

B.A.T.M.A.N. Handover Extension for Routing Nodes in Infrastructure WMNs

Patrick Herrmann
RWTH Aachen University, Germany
herrmann@itsec.rwth-aachen.de

Ulrike Meyer
RWTH Aachen University, Germany
meyer@itsec.rwth-aachen.de

Abstract—In the past, several proposals to support fast handover in wireless mesh networks have been published. However, most handover extensions are only concerned with moving stations that are unaware of the used routing protocol and assume relatively stationary routing nodes. In this paper, we propose an extension to the BATMAN routing protocol to support seamless handover of routing nodes in infrastructure wireless mesh networks. We implement the new extension in our wireless mesh testbed and evaluate its performance in comparison to a standard WLAN handover followed by a route reestablishment using BATMAN. The evaluation shows that route reestablishment is more than 40 times faster with our extension when compared to standard BATMAN.

I. INTRODUCTION

In infrastructure Wireless Mesh Networks (WMNs) Mesh Routers (MRs) are wirelessly interconnected, connect to other networks such as the Internet via Mesh Gateways (MGs), and route traffic on behalf of connected Mesh Clients (MCs) as well as other MRs. MCs are typically considered to be mobile devices such as smartphones, tablets or notebooks. As such, MCs move through a WMN and connect to the network via different MRs while they are moving. Consequently, proposals have been made that aim at supporting seamless handover for MCs in WMNs (e.g., [1]).

On the contrary, MRs are often considered to be rather static. This, however, is not necessarily the case. In particular, an MR may be installed in a moving object such as a bus or any other type of vehicle and MCs may move along with this MR (c.f. Fig. 1). As a consequence, these MCs will not be reachable as long as the MR's connection is disrupted due to its movement. This may significantly reduce the user experience, in particular if network services such as voice or streaming are used. In this paper this issue is addressed in the context of WLAN-based infrastructure WMNs and a proactive handover extension to the popular routing protocol Better Approach to Mobile Ad-hoc Networking (BATMAN) [2] is proposed. In particular, in the proposed handover extension, the moving router pro-actively announces its wish to move to another node as network attachment as well as a list of MCs moving alongside. This announcement is propagated through the network as needed. The approach thus updates routes already before the handover takes place.

The new extension is implemented in our infrastructure WMN testbed and its performance was evaluated in two different scenarios modeling the topology of the WMN and

varying the location of the moving router within the network. The performance comparison of the new extension with the regular route reestablishment procedure of BATMAN after handover shows that the extension is more than 40 times faster than standard BATMAN.

The remaining paper is structured as follows. Related work is discussed in Section II. In Section III the original BATMAN routing protocol is described followed by Section IV in which the handover extension is explained. The evaluation of BATMAN with and without the handover extension is presented in Section V. The paper is concluded by Section VI.

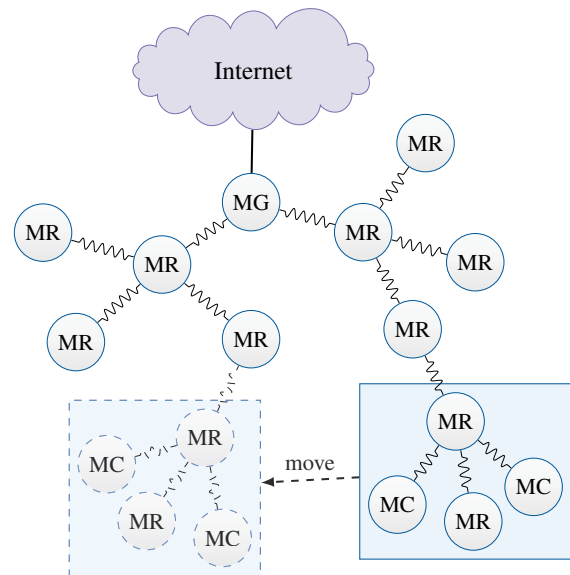


Fig. 1. MR moving with its MCs and MRs in an infrastructure WMN

II. RELATED WORK

Seamless handover procedures are an integral part of cellular networks and have been extensively studied in (single-hop) WLANs (e.g., [3], [4], [5], [6]). Prior work in this area mainly focuses on accelerating the process of reestablishing layer 2 connectivity after handover and typically does not study the effect of the handover on the routing layer. This is particularly interesting in the context of WMNs where the routing protocol used within the WMN has to be able to recover from a node's

movement. While, to the best of our knowledge, this issue has not been studied in the context of infrastructure WMNs so far, it has to some extent been addressed in the Mobile Ad-hoc Networks (MANETs) context.

Quartulli and Cigno [1] proposed an (reactive) extension for the BATMAN routing protocol focusing on speeding up route reestablishment to (non-routing) MCs. When a client connects to a router the router looks up in its client specific routing table if the client was connected to another router beforehand. If that is the case it will notify the old point of attachment about the new location of the client. In contrast, the approach presented in this paper focuses on moving routers, which are themselves aware of the routing protocol. Consider a router R_M , called moving router, that is connected to router R_S , called serving router. Assume that R_M wants to connect to another router R_T , called target router. Before R_M actually executes the handover, it pro-actively announces to R_S that it will connect to R_T soon.

