Electronic Communications of the EASST
Volume 17 (2009)

Workshops der Wissenschaftlichen Konferenz Kommunikation in Verteilten Systemen 2009 (WowKiVS 2009)

Experimental Evaluation of Multi-Path Routing in a Wireless Mesh Network Inside a Building

Thomas Staub, Stefan Ott and Torsten Braun

12 pages

Guest Editors: M. Wagner, D. Hogrefe, K. Geihs, K. David
Managing Editors: Tiziana Margaria, Julia Padberg, Gabriele Taentzer
ECEASST Home Page: http://www.easst.org/eceasst/ ISSN 1863-2122

Experimental Evaluation of Multi-Path Routing in a Wireless Mesh Network Inside a Building

Thomas Staub[1], **Stefan Ott**[1] **and Torsten Braun**[1]

[1]staub|ott|braun@iam.unibe.ch
Institute of Computer Science and Applied Mathematics, University of Bern
Neubrückstrasse 10, 3012 Bern, Switzerland

Abstract: Multi-path routing can provide robustness and load balancing to communication in wireless mesh networks (WMNs). We present a Linux implementation of an existing multi-path routing scheme and evaluate it in our inhouse WMN testbed. In addition to our implementation, we describe the testbed where we performed evaluations regarding end-to-end delay and packet loss. Furthermore, we identify the limitations of the implemented protocol in a real-world scenario with interferences due to dense node placement as well as third-party networks and discuss possible enhancements and general directions of research.

Keywords: wireless mesh networks, multi-path routing, experimentation, AODVM, testbed

1 Introduction

These days, wireless networks are a common element of everyday life. In most cases, however, these networks are infrastructure networks relying on centralized entities ("access points") for coordination and routing. In some cases, setting up such an infrastructure can be impractical, especially when the network has to be up and running as quickly as possible and/or when installing a wired backbone network is undesirable or impossible. This is where so-called wireless mesh networks (WMNs) become important. A WMN consists of independent nodes communicating with each other over wireless links and therefore without the presence of any wired infrastructure.

Such networks, while generally easier to deploy, have a high risk of link failure due to environmental changes, e.g., weather or other wireless networks. Moreover, nodes may leave the network, been shut down or change their position at random times. The nodes in a WMN have to rely on presence announcements from their neighbors. In times of heavy data traffic, however, this information can be lost due to interference between data and management traffic. This can cause nodes to falsely assume link failures and possibly re-run the route discovery process, unnecessarily disrupting network activity.

In order to add additional redundancy and/or capacity to such a network, several multi-path extensions to existing ad hoc routing protocols have been proposed. These extensions allow any member of the network to find multiple paths to any other member which can then be used either as fallback routes or to distribute the payload, thus allowing for better utilization of available bandwidth.

In this paper we are going to present our experience with one of those protocol extensions, explaining its operation, how we implemented it in Linux, how we tested it and which results those tests yielded.

Our contribution is the implementation and evaluation of a multi-path routing scheme in a Linux-based wireless mesh testbed deployed at our institute.

2 Related work

Research on wireless multi-path routing protocols produced several proposals for multi-path extensions to existing single-path ad-hoc routing protocols, namely to the Ad hoc On-Demand Distance Vector (AODV [PBD03]) and the Dynamic Source Routing (DSR [JHM07]) protocols.

Among the distance-vector routing protocols, proposed extensions to AODV include Ad hoc On-Demand Distance Vector Multipath (AODVM [YKT03]), the Ad hoc On-Demand Multipath Distance Vector (AOMDV [MD02]) and the Similar Node-disjoint Multipath Routing (SNDMR [XYX05]) protocols. They differ in the way route discovery works and, as a result, in the amount of "disjointness" between multiple paths, e.g. AODVM paths are node-disjoint while paths in AOMDV are link-disjoint, meaning that in AOMDV several paths can use common nodes which is not possible in AODVM (see Figure 1).

