Tobias Rückelt, Halis Altug, Daniel Burgstahler, Doreen Böhnstedt and Ralf Steinmetz: MoVeNet: Mobility Management for Vehicular Networking. In: Proceedings of the 14th ACM International Symposium on Mobility Management and Wireless Access (MobiWAC 16), ACM, November 2016

MoVeNet: Mobility Management for Vehicular Networking

Tobias Rueckelt
Adam Opel AG
Bahnhofsplatz
Rüsselsheim, Germany
tobias.rueckelt@opel.com

Halis Altug
Daimler AG
Mercedesstr. 137
Stuttgart, Germany
halis.altug@daimler.com

Daniel Burgstahler
Technische Universität
Darmstadt
Rundeturmstr. 10
Darmstadt, Germany
daniel.burgstahler@kom.
tu-darmstadt.de

Doreen Böhnstedt
Technische Universität
Darmstadt
Rundeturmstr. 10
Darmstadt, Germany
doreen.boehnstedt@kom.
tu-darmstadt.de

Ralf Steinmetz
Technische Universität
Darmstadt
Rundeturmstr. 10
Darmstadt, Germany
ralf.steinmetz@kom.
tu-darmstadt.de

ABSTRACT

Vehicle Internet access benefits from using heterogeneous multi-provider networks. However, such access suffers significantly from insufficient handover processes. Due to high vehicle speeds, handover happens frequently and is substantially impaired by high handover delays. To solve correlated issues, we propose MoVeNet, a client-controlled, distributed mobility management approach with three substantial characteristics. Firstly, it pools parallel resources of wireless multi-provider networks for efficient common use. Secondly, MoVeNet reduces overhead for mobility management due to publish-subscribe methods. Thirdly, it provides a low-latency option for critical data flows, which enables flow-wise trade-off between latency and overhead. The architecture splits control from data, introducing lightweight Data Agents, which provide protocol transparency towards communication partners. Data Agents are located near the optimal route to provide low-latency packet routes. Moreover, MoVeNet introduces a Control Agent, which offloads communication-intensive tasks from the mobile node to reduce signaling overhead. The result solves deficiencies of related approaches and satisfies typical requirements of modern mobility management protocols. As shown by simulation, MoVeNet reaches excellent handover performance even for the harsh environment of the connected vehicle scenario.

Keywords

distributed mobility management; client-controlled; handover; multi-homing; multi-operator networks

1. INTRODUCTION

Concurrent use of heterogeneous networks can bring significant benefits in perceived network performance, especially when networks complement each other in their characteristics or spacial availability. Multi-homed handover protocols can unlock this potential. We focus on a vehicle scenario for which the key difference in comparison to other scenarios is high node mobility. Mobility complicates the network management because high speeds shorten the connection duration and lead to frequent changes in network availability and experienced characteristics. This holds especially for network scenarios including short-range and mid-range access points. The default TCP/IP Internet protocol stack cannot cope with such harsh network environments. The connection interrupts and reconnects only after a timeout. Current mobility management protocols provide different methods to handle this problem. However, the extreme case of vehicles with fast movement is still covered insufficiently. Handover process takes too much time or approaches are not compatible with default nodes.

For the vehicle scenario, we collect and analyze mobility management requirements in section 2. Based on those requirements, we discuss design decisions that characterize existing mobility management approaches in section 3. To extract design decisions, we do not analyze complete mobility management approaches. Instead, we dissect approaches into concepts that address individual parts of the problem. This provides a detailed design space analysis.

As key contribution, we design a novel protocol for vehicle mobility management in multi-operator networks. We call it MoVeNet: **Mo**bility Management for **Ve**hicular **Net**working and present it in section 4. It is based on client-controlled, distributed mobility management with a Control Agent that offloads management tasks from the mobile node. It implements an identification of nodes, which is independent from IP addresses. This identification enables multi-homing as well as dynamic IP binding for handover. Distributed proxy servers in the Internet enable low round trip times for connections and hide the protocol from legacy communication partners to provide full compatibility. Finally, MoVeNet enables coordination of individual data flows from the Control Agent as central unit. We evaluate handover delay and round trip times for the concept in an Omnet++ simulation and use Mobile IPv6 for comparison. Simulations show a significant performance gain in handover delay. We present the simulation details and results in section 5.

MobiWac'16, November 13–17, 2016, Malta, Malta.

ISBN 978-1-4503-4503-3/16/11...$15.00

DOI: http://dx.doi.org/10.1145/mobi09

2. VEHICLE SCENARIO REQUIREMENTS

The vehicle scenario imposes extreme requirements to mobility management. High vehicle speeds and potentially short-range networks lead to short network connectivity duration and therefore frequent handovers. In addition, many vehicle use cases profit from a low latency connection [5]. The requirements from the vehicle scenario are tough but default user scenarios profit from the same requirements.

2.1 Multi-Homing in Multi-Provider Networks

Heterogeneous multi-provider networks complement each other in coverage and network characteristics. Access to those heterogeneous networks requires multiple interfaces at the mobile node. To exploit the full potential of such a system, parallel transmission via those interfaces is essential. The method to use interfaces in parallel is called multi-homing and constitutes the first requirement to our system.

2.2 Fast Handover

In the automotive scenario, high node mobility leads to short connection duration. To enable continuous data transfer, the system must be able to switch the network during a running connection. This is called handover and is our second system requirement. A fast handover in particular enables quick reaction on unexpected network changes.

Together with multi-homing, a system can provide so-called make-before-break handovers. This method prepares the target network interface before handover execution. Preparation leads to instant handover, which avoids packet loss or pauses in data transmission. Make-before-break handovers are therefore preferred.

