

Real-World Performance of current Mesh Protocols in a small-scale Dual-Radio Multi-Link Environment

Manuel Hachtkemper, Michael Rademacher, Karl Jonas
Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin, Germany
EMail: firstname.lastname@h-brs.de

Abstract

Two key questions motivated the work in this paper: What is the impact of different usage schemes for multiple channels in a dual-radio Wireless Mesh Network (WMN), and what is the impact of some popular WMN routing protocols on its performance. These two questions were evaluated in a small and simple real-world scenario. A major concern was reproducibility of the results. We show that it is beneficial to use both radios on different frequencies in a fully meshed environment with four routers. The routing protocols Babel, B.A.T.M.A.N. V, BMX7 and OLSRv2 recognize a saturated channel and prefer the other one. We show that in our scenario all of the protocols perform equally well since the protocol overhead is comparably low not influencing the overall performance of the network.

1 Introduction

Broadband connectivity for rural areas is one important challenge in bridging the digital divide. Among others, WiFi-based Long Distance networks (WiLD) are one option to connect a rural village to the next peering point. WiLD solutions exist, such as the Fraunhofer WiBACK technology: It facilitates the usage of off-the-shelf hardware components like WiFi radios and directional antennas combined with intelligent network management to allow an overall cost-effective backhaul solution [6].

To provide local access for the users inside the rural village, again a cost-effective technology is needed. One solution is a WiFi based WMN using omni-directional antennas. A WMN is considered as a multi-hop wireless network, in which mesh nodes relay traffic on behalf of other mesh nodes or connected clients (and networks). In the initial design of WMN, these nodes were equipped with only one radio and a single channel was used for the communication [7].

In this work, we evaluate a fully meshed local access WMN with four dual-radio WiFi routers. Since two independent radios (operated on different channels) are available at each router, different functional assignment schemes for these interfaces exists: A radio can serve solely as mesh interface, solely as client access interface or it provides both functions simultaneously. Our goal is to prospect performance differences if one radio serves both functionalities or if its preferable to dedicate one interface solely to access. Furthermore, several WMN routing protocols are available for this scenario. Our goal is to identify possible performance differences among these protocols.

The rest of this work is structured as follows: Section 2 provides a brief introduction of commonly used WMN routing protocols. Reference to related work dealing with performance comparison of WMN protocols is provided in Section 3. Section 4 describes our testing environment, followed by a summary of our results and some further dis-

ussion in Section 5. The raw data of all measurements, the configuration files and scripts have been made publicly available¹.

2 WMN Routing Protocols

The purpose of this section is to provide a brief overview about the WMN routing protocols evaluated in this work. For additional details, we refer the reader to the citations given for each protocol.

Babel is a proactive distance-vector routing protocol. In order to discover neighbors, every node periodically sends *Hello* messages on all interfaces to all nodes within direct radio range. To establish bi-directionality, a node furthermore sends periodic *I heard you (IHU)* messages. In this way, having both their own and the neighbor's calculation, a node can compute the link cost. Nevertheless, the RFC explicitly states that all cost computations are a subject of local matter [3].

Better Approach To Mobile Adhoc Networking (B.A.T.M.A.N.) advanced² is a proactive routing protocol which operates on layer 2 of the OSI model [12]. In B.A.T.M.A.N. V *Echo Location Protocol (ELP)* messages are used to identify neighbors. Every node periodically broadcasts these messages (default interval 0.5s) on all available B.A.T.M.A.N. interfaces to all potential neighbors within direct radio range [14]. In contrast to previous versions, metrics in B.A.T.M.A.N. V are not packet-loss-based. Instead, throughput-based metrics are used exploiting the capabilities of modern wireless drivers. The driver needs actual traffic for this estimation. Therefore, if a link is idle, B.A.T.M.A.N. generates traffic [13].

BatMan-eXperimental version 7 (BMX7) is a proactive distance-vector routing protocol [2]. BMX7 aims to make

¹<http://mc-lab.inf.h-brs.de/paper/dualradio/>

²The initial protocol version operates on layer 3.

the concepts of BMX6 more flexible and secure. Parts of the protocol are cryptographically signed to allow a determination if a specific node is trustworthy for routing packages [9, 10]. Information about the current version BMX7 is rare, therefore, some of the following information is taken from BMX6. Every node periodically sends (default interval 0.5s) a *Hello* message as multicast to all nodes within direct radio range. To establish a bi-directional relationship, every node sends *Report* messages at the same interval on all interfaces. The reports contain the number of *Hello*s received on this particular interface by every node. Therefore, a node knows both the percentage of *Hello*s received by a neighbor and how many *Hello*s the neighbor has received to compute both the transmission and reception cost [11] respectively.