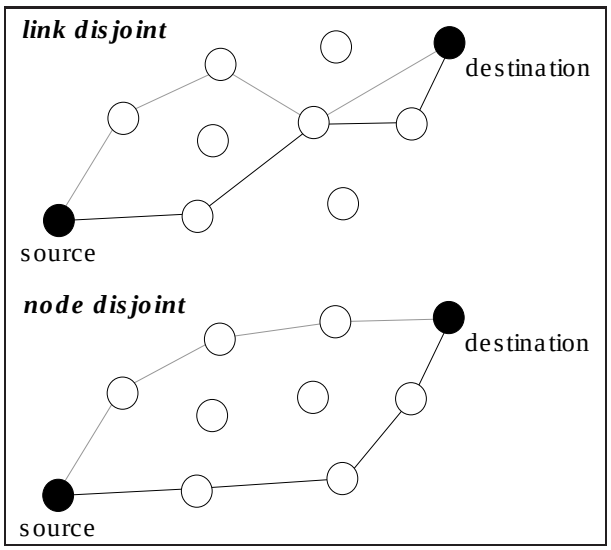

Figure 1: Link-disjoint and node-disjoint paths

However, source-routing also has its multi-path proposals, e.g. Split Multipath Routing (SMR [LG01]), based on DSR. SMR differs from the AODV derivates not only in it being a source-routing protocol but also in the fact that the number of paths is specified before the route discovery process and the protocol will try to make those paths as disjoint as possible ("maximally disjoint").

3 Multi-Path Routing Scheme: AODVM

The Ad-hoc On-demand Distance Vector Multipath (AODVM [YKT03]) routing protocol is a multi-path extension to the Ad-hoc On-demand Distance Vector (AODV [PBD03]) protocol. The main difference lies in the route discovery process (see Figure 2). As in AODV, the process starts with the source node broadcasting a route request (RREQ) packet (1) which is then forwarded by intermediate nodes (2). However, instead of dropping duplicate RREQs packets, intermediate nodes also forward them to the destination as well and store information on all forwarded RREQs in an RREQ table. For each forwarded RREQ, an entry in the so-called neighbor list is created (3). These entries contain information on the neighbor which sent the RREQ as well as the distance (hop count) to the RREQ source.

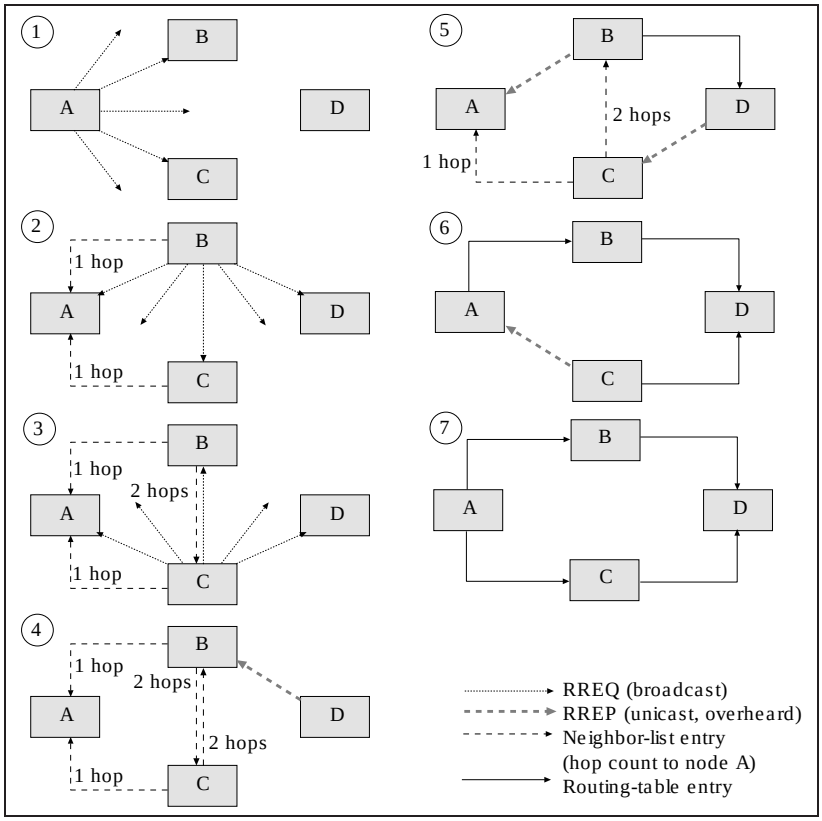

Figure 2: AODVM route discovery

In AODV, the destination only replies to the first RREQ it receives, any further requests are ignored (4). In AODVM, however, each RREQ is answered by sending a route reply (RREP) packet back to the RREQ's origin (5). The destination node then selects the shortest path to the source node in its neighbor list, forwards the RREP to that node and removes the corresponding RREQ entry from its RREQ table (6/7).