2.3 Flow Coordination

Handovers and multi-homing state the question: Which network should transmit which application data? Flows share the bandwidth of available networks and should be assigned to interfaces with care [13]. We do not focus on the decision problem in this paper but argue that a mobility management protocol must provide the means to coordinate flows individually on available networks.

2.4 Low Latency

Mobility management methods often detour packets via proxy servers, which increases transmission latency. However, latency is a key requirement for many safety-related use cases in the automotive scenario. A lower latency value enables the implementation of more safety-related use cases via the Internet connection using hybrid communication as proposed in [2]. Therefore, latency should be kept low.

2.5 Scalability

The number of Internet-enabled vehicles and other mobile nodes rises exponentially [4]. Mobility management approaches must scale well to be applicable in the future. We stress especially on load on the management nodes and on bottlenecks in transmission systems.

2.6 Compatibility

The grown structure of the Internet introduces major challenges on compatibility. This emerges especially from middle-boxes, which violate protocol layer independence and introduce new states. Nonetheless, a mobility management approach must be applicable in the current and future internet architecture and be able to cope with the typical challenges. To provide unlimited Internet access, managed connections must be transparent for legacy servers. This implies compatibility to the default network stack.

3. RELATED WORK

Most performance parameters of handover protocols are only influenced by few design decisions. However, these mobility management protocols are often only evaluated as a whole. Instead of analyzing complete protocols, we extract key design decisions of different approaches and show up how they influence the resulting characteristics. We discuss the extracted ideas and explain their advantages and disadvantages to provide a design space analysis.

3.1 Direct versus Proxy Schemes

Direct handover schemes set up a straight communication between a mobile node and its communication partner. Control traffic is exchanged directly. These systems are usually characterized by high throughput and a low round trip time but high handover delays [15][11][1]. Additionally, both communication partners are able to influence the routing. However, direct handover schemes force servers and mobile nodes to adapt their communication stacks. This is realizable only for a subset of servers in the Internet. Mobility feature use is therefore restricted to servers with adapted communication stack. Prominent examples for this category of direct handover schemes are Multipath TCP (MPTCP) and Host Identity Protocol (HIP).

Proxy schemes, in contrast, use a relay node to hide protocol stack changes from legacy nodes. They provide transparent communication, which results in full compatibility. Additionally, the mobile node gains privacy since communication partners are not aware of the mobile node's local IP address. Protocols that follow this approach are, among others, Mobile IPv6 (MIPv6) and the MPTCP derivative in [8].

Furthermore, there exist handover schemes, which provide transparent communication for legacy servers as well as for mobile nodes. To reach this, they introduce a second relay node, e.g. on access routers. This leads to completely network managed handovers and guarantees full compatibility for the communication partners. Since mobile nodes are not involved in handovers at all, this is especially beneficial for low performance and low power mobile nodes [10]. Especially mobile network providers apply this approach to master mobility management of their customers. Further prominent protocols that follow this concept are Proxy MIPv6 and LISP (Locator-Identifier Separation Protocol).

An additional relay node is usually located nearer at the mobile node than its communication partner but not on the optimal path and, hence, creates packet detours. This leads to faster handover but also to higher round trip times for proxy schemes in contrast to direct ones. Therefore, proxy placement is critical to optimize or determine trade-offs for both.

A new approach to take up this challenge is distributed mobility management (DMM). It assimilates basic ideas from software defined networks but focuses on higher agility in control actions. A good overview is given in [6]. DMM firstly separates control plane from data plane and secondly uses multiple proxy servers for different flows to approximate optimal routes. The control plane might be either centralized or split. This is a promising approach to master the latency and scalability challenges of modern mobility management.

3.2 Controller Location and Triggering

Control functions for handover can be located at the network or at the communication partners. The chosen method is usually coupled to proxy use. When proxies are used to provide transparency

to a legacy node, control is also moved from this node. However, nodes can still influence data flow routing by triggering. This is an optional and lightweight control mechanism to suggest handover to controlling nodes. The control node considers trigger signals as one factor in routing decision.

For complete transparency, like in cellular mobile networks, the operator has to control and manage handovers. This leads to two issues in satisfaction of our initial requirements. Firstly, network operators must cooperate to allow handover in multi-operator networks. This cooperation is unlikely to happen since network operators want to bind their customers. Secondly, flow coordination requires independent routing of different flows. Since the node perceives full transparency on its connection, it cannot select a network interface for its data flows individually. Therefore, full node transparency contradicts with our requirement to support flow coordination and eliminates essential multi-homing benefits.

Client-centric and network-centric mobility management approaches are often discussed as competing strategies. However, we argue that both should be applied simultaneously. Network operators benefit from global knowledge about their network and their customers. They should use this knowledge to provide optimal performance to their customers within their managed network. They should optimize for performance and resource efficiency using network-centric, transparent mobility management. Moreover, mobile nodes should use the opportunity to select between those optimized networks of different providers for individual flows and use them in parallel. They can select the best-suited access networks for their needs, matching individual flows requirements. Indeed, MoVeNet creates an approach to master this second aspect of mobility management and enables dynamic network selection.

3.3 Layer of Mobility

Mobility protocols are based on the idea to establish an abstraction to provide continuous Internet access even though the point of access changes. In products today, this is usually realized in session or application layer using Session Identity Protocol or application logic. For that reason, today, each application has to control inter-network-operator handovers itself. This leads to redundant implementations and therefore to communication and computational overhead. Therefore, dedicated handover protocols act at lower layers [3].

