INTERNATIONAL COMPUTER SCIENCE INSTITUTE

1947 Center St. • Suite 600 • Berkeley, California 94704-1198 • (510) 666-2900 • FAX (510) 666-2983

Network Simulations for

A Mobile Network Architecture for Vehicles

Jörg Widmer*

TR-00-009

May 2000

Abstract

In this paper, we present a network architecture for vehicle communication based on Mobile IP. The special network environment of a car allows optimizations but also requires modifications of existing approaches. We identify these issues and discuss the integration of possible solutions into the framework. For example, location information provided by a car navigation system can be used to improve handoff decisions and connectivity. To evaluate the architecture, simulation studies were carried out with the *ns* Network Simulator. This paper also gives an overview of the necessary modifications and extensions to ns and additional tools to simplify future research in this area.

*This material is based upon work supported by DaimlerChryler Research, Palo Alto. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of DaimlerChryler Research.

Contents

Chapter 1

Introduction

Access to networks such as the Internet and "being accessible" indepently of the current point of attachment to the network is becoming more and more important with an increasing mobility of Internet users. In this project, we investigate an architecture for seamless access to multiple wireless networks from a vehicle[1]. Support for multiple wireless networks is necessary since it is unlikely that a single network can meet all possible requirements. We assume that many wireless networks will be available, differing in size and location of the coverage area, provided bandwidth, access costs, and network characteristics. Making use of several of these networks will allow for a much higher flexibility, which is particularly important since vehicles can move at high speeds and cover large areas. It is unlikely that a car will always stay within the service area of a single provider.

A multitude of applications are conceivable for such an architecture. These applications primarily fall into two categories:

- applications providing information to the driver and passengers in the vehicle
  - local information about travel conditions, weather, restaurants, parking, etc. in addition to the information provided by the car navigation system
  - reservations, ticket booking, etc.
  - (voice controlled) e-mail, datebook/calender, etc.
  - network services for external devices such as notebooks and PDAs
- applications that retrieve state information to allow monitoring of the vehicle
  - constant monitoring of car sensors
  - remote maintenance (e.g. adjustment of ignition timing)
  - providing detailed information about the car to a mechanic *before* the car is turned in, to speed up repair time

While the driver will have to communicate to the applications over a speech interface to prevent distraction from the traffic, passengers can use common user interfaces such as touch sensitive displays.

The different types of applications have specific communication requirements. WWW access results in bursty TCP connections that usually transmit a few KBytes. These connections last only for a short time

[1] In particular, we focus on car communication.

and the quiescence intervals between connections vary. Monitoring sensor data of the car from the "outside" produces a completely different communication pattern. Here, the necessary data is likely to fit into a single data packet. Since old status information is made obsolete by new information, it may be sufficient to just wait for the next status update in case a packet is lost instead of requiring reliable data transport. Thus, status updates should be sent as single UDP packets in regular time intervals. Sending audio data over a wireless connection again has completely different communication characteristics. While it also results in small data packets that are sent in regular intervals, their size varies to a considerably due to compression. Furthermore, audio stream often have strict timing constraints.

By simulating these communication patterns, the suitability of the mobile architecture for this type of traffic and the efficiency of current mobility protocols can be investigated. As pointed out in [PF97], simulations as well as real testbeds are necessary to fully evaluate architectures with respect to robustness and scalability. Thus, this project can serve as a basis for a later implementation of the framework.

In the second chapter, we present the communication framework and protocols for the mobile network architecture. The third chapter gives an overview of the network simulator that was used for the project and discusses necessary changes and extensions. Chapter 4 contains the simulation scenarios and results which are used to evaluate the envisioned architecture. Finally, we give conclusions and an outlook in Chapter 5.

Chapter 2

Network Architecture

Since Mobile IP [JM96, HJ96, Per96] possesses most of the necessary features for mobile communication, it is used as a basis for the wireless network architecture. Nearly all of the concepts of Mobile IP can directly be transferred to the new architecture.

2.1 Mobile IP

In Mobile IP, a Mobile Host always has a Home Agent (e.g. the router of the subnetwork the host usually is attached to). This Home Agent keeps track of the current point of attachment of the mobile host. Whenever the mobile host changes the network it is connected to, it has to register a new care-of address (COA) with the Home Agent. This association of the Mobile Host's home address and the current care-of address is called *binding*. The care-of address can either be the address of a Foreign Agent (e.g. a wireless base station node) that has agreed to provide services for the Mobile Host or the new IP address of the Mobile Host itself. A care-of address can be acquired either through stateless or stateful address autoconfiguration.

Traffic *to* the Mobile Host is always passed through the Home Agent, then tunneled to the care-of address and in case of a Foreign Agent care-of address forwarded to the Mobile Host by the Foreign Agent. Outgoing traffic *from* the Mobile Host does not need to go through the Home Agent but the host can directly communicate with Correspondent Hosts. By using a Home Agent as an intermediary, Correspondent Hosts do not need to know the Mobile IP protocol or the current location of the Mobile Host. The forwarding of packets to the current address of the Mobile Host is transparent for other hosts.

By using this architecture, a Mobile Host can roam between Foreign Agents and its Home Agent. When the Mobile Host leaves the service area of its current Foreign Agent and registers with a new Foreign Agent, the Home Agent has to be informed about the change of address. This procedure is called *handoff*. During such a handoff, it is possible that the Mobile Host loses connectivity for a short period of time. To provide smooth handoffs and speed up the handoff process, the use of several care-of addresses is possible where wireless service areas overlap. However, only *one* of those addresses can be registered with the Home Agent (primary care-of address).

The Mobile IP architecture is well suited for Mobile Hosts that change their point of attachment only over relatively large time intervals. When fast moving Mobile Hosts are forced perform a large number of handoffs per time interval, registering a care-of address with the Home Agent causes too much overhead and a too high delay, which in turn results in decreased protocol performance. Several approaches to solve this problem and to provide a more local, hierarchical form of mobility management are discussed in [SK99, Val99, CP96, BZCS96].

2.1.1 IPv6 vs. IPv4

Mobile IP exists as an additional feature to IPv4. In the next version of IP, IPv6, mobility support is directly built into the protocol. This facilitates the use of Mobile IP and allows several optimizations as compared to IPv4. Among others, the following mechanisms are part of IPv6:

- In IPv6, *route* optimization to avoid triangle routing is built into the protocol and no longer just an additional feature. Whenever the Mobile Host receives packet that was tunneled by the Home Agent, it sends a binding update to the original sender (CH). The Correspondent Host can then communicate directly with the Mobile Host.
- The care-of address is used as the source address for the IP packets instead of the home address. The home address is now specified in in the Home Address Destination Option. Thus, the use of care-of addresses is still transparent above the IP layer, but the approach now supports techniques that require local source adresses in a subnetwork (e.g. ingress filtering).
- The use of care-of addresses as source addresses also facilitates wireless multicasting since the Mobile Host (as a sender) does not have to tunnel packets to Home Agent anymore.
- The notion of Foreign Agents is no longer needed since the Mobile Host can operate in any IPv6 environment.
- IPsec (sender authentication, data integrity check, etc.) is used for secure address binding updates and acknowledgements.
- IPv6 supports bidirectional movement detection (i.e. not only the Mobile Host but also a Foreign Agent can detect loss of connectivity).
- By using the additional IPv6 header options for adressing, IP-in-IP encapsulation is only necessary for packets that are forwarded to the Mobile Host by the Home Agent. In IPv4, encapsulation is necessary for all data packets.
- The Mobile Host uses *anycast* instead of broadcast for Home Agent address discovery which reduces the load on the subnetwork.

These additional features improve usability and efficiency of Mobile IP. Particularly route optimization can result in a considerable reduction of network load and delay. On the other hand, these features require some additional state information to be kept in the Mobile Host and the Correspondent Host. The Correspondent Host caches the current care-of address of the Mobile Host. Whenever the Mobile Host performs a handoff, it not only has to inform the Home Agent of its new care-of address but also all Correspondent Host with which it communicates (binding update). If the Mobile Node would not inform other nodes in time, Correspondent Hosts could send data to an outdated care-of address. Thus, the Mobile Node has to keep a list of all Correspondent Hosts to which it sent binding updates.

2.2 Mobile Network Architecture

The similarity of the mobile network architecture and Mobile IP becomes clear in the architecture overview in Figure 2.1. Packets from a Correspondent Host to the Mobile Host are routed to the corresponding Home Agent. The Home Agent looks up the address of the Mobile Host and tunnels the packet. Since all packets are sent over the Home Agent (unless route optimization is used), the Home Agent can be a performance bottleneck when the number of Mobile Hosts increases. In this case, a hierarchical structure of multiple Home Agents can improve performace and scalability. Hierarchical Home Agents can distribute the Mobile Hosts among themselves to balance the load.

As mentioned in the introduction, the architecture aims to support multiple wireless technologies and thus has to be able to use multiple service providers. These service providers assume the role of the Foreign Agents. A Mobile Host is assigned an IP address by its current Foreign Agent and can now be reached using that address. Thus, the care-of address is the address of the Mobile Host (*co-located* care-of address).[1]

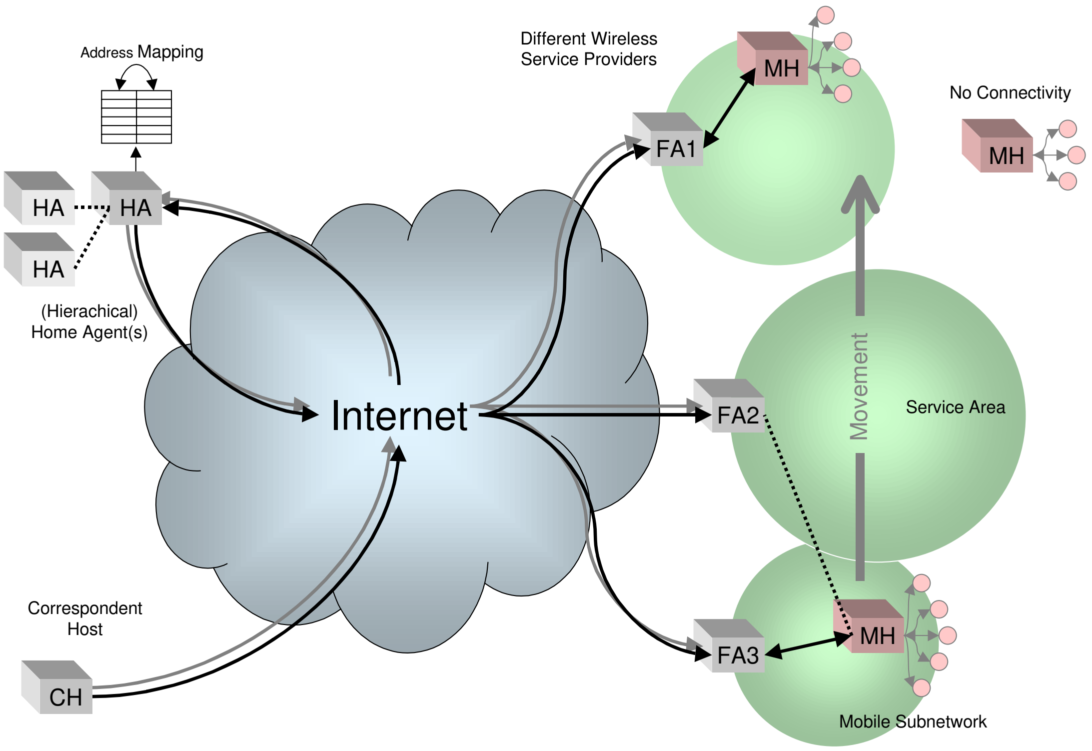


Figure 2.1: Architecture Overview

The internal structure of the service providers with routers and base stations is not modeled in Figure 2.1, since neither the Mobile Host nor the Home Agent need to know about it. The type of mobility support (be it Mobile IP, Cellular IP or a proprietary protocol) used by the service provider is transparent for the Mobile Host. In other words, the framework only handles *vertical* handoffs from one service provider to another

[1]When the care-of address is the address of the Foreign Agent and the Agent forwards incoming packets to the corresponding Mobile Host, the Mobile Host is said to have a *foreign-agent* care-of address.

without taking *horizontal* handoffs within the service area of a single provider into account.