An orthogonal line of work in MANETs focuses on route recovery when nodes loose connection or when the links are unstable. In order to determine the effectiveness of a routing protocol the throughput and rate of packet losses for the whole network are typically measured [7], [8], [9], [10], [11], [12], [13], [14]. In contrast, in this work it is assumed that the node decides itself when to connect to another router, e.g., because it is moving and the link quality is degrading.

The approach described in RFC 3963 [15] is complementary to the work presented here. In this work, only link layer mobility is assumed, i.e., the approach does not take IP address changes into account as it is assumed that routers move through one IP subnet rather than different subnets.

III. B.A.T.M.A.N.

BATMAN is a proactive routing protocol for MANETs that also works well in infrastructure WMNs. It is mainly developed and used by the Freifunk community.¹ The first version operated at the network layer and is specified in an (expired) Internet-Draft [16]. A newer version of the protocol, called BATMAN Advanced (BATMAN-ADV), operates at the link layer. Besides operating at a different layer, BATMAN-ADV is implemented in the Linux kernel instead of user space for performance reasons. At the end of 2015 BATMAN Version 5 (BATMAN V) was announced on the official webpage,² which uses the expected throughput as metric. In the following section BATMAN V is described in more detail as it is the basis for our handover extension (see Section IV).

A. BATMAN V

In BATMAN V, nodes advertise their presence by emitting so called Originator Messages (OGMs) and are therefore called originators. The content of such an OGM (c.f. Fig. 2) includes the MAC address of the originator, the path metric (throughput, initially set to the maximally possible value 2^{32}),

¹<http://freifunk.net>

²<https://www.open-mesh.org/projects/open-mesh/wiki/2015-12-30-batman-adv-v-leaves-the-nest>

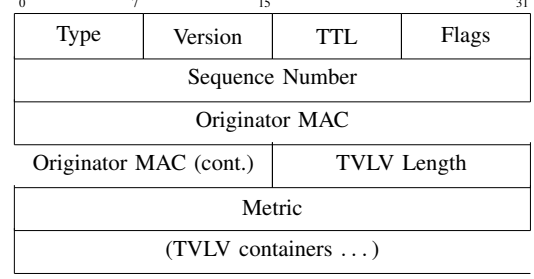


Fig. 2. BATMAN OGM Version 2

the Time-to-Live (TTL) and a sequence number. The Type Version Length Value (TVLV) field specifies the size of TVLV containers the OGM may carry. This container mechanism is not relevant for the handover extension. It was introduced by the BATMAN developers to support better backward compatibility compared to previous versions.

Loosely speaking, a node that receives an OGM from one of its neighbors (not necessarily the originator) updates the path metric and decrements the TTL in the OGM and then broadcasts it on each of its interfaces. In addition, the node updates its best next hop towards the originator for each of its interfaces. In the following we describe in more detail how the path metric and the best next hops are updated.

Consider a node R with n interfaces $\text{iface1}, \dots, \text{iface}n$. These interfaces are called hard interfaces because they are real interfaces. All hard interfaces are assigned to one soft interface bat0 so that BATMAN knows about them. Assume R has a neighbor N_1 that is connected to R 's interface iface1 and that N_1 broadcasts an OGM O with throughput O_T on interface iface1 . Upon receipt of O , R first checks if $O_T = 0$ and drops O if this is the case. Otherwise R updates the metric O_T to O'_T as follows. First, R queries the expected throughput $E_T(\text{iface1}, N_1)$ to N_1 over iface1 as detailed at the end of this section. It then sets O'_T to the minimum of $E_T(\text{iface1}, N_1)$ and O_T . Then R computes a new throughput $O_{T, \text{iface}i}$ for each interface $\text{iface}i$ by:

$$O_{T, \text{iface}i} = \begin{cases} 0.5 \cdot O'_T & , \text{ if } \text{iface}i == \text{iface1} \\ O'_T \cdot \frac{(255 - \text{HP})}{255} & , \text{ otherwise} \end{cases}$$

In other words, the throughput of the interface on which O was received is cut in half, while a Hop Penalty (HP) is applied to each of the other interfaces. Then R broadcasts the OGM $O_{\text{iface}i}$ on each hard interface.

So far it was described how the throughput in OGMs propagating through the network is updated, but not how each node chooses the best next hop towards the originator for each of its interfaces. R stores the best next hop towards the originator for each interface separately. After R has broadcasted an $O_{\text{iface}i}$ on $\text{iface}i$ it checks if the path metric $O_{T, \text{iface}i}$ over N_1 is higher than that of the previously stored best next hop towards the originator on $\text{iface}i$. If it is higher then N_1 is set as the best next hop for packets received on $\text{iface}i$ destined to the originator in question.