Since all nodes constantly overhear the wireless medium they know about all RREPs sent by their neighbors. If a node detects a RREP transmission, it removes the RREP's sender from its

neighbor list to ensure that no further route replies concerning the same route request are sent to that node. This mechanism guarantees that all routes discovered are node-disjoint, i.e. there are no routes to the same destination sharing any common nodes (see Figure 1).

The node which initiated the route discovery stores a route to the destination for each RREP received and the destination stores a reverse-route to the source for each RREQ received. The traffic can then be distributed over the discovered paths according to a packet allocation scheme.

4 Implementation

Our implementation of AODVM [YKT03] is based on AODV-UU [Nor], an AODV implementation which runs in the ns-2 network simulator [BEF+00] as well as on real Linux machines[Wib02].

4.1 Operation

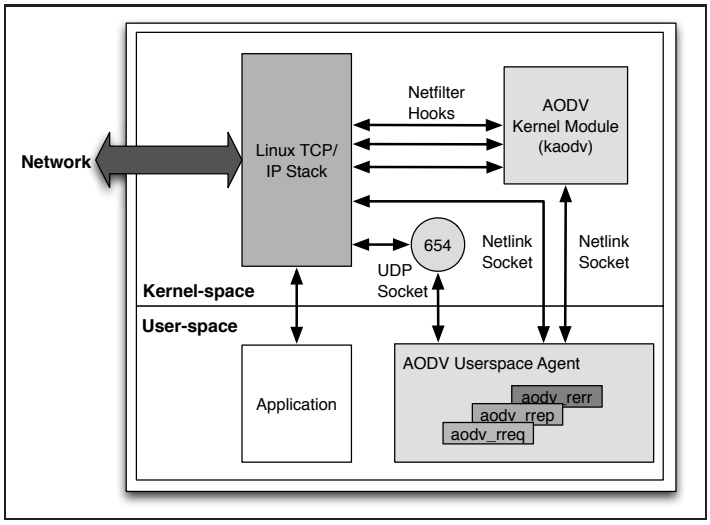

Figure 3: AODV-UU architecture

AODV-UU consists of two main parts: a kernel module (kaodv) and a user space agent (see Figure 3). The kernel part uses netfilter hooks to listen to incoming and outgoing traffic. It communicates with the user space agent over a netlink socket, e.g., to initiate a route discovery process when packets to unknown destination hosts show up or for route maintenance. The user space agent handles all the routing packets for route discovery and route maintenance. In order to implement AODVM, we had to modify both, the kernel module and the user space agent.

Since AODV-UU usually identifies a route by its destination, we had to add information about the next hop because in general in AODVM several routes with the same destination exist. We modified the netlink messages sent from the kernel module to contain the next hop's MAC address.

The user space application was modified to keep track of the next hop's MAC address in each route. Also, the agent uses a netlink socket to communicate with the kernel. We enhanced the