2.2.1 Addressing

In our architecture, the Mobile Host is not just a network node but a vehicle with possibly many communication enabled devices. Unless, the service provider assigns not only a single address but a (small) address space to the Mobile Host, individual adressing of the devices is difficult. When a service provider uses IPv6, its large address space makes the assignment of multiple adresses possible while such a solution is unlikely with IPv4.

In case only a single adress is available for the Mobile Host, devices that only initiate communication can use *Network Address Translation* [EF94] instead of an individual address. This service would then have to be provided by the Mobile Host. One application where NAT is sufficient is to offer network connections to external devices (e.g. the notebook or PDA of a passenger). A second solution for individual addressing (apart from fundamental changes to standardized network protocols) is to define an additional adressing scheme between Home Agent and Mobile Host. This requires only a few additional bytes in the payload for the device address and possibly some flags. The actual data is encapsuled in the packets. The schemes is transparent for all network entities apart from the Home Agent and the Mobile Host. However, such an approach has a major drawback. Since the Home Agent has to encapsulate the data, it always has to be part of the network path from Correspondent Host to Mobile Host. Route optimization is no longer possible. This not only results in longer delays and an increased load on the network but also exacerbates the performance bottleneck at the Home Agent. With route optimization, only the first few packets of a connection are sent via the Home Agent and subsequent data packets are transmitted directly to the Mobile Host. This allows the Home Agent to handle a much larger number of Mobile Hosts.

Some optimizations are possible by reducing the number of addressable devices in the car. Instead of being able to communicate with each single sensor in the car, it might be sufficient (and much more efficient) to communicate with the vehicle bord computer and request the data of a particular sensor. This also allows to combine information into a single status packet which further increases performance.

2.2.2 Optimizations

Adaptive Applications

Wireless technologies vary considerably in offered bandwidth and changes of a few orders of magnitude are possible. Thus, it is necessary that applications adapt to changed network conditions as fast as possible. When a handoff to a low bandwidth provider is necessary but it takes a Voice Over IP application several seconds until it adjust its bandwidth from high quality audio with 64KBit/s to low quality voice with only 8KBit/s, it is possible that due to congestion no traffic at all can be sent from or to the Mobile Host for that period of time. When also Mobile IP control packets are lost (e.g. binding updates), performance can suffer for much longer than the time of congestion.

To better be able to deal with such situations, applications profit from additional information that can be provided by the mobile node. The mobile node has information about the maximum bandwith as well as concurrent applications which allows an applications to determine its “optimal” throughput. A further option might be to install a bandwidth manager in the Mobile Host. The bandwith manager has to ensure that Mobile IP control packets are transmitted with a higher priority than all other traffic and it can discard packets of applications that excessively use scarce bandwidth even before they are passed down the protocol stack.

Optimized Handoff Algorithm

Base stations have to announce their presence by sending periodic beacon messages. Upon reception of a beacon message, a mobile node can request Foreign Agent services from that base station. When the mobile node leaves the service area of the base station it has to perform a handoff to a new base station. A very basic handoff algorithm is to negotiate Foreign Agent services with a new base station as soon as the old base station becomes unavailable. This is detected by the mobile node when it does not receive beacon signals for a certain amount of time. To prevent that base stations are erroneously considered unreachable because beacon messages are lost, the timeout interval for base stations should be a multiple of the beacon period. However, when the mobile node loses its current base station and a handoff is necessary, the mobile node cannot communicate until the base station timer expires and the mobile node negotiates Foreign Agent services with a new base station. This results in a communication dropout of up to a few seconds.

The network architecture presented in this report focuses on car communication. Location information which can be obtained via a GPS system can be used to optimize the handoff algorithm for Mobile IP. By keeping track of its current location, the mobile node can predict when a handoff is likely to happen and negotiate Foreign Agent services with a new base station beforehand. This effectively prevents a communication dropout. When the mobile node is within a certain range of the border of the coverage area, it tries to perform a handoff to a base station that is located closer to the mobile node. The mobile node can obtain this information in several ways:

- The signal strength of the beacon messages can be used to estimate how close the mobile node is to the border of the service area. When it falls below a certain threshold value the mobile node performs a handoff to the nearest base station it can hear. During the handoff, the mobile node is still reachable via the old base station.
- When base stations send information about their maximum service range and their location in the beacon messages, the mobile node can compute its distance to the base station. By comparing the distance to the service range, it can estimate the likeliness of leaving the base stations service range and perform a handoff in time should it be necessary.
- Map information about base stations and their service ranges can be stored in a database at the mobile node. When a mobile node identifies a base station, it can look up its location and service range and perform the same calculations as described above.

Additional information of a car navigation system, such as destination and the anticipated route can be used to further optimize the algorithm.

2.2.3 Wireless Technologies

The following wireless communication technologies are among those available in the Bay Area. Although other technologies exist, the presented communication technologies cover a sufficiently wide range of different characteristics of wireless channels to allow for a comprehensive evaluation of the architecture. With regard to a later realization of the architecture, we chose similar parameters for the wireless connections to the mobile node in the simulations.

Wireless LAN

- bandwidth: 1-10 MBit/s
- frequency: 2.4 GHz
- range (indoor/outdoor): 20-200m / 2-15km

Wide-Area Wireless Network (Metricom)

- bandwidth: 30 KBit/s (raw bandwidth 100 KBit/s)
- frequencies: 2.4 GHz (radio-to-radio), 900 MHz (radio-to-poletop)
- coverage area: SF Bay Area, Washington D.C., Seattle
- geographical routing (GPS in base stations)

CDPD

- bandwidth: 5-12 KBit/s (raw bandwidth 19.2 KBit/s)
- frequencies: 800/900/1900 MHz
- coverage area: good coverage in most of the U.S.

Chapter 3

The Network Simulator *ns*

The *ns* Network Simulator is an event driven simulator for computer networks and network protocols. It was chosen as the simulation tool for this project because of its modular and open architecture. Since it is widely used in the research community, a large number of network components are available for *ns*. By reusing these components, the development of complex architectures can be considerably facilitated. Amongst others, *ns* supports the following technologies:

- Point-to-point connections, LANs, wireless links, satellite links
- Different queueing schemes (DropTail, RED, etc.)
- IP, Mobile IP
- Multicast (DVMRP, PIM, etc.)
- TCP, UDP, and several experimental transport protocols
- RTP/RTCP, SRM
- QoS (InterServ, DiffServ)
- Applications (Telnet, FTP, WWW-like traffic, etc.)
- Math support (random number generation, integrals, etc.)
- Network emulation (i.e. interaction of the network simulator with a "real" operating network node)

The current version of *ns* contains modules from several universities and research groups such as UCB, LBNL, ISI/USC, Sun, Xerox PARC, etc.

The simulator framework uses a split-language programming approach. OTcl, an object oriented version of Tcl, is used for the control structure and the description of the simulation scenarios. Also the scheduling of events and the dynamic configuration of network components during the simulation is usually done in OTcl. The actual core of the simulator (i.e. low level event processing, per-packet actions such as forwarding, etc.) is written in C++ to allow for a fast simulation of large scenarios. While this approach is very flexible, it also adds complexity. To use the simulator it is necessary to have knowledge in OTcl as well as C++. Particularly debugging in both languages simultaneously can be a difficult task.

Currently, no visual scenario generation tools exist for *ns* and scenarios are usually programmed by hand. However, simulation results can be visualized using the Network Animator NAM.

3.1 Scenario Construction

Generally, the generation of scenarios in the TCL script file follows a certain order:

- Instantiation of the event scheduler object
- Creation of the topology
- Specification of the communication patterns
- Scheduling of dynamic changes to the scenario during the simulation
- Tracing of events

The topology consists of network nodes connected by links with a certain queueing strategy, delay, and throughput. To exchange packets between nodes, agents have to be attached to the nodes. Traffic sources such as applications can then use these agents to communicate with traffic sinks at other nodes. There are several types of agents, for example routing agents that decide to which link to forward a packet, agents for sending and receiving TCP or UDP packets, etc.

3.2 Support of Mobile IP and Wireless Communication

The original *ns* architecture only supported stationary nodes connected by wired links. Wireless nodes and channels were added at a later stage by a different research group with focus on the simulation of wireless ad-hoc networks. Their framework allows a very detailed modeling of wireless communication using Radio Propagation Models, Antennas, Link Layer, ARP, and MAC Layer protocols (e.g. IEEE 802.11), as well as ad-hoc routing protocols.

3.2.1 Wireless Nodes

Special wireless nodes have to be used to compose wireless networks. These nodes have additional features in comparison to ordinary wired nodes. Nodes have information about their location and they can move between locations with a specified speed. *ns* only supports linear movement of nodes. More complex movement patterns have to be constructed from linear segments.

For wireless communication, the notion of links between nodes is no longer valid. To be able to send packets between mobile nodes, they have to be attached to the same *channel*. Packets sent over the channel are distributed to all nodes on the channel.

Also, the internal structure of nodes was extended. While wired nodes hand packets directly to the corresponding agent which then processes the packets, packets at wireless nodes are passed through several additional layers. As depicted in Figure 3.1, a packet that is sent over a wireless channel is first handed to the Physical Layer (PHY). The Physical Layer uses a Radio Propagation Model (RPM) to determine the signal strength with which a packet is received. It then compares the signal strength to a *detect threshold* to decide if the receiving node can detect the packet at all. If the signal is too weak, the packet is marked as "not received". When the packet could be detected, the signal strength is further compared to the *receive threshold* to determine if the signal was strong enough to allow a correct interpretation. If the signal strength falls short of that threshold, the packet is marked as "received with errors". In any case, the packet is handed over to the Medium Access Control layer (MAC). The MAC layer discards packets that the node could not have received and packets that were received with bit errors.