Optimized Link State Routing Protocol version 2 (OLSRv2) is a proactive link state routing protocol. The neighbor discovery is based on the mobile ad hoc network (MANET) Neighborhood Discovery Protocol NHDP [4]. It is designed to find the 1- and 2-hop neighborhood of each node. In order to accomplish this, every node periodically sends *HELLO* messages on every interface to all nodes within direct radio range (default interval 0.2s). The *HELLO* messages include a list of neighbors, which a node has recently seen. By receiving a *HELLO*, a node learns the existence of its 1-hop and 2-hop neighbors. In OLSRv2 link metrics are directional [4, 5].

3 Related work

There exists a significant amount of work addressing the performance of WMN protocols with single-radio routers in different scenarios and only a selected subset is presented in this section. However, mostly older protocol versions and setups with multi-hop routes are addressed.

The authors in [11] show that an increase in network size results in a linear increase of the protocol overhead for OLSR, BatMan-eXperimental version 6 (BMX6) and Babel. The additional impact on CPU and memory utilization is marginal.

The authors in [8] evaluate Babel, B.A.T.M.A.N. and OLSR. They show that Babel provides the best throughput in a multi-hop scenario. Additionally, the authors show that the layer 2 implementation of B.A.T.M.A.N. slightly outperforms the legacy layer 3 implementation and Optimized Link State Routing Protocol (OLSR) [8]. In their scenario, Babel has the lowest overhead with 0.35% channel occupancy, followed by both the layer 2 and 3 implementations of B.A.T.M.A.N. with 3.3% and OLSR with 8.4%.

The authors in [1] describe that B.A.T.M.A.N. has a more stable operation and the highest packet delivery rate since routes changes are less frequent. Similar to the results in [8], Babel provides the best throughput.

We are not aware of any comparisons of recent WMN routing protocols using dual-radio routers.

4 Experiments

Our general scenario consists of four fully meshed wireless routers. One of these routers is connected to a PC acting as traffic source and test controller. The three other routers act as access points (APs) for six clients, with two clients connected to each router. The controller sends artificial TCP traffic (using iperf) with large payloads saturating the link to all clients simultaneously. Three different setups are evaluated:

1. A single-radio WMN (the second radio is unused). The same channel is used for the mesh and the client access (cf. **Figure 1a**).
2. One channel/radio for the mesh network and a different channel/radio for all clients (cf. **Figure 1b**).
3. A joint wireless channel/radio for both the mesh network and client access, and a second channel/radio for the mesh network (cf. **Figure 1c**).

All routers and clients are within direct radio range to each other. The first setup is used as a baseline to compare a possible throughput increase with dual-radio routers. The goal of the second and third setup is to compare if it is better to use both radios for the mesh network or use a dedicated access radio. Since we are interested in the behavior of the different protocols in all setups, the protocol overhead is observed as well. Additionally, it is evaluated if a mesh protocol builds paths among multiple hops and which channel a router chooses in Setup 3.

4.1 Methodology

4.1.1 Hardware and Software

Software	Router	Client	Controller
iperf	-	2.0.5-1	2.0.9-1
babeld	1.7.1-1	-	-
kmod-batman-adv	4.4.14+2016.4-0	-	-
bmx7	r2016072001-4	-	-
oonf-olsrd2	0.13.0	-	-

Table 1 Software used during the experiments.

Four MikroTik BaseBox 2 wireless routers are used to build the mesh network. Since each router has only one radio pre-installed, a MikroTik R11e-5HnD with an Atheros AR9580 chipset is added. The pre-installed radio uses the 2.4 GHz and the supplementary ones the 5 GHz band, with two omnidirectional antennas for each radio. The clients are four TP-Link TL-WDR4300 (v1.7) and two TP-Link TL-WDR3600 (v1.5). The controller is a Lenovo ThinkPad X230 with an Intel Core i5-3320M processor, 4 GB of RAM and an Intel 82579LM gigabit Ethernet controller. All routers run a self compiled OpenWrt trunk version³ which is necessary since B.A.T.M.A.N. V needs to be explicitly activated in the kernel configuration. A fresh installation of OpenWrt Chaos Calmer (15.05.1) is used on all

