Routing Performance of Wireless Mesh Networks: A Practical Evaluation of BATMAN Advanced

Daniel Seither, André König
Multimedia Communications Lab (KOM)
Technische Universität Darmstadt
{daniel.seither, andre.koenig}
@kom.tu-darmstadt.de

Matthias Hollick
Secure Mobile Networking Lab (SEEMOO)
Center for Advanced Security Research Darmstadt
Technische Universität Darmstadt
matthias.hollick@cased.de

***Abstract*— The performance of Wireless Mesh Networks under realistic conditions is not well understood. Given the huge design and parameter space for these networks, all-encompassing performance evaluations are unfeasible.**

We follow a practical approach and perform a targeted, in-depth investigation of the current state of the BATMAN Advanced (batman-adv) protocol in a realistic office environment, using the thoroughly studied AODV protocol as a baseline. In particular, we study the reachability, packet loss, delay and throughput of the network. We identify the main parameters influencing the routing performance and demonstrate failure modes of the studied protocols.

***Index Terms*—WMN; mesh; BATMAN; AODV; testbed; performance evaluation**

I. Introduction

The task of routing protocols for Wireless Mesh Networks (WMNs) is to find reliable routes in multi-hop networks using a mostly unreliable wireless medium. This medium causes problems such as packet loss and link fading, which need to be handled by the protocol.

BATMAN is a proactive routing protocol for WMNs. It uses a distance-vector approach and a routing metric which incorporates the reliability of the radio links. Despite being developed and publicly available since 2006, BATMAN, especially its newer batman-adv variant, has received sparse attention in the scientific community.

In contrast, AODV is a reactive distance-vector routing protocol for ad-hoc networks, using the simpler hop count metric. AODV has been well studied since its initial publication in 1999 and serves as a baseline for our measurements. We also use a modified version of AODV as explained in Section IV-B.

Our contribution is a performance comparison of BATMAN and AODV using commodity hardware in a real-world office environment as shown in Figure 1. Using links provided by the ad-hoc mode of IEEE 802.11, we compare the two protocols in terms of reachability, packet loss, delay and throughput. To the best of our knowledge, this is the largest practical scientific evaluation of batman-adv so far.

We first give a concise overview on BATMAN and AODV in Section II. In Section III, we review related work on performance evaluation of BATMAN. We present setup and results of our experiments in Section IV. Finally, we summarize our contribution and describe further work in Section V.

II. Background

In the following, we introduce the routing protocols we used for our measurements.

A. BATMAN

Better Approach to Mobile Ad-Hoc Networking (BATMAN [1]) is a routing protocol for WMNs, based on distance vector routing. It is proactive in that each node maintains a routing table containing potential next hops to all other nodes forming the WMN. Since there is no full specification for the protocol, it is defined by the daemons implementing it.

BATMAN comes in two flavors: Originally, there was a user space daemon (*batmand*), which routes on Layer 3 like most other routing protocols. In 2007, Layer 2 routing (BATMAN Advanced or *batman-adv*) was introduced, moving the daemon into the Linux kernel for better performance. The development of batmand was mostly stopped later on, focusing the efforts on batman-adv. For our experiments, we used batman-adv 0.2 which we extended to introduce route recording comparable to IP's record route option. This patch was accepted for inclusion into batman-adv after the experiments and is now part of the official distribution.

The protocol uses the Transmission Quality (TQ) metric that was inspired by Expected Transmission Count (ETX [2]) to find a tradeoff between a low hop count and stable links. Every node broadcasts hello messages (also called originator messages or OGM) in fixed intervals to its neighbors. Nodes measure the fraction of hello messages they receive from a given neighbor. This fraction is called the receive quality (RQ). Neighbors rebroadcast received OGMs so that nodes more than one hop away get information about a node's existence. Each node only resends OGMs that were received through the

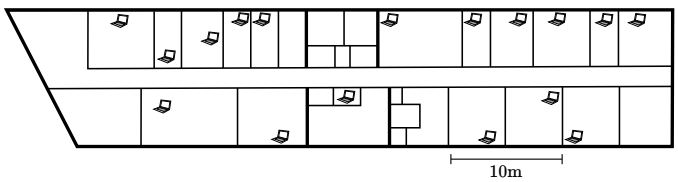


Fig. 1: Top view of Scenario C as described in Table I. Each computer symbol shows the position of a node.

978-1-61284-928-7/11/$26.00 ©2011 IEEE

neighbor with the best metric for the original sender of the message. Nodes measure the fraction of their own OGMs that are resent by neighbors (echo quality, EQ). By dividing EQ by RQ, a node can estimate the fraction of its OGMs that are correctly received by the neighbor (transmit quality or TQ). Finally, penalties for asymmetric links and the number of hops are applied to derive the value of the TQ metric.

Other features of batman-adv include OGM aggregation to reduce the overhead introduced by sending many small frames, optimizations to harness the availability of multiple interfaces and an implementation of a subset of ICMP for Layer 2.