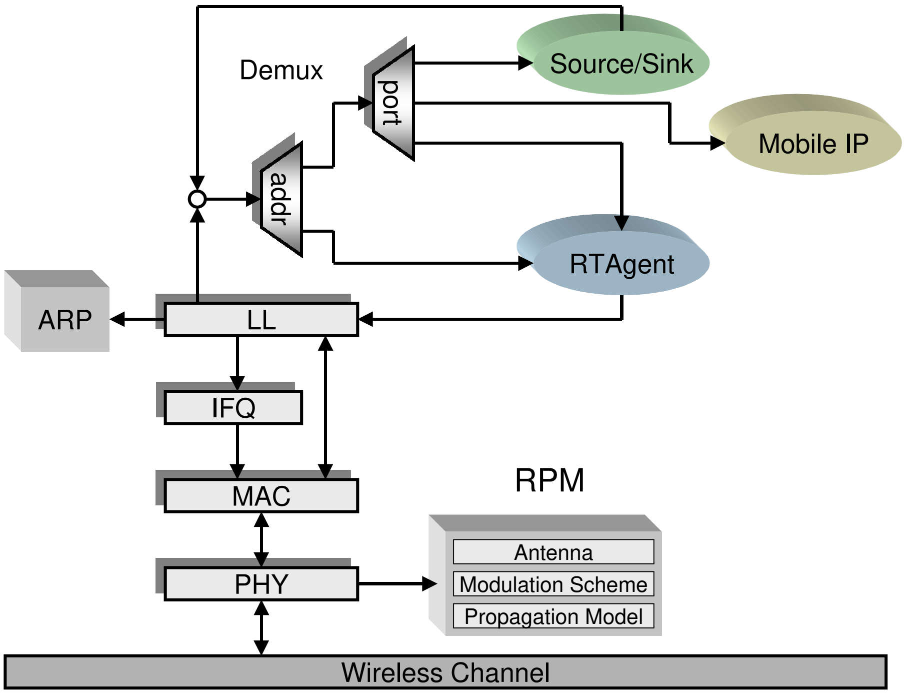


Figure 3.1: Structure of a Wireless Node

Since the wireless extension was targeted at mobile ad-hoc networks, wireless nodes always have an ad-hoc routing agent attached. Currently, *ns* supports the following ad-hoc routing protocols:

- Destination Sequenced Distance Vector Routing (DSDV)
- Dynamic Source Routing (DSR)
- Ad-Hoc On Demand Distance Vector Routing (AODV)
- Temporally-Ordered Routing Algorithm (TORA)

In addition to purely wireless nodes, so called *Base Station* nodes (BS) can be connected to wired links as well as wireless channels. They act as bridges between the wired and the wireless part of the network and are of paramount importance for the simulation of Mobile IP.

3.2.2 Packet Formats

The changes to *ns* for the support of wireless networks also resulted in a different trace file format for wireless traces. This was necessary since packets are not sent directly from source to destination, but all nodes on the corresponding channel can potentially receive the packet. Thus, at the point of time when a packet is sent it is not known which nodes will receive the packet. The wireless trace lines also display information about MAC and IP header fields and, if the packet is a TCP/UDP packet, additional information about the TCP/UDP header.

When postprocessing simulation traces, it is therefore necessary to support both trace file formats. Many

Trace File Format for Wired Links:

		end points		packet				IP address			packet
event	time	from	to	type	size	flags	flow ID	src	dest	seq. no.	uid

Trace File Format for Wireless Channels:

					packet			higher level headers			
event	time	_node_	layer	flags	uid	type	size	[MAC]	[IP]	[TCP]	[...]

Table 3.1: Trace File Formats

trace analysis tools for *ns* have not been adjusted and only work either with wireless traces or with wired traces. Even the Network Animator NAM is not yet fully capable of visualizing wireless simulations, although wireless support will be improved in the future. So far, NAM only supports the visualization of the movement of wireless nodes while packets over wireless channels are not displayed. This makes the tool rather unsuitable for the task.

3.2.3 Mobile IP

The Mobile IP implementation for *ns* was developed at a time when *ns* only supported wired networks. It could not be used together with the wireless extension of *ns* that was added later because of the changes to the structure of wireless nodes. While wireless support for Mobile IP was implemented at a later stage, the Mobile IP implementation was not completely redesigned as would have been necessary. The resulting architecture mismatch considerably aggravates the simulation of Mobile IP.

The implementation of Mobile IP includes the basic components (Home Agents, Mobile Nodes, Foreign Agents) and basic functionality such as registering with a new Foreign Agent. All base stations advertise their presence by sending beacon messages at a preconfigurable time interval. Mobile Nodes store the adresses of the base stations within range in a list. When no beacon message of a registered base station is received for a certain amount of time, the list entry times out and is removed. Mobile Nodes that have to perform a handoff because they left the service range of their current Foreign Agent chose a base station from the list as their new Foreign Agent. If the list does not contain any entries, the Mobile Node sends an Agent Solicitation Message. Base stations that receive this message have to send an advertisement which then allows the Mobile Node to register with them. The handoff is initiated with a Registration Request from the Mobile Node. The base station then forwards the request to the Home Agent of the Mobile Host. The Home Agent updates the care-of-address (COA) of the Mobile Host and installs a so-called encapsulator to tunnel IP packets to the mobile host via the base station. The Home Agent then sends a Registration Reply Message to the base station and the base station in turn informs the Mobile Node that the handoff was successful. From then on, the base station acts as the Mobile Node's Foreign Agent.

However, the handoff algorithm itself is kept very simple. Whenever the Mobile Node receives a beacon message from a Base Station, it sends a Registration Request and from then on uses the Base Station as a Foreign Agent. This results in a dropout until the new connection is established although the Mobile Node could still communicate with the rest of the network over its current Foreign Agent. The method also works only when the Mobile Node "hears" a single Base Station. As soon as service areas of Base Stations overlap,

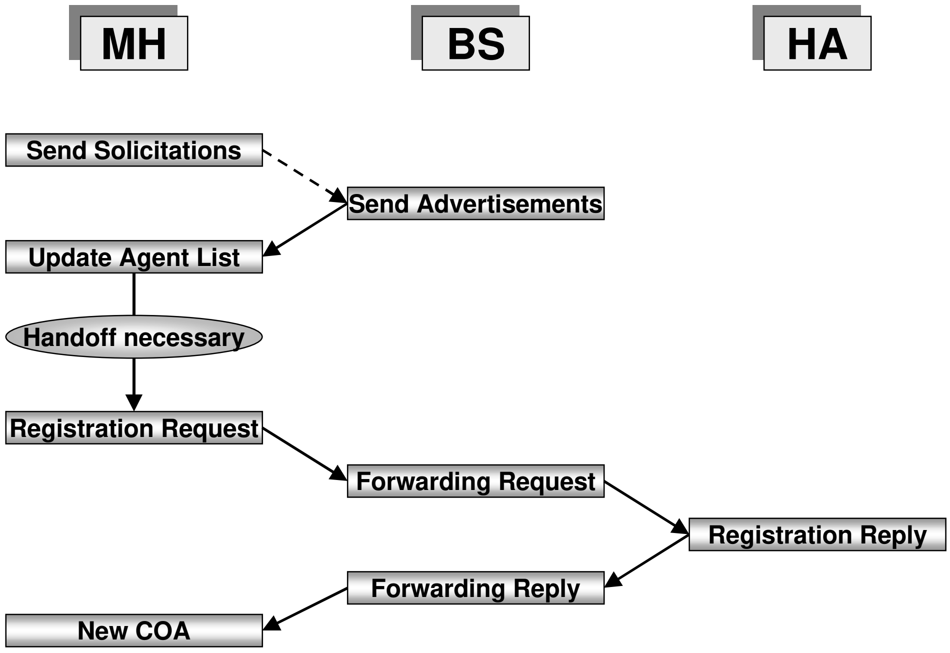


Figure 3.2: Handoff

the Mobile Node constantly switches between Base Stations and because of that often cannot establish any transport connection at all. Since a handoff to a new Base Station generates a certain amount of overhead, the simple handoff algorithm produces an unnecessarily large amount of Mobile IP control packets.

Extensions to Mobile IP such as route optimization are not implemented. Furthermore, packets *from* the Mobile Host are always routed over the Home Agent as well, so even sending packets directly from the Mobile Host to a Correspondent Host as specified in teh original Mobile IP draft (for IPv4) does not work. Also all enhancements to Mobile IP related to IPv6 are not supported by *ns*.

3.3 Modifications and Enhancements of *ns*

To be able to simulate the mobile network architecure of this project, *ns* had to be extended. The following modules were added or modified:

- new routing agent
- new radio propagation model
- modification of the Base Station and the Mobile Host module
- new handoff mechanism

3.3.1 NOAH Agent

Mobile Nodes always contain a full-fledged ad-hoc routing agent. Thus, communication over multiple wireless hops is always possible. In Mobile IP scenarios, this can produce unwanted results when Mobile Hosts use other Mobile Hosts as intermediate “routers” instead of communicating with their Foreign Agent. In many real-world scenarios, Mobile Hosts can only communicate with Base Stations but not with other Mobile Nodes. To make the simulation of these scenario possible, a Non-Ad-Hoc Routing Agent (NOAH) was developed. While nodes keep a list of other wireless nodes that are within range, they no longer have to exchange routing tables for multihop routing. NOAH Agents do not generate any routing related traffic but learn from the packets it receive which hosts are in range. This approach does not allow arbitrary communication patterns but works well with Mobile IP. First, packets have to be broadcasted to initialize the lists of nodes in range. Only then is direct communication between two wireless nodes possible. With Mobile IP, beacon messages (and possibly agent solicitations) have to be sent before nodes can exchange packets. Thus, the NOAH node lists can be initialized accordingly since those messages are always broadcast.

3.3.2 Simple-Distance Radio Propagation Model

The existing wireless module in *ns* models radio propagation with a high level of detail. Wireless nodes send with a certain signal strength and the attenuation of the signal is calculated based on Friss free-space attenuation at near distances and an approximate Two Ray Ground model at far distances [Rap96]. While this is useful for small scale simulations, it complicates simulations with a high abstraction level where only the range of base stations is known. For this reason, a very simple radio propagation model based only on the distance between wireless nodes was developed. Nodes within range of each other have full connectivity and nodes beyond that range have no connectivity at all. There is no signal attenuation or an increased error rate at the border of the service area but an abrupt change in connectivity. This propagation model should only be used when exact information about the signal frequency, signal strength, detect and receive thresholds, etc. are not available.

3.3.3 Mobile IP

Due to the previously discussed shortcomings, substantial changes to Mobile IP were necessary. The handoff procedure was redesigned to improve performance in network environments with a high handoffs frequency. Priorities can be assigned to base stations to establish an order of preference. The Mobile Host always performs a handoff to the base stations with the highest priority within range. This feature can be used to include bandwidth, cost, and provider preferences of the user (i.e. the mobile node) into the simulations. As an additional option, the handoff protocol can take the distance to base stations into account to reduce the number of handoffs and to ensure a timely handoff in order to prevent loss of connectivity.