Transport layer covers flow control. This has a high impact on link performance because changes in wireless network access through handover imply instant changes of QoS parameters. As a result, flow control has to adapt as well. Therefore, handover protocols should be tightly coupled with flow control in order to use networks efficiently. Hence, transport layer is well suited for handover mechanisms. Famous candidates for transport layer mobility protocols are Stream Control Transmission Protocol (SCTP) and MPTCP. As SCTP allows reliable and unreliable transfer, it covers most application requirements. However, it is not applied because many middle boxes drop its packets. Apart from that, MPTCP supports reliable transfer only. If MPTCP is used, additional logic is required to support handovers for unreliable data transport.

Applying handover mechanisms at Network Layer solves problems of multiple protocol dependent mechanisms because it is used for all Internet traffic. Consequently, a mechanism in this layer covers all communication. Nevertheless, applying handover at network layer separates handover from flow control. Prominent examples are MIPv6 and HIP. Finally, handovers could also be executed below network layer using software defined networking (SDN). However, this requires cooperation between operators, which have to give open access to their SDN rule programming interfaces. This presumption limits the potential of SDN for operator-independent mobility management substantially.

3.4 Separation of Data and Control

Handover control traffic can be inserted into data packets using protocol options or additional headers. This implies minimal overhead and ideal synchronization of data and control traffic. However, changes in the default protocol stack often lead to packet drop at middle boxes [7]. Examples are MPTCP and HIP. For full compatibility, changes at data packets must be transparent for the network. Control traffic must then be sent in dedicated packets. In combination with proxy use, control data of several managed connections can be bundled do reduce overhead. This is used in most MIPv6 derivatives.

3.5 Handover Granularity

Handovers can be controlled on different granularity levels, for instance for all traffic, for flows or for individual packets. Especially when different interfaces are used in parallel, a finer granularity is beneficial to balance load. Handover granularity is tightly coupled with the chosen handover protocol layer. Network layer handover protocols usually use IP information only and are therefore not able to differentiate between individual flows. However, deep packet inspection, as applied in SDN, can provide more details and hence enable handover granularity at packet level.

Transport layer handover protocols usually act on flow level. This allows to balance individual flows for efficient use of available network resources. Several protocols even allow flow forking which also enables routing of individual packets. This introduces a higher granularity for load balancing and allows throughput optimization for heavy tailed flows [12]. However, handover control should not be considered per flow only, but moreover coordinate parallel flows in order to create synergies, following requirement 3.

3.6 Routing Mechanism

For adaption of routes, there exist three prominent concepts. The first and most common concept is tunneling. Complete IP packets are encapsulated in an additional IP header, which contains modified addresses. The network uses the outer header addresses to route the packet to its destination. The receiver strips the outer header and processes the original packet. This requires only minimal modification to the network stack and is naturally supported by the network and most nodes.

The second concept is header content modification or IP spoofing. To influence the routing, the sender modifies the IP header of packets before sending it to the destination. However, upper layer protocols would drop the packet because a change of IP address prevents flow identification at the destination node. Therefore, the destination node must revert the modification before processing the packet. Thus, the method requires prior context establishment including change tables for IP addresses. After context establishment, this approach can reduce per-packet overhead to zero bytes. The exchanged context is used for identification while the locator, the IP address, changes. Nevertheless, many access networks use ingress filters, which still drop packets with a foreign source address. This method is used by the multi-homing protocol SHIM6 or in similar way by IPv6 routing extension header.

The third method can be seen as an approach in between: Labeling. Instead of using another IP address in the packet, like tunneling, or pure exchanged context, like IP spoofing, those protocols introduce a kind of label, which identifies the mobile node or the data flow. This targets a clear separation of locator and identifier and, hence, solves the problem in an elegant way. Like in IP spoofing,

this method requires prior a context establishment, but only once: exchanging the identifier. However, the identifier can later be used to determine or even authenticate the source of the packet from an unknown IP address. Well-known examples using this method are ILNP (Identifier-Locator Network Protocol), LISP and HIP.

4. MOVENET CONCEPT

In the following, we present MoVeNet, our client-centric mobility management approach for vehicle Internet access using heterogeneous networks. With our design, we focus on the design space options discussed in related work to satisfy the requirements presented in section 2.

4.1 MoVeNet Overview

MoVeNet is a distributed architecture that is oriented on SDN, splitting control from data flows. To achieve compatibility to legacy communication partners, we apply proxy forwarding which hides the new mobility management methods from legacy servers. We apply labeling using a new layer as IPv6 extension header that identifies flows independent from the used transmission network. This solution integrates multi-homing, handover and flow coordination capability by design. In the following, we firstly explain the architecture of MoVeNet and secondly focus the protocol details.

4.2 Data Agent – Lightweight Proxy

Data Agents are lightweight proxy nodes that manage only few connections for which they are near the optimal route. A mobile node is supposed to use many of them in parallel, which results in a distributed mobility management approach. In the following, we detail the reasons for this design decision.

A data proxy can hide protocol changes to provide compatibility to legacy communication partners. To reduce round trip times in proxy approaches, proxies should be located near to the optimal route. But placement in the automotive scenario with heterogeneous networks encounters two conflicting optimization goals: Firstly, communication to different servers leads to ambivalent data routes, as shown in figure 1 left. Consequently, a central proxy should be located near the mobile node to reduce detours to different servers. Secondly, communication via different network operators leads to Internet access via independent autonomous systems. These autonomous systems of network operators connect to the global internet architecture at different points, as depicted in figure 1 right. To reduce packet detours, a proxy should be located near the server. In fact, Data Agent location additionally impairs handover delay. The farther the Data Agent is located from the MN, the longer takes it to get notifications about and react on IP address changes. Therefore, the Data Agent should be located at a weighted middle between CN, and the attachment points of expectedly used networks.

