNAR home page. http://cn.cs.unibas.ch/projects/lunar and http://www.docs.uu.se/selnet/lunar/.

[NoLu] Erik Nordström und Henrik Lundgren. AODV-UU: AODV Implementation. http://www.docs.uu.se/scanet/aodv.

[PlLa] Adokoe Plakoo und Anis Laouiti. OLSR implementation. http://manetou.inria.fr/olsr.

[TsGo01] Christian Tschudin und Richard Gold. SelNet: A Translating Underlay Network. Submitted for publication, http://www.docs.uu.se/selnet/selnet.pdf, October 2001.

[TsGo02] Christian Tschudin und Richard Gold. LUNAR - A Lightweight Underlay Network Ad-Hoc Routing. Submitted for publication, http://www.docs.uu.se/docs/research/projects/selnet/lunar/lunar.pdf, January 2002.

[TsGo03] Christian Tschudin und Richard Gold. Lightweight Underlay Network Ad-Hoc Routing implementation. Uppsala University Teschnical Report, 2003.

[TsGo04] Christian Tschudin und Richard Gold. Lightweight Underlay Network Ad hoc Routing (LUNAR) Protocol. Internet draft: draft-tschudin-manet-lunar-00.txt, 2004.

Abbildungsverzeichnis

1 LUNAR as an underly to the IP layer 78

2 LUNAR building blocks 79

3 LUNAR node architecture 84

Tabellenverzeichnis

1 RREQ 81

2 RREP 81

3 XRP Message 82

4 SelNet Frame over Ethernet 83

5 LUNAR Lines of Code 85

Mobilitäts- und Benutzermodellierung für drahtlose Kommunikation

Inna Löwen

Kurzfassung

Der Zugang zum Internet mittels Laptop, PDA oder Handy gehört heute dank drahtloser Übertragungstechniken zum Alltag. Allerdings wirft die Mobilität der Teilnehmer immer noch das Problem auf, dass bisherige Kommunikations-Protokolle die Mobilität nicht oder nur unzureichend unterstützen. Um eine einwandfreie drahtlose Kommunikation zu ermöglichen, müssen daher neue Protokolle entwickelt, getestet und ausgewertet werden. Da die Benutzung realer mobiler Netze zur Evaluation von Protokollen zu keiner wirklich brauchbaren Lösung führt, schafft die Benutzermodellierung im Rahmen von Simulationsumgebung Abhilfe. In dieser Ausarbeitung wird speziell auf diese Problematik in den Ad-hoc-Netzen eingegangen. Es werden einige Benutzermodelle vorgestellt und deren Anwendung in der Auswertung der Protokolle erläutert, sowie ihre Brauchbarkeit analysiert.

1 Einleitung

Nicht nur das stationäre Festnetz, sondern in Zukunft auch das mobile Internet ermöglichen den Zugang zu multimedialen Informationen mit unterschiedlichen Endgeräten an jedem Ort und zu jeder Zeit. Spätestens seit der Entwicklung von Bluetooth verfügen mehr und mehr Geräte um uns herum über drahtlose Kommunikationsmöglichkeiten. Neben Bluetooth sind auch weitere praktische Lösungen für drahtlose Netzwerke bereits entstanden, wie zum Beispiel: WLAN, IEEE 802.11b oder HIPERLAN/2. Die Ad-hoc-Netze, aufgrund ihrer vielfältigen Einsatzmöglichkeiten und ihres Verzichts auf Infrastrktur, bieten in diesem Zusammenhang großes Potential. Ad-hoc-Netze bestehen aus mobilen Endgeräten. Die Topologie solcher Netze ist naturgemäß sehr dynamisch, und die Anforderungen an ein Routing Protokoll sind sehr hoch. Eine Veränderung der Netzstruktur wird nicht nur durch die Mobilität hervorgerufen, sondern auch durch Signalverluste, Interferenzen oder einfach durch Abschalten einzelner Geräte. In dieser Ausarbeitung wird insbesondere die Mobilität der Benutzer betrachtet. Zur Evaluierung der Protokolle ist man auf die Approximation der Benutzerbewegungen im Rahmen von Simulationsumgebungen angewiesen. Das optimale Modell wären natürlich die Benutzer und ihre Geräte selbst. Angesichts der Menge der benötigten Geräte, die im folgenden als Knoten bezeichnet werden, wäre der Aufwand für solch eine experimentelle Auswertung allerdings zu groß. Aus diesem Grund bedient man sich eines Bewegungsmodells, das man nach verschiedenen Benutzerbewegungsmustern möglichst realistisch ableitet. So sind schon jegliche Bewegungsmodelle entstanden, von denen hier einige vorgestellt werden. Das bei weitem bekannteste Random-Waypoint-Model wird zuerst erläutert. Anschließend wird auch ein Group-Mobility-Model für Ad-hoc-Netze beschrieben. Da man bestrebt ist die Szenarien so realistisch wie möglich zu simulieren, werden dem gegenüber das Obstacle-Mobility-Model sowie das Activity-Based-User-Model vorgestellt. Zuerst wird allerdings ein kurzer Überblick über die Ad-hoc-Netze und ihre Eigenschaften verschafft.

2 Was sind Ad-hoc-Netzwerke?

Ein Ad-hoc-Netzwerk ist eine Menge von drahtlosen Mobilknoten, die ein Netzwerk ohne vorhandene Infrastruktur aufbauen können und keine zentrale Verwaltung besitzen. Jeder einzelne Mobilknoten in dem Netzwerk arbeitet als Endgerät und Router. So organisiert sich jeder Teilnehmer im Netz selbst und kann sich frei bewegen. Im Gegensatz zu festverdrahteten Netzwerken ist also ein Ad-hoc-Netz sehr dynamisch. Durch die Bewegung von Knoten sind die Zeitpunkte der Lokationsänderungen, die sehr schnell eintreten können, nicht vorhersehbar. Dadurch kann es zu Verbindungsabbrüchen und neuen Verbindungen kommen. Allein das Ein- und Ausschalten der Geräte kann eine Änderung der Netztopologie bewirken [Bild 1]. Eine weitere Eigenschaft der Ad-hoc-Netze sind die asymmetrische Verbindungen. Ein solcher Fall liegt dann vor, wenn die Verbindung zwischen zwei Knoten nicht bidirektional ist. D.h. Wenn es zwar vom Knoten A einen Weg zum Knoten B gibt, jedoch kein Rückweg vom Knoten B zum A. Eine asymmetrische Verbindung ist auch dann vorhanden, wenn Hin- und Rückrichtung eines Kommunikationskanals verschiedene Bandbreiten haben. Eine noch zu erwähnende Eigenschaft bilden die Interferenzen. In Ad-hoc-Netzwerken können Verbindungen entstehen und wieder abbrechen, je nachdem wie die Übertragungsbedingungen sind. Eine Übertragung kann eine andere stark stören. Knoten können Daten empfangen, die nicht für sie bestimmt waren. Sobald zwei Knoten, die nahe beieinander sind, mit einer Übertragung beginnen, können sie sich gegenseitig stören und die Daten vernichten. Anhand dieser Eigenschaften ergeben sich für die Kommunikation, im Gegensatz zu Festnetzen, ganz andere Protokollanforderungen. Dabei ist ein wesentlicher Aspekt die Wegefindung vom Sender zum Empfänger - also das Routing.

Eines der wichtigsten Attribute von Ad-hoc-Netzen ist, dass sie keine Basisstation haben und dadurch in vielen Bereichen einsetzbar sind. Ihre Anwendungen finden sie bisher zum größten Teil dort, wo sich der Nutzer eines Kommunikationssystems nicht auf eine Infrastruktur verlassen kann, die Nutzung zu teuer ist oder gar keine Infrastruktur existiert- nämlich im militärischen Bereich. Es gibt aber auch viele Ideen in anderen Bereichen. Die möglichen Einsatzgebiete von Ad-hoc-Netzen reichen von Rettungseinsätzen, Katastrophenschutz bis zu mobilen Konferenzen und Heimnetzwerken.

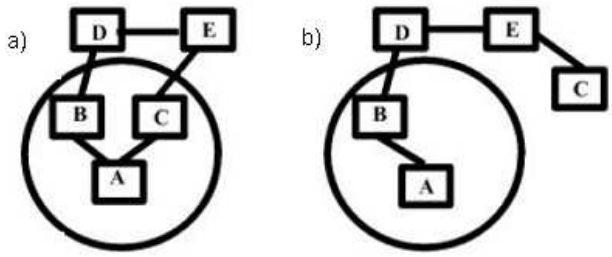


Abbildung 1: Änderung der Netztopologie

2.1 Datenübertragung in Ad-hoc-Netzwerken

Durch das Fehlen der zentralen Einheiten bei Ad-hoc-Netzen müssen die Mobilknoten selbst alle Dienste für die Funktion des Netzwerks erbringen. Das heißt, im Gegensatz zu infrastrukturellen Netzwerken, findet hier die Kommunikation direkt zwischen zwei oder mehreren Knoten statt ohne, dass die Informationen durch eine Basisstation zur richtigen Empfängeradresse weitergeleitet werden.

Die Leistung des gesamten Netzwerks hängt daher sehr von dem kooperativen Verhalten eines jeden Knotens ab. Dabei kommunizieren Knoten, die in Reichweite der drahtlosen Funkschnittstelle des Kommunikationspartners liegen, direkt miteinander, während weiter entfernte Knoten mittels anderer Knoten erreicht werden. Die Knoten, welche Weiterleitung von Informationen übernehmen, fungieren als Router.

2.2 Probleme beim Routing in Ad-hoc-Netzen

Routing beschäftigt sich allgemein damit, Pakete in einem Netz von einem Startpunkt zum Ziel über eine ideale oder annähernd ideale Route zu transportieren. Was eine ideale Route ist wird durch eine Metrik festgelegt. Routing in Ad-hoc-Netzen stellt ein großes Problem dar, bedingt durch den dynamischen Charakter des Netzes. Durch die Mobilität der Knoten wird eine teilweise hohe Änderungsgeschwindigkeit der Netztopologie bewirkt, was zu ständigen Änderung der Übertragungsbedingungen führt. Daher können zentralisierte Ansätze, wie sie im Festnetz üblich sind, nicht verwirklicht werden. Um also einen einwandfreien Informationsaustausch zwischen den Nutzern zu ermöglichen, müssen Wege gefunden werden, wie ein jeweiliger Router die richtige Zieladresse findet ohne sich auf das Wissen einer zentralen Vermittlungsinstanz zu stützen, die in den infrastrukturellen Netzwerken die Basisstation bildet. Die Routingprotokolle, wie sie in Festnetzen benutzt werden, arbeiten nicht effizient genug oder versagen komplett. Daher müssen neue Protokolle entwickelt werden. Hierbei muss berücksichtigt werden, dass zum einen Energievorräte und Speicherkapazitäten eines jeden Knotens eingeschränkt sind und zum anderen viele Knoten die Fähigkeit haben müssen, Informationen weiterzuleiten. Jeder Knoten in Ad-hoc-Netzwerken muss lokal die Entscheidung hinsichtlich der Weiterleitung treffen. Einige von solchen Protokollen werden im nächsten Abschnitt vorgestellt.

Abbildung 2: Routing in Ad-hoc-Netzen

2.3 Routingprotokolle

Es gibt zwei Kategorien von Routingprotokollen, nämlich tabellenbasierte (table-driven) und bedarfsgesteuerte (demand-driven). Die tabellenbasierten Routingprotokolle versuchen fortlaufend, Wege zu allen Knoten zu finden und in der Routingtabelle zu speichern. Jeder Knoten erhält eine oder mehrere Routingtabellen und aktualisiert sie bei Änderungen der Netztopologie. DSDV, CGSR und WRP sind zum Beispiel solche Routingprotokolle.

Protokolle wie AODV, DSR, HSR usw. gehören im Gegenzug zu den bedarfgesteuerten Routingprotokollen. Sie beginnen mit der Suche der jeweiligen Zieladresse erst dann, wenn zu dieser Daten übertragen werden müssen bzw. wenn eine Verbindung zwischen zwei bestimmten Knoten gebraucht wird. Der Vorteil gegenüber tabellenbasierten Protokollen ist hier der, dass die Protokolle nur einen Weg zwischen den betroffenen Knoten aussuchen, ohne eine komplette Tabelle aufzubauen.

3 Bewegungsmodell und Simulation

Wie im Unterabschnitt 2.2 erwähnt, müssen für Ad-hoc-Netze neue Protokolle entwickelt werden. Vor ihrem tatsächlichen Einsatz in der mobilen Kommunikation, müssen diese natürlich auf ihre Leistung getestet werden. Die direkte Nutzung eines realen mobilen Systems zur Evaluierung der Protokolle könnte zwar bedeutende Informationen liefern, ist jedoch aufgrund eines verhältnismäßig hohen Aufwandes nicht umsetzbar. Stattdessen werden hierzu Simulationsumgebungen eingesetzt. Simulationen basieren hier auf der eigentlichen Protokollimplementierung einerseits und einer Approximation des Benutzerverhaltens andererseites. Die Simulation lässt man mit Hilfe eines ausgewählten Bewegungsmodels mehrmals durchlaufen. Die Simulationszeit bleibt für jeden Durchlauf die gleiche (z.B $900sec$). Im Anschluss wird das Durchschnittsergebnis aller Simulationen im Hinblick auf verlorengegangene Datenpakete, Anzahl von Overheads usw. analysiert. Da eine der wichtigsten Eigenschaften von mobilen Systemen typischerweise die Mobilität ist, haben Parameter wie Geschwindigkeit, Richtung und die Änderungsrate der beiden, einen grossen Einfluss auf die Routingprotokolle. In der Realität bewegen sich die Teilnehmer, also Knoten, nach verschiedenartigen Mustern. Um möglichst präzise Aussagen über die System- und Protokollleistung machen zu können, braucht man Bewegungsmodelle, die in möglichst hohem Maße dem Bewegungsverhalten eines echten Teilnehmers ähneln. Die Genauigkeit der Ergebnisse hängt also stark davon ab, wie realitätsnah das gewählte Modell ist. Es ist auch klar, dass Bewegungsmodelle anwendungsabhängig sind. Es sind zum Beispiel bei einem Konferenz-Szenario andere Anforderungen an das Modell zu stellen, als bei einem einzelnen Netzteilnehmer, dessen Bewegungsmuster vollkommen unvorhersehbar ist. Von letzterem geht das zur Evaluierung meistbenutzte Random-Waypoint-Model aus. Dieses Modell sowie das Group-Mobility-Model werden in diesem Kapitel vorgestellt. Es gibt aber auch andere, realistischere Ansätze, wie zum Beispiel das service-orientierte Activity-Based-Model. Hier geht man davon aus, dass der Nutzer des mobilen Systems nicht nach zufälligem Muster umherwandert, sondern eine bestimmte Aufgabe zu erfüllen bestrebt ist. Die Denkansätze entgegen eines realistischen Models im Allgemeinen und das Activity-Based-Model werden am Ende der Ausarbeitung erläutert.

3.1 Random-Waypoint-Mobility-Model

Das zuerst von Johnson und Maltz benutzte Random-Waypoint-Model ist das bislang am häufigsten eingesetzte Modell zur Auswertung von Routingprotokollen. In diesem Modell bewegen sich die Knoten auf einem rechteckigen Simulationsgebiet $[X, Y]^2$. Jeder Knoten hat eine zufällig gewählte Anfangposition (x, y) von der aus er sich mit einer aus dem Intervall $[V_{min}, V_{max}]$ zufällig ausgewählten Geschwindigkeit auf ein zufällig ausgewähltes Ziel (x', y') bewegt. Jedem Knoten wird eine bestimmte Wartezeit zugeordnet, die er am zuletzt erreichten Ziel verbringen muss. Die Pausezeit ist zwischen 0 und $pause_{max}$, der maximalen Pausezeit, normalverteilt. Nach dieser Pausezeit wählt der Knoten das nächste Ziel nach dem oben beschriebenen Schema. Dieses Bewegungsmuster behalten die Knoten während der gesamten Simulationszeit bei.

3.1.1 Nachteile von Random-Waypoint-Mobility-Model

Das Random-Waypoint-Model weißt trotz des häufigsten Einsatzes, einige Schwächen auf. Die Protokollauswertung verlangt realistische Szenarien, die das Random-Waypoint-Model nur bedingt erfüllt, wie es im Folgenden gezeigt wird. Zum einen ist es ein Problem, dass die durchschnittliche Knotengeschwindigkeit V_{avg} des Random-Waypoint-Model im Laufe der Simulation ständig fällt und somit kein realistisches Bewegungsmuster erzeugt wird.

In den Untersuchungen von [JYNo03] stellte man fest, dass diese Tatsache mit der Wahl der minimalen Geschwindigkeit zusammenhängt. Wählt man V_{min}=0, so steigt die Anzahl der Knoten, die lange Strecken mit einer geringen Geschwindigkeit zurücklegen müssen. Eine Vielzahl solcher Knoten bewirkt dann eine kontinuierliche Abnahme der gesamten Durchschnittsgeschwindigkeit V_{avg} des Models, was die Konvergenz der selbigen gegen 0 nach sich zieht [Bild 3]. Eine von [JYNo03] vorgeschlagene Lösung dieses Problems wäre, die Wahl der minimalen Knotengeschwindigkeit auf $V_{min} > 0$ einzugrenzen. So würde die Durchschnittsgeschwindigkeit am Anfang zwar sinken, sich jedoch im Laufe der Simulation stabilisieren.

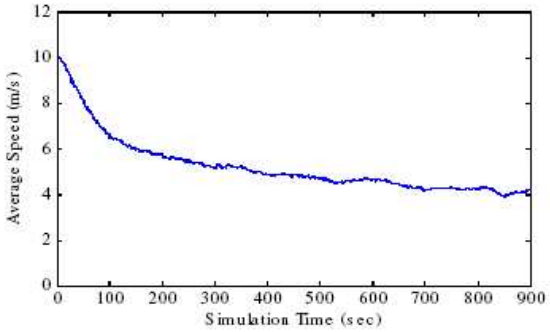


Abbildung 3: $limV_{avg} \rightarrow 0$

Zum anderen taucht bei Random-Waypoint-Model ein Problem der räumlichen Knotenverteilung auf. Man beobachtet, dass bei diesem Model die Aufenthaltswahrscheinlichkeit der Teilnehmer im Zentrum des Simulationsgebiets höher ist als an den Rändern. Es entsteht der sogenannte „Border Effect“ [Bild 4]. Eine höhere Aufenthaltswahrscheinlichkeit bedeutet in der Simulation, dass dort die durchschnittliche Teilnehmeranzahl und damit auch die durchschnittliche Anzahl von Nachbarn größer sind. Dieses Szenario ist in der Realität nicht immer zutreffend. Ein weiteres Beispiel eines Bewegungsmodels, das nicht ganz den Untersuchungsansprüchen genügt, ist das Group-Mobility-Model, das im nächsten Abschnitt vorgestellt wird.