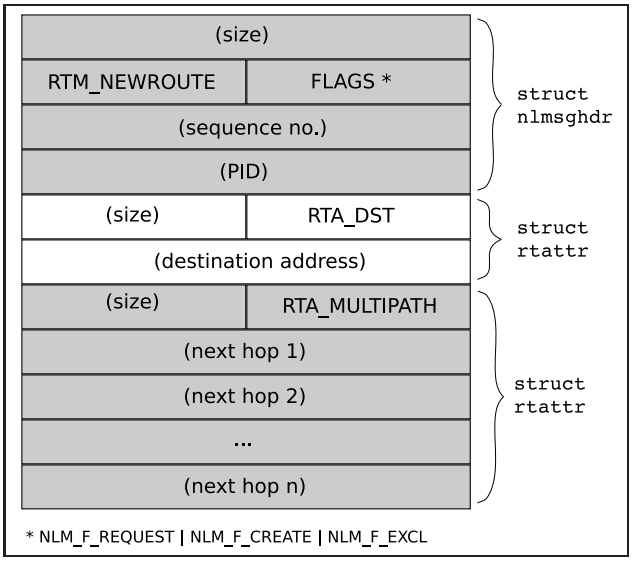

Figure 4: Netlink message to add a multi-path route. Below the RTA_MULTIPATH attribute, the different next hops are contained and form a multi-path route.

corresponding code to allow adding multi-path routes. In AODVM mode we send a message to the kernel, asking to add a new route (RTM_NEWROUTE) with a specific multi-path attribute (RTA_MULTIPATH) describing the possible next hops, that is, all the paths (see Figure 4). Since there is no easy way to remove a single path from a multi-path route, the whole route is deleted and then replaced by a new one without the no-longer valid path. Other changes we made to the user space application include:

- Add support for route discovery error (RDER) messages
- Modify the application to request an acknowledgement (RREP_ACK) for every RREP sent to reduce the effect of lost RREP packets
- Keep track of neighbor nodes (RREQ table, neighbor list)
- Modify the way intermediate nodes reply when they receive an RREQ
- Allow for multiple routes in the internal routing table
- Make sure the kernel loads the multipath_rr module
- Modify route management messages to support multipath routes

4.2 Route Selection

Multipath route selection in AODVM was achieved with the equal cost multi-path feature supported by recent Linux kernels, specifically with the multi-path round robin algorithm (see Figure 5). The kernel routing table holds several routes (next hops) per single destination. The

routing entries maintained by the multi-path routing protocol. A lookup in the kernel routing table only delivers one route per destination out of all the entries. The selection of the route is made according to scheduling algorithms (e.g., round robin). Therefore, the multi-path implementation is fully transparent for the applications. User-space communication software does not need to care about route selection.

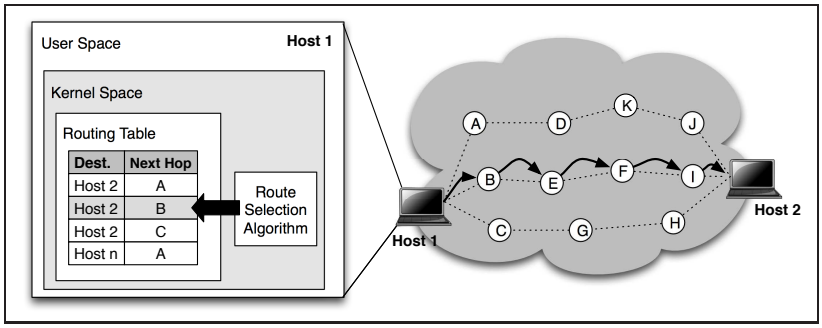

Figure 5: Kernel route selection

4.3 Challenges

During our implementation, we were confronted with several challenges, such as:

- Linux' netlink interface is not particularly well documented. Since, however, we needed to use netlink for managing our routes, we ended up stripping the code of *iproute2* [KH] down to the bare minimum required for our route management and importing the parts of the code which were not already covered by AODV-UU into our code.
- Kernel APIs kept changing (and are still in constant flux) preventing us from using up-to-date kernel versions without constantly adjusting our code. This was especially unfortunate because wireless drivers for older kernels had stability and reliability issues.
- During the course of our work, several new versions of AODV-UU were released which meant additional porting efforts to ensure compatibility of our code with the new versions.

