MANET Testbed with Raspberry PIs

Christoph Claßen
Student number: 2226523 Email: Christoph.Classen@uni-duesseldorf.de
University of Düsseldorf, HHU - 2015 SS - Opportunistic and Peer-to-Peer Networks

***Abstract*—Mobile ad-hoc networks (MANETs) are networks without any central infrastructure and consists only of equal peers which offers and consumes services. In contrast to ordinary peer to peer (P2P) networks the nodes in MANETs doesn't have any static physical position and network connection. Each node can move around and destroy or create links to other nodes automatically. After a stabilisation period, each node is connected the all other nodes in its range.**

This paper presents a testbed for MANET routing protocols. The special approach is the physical testbed. In contrast to many other publications which examines MANET routing protocols, no simulated environment is used. Real live conditions which can influences tests can be examined. For example persons who walk through the wireless link can cause an interruption. The testbed consists of several Raspberry Pis (RPis) without the need of any simulator. All network nodes are physical devices. For making the RPis mobile and independent of any fixed power sockets, each of them is powered with a power bank. On the other hand, for the evaluation of the testbed, two routing protocols are choosen. The first protocol is called BABEL, which uses a distance vector routing algorithm. The other one, the Optimized Link State Routing Protokol (OLSR) uses a link state routing algorithm. Both protocols represents one of the two main algorithm groups for routing in MANETs. These two routing protocols are compared to each other. The examined use case is a multi hop download of some files with different size. The aim is to evaluate how multiple hops with different routing protocols influences the possible bandwidth.

I. Introduction

Wireless communication is today an essential way to connect computer systems with each other. Most of them use stationary infrastructure such as wireless access points or radio masts. This infrastructure is expensive to build and maintain. Especially in sparsely populated regions with a low amount of users. Another application may be to connect taxi drivers with each other or enable communication of the armed forces for example planes and tanks on a battlefield. Furthermore the stationary infrastructure is vulnerable by earth quakes or tsunamis. In such disaster zones, a way to communicate without the desire of stationary infrastructure or central administration. This challenge may be accepted by mobile ad-hoc networks (MANETs). Ad-hoc networks consists of a collection of nodes and works without any central infrastructure. Each node can communicate directly with each other node in its range. MANETs organize dynamically themselves. The routing functionality in MANETs is non-hierarchical. To reach nodes out of its range, a node use multi hop paths to reach the destination. The nodes on the path works as routers and relays the messages. The network topology can change over time if nodes goes up or down or moves their position. MANETs can be combined with peer-to-peer (P2P) overlays to create mobile P2P applications. In many MANETs the traffic is rather P2P like than server to client. The structure of MANETs is located between the wired, fast and reliable communication like Ethernet on the one side and very loosely connected opportunistic networks (OppNets). In opposite to MANETs Oppnets offers only an asynchronous connection between nodes because at an fixed time, it is not sure that all nodes in the network can be reached. The communication depends on the moving of the nodes.

II. Related work

There are a lot of researchers who are working on MANETs. There are two main approaches: first, to use simulators like [1] to retrieve information about the network behavior.

The second approach is using real testbeds. But mostly only one protocol is examined like in [2]. They uses two different test situations. First, a vertical one, which is placed in the stairways. Here is one laptop located at each floor. The second and horizontal environment is located in an corridor. In both cases five laptops creates an ad-hoc network. They shows that the traffic decreases if the source node moves along the other nodes away from the destination.

In [3] a real live testbed is shown. The authors uses 5 laptops to create an ad-hoc network in an office environment. They compares 8 different configurations of positions and movements of the laptops. The maximum hop count is 4. The used routing protocol is OLSR. In maximum three of five laptops are moving while testing the throughput with UDP and TCP. Using TCP the throughput drops about 50 percent after the third hop. Similar results are made in section VI. In this paper a linear topology of a MANET is used to figure out the maximum possible hop count for downloading files from the internet. Two routing protocols, babel and OLSR, will be compared with each other.

In contrast to other papers [2] [3], Raspberry Pis are used as network nodes.

III. Testbed

The first minimum testbed starts with 3 Raspberry Pis (RPis). All RPis are the Raspberry Pi Model B+ Version 1.2. All RPis runs Weezy Raspbian Linux in the version 3.18.7+ #755 armv61 GNU/Linux. As network card a BIG687 WLAN 802.11n USB Adapter in Nano Size with a Ralink RT5370 chip set is used. This adapter supports in the 802.11b/g net a peak throughput up to 27Mbps.