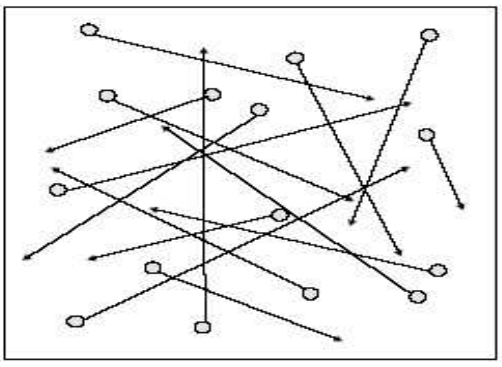


Abbildung 4: „Border Effect“

3.2 Reference-Point-Group-Mobility

Während die Kommunikation in infrastrukturellen Netzwerken auf zwei Knoten ausgelegt ist, findet sie in Ad-hoc-Netzen auch unter Gruppen statt. Oftmals bewegen sich die

Teilnehmer des Ad-hoc-Netzes nicht unabhängig voneinander, vielmehr sind ihre Bewegungen auf bestimmte Weise koordiniert. Zur Simulation solcher Szenarien benötigt man ein Modell, das die Mobilität einer Gruppe von Knoten, und nicht die eines einzelnen Knotens modelliert. Die Mitglieder einer Gruppe wollen meistens ein gemeinsames Ziel erreichen. Ihre Bewegungsmuster sind deshalb aufeinander abgestimmt. Eines der existierenden Gruppenbewegungsmodelle ist das Reference-Point-Group-Mobility-Model (RPGM). Bei diesem Modell werden die in einem geografischen Bereich zufällig verteilten Knoten in Gruppen aufgeteilt. Jede Gruppe hat ein logisches Zentrum, dessen Bewegungsmuster das Bewegungsverhalten der gesamten Gruppe definiert. Der Pfad der Gruppenbewegung wird durch eine Sequenz von sog. check-points festgelegt, die zu Beginn der Simulation zufällig im geografischen Bereich der Gruppe verteilt werden. Die Gruppenbewegung entsteht dadurch, dass das logische Zentrum sich von einem check-point zum nächsten bewegt. Sobald ein check-point vom logischen Zentrum erreicht wird, berechnet es den neuen Richtungsvektor der Gruppenbewegung $\vec{V}_g$ vom momentanen check-point zum nächsten. Jeder Knoten orientiert sich innerhalb der Gruppe an einem ihm zugewiesenen Referenzpunkt [Bild 5]. Der Referenzpunkt für jeden Knoten wird ebenfalls zu Beginn der Simulation relativ zum logischen Zentrum der Gruppe festgelegt und der Bewegung der Gruppe entsprechend verschoben. Im Laufe der Simulation bewegen sich die Knoten beliebig im Umkreis ihres Referenzpunktes. Hierzu berechnet der Knoten nach einem erreichten Ziel seinen nächsten Richtungsvektor $R\vec{M}$, indem er gleichverteilt eine Richtung aus dem Intervall [0,2π] und eine Entfernung zum Referenzpunkt wählt. Das Bewegungsgebiet eines Knotens innerhalb der Gruppe ist durch den maximalen Radius um den Referenzpunkt begrenzt. Die Gesamtbewegung eines Knotens entspricht in diesem Modell der Summe beider Richtungsvektoren $\vec{V}_g$ und $R\vec{M}$, also der Summe der globalen Bewegung der Gruppe und der lokalen Bewegung des jeweiligen Knotens. Das RPGM Modell liefert somit einen allgemeinen Rahmen zur Beschreibung aufgabenorientierter Bewegungsabläufe mehrerer miteinander kollaborierender Teams. Es können verschiedene Gruppenbewegungsszenarien betrachtet werden. Beispiele dafür sind das In-Place-Mobility-Model, bei dem das gesamte betreffende Gebiet in mehrere aneinander angrenzende Bereiche aufgeteilt wird, in denen jeweils eine Gruppe operiert. Das Overlap-Mobility-Model modelliert ein Szenario, bei dem mehrere Gruppen verschiedene Aufgaben im selben geographischen Bereich erledigen müssen. Das Convention-Mobility-Model, modelliert ein Ausstellungsszenario, bei dem alle Gruppen in einer ähnlichen Reihenfolge sich von einem Raum in den nächsten bewegen. Der Unterschied zwischen den Gruppen liegt beispielsweise nur in der Betrachtungsdauer der Ausstellungsstücke bzw. der Zeitdauer des Aufenthaltes in einem bestimmten Raum.

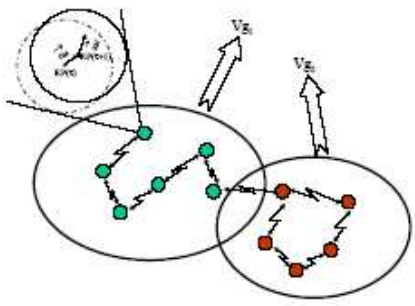


Abbildung 5: Group-Mobility-Model

4 Realistische Bewegungsmodelle

Die Topologie und die Bewegungsmuster der mobilen Teilnehmer eines Ad-hoc-Netzwerks haben bei der Evaluierung einen starken Einfluss auf die Protokollleistung. Sobald die Knoten, wie zum Beispiel in Random-Waypoint-Model, zum Anfang der Simulation verstreut werden, legt das Bewegungsmodell für jeden Knoten fest, wie er sich innerhalb des Netzwerks bewegen soll. Da genau diese Mobilität die Protokollleistung direkt beeinflusst, spiegeln Auswertungsergebnisse, die durch die Simulation mit unrealistischen Modellen erreicht werden, die tatsächliche Protokollleistung nicht korrekt wider. Es haben sich einige Modelle als nicht genügend realistisch behauptet. Sowohl das Random-Walk-Model, als auch das Random-Direction-Model, die hiermit nur kurz angeschnitten werden, gehören zu diesen Kandidaten. Ein weiteres Beispiel ist das im Unterabschnitt 3.1 beschriebene Random-Waypoint-Model. Alle diese Modelle gehen davon aus, dass die mobilen Teilnehmer eines Netzwerks sich ziellos in ihrer Umwelt bewegen. Dies ist keine realistische Ausgangsbasis für die Protokollauswertung. In der realen Welt, wie zum Beispiel auf dem Uni-Campus oder in einem Einkaufscenter, bewegen sich die Teilnehmer keineswegs in zufällig gewählte Richtungen. Die Auswahl der Bewegungsrichtung ist vielmehr abhängig davon, welche Aufgaben der Teilnehmer zu erledigen hat bzw. welches Ziel (Gebäude) er erreichen will. Nicht die Wahl der Richtung ist somit zufällig sondern die Wahl eines Ziels. Ein weiterer unrealistischer Aspekt ist, dass diese Modelle nur auf einem begrenzten, rechteckigen Simulationsgebiet eingesetzt werden. Diese Grenzen werden im Laufe der Simulation nicht überschritten. Das bringt mit sich, dass der Teilnehmer nicht die Möglichkeit hat, sich auf die andere Seite des Simulationsgebietes zu begeben, was in der realen Welt durchaus vorkommen kann. In Simulationsgebieten dieser Modelle wird auch nicht berücksichtigt, dass Hindernisse auftauchen können, die eine Bewegung von einem Ziel zum nächsten beeinflussen können. Diesen Ansatz wählt das im nächsten Abschnitt vorgestellte Obstacle-Mobility-Model. Hier werden Hindernisse und Wege, die der Teilnehmer wählen könnte, um von einem Ziel zum anderen zu gelangen, exakt modelliert. Im darauffolgenden Abschnitt wird ein weiteres Bewegungsmodell vorgestellt, das im Gegensatz zu den oben erwähnten, den Szenarien der wirklichen Welt näher kommt. Das Activity-Based-User-Model für service-orientierte Ad-hoc-Netzwerke, modelliert die Bewegungsmuster der Teilnehmer im Hinblick darauf, dass jeder von ihnen eine bestimmte Aufgabe zu erledigen bestrebt ist. Das charakteristische Merkmal dieses Modells ist, dass die Bewegungen einzelner Knoten und deren Nutzung der Netzwerkressourcen, nicht voneinander unabhängig betrachtet werden.

4.1 Obstacle-Mobility-Model

Es gibt eine Vielzahl von verschiedenen Umgebungen, wie zum Beispiel Städte, Autobahnen oder Schlachtfelder im militärischen Bereich, an denen man bestrebt ist Ad-hoc-Netzwerke einzusetzen. Für jedes dieser Einsatzgebiete ist es charakteristisch, Hindernisse verschiedener Art zu haben, welche die Mobilität der Knoten stören und die Übertragung drahtloser Signale blockieren können. In den meisten bekannten Bewegungsmodellen bewegen sich die Knoten auf einem hindernisfreiem Gebiet nach einem bestimmten, vom Modell festgelegten Bewegungsschema. In der realen Welt ist es selten der Fall, dass sich Menschen in einem von Hindernissen freien Gebiet aufhalten. In einer Stadt findet man typischer Weise Gebäude, Autos, Bäume sowie andere Objekte vor, die einem den Weg zu einem Ziel versperren können. Außerdem ist es unwahrscheinlich, dass Menschen zufällig gewählten Richtungen und Wegen folgen, wovon im Random-Waypoint-Model z.B. ausgegangen wird. Im Campusszenario werden zum erreichen eines Gebäudes meistens die dafür vorgesehenen Wege von einem Gebäude zum anderen benutzt. In der Stadt bedient man sich der Bürgersteige, in den Gebäuden - der Gänge. Während manche von den vorgesehenen

Wegen abweichen, ist die Aufenthaltswahrscheinlichkeit der Knoten entlang dieser am höchsten. Das Obstacle-Mobility-Model berücksichtigt genau diese Faktoren. Anhand dieses Modells ist es in erster Linie möglich, verschiedene realistische Szenarien zu erzeugen. Dazu können Positionen, Formen und Größen der im Simulationsareal zu platzierenden Objekte definiert und somit ein hindernisreiches Terrain kreiert werden. Allein die Platzierung der Objekte im Simulationsgebiet ist allerdings nicht ausreichend, um ein realitätsnahes Bewegungsverhalten der Knoten zu erzeugen. Dürften z.B. die Knoten zur Fortbewegung zufällige Wege innerhalb des Simulationsgebietes wählen, würde ein dem auf [Bild 6] ähnliches Bewegungsmuster entstehen. Die Knoten würden beim ersten Antreffen auf ein Hindernis eine andere Bewegungsrichtung wählen. In manchen Fällen trifft dies auch auf die reale Welt zu, meistens haben aber Menschen die Intention, ein Gebäude zu betreten, wenn sie sich gezielt auf dieses Gebäude zu bewegen. Um von einem Gebäude zum anderen zu gelangen oder ein Gebäude zu betreten, benutzt man meistens vorhandene Wege, die bei diesem Modell mit Hilfe von Voronoi-Diagrammen berechnet werden. Als Eingabeparameter nimmt die Voronoi-Wege-Berechnung die Koordinaten des Objektes, und berechnet die Voronoi-Regionen für das Simulationsgebiet. Meistens ist es der Fall, dass ein Weg zwischen zwei Gebäuden einen näherungsweise gleichen Abstand zu beiden besitzt. Daher werden die Voronoi-Regionen um die Ecken der Objekte berechnet. Die Kanten des so entstanden ungerichteten Bewegungsgraphs dienen nun als Wege für die Knotenbewegung, die jeweils durch die Euklidische Länge gewichtet sind. Diese Wege werden zu Beginn der Simulation berechnet und ändern sich im Verlauf nicht. Die mobilen Knoten werden ebenfalls zu Beginn der Simulation zufällig, entlang der Wege im Simulationsgebiet verteilt. Die Knoten bewegen sich, indem sie sich ein zufälliges Ziel, wie einen Gebäudeeingang, aussuchen und dieses Ziel dann mittels der vorhandenen Wege erreichen. Jeder mobile Knoten wählt den Weg zu seinem Ziel nicht zufällig, sondern berechnet die von seiner Ausgangsposition kürzeste Strecke. Sobald der Knoten sein Ziel erreicht hat, macht er eine Pause. Dann wählt er ein neues Ziel aus und wiederholt das beschriebene Bewegungsschema. (Im [Bild 7] kann man leicht den Vorteil der vordefinierten Wege ersehen.) In der Realität ist es oft der Fall, dass der kürzeste Weg zu einem Ziel durch ein Gebäude hindurchführt. Durch die Aufteilung des Gebietes in Voronoi-Regionen führen Wege auch durch die Objekte durch, sodass dieser Fall auch abgedeckt ist, wie es im [Bild 7] zu erkennen ist.

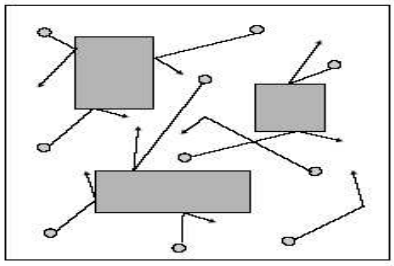


Abbildung 6: Bewegungsverhalten der Knoten bei zufälliger Wegwahl

4.1.1 Konstruktion von Hindernissen und Wegen

Manche Hindernisse in der wirklichen Welt haben eine Auswirkung auf die Ausbreitung drahtloser Signale. Die Qualität der Übertragung hängt in den meisten Fällen von der Bauweise der Objekte und von dem dazu benutzten Baumaterial bzw. der Dicke der Wände eines Objektes ab. Deshalb muss man damit rechnen, dass die Übertragung durch manche Objekte blockiert wird. Um für den Anfang die Komplexität des Modells zu minimieren, werden in diesem Modell alle vorkommenden Hindernisse als übertragungsstörend betrachtet. Die Hindernisse werden mit Hilfe der Polygone konstruiert, sodass die Angabe genauer

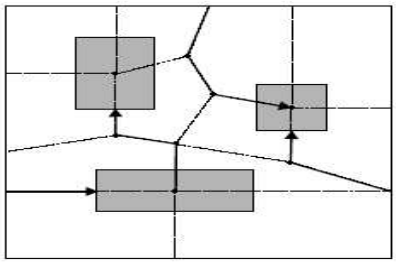


Abbildung 7: Simulatonsgebiet mit Hindernissen

Koordinaten der Objekte möglich ist. Rechteckige Polygone bieten sich am besten an, da sie zur Modellierung einer Vielzahl verschiedener Gebäudeformen geeignet sind, wie z.B. *U*- oder *L*-Form eines Gebäudes. Kreisförmige Formen können ebenfalls mit Hilfe der Polygone approximiert werden. In der Konstruktion der Hindernisse wird auch berücksichtigt, dass jedes Objekt einen oder mehrere Eingänge besitzt, sodass die Knoten das Gebäude betreten und verlassen können. Die Formen der Objekte sowie deren Platzierung im Simulationsareal haben eine starke Auswirkung auf die Mobilität der Knoten sowie ihre Zusammenarbeit im Netzwerk.

Wie die Wege verlegt werden hängt in erster Linie von der Platzierung der Gebäude ab. Die Benutzung der Voronoi-Diagramme zu Konstruktion der Wege ist sehr geeignet, da man intuitiv davon ausgeht, dass Wege zwischen zwei angrenzenden Objekte meistens in der Mitte der beiden verlaufen. Durch die Konstruktion der Voronoi-Regionen um die Eckpunkte der Gebäude, wird genau dieses erreicht. Im [Bild 7] ist dies ersichtlich. Hier werden für zwei Objekte die Wege um die Objekte berechnet. Zur Konstruktion der Voronoi-Pfade benutzt man hier die acht Ecken der Objekte. Wie man ersehen kann, wird jeder Ecke eine Region zugeordnet. Die Kanten dienen nun als Wege für die Knoten. Die Schnittpunkte der Objekte mit den Graphenkanten fungieren als Ein- und Ausgänge.

4.1.2 Datenübertragung

In diesem Modell wird angenommen, dass die Objekte so robust sind, dass die Datenübertragung durch ihre Wände verhindert wird. Um die Objekte entstehen dadurch Gebiete, an denen die Datenübertragung gestört ist. Diese Störungsgebiete befinden sich relativ zu den Ausgangspositionen mobiler Knoten, wie es im [Bild 8] dargestellt ist. Der $Knoten_i$ wird bei Bedarf keine Verbindung zu einem in seinem Störungsgebiet, begrenzt durch die Punkte $i1, i2, i3$ und $i4$, liegenden Knoten aufbauen können. Das gleiche gilt auch für den $Knoten_j$, dessen Störungsgebiet im Bereich der Punkte $j1, j2, j3$, und $j4$ liegt. Die Kommunikation zwischen den $Knoten_i$ und $Knoten_j$, würde dagegen einwandfrei funktionieren. D.h. die Signalübertragung zwischen zwei Knoten kommt nur dann zustande, wenn der $Knoten_k$ im Sichtbereich des $Knoten_i$ liegt. Die $Knoten_k$, die im Störungsbereich des $Knoten_i$ liegen und daher von ihm nicht erreicht werden können, werden durch folgende Funktion dargestellt:

$$OS(Knoten_i) = \{Knoten_k \mid k \text{ ist nicht im Sichtbereich von } i\}.$$

Diese Funktion ist verständlicher Weise symmetrisch, denn:

$$\text{Wenn } Knoten_i \in OS(Knoten_k), \text{ dann } Knoten_k \in OS(Knoten_i).$$

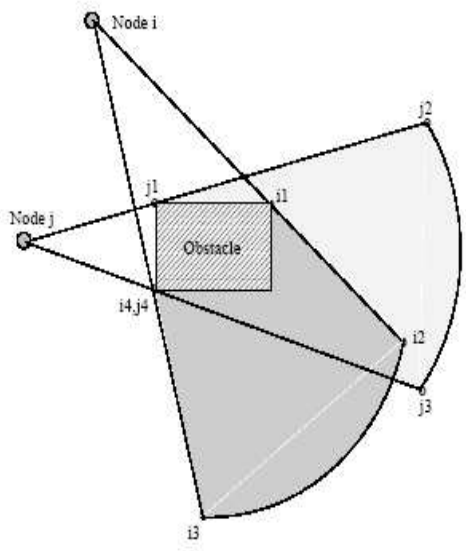


Abbildung 8: Störungsbereiche von Knoten i und j

Es sind zwei Fälle, die betrachtet werden müssen, in denen es zu Verbindungsstörungen kommen kann. Zum einen ist es der oben beschriebene Fall, zum anderen kann es in diesem Modell, wie auch in der Realität, vorkommen, dass sich ein Knoten innerhalb eines Objektes der andere sich jedoch außerhalb dessen befindet. Um die momentane Position des jeweiligen Knotens zu berücksichtigen wird eine weitere Funktion definiert. Hierbei steht k für die Nummer eines sich im Simulationsareal befindenden Objektes und tag für die Funktion, die momentane Position des Knoten beschreibt.