0	7	15	31
Type	Version	Originator MAC	
Originator MAC (cont.)		Sequence No.	
ELP Interval		Padding . . .	

Fig. 3. BATMAN ELP Packet

In BATMAN V OGMs are not used to detect direct, i.e., one hop, neighbors. This is one purpose of the Echo Location Protocol (ELP) (c.f. Fig. 3). Similar to the OGM packet it contains type, version, MAC of the originator, a sequence number. It also contains the interval in which ELP messages are emitted by the originator. The ELP packet is padded so that its minimum size is 200 bytes. A node broadcasts on each of its interfaces $iface_i$ ($1 \leq i \leq n$) ELP messages in regular intervals and each recipient adds a neighbor entry for that node. However, ELP has another purpose when it is used on wireless interfaces. In BATMAN V a node queries `cfg80211` [17] for the expected throughput towards a neighbor connected to one of its interfaces. The expected throughput is computed with some Rate Control (RC) algorithm. Such an algorithm needs traffic between the node and its neighbor to function. Therefore, ELP sends two unicast messages to a neighbor to trigger the RC algorithm if the neighbor has not sent enough packets in the last 0.1 s. Furthermore, BATMAN V enables the user to overwrite the throughput with custom values so that routes can be influenced. BATMAN V also supports Ethernet devices where the throughput is determined using `ethtool`.

Note that in BATMAN V other nodes are not aware of the different hard interfaces of an originator. If a node has multiple interfaces then BATMAN V uses the Medium Access Control (MAC) address of the interface first assigned to `bat0` as the originator MAC address.

When a node in BATMAN V moves and therefore disconnects from its current neighbor and connects to new neighbor, it takes time until the new location of the node is spread through the network with the help of OGMs and packets can correctly be rerouted to the node. The proposed handover extension described in the next section aims at accelerating this process.

IV. HANDOVER EXTENSION

In this section, a high-level overview of the handover extension is provided first. This is followed by a detailed description of the extension and a discussion on the rationale and limitations. Finally, the implementation is described briefly.

A. High-level Overview

The proposed handover extension is based on BATMAN V. It enables a Moving Mesh Router (MMR) to pro-actively announce its wish to handover from a particular Serving Mesh Router (SMR) to a specific Target Mesh Router (TMR).³ The underlying network model as well as the used notions and

assumptions are depicted in Fig. 4. Here, the MMR R_{MR} is currently connected to R_n as SMR and wants to move on to the TMR R_0 . It is assumed that further nodes (mesh routers as well as clients) are connected to MMR as indicated by the cloud in Fig. 4 and that these nodes move along with R_{MR} . Furthermore, it is assumed that there is a path from R_0 to R_n and that the best path is $R_1, \dots, R_{n-1}$.⁴

The handover involves a disconnection of R_{MR} from R_n followed by a connection to R_0 . After the disconnection R_{MR} is not able to send nor receive any messages until it is connected to R_0 . Now R_{MR} can send and receive messages again including routing messages. In the extension the MMR R_{MR} sends an Handover Announcement (HA) to SMR R_n shortly before it starts the handover, i.e., before it disconnects from R_n . The announcement is forwarded by SMR R_n to the best next hop R_{n-1} towards TMR R_0 . Each intermediate node on the path from R_n to R_0 forwards the announcement and updates its route to MMR by replacing the current next hop towards R_{MR} with the best next hop towards R_0 . The goal of this procedure is that the route is adapted before the handover is executed and fewer packets are lost than would have been waiting for the routing protocol to propagate the change in the network. Now the handover extension is described in more detail.

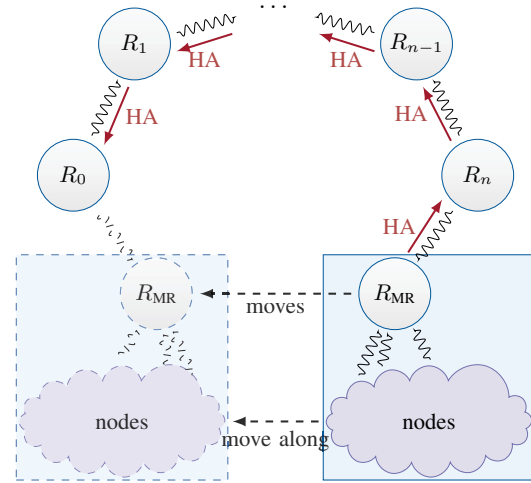


Fig. 4. R_{MR} moving with connected nodes from R_n to R_0

B. Handover Extension in Detail

As above, it is assumed that MMR R_{MR} is currently connected to SMR R_n and wants to handover to TMR R_0 . Furthermore we define: $bh_X(Y) := \{N \in \text{routing table of } X \mid Y \text{ is best next hop towards } N\}$. Before R_{MR} executes the handover, it stops sending ELPs and OGMs to R_n . It still sends ELPs and OGMs to its connected nodes. Then R_{MR} sends a

³The trigger for the handover decision, i.e., when MMR needs to handover is out of scope of this paper.

