A Performance Analysis of Wireless Mesh Networks Implementations Based on Open Source Software

Iván Armuelles Voinov, Aidelen Cedeño, Joaquín Chung, Grace González

A Performance Analysis of Wireless Mesh Networks Implementations Based on Open Source Software

Iván Armuelles Voinov, Aidelen Chung Cedeño, Joaquín Chung, Grace González

Research Center for Information and Communication Technologies,
University of Panama, Republic of Panama
{iarmuelles,achung,jchung, ggonzalez}@citicup.org

Abstract. Wireless mesh networks (WMNs) have emerged as a promising technology, capable of provide broadband connectivity at low cost. Implementations based on Open Source Software of these networks offer advantages for providing broadband networking communications in scenarios where cabling is too expensive or prohibitive such as rural environments. In this paper we evaluate the performance of small scale wireless mesh WMN routing protocols for WMNs: B.A.T.M.A.N. Advanced and the 802.11s standard. We also compare an OpenFlow controller implemented over the WMN, verifying their bandwidth, datagram loss and jitter.

Keywords: Open Source Software for research and innovation, Wireless Mesh Networks, OpenFlow, OpenWRT, network performance.

1 Introduction

Providing telecommunication services to difficult access areas (such as rural environments) is still difficult due to the lack of appropriate or inexpensive infrastructure. In this context, Wireless Mesh Networks (WMN) are an attractive solution for these scenarios due to their lower deployment costs and ease of expansion. WMNs could be used in community networks, home networking, video surveillance and emergency/disaster situations [1]. However, WMNs face several restrictions such as low-end equipment, single wireless channel and interferences, which degrade the overall performance of the network, and impose many drawbacks to even current and standard Internet's services. In this paper we evaluate the performance of WMN implementations based on Open Source Software for multimedia service transport. In our case, WMNs based on layer 2 routing protocol raise as a suitable and optimal connectivity choice, as any layer 3 addressing protocol could be used on top, either IPv4 or IPv6. Hence, we compare IEEE 802.11s standard [2] against Better Approach To Mobile Adhoc Networking Advanced protocol (B.A.T.M.A.N.) [3]. Also, we compare these two protocols with the OpenFlow protocol [4], which is used to control the forwarding tables of switches, routers and access points from a remote server, leveraging innovative services over the network such as access control, network virtualization,

mobility, network management and visualization. The paper is organized as follows: in Section 2, we describe the implementation of our testbed. In Section 3, we describe the test and show the results obtained. Finally, in Section 4, we describe our conclusions.

2 Testbed Implementation

The testbed consisted of a small scale WMN composed by four wireless routers. The experiments were conducted inside a small laboratory, because this indoor environment is very similar to the conditions of a real home networking scenario (Fig 1). All the wireless routers used in this testbed were TP-Link TL-WR1043ND v1.8, with four LAN ports of 1 Gbit/s Ethernet, one WAN port of 1 Gbit/s Ethernet, and one 802.11n wireless interface that works in the 2.4 GHz frequency band. We replaced the firmware of the wireless routers with the open firmware OpenWRT [5], which is a very well-known Linux distribution for embedded devices. OpenWRT supports WMN with routing protocols like OLSR, B.A.T.M.A.N. and the new standard for WMN networks 802.11s. For the experiments we used the OpenWRT Backfire 10.03.1, which comes with 802.11s support by default. To evaluate B.A.T.M.A.N. Advanced, was necessary to install the batman-adv package. For the experiments with OpenFlow we compiled OpenWRT with a package called Pantou that supports the OpenFlow protocol. The OpenFlow controller used in the experiments was POX, which is a controller based on NOX for rapid deployments of SDNs using Python. A list of all the components of the testbed and used tools based on Open Source Software is shown in the table 1

3 Performance Evaluation

Our experiment had three scenarios, the first one was a WMN composed by four MP configured with the 802.11s standard. The second scenario was the same four MP but using the batman-adv protocol. The last one was the four MP connected to an OpenFlow controller using an out of band network for control and a data network , i.e, a wired network for the control signaling, due to a limitation of the hardware selected . The tests were conducted in a low interference environment, which maintains the optimum conditions for VoIP traffic (packet loss should not exceed 1%, the maximum delay should <150 ms and jitter must be kept below 20 ms). For our study we took measurements using Iperf either in UDP and TCP mode. Every UDP test was maintained for 300 seconds and repeated ten times, while the TCP test where maintained for 180 seconds and repeated five times. Every test was made for one hop, two hops, three hops and four hops, for both scenarios.