³Last commit: 3f98448d670ab2e908c8d6002d8c6f8ff5d1d9bd

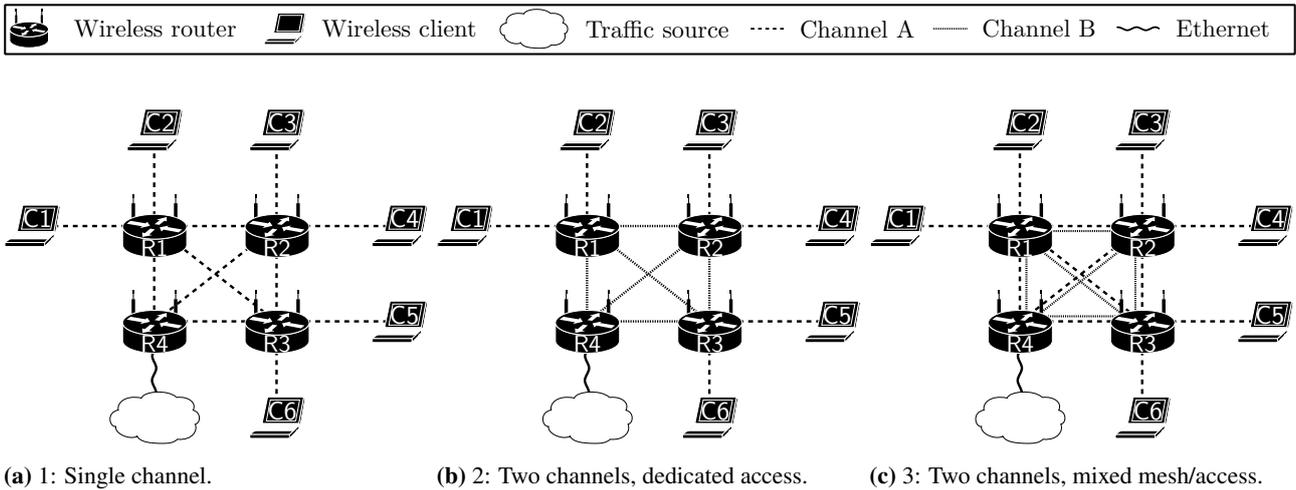


Figure 1 Visualization of the different evaluated setups.

clients and the controller has an Arch Linux operating system installed (last update: January 20, 2017). **Table 1** lists the versions of applications used during the experiments.

The standard behavior of each routing protocol is not changed. For OLSRv2 this means that the directional airtime metric and for Babel a form of expected transmission count (ETX) is used. One exception is the activation of the `iwinfo` plugin for BMX7 (to obtain link characteristics), since its usage is recommended in the kernel configuration. Lastly, all network daemons possibly transmitting data on all machines are disabled to avoid unsolicited data (e.g. a NTP client).

4.1.2 Physical setup

The tests were conducted in Bonn in an underground parking lot on January 29, 2017. Despite the fact that in the parking lot no other WiFi APs were detectable other devices may interfere during the measurements. To reduce self-interference (reflections), the wireless transmission power of all devices is reduced to 1mW. Channel 11 is used in the 2.4 GHz band (client access in all setups, and additionally mesh in Setup 3 – “Channel A”) and channel 44 is used in the 5 GHz band (“Channel B”); both in a 802.11n mode using 20 MHz channel width.

All nodes are statically placed in a grid with a distance of



Figure 2 Physical setup of the experiment.

1m between nodes. The bases of the antennas were fixed at a height of 50 cm – all upright. The ceiling is 2.9m high and the nearest wall to the first client (C1) is at a distance of 1.6m. The two TP-Link TL-WDR3600 devices are connected to Router 3: These devices are very similar to the TL-WDR4300 any influence on the results is not expected. The physical setup is shown in **Figure 2**.

4.1.3 Test procedure

Figure 3 provides an overview of the test procedure. The tests run automatically and are governed by the test controller. For each run, the controller activates the appropriate configuration on each device via Secure Shell (SSH). All protocols are tested once in the first setup, then in the second and then in the third setup. This process is then repeated for a defined number of times.