⁴If there is no path between R_0 and R_n , then the handover announcement cannot be delivered and the regular BATMAN route reestablishment will be used after R_{MR} connects to R_0 , such that the extension is not effective but does also not have a negative effect

HA (see Fig. 5) to R_n consisting of R_{MR} 's originator and interface addresses, and TMR R_0 's originator and interface addresses. The address of R_{MR} 's interface performing the handover and TMR's R_0 interface address to which R_{MR} 's interface will connect after handover are thus included in the HA. This enables each intermediate router R_i including SMR R_n to gain knowledge of the interface address R_{MR} is connecting to. R_{MR} knows the interface address as well as the originator address of R_0 from the beacons R_0 sends out.

SMR R_n handles the HA in the following way. First it deletes MMR from its neighbor list. Then R_n generates a set $bh_{R_n}(R_{MR})$ that contains all originators for which R_{MR} has been the best next hop, i.e., the routers represented by the cloud in Fig. 4. Now R_n looks up the best next hop R_{n-1} towards R_0 . R_n replaces the best next hop for R_{MR} with R_{n-1} . The replacement involves the following steps. For each originator BATMAN V stores the current throughput, last sequence number, last TTL and when the originator was seen last. The throughput to R_{n-1} is also used for the new best next hop for R_{MR} . From the old best next hop the sequence number and TTL are taken. The last seen value is updated with the time when the HA was received. For all nodes in $bh_{R_n}(R_{MR})$ the R_n sets the best next hop to R_{n-1} using the same procedure as for changing the best next hop for R_0 . Now R_n forwards the HA to R_{n-1} containing the addresses of R_{MR} and R_0 , and the originator addresses of each router in $bh_{R_n}(R_{MR}) = HA_{bh}$. HA_{bh} may be updated by each intermediate node.

Each intermediate node R_i with $i \in (1, \dots, n-1)$, i.e., a router that is neither SMR R_n nor TMR R_0 , handles the HA as follows. The intermediate nodes have no direct neighbor entry created by ELP. Therefore, no such neighbor has to be deleted. R_i looks up the best next hop R_{i-1} towards R_0 . The HA contains HA_{bh} . R_i sets the best next hop for each originator in $bh_{R_i}(R_{i+1}) \cap HA_{bh} = bh'$ to R_{i-1} . The same procedure as replacing the next hop for R_{MR} is applied. The throughput is taken from the new best next hop and all other values are copied from the old best next hop. Furthermore, the last seen value is set to the time when the HA was received. Then R_i forwards the HA consisting of the addresses of R_n and R_0 and sets $HA_{bh} = bh'$.

Finally, the TMR R_0 handles the HA in following way. It creates a new neighbor entry on the interface specified in the HA for R_{MR} . Furthermore, if HA_{bh} still contains nodes for which R_1 would have been the best next hop, their best next hop will be set to the new neighbor R_0 . The HA will not be forwarded any further. After R_{MR} has completed the handover it starts sending ELPs and OGMs messages again.

Note that only routers on the path from SMR R_n to TMR R_0 update their routes. Other nodes in the WMN may still route the traffic destined to R_{MR} on another path to SMR or a router on the path, but it will then be directed towards R_0 . After the handover the BATMAN V routing protocol will propagate the change in the WMN and it will finally converge again.

C. Discussion

The rational behind copying the throughput from the best next towards R_0 to the new next hop towards R_{MR} is the following: It is unlikely that the throughput to R_{MR} will be higher than that of R_0 because the packets have to travel along the path to R_0 . If the throughput between R_{MR} and R_0 is higher than the throughput of the whole path it will be adopted by each intermediate router forwarding OGMs from R_{MR} . Therefore, the throughput of best next hop towards R_0 is an upper limit for throughput to R_{MR} after the handover.

A potential drawback of the approach presented here is that a complete list of nodes that are reachable over the moving router has to be sent along with the HA, which, depending on the size of the list, may introduce some overhead. In the worst case, the size of the list results in fragmentation of the HA and thus causes a delay of the HA.

A limitation of the presented approach is that it currently does not cope well with nodes that are connected with R_{MR} before handover but do not move along with R_{MR} . These nodes are currently included in the HA which potentially interferes with the route reestablishment triggered when they reconnect to the network over some other mesh router after R_{MR} has moved out of their range.

D. Implementation

The implementation is based on the *batman_v* branch of BATMAN V.⁵ The content of the HA is shown in Fig. 5. The packets type is set to `BATADV_UNICAST_HOAN` indicating to BATMAN that it is a handover announcement and that it is a unicast message. The version is set to the current BATMAN version, i.e., the version it is compiled for. The TTL is set to the default BATMAN broadcast TTL `BATADV_TTL`. Eight bits are reserved for future usage. This is followed by the MMR originator address and the address of the interface performing the handover. Then the originator address and the handover interface address of the TMR follow. Finally, the number of (other) originators which are reachable via MMR and their originator addresses are included in the announcement. Note that this list is empty when MMR sends out the announcement and is filled by SMR before it forwards the announcement to its best next hop towards TMR.