$if\ k \geq 1,\ tag(Knoten_i) = k,$
else $tag(Knote_i) = 0$
mit $k \in \{1, ...AnzahlderObjekte\}$ und
für k=0 gilt, dass der Knoten sich momentan außerhalb eines Objektes befindet.

Um im Laufe der Simulation festzustellen ob ein Objekt auf die Signalübertragung zwischen zwei Knoten einen Einfluss hat, bietet sich die sog. Erreichbarkeits-Matrix sehr gut an [Bild 9]. Zeilen und Spalten dieser Matrix repräsentieren zwei kommunizierende Knoten. 1 bedeutet vollkommene Erreichbarkeit und 0 , eine blockierte Signalübertragung.

4.1.3 Auswertung von AODV mit Obstacle-Mobility-Model und Random-Waypoint-Mobility-Model

Das Obstacle-Mobility-Model wurde an der kalifornischen Universität von Santa Barbara zur Auswertung des AODV Protokolls eingesetzt. Die Intention war in erster Linie zu erfahren, wie sich das Vorhandensein der Gebäude im Simulationsgebiet auf die Datenübertragung auswirkt. Hierzu untersuchte man anhand dieses Modells die Dichte der Knotenverteilung im Simulationsgebiet sowie die Leistung des AODV Routing-Protokolls. Der Vergleich der Ergebnisse mit den Ergebnissen des Random-Waypoint-Mobility-Models zeigt deutliche Unterschiede auf.

4.1.4 Simulation

Zu Beginn der Simulation mit „Obstacle-Mobility-Model“ werden die Knoten an den Punkten $\{s1, s2, s3, s4, s5, s6, s7\}$ der Grenzkanten des Voronoi-Graphen, wie im [Bild 10] platziert und

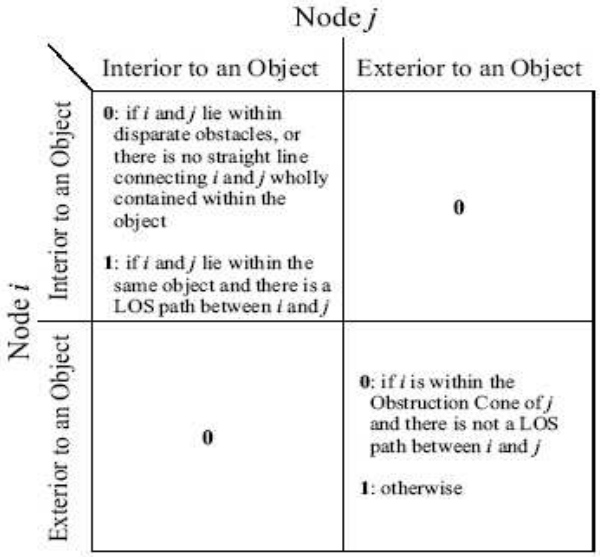


Abbildung 9: Erreichbarkeits-Matrix

bewegen sich die ersten 60 Sekunden mit der Geschwindigkeit zwischen 0 und 5 m/s entlang der vordefinierten Wege, so dass sie sich zufällig im Simulationsgebiet verteilen. Erst nach dieser Zeit werden die Ergebnisse der Simulation gespeichert. Die Auswertung mit Random-Waypoint-Model wird zu Vergleichszwecken in der gleichen Umgebung durchgeführt. Hier werden, wie das RWP festlegt, die Knoten zufällig im Simulationsbereich verteilt und die Hindernisse beseitigt. Der Vergleich beider Ergebnisse weist folgende Unterschiede bezüglich der Knotendichte sowie der Leistung des AODV auf.

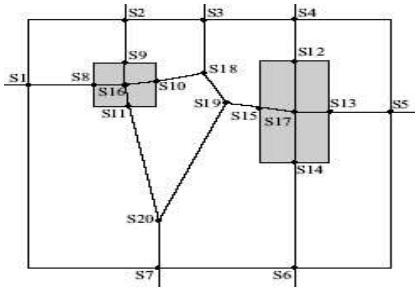


Abbildung 10: Beispiel eines Simulationsareals mit Voronoi-Wegen

4.1.5 Ergebnisse der Simulation

Die mit dem Obstacle-Model erreichte Knotendichte ist im Vergleich zu Random-Waypoint-Model beträchtlich kleiner. D.h., dass die durchschnittliche Anzahl der Nachbarknoten für jeden Knoten um einiges geringer ist als bei der Simulation mit dem Random-Waypoint-Model. Die Erklärung dafür liefern folgende zwei Gründe. Zum einen unterscheidet das Obstacle-Model zwischen internen und externen mobilen Knoten. Das zieht nach sich, dass während der Simulation ein externer Knoten einen anderen Knoten, der sich innerhalb eines Objektes befindet, nicht als Nachbar erkennt und umgekehrt. Die Knoten, die sich innerhalb eines Gebäudes bewegen, haben nur die Knoten zu Nachbarn, die ebenfalls innerhalb desselben Gebäudes sind und in ihrem Sichtbereich liegen. Das gleiche gilt für die Knoten, die sich außerhalb befinden. Diese erkennen nur die externen Nachbarknoten. Zum anderen wird die Signalübertragung bei Obstacle-Model durch die in der Umgebung liegenden Objekte gestört. Das führt dazu, dass die mobilen Knoten nur die Knoten als Nachbarn

erkennen, die sich in ihrem Sichtbereich aufhalten. Die Übertragungsreichweite eines jeden Knotens ist somit, im Gegensatz zum Random-Waypoint-Model, begrenzt, was die Anzahl der Nachbarn im Umkreis eines Knotens vermindert. Die Ergebnisse der Protokollauswertung mit Random-Waypoint-Model und Obstacle-Model zeigen auch große Unterschiede bezüglich der Datenübertragung. Da bei RWP keine Hindernisse auftauchen, ist die Wahrscheinlichkeit, dass eine Route zwischen dem Sendeknoten und Empfängerknoten nicht gefunden wird, vergleichsweise klein. Der Datenverlust ist demzufolge, im Gegensatz zum Obstacle-Model, gering. Beim Obstacle-Model ist es der Fall, dass sobald eine Route als nicht vorhanden eingeschätzt wird, was wegen der vorhandenen Hindernisse nicht selten vorkommt, die Datenübertragung zwischen der Quelle und dem Empfängerknoten abgebrochen wird. Das Paket wird nicht geliefert und geht verloren. Dies ist z.B. der Fall, wenn eine Verbindung zwischen einem internen und einem externen Knoten verlangt wird. Die zahlreichen Abbrüche der Verbindungen bewirken, dass beim Obstacle-Model die Datenverlustrate höher ist als bei RWP. Diese Eigenschaft hat natürlich eine Auswirkung auf die Datenübertragungszeit. Aufgrund der wenigen Daten verringert sich die Belastung des Netzwerkes, was zu kleineren Verzögerungszeiten führt.

Die hier vorgestellten Ergebnisse können mit Hilfe zweier Mechanismen verbessert werden, woran auch im Moment gearbeitet wird. Als erstes muss die Tatsache umgesetzt werden, dass nicht alle Gebäude die Signalübertragung blockieren. Also müssen in manchen Fällen interne und externe Knoten auch im Modell die Möglichkeit haben, eine Verbindung aufzubauen. Des weiteren muss die Quelle , bei nicht erfolgreichem Aufbau der Verbindung, immer wieder versuchen, die Daten zu übermitteln. Wenn die Route, aufgrund der sich durch die Mobilität verändernden Netztopologie, verfügbar wird, können die Daten übertragen werden. Es sind viele weitere Möglichkeiten offen, dieses Modell realistischer zu gestalten. Wie z.B. die Zielauswahl jedes Knotens auf einen beschränktes Gebiet zu reduzieren, da Menschen sich meistens zwischen Gebäuden bewegen, die nah beieinander platziert sind; oder die Zielwahl eines Knotens von einer bestimmten Funktion abhängig zu machen. Zum Beispiel wird zu Mittagszeit die Mensa auf dem Uni-Campus das mit Sicherheit am wahrscheinlichsten gewähltes Ziel der Studenten sein. Einen ähnlichen Denkansatz verfolgt auch das im Folgenden vorgestellte Activity-Based-User-Model.

4.2 Activity-Based-User-Model

Während die meisten Modelle ein zufälliges Bewegungsmuster der mobilen Teilnehmer (Netzwerknutzer) modellieren, berücksichtigen die Entwickler des Activity-Based-Models die Tatsache, dass Menschen in der Regel nicht ziellos umherwandern. Ihr Bewegungsverhalten hängt vielmehr davon ab, welche Aufgaben sie zu erledigen bestrebt sind. Das impliziert aber auch, dass die Mobilität der Teilnehmer und deren Nutzung der Netzwerkressourcen unmittelbar zusammenhängen. Ein Teilnehmer wird zum Beispiel mit großer Wahrscheinlichkeit kein Dokument bearbeiten, während er seine Emails abruft. Dieser Zusammenhang wird von den momentan vorhandenen Modellen nicht berücksichtigt, vielmehr werden diese zwei Komponenten voneinander getrennt betracht. Das Activity-Based-User-Model modelliert Teilnehmer eines service-orientierten Ad-hoc-Netzwerkes. Das Charakteristikum dieser Netzwerke ist, dass hier mobile Dienste angeboten und Daten ausgetauscht werden. Bei der Modellierung muss also berücksichtigt werden, dass Teilnehmer und Institutionen miteinander kooperieren. In den sog. analytischen Bewegungsmodellen, wie das oben beschriebene Random-Waypoint-Model, wird nur das Bewegungsverhalten der Teilnehmer innerhalb eines Simulationsareals festgelegt, vollkommen unbeachtet bleiben aber die Zusammenhänge ihrer Mobilität und der Nutzung der Netzwerkressourcen. Daher spiegeln diese Modelle nicht das tatsächliche Benutzerverhalten in Ad-hoc-Netzwerken wider. Das

Activity-Based-User-Modeling ist dagegen ein Verfahren, das die Mobilität der Teilnehmer in Abhängigkeit von verschiedenen Komponenten untersucht [Bild 11].

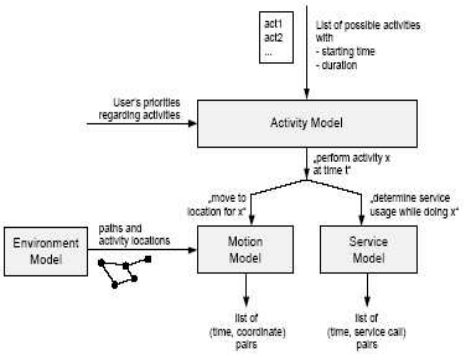


Abbildung 11: Die vier Bestandteile des Activity-Based-User-Models

Um ein möglichst realistisches Benutzerverhalten zu erzeugen, werden hier vier Modelle unterschieden, die gemeinsam das Activity-Based-User-Model bilden. Zum einen ist es das Activity-Model, das bestimmt, welche Tätigkeit ein Teilnehmer des Netzwerks gerade ausführt. Des weiteren, wird die Wegberechnung vom Motion-Model übernommen. Das Service-Model erkennt den Dienst, den ein Teilnehmer in der momentanen Situation auswählen wird. Schließlich liefert das Environment-Model die in dem Simulationsareal vorhandenen Wege und Lokalitäten. Im Folgenden werden diese Modelle näher vorgestellt.

4.2.1 Activity-Model

Mit Hilfe des Activity-Models wird die momentane, netzwerk-unabhängige Tätigkeit des Teilnehmers festgelegt. Jeder Teilnehmer erledigt im Alltag eine Vielzahl von Aufgaben. Dazu gehören zum Beispiel im Uni-Campus-Szenario der Besuch einer Vorlesung oder das Mittagessen in der Mensa. Diese Tätigkeiten werden zu einem bestimmten Zeitpunkt in einer bestimmten Zeitspanne, an einer bestimmten, von der Tätigkeit abhängigen, Lokalität und mit einer bestimmten Priorität erledigt. Die Aufgabe dieses Modells besteht nun darin, einen Zeitplan zu erstellen, nach dem der Benutzer diese Tätigkeiten ausführt. Als erstes werden die für ein bestimmtes Szenario typische Aufgaben festgelegt. Die vom Ablauf her ähnliche Tätigkeiten, wie Besuch der Vorlesung und Seminarbesprechung, werden zu Klassen zusammengefasst. Als nächstes wird der Zeitpunkt der Ausführungen festgelegt. Man unterscheidet zwischen den Tätigkeiten, die immer zu einem bestimmten Zeitpunkt und denen, die unregelmäßig ausgeführt werden. Die Zeitspanne, in der die Aufgaben erledigt werden, ist wie auch bei der Planung im täglichen Leben, ebenfalls wichtig. Da es bei der Erstellung des Zeitplanes oft zur Überlappung verschiedener Aufgaben kommt, wird jede von ihnen mit einer Priorität gewichtet. Die Priorität eines Vorlesungsbesuches ist i.A. höher als das Ausleihen eines Buches. Der anhand dieser Faktoren für einen Benutzer erstellte Zeitplan, ist nun die Basis für die weiteren drei Modelle.

4.2.2 Environment-Model

Mit Hilfe eines ungerichteten Multigraphen wird im Environment-Model festgelegt, welche Wege und Lokalitäten im Simulationsgebiet existieren [Bild 12]. Die Knoten des Graphen befinden sich dort, wo mehrere Wege zusammenkommen; die Kanten stellen, die in der Umgebung vorhandenen Wege dar. Die Anzahl der in eine Richtung laufenden Wege deutet auf die Breite des Weges hin. Je breiter der Weg, desto höhere Mobilität ist hier zu erwarten.

Das Environment-Model beschreibt außerdem die in der Umgebung liegenden Lokalitäten, wie z.B. die Bibliothek oder Mensa im Campus-Szenario. Diese werden als rechteckige Objekte, wie im [Bild 12] zu sehen ist, dargestellt.

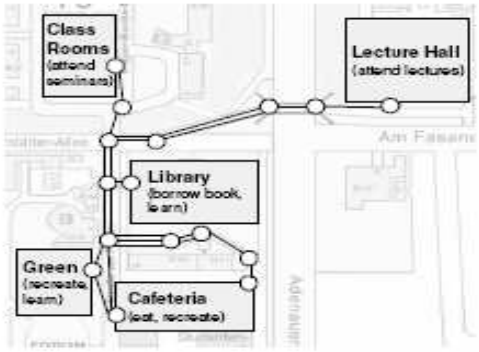


Abbildung 12: Environment-Model

Typischerweise wird in jeder dieser Lokalitäten eine für sie charakteristische Tätigkeit ausgeübt. In der Mensa befindet man sich meistens, um ein Mittagessen zu sich zu nehmen, in der Bibliothek, um ein Buch zu finden und/oder auszuleihen. Das Bewegungsverhalten des Nutzers ist je nach Lokalität in der er sich befindet, verschieden, was ebenfalls zu berücksichtigen ist. Der Prozess des „Essenholens“ und anschließende Bewegung zum Tisch ist z.B. ein eigenes Mobilitätsprofil für die Lokalität Mensa. In einem Simulationsgebiet bewegt sich der Nutzer nun nicht beliebig sondern muss sich an die vom Environment-Model vorgeschriebene Einschränkungen, wie Gebäude, Routen etc. halten. Während der Simulation gilt im allgemeinen, dass wenn ein Nutzer eine Lokalität erreicht, er dort seine Aufgabe ausführt.

4.2.3 Motion-Model

Die Aufgabe vom Motion-Model besteht darin, den kürzesten Weg zum gewünschten Ziel eines Nutzers zu berechnen. Hierzu wird als erstes das Ziel selbst festgestellt. Die Informationen über die gewünschte Lokalität zur Ausführung einer bestimmten Tätigkeit bezieht das Motion-Model vom Environment-Model. Je nach Tätigkeit stellt das Environment-Model alle möglichen Lokalitäten zur Verfügung, woraus im allgemeinen die nächstgelegene ausgewählt wird. Als nächstes berechnet das Motion-Model die möglichen Wege zu diesem Ziel. Dazu legt es die Reihenfolge der Graphenknoten des Environment-Models fest, die bis zum Ziel besucht werden müssen. Da der Nutzer meistens versucht den kürzesten Weg auszuwählen, wird dieses Kriterium auch hier benutzt. Anschließend wird die Bewegung des Nutzers entlang einer der berechneten Route ausgeführt. Der günstigste Weg sowie die Geschwindigkeit des Nutzers wird im Hinblick auf die Höhe der vorhandenen Mobilität auf allen berechneten Wegen festgelegt. Sinnvoll ist dieses Vorgehen deshalb, weil die Wege durch andere Nutzer überfüllt werden können. Hier muss der Nutzer die Möglichkeit haben, auf parallel laufende Wege zu wechseln, um sein Ziel so schnell wie möglich zu erreichen. Durch die überfüllten Wege wird der Nutzer ebenfalls seine Geschwindigkeit reduzieren müssen. Also wird sie erst nach der Auswahl des jeweiligen Weges festgelegt.

4.2.4 Service-Model

Das Service-Model liefert die Art des Netzwerkdienstes, den ein Nutzer während seiner momentanen Tätigkeit am wahrscheinlichsten nutzen wird. Z.B. während der Vorbereitung

auf eine Klausur wird er eher eine Datei herunterladen als ein Chatportal nutzen. Die Information über den momentanen Zustand und die Beschäftigung des Nutzers erhält das Service-Model vom Activity-Model. Ein weiterer Aspekt, von dem die Dienstwahl abhängt, ist welche Aufmerksamkeit der Nutzer seiner momentanen Aufgabe widmet. Es ist einleuchtend, dass bei schneller Bewegung ein Teilnehmer die Nutzung der Dienste einschränkt, da es an höherer Aufmerksamkeit verlangt.