In the *UDP* test we found that for one and two hops the maximum *throughput* was higher in the Batman-adv scenario. This is because in the Batman-adv implementation, the wireless interfaces synchronize at 802.11n (transmission rates up to 300 Mbit/s). On the other hand, the implementation of 802.11s only synchronizes at

Fig. 1. WMN implementation

Table 1. Tools based on Open Source Software for WMN implementation and their evaluation

Name	Supported Platform	Features
OpenWRT	Wide variety of wireless routers	Allows you to customize the applications on the wireless router. It implements routing protocols such as OLSR and BATMAN. It can be adapted to work with IPv6 and supports 802.11s. OpenWrt is the framework to build an application without having to build a complete firmware around it.
Pantou	Linksys and TP-Link Wireless Router	Implementation of the OpenFlow protocol for the OpenWRT firmware
NOX/POX	Linux	OpenFlow controller based on Python and C + +
Mininet	Linux	Allows you to create scalable software defined networks within a single PC.
Insider	Linux and Windows	Locate wireless networks and measures the intensity of their signals.
Iperf	Linux and Windows	Creates TCP and UDP data flows to measure the behaivor of the network with respect to some QoS parameters.
Wireshark	Linux and Windows	Protocol analyzer used for analyzing and solving problems in communication networks

802.11g (54 Mbit/s). For the OpenFlow scenario, the data network is based on a 802.11s WMN, where the mesh interfaces are controlled by OpenFlow. We did not get results for OpenFlow at three and four hops because Iperf did not show the report. The reason is the default behavior of the OpenFlow switch, it cannot send a packet through the incoming port. However, in a WMN this behavior is valid in a node acting as a relay. This behavior caused a large amount of errors and Iperf did not show a report. As a matter of fact, batman-adv is able to achieve higher throughput at three hops, but at the expense of a higher percentage of packet loss and jitter. Finally, at four hops the throughput of batman-adv is again greater than the throughput of 802.11s, and OpenFlow was not able to pass traffic correctly. Batman-adv has an overall greater throughput, albeit at three hops 802.11s shows a better performance. The control signaling of OpenFlow have a low impact in the performance of the WMN routing protocol.

In the *jitter* measurements we found that for all hops Batman-adv had a greater jitter than the 802.11s standard, however the maximum value is still below the 20 ms permitted for a good VoIP call. The results for three and four hops are missed because the same reason of the UDP throughput test. The jitter for one hop was greater than the jitter for two hops, for all the three protocols. This is because at one hop the test

was conducted using a PC for the Iperf server and one of the wireless routers as the Iperf client, which have a lower computing power.

With respect to the *loss*, it was determined that batman-adv has a higher percentage of lost datagrams than 802.11s until the third hop, while sending UDP traffic. At the fourth hop, 802.11s had too many errors and duplicated packets, so the results reported by Iperf were unreliable. The same behavior was observed for OpenFlow at three and four hops. We made tests sending *TCP traffic* to measure the maximum throughput allowed. Batman-adv obtained greater throughput than the 802.11s standard and OpenFlow. Since TCP has mechanisms for detecting and correcting errors, the throughput of batman-adv at three hops is greater than the throughput of 802.11s and OpenFlow in this case, regarding the results obtained in UDP. Besides, for all the cases batman-adv showed the best performance of the three protocols under study.

4 Conclusion

From the results of this experience we can conclude that both layer 2 routing protocols implemented with Open Source Software for WMNs have advantages and disadvantages. The B.A.T.M.A.N. Advanced protocol achieves higher transmission rates than 802.11s, but at the expense of a higher percentage of datagram loss. However, the throughput of the WMN is not an impediment for services such as videoconference; the current video codecs allow high quality videos with lower bandwidth requirements. Besides, the 802.11s showed a lower jitter than batman-adv, which is better for real-time communications. 802.11s is an IEEE standard, consequently many equipments in the future will support this protocol. Also, 802.11s is more secure because it does not have a SSID field in the frame, so it cannot be easily sniffed. Finally, 802.11s has support for multicast inherently. Regarding OpenFlow, the architecture based on a control network separated from the data network (as proposed by Dely et al. in [4]) shows an acceptable performance compared to the 802.11s standard. The in-band control approach is not recommended for WMN deployments for rural communities, due to its bad performance.

5 References

[1] I. F. Akyildiz, "A survey on wireless mesh networks," IEEE Communications Magazine, vol. 43, no. 9, p. S23-S30, Sep. 2005

[2] "IEEE Standard for Information Technology--Telecommunications and information exchange between systems--Local and metropolitan area networks--Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications Am," IEEE Std 802.11s-2011, pp. 1-372, 2011.

[3] D. Seither, A. Konig, and M. Hollick, "Routing performance of Wireless Mesh Networks: A practical evaluation of BATMAN advanced," Local Computer Networks (LCN), 2011 IEEE 36th Conference on, pp. 897-904, 2011.

[4] P. Dely, A. Kassler, and N. Bayer, "OpenFlow for Wireless Mesh Networks," in Computer Communications and Networks (ICCCN), 2011 Proceedings of 20th International Conference on, 2011, pp. 1-6.

[5] OpenWrt, https://openwrt.org/