5 Testbed Setup

The nodes in our testbed were connected to a wired management network. We used two notebook computers and thirteen intermediate wireless mesh nodes. The notebooks served as source and destination. They were both run Linux with 2.6 kernels. For wireless connectivity, they were equipped with Cisco Aironet PCMCIA cards (Atheros chip, madwifi driver). For the sake of simplicity, both notebook were connected by Ethernet to the management network. For the intermediate mesh nodes we used the Wireless Router Application Platform (WRAP[Dor]), loaded with a custom-made Linux distribution [SBLB07] based on Kernel 2.6.22.2. Like the notebooks,

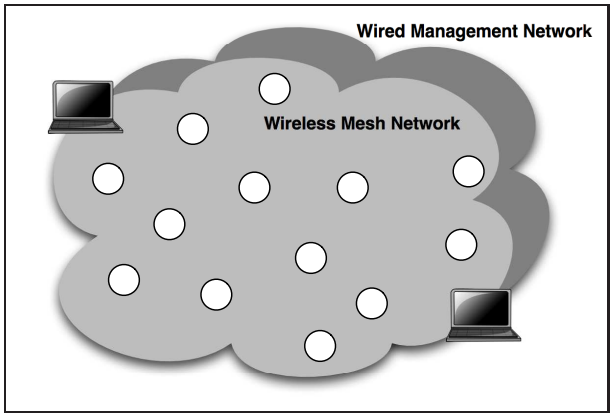

Figure 6: Setup of our testbed with Wireless Mesh Network and management network.

the mesh nodes were connected to the management network via Ethernet, the ad-hoc network interface was running on an Atheros-based mini-PCI cards with the madwifi driver (0.9.3.3).

The nodes were distributed in two buildings of our institute to create a scenario where most of the intermediate nodes had a direct link, the endpoints, however, did not. This setup was chosen in order to have a realistic scenario where interference and weak links pose additional challenges to the routing protocol.

The measurements were coordinated via the management network. A script would initialize all nodes with the correct routing protocol, start a video stream (see Section 5.1), shut down some nodes at the appropriate time (in scenarios with forced node failure), terminate the experiment and collect and store the log files afterwards.

5.1 Traffic and Measurement Parameters

The network load was created by streaming a high-resolution video file. We used the movie "Elephants Dream" [1] and converted it to an MPEG-1 video (1024x576, 24fps, MPEG1 200kbps) using *ffmpeg*. The movie was streamed via UDP from the source notebook to the destination notebook for five minutes, streaming was done using the VLC media player, the movie was stored on the source notebook.

The parameters measured were end-to-end delay and packet loss. For the delay, an ICMP echo request ("ping") packet was sent from the source to the destination. To account for lost packets, both the source and the destination notebook were running tcpdump during the experiment. The generated log files were analyzed with *tshark*[Com] and filtered using Unix shell programs.

5.2 Failure Scenarios

Each protocol was evaluated in two scenarios: First, the experiment was run for five minutes without node failure. Then, in order to see how well the protocols recover from lost routes, the same scenario was repeated with a forced node failure at the next hop in the route from the source

[1] http://orange.blender.org/download

notebook to the destination after two and a half minutes. In multi-path scenarios, only one of the routes was disabled this way. We repeated our real-world experiments 20 times in order to reduce the effects of temporary interferences.

6 Performance Evaluation

6.1 Packet Loss

Figure 7 shows the average packet loss with each protocol and scenario over 20 runs in the 2.4 GHz ISM band. This was calculated by comparing the number of packets sent with the number of packets which actually arrived at the receiver.

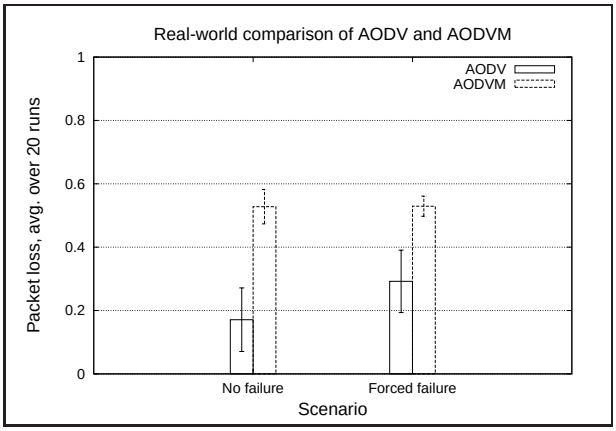

Figure 7: Average packet loss in 2.4 GHz ISM band

Strong interferences with several neighboring wireless networks around our institute caused some base packet loss in all the experiments inside the 2.4 GHz ISM band (IEEE 802.11b). In general, packet loss with AODVM was higher than with AODV, meaning that the multi-path extension actually performed worse than the single-path variant. While these results might come as a surprise initially they do make sense in our case for the following reasons.