4.2.5 Zusammenfassung

Das hier beschriebene Verfahren bietet eine Alternative zum Modellieren eines realistischen Bewegungsmusters eines mobilen Teilnehmers. Die meisten vorhandenen Modelle nutzen die Informationen über die Aktivität des Teilnehmers im Netzwerk nicht aus, sondern modellieren das Bewegen als einen Selbstzweck. Beim Activity-Based-User-Modeling werden die Art der Beschäftigung, die Nutzung eines Dienstes sowie die Mobilität des Nutzers im Zusammenhang betrachtet. Diese Vorgehensweise bietet eine wichtige Basis für die Entwicklung realistischer Bewegungsmodelle.

5 Schlussfolgerung

In dieser Ausarbeitung wurden einige heute existierende Bewegungsmodelle vorgestellt. Die Notwendigkeit dieser Modelle ist unbestritten, da Bedingungen der realen Welt Protokollentwicklern Wiederholungen oder Änderungen eines Szenarios im Rahmen der Protokolloptimierung nicht bieten können. Es wurde gezeigt, dass die Entwicklung der Bewegungsmodelle eine wichtige Aufgabe im Bereich der Ad-hoc-Netzwerke darstellt. Die meisten Modelle erzeugen jedoch ein nicht ausreichend realistisches Bewegungsverhalten der mobilen Teilnehmer, so dass die Ergebnisse der Protokollauswertung oft nicht die tatsächliche Protokollleistung wiedergeben. Die Entwicklung besserer Modelle ist notwendig, wozu das Acivity-Based-User-Modeling einen Ansatz bildet.

Literatur

[JYNo03] M. Liu J. Yoon und B. Noble. Random Waypoint Considered Harmful. University of Michigan, 2003.

Abbildungsverzeichnis

1 Änderung der Netztopologie 90
2 Routing in Ad-hoc-Netzen 91
3 $limV_{avg} \rightarrow 0$ 93
4 „Border Effect“ 93
5 Group-Mobility-Model 94
6 Bewegungsverhalten der Knoten bei zufälliger Wegwahl 96
7 Simulatonsgebiet mit Hindernissen 97
8 Störungsbereiche von Knoten i und j 98
9 Erreichbarkeits-Matrix 99
10 Beispiel eines Simulationsareals mit Voronoi-Wegen 99
11 Die vier Bestandteile des Activity-Based-User-Models 101
12 Environment-Model 102

Ansätze für drahtlose Endsysteme-basierte Gruppenkommunikation

Andreas Süß

1 Einführung

1.1 Funktionsweise von Overlay-Netzen

Overlay-Netze sind virtuelle Netze, die physikalischen Netzen überlagert sind. Sie sind grundsätzlich anders als physikalische Netzwerke, denn sie haben die Freiheit sich selbst eine Topologie zu geben. Bildet ein Protokoll einer höheren Schicht eines Kommunikationssystems Funktionalitäten tieferer Schichten so nach, daß die daran beteiligten Netzwerkknoten Knoten in einem neuen, virtuellen Netz werden, so spricht man von einem Overlay-Protokoll.
Die beiden typischen Funktionen tieferer Schichten, die dabei auf einer höheren Schicht (oft der Anwendungsschicht) nachgebildet werden, sind Routing und Adressierung. Die beiden dazu häufig angewandten Mechanismen sind das Tunneln von Datagrammen durch eine Verbindung der Transportschicht oder das Kapseln von Datagrammen auf der Vermittlungsschicht. Das Auftreten eines dieser Mechanismen ist ein Anzeichen dafür, daß es sich in einem System um ein Overlay- Netz handelt. Beispiel für ein Overlay-Netz ist Gnutella.

1.2 Endsystem-Multicast

Endsystem-Multicast Protokolle erstellen ein Overlay-Netz oberhalb der physikalischen Netzwerkstruktur. Das Overlay-Netz enthält nur die Knoten, die an der Multicastgruppe beteiligt sind. Allein diese Mitglieder sind verantwortlich für das Multicast-Routing und die Paketduplizierung innerhalb des Overlays. Alle anderen Knoten müssen nur ein Unicast-Routing Protokoll unterstützen. Das Prinzip ist in Abbildung 1 dargestellt.

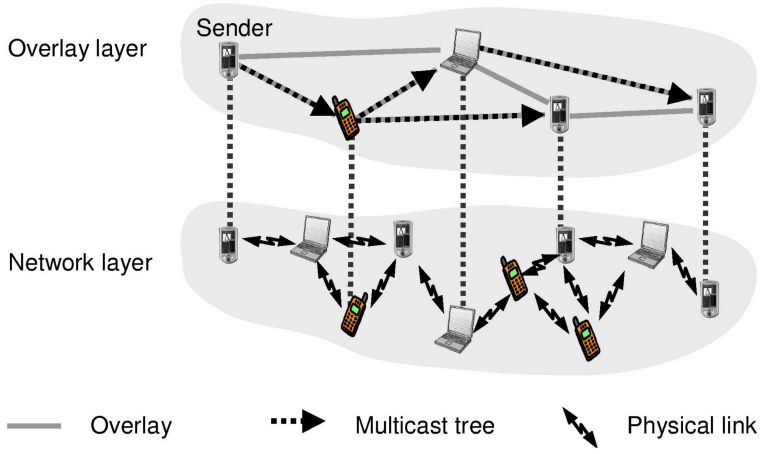


Abb. 1: Prinzip von Endsystem-Multicast

Existierende für das Internet entwickelte Ansätze, können in zwei Kategorien klassifiziert werden: unstrukturierte und strukturierte Overlays. Protokolle die unstrukturierte Overlays verwenden, wie z.B. Narada und Gossamer, werden auch als sogenannte *mesh-first* Ansätze bezeichnet, da sie ein robustes Mesh als Overlay-Netz erstellen. Ein separater Algorithmus ist in jedem Knoten verantwortlich für das Multicast-Routing. Strukturierte Overlays können desweiteren unterteilt werden in *tree-first* Ansätze und *implizite* Ansätze. Tree-first Ansätze, wie z.B. Overcast und ALMI, erstellen unmittelbar einen Multicast-Baum. Dieser Ansatz hat den Nachteil, daß der Baum nicht sehr stabil und stark anfällig für Knotenausfälle ist. Implizite Ansätze, wie z.B. NICE, Hypercast, CAN und Bayeux, erzeugen eine spezielle Struktur, die zur Verteilung der Multicast-Daten verwendet wird. Der Vorteil dieser impliziten Ansätze ist, daß die Struktur schon die Datenweitergabe vorschreibt und somit kein Multicast-Routing Algorithmus im Overlay benötigt wird.
Viele existierende Multicast Routing Protokolle für MANETs haben das Problem, daß sie auf der Vermittlungsschicht operieren. Dies hat zur Folge, daß alle Netzwerkknoten dasselbe Multicast-Routing Protokoll unterstützen müssen, und somit für das Multicast-Routing sowie für die Speicherung von Zustandsinformationen über die Multicastgruppe verantwortlich sind. Ineffizient wird das ganze dadurch, daß auch die Netzwerkknoten, die nicht der Multicastgruppe zugehörig sind, den Multicast-Verkehr weiterleiten und Zustandsinformationen über die Multicastgruppe speichern müssen. Falls Knoten der Multicastgruppe beitreten oder sie verlassen, müssen die Zustandsinformationen ebenfalls in allen Knoten aktualisiert werden.
Allgemein ist zu sagen, daß Endsystem-Multicast Ansätze sehr vielversprechend sind, da sie mit MANETs sehr viel gemeinsam haben. Beide sind dezentralisiert, selbst-organisierend und alle Knoten sind Client, Server und Router gleichzeitig.

1.3 Mobile Ad-Hoc Netzwerke

Netze aus Knoten, die sich einfach zufällig in Übertragungsreichweite voneinander befinden können, werden als *Ad-Hoc-Netze* oder *MANETs (Mobile Ad-Hoc Networks)* bezeichnet. Dabei kommunizieren die Knoten, ohne eine Basisstation, direkt miteinander und sind Router und Host gleichzeitig.
Ad-Hoc-Netze unterscheiden sich von Festnetzen darin, dass sie keine feste Topologie mit festen und bekannten Nachbarn, sowie festen Standorten haben. Knoten können kommen und gehn oder sich an andere Orte bewegen. Der Pfad für die Verteilung von Daten, eines Knotens an ein Ziel, bleibt in Festnetzen unbegrenzt gültig (vorbehaltlich eines Systemausfalls). Bei einem Ad-Hoc-Netzwerk kann sich die Topolgie ständig verändern, so dass sich der angestrebte Pfad sowie die Gültigkeit von Pfaden spontan ohne Vorwarnung ändern können.
Der wohl herausragendste Unterschied zwischen Festnetzen und MANETs stellt das zum Einsatz kommende Medium und dessen Teilbarkeit dar. In Festnetzen kommen einzelne Punkt-zu-Punkt-Verbindungen zum Einsatz, bei denen sich lediglich zwei Kommunikationspartner das Medium teilen. Durch den Einsatz moderner Verfahren können somit Datenkollisionen weitgehend vermieden werden. In MANETs hingegen steht lediglich das geteilte und drahtlose Medium zur Verfügung, woraus sich ständig Kollisionen mit darausfolgenden Datenverlusten ergeben. Spezielle Techniken machen es möglich diese Kollisionen zu erkennen und die Daten erneut zu übertragen. Es müssen deshalb Richtlinien vereinbart werden, um den Medienzugriff und die erneute Datenübertragung zu kontollieren.

1.4 Grundsätzliche Probleme bisheriger Ansätze

Nachteilig ist, daß die meisten bisher existierenden Ansätze für das traditionelle Internet entwickelt wurden und nicht die Eigenheiten von MANETs (Mobile Ad-Hoc Networks)

berücksichtigen. MANETs schaffen durch häufige Verbindungs- und Knotenausfälle eine wesentlich dynamischere Netzwerktopologie. Aufgrund des geteilten Mediums ist mit weitaus mehr Paketverlusten zu rechnen, als bei den im statischen Internet verwendeten Punkt-zu-Punkt-Verbindungen. Das Overlay sollte deshalb ständig auf die unterliegende Netzwerkstruktur abgestimmt werden und sich dessen Topologie bewusst sein. Es ist jedoch sehr schwer zu verhindern, daß bei der Erstellung des Overlay-Netzes sich verschiedene Unicasttunnel ein und dieselbe Verbindung des zugrunde liegenden physikalischen Netzes teilen, was zu redundantem Verkehr in diesem führt.
In MANETs gibt es zwei wichtige Gründe für dynamische Netzwerkstrukturen: Mobilität und dynamische Gruppenmitgliedschaft. Die meisten Endsystem-Multicast Protokolle können jedoch nur mit dynamischer Gruppenmitgliedschaft umgehen, weil dieses Verhalten auch im statischen Internet gegeben ist. Jedoch können sich in MANETs die Knoten auch bewegen und das Overlay muss somit ständig an die neue Netzwerkstruktur angepasst werden. Abbildung 2 zeigt die Auswirkungen auf das Overlay wenn sich die Knoten bewegen.

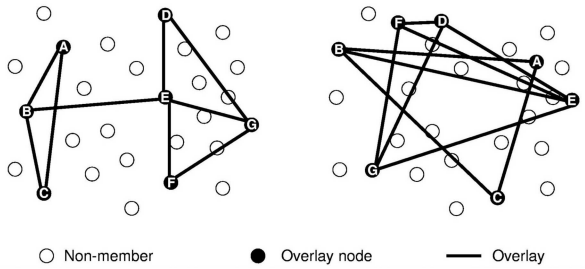


Abb. 2: Auswirkung sich bewegender Knoten auf das Overlay

2 Ansätze

In den folgenden Abschnitten werden zwei Ansätze vorgestellt, welche die Eigenheiten von MANETs berücksichtigen.

2.1 NICE-MAN

In diesem Abschnitt wird zuerst auf das als Basis für NICE-MAN dienende Festnetzprotokoll NICE eingegangen. Daraufhin werden die Verbesserungen an NICE beschrieben, die es an die dynamische Netzwerkarchitektur, Verbindungsausfälle und weitere typische Eigenheiten von MANETs anpassen.
Das verbesserte NICE Protokoll wird NICE-MAN (NICE for Mobile Ad-Hoc Networks) Protokoll genannt. NICE-MAN erreicht seine Effizienz in MANETs durch Ausnutzung der Broadcast-Fähigkeit des Mediums, permanente Anpassung des Overlays an die dynamische Netzwerktopologie und eine auf der Anzahl der Übertragungsschritte basierenden Abstandsmetrik.

2.1.1 NICE

NICE ist ein skalierbares Endsystem-Multicast Protokoll, welches eine mehrschichtige Hierarchie vollvermaschter Cluster erstellt. Seine spezielle Struktur hat den Vorteil, daß quellenspezifische Datenbäume indirekt durch das gemeinsame hierarchische Overlay-Netz definiert sind, und somit kein weiterer Multicast-Routing Algorithmus im Overlay benötigt wird. Jedes Mitglied leitet die Multicast-Nachrichten an alle Mitglieder seiner Cluster

weiter, bis auf die Cluster denen der Sender der Nachricht selbst angehört. Ein weitere Vorteil von NICE ist, daß es aufgrund seiner vollvermaschten Clusterhierarchie sowohl stabil gegen Knotenausfälle ist, als auch geringeren Kontrolloverhead erzeugt. Um die von NICE durchgeführte Aufteilung nah beieinanderliegender Mitglieder in Cluster zu erreichen, wird als Abstandsmetrik eine Ende-zu-Ende Latenz verwendet. Die Größe jedes einzelnen Clusters liegt zwischen k und *3k-1*, wobei k eine feste Konstante sei. Wird ein Cluster einerseits größer als *3k-1* Knoten, führt dies zu einer Aufspaltung dieses Clusters in zwei neue Cluster. Andererseits wird ein Cluster mit weniger als k Knoten mit einem anderen Cluster verschmolzen. Jeder Knoten ist Mitglied eines bestimmten Clusters in Schicht 0. Innerhalb eines jeden Clusters wird in dessen Mitte ein Knoten als Clusterleader ausgewählt. Clusterleader werden wiederum selbst in der nächst höheren Schicht der NICE-Hierarchie zu Clustern gruppiert. Dieses rekursive Verfahren wird solange durchgeführt, bis letztendlich an der Spitze der Hierarchie nur ein einziger Knoten vorhanden ist (Abbildung 3 verdeutlicht dieses Vorgehen nocheinmal anschaulich).

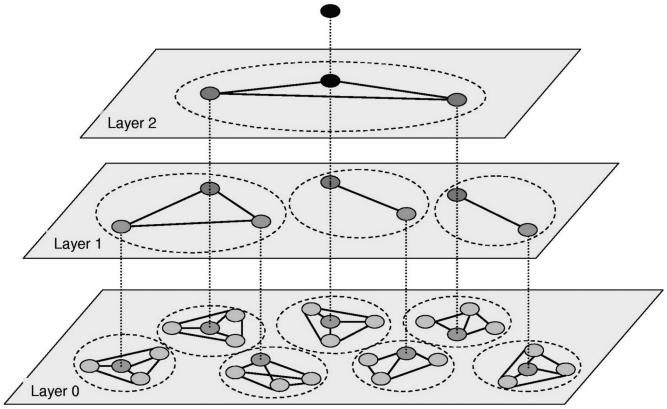


Abb. 3: Das hierarchische NICE-Overlay

Jeder Knoten verwaltet Zustandsinformationen über alle Cluster denen er angehört, sowie seinen Supercluster. Der Supercluster eines Knotens ist dabei der Cluster in der nächst höheren Schicht der Hierarchie, welchem sein Clusterleader angehört. Knoten erhalten Informationen über andere Knoten ihrer Cluster durch *HeartBeat-Nachrichten*, die jeder Knoten in periodischen Abständen an alle Knoten seiner Cluster sendet. Diese Nachrichten enthalten Abstandsabschätzungen, basierend auf Round Trip Time (RTT) Messungen, zu jedem anderen Mitglied des entsprechenden Clusters. Die HeartBeat-Nachrichten des Clusterleaders enthalten die aktuellsten Informationen über die Knotenmitgliedschaften seiner Cluster.
Ein Mitglied signalisiert das Verlassen eines Cluster, indem es in diesem Cluster keine HeartBeat-Nachrichten mehr sendet. Nach einem vorher festgelegten Zeitintervall wird dann dieser Knoten als nicht mehr zugehörig zu diesem Cluster betrachtet. Neue Knoten werden in die NICE-Hierarchie aufgenommen, indem sie zuerst einen *Rendezvous-Punkt (RP)* kontaktieren. Dieser RP antwortet dann mit der Adresse des Knotens in der obersten Schicht der Hierarchie. Der neue Knoten nimmt danach Kontakt mit diesem obersten Knoten auf, um seinen Abstand zu allen Knoten einer Schicht tiefer sowie deren Adressen zu bestimmen. Danach wiederholt der neue Knoten diese Vorgehensweise mit den neuen Adressen und wählt dabei, auf allen Schichten, immer den Knoten der ihm bezogen auf die Abstandsmetrik am Nächsten ist. Letztendlich findet der neue Knoten den ihm am Nächsten gelegenen Cluster in Schicht 0 und tritt diesem bei.
Die NICE-Hierarchie wird ständig verändert und verfeinert, da Knoten jederzeit der Gruppe beitreten oder sie verlassen können. Die Verfeinerung findet statt, indem jeder Knoten die Mitglieder seines Superclusters periodisch untersucht. Falls der eigene Clusterleader nicht der

am Nächsten gelegene ist, verlässt der Knoten seinen Cluster und wechselt in den Cluster des entsprechenden am Nächsten gelegenen Clusterleaders.

2.1.2 Topologie-Bewusstsein / Anpassen der Clusterleader

NICE wurde unter anderem als Grundprotokoll gewählt, weil eine Verbesserung des Overlays bei sich bewegenden Knoten dadurch erreicht wird, daß jeder Knoten die Mitglieder seines Superclusters periodisch untersucht und wenn nötig seinen Cluster wechselt. Der Nachteil von NICE dabei ist, daß der Clusterleader sich in MANETs aus der Mitte seines Clusters bewegen kann und es somit besser wäre, einen neuen Clusterleader zu wählen. NICE-MAN vermeidet diesen Nachteil indem es ständig neue Clusterleader auswählt (Abbildung 4).