When the Mobile Host receives a beacon message, it checks wether the corresponding base station is already in the list of base stations within range. If yes, the entry is updated and the expiration time of the entry is adjusted. If the base station is not in the list, an new entry is created. The Mobile Host then checks if it currently has a care-of-address. If not, it immediately sends a Registration Request to the base station that sent the beacon message. If the advertisement was from the current Foreign Agent, the Mobile Node reregisters to refresh the binding to the Foreign Agent. Finally, a Registration Request is also sent when the priority of the beacon is higher than the priority of the base station that is currently used as the Foreign Agent.

The Mobile Host caches the addresses of all base stations to which is sends requests. The Mobile Host only processes Registration Reply Messages from base stations in the cache. Other Reply Messages from nodes that the Mobile Host did not contact are discarded. When a valid Registration Reply arrives at the Mobile Host, it updates its care-of-address as well as other parameters (e.g. priority of the current base station) and clears the cache. Thus, cache entries have a timeout interval in proportion to the beacon interval.

The handoff procedure can be further optimized by taking the distance to base stations into account. For each Mobile Hosts, base stations are divided into the categories *near*, *distant*, and *unreachable*. The ratio of *near* range to overall range is called boundary parameter b. For example a b value of 0.2 means that the Mobil Host tries to perform handoff when its distance to the current base station is longer than 80% of the total service range. The overall preference structure is as follows:

- If base stations are within *near* range choose the one with the highest priority. The Home Agent is generally set to a higher priority than Foreign Agents to prevent unnecessary tunneling of packets.
- If several base stations fall into this category choose the nearest of them.
- If no base station is within *near* range choose the *distant* base station with the highest priority.
- If several base stations fall into this category choose the nearest of them.
- If no base station is reachable send a Solicitation Message

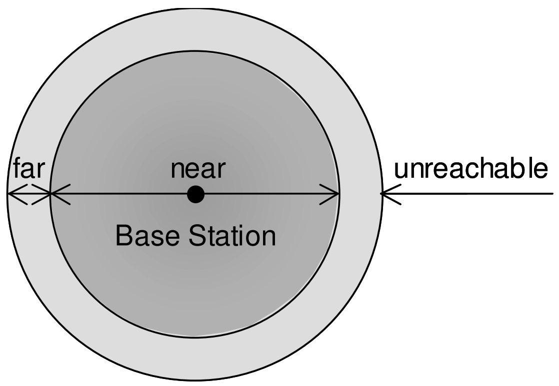


Figure 3.3: Coverage Area Boundary

Thus, as soon as the Mobile Host leaves the *near* range of the current Foreign Agent, it performs a handoff if there is another *near* base station. The Mobile Node does not perform a handoff when it receives a beacon message from a closer base station which has the same range classification. During the transition from one Foreign Agent to another, the Mobile Node retains its reachability via the old care-of-address (provided the old Base Station is still within “hearing range”).

3.4 Ad-Hockey

The Ad-Hockey tool for *ns* that was originally developed to visualize wireless ad-hoc networks. Hence, it neither supports wired links and nodes nor Mobile IP. To be able to use Ad-Hockey for the simulations if this project, these features had to be added. Furthermore, the extended version of Ad-Hockey now supports the original *ns* trace file format as well as the new wireless trace file format. Indication of packet events (e.g. send, receive, forward, drop) was improved and some optimizations to speed up the visualization of large scenarion were implemented.

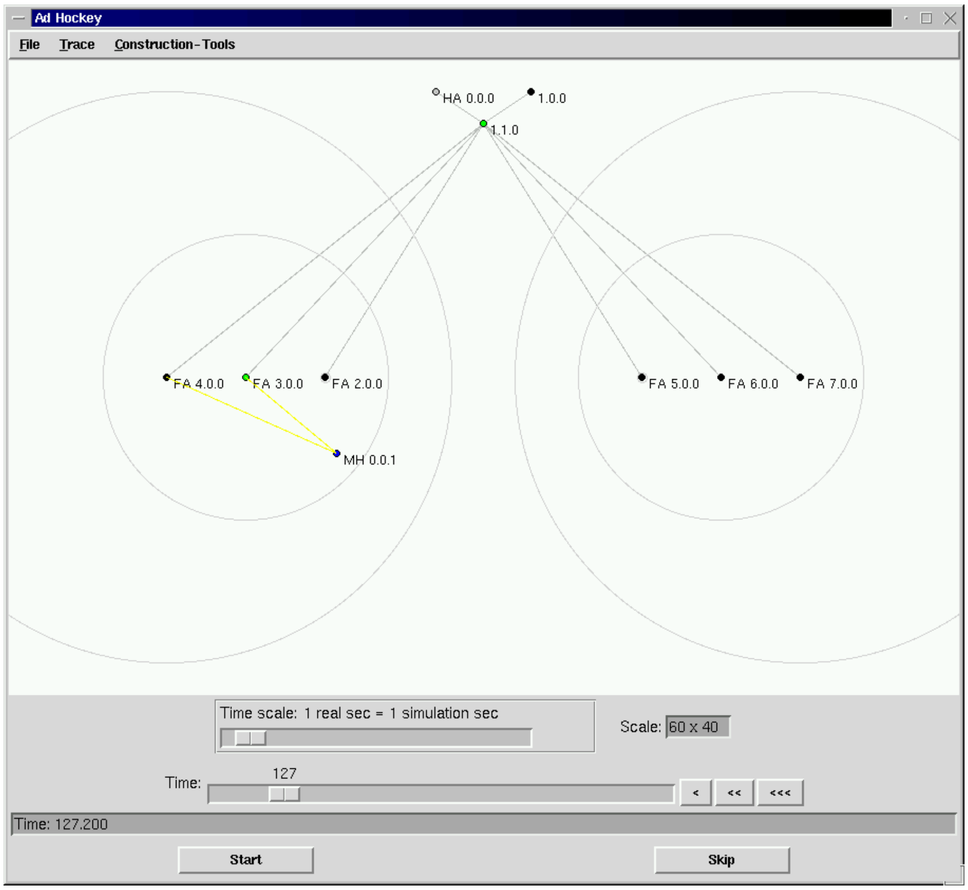


Figure 3.4: Ad-Hockey Screenshot

Ad-hockey has a simple user interface that allows to modify the current time and speed as well as start and stop the simulation. Furthermore, ad-hockey has (limited) support for scenario creation.

Chapter 4

Simulations

We created several simulation scenarios for *ns* to evalute the network architecture and study the effectiveness of the optimizations. Among the aspects we focused on are:

- Evaluation of the optimized handoff protocol
- Traffic analysis
- Scalability analysis

4.1 Evaluation of the Optimized Handoff Protocol

In Chapter 2.2.2, we presented an optimization for the handoff mechanism based on location information. The purpose of the following simulations is to determine optimal parameters for this mechanism and investigate the impact of the optimization on network traffic.

4.1.1 Throughput Comparison

The simulation topology consists of three large and one small service areas, as shown in Figure 4.1. The wired node 1.2.0 connects service area 1 and wired node 1.3.0 connects service areas 2, 3, and 4 to the rest of the wired network via node 1.1.0. The sending node in the scenarios is node 1.0.0. The Home Agent has the address 0.0.0 and is also connected to node 1.1.0. All wired links have a bandwidth of 128 KBit/s, a propagation delay between 10 ms and 50 ms, and use RED [FJ93] as queueing mechanism.

No.	Address	Radius	# Base Stations	Base Station Range	Priority	Color
1	2.0.0	100	29	25.0	1	grey
2	3.0.0	100	29	20.0	2	light yellow
3	4.0.0	50	13	16.0	3	yellow
4	5.0.0	100	29	30.0	1	grey

Table 4.1: Scenario Parameters

The duration of the simulation is 160.0 seconds. From time 10.0 on, the Mobile Host moves from its original position (120,320) towards position (520,140), thus traversing the coverage areas of all four service providers. It is always within reach of one or more base stations except from time 113.0 to time 117.0 before the handoff to base station 5.0.0. Starting at time 5.0 seconds, the sender (1.0.0) opens a connection to the mobile node and transmits packets with the maximum possible throughput. TCP as well as UDP are used as transport protocols.