To solve the placement conflict, MoVeNet follows distributed mobility management concepts and breaks up the central proxy server into multiple lightweight instances: Data Agents. During connection setup, a Data Agent is chosen or created as communication gateway for the mobile node to provide transparency to communication partners. The exact selection algorithm is out of scope of this work. This distributed mobility management scheme firstly enables more efficient placement to reduce packet latency and secondly distributes load to provide system scalability.

4.3 Control Agent – System Orchestration

During connection setup, Data Agents must be initialized at appropriate locations. For efficient Data Agent location selection, further information from the Internet is required. If the mobile node

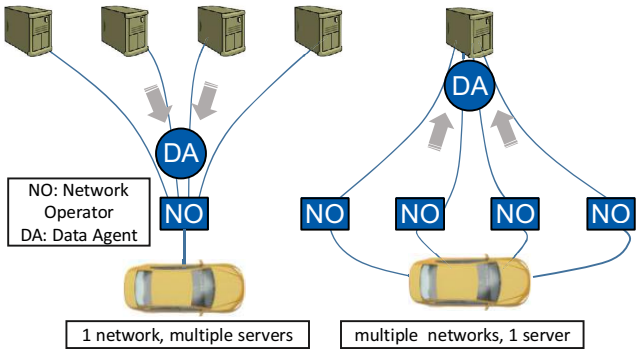

Figure 1: Data Agent placement: multiple communication partners (left) and multi-homing (right)

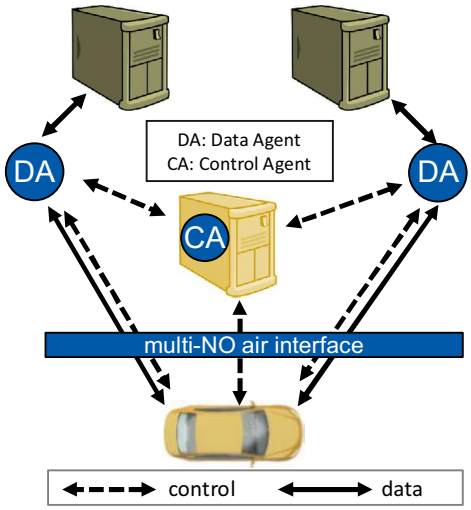

Figure 2: MoVeNet architecture

itself selected Data Agent locations, it would create additional overhead via the wireless interfaces under optimization. To avoid this additional data overhead for proxy orchestration, we introduce a new node in the Internet: the Control Agent. It supports the mobile node in all control actions of mobility management.

For mobility management, Data Agents must know the current IP addresses of the mobile node's interfaces, which can be used. If the mobile node sent all IP address changes to all Data Agents, the air interface is strained. To release load from the wireless connections, we select a publish-subscribe-based control data structure, as proposed in [14]. The mobile node informs the Control Agent about IP address updates, which distributes them to relevant Data Agents. This contributes to system scalability and reduces management overhead.

To use available networks in the best possible way, data flows should be coordinated. The coordinating node needs all information about existing data flows and estimated network characteristics. We select the Control Agent for flow coordination. It knows all active nodes of the distributed system and has a wired Internet connection.

However, handover triggering via Control Agent causes additional handover latency because the final handover is executed at Data Agents and, hence, messages must pass an additional mandatory hop. To avoid this overhead especially for time-critical handovers, we introduce an optional fast piggybacking handover con-

trol mechanism. IP address updates of the mobile node are directly attached to the data packets. This way, messages can be sent via new routes and are accepted from Data Agents without prior announcement. To enable this, we propose a new identification layer.

The ideal location of the Control Agent differs from those of the Data Agents since the Control Agent exists persistently, in contrast to the lightweight Data Agents that might exist only for the duration of a single connection. Furthermore, the Control Agent is not responsible for data connections. Hence, its location does only affect handover delay of connections, which do not use piggybacking and are therefore not considered to have hard handover delay requirements. The Control Agent's placement is therefore less critical. Its ideal location can be determined using a weighted middle. Hereby, only the points of attachment of network operators should be considered to which in sum the MN is (1) expected to be connected for most of the time and that (2) preferably provide a low latency. Since Control Agent is a lightweight agent, which is not responsible for data routing, it can even be migrated during long range trips of the MN.

4.4 Node Identification Layer

The basic problem of mobility management is the so-called double role of IP addresses. End-points identify communication partners using flow IP addresses. If a node continues to transmit the same flow via another IP address, its communication partner identifies it as a distinct flow. This enforces complete connection reestablishment with a default TCP/IP stack. Flows therefore cannot continue.

To enable a handover, we separate end-point identification from mobile node's IP addresses. According to the concept of Host Identity Protocol (HIP), we introduce a node identifier. This identifier is passed to upper layer protocols instead of the IP address. The identity abstraction enables dynamic binding of IP addresses to a flow while hiding handover from upper layers. This dynamic IP address binding enables firstly multi-homing and secondly handover.

To preserve compatibility with middle boxes, we use IPv6 destination options header to transport the identification. To keep overhead small, we keep it as simple as possible. In contrast to HIP, we do not target to solve security issues, which are usually covered in other layers. The header is for identification in the first place but can also cover new node IP addresses. Hence, packets can be sent via different routes because identification is separated from IP addresses. This enables fine-grained flow coordination via parallel networks, according to requirement 3.