B. AODV

Ad hoc On-Demand Distance Vector Routing (AODV [3]) adapts the well known principle of distance vector routing to the short-lived topology of mobile networks. In contrast to a traditional distance vector algorithm, AODV works in a reactive manner: A new route is only established when data needs to be sent and no route to the destination is known at the sender. AODV uses the hop-count metric which aims at minimizing the number of hops between sender and receiver.

For our experiments, we used AODV-UU [4] from SVN, Revision 15. We needed to apply a small patch that adjusts the code to the latest Linux kernel API changes.

III. Related Work

An extensive amount of work has been done in the field of ad hoc network testbeds [5], for example in the MIT Roofnet project [6]. In the following, we present related work with a focus on the evaluation of BATMAN.

Reineri et al. [7] used ns-2 to simulate a mesh network with one mobile station. They found that BATMAN generally offers better throughput from the mobile station to the infrastructure (uplink) than AODV and the other protocols considered. In the downlink case, the authors needed to apply a small change to the algorithm to achieve comparable results.

Annese et al. [8] applied the changes from [7] to batman-adv and evaluated it on a small testbed with three fixed nodes and one mobile station. The network showed good throughput, but since no other routing protocol was used (especially not plain batman-adv without changes), it is hard to tell whether the unmodified version of the protocol performs as well.

Zeiger et al. [9] deployed batmand and AODV, amongst other protocols, on a mobile robot traveling along a predetermined path and three fixed nodes which were placed along this path. With default settings, BATMAN failed to re-establish a route to the controller node after getting out of range for a direct connection. With a reduced interval for sending OGMs and thus increasing reactivity, the experiment succeeded with an average rerouting time of around 8 s and a packet loss of 8 %. When using AODV, Zeiger et al. found the network in quite an unstable state with rerouting times varying between 2 and over 30 s and a packet loss of 20 %. They could not find settings which significantly improved these results.

Abolhasan et al. [10] built a small indoor testbed of 7 stationary wireless nodes using batmand, AODV, OLSR and Babel. They found that BATMAN was the most reliable and stable protocol under test, while Babel was slightly superior in terms of throughput and convergence time. AODV failed to maintain a consistent multi-hop connection and thus was not evaluated.

Garroppo et al. [11] compared batman-adv to a pre-standard implementation of IEEE 802.11s (open80211s) using a small testbed of 4 nodes. They found that routes established by batman-adv were more stable. However, batman-adv only recovered from a node failure in a route after switching the failed node on again.

Murray et al. [12] compared batmand, batman-adv, OLSR and Babel in a testbed of unknown size. All protocols were able to form a network with low packet loss and similar throughput, with Babel leading the field. The authors explain this result by the comparatively lower routing protocol overhead.

IV. Evaluation

The goals of our evaluation are to assess the ability of BATMAN and AODV to create functional networks in a real-world environment and to identify the aspects influencing the performance of these networks. We focus on the quality of the routes that are established, quantified by the properties described in the following.

A. Metrics

A perfect routing protocol should be able to provide a route between any two nodes in the network (*reachability*). Packets that travel along these routes should arrive without being dropped (*packet loss*), timely enough for interactive applications (*delay*) and fast enough for high-bandwidth transfers (*throughput*). The wireless medium sets limits for these properties as interference and collisions occur, switching between sending and receiving takes small amounts of time and the bandwidth is relatively small when compared to current wireline technologies.

These properties are influenced by multiple parameters of which we will discuss a subset that showed to affect the metrics considered strongly: The *location* of the communicating nodes is relevant as it determines for each node the set of neighbors that can be reached directly and the quality of these links. The time at which communication takes place can influence the amount of *interference* due to other users of the 2.4 GHz band. Finally, the choice of *routing protocols* and their configuration determines the performance of the network relatively to a network with perfect routing.

B. Experimental Design

To get a valid estimate of the impact that the routing protocol has on the network, we built a testbed consisting of netbooks (cf. Table I) that were placed in different rooms of an office building. The wireless interfaces were configured to ad-hoc mode on a channel that was in use by some third-party nodes since no free non-overlapping channel was available.

TABLE I: Experimental Design

Hardware Platform	Asus Eee PC 1005HA-H
WLAN Chipset	Atheros AR9285
Operating System	Ubuntu 9.10 i386
Kernel	Linux 2.6.31-wl
WLAN Driver	ath9k
Traffic Pattern I	Send 50 ICMP ping messages (interval: 1 s) from Node 1 to each other node consecutively, with record route option set
Command (L3)	`ping -R -c 50 -i 1 <dest>`
Command (L2)	`batctl ping -R -c 50 -i 1 <dest>`
Traffic Pattern II	Send TCP stream with unlimited bandwidth from Node 1 to each other node consecutively for 30 s
Command	`iperf -c <dest> -t 30`
Scenario A	3 netbooks, placed in adjacent rooms of an office building during working hours. All nodes were in direct radio range, thus, no routing protocol was used.
Scenario B	14 netbooks, sparsely distributed over two levels of the building during working hours (cf. Figure 2)
Scenario C	17 netbooks, distributed over one level; measurements taking place after working hours (cf. Figure 3)