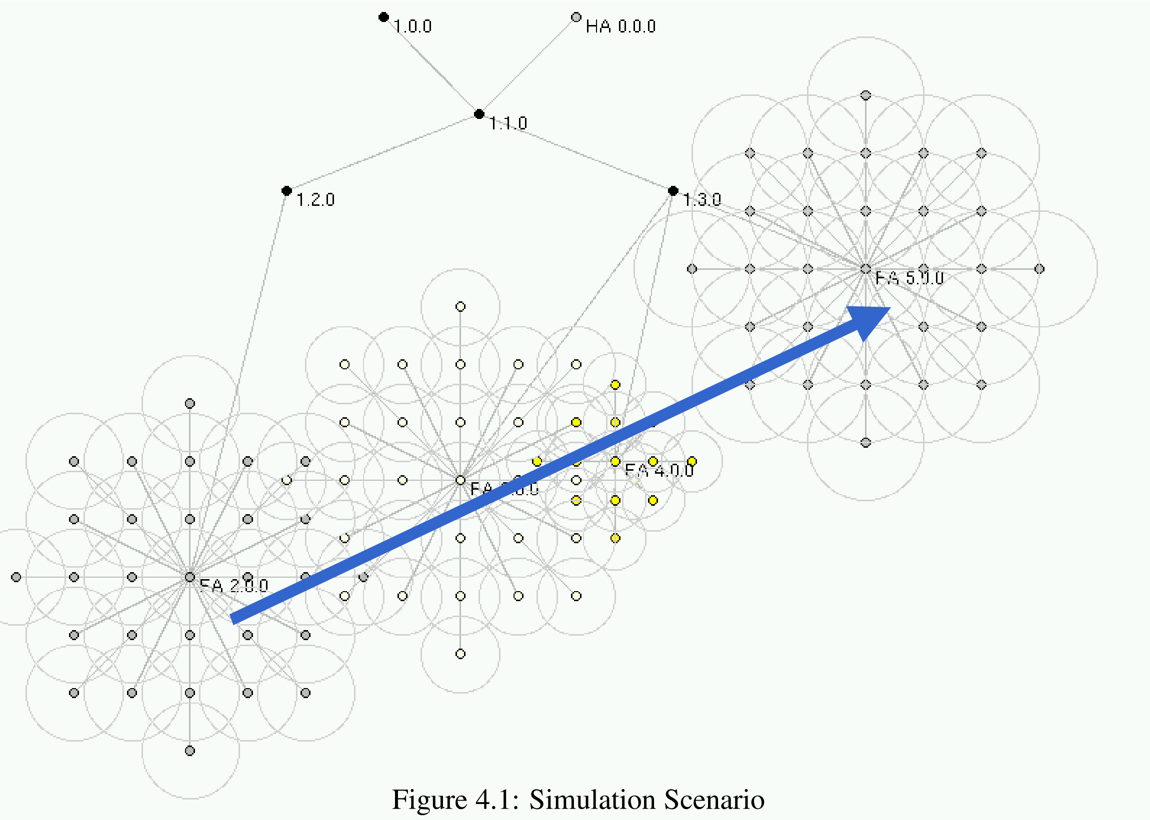


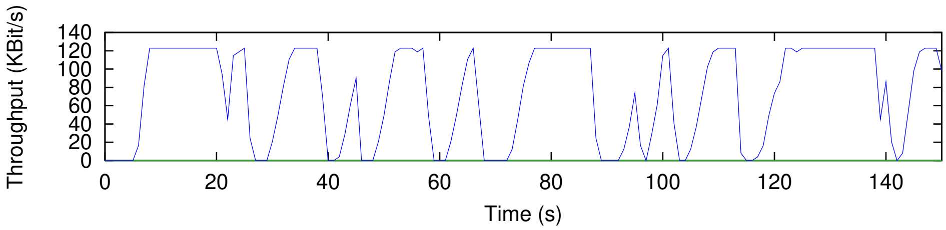
Figure 4.1: Simulation Scenario

TCP

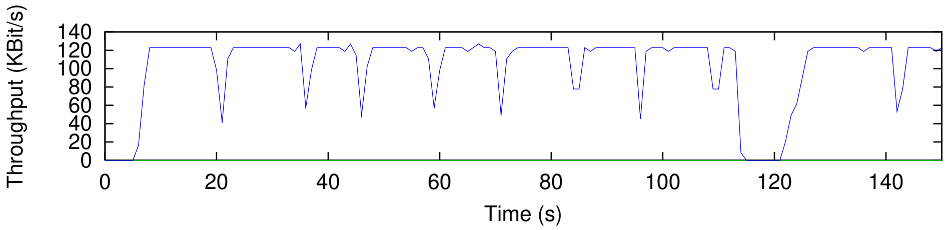
Since TCP reduces its bandwidth when it encounters packet loss (as can be the case during a handoff), it is very sensitive to connection dropouts. Furthermore, when no packets get through to the receiver for a longer period of time, TCP performs an exponential back off (i.e. the time before retransmission is tried again increases exponentially). Bandwidth reduction and exponential backoff are part of TCP's congestion control mechanism.[1] As long as no packet loss occurs, TCP slowly increases its throughput.

Figures 4.2a) and 4.2b) show the throughput of the TCP connection to the Mobile Host over the entire duration of the simulation. The simple handoff mechanism causes many triple duplicate ACKS and TCP timeouts which severely impair TCP throughput. All handoffs cause the loss of one or more packets. With the optimized handoff protocol on the other hand, no packets are lost due to handoffs at all. The periodic

[1]For a more thorough discussion see e.g. [Ste94].



a) Standard Handoff Mechanism



b) Optimized Handoff Mechanism

Figure 4.2: TCP Throughput Comparison

"dents" in the throughput are not caused by packet loss on the wireless link but reflect the typical sawtooth-like TCP throughput. TCP increases its rate until it reaches the maximum bandwidth of the link, experiences a loss, and then backs off again. In this simulation scenario, the utilization of the optimized handoff protocol increases overall TCP throughput by 45%.

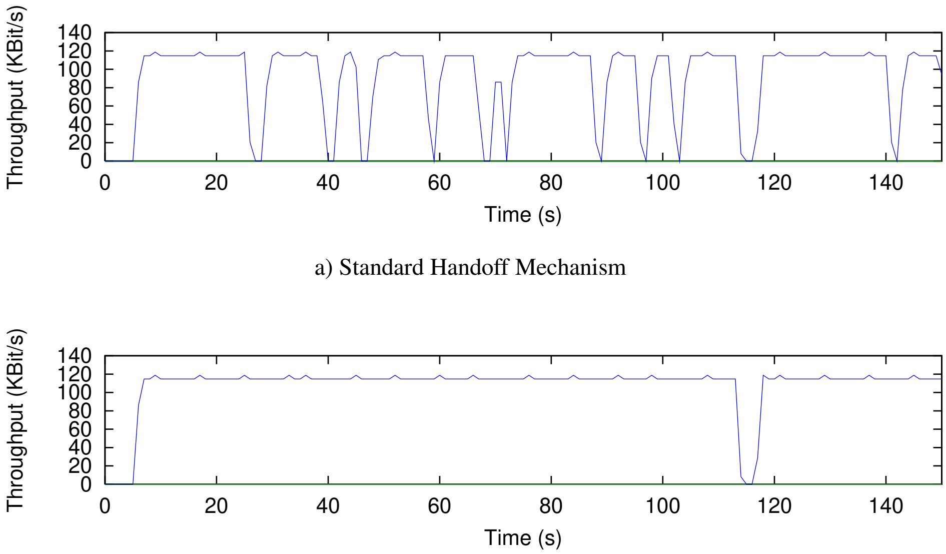
Surprisingly enough, the period with no connectivity from second 113.4 to second 116.9 results in a longer dropout of the TCP connection in the trace with the optimized protocol. However, this is not caused by the handoff algorithm itself but is a random result. Due to the loss of connectivity, TCP performs an exponential backoff and does not send packets for a few seconds. In simulation B with the optimized version of the handoff protocol, TCP sends a packet at time 116.96 which is lost. The exponential backoff mechanism causes TCP to wait for roughly four seconds before the retransmission of the lost packet.

Simulation A:

```
D 113.721459519 _68_ MAC  --- 15452 tcp 512 [a2 1 40 800]
D 113.773525519 _68_ MAC  --- 15455 tcp 512 [a2 1 40 800]
D 114.587995199 _68_ MAC  --- 15570 tcp 512 [a2 1 40 800]
r 116.980963479 _5_ MAC  --- 15918 tcp 512 [a2 1 60 800]
r 117.340283391 _5_ MAC  --- 15964 tcp 512 [a2 1 60 800]
```

Simulation B:

```
D 113.733876129 _68_ MAC  --- 17612 tcp 512 [a2 1 40 800]
D 113.752362129 _68_ MAC  --- 17617 tcp 512 [a2 1 40 800]
D 114.462023808 _68_ MAC  --- 17721 tcp 512 [a2 1 40 800]
D 116.964472833 _68_ MAC  --- 18045 tcp 512 [a2 1 40 800]
r 121.068531986 _5_ MAC  --- 18502 tcp 512 [a2 1 60 800]
```



a) Standard Handoff Mechanism

b) Optimized Handoff Mechanism

Figure 4.3: UDP Throughput Comparison

```
r 121.427971694 _5_ MAC  --- 18540 tcp 512 [a2 1 60 800]
```

The corresponding trace file for simulation A with the simple handoff protocol shows a very similar sequence of packets. However, the packet that is lost in simulation B is transmitted 16 ms later. During these 16 ms, the mobile node gets close enough to the next base station and is able to receive the packet. The connection is reestablished and packets start to flow again. Thus, the negligible difference in send time of 16 ms actually results in a four second difference in the inter-packet intervals. Since packets of different flows are never sent at exactly the same time, this is a random result not related to the handoff mechanism.

UDP

In contrast to TCP, the UDP transport protocol does not react to packet loss and thus often achieves a higher throughput than TCP. It never performs a backoff and always sends data when transmission is possible.[2] For this reason, UDP throughput as depicted in Figures 4.3a) and 4.3b) can be used to investigate connectivity. It presents the optimum throughput a TCP connection could achieve, if additional measures were taken to prevent TCP from backing off in case of non-congestion related packet loss.

As UDP throughput is not sensitive to packet loss, the difference in throughput between the simulations with the optimized and the normal handoff protocol is not as distinct. Sill, in case of the optimized handoff protocol UDP achieves an overall throughput that is 19% higher. Bandwidth utilization is exactly the link bandwidth and throughput is completely smooth apart from the short period of time without connectivity.

[2]This can be unfair to competing traffic and may lead to a congestion collaps [FF99].

No.	Address	Radius	# Base Stations	Base Station Range	Priority	Color
1	2.0.0	-	1	3.0	1	darkgreen
2	3.0.0	-	1	20.0	1	green
3	4.0.0	-	1	5.0	2	lightblue
4	5.0.0	-	1	5.0	2	lightblue
5	6.0.0	-	1	5.0	2	lightblue
6	7.0.0	-	1	5.0	2	lightblue
7	8.0.0	-	1	20.0	1	darkgreen
8	9.0.0	-	1	30.0	1	darkgreen
9	10.0.0	-	1	5.0	1	blue
10	11.0.0	-	1	5.0	1	blue
11	12.0.0	-	1	5.0	1	blue

Table 4.2: Scenario Parameters 1

No.	Address	Radius	# Base Stations	Base Station Range	Priority	Color
1	2.0.0	-	1	3.0	1	darkgreen
2	3.0.0	20.0	49	3.6	1	green
3	4.0.0	8.0	5	6.0	2	lightgreen
4	5.0.0	30.0	29	8.0	1	blue
5	6.0.0	20.0	13	8.0	1	darkgreen
6	7.0.0	5.0	5	3.0	1	lightblue
7	8.0.0	5.0	5	3.0	1	lightblue