4.5 IP Address Table Synchronization

Distributed mobility management includes sharing of usable IP addresses to Data Agents. Since we target multi-homing in our approach, this requires using an IP-Address table T, which has to be synchronized between all entities of the distributed mobility management system. Hence, we introduce a synchronization mechanism, which is able to let both sides add and remove multiple entries from the IP address table simultaneously.

The mechanism uses the following four fields in the identification header: a sequence number s (Seq#) and an acknowledgment number a (Ack#) of each 8 bit, add-IP add (IP Add) and remove-IP rem (IP Rem) counters of each 4 bit, and a IPv6 addresses list of length $(add + rem) \cdot 128bit$. They are shown in figure 3.

Locally stored information covers the IP address table T of valid IPv6 address changes and a fifo-list L of unacknowledged IPv6 addresses including the information if they are added or removed. In addition, there are two local variables for sequence and acknowledgment numbers s_l and a_l. These two local values are used to fill

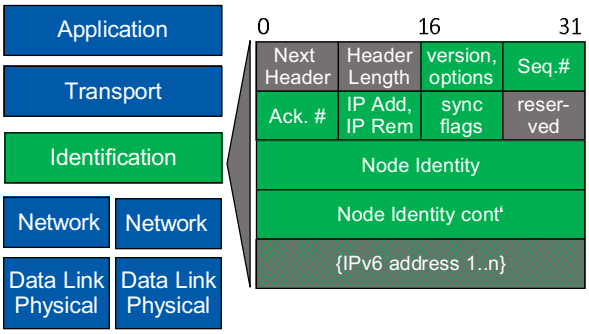

Figure 3: Identification Header

the header fields s and a in identification header of outgoing packets. The header fields add, rem and the IPv6 address changes are set according to the entries of L. The list in identification header starts with IPv6 addresses to be added, followed by IP addresses to be removed from the communication partner's IP address table T. In the following, we describe the steps of the process:

- **Send IP address change:** (1) IP address changes in T. (2) add change entry to L. (3) $s_l = s_l + 1$.
  When there is a change in T, the changed IPv6 address is added to the fifo L including the information if it was added or removed. For each change, the local sequence number s_l is increased by one.

- **Send acknowledgment:** (1) $c = s - a$. (2) $s_l = s_l + c$, $a_l = a_l + c$. (3) apply all c IP update(s) to T.
  Each IP address change at the communication partner leads to an increase of the received sequence number s and therefore to a difference to a in the identification header. Instead of comparing s of the received message to the local s_l, we use the temporary variable c, which is the difference of s to the received acknowledgment number a. c exposes the number of IP address changes of the communication partner. This decouples detection from local data and finally enables multiple simultaneous IP address updates of both communication partners. Subsequently, the local variables s_l and a_l are increased by c to acknowledge all updates within the next outgoing packet. Finally, the updates are applied to table T.

- **Receive acknowledgment:** (1) $c = a - a_l$. (2) $a_l = a$. (3) drop c items from fifo L.
  To show that acknowledgment has been received, a_l is set to the received a. In addition, all acknowledged IP address changes are removed from the fifo list L. This makes the node stop sending confirmed IP address changes and completes synchronization of the two nodes.

Synchronization is firstly covered in mobile node connections to Data Agents with piggybacking enabled and secondly to the Control Agent. Finally, the Control Agent synchronizes all IP address changes to Data Agents, following the publish-subscribe structure. The algorithm ensures that both communication partners can add and remove IP addresses simultaneously and each of them knows the latest synchronization state of the other. Finally, it guarantees that only unacknowledged addresses are transmitted.

4.6 Identification Layer Structure

The complete identification header structure is shown in figure 3. It follows the rules of IPv6 extension headers and adopts the IPv6

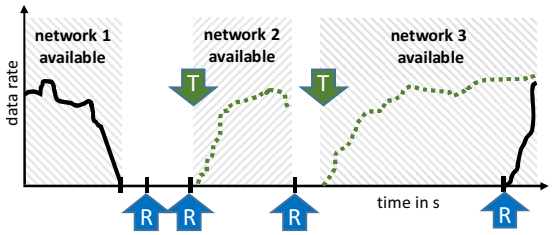

Figure 4: TCP retransmission timer, worst case example

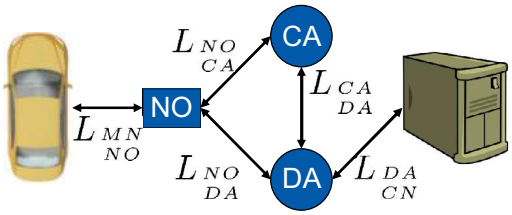

Figure 5: Latency between nodes

Destination Options Header format in simple case. Therefore, the first two bytes are reserved for the *next header* and *header length* fields. The next byte describes option type and version. These are followed by three bytes for the IP address synchronization mechanism, which cover sequence numbers and counters. Next, it covers synchronization flags, which support the initialization process of MoVeNet's distributed structure. As final mandatory part, we add the node's identification number, optionally followed by the list of unsynchronized IPv6 address changes.

4.7 Advanced Retransmission Trigger

In early testing phase, we observed unexpectedly high transmission pauses with TCP. During long disconnection phases, TCP runs into a timeout and starts its retransmission algorithm. It tries to retransmit a packet and doubles retransmission trigger timeout for every next retransmission attempt. Since in our scenario, network availabilities might be very short, retransmission might fall into the next down-phase of the network interface. We show this worst-case scenario qualitatively in figure 4, marking TCP retransmission attempts with an R and the exemplary data rate with a black line. Transmission stops after disconnecting from network 1. Network 2 is skipped completely because of bad timing of TCP packet retransmissions. Then a long timer expiry time leads to unused network resources and exceptionally long handover delay for TCP in network 3.