Our issue with AODVM was that our testbed was pretty small. Thus most nodes had a direct link to most other nodes, meaning not only that communication would be possible between them, but also that traffic sent over one path would interfere with traffic sent over another path. In AODV, only one such path is used, thus traffic can flow without any major inter-flow interferences. In AODVM however, traffic is spread over multiple paths. This now leads to a high risk of collisions on the wireless medium meaning that instead of a single sent packet receiving its destination, two sent packets collide and are both lost.

This became obvious after the first attempts at running real-world experiments: AODV periodically sends out HELLO packets (RREP packets with a TTL of 1) to inform a node's neighbors about the node's presence. Initially, one such packet was sent every second and if a node missed two consecutive packets from a neighbor it assumed that the neighbor was no longer there. This worked fine as long as the network was idle. As soon as we started to generate traffic, however, we could literally see the network break down. Many HELLO packets never reached their

destination, supposedly due to collisions with data packets. Our first solution was to double the amount of HELLO packets. This did not work particularly well (the network was still very unstable) so we drastically increased the HELLO timeout (to 20 seconds) which finally resulted in a stable network. As there is no mobility in our network the increased HELLO timeout does not limit the protocol's performance.

Still, we do not consider this an ideal solution since other management traffic such as RREP / RREQ messages might also collide with data traffic. The problem could be addressed by multi-channel protocols where not all links use the same wireless channel for communicating.

Also, when we switched our testbed to IEEE 802.11a, the lower range of wireless signals at 5 GHz meant that more hops per path were required and this in turn made us realize an even deeper problem: due to the nature of AODV's route discovery process, routes with very bad connectivity were chosen over much better routes with more hops (see Figure 8). In addition, the communication gray zones [LNT02] significantly affected the measurements in the 5 GHz setup. Some nodes were able to successfully exchange HELLO messages between each other, but no data messages could be exchanged. This meant that we were hardly ever able to communicate through the network with the other end point because we would always have at least one extremely poor link in between. In order to make the network operating, we eliminated the completely dead links (with no possible data transmission) by MAC filter. But still when doing this, we observed similar packet losses than shown in Figure 7 for the 2.4 GHz ISM band due to selection of bad paths with low hop count. The integration of a link quality metric such as ETX [DABM03] in the route-selection process taking the link quality into account would certainly improve the results. Moreover, the proposed solutions to the communication gray zone problem of AODV presented in [LNT02] have to be implemented for future measurements. The disadvantageous managing of unstable links has been also observed in [BD07].

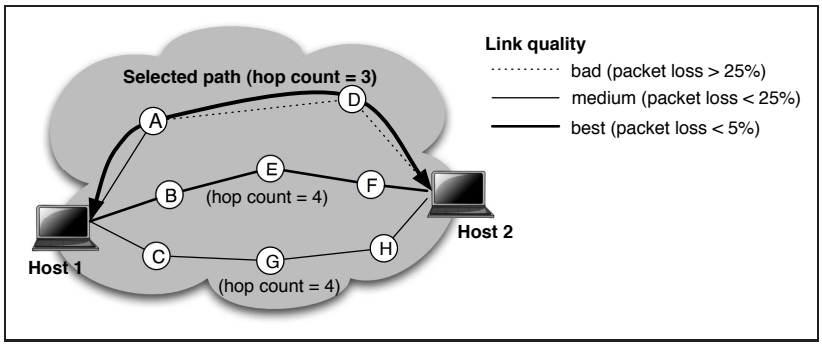

Figure 8: Selecting bad paths over good ones

In order to reduce the problem of interfering multiple paths, a modified route discovery process which tries to find interference-disjoint paths is required. While several such paths might not always exist, taking interference into consideration during route discovery could also be done on a best-effort base, that is the discovery process could try to avoid interfering paths if possible, yet not completely forbid them.