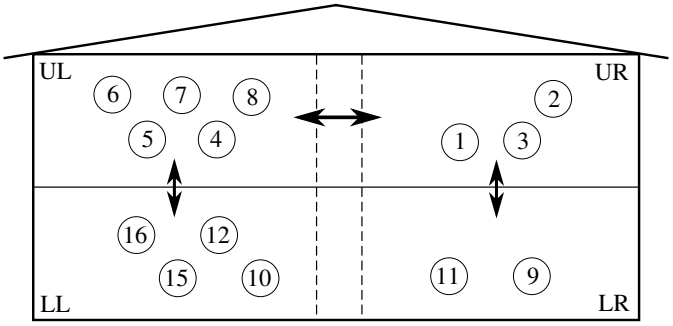


Fig. 2: Node distribution for Scenario B. The upper/lower left/right section is denoted by {U,L}{L,R}. Dashed lines show reinforced concrete walls that significantly affected radio conditions.

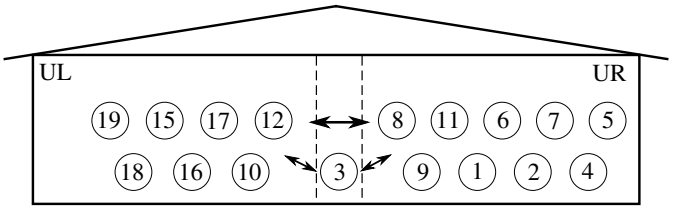


Fig. 3: Node distribution for Scenario C. The upper left/right section is denoted by U{L,R}.

After configuring the nodes with the protocols, the properties outlined above were measured quantitatively.

To measure reachability, packet loss and delay, the networks created by both protocols were – one at a time – tested using ping messages, sent from Node 1 to each of the other nodes consecutively (Traffic Pattern I). We sent 50 packets per node at an interval of one second. For AODV, we used the regular Layer 3 ping command with the option to record every host on the path to the destination and back to the sender. batman-adv routes on Layer 2 which leads to every other host seeming to be placed at a distance of one hop on Layer 3. The routing daemon therefore brings its own implementation of the ping command on Layer 2, which we used to achieve a behavior similar to the regular command.

As a second traffic pattern, we used *iperf* to measure TCP throughput for connections from Node 1 to each other node. After 30 seconds, we logged the amount of data that was correctly received and acknowledged.

Since we could not control the behavior of stations that were not part of our testbed but used the same 802.11 channel, we needed to monitor whether the radio conditions were comparable during our consecutive tests of both protocols. This was done at each node by monitoring the received signal strength (RSS) of all other nodes using the same channel.

AODV, using the hop-count metric, prefers routes with few rather long links and potentially poor reliability over more but also more reliable links. To remediate this problem, we apply a neighbor selection process [13] in selected measurements to limit the set of possible neighbors to those that can be reached with sufficient signal strength. The neighbor selection process runs on every node and continuously observes the channel. Initially, receiving from all other nodes is blocked using the Linux iptables infrastructure. When the RSS for frames from a neighbor exceeds a chosen threshold, a notification is sent to the neighbor's instance of the neighbor selection process. If the link in the opposite direction also fulfills the RSS requirements, the neighbor replies with an acknowledgment and both nodes unblock the link to make higher-layer communication possible. If, at a later point in time, the average RSS falls below the chosen threshold minus 5 dBm, the link is blocked again and the neighbor is notified to do the same.

C. Scenarios

Our experiments were conducted in three scenarios, featuring different node locations and radio conditions.

Scenario A: To establish a baseline for our later measurements, we placed three netbooks in adjacent rooms of the building during working hours. Running experiments during working hours puts an additional burden on the interfaces as some other stations use the same channel and other devices using the 2.4 GHz band (e.g. Bluetooth) could interfere. The wireless interfaces were configured to use ad-hoc mode but no routing daemon was started so that only single-hop communication was possible.

Scenario B: We started the actual comparison of the routing protocols by distributing 14 netbooks over two adjacent levels of the building. The building's structure divided the nodes into four internally well connected sections, separated by a floor or wall made of reinforced concrete (see Figure 2). Communication between the sections was only possible on the upper level and between the vertically adjacent sections.

Scenario C: We placed 17 nodes in only the upper level of the building (see Figures 1 and 3) and performed our