Each device has a static IP address in the local ad-hoc net 192.168.178.0. With a subnet mask 255.255.255.0 our network supports 254 devices. All RPis uses their interface wlan0 to create an wireless ad-hoc network. RPi 1 has an special role, because it is configured as an internet gateway. To connect the device physically to the internet, the LAN connection eth0 is used. In order to work as a gateway, the routing table have to be extended manually. The fist step ist to allow forwarding between interfaces. After that all traffic wich comes out of the local MANET over wlan0 will be forwarded to eth0. But in the opposite direction only packages which belongs to an established or related connection will be forwarded into the MANET.

- iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
- iptables -A FORWARD -i eth0 -o wlan0 -m conntrack –ctstate RELATED,ESTABLISHED -j ACCEPT
- iptables -A FORWARD -i wlan0 -o eth0 -j ACCEPT

To enable DNS so that it is possible to route not only via an IP address, the line

- nameserver 192.168.2.1

into the file /etc/resolv.conf. The ip which identifies the name server depends on the local network.

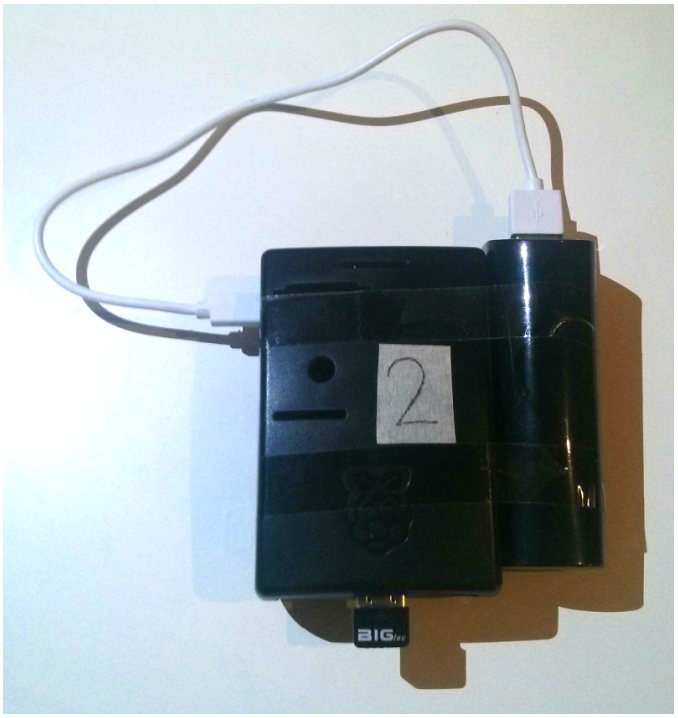

Fig. 1. Raspberry Pi powered by an accupack on the right.

The first connection tests runs with static positions but these are not flexible enough. So power banks are added to the RPis to create completely mobile devices. Each accu pack has an charge of 2200 mAh and powers a RPi for about 4 hours. Now we are able to place each RPi at nearly every position. It is important to say that no limitations like MAC-filter or other techniques to create an line configuration. Only the limited range of each wireless interfaces is used to control, which RPi can talk to each other. A schematic visualization is presented in fig 2 The RPi 1 which is connected to the internet over ethernet is shown right on the top.

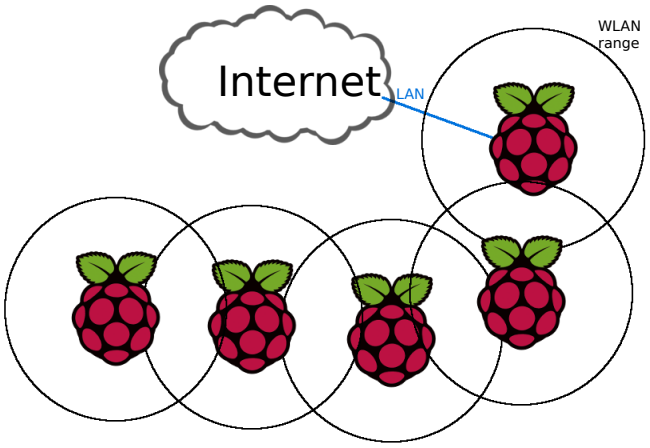

Fig. 2. Raspberry Pis loosely connected via wireless lan.

