DynTun: A Tool for Providing Multihoming Support in Wireless Mesh Networks

Clayton Reis da Silva, Diego Passos, Jairo Duarte, Igor M. Moraes, Célio V. N. Albuquerque
Laboratório MidiaCom
Instituto de Computação
Universidade Federal Fluminense, Brazil
Email: {creis, dpassos, jduarte, igor, celio} at ic.uff.br

***Abstract*—This paper proposes DynTun, a tool to provide multihoming support in wireless mesh networks (WMNs). This problem rises from the interaction between dynamic routing usually employed in WMNs and the use of NAT (Network Address Translation), which is largely used by access networks to distribute IP addresses between clients. The high variability in routing choices may result in frequent changes of the default gateway, which associated with the use of NAT may lead to connection disruptions. Experiments are performed on two WMN testbeds in order to verify and validate DynTun. Our evaluation shows that it is possible to harmonize the use of multiple gateways and NAT on the same access network. With DynTun, no connections are broken and the WMN aggregated bandwidth increases up to 7%.**

I. Introduction

Wireless mesh networks (WMNs) are based on a backbone of stationary wireless routers, which cooperatively provide Internet access to mobile clients through multihop communication [1]. Therefore, one router typically plays the role of a gateway to the Internet.

In this scenario, traffic is concentrated on links close to the gateway. Thus, clients have to share the fixed bandwidth of this gateway, which limits the number of clients accessing it. In addition, a WMN typically becomes inefficient as the hop count towards destination increases [2], [3]. Therefore, a client connected to a router far from the gateway experiences a maximum throughput lower than the one experienced by a nearby node, which leads to an unfairness scenario [4], [5]

A technique used to increase scalability and fairness in WMNs is to employ multiple gateways to the Internet [6], [7], [8]. This technique, referred to as multihoming, allows load balancing between multiple gateways and also reduces unfairness because clients can access the Internet by using the closest gateway. In this scenario, clients frequently experience gateway changes mainly because of two factors. First, clients must be able to choose a better gateway or a better route towards it. Second, clients are potentially mobile and thus they traverse different areas covered by different gateways.

In practice, gateways also have to allocate IP addresses to the clients. Typically, clients use private IP addresses and gateways are Network Address Translator (NAT) devices. When a client that is behind a NAT-enabled device moves to a new location that is also served by another NAT-enabled device, TCP connections may be broken because of the reconfiguration of the source address of the TCP segments. Thus, during gateway changes, applications running on clients may suffer with packet losses and disruptions.

This paper proposes DynTun (Dynamic Tunnels), an efficient solution to the problem of multihoming in NAT-enabled WMNs. Our solution creates tunnels dynamically and are also based on logical packet marking and routing policies. We have implemented DynTun and evaluated our tool in two different WMN testbeds. Our results show that DynTun maintains clients' connections as they change their gateways and also has the potential of increasing the WMN performance and scalability. With DynTun, no connections are broken and we have a side effect of a 7%-increase in WMN aggregated bandwidth.

The text is organized as follows. Section II explains in details the connection disruption problem when we are considering multiple gateways and NAT. This section also presents the requirements in order to define a solution that supports multihoming with NAT. In Section III, several existing proposals to solve this problem are analyzed. Section IV introduces DynTun and presents the implementation aspects of our tool. Section V presents the performance evaluation of DynTun in two real WMN testbeds. Section VI finally concludes this paper and discusses future works.

II. Multihoming with NAT-enabled Gateways

In WMNs, connection disruptions may occur when a client is accessing the Internet through a NAT-enabled gateway and then moves to a new location that is also served by another NAT-enabled gateway. As mentioned before, gateway changes are caused by dynamic gateway selection and clients' mobility. Figure 1 shows an example of how a given routing protocol deals with a link failure. Client *mh* initially uses the shortest route (the bold route) to the gateway *A*. At some point, a link failure occurs in the route between *mh* and the gateway. At this moment, the protocol must detect the failure and change the chosen path to an alternative one to reach the same gateway *A*, as shown in Figure 1(b).