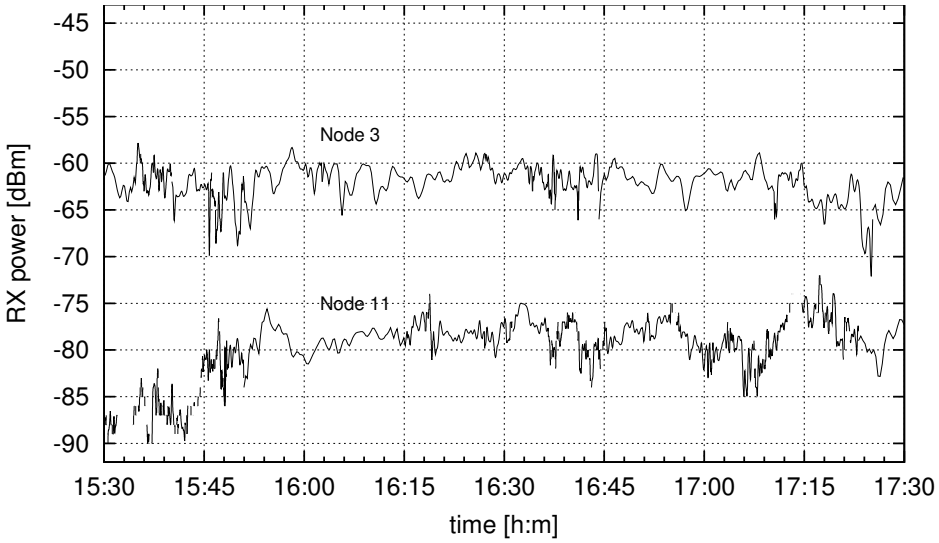


Fig. 4: Receiving signal strength of selected neighbors, measured on Node 1 in Scenario B during working hours.

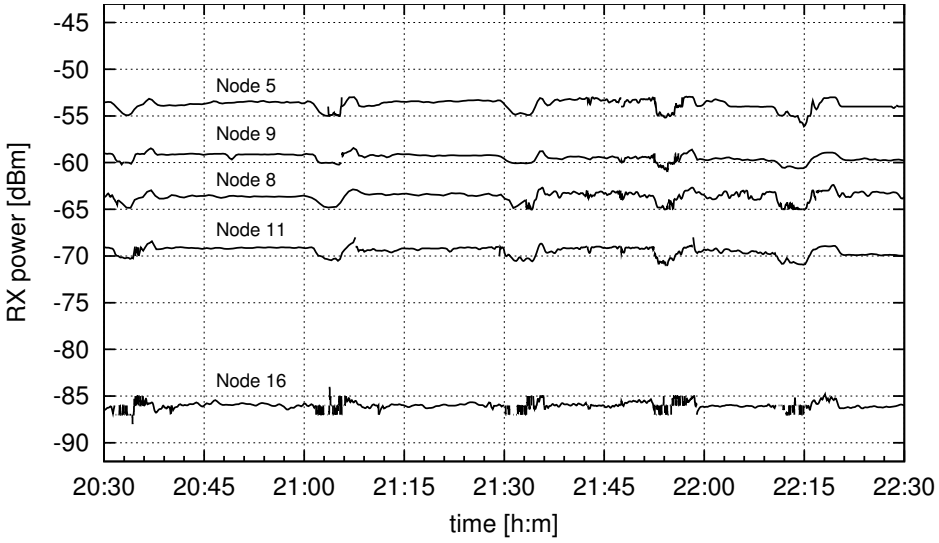


Fig. 5: Receiving signal strength of selected neighbors, measured on Node 1 in Scenario C after working hours.

measurements after working hours, thus increasing RSS and reducing interference. The nodes were split into two well connected sections, separated by reinforced concrete walls. Between these sections, an additional node was placed to improve connectivity.

D. Analysis of the Results

In the following, we present the results of our measurements. We start with discussing the radio conditions that were faced during the experiments. Afterward, we evaluate the network's performance with regard to the metrics described in Section IV-A. Finally, we look into route switching behavior under normal conditions and in presence of node failures.

1) Radio Conditions: In Scenario A and B, we found similar conditions in terms of interference, since both experiments took place during working hours. Figure 4 shows a selection of two RSS profiles that were characteristic for these setups. It can be seen that the RSS was subject to serious fading.

In Scenario A, all three nodes could receive the signals of the other nodes with a signal strength of -57 to -67 dBm and a mean deviation between 1.1 and 2.9 dBm.

For half of the nodes in Scenario B, including all the nodes on the lower level, the RSS for the neighbor with the strongest signal stayed below -60 dBm. Most of the nodes could only receive signals of one to three nodes with a signal strength of more than -70 dBm. The mean deviation of the RSS amounted to between 1.3 and 4.6 dBm. We attribute the differences between the results for Scenarios A and B to the greater number of nodes that can interfere and the larger distance between these in B.

In Scenario C, only one node in both UL and UR could receive its strongest neighbor with less than -60 dBm. Most