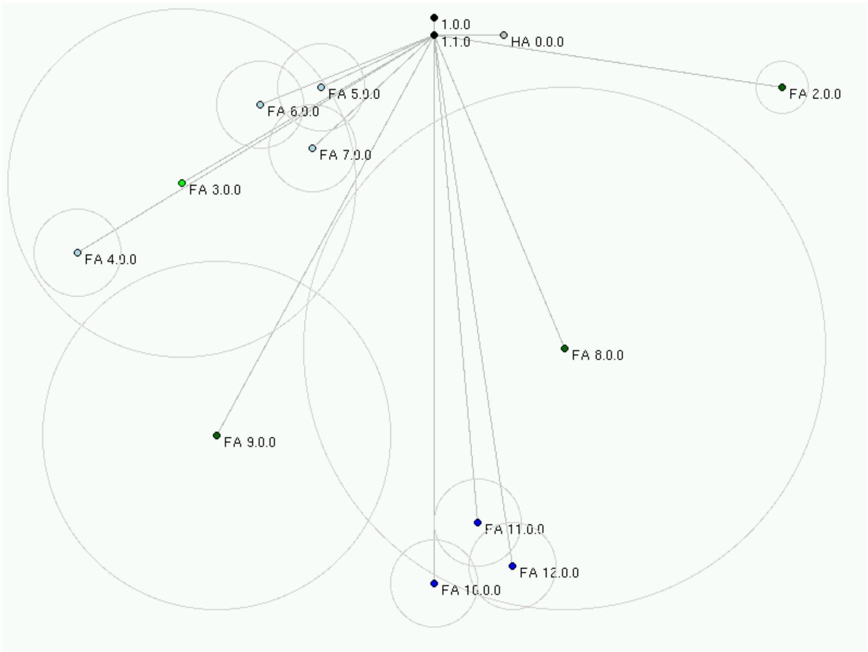
Table 4.3: Scenario Parameters 2

4.1.2 Coverage Area Boundary Size

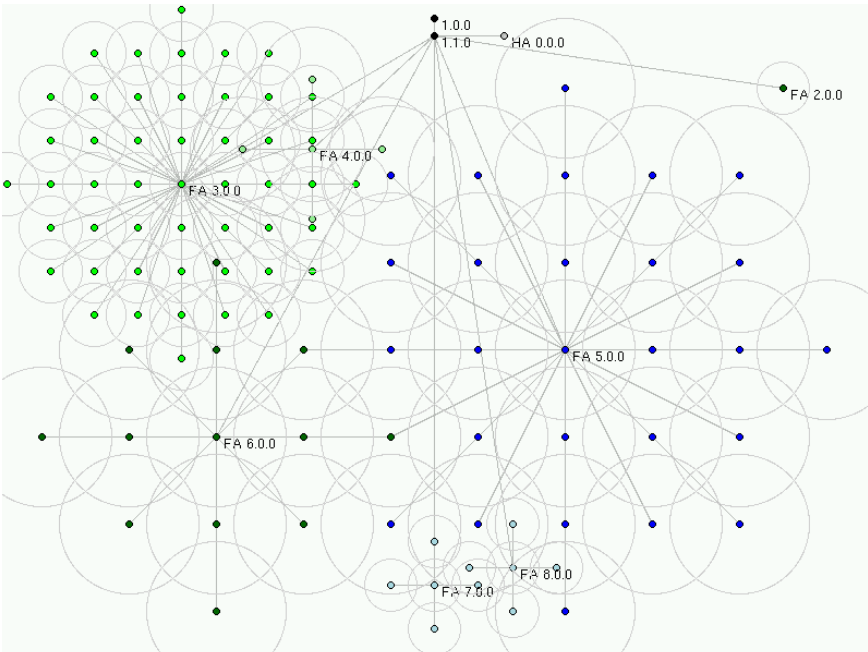
In the above simulations, the coverage area boundary parameter for the handoff protocol was set to the outer 20% of the radius of the service range (i.e. the Mobile Host stayed with its current Foreign Agent within 80% of the overall radius of the coverage area). This parameter greatly influences the performance of the handoff. If it set to a too low value, it cannot guarantee a smooth handoff without packet loss. If however it is set to a too high value, performance is decreased by the overhead caused by too many unnecessary handoffs. The following simulations compare protocol performance for different boundary parameter values and are aimed to find an optimal value.

For the simulations, the two basic scenario topologies depicted in Figures 4.4a) and 4.4b) were used. Scenario 1 has only a small number of base stations with large service areas, while in scenario 2 the number of base stations is much higher and the base station range is smaller. This results in different time intervals between consecutive handoffs. For each simulation run, these topologies were populated with 10 Mobile Hosts located at random positions. Throughout the simulations, the hosts moved to a random location at a random speed, thus creating a large number of different connectivity and handoff patterns.

We ran simulations with a boundary parameter b of 0.02, 0.05, 0.1, 0.2, and 0.5. Multiple simulations were conducted for each setting and the results were averaged to reduce the impact of outliers. The total number of

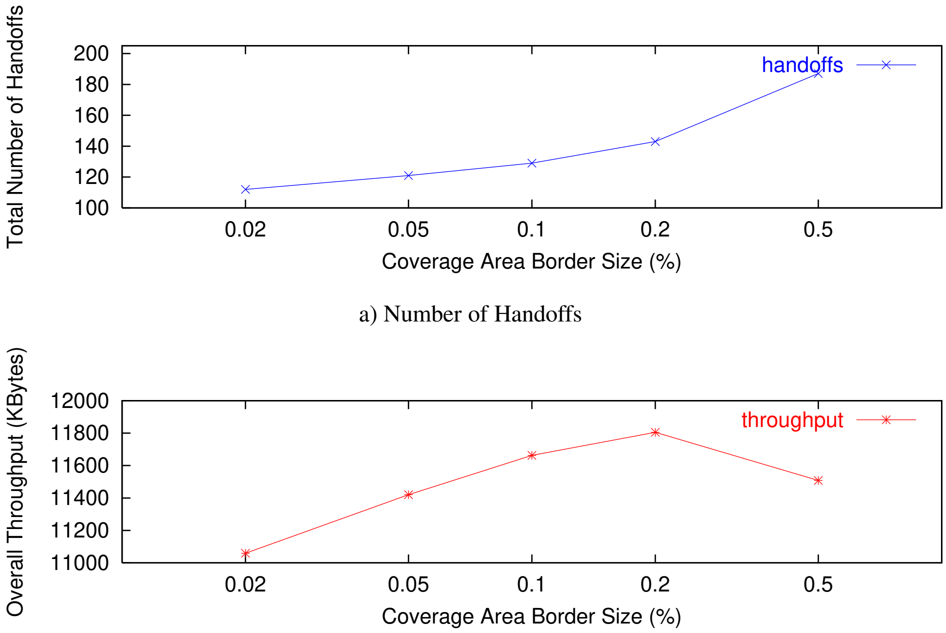


a) Scenario 1



b) Scenario 2

Figure 4.4: Scenarios for Coverage Area Boundary Simulations



a) Number of Handoffs

b) Throughput

Figure 4.5: Comparison of Different Service Area Boundaries

handoffs increases in proportion to the boundary parameter from 112 ($b = 0.02$) to 187 ($b = 0.5$). Also the overall throughput of the Mobile Hosts is proportional to the size of the coverage area boundary for smaller boundary parameter values. Handoffs are performed well before the Mobile Host leaves the coverage area and handoff related packet drops are rare. The highest throughput is achieved for $b = 0.2$. For higher values of b, the throughput decreases again, caused by a very high number of handoffs and the resulting overhead. Thus, optimal values for b are in the range of 0.1 to 0.2, depending on the actual overhead a handoff causes in a real environment.

4.2 Traffic Analysis

In the simulations in this section, we focus on the interaction of different types of traffic. The total simulation time is 920.0 seconds. The topology consists of two base stations with a large coverage area, two base stations with a medium coverage area, and two base stations with a small coverage area. The base stations and the movement pattern of the Mobile Host are arranged so that the Mobile Host starts out with a base station with a small coverage area and a high bandwidth (1 MBit/s). After 100 seconds, the Mobile Host starts to move and as a consequence has to switch to a medium size base station with a bandwidth of 30 KBit/s. When it leaves the service range of that second base station it is forced to swich to the base station with the largest coverage area and the lowest bandwidth (10 KBit/s). The Mobile Host completely loses connectivity from time 385.6 to time 529.3. After regaining connectivity, it performs a similar sequence of handoffs in reverse order (i.e. first to the large base station, then to the medium base station, and then to the small base station).

No.	Address	Radius	# Base Stations	Base Station Range	Priority	Color
1	2.0.0	-	1	0.3	3	black
2	3.0.0	-	1	9.0	2	black
3	4.0.0	-	1	18.0	1	black
4	5.0.0	-	1	0.3	3	black
5	6.0.0	-	1	9.0	2	black
6	7.0.0	-	1	18.0	1	black

Table 4.4: Scenario Parameters

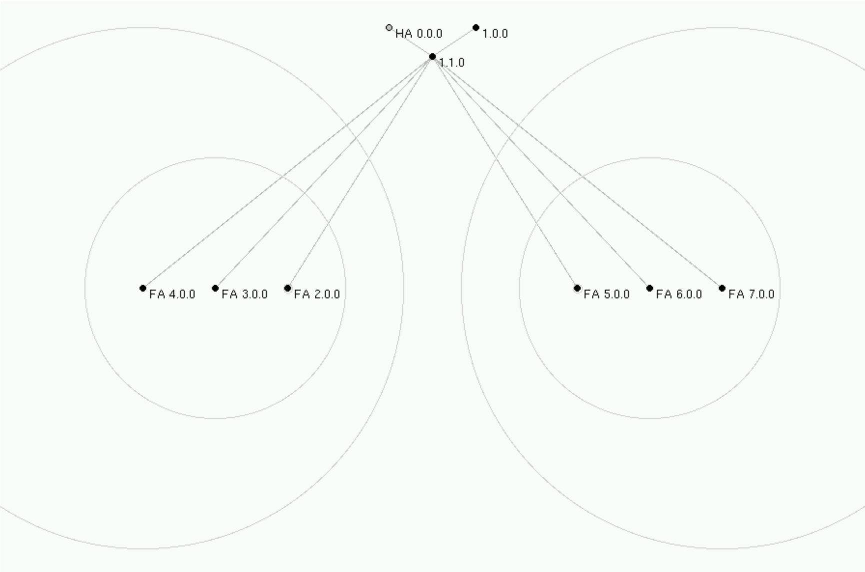


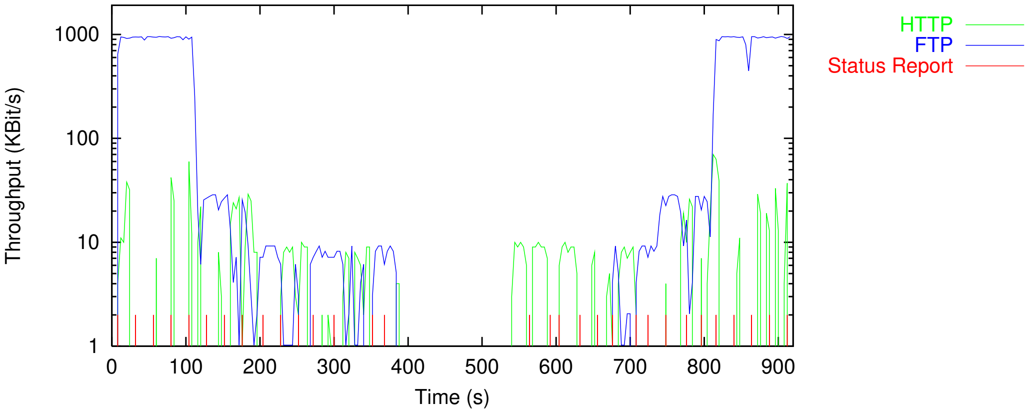
Figure 4.6: Traffic Analysis Scenario

Four different types of traffic were used in the simulations:

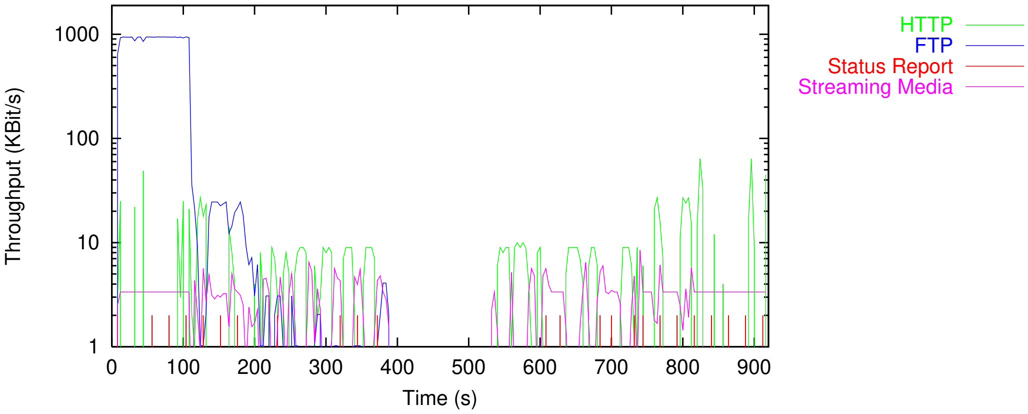
- FTP (TCP traffic with the maximum avaliable bandwidth)
- HTTP (short UDP bursts[3] with an average duration of 2 seconds, followed by a quiet period with an average duration of 15.5 seconds. The duration of send and quiet periods is exponentially distributed and the rate during send periods is about 4 KBit/s.)
- Status updates (a single 1 KByte UDP packet that is sent every 5 seconds. It could be used to transport information about the status of the vehicle to the Home Agent or other entities.)
- very low bandwith streaming media (UDP flows with raw data rates between 1 KBit/s and 3.2 KBit/s)

We created three scenarios, all of which use FTP and HTTP traffic as well as status updates. In the first scenario, no streaming media traffic is used. The second scenario has an additional flow of streaming media traffic with a bandwidth of 1 KBit/s and a packet size of 60 Bytes and the third scenario has a flow with a bandwidth of 3.2 KBit/s and a packet size of 30 Bytes.