Each test starts with the initialization of the selected protocol. After the routing entries become stable, the connectivity is checked and local clocks are adjusted due to noticeably drift during the experiment. Statistics like memory and CPU consumption are collected with the help of a software called “`collectd`”. In addition, protocol specific information (e.g. neighbor information) and packet captures are gathered during the experiments. All collected data is saved locally on the `/tmp` RAM disk. For overhead calculations of each protocol, post-processing of the collected captures is necessary. The problem is that besides its own messages, every router also captures messages from other routers: Unfortunately, some messages originating from

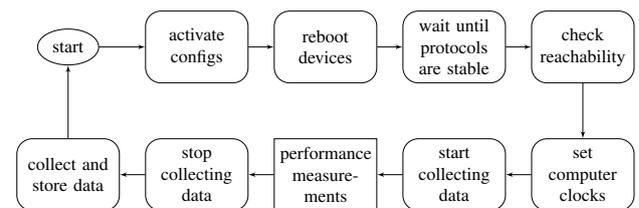


Figure 3 Fully automatic test sequence for the experiments. Repeatedly run for each combination of setup (1-3) and protocol (Babel, B.A.T.M.A.N. V., BMX7, OLSRv2).

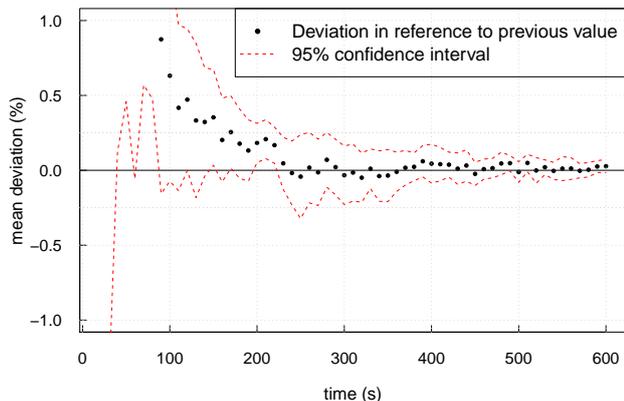


Figure 4 Example Results for minimum measurement duration for BMX7.

other routers get lost during the transmission, which leads to varying results on all routers. Therefore, the captures are filtered, so that each router’s capture only contains data that originated on that machine. These captures are useful to obtain the ratio of control traffic compared to the overall used airtime.

All clients start iperf in server mode. The controller starts an iperf client process instance for each client at nearly the same time. The default maximum segment size (MSS) (1448 bytes) of iperf is used. During the test, the controller also captures traffic using tcpdump. After the tests are finished, the controller immediately stops the data collection on all routers and stores the results.

And important aspect for the test produce is overall duration of each experiment. The duration should be long enough to obtain valid results, however, an unnecessary long duration leads to an unnecessary effort. Therefore, we conducted 10 preliminary measurements for each protocol and setup to obtain the duration needed by the routing protocols to stabilize and the time until the measured performance became constant.

Routing protocol stabilization is observed if either the overhead stabilizes on a constant level or a regular pattern emerges. This duration differs among the protocols and the experiments. For the majority this is already the case after 10s, however also a duration of 40s was obtained. Together with the static boot time of the routers (40s), 90s is chosen as the stabilization waiting time.

The measuring time is the duration needed to obtain stable throughput results. During our preliminary measurements, the network performs a 10m iperf test with intermediate values gathered every 10s. To obtain the minimum duration after the throughput becomes stable, a relative deviation in percent to the previous value is calculated for every intermediate value of every measurement⁴. For example,

$$rel_dev_{x_{10}-x_{20}} = \frac{x_{20} - x_{10}}{x_{10}}$$

describes the relative deviation for one preliminary measurement after 20s, where x_{10} is the overall throughput after 10s and x_{20} the overall throughput after 20s.

⁴The authors want to thank Prof. Dr. Marlis von der Hude (Bonn-Rhein-Sieg University) for her support on this aspect.

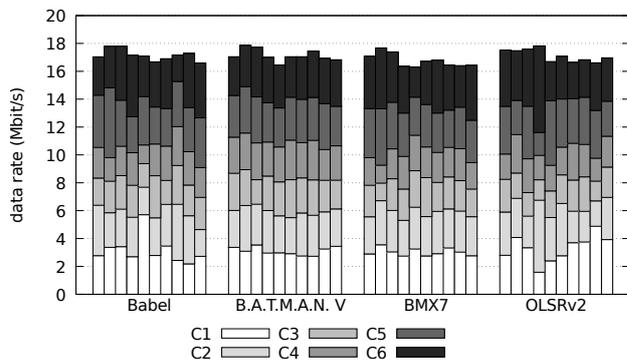


Figure 5 Setup 1: Overall iperf payload throughput of all measurements, including the distribution to clients C1-C6. (The bars for each protocol are ordered: The leftmost is the first measurement and the bar on the right is the tenth measurement.)