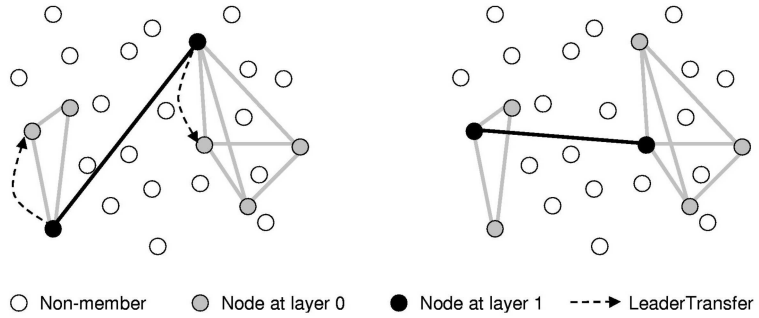


Abb. 4: Wechsel der Clusterleader

Falls ein Clusterleader erkennt, daß ein anderes Clustermitglied ein passenderer Clusterleader wäre, sendet er eine *LeaderTransfer-Nachricht* an dieses Mitglied und übergibt ihm somit die Führung des Clusters.
Clusterleader können eine solche Übergabe nur in einem ihrer Cluster anstoßen, der in der höchsten Schicht der Hierarchie liegt. Dieses Top-down Vorgehen vermeidet Änderungen in tiefereren Schichten der Hierarchie, was weitreichende Strukturänderungen zur Folge hätte.

2.1.3 Optimierung der Metrik

Die meisten im Internet verwendeten Ansätze benutzen RTT (round trip time) Messungen, um in der Nähe befindliche Knoten zu finden und um das Overlay anzupassen. Diese Messungen lassen sich in MANETs nicht einsetzen, da Latenzzeitmessungen, aufgrund von unvorhersagbaren Verzögerungen durch Medienzugriff und erneute Übertragung auf der MAC-Schicht, unzuverlässig sind. Latenzzeiten hängen demnach stark von der gegenwärtigen Netzlast ab.
Eine aussagekräftigere Metrik in MANETs ist die Anzahl der Übertragungsschritte (Hops). Diese Metrik ist wesentlich stabiler und kann oft aus den Unicast-Routingtabellen der einzelnen Knoten abgeleitet werden. Simulationen (in [1]) haben gezeigt, daß RTT Messungen mit zunehmendem Hintergrundverkehr ansteigen und Schwankungen unterliegen, wohingegen die Anzahl der Übertragungsschritte weitgehend stabil bleibt. Aus diesen Gründen wurde für NICE-MAN als Metrik die Anzahl der Übertragungsschritte gewählt, und nicht die in NICE benutzten RTT Messungen.

2.1.4 Local Broadcast Cluster

Ein kennzeichnendes Merkmal von MANETs ist die Broadcast-Fähigkeit des drahtlosen Mediums. Abbildung 5 zeigt einen kleinen Ausschnitt eines MANETs, bei dem alle Knoten in gegenseitiger Übertragungsreichweite liegen.

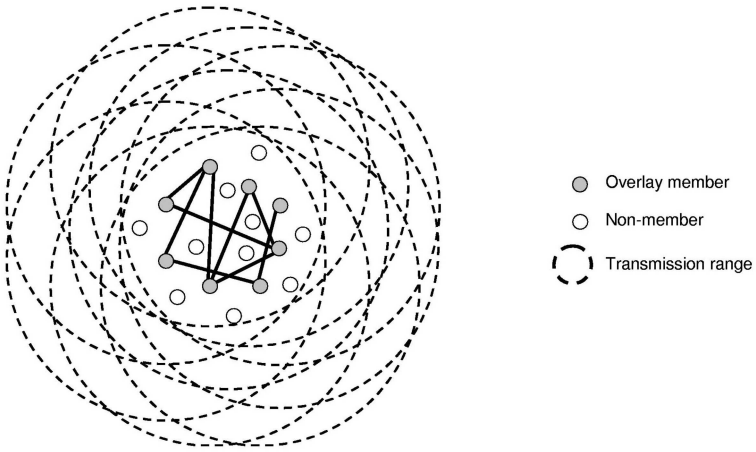


Abb. 5: Alle Knoten liegen in der Übertragungsreichweite voneinander

In diesem Fall ist es sehr unvorteilhaft ein Overlay zwischen den Knoten zu erstellen, da sie sich alle gegenseitig mit nur einem einzigen Übertragungsschritt (Hop) erreichen können. Da der Sendebereich sehr groß sein kann und somit viele Gruppenmitglieder in solch einem Gebiet liegen würden, führt ein Overlay hier zu zunehmender Verbindungsbelastung, längeren Latenzzeiten und Ansteigen des gesamten Netzverkehrs. Vermeiden läßt sich dieses Problem, indem das Overlay als Basisnetz verwendet und die Broadcast-Fähigkeit des Mediums ausgenutzt wird. NICE-MAN führt hierzu die Idee der *Local Broadcast Cluster* (LBC) ein.
Alle Knoten des Overlays sind LBC-Leader deren Sendebereiche die Local Broadcast Cluster bilden. Mit Rücksicht auf die Heterogenität der Knoten sollten dies stärkere Knoten wie z.B. Laptops sein. Knoten die im Sendebereich eines LBC-Leaders liegen (LBC-Mitglieder), senden und empfangen Multicast-Nachrichten über diesen, und treten dem Overlay selbst nicht bei (Abbildung 6). Da die LBC-Mitglieder keine periodischen Kontrollnachrichten versenden, sind sie insbesondere dem Clusterführer unbekannt. Sie können somit ohne vorherige Meldungen in einen anderen LBC wechseln. Die LBC-Leader wiederum selbst, senden periodisch *LocalHeartBeat*-Nachrichten mit Informationen über den Cluster an ihre LBC-Mitglieder. Bewegt sich ein Knoten aus dem Sendebereich seines LBC-Leaders hinaus, empfängt dieser keine LocalHeartBeat-Nachrichten mehr. Er kann sich nun einem anderen LBC anschließen oder sich selbst in das Overlay integrieren. Dies ist sehr einfach, da der Knoten zuvor durch die LocalHeartBeat-Nachrichten Informationen über die anderen Cluster empfangen hat.

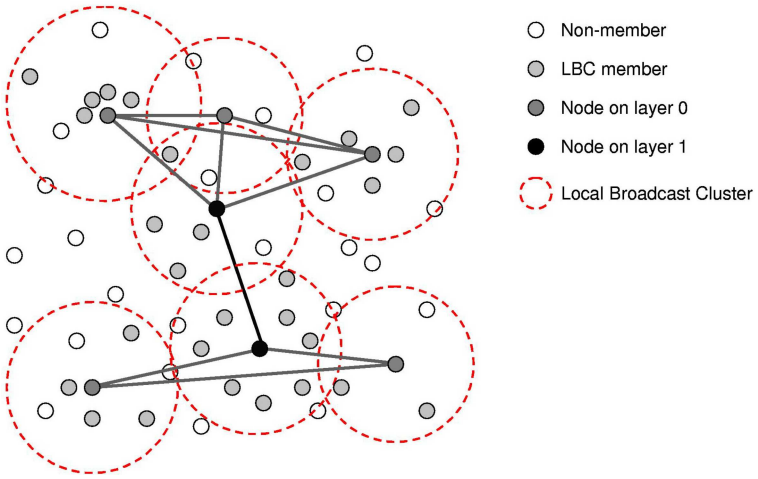


Abb. 6: Overlay mit Local Broadcast Cluster

Zwei LBCs kollidieren, falls die jeweiligen LBC-Leader gegenseitig ihre LocalHeartBeat-Nachrichten empfangen. Hält dieser Zustand eine längere Zeit an, verläßt einer der beiden

Knoten das Overlay, und tritt dem LBC des anderen bei. Um zu vermeiden, daß zwei Knoten dies gleichzeitig tun, verläßt immer der Knoten mit der kleinsten Adresse das Overlay. Eine instabile Hierarchie wird dadurch vermieden, daß dies nur von Knoten durchgeführt werden kann, die auschließlich auf Schicht 0 vertreten sind. Das Prinzip der Local Broadcast Cluster reduziert somit die Anzahl notwendiger Overlay-Änderungen und den gesamten Kontrolloverhead.
Anmerkend sei gesagt, daß die Anzahl der Overlay-Knoten hier nicht von der Gesamtzahl der Mitglieder abhängt, sondern von der Größe des Gebietes in dem die Gruppenmitglieder verteilt sind.

2.1.5 Bootstrapping

Neue Knoten, die dem NICE-Overlay beitreten wollen, müssen zuerst einen Rendezvous-Punkt (RP) kontaktieren. Dieser RP ist als zentrale Instanz in MANETs jedoch nicht verfügbar. Desweiteren müssen neue Knoten in jeder Schicht der NICE-Hierarchie etliche Knoten untersuchen um letztendlich den passenden Cluster in Schicht 0 zu finden. NICE-MAN umgeht diese Nachteile, indem es *Expanding Ring Search* zur Suche nach bereits vorhandenen Knoten im Overlay verwendet. Jeder Knoten, welcher der Multicastgruppe beitreten will, sendet zuerst eine *LocalJoin-Anfrage* an alle Knoten in seinem Sendebereich. LBC-Leader die eine solche Anfrage empfangen, reagieren daraufhin mit einer *LocalHeartbeat-Nachricht*. In diesem Fall tritt der neue Knoten dem Cluster des antwortenden LBC-Leaders bei. Falls jedoch die anfängliche LocalJoin-Anfrage nicht erfolgreich war, beginnt der betretende Knoten das Netzwerk mit exponentiell wachsenden Lebenszeiten (time-to-live) zu fluten, und tritt dem Cluster des ersten antwortenden Overlay-Knotens bei. Das Vorgehen ist in Abbildung 7 nocheinmal als vereinfachtes Zustandsübergangsdiagramm dargestellt.

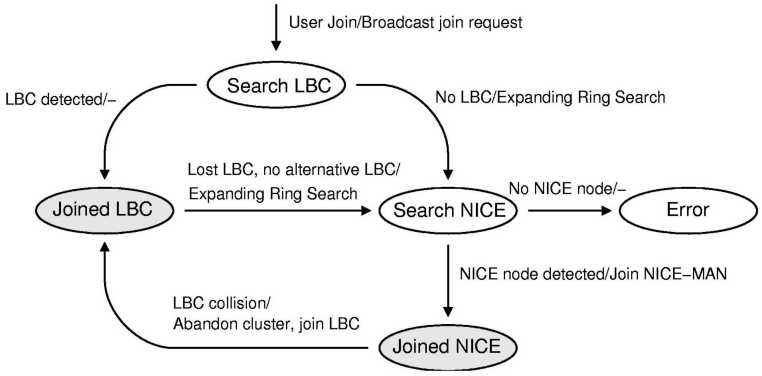


Abb. 7: Zustandsübergangsdiagramm

2.2 PAST-DM

In diesem Abschnitt wird das Overlay-Multicast-Protokoll *Progressively Adapted Sub-Tree in Dynamic Mesh* (PAST-DM) vorgestellt, welches ebenfalls für MANETs entwickelt wurde und einen *mesh-first* Ansatz verfolgt.
PAST-DM passt das Overlay-Netz, sowie den Multicast-Baum für die Zustellung der Datenpakete, schrittweise an die Änderungen der zugrunde liegenden Netzwerkstruktur an, und hält das Beitreten (Join) und Verlassen (Leave) von Knoten einfach und robust.

2.2.1 Dynamisches Overlay-Netz durch link state-Austausch

Bevor eine Multicast-Session beginnt, muß zuerst ein virtuelles Netz (Overlay-Netz) aus allen Gruppenmitgliedern gebildet werden. Um dies zu erreichen, beginnt jeder Mitgliedsknoten

damit, seine virtuellen Nachbarknoten mit Hilfe von *Expanding Ring Search* (ERS) zu finden und ihnen eine *Group_REQ*-Nachricht zu senden. Der maximale Radius sollte dabei recht klein sein. Wenn ein Knoten A eine Group_REQ-Nachricht eines Knoten B empfängt, speichert er diesen Knoten als seinen Nachbarn im Overlay-Netz, zusammen mit der Anzahl von Übertragungsschritten, um diesen zu erreichen. Knoten A sendet daraufhin eine Group_REP-Nachricht an B, damit auch dieser A als seinen Nachbarn und die Anzahl der Übertragungsschritte speichern kann. Falls die Anzahl der virtuellen Nachbarn eines Knoten die obere Grenze erreicht, stoppt der Knoten seine Nachbarsuche.
Jeder Mitgliedsknoten beobachtet die anderen Mitglieder in seiner Umgebung. Dies kann entweder dadurch geschehen, daß er seine Routingtabelle, verwaltet durch ein Unicast-Protokoll, abfragt, oder periodisch eine Nachbarsuche durchführt. Jeder Knoten speichert seine virtuellen Nachbarn als seinen *virtual link state*.
In PAST-DM kennt jeder Mitgliedsknoten die Struktur des Overlay-Netzes. Dies wird durch den Autausch der „link states“ erreicht, eine Technik die im *Fisheye State Routing Protokoll für Ad-Hoc Netze* Anwendung findet. In jedem Knoten wird das Abbild der Overlay-Netzstruktur durch eine „link state“-Tabelle dargestellt. Die Einträge dieser Tabelle sind die von den virtuellen Nachbarn empfangenen „link state“-Informationen aller Mitgliedsknoten. Jeder Knoten tauscht seine „link state“-Tabelle periodisch mit seinen Nachbarknoten aus. Die Einträge enthalten außerdem eine Sequenznummer, dabei werden beim Austausch der Tabelle Einträge mit niedrigeren Sequenznummern durch höhere Sequenznummern ersetzt.
Das Konzept der „link state“-Tabellen führt dazu, daß jeder Knoten eine eigene Sicht auf die Struktur des Overlay-Netzes hat. Um zu vermeiden, daß alle Knoten ihre Tabellen gleichzeitig austauschen, kann jeder Knoten zu der Austauschperiode einen zufälligen Wert addieren. Dies führt zu einer stabileren Leistung des gesamten Netzwerks.

2.2.2 Datenverteilungsbaum

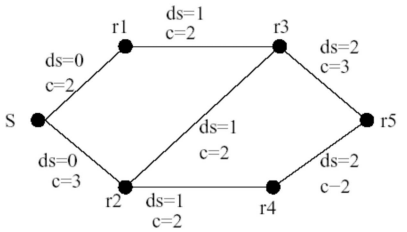
PAST-DM verfolgt den Ansatz quellen-basierter Bäume zur Datenverteilung. Der Unterschied zu anderen quellen-basierten Ansätzen ist, daß in PAST-DM jede Quelle ihren Datenverteilungsbaum aus der eigenen „link state“-Tabelle konstruiert. Es ist somit kein weiterer Aufwand für Kontrollnachrichten notwendig.
Für die Berechnung des Baumes wurde ein neuartiger *quellen-basierter Steiner-Baum* Algorithmus entwickelt. Um die Gesamtkosten des Multicast-Baumes zu reduzieren, ist es notwendig das die Quelle einen *Steiner*-Baum für das Overlay-Netz konstruiert. Da die Sicht des Quellknotens auf das Overlay-Netz von seiner „link state“-Tabelle abhängt, sind die Informationen über die Struktur der Knoten in seiner Nähe sehr aktuell und genau. Je weiter jedoch die Knoten von der Quelle entfernt sind, umso ungenauer werden diese Informationen. Aus diesem Grund sollte stets, wenn bei der Baumkonstruktion zwei Verbindungen des Overlay-Netzes die gleichen Kosten verursachen, die Verbindung bevorzugt werden, die der Quelle am Nächsten ist. Weiterhin sollten auf alle Fälle die an die Quelle anschließenden virtuellen Verbindungen im Baum enthalten sein, da die „link state“-Informationen über diese am aktuellsten und genauesten sind. Um diese Eigenschaften zu erreichen wird der im folgenden beschriebene *quellen-basierter Steiner-Baum* Algorithmus verwendet.
Sei *ds(n)* die Anzahl der Übertragungsschritte (Hops) vom Quellknoten s zum Knoten n, ungeachtet der Verbindungskosten. Für eine virtuelle Verbindung (n_1, n_2) ist die Anzahl der Übertragungsschritte zur Quelle s definiert als

$$ds(n_1, n_2) = min[ds(n_1), ds(n_2)].$$

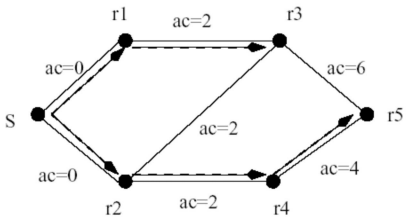
Sei $c(n_1, n_2)$ die Kosten der virtuellen Verbindung (n_1, n_2). Die *angepassten Kosten* der Verbindung definieren sich aus der Multiplikation ihrer Kosten mit der Anzahl der Übertragungsschritte zur Quelle s,

$$ac(n_1, n_2) = ds(n_1, n_2) * c(n_1, n_2).$$

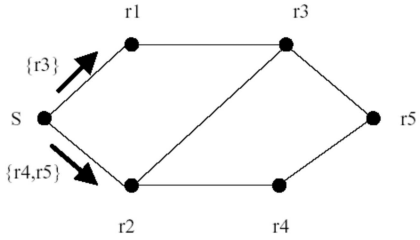
Aus diesen Definitionen folgt, daß die Anzahl der Übertragungsschritte und die angepassten Kosten, jeder an die Quelle anschließenden Verbindung, gleich 0 sein müssen. Folgende methodische Vorgehensweise wird angewendet, um den Steiner-Baum zu berechnen. Jede Verbindung benutzt dabei ihre *angepassten Kosten*.
Der Initialbaum enthält den Quellknoten als Wurzel, und alle mit dem Quellknoten verbundenen Nachbarknoten als Kindknoten der ersten Ebene. Der nun konstruierte Teilbaum wird zum Steiner-Baum zunehmend erweitert, indem immer der am Nächsten gelegene Empfänger, zusammen mit dem kürzesten Weg, um diesen zu erreichen, hinzugefügt wird. Durch anwenden des *quellen-basierten Steiner-Baum* Algorithmus macht die Quelle alle ihre Nachbarknoten zu ihren Kindknoten der ersten Ebene im Multicast-Baum, und unterteilt die verbleibenden Knoten in Untergruppen. Jede Untergruppe wiederum bildet einen Unterbaum, dessen Wurzel einer der Kindknoten erster Ebene ist. Die Kindknoten wenden dann auf ihren Unterbaum ebenfalls den *quellen-basierten Steiner-Baum* Algorithmus an. Dieser Vorgang stoppt erst, wenn die Untergruppe entweder leer ist, oder nur einen Knoten enthält. Der Quellknoten muß somit nicht den ganzen Multicast-Baum berechnen. Er fügt eine Liste jeder Untergruppe in den Paketkopf ein, verbindet ihn mit einer Kopie des Datenpaketes, und sendet das gesamte Paket an den entsprechenden Kindknoten, welcher dann für die weitere Auslieferung an die Knoten seiner Untergruppe verantwortlich ist.
Um dieses Vorgehen nocheinmal zu verdeutlichen, zeigt Abbildung 8 ein Beispiel. Die Empfängerliste des Quellknoten S ist $\{r_1 \dots r_5\}$.