To initiate the handover procedure on the MMR two virtual files in the sysfs: `handover_lock` and `handover_dest` were added. First, the lock can be set by putting a one into `handover_lock`. This disables sending and receiving of ELPs and OGMs. Then the MAC address of the handover destination interface is put into `handover_dest`. The implementation currently assumes that this coincides with the originator address of the TMR. Note that sending the originator address in the beacons of the TMR would allow to get the originator address of the TMR along with the destinations interface which is just the BSSID of the TMR. So far updating the routes of nodes that are connected to the MMR has not been implemented (and evaluated) yet. The extension

⁵<http://git.open-mesh.org/batman-adv.git>

7		15		31	
Type	Version	TTL		Reserved	
MMR originator MAC					
originator MAC (cont.)		MMR interface MAC			
interface MAC (cont.)					
TMR originator MAC					
originator MAC (cont.)		TMR interface MAC			
interface MAC (cont.)					
# of orig.	orig. addresses				

Fig. 5. BATMAN V HA Packet

can be disabled for compilation and does nothing if it is not triggered via the virtual files mentioned before.

`wpa_suppl` has been modified to trigger the handover extension shortly before the MMR disconnects from the SMR. `wpa_suppl` sets the lock and then sets the destination MAC in `handover_dest`. A command was added to the `wpa_cli` tool allowing to set the BSSID and then initiates a scan so that `wpa_suppl` connects to the Access Point (AP) with the specified BSSID. This BSSID is then written into `handover_dest`.

V. EVALUATION

First, the testbed used for the evaluation is described in Section V-A. In Section V-B the technique used to measure the performance of BATMAN V and the handover extension is detailed. Finally, in Section V-C the measurement results of two evaluation scenarios are presented and discussed.

A. Testbed

The evaluation was performed in our WMN testbed. The testbed consists of MRs with the same hardware and one virtual host called `meshctrl`. Each router has a PC Engines alix system board with an AMD Geode LX800 CPU supporting AES 128 and two identical Qualcomm Atheros AR5416 WLAN chips with one antenna each.⁶ The used operating system on the MRs and the `meshctrl` is Arch Linux with a recent Linux kernel (Version 4.4.1).⁷ The testbed is an infrastructure WMN. On each MR `hostapd` [18] is used to provide an AP on one interface and `wpa_suppl` [18] is used to connect to the WMN on the other. All MRs are authenticated using RSNA [19] with EAP-TTLS by a AAA server running on the `meshctrl`.⁸ In the measurements presented in Section V-C the moving router already shared a cached key with each target router. Therefore, the AAA was not involved during the measurements.

To setup BATMAN V the official guide was followed and OGM aggregation was disabled.⁹ An MG is connected via

cable with the `meshctrl` and potentially other MGs. On the MGs a bridge `br0` with an statically assigned IP was created. Then the `bat0` and `eth0` interfaces were added to the bridge. The wireless interface running the AP was added to the BATMAN interface `bat0`. Then the MG was configured to announce its presence in the network with the corresponding bandwidth. The second wireless interface was disabled on the MGs. On the MRs both wireless interfaces were bridged to the `bat0` interface using `batctl` and a static IP address was assigned to `bat0`.

The MMR was additionally connected via cable with the `meshctrl`. Any commands controlling the MMR were sent over the cable connection. One task of the `meshctrl` was to tell the MMR to which node it had to handover next. The MMR gave feedback to the `meshctrl` when the WLAN handover finished.

B. Measurement Technique

The goal was to measure the time span during which packets are lost due to the handover. In the setup the command line tool `ping` is used to send Echo Request at a specified interval $i = 0.05s$ and the reception of the Echo Request Reply is timestamped. The timespan during which no replies are received also contains the network interruption timespan caused by the WLAN handover which includes scanning for available TMRs, disconnecting from the SMR and connecting to the TMR.

In order to simulate the handover process in such a way that it can be automatically executed several times a simple server application was written. The application runs on the MMR and is used to initiate the handover, notifying the `meshctrl` about the result of the handover along with the information when the handover was started and finished. As in any wireless network packets can get lost due to varying conditions. It was also timestamped when `wpa_suppl` had disconnected and connected. Therefore, it was possible to determine whether ping loss was due to the handover losing the path to the MMR. A route was considered as stable after 50 consecutive successful pings. Then the MR was allowed to move to the next node. Two different scenarios were measured and are described in the following subsection.

C. Scenarios and Results

In the chain scenario the MRs are connected in a chain as shown in Fig. 6. The MMR is initially connected to R_n and then moves to R_{n-1} , then to R_{n-2} and so forth until it reaches R_0 . Then it moves back to R_n by first moving to R_1 , then to R_2 and so forth. This procedure is repeated several times until the resulting values stabilize.