6.2 End-To-End Delay

In order to calculate the end-to-end delay, 20 ICMP echo requests were sent from the source to the destination at the start of the experiment in the setup with transmission in the 5 GHz band (IEEE 802.11a).

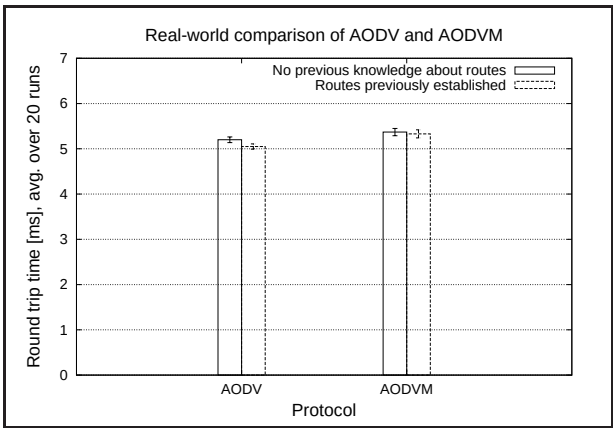

Figure 9: Average round trip time using ICMP echo in 5 GHz

Figure 9 compares the end-to-end delays for the different protocols. We ran the end-to-end delay measurement twice in each scenario, in order to eliminate the initial delay due to route discovery in the second run.

As the Figure 9 shows, we observed good round trip times. The bad links do not influence the results for the small ICMP echo packets (64 bytes). In general, the results were slightly better when the route-discovery process was done before the actual round-trip-time measurement.

In general, as we have expected, there is no significant difference between the end-to-end delays of AODV and AODVM. Nevertheless, the end-to-end delays in AODVM are usually a little higher. Other than AODV, AODVM does not only use the best path but also other, worse alternatives (see Figure 10). As traffic is now sent over all routes using a simple round-robin scheme, there are packets using the fastest path and other packets using slower ones, thus producing higher average values. Note that these routes typically went via four (or more) intermediate nodes.

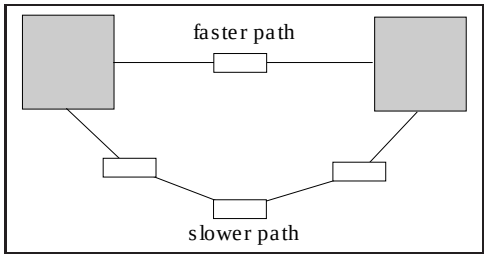

Figure 10: AODVM may create additional, slower paths

7 Conclusion

Our results confirmed the disadvantageous route selection in AODV and AODVM. By simple selection of the shortest paths (hop count), the protocols integrate links with high packet losses or even completely dead links due to the communication gray zone problems. They further show that "compact" network topologies yield increased interferences when using multi-path routing, up to a point where the bandwidth benefits from the additional paths are more than annihilated by the additional interferences. If tested in a bigger testbed, AODVM might create more interference-disjoint paths than in our case.

In order to fully profit from the strengths (redundancy, load balancing) of multi-path routing protocols in compact networks, we propose combining multi-path routing protocols with multi-channel communication. For this, improved metrics such as MIC [YWK05] + iAWARE ([SBM06]) are necessary. This will be addressed in further research.

Moreover, efficient usage of the frequencies in one of the license-free industrial, scientific and medical (ISM) bands is rather complex and remains a yet unsolved issue. Today, for example, the 2.400-2.500 GHz range is crowded with lots of different communication networks. They are all causing interferences and therefore affect the overall communication performance. We have observed this even at our institute during the measurements as there are several networks operated by other groups which caused some base packet loss in our experiment. Therefore, investigations on dynamic channel allocation schemes that take their environment into account are required. They should try to employ the temporarily unused "white spaces" in the available frequency band to reduce the interferences to a minimum. This would lead to a more efficient and robust communication infrastructure.

Acknowledgements: The work presented in this paper was supported by the Swiss National Science Foundation under grant number 200020-113677/1.