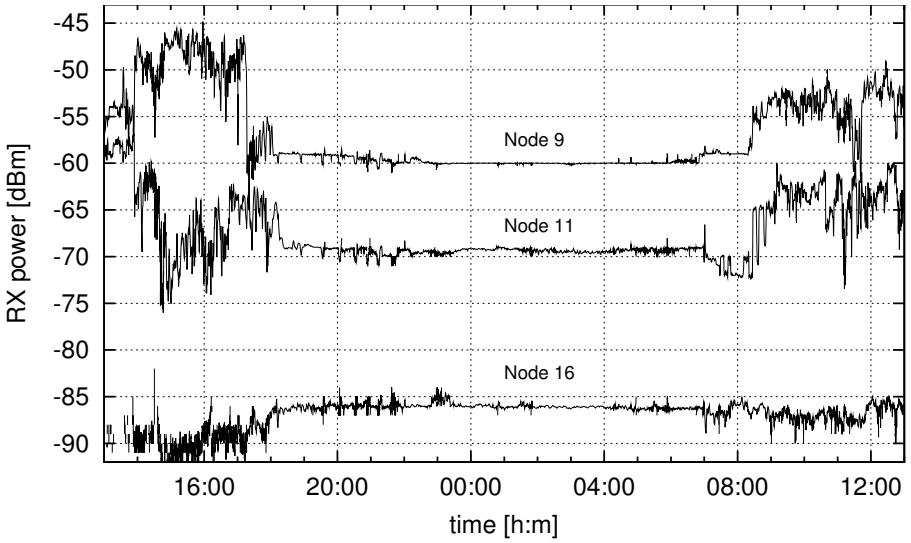


Fig. 6: Receiving signal strength of selected neighbors, measured on Node 1 in the setup from Scenario C over 24 hours, showing the difference between daytime and nighttime interference.

nodes received two to four neighbors with a signal strength between -40 and -60 dBm. Since the experiments were conducted after working hours, only a few idle access points shared the used channel which led to more stable receiving conditions than in Scenario B. In Figure 5, the RSS of the neighbors which could be received at Node 1 is shown. Compared to the second experiment which was conducted during working hours (Figure 4), it can be seen that there was much less fading in Scenario C. The mean deviation of the RSS was reduced to around 0.7 dBm for all neighbors.

To be sure that this significant change was not an error in our measurements, we measured the RSS over a period of 24 hours (cf. Figure 6). The results show that the variability of RSS is much higher during working hours which indicates a higher level of interference than can be observed at nighttime.

2) Reachability and Packet Loss: All nodes in Scenario A were in direct radio range. When applying Traffic Pattern I to the network, there was no packet loss.

Scenario B led to a topology where four sections could be identified that were well connected internally, separated by concrete walls and the floor (cf. Figure 2). Only the vertically adjacent sections and the sections on the upper level could communicate directly with each other.

In this environment, batman-adv succeeded in creating a usable network. With one exception, the packet loss between Node 1 and all other nodes did not exceed 6 % (cf. Table II).

AODV showed a largely different behavior. Without neighbor selection as described in Section IV-B, only nodes in Sections UR (which contains Node 1) and LR could be reached with relatively low packet loss of up to 6 %. This is exactly the set of nodes which Node 1 could reach directly without using intermediary hops, that is, without making use of the routing protocol. The loss for all other nodes amounted to between 26 % and 100 %. This basically shows that all routes which AODV established were of insufficient quality for usual applications.

To improve the performance of AODV, we activated the neighbor selection filter. With the filter set to -82 dBm, no significant changes of the network's behavior could be measured. Stronger filters immediately lead to a partitioned network.

In Scenario C, batman-adv showed a behavior similar to the results in B. The packet loss to each of the stations did not exceed 4 % (cf. Table III). In Figure 7a, it can be seen that batman-adv makes heavy use of asymmetric links and, due to the TQ metric, prefers short links over short paths.

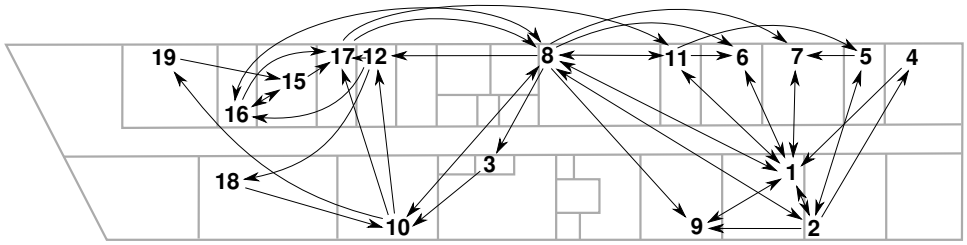
In contrast, unfiltered AODV created a spanning tree of all reachable nodes using only bidirectional links (cf. Figure 7b), which is the expected behavior of a protocol that assumes a symmetric channel. Connections to most of the nodes in UL showed packet loss rates of 20 % to 30 %, while nodes in UR could all but one be reached without loss, using only single-hop connections that do not require routing. Three nodes could almost never be reached via ping.

When applying the neighbor selection filter to the AODV network, the situation could almost never be improved. Since Node 8 which served as an important intermediate node for connections between UL and UR received signals from its nearest neighbors in UL with less than -80 dBm, and other connections between the sections were comparably weak, the stronger filters completely blocked communication between the sections (cf. Figures 7d through 7f). Only the weakest filter (-82 dBm, Figure 7c) allowed for connections but performed roughly like unfiltered AODV.

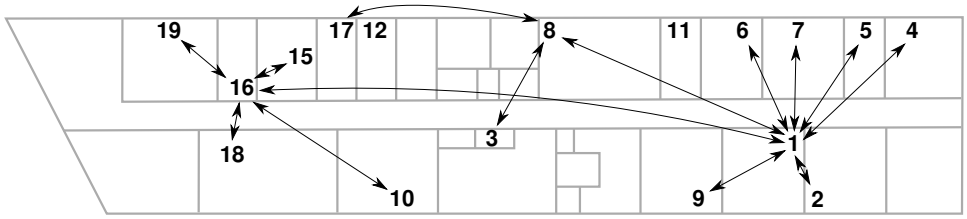
3) Delay: In Scenario A with direct connectivity, the round trip times amounted to 5 ms with little variance.