To counteract this behavior, MoVeNet applies its knowledge about network availability and triggers TCP externally. For each active flow, it therefore stores the last packet that passes identification layer in Data Agent and mobile node. As soon as an interface is known to be up again, MoVeNet resends the stored packet for inactive flows. For Data Agent, this happens at reception of a new IPv6 address via our synchronization mechanism. This simple resend mechanism triggers TCP connections to continue transmission faster. The resulting behavior is shown qualitatively in figure 4. We marked MoVeNet triggers with Ts and sketched resulting TCP behavior qualitatively with green dotted lines.

4.8 Design Implications

The design of MoVeNet has direct impact on different characteristics that correspond to the targeted requirements. The distributed design using Data Agents implies a spread of traffic via multiple nodes. Since Data Agents are dynamically allocated, traffic distribution can prevent overload of single nodes and therefore make the system scalable and less prone to system failure or attacks. Data Agents furthermore hide the MoVeNet actions and protocol stack extensions from communication partners, which provides full transparency and compatibility to current Internet services. A clever placement can moreover reduce handover delays and round trip times between communication partners.

In addition, MoVeNet optimizes connections selectively for low overhead or low handover delay and round trip times. Due to the separation of data and control through the Control Agent, the novel protocol reduces signaling overhead. The mobile node sends IP address updates only to the Control Agent, which distributes the information to all Data Agents via cable network. Moreover, due to optional piggybacking of IP addresses, Data Agents can handle data packets that arrive via new links instantly, without prior announcement. This mechanism can reduce handover latency to about zero. To introduce no additional security risks with MoVeNet, we propose to use IPv6 Authentication Header to encapsulate the identification layer, whenever IP address updates are transmitted via piggybacking. This optional piggybacking mechanism is used only for connections with critical handover requirements.

Finally, the identification layer provides a multi-homing solution that solves the handover problem in a natural way. It separates the node identity from IP addresses, which are coupled to possibly short lived L3 connections. Using permanent node identity as identifier, L4-connections can be continued despite of L3-connection switches. With this design, we overcome most deficiencies of existing mobility management protocols and can even satisfy the tough requirements for connected vehicle Internet access.

5. EVALUATION

We firstly explain the evaluated variables and used metrics, followed by a brief system latency analysis. Secondly, we explain our simulation scenario and finally present and discuss our simulation results.

5.1 Evaluation Metrics and Analysis

Most intended characteristics of MoVeNet are given by design, as shown in section 4.8. This includes multi-homing, scalability, flow coordination, compatibility, overhead and latency. To analyze handover delay, we additionally simulate the protocol and compare it to Mobile IPv6. We therefore measure time from handover decision until the first packet arrives at the end-point via the new route. To investigate handover delay of MoVeNet, we first have a look on how partial delays of it are compound. We give an overview about influencing latencies in figure 5.

MoVeNet operates on network layer. Therefore, we omit latency effects of lower layers in analysis because their characteristics especially depend on technology and on the autonomous system of the network operator. Instead, we regard the latency from mobile node (MN) to the point-of-presence of the network operator (NO) as estimable value using $L_{\substack{MN\\NO}}$ according to [9]. Furthermore, latency values between NO and Control Agent $L_{\substack{NO\\CA}}$ and between NO and Data Agent $L_{\substack{NO\\DA}}$ differ from NO to NO because their points-of-presence are not located at the same place. For simplification, we reuse the acronym NO in the context of routing for the point-of-presence of the network operator in the following.

Handover is an update of packet routing, which requires prior exchange of information about the new route. This may cover a new

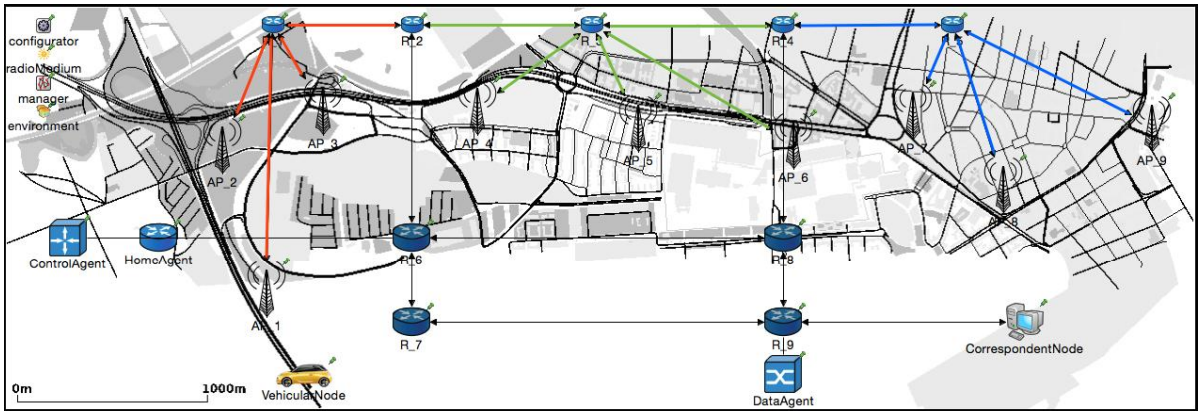

Figure 6: Simulation scenario Frankfurt

IP address or just a trigger to send data via another route. The handover delay therefore covers firstly the time to inform all required nodes about the new route and secondly the delay until new packets arrive via the new route. Consequently, the paths for information and rerouting should be as small as possible. In our analysis, we use the terminology of MoVeNet to analyze handover strategies for control and data planes: CA and DA.