(a) Distance and cost of each link



(b)Adapted cost of each link and Source–Based Steiner tree



(c) Receiver lists in the header of the packets sent from S to its children

Abb. 8: Beispiel einer Baumerzeugung

2.2.3 Beitreten und Verlassen einer Multicastgruppe

Wenn ein Knoten beabsichtigt, einer Multicastgruppe beizutreten, beginnt er zuerst, wie in 2.2.1 beschrieben, nach der Suche seiner Nachbarknoten. Da es im Netzwerk mehrere Multicastgruppen gleichzeitig geben kann, muss die *Group_REQ*-Nachricht eine Gruppenadresse beinhalten. Durch die *Group_REP*-Nachricht, der Mitgliedsknoten der adressierten Gruppe, erkennt der neue Knoten alle seine virtuellen Nachbarn, und speichert sie als seinen *virtual link state*. Die antwortenden Mitgliedsknoten erkennen den neuen Knoten ebenfalls als ihren Nachbarn, und beginnen ihre „link state"-Tabelle mit ihm auszutauschen. Nach einer gewissen Zeit wird dann der neue Knoten auch von im Netzwerk weit entfernten Mitgliedsknoten erkannt.
Obwohl der neue Knoten erst später vom weit enfernten Quellknoten erkannt wird, kann er sofort, nachdem ihn seine Nachbarknoten erkannt haben, Datenpakete empfangen. Dies wird durch eine zusätzliche *Datenweiterleitungsfunktion* in jedem Kindknoten des Multicastbaums erreicht. Nachdem ein Kindknoten ein Datenpaket an alle Knoten seiner Untergruppe weitergeleitet hat, überprüft er, ob auch alle seine benachbarten Empfänger in der Untergruppe enthalten sind. Fehlenden Empfängern wird das Datenpaket dann nocheinmal extra zugesendet. Da es möglich ist, daß neue Knoten mehrere virtuelle Nachbarn haben, können auch mehrere Kopien identischer Datenpakete empfangen werden. Die Duplikate sollten draufhin verworfen werden. Wenn die Quelle den neuen Knoten dann letztendlich erkannt hat, und ihn in ihre Auslieferungsliste eingefügt hat, wird dieser keine vervielfältigten Datenpakete mehr empfangen.
Um eine Multicastgruppe zu verlassen, sendet ein Mitglied eine *Group_LV*-Nachricht an all seine derzeitigen virtuellen Nachbarknoten. Der Austausch der „link state"-Tabelle zwischen diesem Mitglied und seinen Nachbarn wird dann sofort eingestellt. Desweiteren liefern die Nachbarknoten keine Datenpakete mehr an diesen Knoten aus, obwohl dieser noch eine gewisse Zeit in der Knotenliste der Untergruppe vorhanden ist.

3 Simulationen und Auswertungen

3.1 NICE-MAN

Simulationen (in [1]) mit dem QualNet Simulator haben ergeben, daß sich das Overlay bei der Verwendung von LBCs bedeutend schneller stabilisiert, und auch weniger Overlayänderungen nötig sind, da die LBC-Mitglieder nicht teil des Overlays sind. Dies unterstreicht die Bedeutung des LBC-Konzeptes und dessen Eigenschaft, häufige Umgestaltungen des Overlays zu reduzieren. Ein weiteres interessantes Simulationsergebnis ist die Anzahl der Knoten die Teil des Overlays sind, und die Anzahl der LBC-Mitglieder. Es hat sich herausgestellt, daß die meisten Knoten Mitglieder der Local Broadcast Cluster sind, und somit der Kontrollaufwand des gesamten Overlays mit LBCs verringert wird.
Das Zustellrate ist ein sehr wichtiges Kriterium für Multicast-Protokolle. Im Vergleich zu ODMRP (On-demand multicast routing protocol) und NICE-MAN ohne LBCs, haben die Simulationen ergeben, daß NICE-MAN mit LBCs das beste Zustellrate besitzt. Insgesamt ergibt sich aus den Simulationen, daß NICE-MAN die Latenzzeit und das Zustellrate verbessert, sowie den Kontrollaufwand stark reduziert.

3.2 PAST-DM

Simulationen (in [2]) zeigen, daß PAST-DM im Vergleich zu AMRoute (Ad hoc Multicast Routing Protocol) nahezu die ganze Zeit effizienter Multicast-Bäume konstruiert, und die

relativen Baumkosten wesentlich geringer ausfallen. Die relativen Baumkosten sind das Verhältnis der gemessenen Baumkosten zu den optimalen Baumkosten. Hierbei definieren sich die gemessenen Baumkosten als die Summe der Übertragunsschritte aller virtuellen Verbindungen des Baumes, und die optimalen Baumkosten als die Gesamtheit aller Übertragungsschritte eines in der physikalischenen Netzwerkstruktur konstruierten Steiner-Baumes. In PAST-DM unterliegen die relativen Baumkosten nicht so starken Schwankungen wie bei AMRoute, was eine stabilere Netzwerkleistung bedeutet.

4 NICE-MAN vs. PAST-DM

NICE-MAN gehört zur Kategorie der strukturierten Overlays die einen impliziten Ansatz verfolgen. Die spezielle Struktur dieses Ansatzes wird direkt dazu benutzt, die Multicastdaten zuzustellen. Im Gegensatz dazu gehört PAST-DM zur Kategorie der unstrukturierten Overlays, auch mesh-first Ansatz genannt, und liefert die Daten mit Hilfe quellenspezifischer Multicastbäume aus. NICE-MAN und PAST-DM haben gemeinsam, daß sie beide permanent die Struktur des Overlay-Netzes optimieren, indem sie es immer wieder an das zugrunde liegende physikalische Netz anpassen. Der Nachteil von PAST-DM ist, daß das geteilte Medium nicht ausreichend betrachtet wird, und dies zu ineffizienter Verteilung der Daten führt wenn sich mehrere Knoten in gegenseitiger Übertragungsreichweite befinden. NICE-MAN löst dieses Problem, indem es das Konzept der Local Broadcast Cluster einführt.

5 Zusammenfassung

Es wurden zwei Ansätze vorgestellt: NICE-MAN und PAST-DM.
NICE-MAN baut auf dem Festnetzprotokoll NICE auf und führt an diesem einige Verbesserungen zur Leistungssteigerung in MANETs durch. Die Verbesserungen umfassen eine permanente Anpassung des Overlays an das zugrunde liegende Netzwerk, eine auf der Anzahl der Übertragungsschritte basierende Abstandsmetrik, sowie die Nutzung der Broadcast-Fähigkeit des Mediums durch das Konzept der Local Broadcast Cluster. Jedoch sind die Verbesserungen nicht alleine auf NICE oder Multicast-Overlays beschränkt, sondern können auch Anwendung in willkürlichen Overlays, für z.B. File Sharing, finden. Die Vorteile des vorgestellten NICE-MAN Protokolls sind reduzierter Kontrollaufwand und eine gute Ausnutzung der Netzwerkresourcen.
PAST-DM zeichnet sich dadurch aus, daß es das Overlay ständig optimiert und an die Mobilität der Netzwerkknoten anpasst. Durch den Austausch der „link state“-Tabellen hat jeder Knoten eine eigene Sicht über das Overlay. Die Auslieferung der Multicastdaten erfolgt in PAST-DM durch quellen-basierte Multicastbäume.

Literatur

[1] Steffen Blödt. Efficient End System Multicast for Mobile Ad Hoc Networks. University of Karlsruhe, Institute of Telematics.

[2] Chao Gui and Prasant Mohapatra. Efficient Overlay Multicast for Mobile Ad Hoc Networks. University of California (Davis), Computer Science Department.

TCP-Staukontrolle in drahtlosen Multi-hop-Netzwerken

Christian Biedermann

Kurzfassung

Eines der Standardprotokolle des Internets ist das TCP (Transmission Control Protocol). Seine Mechanismen zur Behebung von Problemen bei der Übertragung von Daten beruhen auf der Annahme, dass Kommunikationsprobleme ihren Ursprung in der Überlastung des Netzwerkes haben und es somit zu Stausituationen kommt. Eine andere Fehlerquelle wurde nahezu ausgeschlossen. Zum Zeitpunkt der Implementierung des TCPs gab es fast ausschliesslich drahtgebundene Verbindungen, welche heute allerdings mehr und mehr durch die drahtlose Kommunikation ergänzt bzw. ersetzt werden. Die bisherigen Mechanismen des TCPs sind nicht mehr in der Lage, auf diese geänderten Voraussetzungen adäquat zu reagieren. Einige der derzeitigen Bestrebungen, die ursprüngliche TCP-Staukontrolle für die Verwendung in drahtlosen Ad-Hoc-Netzwerken (MANETs) zu modifizieren, sind Gegenstand dieses Vortrages.

1 Einleitung

1.1 TCP - Transmission Control Protocol

Das TCP wurde 1981 von Jon Postel im RFC 793 [Post81] vorgestellt und entwickelte sich rasch zum quasi Standardprotokoll des Internets. TCP ist im OSI-Schichtenmodell auf Schicht 4 (Transport) angesiedelt und setzt damit auf dem IP (Internet Protocol) auf. Es handelt sich bei TCP um ein Transportprotokoll zur Übertragung von Byteströmen, welche von den oberen Schichten übergebenen werden. Diese Ströme werden in Segmente aufgeteilt und per IP übertragen. Im Gegensatz zum UDP (User Datagram Protocol), welches nur die verbindungslosen, unzuverlässigen Dienstmerkmale des IP an obere Schichten direkt weitergibt, handelt es sich bei TCP um ein verbindungsorientiertes Ende-zu-Ende-Transportprotokoll mit garantierter Übertragung der Datenpakete. Die Verbindungsendpunkte werden durch sogenannte Sockets, bestehend aus der (IP-)Adresse des Gerätes und des (TCP-)Ports der Anwendung, eindeutig bestimmt. Der Verbindungsaufbau wird durch das Drei-Wege-Handshakeverfahren initiiert. Während dieses Verfahrens wird auch der Sendekredit zwischen Sender und Empfänger ausgehandelt. Dieser Parameter spielt in der Flusssteuerung später noch eine entscheidende Rolle. Neben dem Sendekredit werden auch Sendefolgenummer und Empfangsfolgenummer zwischen Sender und Empfänger ausgetauscht. Das jeweils nächste Paket erhält einen um eins erhöhten Wert. So ist es dem Empfänger möglich, die empfangenen Pakete wieder anhand der fortlaufenden Sendefolgenummern zu dem ursprünglichen Bytestrom zusammenzusetzen. Kommt es bei der Übertragung zu Störungen in den darunterliegenden Schichten, ist TCP nicht in der Lage zu erkennen, wo das Problem exakt liegt. TCP ist aufgrund der Statusmeldungen des Empfängers nur in der Lage folgende Probleme zu erkennen:

- **Dupklikate**
  Erhält ein Empfänger ein Paket mehrfach, geht TCP davon aus, dass das Paket dupliziert wurde und verwirft die Dupklikate.

- **Überholen**
  Erhält der Empfänger die Pakete nicht in der richtigen Reihenfolge, wartet der Empfänger einen bestimmten Zeitraum, um eventuell noch die ausstehenden Pakete zu empfangen und damit die Paketfolge wieder zu komplettieren, bevor er die entsprechende Quittierung an den Sender schickt.

- **Verlust - keine Bestätigung**
  Erhält der Sender keine Bestätigung vom Empfänger, gibt es zwei mögliche Gründe hierfür. Entweder das gesendete Paket oder die Bestätigung ist unterwegs verloren gegangen. Nach Ablauf eines Timers übermittelt der Sender sein Paket nochmals an den Empfänger.

- **Verlust - mit Bestätigung**
  Erhält der Sender Bestätigungen vom Empfänger mit immer gleich lautender Sendefolgenummer X, obwohl der Sender bereits mehrere neue Pakete übermittelt hat, geht der Sender von einem Verlust des Paketes X+1 aus, da der Empfänger immer nur das höchste Paket aus der kompletten Paketfolge quittiert. Der Sender übermittelt also Paket X+1 erneut. (siehe auch 1.2.2)

1.2 TCP/IP - Flusssterung bzw. Staukontrolle

TCP wurde ursprünglich primär für den Einsatz in Festnetzen konzipiert und entwickelt. Der Hauptgrund für Paketverluste in Festnetzen ist die temporäre oder dauerhafte Überlastung einzelner Netzwerksegmente. Erhält z.B. ein Router mehr Pakete als er weiterleiten kann, füllt sich sein Puffer bis dieser „überläuft“. Bei einem solchen „Buffer Overflow“ werden alle weiteren ankommenden Datenpakete solange verworfen, bis der Eingangspuffer des Routers wieder aufnahmefähig ist, d.h. der Router einige der wartenden Pakete weitervermitteln konnte. Das Verwerfen der Datenpakete geschieht auf IP-Schicht, so dass die TCP-Instanz des Senders oder Empfängers keine Benachrichtigung über den Verlust erhalten. Dies würde aber bedeuten, dass der Sender weiterhin Datenpakete an den Empfänger verschickt und so die Stausituation weiter eskalieren würde. Erhält der Sender aber nicht die erwarteten Quittungen des Empfängers, sei es die falsche Sendefolgenummer oder gar keine Quittung, geht TCP von einer Stausituation aus und startet je nach Ausgangslage einen entsprechenden Mechanismus, um auf die geänderte Netzlage zu reagieren.

1.2.1 Slow-Start Algorithmus / Congestion avoidance

Wie bereits beschrieben können auf den unteren Schichten Daten aufgrund von Stausituation verloren gehen. Eine „blinde“ Sendewiederholung würde die Überlastsituation nur verstärken. Daher setzt TCP den sogenannten Slow-Start-Algorithmus ein, um diese Eskalation zu vermeiden. In diesem Mechanismus finden folgende Kennzahlen ihre Anwendung:

- **Staufenster (Congestion Window)**
  Wird zu Beginn auf 1 gesetzt und kennzeichnet die aktuelle Anzahl an Paketen, bevor der Sender auf die eingehenden Quittungen warten muss. Dies ist nicht gleichzusetzen mit der maximal erlaubten Anzahl Bytes pro Paket.

- **Stauschwellwert (Congestion Threshold)**
  Wird zu Beginn der Übertragung offen gelassen. Bei jedem Einsetzen des Slow-Start-Algorithmus wird der Wert auf $\frac{Staufenster}{2}$ gesetzt. Solange das Staufenster den Schwellwert noch nicht erreicht hat, wächst das Fenster durch Verdoppelung exponentiell. Überschreitet das Staufenster den Schwellwert, wird das Wachstum linear durch jeweilige Erhöhung um 1 fortgesetzt.

- **Staufensterlimit (Congestion Window Limit)**
  Dieser Wert begrenzt die Kapazität des Staufensters nach oben hin. Das Staufensterlimit ist in Festnetzen üblicherweise nicht gesetzt. Die Entwicklung des Staufensterlimits ist neueren Datums und wurde zur Verbesserung der TCP-Leistungsfähigkeit in drahtlosen Netzwerken konzipiert.

Zu Beginn der Übertragung ist der Wert des Staufensters 1, daher wird ein Paket gesendet. Beim Eingang der korrekten Quittung wird das Staufenster um den Wert 1 erhöht und es werden zwei weitere Pakete gesendet ($Staufenster = 2$). Nach Erhalt der Bestätigungen erhöht sich das Staufenster um 2 (pro Paket um eins). Der Algorithmus fährt exponentiell mit der Erhöhung des Staufensters fort, solange er korrekte Bestätigungen erhält oder die Kapazität des Staufensters den Stauschwellwert erreicht. Im zweiten Fall schaltet der Algorithmus nun von exponentieller auf lineare Steigerung um. Dieses Wachstum wird solange fortgesetzt bis das Staufensterlimit erreicht wird. Sollten Pakete verlorengehen, wird der Wert des Stauschwellwert auf die Hälfte des aktuellen Staufensters und das Staufenster danach auf 1 gesetzt. Die Übertragung wird mit den neuen Parametern fortgesetzt. Durch dieses rigorose Rücksetzen des Staufensters auf 1 und das exponentielle Wachstum bekommt ein Paketnummer-Staufenster-Graph den charakteristischen Sägezahneffekt (Abb. 1).