The RTT for batman-adv in Scenario B was around 20 ms without much variation. The round trip times on the AODV network were comparable to the results for batman-adv with the exception of higher variance.

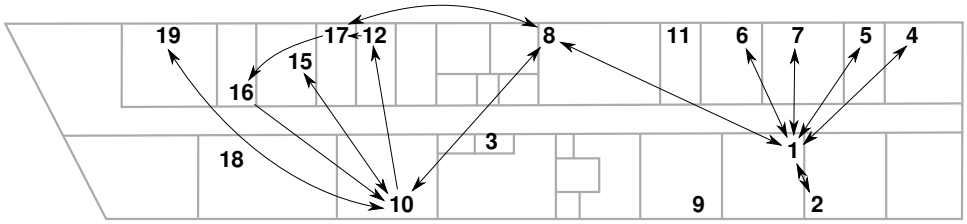
With batman-adv in Scenario C, we measured an RTT of



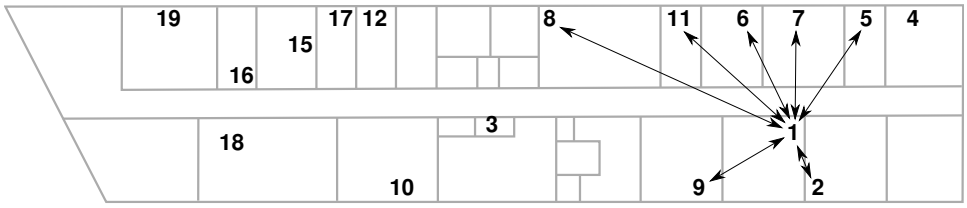
(a) BATMAN



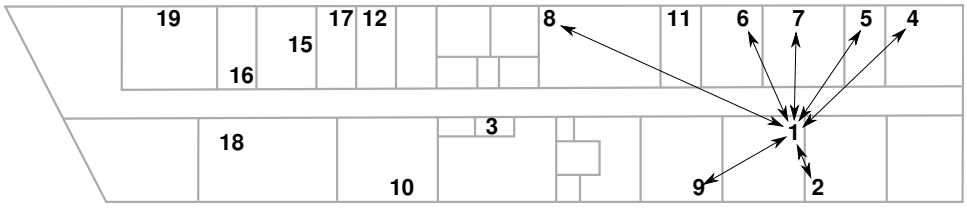
(b) AODV



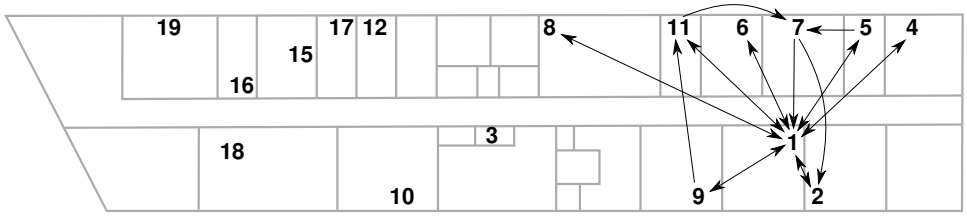
(c) AODV-82



(d) AODV-77



(e) AODV-72



(f) AODV-67

Fig. 7: Links that were utilized in Scenario C for Traffic Pattern I. For each pair of nodes, only links belonging to the route which was used for the highest number of successful transmissions are shown. Since not all links were used bidirectionally, arrows are used to indicate the direction of data transmission. AODV-xy denotes AODV with an active neighbor selection filter with the given threshold.

TABLE II: Reachability and median number of hops for a round-trip in Scenario B

Sec.	Node	BATMAN	AODV, no filter	AODV, 82 dBm filter
UR	2	1.00 (3)	1.00 (2)	1.00 (2)
	3	1.00 (2)	1.00 (2)	1.00 (2)
UL	4	0.96 (7)	0.14 (6)	–
	5	1.00 (8)	0.08 (5)	–
	6	1.00 (7)	0.46 (4)	0.66 (5)
	7	1.00 (8)	0.22 (7)	0.52 (5)
	8	1.00 (5)	0.74 (4)	0.50 (4)
LR	9	1.00 (4)	0.98 (2)	1.00 (2)
	11	1.00 (5)	0.96 (2)	0.82 (2)
LL	10	1.00 (10)	0.16 (5)	0.38 (6)
	12	0.94 (7)	0.60 (4)	–
	15	1.00 (9)	–	–
	16	0.46 (11)	0.26 (4)	–