The first test environment is the stairways to my apartment. The first RPi is connected via ethernet to the internet and shares its internet connection to the ad-hoc network. It is located in my apartment. The second RPi is in front of my apartment a half floor level downstairs, the third two floor levels below the second RPi.

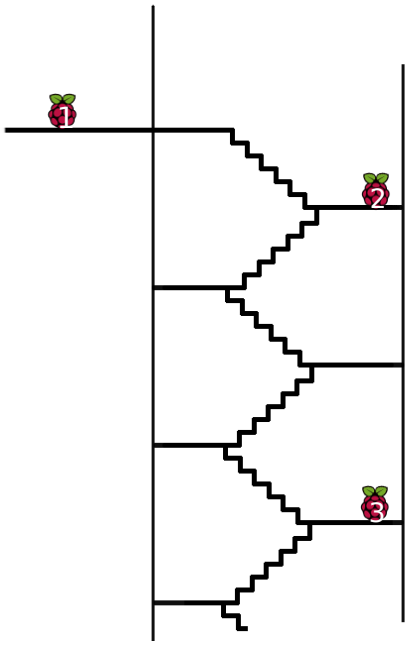

Fig. 3. Raspberry Pi powered by an accu pack on the right.

A bigger test take place at the university. First in the corridors of the buildings. The first RPi is placed in an office and with wired internet connection. Three other nodes are

placed in the corridors with visual contact to their direct neighbors.

The third and main test environment is on the campus between complexes of the buildings 25 and 26. On the balcony of the first floor the first RPi is placed. The next RPi is placed on a railings outside of the building on the ground floor. Each RPi's position is marked in figure 4. RPis 2-7 are placed on the ground floor.

Each test run starts with a 1-hop connection between RPi 1 and the next RPi. At first, the download speed with 1 hop is tested with several files. The file sizes are 10kb, 100kb, 500kb, 1mb, 10mb, 20mb and 1gb. The file content consists of random binary data. To control the RPi it has a second wireless interface over which it can be accessed via ssh.

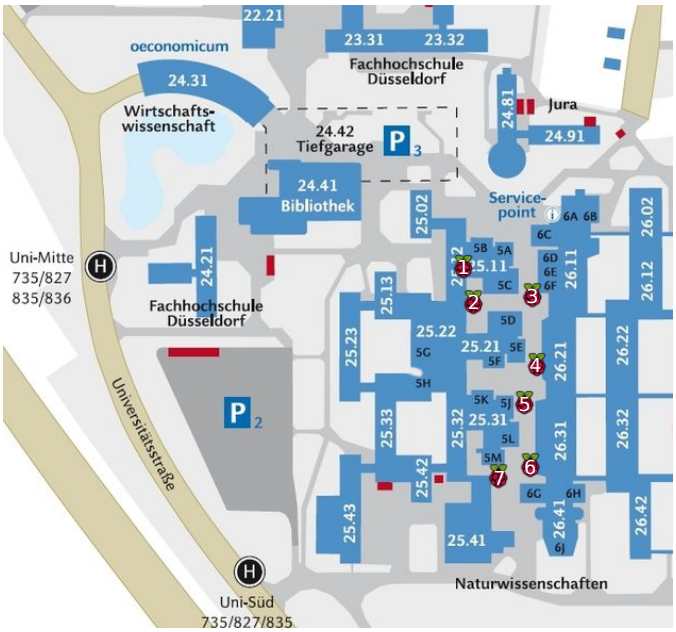

Fig. 4. area map. The red number denotes the positions of the devices

IV. Routing protocols

The two routing protocols called Babel and OLSR represents each one of the two main algorithms for proactive routing in MANETs. Both works without the need of an central instance like a server. The offered protocol Better Approach To Mobile Adhoc Networking with the abbreviation B.A.T.M.A.N. is not mentioned because it was not possible to install and configure it properly on the Raspberry Pis.

A. Babel

Babel is a routing protocol for mesh networks. It was developed at the university Paris-Diderot. Even though Babel is optimized for wireless networks, it works with wired and wireless networks Furthermore IPv4 and IPv6 are supported.

If the network topology changes, babel reorganizes itself. Routing loops may occur while reorganizing but they will be remove when babel convergences to an optimal configuration. This convergence works slowly. It depends on the network size, but the used time can be measured in minutes.