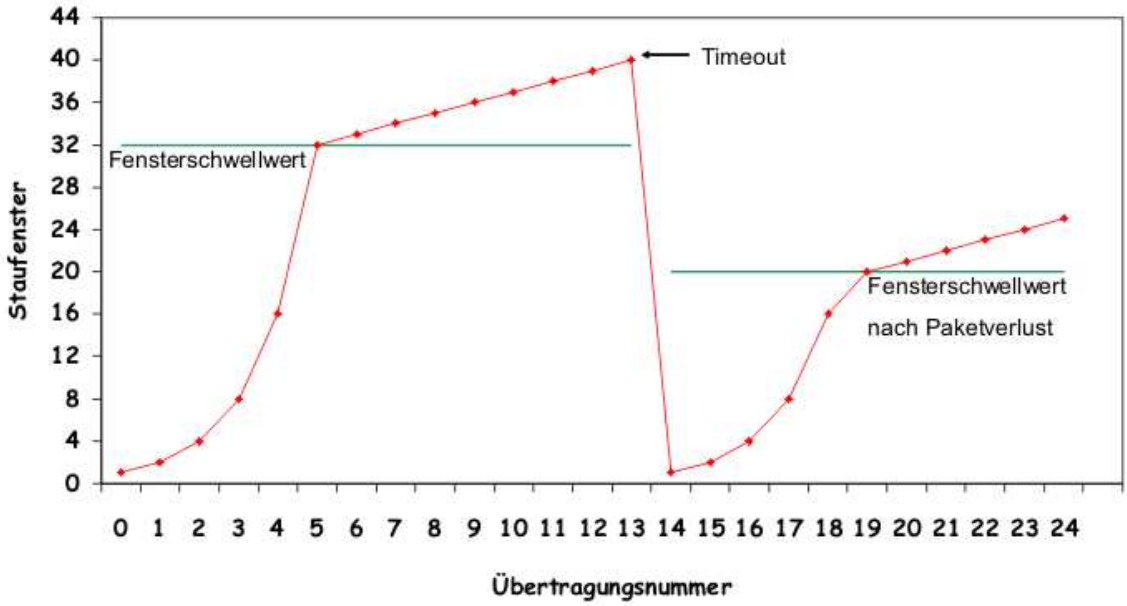


Abbildung 1: Verlauf des Staufensters bei Slow-Start-Algorithmus

1.2.2 Fast Retransmit / Fast Recovery

Empfängt ein Sender nur noch Bestätigungen desselben Paketes X, so geht er davon aus, dass das Paket X+1 verloren gegangen ist. Er startet somit den Slow-Start-Algorithmus und überträgt die Pakete ab X+1 neu. Gleichzeitig setzt der Algorithmus den Stauschwellwert auf die Hälfte des aktuellen Staufensters und weist diesem den Wert 1 zu. Durch diese Vorgehensweise würde der Verlust eines einzelnen Paketes die zügige Übertragung ausbremsen und das Staufenster müsste erst langsam wieder aufgebaut werden. Um dies zu verhindern, wurde das Fast Retransmit-Verfahren entwickelt. Erhält der Sender Bestätigungen des Paketes X, erkennt er, dass das Paket X+1 verloren gegangen ist. Gleichzeitig nimmt er

an, dass die nachfolgenden Pakete beim Empfänger angekommen sein müssen, die Quittungen jedoch alle durch das Fehlen des Paketes X+1 für das Paket X ausgestellt wurden. Anstatt nun den Slow-Start-Algorithmus zu initiieren, wird das betreffende Paket X+1 neu übertragen. Der Empfänger kann so die Lücke in seiner Paketfolge schließen und wird das höchste Paket aus dieser kompletten Folge quittieren. Der Wert des Staufensters bleibt unverändert. Da hier auf den Slow-Start-Algorithmus und die damit einhergehende Verkleinerung des Staufensters verzichtet werden konnte, spricht man bei diesem Verfahren auch von „Fast Recovery“ (Abb. 2).

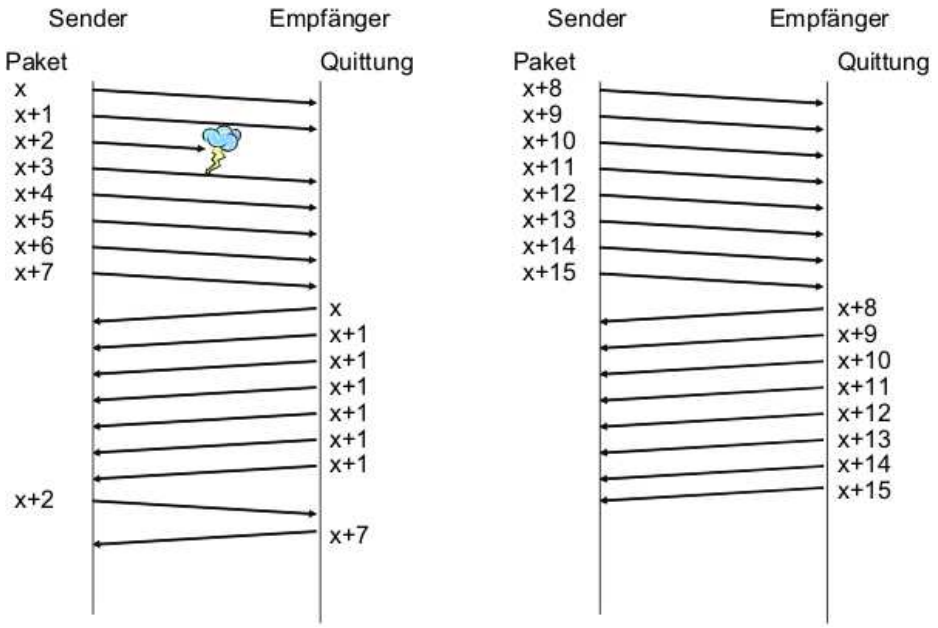


Abbildung 2: Ablaufdiagramm des Fast ReTransmit/Fast ReCovery-Algorithmus

1.3 MANET (Mobile Adhoc Network)

MANETs sind drahtlose, mobile Netzwerke, welche spontan gebildet werden können. Der entscheidene Unterschied zu den ortsgebundenen WLANs ist die Tatsache, dass keine Infrastruktur wie z.B. ein AccessPoint benötigt wird, um das Netzwerk aufzubauen. Die Teilnehmer des MANETs erkennen und kommunizieren mit ihren Nachbarn und erfahren über diese von weiteren Teilnehmern des MANETs. Ein problematischer Punkt ist das Routing in solchen Netzwerken, da man davon ausgehen kann, dass die Teilnehmer im Netz wandern. Dadurch kann es vermehrt zu Unterbrechungen der Route bzw. zur Wiederholung des Sendevorgangs kommen. Es gelten für MANETs im übrigen die gleichen Probleme wie für strukturelle WLANs, d.h. das Medium Luft wird von allen Teilnehmern gleichermassen genutzt und somit geteilt. Das vom WLAN bekannte Problem der versteckten Endgeräte, d.h. zwei Endgeräte erkennen ein freies Medium und erzeugen durch ihren Sendevorgang einen Konflikt bei einem dritten, spielt in MANETs sicherlich ebenfalls eine große Rolle.

1.4 Probleme der Staukontrolle in MANETs

Das wohl größte Problem stellt die Eigenschaft TCPs dar, Übertragungsprobleme auf Stausituationen zurückzuführen. Sicherlich kommt es in mobilen, drahtlosen Netzwerken auch zu Überlastungsmomenten, jedoch können zahlreiche andere Ursachen für Paketverluste identifiziert werden:

- Hohe Fehlerrate
  das Medium Luft hat im Gegensatz zum Festnetz eine wesentlich höhere Bitfehlerrate. Diverse Störeinflüße wie z.B. Dopplereffekt, Mehrwegeproblem, Abdeckung durch Bauwerke, Reflexion zeichnen hierfür verantwortlich

- plötzlicher Verbindungsabbau
  Ein Knoten kann in einem MANET plötzlich und unerwartet verlorengehen. Die Gründe hierfür können einerseits beim Gerät selbst liegen (z.B. Ausfall der Stromversorgung) oder aber in der Änderung der äußeren Einflüße (z.B. plötzliche Abdeckung beim Betreten eines Gebäudes)

- Limitierte bzw. wechselhafte Bandbreite
  Das Medium Luft wird von allen Teilnehmern in gleicher Weise genutzt. Daher wird die jedem Teilnehmer gleichermaßen zugewiesene Bandbreite durch das Hinzukommen neuer Knoten reduziert, um auch den neuen Teilnehmern ihren Anteil an der Gesamtbandbreite zu garantieren. Bei MANET geht man außerdem von der Mobilität der Knoten aus. Dies kann mit zunehmender Entfernung eine Verschlechterung der Übertragung und damit eine Reduzierung der Bandbreite bedeuten.

- Energieverbrauch
  Mobile Endgeräte werden üblicherweise mit Akkumulatoren betrieben und sollten daher mit ihren Energiereserven sparsam umgehen. Dies bedeutet aber auch, dass die eingesetzten Protokolle nicht unnötig mit den anderen Teilnehmern kommunzieren bzw. unnötige Sendungswiederholung unternehmen. Jede Sendung kostet hier Energie, welche nur in begrenztem Umfang zur Verfügung steht.

- Dynamische Topologie
  Durch die Bewegung der Knoten in einem MANET kommt es permanent zu Änderungen in der Wegewahl. Dieser Effekt wird durch das Hinzukommen und Entfernen von Knoten noch drastisch verstärkt.

Aufgrund dieser zahlreichen Ursachen kommt es bei den Übertragungen in MANETs häufig zu Paketverlusten. Da TCP fast immer mit dem Slow-Start-Algorithmus reagiert, ist die durchschnittliche Übertragungsrate niedrig. Auch Fast Retransmit / Fast Recovery (1.2.2) kann hier nicht eingesetzt werden, da durchaus mehr wie ein Paket pro Sendung verlorengehen können. Der Einsatz des Fast Retransmit hätte in diesem Fall zur Folge, dass jedes fehlende Paket einzeln übertragen werden würde. Auf längere Zeit betrachtet, würde die durchschnittliche Übertragungsrate gegen 1 tendieren.

2 Betrachtung einiger Ansätze

Die in der Einleitung aufgeführten Probleme lassen den Schluss zu, dass eine Neuimplementation des TCPs der wohl beste Lösungsansatz wäre. Jedoch ist TCP so weit verbreitet, dass dies in der Realität wohl keine Umsetzung erfahren würde. Man kann dies vergleichen mit der Einführung des IPv6, welches Dank der zahlreichen Modifikationen des IPv4, kaum eingesetzt wird. Eine Umstellung auf ein neues TCP wäre mit viel Aufwand und Kosten verbunden. Daher gehen die meisten Bemühungen in Richtung Beibehaltung des derzeitigen TCP bzw. dessen Modifikation. Allgemein kann man die Ansätze in drei Kategorien einteilen:

- Link Layer Protokolle sind unterhalb von TCP angesiedelt und versuchen, die Probleme der drahtlosen Übertragung vor TCP zu verbergen.

- End-to-End Protokolle stellen Modifikationen des bisherigen TCPs dar. Speziell die Entwicklung einer neuen „Staukontrolle“ steht hier im Vordergrund.
- Split-Connection Protokolle trennen die Verbindung zwischen Festnetz- und Wireless-Host am Access-Point in zwei Teile. Der Festnetzteil der Kommunikation läuft mit den bisherigen Mechanismen ab. Der drahtlose Teil wird mit einem neuen Übertragungsprotokoll realisiert.

2.1 Adaptives Setzen eines Staufensterlimits (Congestion Windows Limit)

Die ursprüngliche Implementierung von TCP sieht den Wert eines Staufensterlimits nicht vor. Dieser Ansatz ist erst später entwickelt worden, um die Probleme, welche das TCP durch den Einsatz seiner Staukontrolle in drahtlosen Netzwerken hervorruft, zu schmälern. Das Staufensterlimit ist eine obere Schranke für die Kapazität des Staufensters und darf von diesem nicht überschritten werden. Die Abb. 3 verdeutlicht die Vorgehensweise des Slow-Start-Algorithmus unter Verwendung des Staufensterlimits an einem Beispiel. Das Staufensterlimit

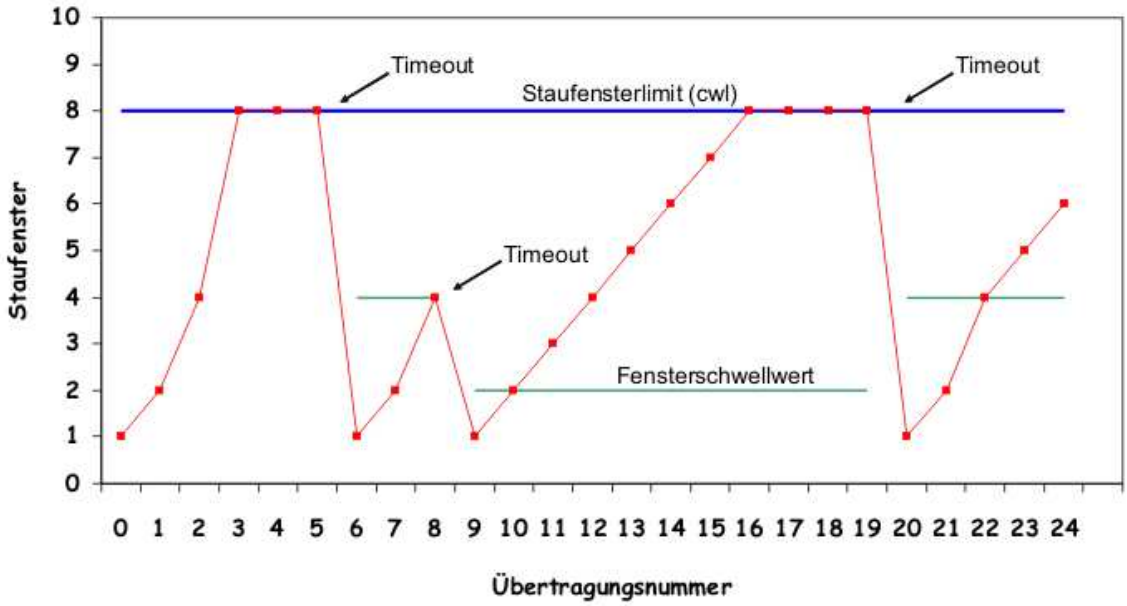


Abbildung 3: Ablauf des Slow-Start-Algorithmus unter Verwendung des Staufensterlimits

wird im Allgemeinen fest gewählt, wobei sich auch hier unterschiedlichen Ansätze bzw. „Standardwerte“ herausgestellt haben. Es werden hier Werte von 1,2 oder 8 als geeignetes Staufensterlimit in drahtlosen Netzwerken genannt.

2.1.1 Finden einer Obergrenze für das Staufensterlimit

In [Nahr03] unternehmen Chen, Xue und Nahrstedt den Versuch, den Wert des Staufensterlimits an die jeweils aktuellen Gegebenheiten anzupassen. Nach einigen Vorüberlegungen kommen die Autoren zum Schluss, das Staufensterlimit mit dem BDP nach oben hin zu beschränken. Das BDP ist die Kapazität der network-pipe vom Sender zum Empfänger und zurück. Es wird als Produkt aus der kleinsten Bandbreite (bootleneck) des Hinweges und der Verzögerungszeit des Hin- und Rückweges errechnet. Die einzelnen Faktoren seien hier kurz erläutert:

- kleinste Bandbreite des Hinweges

$$B_{min} = min(b_1, b_2, ..., b_n)$$

b_x ist die Bandbreite des Teilabschnittes x
n ist die Anzahl der Knoten auf dem Hinweg.

- Verzögerungszeit des Hin- und Rückweges

$$\frac{S}{b_1} + \frac{S}{b_2} + ... + \frac{S}{b_n} + \frac{S'}{b'_1} + \frac{S'}{b'_1} + ... + \frac{S'}{b'_m}$$

S bzw. S' ist die Größe des Pakets auf dem Hin- bzw. Rückweg.
b_x bzw. b'_x ist die Bandbreite des Teilabschnittes x des Hin- bzw. Rückweges.
n bzw. m ist die Anzahl der Knoten auf dem Hin- bzw. Rükweg.

Das BDP wurde ursprünglich für Berechnungen in Festnetzen konzipiert. Der Ansatz läßt sich aber auch in drahtlose Netze transferieren. Das BDP sollte vom Staufensterlimit nicht überschritten werden. Dazu werden in [Nahr03] folgende Überlegungen angestellt:
Verzögerung Hinweg:

$$\frac{S}{b_1} + ... + \frac{S}{b_n} \leq \frac{S}{B_{min}} + ... + \frac{S}{B_{min}} = n\frac{S}{B_{min}}$$

Verzögerung Rückweg:

$$\frac{S'}{b'_1} + ... + \frac{S'}{b'_m} \leq \frac{S'}{B'_{min}} + ... + \frac{S'}{B'_{min}} = m\frac{S'}{B'_{min}}$$

Mit $S' \leq S$ folgt

$$B_{min}(\frac{S}{b_1} + ... + \frac{S}{b_n} + \frac{S'}{b'_1} + ... + \frac{S'}{b'_m}) \leq B_{min}(n\frac{S}{B_{min}} + m\frac{S}{B'_{min}}) = S(n + m\frac{B_{min}}{B'min})$$

Wenngleich es nicht gewährleistet ist, das Hin- und Rückweg die gleichen Knoten benutzen, kann dennoch davon ausgegangen werden, dass $B_{min} \cong B'_{min}$. Daraus resultiert

$$BDP \leq S(n + m) = SN$$

Hierbei ist S die Größe des Paketes auf dem Hinweg und N die Summe der Hops des Hin- und Rückweges. Für das Staufensterlimit (CWL) gilt somit:

$$CWL \leq N$$

2.1.2 Genauere Abschätzung der oberen Grenze unter Berücksichtigung der benutzen MAC-Schicht

Die in Abschnitt 2.1.1 hergeleitete Abschätzung $CWL \leq N$ ist noch unabhängig vom eingesetzten MAC-Layer gewählt. Um nun auch die Gegebenheiten des in der drahtlosen Kommunikation quasi als Standardprotokoll eingesetzen IEEE 802.11 in die Abschätzung mit einzuarbeiten, muss man zuerst die Versuchsanordnung betrachten. In Abb. 4 ist ein solcher Versuchsaufbau schematisch dargestellt. Die Knoten liegen hier auf einer Geraden in Abständen von jeweils 250m. Die Sendereichweite eines Knotes beträgt ebenfalls 250m, so dass ein Knoten genau beide Nachbarn erreicht. In Abb. 4 bedeutet dies, dass der Knoten E mit den Knoten D und F Datenpakete austauschen kann. Die Störreichweite beträgt hingegen 500m, d.h. mit den beiden nächsten Nachbarn C und G kann zwar kein direkter Datenaustausch

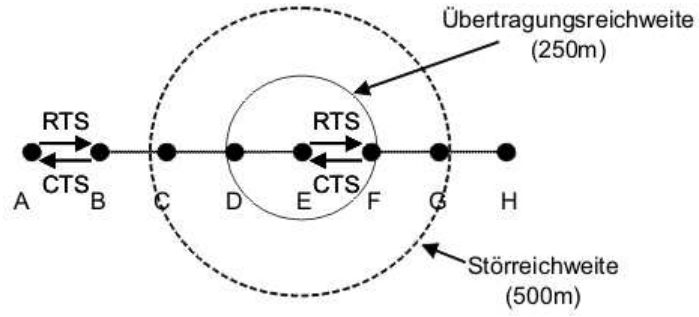


Abbildung 4: Versuchsanordnung: statische,symmetrische Kette; Abstand der Knoten 250m; Übertragungsreichweite 250m; Störreichweite 500m

von E durchgeführt werden, jedoch erkennen beide Knoten eine Sendung des Knotens E und sind daher während eines Sendevorganges des Knoten E selbst nicht in der Lage eine Transmission zu initiieren. Durch diese Anordnung der Knoten ist es immer nur einem Viertel der Teilnehmer möglich, eine Sendung zu starten. Verfeinert man nun die aus Abschnitt 2.1.1 bekannte Abschätzung $CWL \leq N$, so gilt

$$CWL \leq kN, \; mit \; k < \frac{1}{4}.$$

Eine zweites Problem der drahtlosen Netzwerke ist, dass die Daten- und Quittungspakete auf Ihrem Weg zwischen Sender und Empfänger an einer Stelle der Route kollidieren können. Es kommt in diesem Fall zu Verzögerungen in der Auslieferung der Daten- oder der Quittungspakete. Um diese potentielle Staugefahr zu reduzieren, halbiert man die Anzahl der möglichen Pakete. Die Abschätzung für das Staudfensterlimit lautet daher abschließend:

$$CWL \leq kN, \; mit \; \frac{1}{8} < k < \frac{1}{4}$$

2.1.3 Testreihe

Um die in Abschnitt 2.1.2 entwickelte Abschätzung $CWL \leq kN, \; mit \; \frac{1}{8} < k < \frac{1}{4}$ zu präzisieren, wurde die in Abb. 4 vorgestellte Versuchsanordnung eingesetzt. Pro Testreihe wurde eine fixe Anzahl von Knoten zwischen 1 und 15 vorgegeben. Innerhalb einer solchen Testreihe wurde das Staufensterlimit pro Testlauf linear um 1 erhöht. Startwert hierfür war 1, maximaler Wert des Staufensterlimits war 20. Gemessen wurden in Abhängigkeit des eingesetzten Staufensterlimits sowohl die insgesamt übermittelten Pakete sowie die durchschnittliche Größe des Staufensters. Abb. 5 und 6 zeigen die Ergebnisse für 6 bis 10 Hops.
Aus Abb. 5 wurde daraufhin der empirisch ermittelte, optimale Wert des Staufensterlimits in Abhängigkeit zu der Anzahl der Hops ermittelt. Als Beispiel sei hier für die 6-Hops-Kette erwähnt, dass deren optimales Staufensterlimit mit 2 identifiziert wurde. Trägt man nun alle so ermittelten CWL-Werte gegen N (=RTHC) auf, erhält man den Graphen aus Abb. 7. Sieht man von den beiden Fällen $N = 1$ und $N = 2$ ab, so kann man mit einer Geraden $k = \frac{1}{5}$ die übrigen CWL-Werte näherungsweise gut beschreiben. Die beiden Staufensterlimit für $N = 1$ und $N = 2$ weisen eine Abweichung von der Regel $CWL \leq \frac{1}{5}N$ auf. Dies ist in der Tatsache begründet, dass die in Abschnitt 2.1.2 aufgeführten Punkte bei einem Netzwerk mit zwei oder drei Knoten nicht zum Tragen kommen.
Insgesamt kann man also den Wert des Staufensterlimits entweder direkt aus der Gleichung

$$CWL \leq \frac{1}{5}N$$

berechnen oder aus der Tabelle 1 ablesen.