Among the 10 independent preliminary measurements, for all relative deviations for one time (e.g. $rel_dev_{x_{10}-x_{20}}$) a mean value and a 95% confidence interval using the t-distribution is calculated (individually for each protocol in each setup). An example of the results of such a calculation is shown in **Figure 4**.

A large confidence interval means that the measurements are strongly scattered at this point, therefore, the throughput is not yet stable. The mean values should remain around 0 or 0 should be within the confidence interval to indicate that the result will no longer change significantly. It has to be noted that the previous value has a direct influence on the successor value, since the throughput are always determined over the entire time up to the value.

The result with this method is similar to the previous observations, but it is noticeable that there are also fluctuations at later times which could be a result of the test environment. The throughput stabilizes between 200 and 250s for all protocols, so 300s is selected as the measuring time.

Combining all steps, one measurement for one protocol in one setup takes approximately 7m in our scenario. Again, we conducted 10 repetitions for each combination representing the final results in the next section. The tests went well, except for a minor glitch: During one experiment (Babel, Setup 2, round 3), the controller shortly lost the Ethernet link; therefore this result was marked as damaged. This particular experiment was then repeated after all the other experiments had been completed.

5 Results

5.1 Setup 1: Single Channel

Figure 5 shows the results for each protocol in Setup 1. It is evident that the achievable throughput is similar for all protocols. The median for all except BMX7 is around 17.5Mbit/s. Nevertheless, for all protocols the measurements are scattered, with up to 1.5Mbit/s between the smallest and highest measured value. We expected these scatterings since real-world measurement are prone

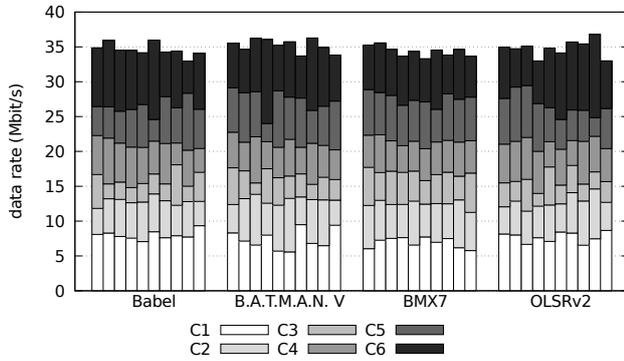


Figure 6 Setup 2: Overall iperf payload throughput of all measurements, including the distribution to clients C1-C6.

to different random factors (i.e. interference from external clients). An evaluation of route changes reveals that routes via other neighbors are established just for short time frames. In particular, OLSRv2 shows this behavior more often than other protocols. Nonetheless, the impact on the result is negligible, as these time frames compared to the overall measuring duration are small. Since routing changes are not noticeable for BMX7, the question arises why the resulting throughput is slightly lower than for its counterparts. This can be explained by the fact that the CPU utilization during the BMX7 tests is much higher. Especially on R4 (the router connected to the traffic source) software interrupts take up to 50% of the CPU time. A cause can be the encryption features of BMX7 to secure the protocol operation.

5.2 Setup 2: Dedicated Access

As shown in **Figure 6**, the throughput for Setup 2 has approximately doubled. This is expected since the channel capacity is doubled by using a second channel.

The B.A.T.M.A.N. V protocol on R2 chooses in 8 of 10 experiments a non-direct route to R4 despite the fact that this path is not optimal since additional airtime is consumed. Nevertheless, the overall achievable throughput is among the best since the TCP flow is directly routed from R4 to R2. R2 only forwards the TCP acknowledgments to R4 via the non-direct route.