A handover requires sending an update to the CA, which informs the DA about the new route. In addition, data packets have to take the new route and travel from DA to MN via a NO. Therefore, handover delay in MoVeNet is defined according to equation 1. It shows, that for fast handover in multi-homing, low latency links should be preferred for signaling. For MIPv6 as well as MoVeNet with enabled piggybacking, the CA can be seen as integrated into the DA. For these two setups, we can set $L_{\substack{CA \\ DA}} = 0$. This saves additional latency for handovers by the expense of some overhead. For MIPv6, the Home Agent is usually the router of first attachment. For fast moving mobile nodes, this means that the mobile node gradually departs from the managing node, which increases $L_{\substack{NO \\ CA}}$ and therefore the handover delay and leads to rising detours especially when using the networks of other NOs. In MoVeNet, the DA is supposed to be selected for each connection separately, depending on the points-of-presence of expectedly used NOs and the corresponding node location. This leads to low detours and therefore low latencies for HO and RTT.

$$L_{HO} = 2L_{\substack{MN \\ NO}} + L_{\substack{NO \\ CA}} + L_{\substack{CA \\ DA}} + L_{\substack{DA \\ NO}} \quad (1)$$

5.2 Simulation Scenario

We simulated MoVeNet and Mobile IPv6 communication using Omnet++ 4.6 with INET 3.0 Framework. To simulate vehicle mobility, we used SUMO 0.26 (Simulation of Urban MObility). For Mobile IPv6 simulation, we apply the INET xmipv6 reference model. We created a mixed scenario of Frankfurt, which covers motorway, suburban area and urban area. Along the track, we placed 9 WiFi (802.11g) hot spots with extended range of about 500 meters, as shown in figure 6. They cover the track completely. This results in 8 handovers per simulation run. We simulate 100 runs, which leads to 800 handover events per scenario for evaluation. However, MoVeNet is technology independent. Therefore, any other IP transport technology could be used.

We use four scenarios, varying between UDP and TCP traffic and the send direction of data: from server to mobile node and from mobile node to server. We simulate each scenario with (1) a single-homed mobile node using Mobile IPv6, (2) a single-homed mobile node using MoVeNet and (3) a multi-homed mobile node using MoVeNet. Piggybacking is activated.

5.3 Simulation Results Handover Delay

Figure 7 shows cumulative distribution functions for handover delay of the protocols in the four scenarios. Note, that the x-axis is in logarithmic scale.

Since piggybacking is activated in MoVeNet, the handover performance in TCP in both directions and UDP to server is very good. MoVeNet outperforms MIPv6 (4408ms) for handover delay in median by factor 47.41 (93ms) for single-homed mobile nodes and factor 407.45 (11ms) for multi-homed mobile nodes.

This significant difference emerges from MoVeNet design. While MIPv6 sends a handover request to the mobile node and then waits for an acknowledgement before sending data via the new route, MoVeNet uses the new route as soon as it is available. Piggybacking of the new IPv6 addresses in the identification header authenticates packets from any source at the Data Agent. Therefore, new routes do not have to be announced but can be used for sent out packets instantly.

For the case of UDP from a server to mobile node, no packets are sent from mobile node to the Data Agent. Therefore, MoVeNet is not able to apply piggybacking in this case. We do not enforce direct communication to the Data Agent but let MoVeNet fall back to its low-overhead signaling, which uses the Control Agent as publish-subscribe broker to distribute routing information to Data Agents. Therefore, signaling packets take minor detours, which cause delay in handover. Accordingly, HO delays are slightly higher. We measure a gain of factor 1.33 (95ms) for single-homed and 1.83 (69ms) in median for multi-homed MoVeNet mobile nodes over MIPv6 performance.

For TCP, we observe a much higher handover performance gain of MoVeNet relative to MIPv6 than for UDP transfer. In fact, MIPv6 performance for TCP is bad. This results from the TCP retransmission algorithm, as explained in section 4.7. TCP cannot cope with the harsh network environment of fast changing network availability. Its retransmission algorithm fails in many cases. In contrast to MIPv6, MoVeNet is able to support TCP with its event triggered retransmission algorithm and therefore boosts TCP performance for the use case.

The observed difference in handover delay between single-homed and multi-homed mobile nodes emerges from the effects of the make-before-break principle. For single-homed mobile nodes, an interface has to disconnect from the current access point and con-