Bibliography

[BD07] E. Borgia, F. Delmastro. Effects of unstable links on AODV performance in real testbeds. *EURASIP J. Wirel. Commun. Netw.* 2007(1):32–32, 2007.

[BEF+00] L. Breslau, D. Estrin, K. Fall, S. Floyd, J. Heidemann, A. Helmy, P. Huang, S. McCanne, K. Varadhan, Y. Xu, H. Yu. Advances in network simulation. *IEEE Communications Magazine* 33(5):59–67, May 2000.

[Com] G. Combs. Wireshark. http://www.wireshark.org.

[DABM03] D. S. J. De Couto, D. Aguayo, J. Bicket, R. Morris. A High-Throughput Path Metric for Multi-Hop Wireless Routing. In *Proceedings of the 9th ACM International Conference on Mobile Computing and Networking (MobiCom '03)*. San Diego, California, September 2003.

[Dor] P. Dornier. PC Engines Wireless Router Application Platform (WRAP). http://www.pcengines.ch.

[JHM07] D. B. Johnson, Y. C. Hu, D. A. Maltz. The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks. IETF RFC 4728, February 2007.

[KH] A. Kuznetsov, S. Hemminger. Net:Iproute2. http://www.linuxfoundation.org/en/Net:Iproute2.

[LG01] S.-J. Lee, M. Gerla. Split Multipath Routing with Maximally Disjoint Paths in Ad Hoc Networks. In *IEEE International Conference on Communications (ICC)*. Volume 10, pp. 3201–3205. Helsinki, Finlandia, June 11-14 2001.

[LNT02] H. Lundgren, E. Nordström, C. Tschudin. Coping with communication gray zones in IEEE 802.11b based ad hoc networks. In *WOWMOM '02: Proceedings of the 5th ACM international workshop on Wireless mobile multimedia*. Pp. 49–55. ACM, New York, NY, USA, 2002.

[MD02] M. K. Marina, S. R. Das. Ad hoc On-demand Multipath Distance Vector Routing. *ACM SIGMOBILE Mobile Computing and Communications Review* 6(3):92–93, July 2002.

[Nor] E. Nordström. AODV-UU. http://core.it.uu.se/core/index.php/AODV-UU.

[PBD03] C. Perkins, E. Belding-Royer, S. Das. Ad hoc On-demand Distance Vector (AODV) Routing. IETF RFC 3561, July 2003.

[SBLB07] T. Staub, D. Balsiger, M. Lustenberger, T. Braun. Secure Remote Management and Software Distribution for Wireless Mesh Networks. In *7th International Workshop on Applications and Services in Wireless Networks (ASWN 2007)*. Pp. 47–54. Santander, Spain, May 24-26 2007.

[SBM06] A. P. Subramanian, M. M. Buddhikot, S. Miller. Interference aware routing in multi-radio wireless mesh networks. In *2nd IEEE Workshop on Wireless Mesh Networks (WiMesh 2006)*. Pp. 55–63. Reston, Virginia, USA, September 25 2006.

[Wib02] B. Wiberg. Porting AODV-UU Implementation to ns-2 and Enabling Trace-based Simulation. Master's thesis, Uppsala University, Department of Information Technology, Communication Research, Uppsala, Sweden, December 18 2002.

[XYX05] W. Xu, P. Yan, D. Xia. Similar node-disjoint multi-paths routing in wireless ad hoc networks. In *International Conference on Wireless Communications, Networking and Mobile Computing (WiMob'05)*. Volume 2, pp. 731–734. Montreal, Quebec, Canada, August 22-24 2005.

[YKT03] Z. Ye, S. V. Krishnamurthy, S. K. Tripathi. A Framework for Reliable Routing in Mobile Ad Hoc Networks. In *IEEE Infocom 2003 (INFOCOM)*. San Francisco, CA, USA, May 30 - April 3 2003.

[YWK05] Y. Yang, J. Wang, R. Kravets. Designing Routing Metrics for Mesh Networks. In *First IEEE Workshop on Wireless Mesh Networks (WiMesh)*. Santa Clara, CA, USA, September 26 2005.