5.3 Setup 3: Mixed Mesh/Access

The results for Setup 3 are shown in **Figure 7**. The protocols in comparison to each other are similar again, but outliers are observable. The negative outlier for B.A.T.M.A.N. V is caused due to the fact that R4 indirectly sends the iperf data to R3 (via R2). In general it is particularly interesting, if besides Channel B (the channel only routers are using), also Channel A is utilized. The data shows that both channels are used, but the usage of Channel B predominates. Once more it is evident, that B.A.T.M.A.N. V chooses indirect paths, but in all but one case only in the direction to R4: For transmitting the payloads from R4 always Channel B is used (except for this one case). In the majority of experiments using BMX7, the routers continuously chose Channel B after 50s. Lastly,

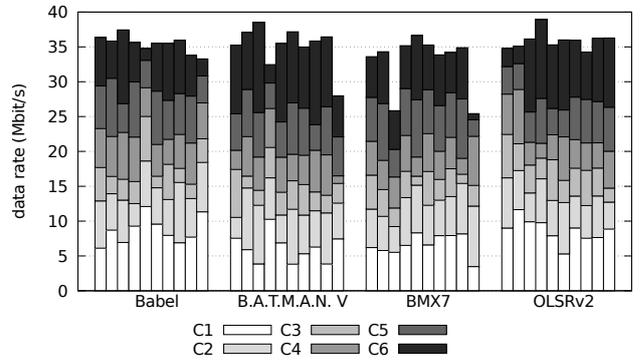


Figure 7 Setup 3: Overall iperf payload throughput of all measurements, including the distribution to clients C1-C6.

Babel and OLSRv2 show the same pattern: Channel B is used, but occasionally Channel A is utilized for a short period.

5.4 Protocol overhead

In order to compare the results for Setup 2 and Setup 3 it is necessary to take a look at the protocol overhead. It is evident that Babel generates the lowest and B.A.T.M.A.N. V the highest overhead in terms of overall throughput. A further observation is that the average packet size B.A.T.M.A.N. V transmits is smaller than for all other protocols, but packets are transmitted more often. OLSRv2 packets have the highest average size and are transmitted more often compared to Babel and BMX7. Moreover, the data in **Table 2** reveals that the overhead more or less doubles in Setup 3 compared to the other setups. This is expected since neighbor and routing information need to be exchanged for both radios.

Table 2 reveals the ratio of overhead to overall traffic. For all ten repetitions of each measurement the transmitted bytes of the protocol (including the protocol headers from

Protocol	iperf data (MByte)	Protocol data (MByte)	Ratio: protocol data to total data (%)
Setup 1			
Babel	6702	0.62	0.09
B.A.T.M.A.N. V	6700	18.19	2.71
BMX7	6567	4.32	0.66
OLSRv2	6700	12.85	1.91
Setup 2			
Babel	13559	0.48	0.04
B.A.T.M.A.N. V	13811	18.48	1.34
BMX7	13496	3.88	0.29
OLSRv2	13642	12.50	0.92
Setup 3			
Babel	13891	1.17	0.08
B.A.T.M.A.N. V	13768	51.35	3.72
BMX7	12943	8.16	0.63
OLSRv2	14083	20.57	1.46

Table 2 Protocol overhead compared to iperf traffic (including packet headers) for each protocol in each setup. The values are sums of all measuring rounds (10 * 300s).

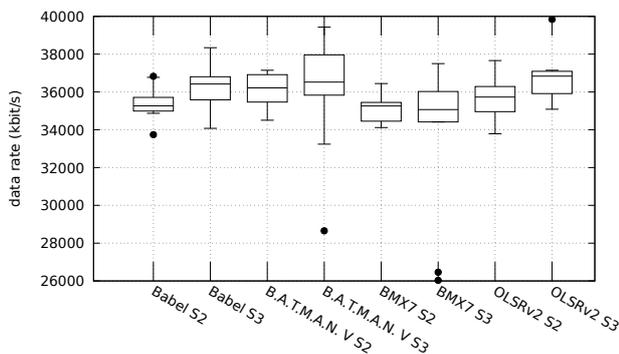


Figure 8 Comparison of the results from Setup 2 (dedicated access) and 3 (mixed mesh/access).

lower layers) are added; and also the transmitted iperf data from the packet captures of the initiator (including the protocol headers) are added. Afterwards, the ratio of the overhead to the overall traffic (protocol overhead + iperf traffic) is calculated. It is evident that the proportion of the overhead for all protocols is in the low per mille range. Therefore, in this scenario, the overhead of the WMN protocol has only a minor influence on the achievable throughput.