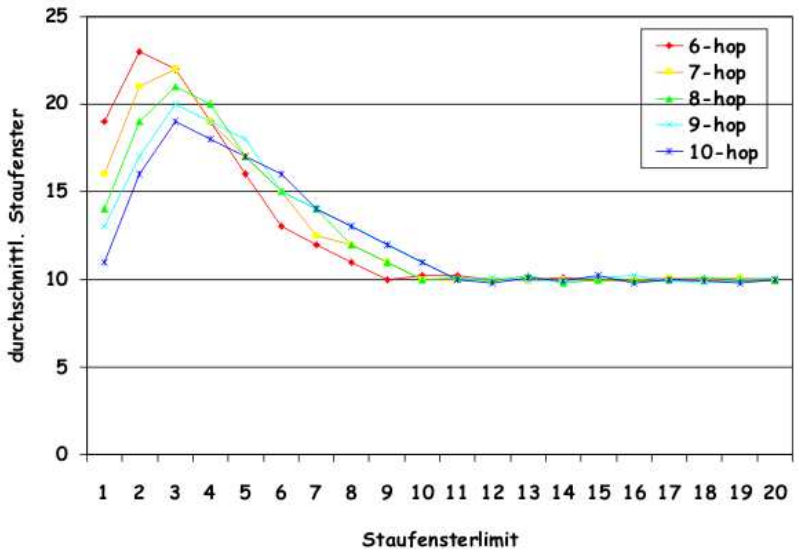


Abbildung 5: Zusammenhang der erfolgreich übertragenen Pakete mit der Größe des Staufensterlimits in einer Ketten-Topologie mit 6 bis 10 Knoten

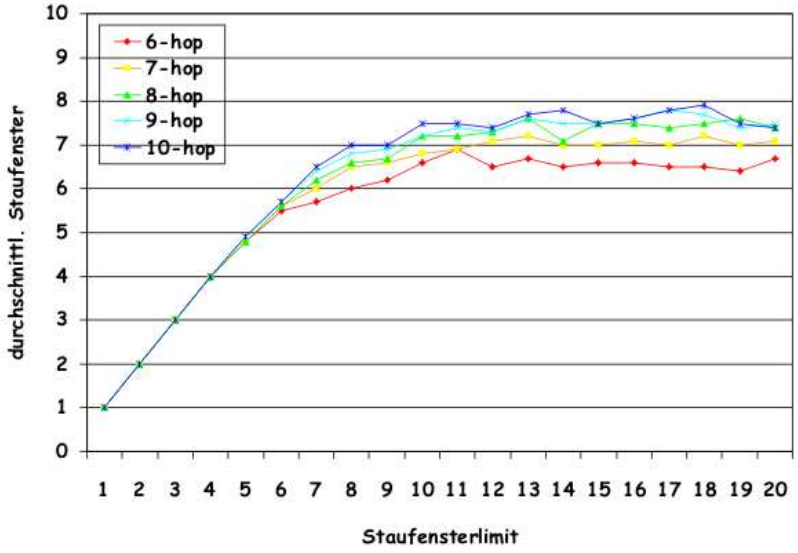


Abbildung 6: Zusammenhang der durchschnittlichen Staufenstergröße mit der Größe des Staufensterlimits in einer Ketten-Topologie mit 6 bis 10 Knoten

2.1.4 Leistungsfähigkeit

Die aus Abschnitt 2.1.3 hervorgegangene Gleichung $CWL \leq \frac{1}{5}N$ soll nun in einer dynamischen Testumgebung eingesetzt werden. Es soll ermittelt werden, inwieweit die Adaption des Staufensterlimits Veränderungen der Leistungsfähigkeit in einem MANET im Gegensatz zu einem unbeschränkten Staufensterlimit bewirkt. Das Testfeld umfasst folgende Kenndaten:

- räumliche Ausdehnung: 1500m x 300m

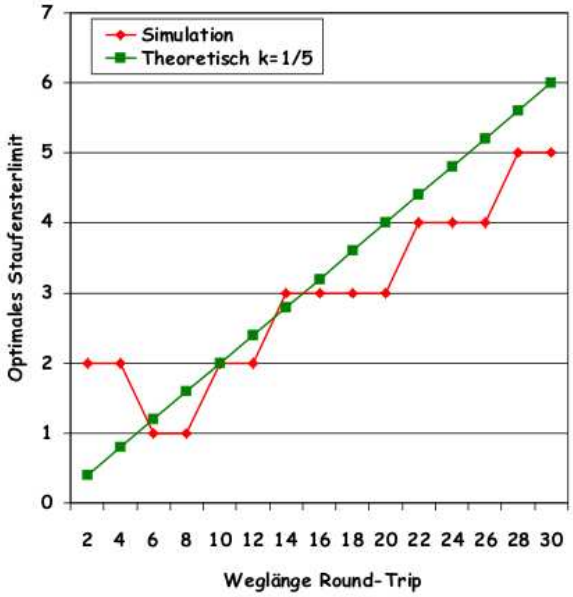


Abbildung 7: Gemessene optimale Staufensterlimits in Abhängigkeit der RTHC (Round-Trip Hop Count)

N (=RTHC)	CWL
$N \leq 4$	2
$4 < N \leq 8$	1
$8 < N \leq 12$	2
$12 < N \leq 20$	3
$20 < N \leq 26$	4
$26 < N \leq 30$	5

Tabelle 1: Simulationsergebnis des optimalen Staufensterlimits des TCP-Flusses

- Anzahl der Knoten: 30
- maximale Geschwindigkeit eines Knotens: 5 m/s
- Pausenzeit (d.h. die Zeit, die ein Knoten innehält ehe er ein neues Ziel und eine neue Geschwindigkeit wählt): 0s

Das Ergebnis wird in Abb. 8 wiedergegeben. Eine Leistungssteigerung ist klar erkennbar und kann mit ca. 8% bis 16% beziffert werden.

2.2 Einsatz von TCP Friendly Rate Control (TFRC) in MANET

TFRC wurde ursprünglich für das Festnetz entwickelt, um die gravierende „Sprünge“ des Slow-Start-Algorithmus in den Übertragungsraten abzumildern. Das TFRC beruht auf der Gleichung

$$X = \frac{s}{R\sqrt{\frac{2bp}{3}} + t_{RTO}(3\sqrt{\frac{3bp}{8}})p(1+32p^2)}$$

Hierbei kommen folgende Kennwerte zum Einsatz:

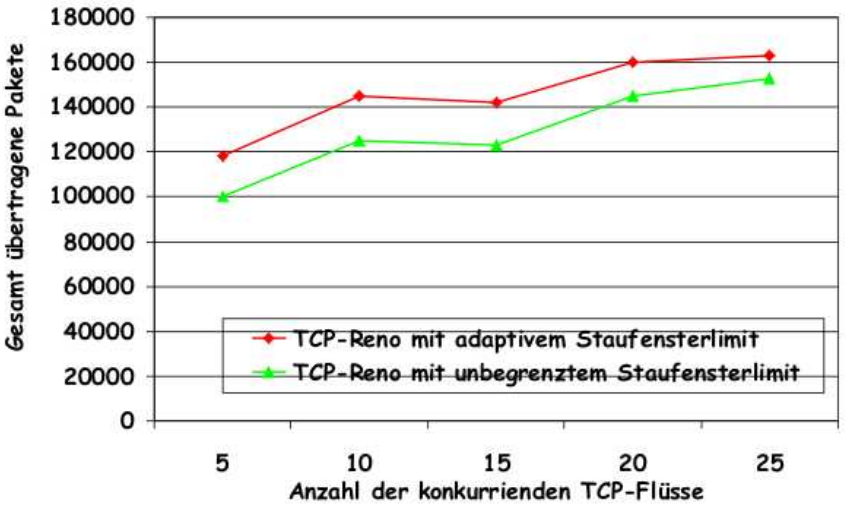


Abbildung 8: Leistungsvergleich in dynamischer Testumgebung (50 Knoten, 1500x300m, max v = 5 m/s, Pause = 0s

- X: Übertragungsrate
- s: Paketgröße
- R: RTT (=Round Trip Time)
- p: Fehlerereignis-Rate
- t_{RTO}: TCP Retransmission timeout-Wert
- b: Anzahl der Pakete, welche mit einem einzigen ACK bestätigt werden können

Die Fehlerereignis-Rate p ist die Anzahl der Pakete zwischen zwei Verlust-Ereignissen und wird wie folgt berechnet:

$$p_n = \frac{1}{\hat{\Theta}_n} \quad und \quad \hat{\Theta}_n = \sum_{l=1}^{L} w_l \Theta_{n-l}$$

$\hat{\Theta}_n$ ist der gewichtete Durchschnitt der Verlustintervalle beim n-ten Verlustereignis. Θ_{n-l} ($l = 1$ *to* L) sind die letzten L Verlustintervalle und w_l ($l = 1$ *to* L, $\sum_{l=1}^{L} w_l = 1$) sind die Gewichtungen. Im TFRC ist $L = 8$, d.h. die letzten 8 Intervalle werden betrachtet. Die Gewichtungen lauten (1.0, 1.0, 1.0, 1.0, 0.8, 0.6, 0.4, 0.2), d.h. die letzten vier Verlustintervalle haben die gleiche Gewichtungen, danach werden diese linear abgeschwächt.

2.2.1 Leistungsvergleich zwischen TCP und TFRC in MANET

Obwohl TFRC in Festnetzen durchaus TCP-Fairness zu seinen Eigenschaften zählen kann, haben die Messungen in der MANET-Testumgebungen gezeigt, dass TFRC unter dynamischen, drahtlosen Umständen die Eigenschaft der TCP-Fairness verliert. Auf lange Sicht gesehen, erreicht TFRC nur Durchsatzraten von 20% bis 80% des klassischen TCP. Je höher die Dynamik der Topologie desto konservativer und ineffizienter wird TFRC. Betrachtet man die kurzfristige Fairness, so erkennt man auch hier deutliche Unterschiede in der

Durchsatzrate zwischen TCP und TFRC (vgl. Abb. 9). TCP ist im Allgemeinen aggressiver in der Erhöhung und Absenkung seiner Übertragungsrate, was aber die Möglichkeit einräumt, auf veränderte Netzwerksituationen schneller zu reagieren. Gerade in mobilen Adhoc-Netzwerken zeichnet sich diese Eigenschaft gegenüber dem TFRC und dessen Ziel, eben diese starken Schwankungen in der Übertragungsrate zu minimieren, aus.

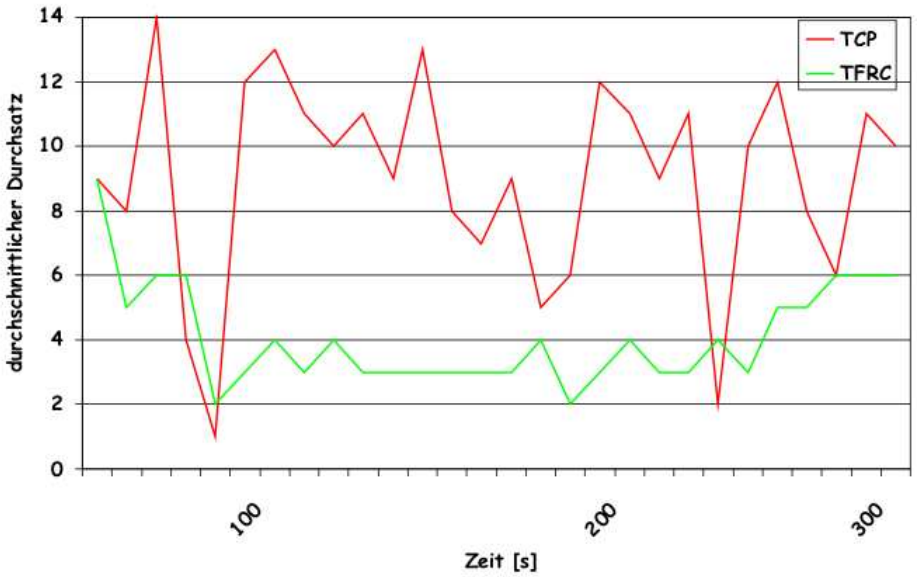


Abbildung 9: Leistungsvergleich zwischen TCP und TFRC in MANET-Testumgebung (7 statische Knoten in Reihe, Hintergrundverkehr 10 Kbps)

3 Zusammenfassung

Abschließend kann man folgende Beurteilung der beiden Ansätze abgeben:

- Adaptives Staufensterlimit
  Dieser Ansatz brachte 8% bis 16% mehr Leistung im Gegensatz zu einem unbegrenztem Staufensterlimit. Der Ansatz geht auf die Problematiken eines drahtlosen Netzwerkes ein und kann auf Veränderungen in der Route (d.h. Veränderungen der Anzahl der Hops) reagieren. Ein Nachteil ist sicherlich, dass das Staufensterlimit als Teil der TCP-Staukontrolle im klassichen TCP nicht implementiert ist und somit auf den derzeit im Einsatz befindlichen drahtlosen Hosts unbekannt ist.

- TFRC
  Hier hat die Adaption eines im Festnetz bewährten Verfahren nicht wie gewünscht funktioniert. Die Fehlerratenvorhersage des TFRC ist nicht in der Lage, die Problematiken eines drahtlosen Netzwerkes und speziell eines mobilen Adhoc-Netzwerkes zu berücksichtigen. Die eigentliche Idee der geringeren Schwankungen in der Übertragungsrate wurde auch hier umgesetzt. Jedoch bleiben die tatsächlichen Übertragungsraten hinter denen des klassischen TCP zurück.

Die beiden Ansätze sind derzeit nur zwei unter vielen Überlegungen, das Problem der TCP-Staukontrolle in mobilen Adhoc-Netzwerken zu lösen. Hierbei gehen die Ansätze durchaus von verschiedenen Richtungen an die Problematik heran. Eine nicht zu unterschätzende

Eigenschaft des neuen TCP muss die vollständige Kompatibilität zum bisherigen TCP sein. Das klassische TCP hat inzwischen einen derartigen Verbreitungsgrad, dass ein flächendeckender Austausch des TCP nicht in Frage kommen kann. Dass aber zumindest im mobilen Bereich etwas geschehen muss, ist allen Beteiligten klar. Die klassische TCP-Staukontrolle ist nicht für den Einsatz in MANET konzipiert worden und beweist dies täglich aufs Neue.

Literatur

[Nahr03] Kai Chen; Yuan Xue; Klara Nahrstedt. On Setting TCP's Congestion Window Limit in Mobile Ad HOc Networks. *in Proc. of IEEE International Conference on Communications (ICC 2003)*, Mai 2003.

[Post81] Jon Postel. *Transmission Control Protocol.* IETF Network Working Group. Request for Comments 793. 1981.

Abbildungsverzeichnis

1 Verlauf des Staufensters bei Slow-Start-Algorithmus 119

2 Ablaufdiagramm des Fast ReTransmit/Fast ReCovery-Algorithmus 120

3 Ablauf des Slow-Start-Algorithmus unter Verwendung des Staufensterlimits . 122

4 Versuchsanordnung: statische,symmetrische Kette; Abstand der Knoten 250m; Übertragungsreichweite 250m; Störreichweite 500m 124

5 Zusammenhang der erfolgreich übertragenen Pakete mit der Größe des Staufensterlimits in einer Ketten-Topologie mit 6 bis 10 Knoten 125

6 Zusammenhang der durchschnittlichen Staufenstergröße mit der Größe des Staufensterlimits in einer Ketten-Topologie mit 6 bis 10 Knoten 125

7 Gemessene optimale Staufensterlimits in Abhängigkeit der RTHC (Round-Trip Hop Count) 126

8 Leistungsvergleich in dynamischer Testumgebung (50 Knoten, 1500x300m, max v = 5 m/s, Pause = 0s 127

9 Leistungsvergleich zwischen TCP und TFRC in MANET-Testumgebung (7 statische Knoten in Reihe, Hintergrundverkehr 10 Kbps) 128

Tabellenverzeichnis

1 Simulationsergebnis des optimalen Staufensterlimits des TCP-Flusses 126