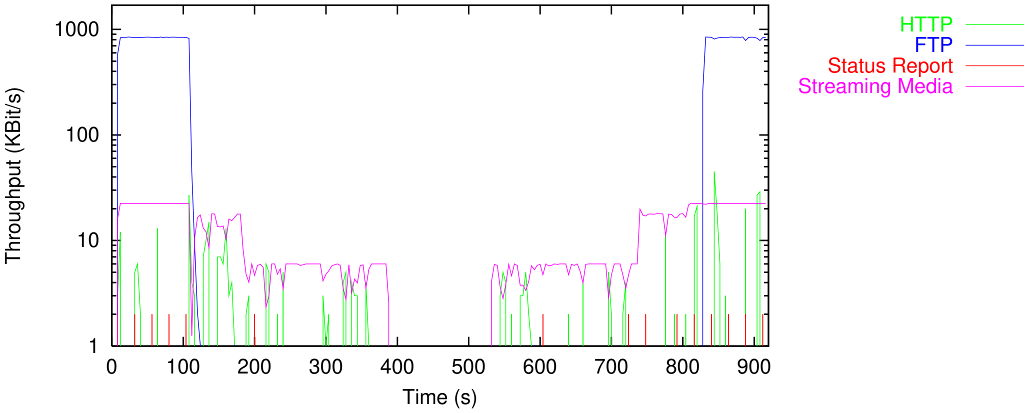
[3]The HTTP protocol actually uses the TCP protocol. However, the HTTP connections usually only transmit very few packets and TCP has no time to get to a state of equilibrium and perform congestion control. Thus, UDP can be used to approximate TCP traffic under such conditions.



a) No Streaming Media Flow



b) 1 KBit/s Streaming Media Flow



c) 3.2 KBit/s Streaming Media Flow

Figure 4.7: Traffic Analysis

Throughput results for the three scenarios are shown in Figure 4.7. In all three scenarios, in the beginning FTP consumes most of the available bandwidth since other flow are rate limited. This changes when bandwith is reduced to 30 KBit/s or 10 KBit/s. In the first scenario, the bursts of HTTP-like traffic reduce FTP throughput but FTP quickly recovers when the burst is over. The sensor updates have little impact on other traffic. After the period without connectivity, HTTP traffic and the status requests immediately resume sending, while this takes TCP more time because of the exponential backoff.

The results of the second scenario are similar as long as the available bandwith is 30 KBit/s or above. Below that bandwidth, the increased amount of UDP traffic prevents that TCP can reach its fair share of bandwidth. After the no-connectivity period, TCP is not able to regain bandwidth again. Whenever it attempts to send packets they are dropped because of the competing UDP traffic although unused bandwidth is available. However, with an overall bandwidth of 1 MBit/s TCP would eventually get enough packets through to the Mobile Host to reachs its equilibrium again and consume the available bandwidth. This can be seen in the third scenario, where TCP manages to resume the connection after the Mobile Host reaches the high bandwidth base station, although even less bandwidth is available due to the streaming media flow. When the Mobile Host is within the coverage area of the 30 KBit/s base stations, TCP does not get any throughput at all and also the HTTP-like traffic is severely harmed by the streaming media flow. Also most of the status updates are lost.

The bandwidth consumed by the streaming media flows is extremely low. Typical throughput would rather be in the range of 10 KBit/s to 64 KBit/s (i.e. 3 to 20 times higher). When running simulations with high bandwidth streaming media flows, Mobile Hosts are often not even able to get any connectivity although they are within the service range of a base station. Since the network nodes do not distinguish between normal data packets and Mobile IP control packets, even agent solicitations, beacon messages, and binding request/replies are frequently lost when flows send at a much higher rate than the available bandwidth. This prevents the Mobile Host from performing a handoff to the next base station or refresh the binding to the current Foreign Agent to keep up its connection.

The above simulations show, that bandwidth management is imperative for an acceptable throughput distribution of competing flows when bandwidth is scarce. In particular, Mobile IP control packets should always be given priority over normal data packets. Such a bandwidth manager could for example be implemented in the Mobile Host. Since the number of flows to the Mobile Host is typically low, keeping state information about each of the flows is unproblematic. The bandwidth manager can prevent packets from being passed down the protocol stack when they belong to flows that should be restricted. This effectively prevents packet collisions caused by the flows. Preferential treatment of Mobile IP control packets can be achieved by using two separate queues for data and control packets with a higher priority for the control packet queue.

4.3 Scalability Simulations

To determine how the framework performs when the number of Mobile Host per Home Agent is increased, we set up simulation scenarios with 1 to 256 Mobile Hosts. For these simulations, the scenario topology of the boundary size simulations was reused (Figure 4.4a)). The Mobile Hosts were randomly placed and assigned random movement patterns. Simulations were conducted with Mobile Hosts using TCP as well as UDP for the transport protocol.

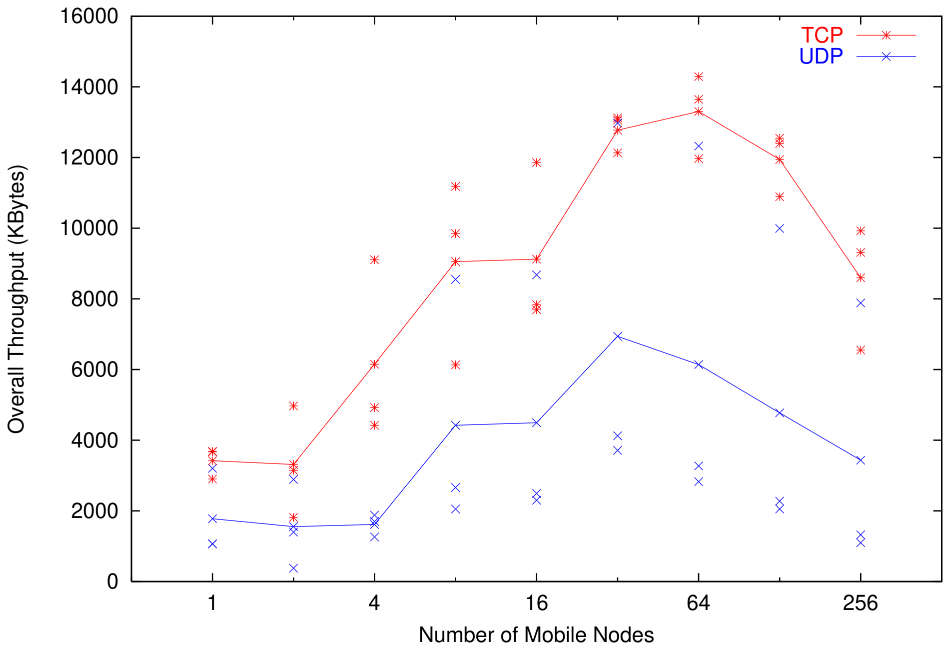


Figure 4.8: Overall Throughput for Different Number of Mobile Nodes

Since a potentially large number of Mobile Hosts can be in the same service area, it was necessary to restrict maximum UDP throughput in the simulations to prevent the loss of Mobile IP control packets. TCP's backoff mechanism makes this unnecessary for the TCP simulations. Without such a restriction, overall throughput in the UDP simulations is higher than TCP throughput when the number of Mobile Nodes is very low, but as soon as the number increases, the above mentioned phenomenon of lost Mobile IP control packets drastically reduces throughput.

Figure 4.8 shows overall throughput for all mobile nodes. TCP and UDP throughput both increase in proportion to the number of nodes for smaller numbers of nodes. Peak throughput is achieved for 32 nodes (UDP) and 64 nodes (TCP) respectively. However, this is not an approximation for the scalability of the whole framework since these simulations assume, that a single Mobile Host can potentially consume all of the available bandwidth of a base station. In real-life architectures, this is usually not the case and base stations can handle multiple Mobile Hosts without reduced performance. The optimal number of Mobile Hosts for a topology increases in proportion to the number of clients a base station can handle without reduced performance.

The problem of the Home Agent being a bottleneck (in particular since the *ns* implementation of Mobile IP does not use route optimization) can be solved by using a cluster of hierarchical Home Agents. These can be distributed over the network topology to achieve an optimal distribution of the load of the network. In addition, a load balancing algorithm can be used to assign Mobile Hosts to Home Agents with a low load.

Chapter 5

Outlook and Conclusions

The simulations of the communication framework that were conducted for this project show that it is well suited as a basis for a vehicle communication architecture. In many cases, modifications to existing mechanism can greatly improve performance. When bandwidth is scarce, a bandwidth manager can help to considerably improve connectivity and inter-flow fairness. It is particularly important to ensure the delivery of Mobile IP control packets since otherwise the connectivity of the Mobile Host suffers. Such a bandwidth manager is not yet implemented in *ns*. An implementaion will be necessary for further in-depth simulations of the framework and to assess possible performance improvements.

Additional location information often available in vehicles via GPS can be used to optimize the handoff process. This leads to throughput improvements for flows that use some form of congestion control. The handoff mechanism presented in this report only takes the location into account. The method can be further optimized, when information about velocity, destination, and anticipated route can be used. This information is often available, for example through a car navigation system. The Mobile Host can then preferentially perform a handoff to the base station with which it will have connectivity for the longest period of time to reduce the total number of handoffs that is necessary.

Also further improvements to the simulation evironment are desirable. So far, Mobile Hosts just use one communication channel and only the base stations differ in available bandwidth. In reality, however, the Mobile Host would have to use multiple wireless devices that are on different wireless channels and have different network interfaces. Basic support for multiple interfaces in a mobile node exists in *ns* but has to be extended for the framework.