This scenario was evaluated for original BATMAN V and for BATMAN V with the Handover Extension. The results are shown side-by-side in Fig. 7. On the x axis the wireless hop distance between the ping-source S and R_{MR} after the handover is shown. If R_{MR} is connected to R_4 and moves to R_3 the ping source S is 4 wireless hops away from R_{MR} after the handover. Therefore, the route is shortened by one

⁶<http://pcengines.ch/alix3d3.htm>

⁷<https://www.archlinux.org/>

⁸<http://freeradius.org/>

⁹<https://www.open-mesh.org/projects/batman-adv/wiki/Quick-start-guide>

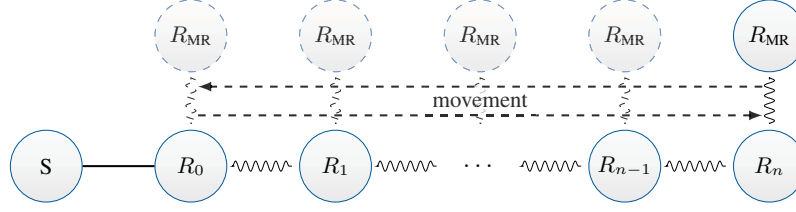


Fig. 6. Chain Scenario Topology (R_{MR} moves subsequently from R_n to R_0 and back)

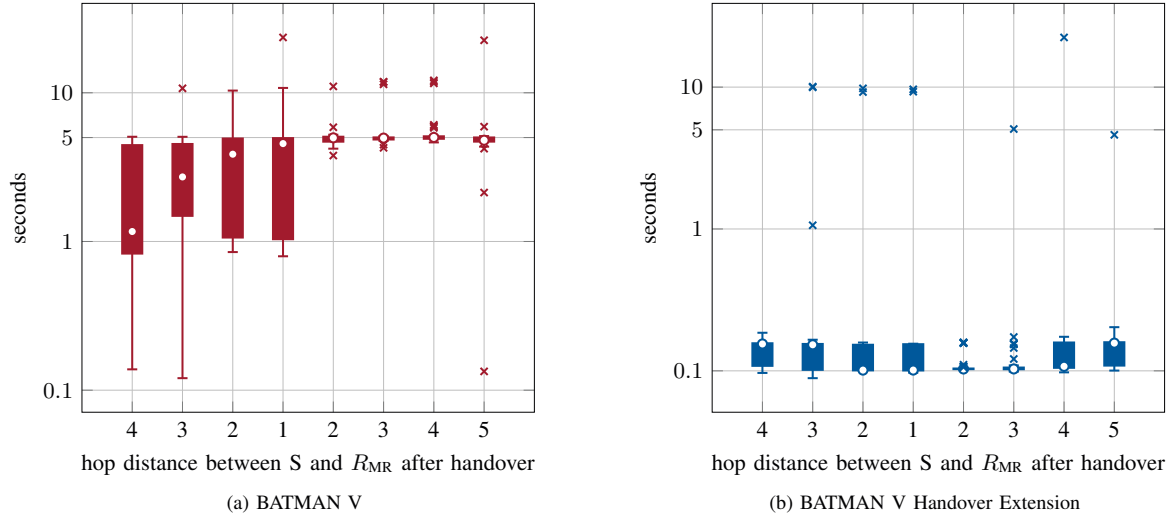


Fig. 7. Chain Scenario Measurement Results

wireless hop. If R_{MR} is connected to R_3 and moves to R_4 then the wireless hop distance between the ping source S and R_{MR} is 5 after the handover. Therefore, the route is lengthened by one wireless hop.

The results clearly show that when the path to the source is shortened than BATMAN V adapts the route quicker than it does if the path to the source is lengthened. But even in the former case, it nevertheless takes more than a second until S receives ping replies again after handover with BATMAN V. If the wireless hop distance increases then BATMAN V needs about five seconds until routes are updated and ping replies are received by S again.

When the BATMAN V handover extension is enabled, packets are received again between 6 to 45 times faster. The timespan during which no ping replies are received lies around 0.1s with the extension and 4.7s without. Furthermore, the variance is much smaller with the extension than without it. The share of the WLAN handover on the timespan during which no packets can be received is 73% with the handover extension and 2% without.

In the branching point scenario shown in Fig. 8 the source is connected to the MG G and there are two branches: one left with MRs from L_1 to L_m and one right with MRs from R_1 to R_n . On each branch the MRs are connected as a chain where the root nodes L_1 and R_1 are connected to the MG G .