In this example, the only effect that the client *mh* may perceive is a temporary packet loss and an increase of the end-to-end delay. Once the route change is complete, the connection returns to its standard behavior. If we analyze a similar topology in a network with two gateways, the

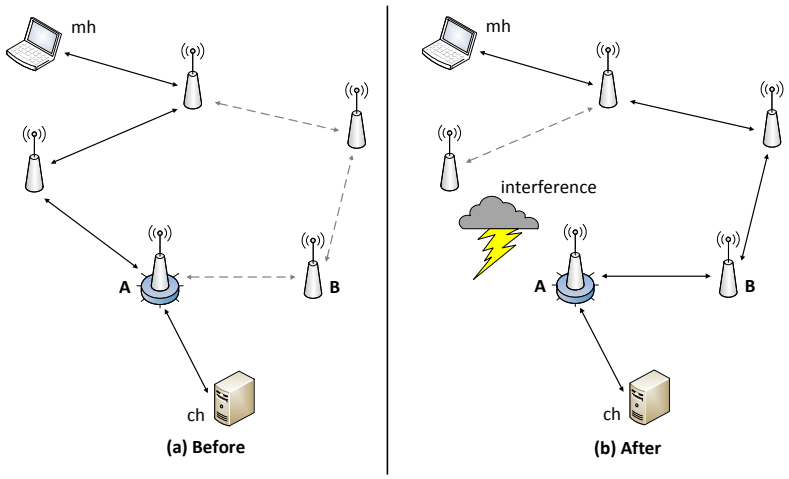

Fig. 1. Route change with one gateway.

connection problem becomes clear, as shown in Figure 2. In this case, we have another gateway, node *B*. Both *A* and *B* are NAT-enabled devices.

Assume that a connection is established as in the previous example. When a link failure occurs in the path between *mh* and the gateway *A* (the bold route), the routing protocol typically detects the failure and provides a new route that uses gateway *B*. In this situation, data packets will reach the server *ch*. The source address of these packets is the address of *B* because of NAT. Notice that before the link failure, packets addressed to *ch* leave the WMN with A's IP address in their source address fields. When *ch* receives the packets sent after the link failure, they will not be correctly associated to their previously established connection, resulting in a connection disruption. In this example, a gateway change is caused by a link failure. Gateway changes, however, may happen naturally with a WMN that employs quality-aware routing metrics, such as ETX and ETT, because the wireless link quality is highly variable [1]. It is possible to argue that a trivial solution to this problem is not to use NAT in WMNs. The NAT technique, however, is a key point to the development of wireless networks due to the reduced number of public IPv4 addresses available today. Thus it is necessary to propose solutions that allow the use of multiple gateways and NAT in WMNs.

A. Solution Requirements

In order to be easily deployed, we argue that a solution to avoid connection disruptions in NAT-enabled WMNs must have the following requirements:

- Be transparent and not be dependent on the Internet Service Provider (ISP).
- Have low implementation and maintenance cost.
- Operate in an autonomous and dynamic way.
- Be efficient in the WMN context.
- Be compatible with NAT technique.

All these requirements are considered in the solution that is proposed in this work.

III. Related Work

Once a connection begins, packets of a given client cannot arrive at the destination with a source IP address different from the address of the first packet. We have identified two classes of proposals that try to use multiple gateways and NAT in WMNs. In the first one, all packets leave the WMN with the same source IP address. In the second one, each connection uses only one gateway while it is active.

Most of the solutions of the first class employ a central NAT server that is placed externally to the WMN, which introduces a single point of failure and is costly to provide. In addition, all network traffic would be concentrated in this node, which may lead to server overload. An alternative would be changing the NAT mechanism to force gateways to forge an IP address. Thus, all packets of a connection leave the network with the same IP address. This solution, however, can violate some ISPs security policies, as many of them use the ingress filtering technique in order to avoid DoS (Denial of Service) attacks.