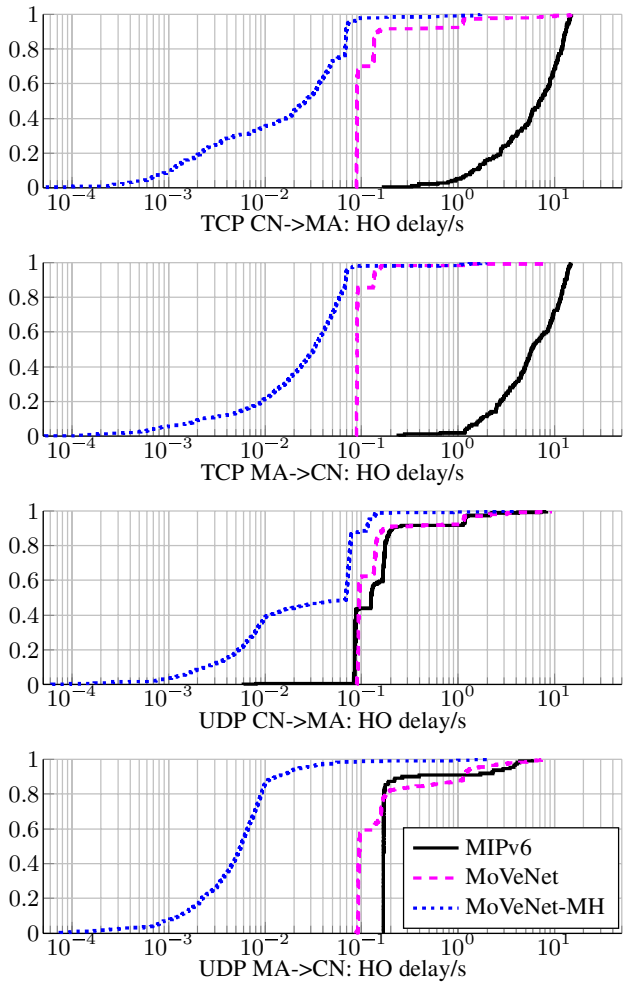

Figure 7: Cumulative distribution of handover(HO) delay in s

nect to the next access point before data transmission can continue. This leads to high handover delays. For multi-homed mobile nodes, data transmission continues via the first network interface while a second network interface can establish a L3 connection to the new access point in parallel. After successful connection establishment, the mobile node performs a smooth handover to the second interface. This eliminates effects from lower layer connection setup. In our simulation, this resulted in a median handover performance gain of factor 6.72 when multi-homing is used. We therefore strongly encourage using multi-homing for vehicle Internet access.

6. CONCLUSION

Vehicular Internet connectivity using multiple network operators imposes a harsh network environment. Frequent handovers make it challenging for mobility management protocols. To master this challenge, we present MoVeNet, a client-centric, multi-homing-enabled and distributed mobility management protocol, which operates on network layer with the ability to control individual flows. It splits data plane from control plane. Data plane manages routing of packets using so called Data Agents, that are located near the optimal routes of individual data flows to result in low packet latency. To orchestrate Data Agents and to reduce signaling traffic, we introduce a central Control Agent, which takes over management jobs from the mobile node and acts as a publish-subscribe signaling broker for uncritical flows. Integrated mechanisms for TCP retransmission triggering and for optional instant handover without prior announcement complete MoVeNet's strengths in handover delay performance, as shown in simulation. It masters many deficiencies of related protocols by design, provides substantial flexibility, excellent performance and can even cope with the harsh network environment experienced from moving connected vehicles.

7. REFERENCES

[1] Y.-C. Chen, Y.-s. Lim, R. J. Gibbens, E. M. Nahum, R. Khalili, and D. Towsley. A measurement-based study of MultiPath TCP performance over wireless networks. In *In Proc. of ACM Internet Measurement Conference*, 2013.

[2] CONVERGE Consortium. Deliverable D3. http://www.converge-online.de/doc/download/CONVERGE_AP2_Deliverable_D3_final.zip 4/2/2016, 2013.

[3] W. Eddy. At what layer does mobility belong? *IEEE Communications Magazine*, 42(10), 2004.

[4] Ericsson. Mobility Report. Technical report, 2016.

[5] ETSI. TS 102 637-1: Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 1 : Functional Requirements. Technical report, 2010.

[6] F. Giust, C. J. Bernardos, and A. D. Oliva. HDMM : deploying client and network-based distributed mobility A hybrid approach. *Springer Telecommunication Systems*, 59(2), 2015.

[7] F. Gont, J. Linkova, T. Chown, and W. Liu. Observations on the Dropping of Packets with IPv6 Extension Headers in the Real World. *RFC 7872*, 2016.

[8] G. Hampel, A. Rana, and T. Klein. Seamless TCP Mobility Using Lightweight MPTCP Proxy. *In Proc. of ACM International Symposium on Mobility Management and Wireless Access (MobiWAc)*, 2013.

[9] F. Kaup, F. Michelinakis, N. Bui, J. Widmer, K. Wac, and D. Hausheer. Behind the NAT - A Measurement Based Evaluation of Cellular Service Quality. In *In Proc. of International Conference on Network and Service Management (CNSM)*, 2015.

[10] K. Kong, W. Lee, and Y. Han. Mobility Management for All-IP Mobile Networks: Mobile IPv6 vs. Proxy Mobile IPv6. *IEEE Wireless Communications*, 15(2), 2008.

[11] M. S. Rahman, M. Atiquzzaman, W. Eddy, and W. Ivancic. Performance Comparison between MIPv6 and SEMO6. *In Proc. of GLOBECOM*, 2010.

[12] C. Raiciu, D. Niculescu, M. Bagnulo, and M. J. Handley. Opportunistic Mobility with Multipath TCP. In *In Proc. of International workshop on MobiArch*, New York, New York, USA, 2011. ACM Press.

[13] T. Rueckelt, D. Burgstahler, F. Jomrich, D. Böhnstedt, and R. Steinmetz. Impact of Time in Network Selection for Mobile Nodes. In *In Proc. of ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM)*, 2016.

[14] T. Rueckelt, F. Jomrich, D. Burgstahler, D. Böhnstedt, and R. Steinmetz. Publish-Subscribe-Based Control Mechanism for Scheduling Integration in Mobile IPv6. In *In Proc. of IEEE Local Computer Networks Conference (LCN)*, 2015.

[15] J. Wang. HIP Based Mobility Management for UMTS / WLAN Integrated Networks. In *In Proc. of IEEE Australian Telecommunication Networks & Applications Conference*, 2006.