Before the realization of the framework, it is also advisable to conduct further simulations to study transport protocol behavior. Improved network protocols can be used between Home Agent and Mobile Host that offer congestion control while being insensitive to losses on the wireless channel. A comparison of some of these approaches can be found in [BPSK97].

However, the simulations we presented in this report are an important first step towards the realization of the communication framework.

Bibliography

[BKA+98] Eric A. Brewer, Randy H. Katz, Elan Amir, Hari Balakrishnan, Yatin Chawathe, Armando Fox, Steven D. Gribble, Todd Hodes, Giao Nguyen, Venkata Padmanabhan, Mark Stemm, Srinivasan Seshan, and Tom Henderson. "A Network Architecture for Heterogeneous Mobile Computing". *IEEE Personal Communications*, October 1998.

[BMJ+98] J. Broch, D. A. Maltz, D. B. Johnson, Y. Hu, and J. Jetcheva. "A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols". *Proceedings of the Fourth Annual International Conference on Mobile Computing and Networking*, October 1998.

[BPSK97] H. Balakrishnan, V. N. Padmanabhan, S. Seshan, and R. H. Katz. "A Comparison of Mechanisms for Improving TCP Performance over Wireless Links". *IEEE/ACM Transactions on Networking*, December 1997.

[BZCS96] M. G. Baker, X. Zhao, S. Cheshire, and J. Stone. "Supporting Mobility in MosquitoNet". *Proceedings of the 1996 USENIX Technical Conference*, January 1996.

[CP96] R. Caceres and V. N. Padmanabham. "Fast and Scalable Handoffs for Wireless Internetworks". *Proc. ACM Conference on Mobile Computing and Networking (Mobicom'96)*, pages 56–66, 1996.

[EF94] K. Egevang and P. Francis. "The IP Network Address Translator (NAT)". *RFC 1631*, May 1994.

[FF99] Sally Floyd and Kevin Fall. "Promoting the Use of End-to-End Congestion Control in the Internet". *IEEE/ACM Transactions on Networking*, August 1999.

[FJ93] Sally Floyd and Van Jacobson. "Random Early Detection gateways for Congestion Avoidance". *IEEE/ACM Transactions on Networking*, 1(4):397–413, August 1993.

[FMM+99] D. Forsberg, J. T. Malinen, J. K. Malinen, T. Weckström, and M. Tiusanen. "Distributing Mobility Agents Hierarchically under Frequent Location Updates". *Sixth IEEE International Workshop on Mobile Multimedia Communications (MOMUC'99)*, 1999.

[FV00] K. Fall and K. Varadhan. *ns Notes and Documentation*, February 2000.

[HJ96] Alex Hills and David B. Johnson. "A Wireless Data Network Infrastructure at Carnegie Mellon University". *IEEE Personal Communications*, 3(1):56–63, February 1996.

[JM96] David B. Johnson and David A. Maltz. "Protocols for Adaptive Wireless and Mobile Networking".

IEEE Personal Communications, 3(1):34–42, February 1996.

[JP00] David B. Johnson and Charles E. Perkins. “Mobility Support in IPv6”. *Internet draft, work in progress*, February 2000. URL http://www.monarch.cs.cmu.edu/internet-drafts/draft-ietf-mobileip-ipv6-10.txt.

[PB94] C. E. Perkins and P. Bhagwat. “Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers”. *ACM SIGCOMM ’94*, August 1994.

[Per96] Charles E. Perkins. “IP Mobility Support”. *RFC 2002*, October 1996.

[PF97] Vern Paxson and Sally Floyd. “Why We Don’t Know How To Simulate The Internet”. *Proceedings of the 1997 Winter Simulation Conference*, December 1997.

[PJ96] Charles E. Perkins and David B. Johnson. “Mobility Support in IPv6”. *Proceedings of the Second Annual International Conference on Mobile Computing and Networking (MobiCom’96)*, November 1996.

[Rap96] T. S. Rappaport. *Wireless Communications: Principles and Practice*. Prentice Hall, New Jersey, 1996.

[SK99] M. Stemm and R. H. Katz. “Vertical Handoffs in Wireless Overlay Networks”. *ACM Mobile Networking and Applications (MONET), Special Issue on ”Mobile Networking in the Internet”*, 3(4):319–334, January 1999.

[Ste94] Richard W. Stevens. *TCP/IP Illustrated, Volume 1. The Protocols*. Addison-Wesley Publishing Company, 1994.

[Val99] A. G. Valko. “Cellular IP - A New Approach to Internet Host Mobility”. *ACM Computer Communication Review*, January 1999.

Appendix A

Installation of the *ns* Extensions

To be able to use the extensions to *ns* presented in this project, it is necessary to replace or modify some files of the *ns* installation. The extensions were tested with *ns* version 2.1b6 and might not work with other versions of *ns*.

- In the main *ns* directory, the files `cmu-trace.cc`, `mip.h`, `mip-reg.cc`, `sdist.h`, `sdist.cc`, and `wireless-phy.h` have to be replaced.
- The `noah/` directory has to be copied into the main *ns* directory.
- The file `noah.tcl` has to be added in the subdirectory `tcl/mobility/`.
- The files `ns-lib.tcl` and `ns-default.tcl` in the `tcl/lib/` subdirectory have to be replaced.
- The file `Makefile.in` has to be replaced or modified to include the NOAH routing agent in *ns*.

Appendix B

Additional Tools

B.1 Plotting Tools

The plotting tool consists of a script that transforms the *ns* trace files into graphs depicting the send and receive rate for each node.[1] To better visualize which node is communicating with what other node, the graphs are plotted one above the other.

Usage: `plot_tp.pl [-s size] <trace file> [node1 node2 ...]`

The optional size parameter sets the size of the y-axis of each subgraph. `<trace file>` is the name of the *ns* trace file. In addition, a list of nodes to plot can be specified. If omitted, the graphs for all nodes that participate in any communication are plotted. The resulting graphic is saved as a postscript file with the same name as the trace file end ending with a `.ps` suffix.

To plot all the flows to a specific network node, a flow plotting utility can be used. The output is saved in a postscript file with the name `flow_n<node>.ps`.

Usage: `plot_flows.pl <trace file> <node>`

B.2 Scenario Generation

B.2.1 Script Scenario Generator

The scenario generator can be used to easily generate complex scenarios with hundreds of nodes. It features its own simple scripting language to specify service areas consisting of multiple base stations, their density and transmission range, network links, wired nodes, etc. From this script file, the scenario generator produces three TCL scripts for *ns* to run the simulations. These files are formatted so that they can be used together with ad-hockey to visualize the results. They consist of the main simulation file (suffix `.tcl`), the

[1]The script parses the MAC trace of *ns*. For this reason, MAC trace has to be enabled for the script to work (`$ns_ node-config -macTrace ON`).

scenario description file (suffix `.scn`), and the communication pattern file (suffix `.com`).

Usage: `gen_scen.pl <script file>`

Scripting Language Commands

- `size-x <sx>`
  set width of the network topology to `sx`
- `size-y <sy>`
  set height of the network topology to `sy`
- `time <t>`
  set end time of the simulation to `t`
- `wired-node <nr> <x> <y>`
  create wired node `nr` at location `(x,y)`
- `home-agent <nr> <x> <y> [-t txr] [-p p] [-c col]`
  create home agent `nr` at location `(x,y)`, optional parameters: transmission range `txr`, node priority `p`, color `col`
- `service-area <nr> <x> <y> [-r rng] [-d dy] [-t txr] [-p] [-c col]`
  create service area `nr` with its center at location `(x,y)`, optional parameters: range of the service area `rng`, density of the base stations within the service area `dy`, transmission range `txw`, node priority `p`, color `col`. To create just a single foreign agent use a service area range of 0.
- `link <nr1> <nr2> <bw> <delay> <queue>`
  connect node `nr1` and `nr2` over a link with bandwidth `bw`, link delay `delay` and queueing policy `queue` (DropTail, RED)
- `movement <nr> <t> <dx> <dy> <s>`
  at time `t` node `nr` starts to move from its current location towards location `(dx,dy)` with speed `s`
- `connection <nr1> <nr2> <t1> <t2> <type> [packet size [interval]]`
  open a network connection between node `nr1` and `nr2` and send data from time `t1` to time `t2` using the transport protocol `type`

B.2.2 Meta Scenario Generator

The Meta Scenario Generator can be used to easily generate a large number of input files for the Scenario Generator. It allows to create scenarios for different numbers of Mobile Nodes with random node placement and movement.

Usage: `meta_gen.pl <type of traffic (udp|tcp)>`

Appendix C

Usage Instructions

This section gives a concrete example how to create a scenario, run simulations, and analyze the results.

1. Create the simulation description file `simulation.gen`:
   - Specify the duration of the simulation and set the size of the simulation topology:
     ```
     time   920.0
     size-x 60.0
     size-y 40.0
     ```
   - Create the wired nodes:
     ```
     wired-node   0 33.0 2.0
     wired-node   1 30.0 4.0
     ```
   - Create Home Agents and Mobile Nodes:
     ```
     home-agent   2 27.0  2.0 -t  0.1 -p 3
     mobile-node  3 20.0 20.1
     ```
   - Specify location and size of the service areas:
     ```
     service-area 4 20.0 20.0 -t  0.3 -p 3
     service-area 5 15.0 20.0 -t  8.0 -p 2
     service-area 6 10.0 20.0 -t 16.0 -p 1
     service-area 7 40.0 20.0 -t  0.3 -p 3
     service-area 8 45.0 20.0 -t  8.0 -p 2
     service-area 9 50.0 20.0 -t 16.0 -p 1
     ```
   - Connect wired nodes and service areas with wired links:
     ```
     link 0 1 1000Kb  10ms RED
     link 2 1 1000Kb  10ms RED
     link 1 4 1000Kb  10ms DropTail
     link 1 5   30Kb  50ms DropTail
     link 1 6   10Kb 100ms DropTail
     link 1 7 1000Kb  10ms DropTail
     link 1 8   30Kb  50ms DropTail
     link 1 9   10Kb 100ms DropTail
     ```
   - Specify movement and communication patterns for the Mobile Nodes:

```
movement     3 100.0 40.0 20.1 0.028
connection 0 3  10.0 920.0 tcp
```

2. Generate TCL simulation files for *ns* with the Scenario Generator:
   `gen_scen.pl simulation.gen`

3. Run the simulation:
   `ns simulation.tcl`

4. Plot a ghraph for node throughput: `plot_tp.pl out.tr` (assuming that `out.tr` is the the trace file created by *ns*)[1]

5. Analyze network flows of single nodes in detail (e.g. node 2): `plot_node.pl out.tr 2`

6. Visualize the simulation with Ad-Hockey: `ad-hockey simulation.scn out.tr`

[1] Instead of the general trace file, a smaller trace file for the transport protocol packets `out.tr1` can be used. This speeds up processing time considerably.