TABLE III: Reachability and median number of hops for a round-trip in Scenario C

	BATMAN	AODV				
		none	82 dBm	77 dBm	72 dBm	67 dBm
2	1.00 (2)	1.00 (2)	1.00 (2)	1.00 (2)	1.00 (2)	1.00 (2)
4	1.00 (3)	1.00 (2)	1.00 (2)	–	1.00 (2)	1.00 (2)
5	1.00 (4)	1.00 (2)	1.00 (2)	1.00 (2)	1.00 (2)	0.94 (2)
6	1.00 (2)	1.00 (2)	1.00 (2)	1.00 (2)	1.00 (2)	1.00 (2)
7	1.00 (2)	1.00 (2)	1.00 (2)	1.00 (2)	1.00 (2)	0.96 (4)
8	1.00 (4)	1.00 (2)	1.00 (2)	1.00 (2)	1.00 (2)	1.00 (2)
9	1.00 (2)	1.00 (2)	–	1.00 (2)	1.00 (2)	1.00 (2)
11	1.00 (4)	–	–	1.00 (2)	–	0.98 (4)
3	1.00 (6)	0.06 (4)	–	–	–	–
10	0.98 (7)	0.64 (4)	–	–	–	–
12	1.00 (7)	–	0.26 (6)	–	–	–
15	1.00 (7)	0.78 (4)	0.76 (6)	–	–	–
16	1.00 (8)	1.00 (2)	0.78 (6)	–	–	–
17	1.00 (8)	0.68 (4)	0.82 (4)	–	–	–
18	0.96 (7)	0.96 (4)	–	–	–	–
19	0.98 (8)	0.84 (4)	0.86 (6)	–	–	–

about 10 ms for nodes in the right section and 20 to 30 ms for the other nodes. AODV differs from batman-adv only in UL: Due to shorter routes, the delay is generally slightly lower (cf. Figure 8). Since the neighbor selection filter suppresses routes to nodes outside the right section for thresholds of -77 dBm and higher, the mean RTT is lowest for this configuration due to short routes. When applying the filter with a threshold of -82 dBm, routes established by AODV consist of more hops than without a filter which increases the mean RTT.

4) Throughput: We observed no packet loss and a TCP throughput of between 700 and 800 kBit/s in Scenario A. The relatively low throughput seems to be a problem with rate control of the wireless card and/or its driver when operating in ad-hoc mode since performance did not improve when the nodes were placed in the same room with a spacing of only 1 m. The hardware and software we were using is able to achieve significantly higher rates in infrastructure mode.

In Scenario B, a TCP connection with a throughput of 100 to 650 kBit/s could be established with all nodes when using batman-adv. The throughput was roughly inversely proportional to the number of hops between source and destination. We did not perform throughput measurements with AODV because almost no stable multi-hop route was available.

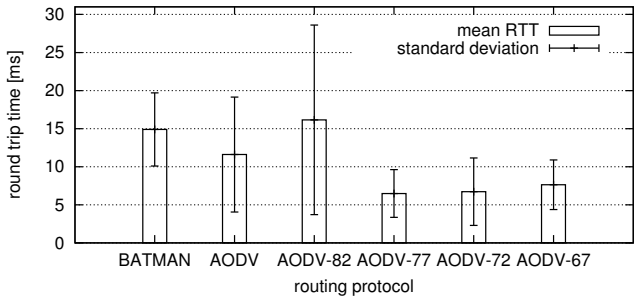


Fig. 8: Mean round trip times, averaged over all ping messages for a given protocol, in Scenario C. AODV-xy denotes AODV with an active neighbor selection filter with the given threshold.

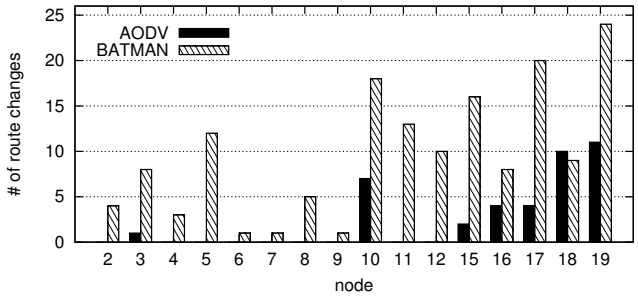


Fig. 9: Number of route changes per 50 round-trips in Scenario C.

Scenario C finally gave us the opportunity to compare the throughput using routes established by batman-adv and AODV. batman-adv performed as in Scenario B, with the exception of slightly higher data rates when comparing routes with the same number of hops, supposedly due to shorter links.

Using the mostly single hop routes to nodes in UR, both protocols performed similarly. When transmitting to nodes in UL, excessive packet loss prohibited AODV in most cases from providing routes that were reliable enough to transmit significant amounts of data.

In the cases where AODV produced stable routes, both routing protocols performed alike. We attribute this to AODV establishing routes with fewer hops and thus requiring less airtime to complete a transfer, but at the same time triggering more TCP retransmissions due to higher packet loss. We will investigate this effect in further studies.

5) Route Changes and Convergence: Figure 9 shows the number of route changes that the packets from Traffic Pattern I experienced in Scenario C. As can be seen, BATMAN causes routes to change at a much higher rate than AODV, which can be explained by looking into the route setup process.