In each measurement, the MMR R_{MR} initially is connected

to R_2 . In the first measurement R_{MR} moves to R_2 and back. Therefore, the distance between the ping source S and R_{MR} is always 3 wireless hops. In the next measurement R_{MR} moves to R_1 and back. This time the wireless hop distance between R_{MR} and S when the node connects to L_1 was 2 whereas when R_{MR} moved back it was still 3. In the next measurement R_{MR} moved between B and R_2 . This means it was actually jumping over some intermediate MRs. Finally R_{MR} was connected to R_1 . This means that R_{MR} was actually oscillating between the gateway and a router one wireless hop away from the source. All measurements were repeated until the results stabilized.

The performance of BATMAN V was measured with and without the handover extension. The results are shown in Fig. 9. Pings of S are answered between 40 and 45 times faster again after the handover when the handover extension is used. BATMAN V needs around 4.7s to adapt the route whereas with the extension R_{MR} can be reached in around 0.1s after the handover. The WLAN share of the time span during which no packets are received is 74% with and 2% without the extension.

Some remarks on the outliers. The performance of the extension is in the range of the original BATMAN V if the HA gets lost.

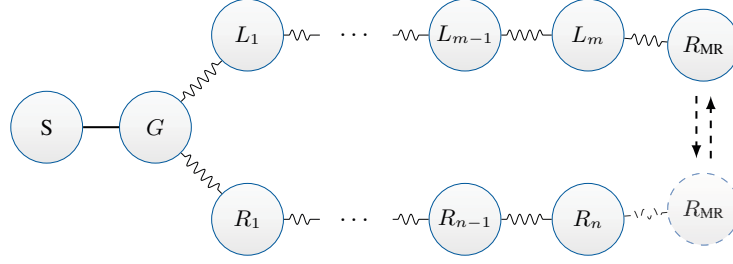


Fig. 8. Branching Point Scenario: R_{MR} moves back and forth between L_m and R_n

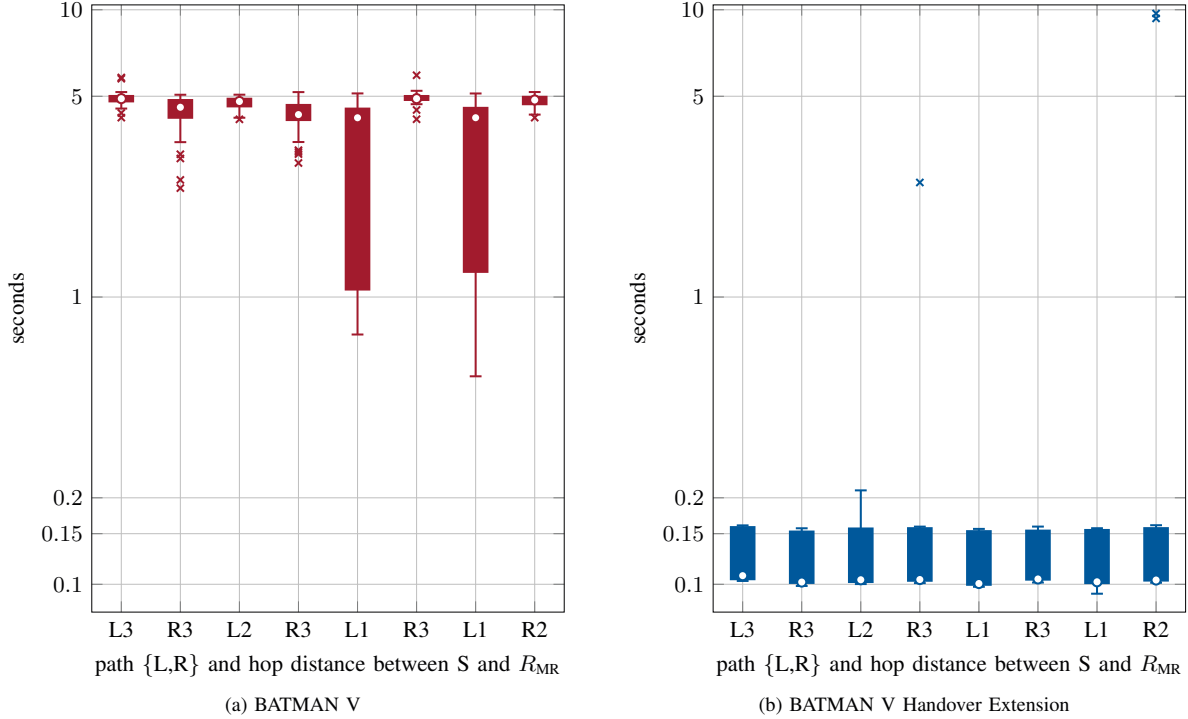


Fig. 9. Branching Point Measurement Results (Time until route to R_{MR} is reestablished)

VI. CONCLUSION AND FUTURE WORK

In this paper an extension to BATMAN V speeding up route reestablishment for moving mesh routers in infrastructure WMNs was proposed. In the extension, the moving mesh router pro-actively announces its wish to handover from some serving mesh router to some target mesh router. The announcement is sent along the currently best path to the target router via the serving router. Each intermediate mesh router on the path updates its route to the moving mesh router (and to nodes moving along with it) by redirecting packets towards the target mesh router. The evaluation of the implementation shows that the extension considerably accelerates route reestablishment such that the WLAN handover itself accounts for the major part of the overall disruption time.