Babel uses a bunch of message types to maintain the network. The HELLO message is very important message type. It is used to discover neighbors and determining link costs. HELLO messages are multicasts. Nodes which receive this message, answers with the IHU message. The IHU will be send as an unicast message to the sender of the HELLO message. It is called the "I hear you" message and shows that an link can be used in an bidirectional way. The link cost is transmitted to the HELLO sender with the IHU message.

Each babel node sends the HELLO message periodically to all of its neighbors even the network topology doesn't change. Every node answers every HELLO message with an IHU message. This causes an overhead of protocol messages.

An interesting feature of babel is, that it is able to connect devices with distinct configurations for example the HELLO interval. This would be on a mobile battery powered device much longer to save battery power than on an device which is connected to a power socket. A device which is moving all the time would use a shorter interval stay connected to the network. If the count of correct received HELLO messages decreases below a threshold, the link will be marked with infinite costs.

Depending on the configuration babel can use different information to calculate the routing tree for each node. The cost of a route is the sum of all hop between source and destination.This is called metric. The metric can be nearly every information. But it has to satisfy two conditions. The first condition is that if a part of a route has infinite costs, then the metric of the route must be infinite. Second, no hop can have no costs. The metric of the route have to raise strictly monotonic with each hop. Although the Bellman-Ford algorithm can handle negative costs, babel uses only positive costs.

To forward user content over multiple hops, babel can select the route depending on several information about the hops. For example routes with small metrics will be preferred. Links to stable neighbors will be preferred, too. The neighbors are not mentioned as devices. If a device has two or more network interfaces it can support multiple links to another node. One link per interface. With a view to the decentral structure of a babel net, the routing tree for each node is calculated on it self. The Bellman-Ford algorithm is used for the route calculation.

1) Bellman-Ford algorithm: The Bellman-Ford algorithm is an algorithm to calculate the costs from one node to all other nodes. In contrast to the Dijkstra algorithm, the Bellman-Ford algorithm can handle negative edge costs. Edges are unidirectional. Bidirectional edges consists of two links with opposite direction. Costs of a node are the sum of the edges to reach this node from the source node. The source node starts with cost 0. Because to reach itself, no edge have to be used. All other nodes have infinite costs. Now the algorithm runs several phases to determine new and possible better costs. If the graph has n edges, then the algorithm use n-1 phases. A path, which use more edges would visit one node more than

one time and create a circle. In each phase, the algorithm checks all edges and calculates the cost of the corresponding end node. Therefore the sum of cost of the start node and the edge is calculated. If the new value is lower than the old one of the end node, the old value will be replaced. Now the predecessor edge is stored to the end node.In phase i the algorithm has detected the shortest path with i hops. At the end, after n-1 phases, the algorithm checks whether there are loops with negative costs. This is done by comparing the sum from cost of the start node and edge with the cost of the end node. If the end node has higher costs, a negative circle is found. A negative circle causes that no minimal route can be found. The algorithm has an complexity of $\mathcal{O}(n * m)$ with n nodes and m edges. It has to be executed on each node which causes an complexity of $\mathcal{O}(n^2 * m)$

2) Routing table updates: After a Babel node has computed its routing tree, it advertises its routing table to its neighbors. They now compare their own with the new table and updates the local table if necessary. Sending periodically the whole routing table causes an extreme communication overhead especially in stable networks.

B. OLSR

The Optimized Link State Routing Protocol (OLSR) is used like Babel to allow routing in MANETs. OLRS is designed for wireless connections and uses hob-by-hop routing. OLSR is a proactive routing protocol which means that possible routes are estimated before a package has to be transmitted. In opposite to reactive protocols the user needn't wait for a route calculation. But proactive routing causes an overhead because all routes are calculated, even those which would never be used.

A node X uses HELLO messages to detect periodically its 1-hop and 2-hop neighbors. The answer form Y corresponding to the HELLO message contains a list with 1-hop neighbors of Y. Node Y is a neighbor node of X if there exists a direct symmetric link between X and Y. To reduce overhead produced by broadcast messages, there are nodes called multi point relays (MPRs) which are the only nodes which forwards broadcasts of control messages. Each node choses its own set of MPRs out of its 1-hop neighbors. The MPRs are selected to reach all 2-hop neighbors with a minimum of traffic. The other neighbor nodes receive and process the broadcast messages, but they don't retransmit them. In this manner every broadcast message is send to all node with a heavily reduced overhead. The set of MPRs can change with the re-transformation of the network if some devices are changing its positions. Each MPR maintain a list of nodes which uses this node as an MPR. Additionally the MPR sends topology-control (TC) messages to distribute information of possible links. Based upon these TC messages, each node creates or updates its routing table.