When a route to a given destination is needed and none

is available, AODV, as a reactive protocol, establishes a new route if the destination is reachable. This route remains unchanged as long as it does not break due to a link failure.

BATMAN, however, is proactive and, thus, maintains a table of all reachable nodes and a list of potential next hops for each node. From this list, BATMAN chooses the node with the best metric for the destination. Since the metric may change over the course of time, the protocol chooses a next hop offering a better metric as soon as it gets available. This leads to more changes that usually do not lead to a disruption of the data flow since, in most cases, the old route is still working and can be used to deliver packets that are still in transit.

Finally, we were interested in the time until recovery if a node which is used in a route gets dysfunctional. This was analyzed in Scenario C, since it provided the protocols with more choices of possible routes to a given destination. We used the ping program to send a message every second to a node which was – using the shortest route – two hops away from the measurement node. The output of ping was used to observe the reachability of the destination and the actual route that was used by the messages. We then switched off the active routing protocol on one of the intermediate nodes used for forwarding the messages. The time until recovery using an alternative route was measured by the number of dropped messages after the topology change. Afterward, the intermediate node was switched on again and the routing protocol was given at least 30 seconds to reach a stable state. This procedure was followed 20 times for both routing protocols.

AODV took a mean time of 3.5 s to restore connectivity, batman-adv needed around 5 s. In two cases, AODV took much longer than the mean value which leads to a standard deviation of 1.8 s while batman-adv showed a constant behavior with a standard deviation of 1.0 s. The relatively high variance for AODV follows from searching for a new route and repeating the RREQ if no RREP is received after a certain time. Since batman-adv stores multiple next hops in its routing tables, it only needs to detect the failure of the old route to choose a new next hop. We attribute the relatively long period until restoration to BATMAN attenuating rapid changes in the TQ metric by averaging over the last n received values. A detailed evaluation of this effect will be part of our future research.

V. Conclusion

We evaluated the performance of the BATMAN routing protocol in a real-world office environment, using plain AODV and a modified version as a baseline. BATMAN was proven to work as well in an environment with significant interference and longer distances between nodes as in a friendlier setting. In contrast, AODV often failed to provide stable routes in cases where multi-hop communication was necessary.

In our further studies, we are going to investigate opportunities for improvements of the BATMAN protocol, especially concerning faster convergence after node failures. Sending hello messages with a variable interval that is shortened in the case of rapid TQ changes might be a promising approach.

References

[1] M. Lindner, S. Eckelmann, S. Wunderlich *et al.*, "The B.A.T.M.A.N. Project." [Online]. Available: http://www.open-mesh.org/

[2] D. S. J. De Couto, D. Aguayo, J. Bicket, and R. Morris, "A high-throughput path metric for multi-hop wireless routing," *Wireless Networks*, vol. 11, pp. 419–434, 2005.

[3] C. Perkins, E. Belding-Royer, and S. Das, "RFC 3561: Ad hoc On-Demand Distance Vector (AODV) Routing," http://www.ietf.org/rfc/rfc3561.txt, 2003.

[4] E. Nordström and H. Lundgren, "AODV-UU: Implementation of AODV from Uppsala University." [Online]. Available: http://sourceforge.net/projects/aodvuu/

[5] W. Kiess and M. Mauve, "A survey on real-world implementations of mobile ad-hoc networks," *Ad Hoc Networks*, vol. 5, no. 3, pp. 324 – 339, 2007.

[6] J. Bicket, D. Aguayo, S. Biswas, and R. Morris, "Architecture and evaluation of an unplanned 802.11b mesh network," in *Proc. MobiCom*, ser. MobiCom '05. New York, NY, USA: ACM, 2005, pp. 31–42.

[7] M. Reineri, C. Casetti, and C.-F. Chiasserini, "Routing protocols for mesh networks with mobility support," in *Proc. ISWCS*, 2009.

[8] S. Annese, C. Casetti, C. F. Chiasserini, P. Cipollone, A. Ghittino, and M. Reineri, "Assessing mobility support in mesh networks," in *Proc. WiNTECH*, 2009.

[9] F. Zeiger, N. Krämer, M. Sauer, and K. Schilling, "Mobile robot teleoperation via wireless multihop networks - parameter tuning of protocols and real world application scenarios," in *Informatics in Control, Automation and Robotics*, ser. LNEE. Springer, 2009, vol. 37, pp. 139–152.

[10] M. Abolhasan, B. Hagelstein, and J. C.-P. Wang, "Real-world performance of current proactive multi-hop mesh protocols," in *Proc. APCC*, 2009.

[11] R. G. Garroppo, S. Giordano, and L. Tavanti, "Experimental evaluation of two open source solutions for wireless mesh routing at layer two," in *Proc. ISWPC*, 2010.

[12] D. E. Murray, M. Dixon, and T. Koziniec, "An experimental comparison of routing protocols in multi hop ad hoc networks," in *Proc. ATNAC*, 2010.

[13] C. Gottron, P. Larbig, A. König, M. Hollick, and R. Steinmetz, "The Rise and Fall of the AODV Protocol: A Testbed Study on Practical Routing Attacks," in *Proc. LCN*, 2010.