In the future it is planned to incorporate a mechanism that allows the network to reliably detect which nodes connected to a moving mesh router move along with the router and which

do not. In addition, it is planned to integrate the handover extension in our security framework for WMNs.

REFERENCES

- [1] A. Quartulli and R. Lo Cigno, "Improving mesh-agnostic client announcement in BATMAN-advanced," in *2012 IEEE 8th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. Barcelona, Spain: IEEE, 2012, pp. 659–666.
- [2] Better approach to mobile ad hoc networking. [Online]. Available: <https://www.open-mesh.net>
- [3] B. O'Hara, P. Calhoun, and J. Kempf, "Configuration and provisioning for wireless access points (capwap) problem statement," Internet Requests for Comments, RFC Editor, RFC 3990, February 2005.
- [4] A. M. Min-Ho, M. Shin, and W. A. Arbaugh, "Pro-active key distribution using neighbor graphs," *IEEE Wireless Communications*, vol. 11, no. 1, pp. 26–36, 2004.
- [5] *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, Amendment 2: Fast Basic Service Set (BSS) Transition*, IEEE Std. 802.11r, 2007.
- [6] T. Charles Clancy, "Secure handover in enterprise w lans: Capwap, hokey, and ieee 802.11r," *IEEE Wireless Communications*, vol. 15, no. 5, pp. 80–85, 2008.

- [7] S. Rosati, K. Kruzelecki, L. Traynard, and B. Rimoldi, "Speed-aware routing for UAV ad-hoc networks," in *2013 IEEE Globecom Workshops (GC Wkshps)*. Atlanta, GA: IEEE, 2013, pp. 1367–1373.
- [8] G. S. Aujla and S. S. Kang, "Comprehensive evaluation of AODV, DSR, GRP, OLSR and TORA routing protocols with varying number of nodes and traffic applications over MANETs," *IOSR Journal of Computer Engineering*, vol. 9, no. 3, pp. 54–61, 2013.
- [9] D. Kaur and N. Kumar, "Comparative Analysis of AODV, OLSR, TORA, DSR and DSDV Routing Protocols in Mobile Ad-Hoc Networks," *International Journal of Computer Network and Information Security*, vol. 5, no. 3, pp. 39–46, 2013.
- [10] P. Kuppusamy, K. Thirunavukkarasu, and B. Kalaavathi, "A study and comparison of OLSR, AODV and TORA routing protocols in ad hoc networks," in *Electronics Computer Technology (ICECT), 2011 3rd International Conference on*, vol. 5. Kanyakumari, India: IEEE, 2011, pp. 143–147.
- [11] R. G. Garroppo, S. Giordano, and L. Tavanti, "Experimental evaluation of two open source solutions for wireless mesh routing at layer two," in *Wireless Pervasive Computing (ISWPC), 2010 5th IEEE International Symposium on*. Modena, Italy: IEEE, 2010, pp. 232–237.
- [12] A. K. Gupta, H. Sadawarti, and A. K. Verma, "Performance analysis of AODV, DSR & TORA Routing Protocols," *International Journal of Engineering and Technology*, vol. 2, no. 2, pp. 226–231, 2010.
- [13] C. Mbarushimana and A. Shahrabi, "Comparative study of reactive and proactive routing protocols performance in mobile ad hoc networks," in *Advanced Information Networking and Applications Workshops, 2007, AINAW '07. 21st International Conference on*, vol. 2. Niagara Falls, Ontario: IEEE, 2007, pp. 679–684.
- [14] M. Benzaid, P. Minet, and K. Al Agha, "Integrating fast mobility in the OLSR routing protocol," in *Mobile and Wireless Communications Network, 2002. 4th International Workshop on*. IEEE, 2002, pp. 217–221.
- [15] V. Devarapalli, R. Wakikawa, A. Petrescu, and P. Thubert, "Network mobility (nemo) basic support protocol," Internet Request for Comments, IETF Secretariat, RFC 3963, January 2005.
- [16] A. Neumann, C. Aichele, M. Lindner, and S. Wunderlich, "Better approach to mobile ad-hoc networking (b.a.t.m.a.n.)," Working Draft, IETF Secretariat, Internet-Draft draft-openmesh-b-a-t-m-a-n-00, March 2008.
- [17] The cfg80211 subsystem. [Online]. Available: <https://www.kernel.org/doc/html/docs/80211/cfg80211-developers-guide.html>
- [18] hostapd: IEEE 802.11 AP, IEEE 802.1X/WPA/WPA2/EAP/RADIUS Authenticator. [Online]. Available: <https://w1.fi/hostapd/>
- [19] *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, Amendment 6: Medium Access Control Security Enhancements*, IEEE Std. 802.11i, 2004.