Functions of OLSR can be divided into a core and an auxiliary part. The core is mandatory for the working of OLSR. The auxiliary functions supports the core or offers additional services. The core aggregates functions for packet format and forwarding, neighbor detection, MPR selection, topology control message diffusion and route calculation.

The auxilary functions offers for example the connection to non-OLSR interfaces. A non-OLSR interface is an network interface of a node which doesn't participate in OLSR but offers for example an internet connection. To create a connection to a local network or the internet, OLSR uses host and network association (HNA) messages. These announces than a node has the connection to a given network. IPv6 is supported by OLSR, too. Therefore the addresses have to be changed to IPv6 an the message size have to be increased.

OLSR is like B.A.T.M.A.N. used in the Freifunk network. Since 2015 OLSR will not be developed furthermore. It will be replaced by OLSRv2. Until now OLSRv2 is not relay published, also in this paper OLSR is used.

V. Testbed software

The both routing protocols mentioned in section IV-A and IV-B has each different implementations.

A. Babel configuration

On the RPis babel is installed via the package manager. This implementation is used because it offers a stable and maintained version. Before starting the daemon of the gateway RPi, its configuration file has to be edited. The RPi should announce that it has internet connection. To do that, the line "redistribute ip 0.0.0.0/0 le 0 metric 128" has to be added to the configuration file. Now babel can be started with the command "babeld wlan0 -d 1". wlan0 is the interface, on which babel should run. With the option -d 1 the debug output is enabled and shows the known neighbors.

B. OLSR configuration

OLSR is also installed via the package manager for the same reasons. The configuration file of the ethernet connected node has to be edited. The HNA must be activated. To announce an internet gateway, the line "0.0.0.0 0.0.0.0" has to be added into the HNA4 block. The first four zero describes the network address, and the following four zeros the netmask. For IPv6 the line ":: 0" should be added into the HNA6 block. With these settings, OLSR can be started with the command "olsrd -i wlan0 -d 1". In contrast to babel we have to specify the interface with the named parameter -i. The parameter -d 1 enables the debug output with 1-hop and 2-hop neighbors and their costs.

VI. Experimental results

The first testbed in the stairways was only used to get familiar with the protocols and their functionalities. Nevertheless this little experiment shows that rain can have an impact on wireless networks. The experiment started during sunshine an RPi 1 has two hops to reach RPi 3 (cf. 3). After a while it starts raining and suddenly RPi 1 is directly connected to RPi 3. The testbed was not changed. Unfortunately the weather can not be controlled, it was impossible to reproduce this behavior.

In the second test area, only OLSR was tested. Babel was not tested because there are so many troubles to establish an

hopcount	file size	bandwidth kb/s	time min:sec
1	10kb	654	00:00,02
1	100kb	535	00:00,20
1	500kb	552	00:00,90
1	1mb	571	00:01,80
1	10mb	572	00:18,00
1	20mb	560	00:37,00
1	100mb	570	03:10,00
1	1gb	570	30:40,00
2	10kb	35	00:00,30
2	100kb	40,1	00:02,50
2	500kb	31,3	00:16,00
2	1mb	48,4	00:21,00
2	10mb	44,7	03:49,00
2	20mb	67	05:06,00
3	10kb	26,4	00:00,40
3	100kb	9,8	00:11,00

TABLE I
ENVIRONMENT 2, OLSR

ad-hoc connection between two RPis. On the one hand it is caused by static barriers like thick concrete walls and metal doors. On the other hand by moving barriers like persons who walks around. On the smooth surface of the walls the wireless signals may be reflected and causes so interferences or over ranges. In table I is shown that the first hop works great and offers a good throughput. But in the second hop the bandwidth drops down to about 10 percent. The package loss increases rapidly. In the third hop, nearly no connection could be hold. Downloading the 10 and 100 kb test file was successful after a lot of trials. A package loss of 85 percent was measured. The main problem is that OLSR uses generally the shortest path in count of hop. The link quality doesn't matter. This causes a row of very long and unreliable links between the RPis. Although different positions for each RPi are used, no enhancement was observed. Even if each link has an visible connection between two RPis and the doors are kept open, no reliable communication was possible. Downloading the two files may be therefore only a fluke.