5.5 Dedicated Access vs. Mesh/Access

Finally, we are interested if Setup 2 or 3 is preferable in terms of achievable throughput. To simplify this comparison, the data from Figures 6 and 7 is combined in **Figure 8**. At first sight, Setup 3 seem preferable. Nevertheless, an evaluation using the 2-sample t-test (or the Welch 2-sample t-test) for each protocol in Setup 2 and Setup 3 shows that the mean values do not differ from one another. For OLSRV2, however, the result is close to a statistically detectable difference.

The protocols draw the right conclusions to a large extent with the presence of client traffic: choosing the less busy channel. Consequently, it is advantageous here to use both channels in the mesh network. A second channel provides redundancy, if the first channel is disturbed.

5.6 Additional Aspects and Future Work

It is apparent that the data rate to the clients connected to R2 is lower. Since this factor is constant, it is unlikely that this is related to the routing protocols. More plausible is an influence caused by to the positioning of R2 or by the device itself. The node is farthest away from R4 and has the most devices directly around it in the grid. Ideally, the node could have been interchanged to exclude hardware related effects.

An interesting additional setup 4 could take the (simple) single channel approach, but with the same (fairer) channel capacity as the two other scenarios (40 MHz).

An important question is the impact of the scenario on the results. In particular, the number of routers and clients and the distances between them. In case not all nodes see each other, hidden terminal or exposed terminal effects become relevant. The chosen environment corresponds to our mo-

tivated use-case of radio coverage inside a small rural village. To further investigate scaling effects, extending the experiments to different topologies (i.e. a chain) with an increased number of routers and clients seems desirable. The proposed methodology in this work is suitable for such an extension.

References

- [1] M. Abolhasan, B. Hagelstein, and J. C.-P. Wang. Real-world Performance of Current Proactive Multi-hop Mesh Protocols. In *Proceedings of the 15th Asia-Pacific Conference on Communications, APCC'09*, pages 44–47, Piscataway, NJ, USA, 2009. IEEE Press.
- [2] L. Cerdà-Alabern, A. Neumann, and L. Maccari. Experimental Evaluation of BMX6 Routing Metrics in a 802.11an Wireless-Community Mesh Network. In *2015 3rd International Conference on Future Internet of Things and Cloud*, pages 770–775, Aug 2015.
- [3] J. Chroboczek. The Babel Routing Protocol. RFC 6126 (Experimental), April 2011.
- [4] T. Clausen, C. Dearlove, and J. Dean. Mobile Ad Hoc Network (MANET) Neighborhood Discovery Protocol (NHDP). RFC 6130 (Proposed Standard), April 2011.
- [5] T. Clausen, C. Dearlove, P. Jacquet, and U. Herberg. The Optimized Link State Routing Protocol Version 2. RFC 7181 (Proposed Standard), April 2014.
- [6] Fraunhofer WiBACK. Fields of application. http://www.wiback.org/en/depolyment_senarios.html, 2016. [Online; last visit 2016-10-5].
- [7] Piyush Gupta and P R Kumar. The Capacity of Wireless Networks. *IEEE Transactions on Information Theory*, 46(2):388–404, March 2000.
- [8] D. Murray, M. Dixon, and T. Koziniec. An experimental comparison of routing protocols in multi hop ad hoc networks. In *2010 Australasian Telecommunication Networks and Applications Conference*, pages 159–164, October 2010.
- [9] A. Neumann. [BMXd] BMX7. <https://lists.bmx6.net/pipermail/bmxd/2016-March/000034.html>, 2016. [Online; last visit 2016-12-11].
- [10] A. Neumann, E. López, L. Cerda-Alabern, and L. Navarro. Securely-entrusted multi-topology routing for community networks. In *12th Annual Conference on Wireless On-demand Network Systems and Services*, pages 1–8, January 2016.
- [11] A. Neumann, E. López, and L. Navarro. Evaluation of mesh routing protocols for wireless community networks. *Computer Networks*, 93, Part 2:308 – 323, 2015.
- [12] Open Mesh. B.A.T.M.A.N. advanced. <https://www.open-mesh.org/projects/batman-adv/wiki/Wiki>, 2011. [Online; last visit 2016-11-8].
- [13] Open Mesh. B.A.T.M.A.N. V. https://www.open-mesh.org/projects/batman-adv/wiki/BATMAN_V, 2016. [Online; last visit 2016-11-8].
- [14] Open Mesh. Echo Location Protocol (ELP). <https://www.open-mesh.org/projects/batman-adv/wiki/ELP>, 2016. [Online; last visit 2016-11-8].