The third test environment which is located outdoors between two building complexes, delivers the best results. Here the testbed used up to 7 RPis in a row. In maximum a 6-hop routing can be evaluated. The first challenge was to figure out the optimal locations for each RPi. A pi which is lying on the ground has got a bad range. Also all RPis are placed elevated to increase their range. For example fireplugs and seatings are used. In this testbed narrowly no issues with over ranges of wifi signals are mentioned. This may be related to the surface of the walls. It isn't smooth but has got big grooves. Again we have for each link between two neighboring RPis a visible connection. In general, the testbed was suiting, but the links crosses commonly uses ways of students. Especially small groups of students, moving slowly through the link, causes outages of the affected link. This outages leads to unsuccessful download trials. The measurement for OLSR was taken after lunchtime and in the middle of a lecture time block, so most students are in the auditoriums. There was still an amount of students who interrupts some links. This leads despite multiple download attempts to many unsuccessful trials. This is the reason for the sparse table II. The 1-hop connection was not affected by walking students because it doesn't cross a way. The 2-hop connection is slightly affected. The 100mb and 1gb test wasn't successful because the takes to long time. With the following hops, the count of crossing the way increases just like the package loss and the throughput decreases. RPi 7 was place near the entrance to building 25.41 cf. image 4. It could reach the RPi 1 only with 6 hops. With each hop, the throughput convergences to zero. Tests with an eighth RPi was not successful. Nearly 95 percent of package loss prevent form downloading a file.

As an final test, the last RPi was moved several times between the positions 2 to 7 according to image 4. Downloading the 10mb file takes 6 minutes and 54 seconds. With a slow walking speed, it was no problem to move from 6 to 2, but in the opposite direction, the update of the routing table takes to much time. This means that at each position, we have to pause walking. The RPi was moved from position 7 to 2, stays there for some seconds, and moved back to position 7. Now it moves again back to position 2. After these the moves the 10 mb file was downloaded completely.

hopcount	file size	bandwidth kb/s	time min:sec
1	10kb	554	00:00,02
1	100kb	535	00:00,20
1	500kb	560	00:00,82
1	1mb	567	00:01,90
1	10mb	576	00:18,00
1	20mb	567	00:35,00
1	100mb	558	03:13,00
1	1gb	579	29:54,00
2	10kb	235	00:00,04
2	100kb	78,6	00:01,30
2	500kb	75,3	00:06,60
2	1mb	81,7	00:13,00
2	10mb	84,6	02:01,00
2	20mb	90,4	03:47,00
3	10kb	20,8	00:00,50
3	100kb	11,2	00:19,00
3	500kb	16,6	00:45,00
4	10kb	12,3	00:00,80
4	100kb	7,54	00:13,00
4	500kb	33,2	00:21,00
4	1mb	27,5	00:37,00
4	10mb	19,5	11:36,00
5	10kb	4,34	00:08,70
5	100kb	3,98	01:04,00
5	500kb	10,1	04:38,00
6	10kb	9,2	00:01,10

TABLE II
ENVIRONMENT 3, OLSR

A little bit later, the tests for babel are processed in the evening. At this time, nearly all students are home. This leads to smoother values with lower outliers in count of the bandwidth within each hop. But just like OLSR the bandwidth decreases with each hop. The measured values are presented in table III. The results shows that the bandwidth is nearly halved at each hop. An exception is between the firs and second hop, here decreases the throughput roughly down to one fifth. The reason for this relatively large contraction is unknown. Maybe

it is related to the double use of the wireless interface. At the first hop, both devices has only one communication partner. In a 2-hop routing, the RPi in the middle has to communicate to two partners. The interface can't send or receive different packages to both communication partners simultaneously, so it has to send or receive it in a row. The RPis in line has got the same problem. This is one reason for the deceased throughput.

All experiments downloading the test files the TCP protocol was used. TCP offers an reliable communication in unreliable networks. To make this possible, each not properly received package is retransmitted. In combination with a high package loss rate, the throughput decreases quickly. If the retransmitted packet will get lost too, we can get an downward trend, so that nearly no new package will reach its destination. This packet loss and retransmission problem take place on each link between two RPis. Hence it is not surprisingly that babel can't communicate reliable over more than 6 hops in a line configuration.

hopcount	**file size**	**bandwidth kb/s**	**time** min:sec
1	10kb	552	00:00,02
1	100kb	542	00:00,20
1	500kb	565	00:00,90
1	1mb	545	00:01,90
1	10mb	572	00:18,00
1	20mb	568	00:36:00
1	100mb	572	03:10,00
1	1gb	567	29:24,00
2	10kb	84,5	00:00,12
2	100kb	79,9	00:01,20
2	500kb	91,4	00:06,10
2	1mb	75,2	00:13,30
2	10mb	82,3	02:02,00
2	20mb	84,1	03:35,00
2	100mb	90,3	18:26,00
3	10kb	60,4	00:00,16
3	100kb	45,3	00:02,20
3	500kb	50,1	00:09,90
3	1mb	44,8	00:22,30
3	10mb	56,6	02:56,00
4	10kb	23,5	00:00,40
4	100kb	26,2	00:03,80
4	500kb	9,1	00:53,00
4	1mb	21,8	00:45,60
4	10mb		
5	10kb	15,2	00:00,65
5	100kb	9,8	00:10,20
5	500kb	10,1	00:51,20
5	1mb	5,6	02:58,00
6	10kb	5,7	00:01,80
6	100kb	4,6	00:21,70

TABLE III
ENVIRONMENT 3, BABEL

In the shown use case both protocols are very similar with regard to multi hop throughput and package loss in a line configuration. As mentioned in section III we use only the limited range of the wireless interfaces. This leads to the problem that the signal strength doesn't breaks down abruptly but rather decreases slowly with increasing distance. The main problem in all testbeds was to figure out the best position so that a RPi can see its 1-hop neighbors but not its 2-hop neighbors. For that was the environment three the best one because it has got many corners and inlets so that all RPis can place relative near to each other without enabling direct connection to the 2-hop neighbors.

Comparing OLSR and babel, it can be seen that in the present configuration no large differences are mentioned. The slightly better values measured with babel are not significant. It may be caused by the less moving obstacles in the links during the test session. Furthermore a very static testbed was used so that if the routing table is once created, it need no updates anymore. At this point, the protocols fits equal for the test scenario.

VII. CONCLUSION

This paper presents a MANET testbed with Raspberry Pis. The RPis which runs a linux version creates an ad-hoc network with an MANET routing protocol. Two protocols with different route finding algorithms are compared to each other. OLSR with its implementation olsrd on the one hand and babel on the other hand. At first there are three test environments checked whether they are suitable for testing or not. Only the third place provides a matching environment. It is contorted but due to the structure of the surrounding walls, reflection on them are nearly eliminated. Another problem was the short range of the little USB-wireless lan adapter. Compared to an Thinkpad T440p with an Intel corporation wireless 7260 network device, the USB device has roughly only the half range. As a next result, it could observed that rain influences wireless communication.

To our knowledge,this is the first study to examine a MANET consisting of Raspberry Pis. Especially the shared internet connection in the MANET is examined. The throughput data is collected in an static network. Only one test with a moving destination was made. In opposite to [3] this paper shows not only the connection between nodes of the mesh network. The main challenge is the multi hop download. This is presented by a line configuration of the RPis. As following research the dense of RPis can be increased to create a real mesh topology to offer more than one possible route. A next step can be to evaluate the bandwidth if more than one node downloads a file. For example RPi 2 starts a download and RPi 5 starts another file download at the same time. Furthermore a MANET with more than one gateway could be established to improve the internet connectivity. Based upon the created MANET, in future will be created a publish subscribe network with moving nodes.

REFERENCES

[1] J. S. G. Ramandeep Kaur Nagra and G. S. Grewal, "Simulation based Analysis of AODV, BABEL and PUMA Protocols for Adhoc Network," in *International Journal of Computer Application*, 2012.

[2] M. I. Elis Kulla, Masahiro Hiyama and L. Barolli, "Comparison of Experimental Results of a MANET Testbed in Different Environments Considering BATMAN Protocol," in *Third International Conference on Intelligent Networking and Collaborative Systems*, 2011.

[3] F. X. Leonard Barolli, Makoto Ikeda and A. Duresi, "A Testbed for MANETs: Implementation, Experiences and Learned Lessons," *IEEE Systems Journal*, vol. 4, no. 2, 2010.