Suciu *et al.* [9] and Kniveton *et al.* [10] propose to create tunnels to Home Agents, which are additional servers that are external to the WMN with fixed IP addresses. Therefore, all packets are sent to these home agents and only after that packets are sent to the original destinations. Thus, these solutions introduce communication delay and violate the low cost requirement defined in Section II-A. Ancilotti *et al.* propose an adaptive solution that performs real-time measurements to maximize a given connection parameter and to achieve load balancing between gateways with different capacities. The authors, however, deal with the NAT problem by introducing a centralized server located outside the WMN. In this case, this centralized NAT server is a central point of failure. Shin *et al.* [11] propose a mechanism that deals with NAT problem requiring no additional server. This proposal, however, has two main drawbacks. First, the decision of

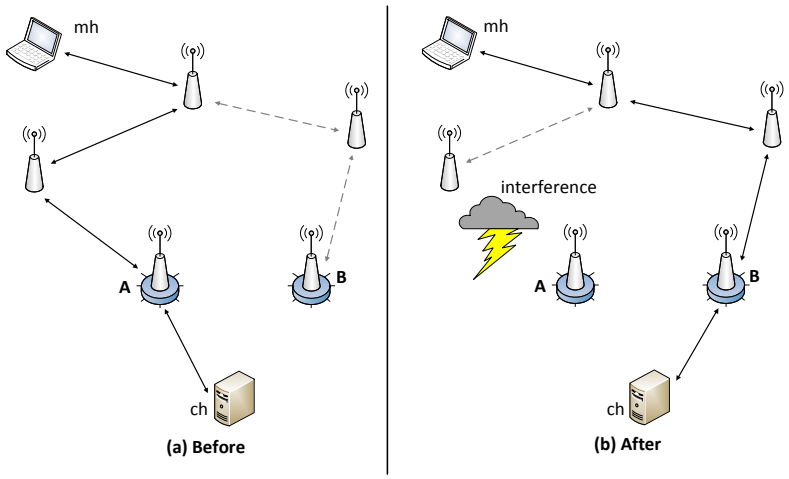

Fig. 2. Route change with two gateways.

selecting the best gateway is taken by the client device and thus this proposal is not transparent to the client because it has to manage the multihoming. Second, authors propose a conservative mechanism that uses the same gateway for all active connections. Another gateway is only selected when there are no more active connections.

The second class of proposals considers that each connection uses only one gateway while it is active. These proposals do not suffer from the problems that affect the solutions of the other class. Therefore, it fills most of the requirements specified in Section II-A. Engelstad *et al.* consider a network with characteristics similar to those of a WMN, using the NAT technique in the routers and creating tunnels to these routers. This proposal, however, is not transparent to client devices.

IV. The DynTun tool

Our tool, called DynTun (Dynamic Tunnels), monitors all gateways in order to identify new connection establishments. When DynTun identifies a new connection request, it chooses the best gateway for this connection and forces that all the following packets are forwarded through this same gateway. To achieve this goal, a wireless router dynamically creates a tunnel to the chosen gateway. When a router receives a data packet that comes from one of its clients, the connection to which the packet belongs is identified, along with the chosen gateway. The router then forwards the packet to the respective tunnel.

The information about available gateways can be obtained from the routing protocol. If a WMN employs a link state protocol, this information can be directly obtained because the whole WMN topology is known. If the routing protocol employed, however, is based on distance vector, it would be necessary to implement a lightweight gateway announcement mechanism [11]. Thus, routers can discover which is the best gateway in given moment. The best gateway decision is made through the underlying routing protocol.

DynTun affects only WMN routers, thus clients are not modified. Consequently, our tool is transparent to clients. On the other hand, with DynTun, routers must maintain state of active connections. The results in Section V, however, show that our tool is scalable.

The best gateway for each new connection is specified at the time the connection is established. For short-time connections, the selected gateway probably continues to be the best choice throughout the entire communication. For long-time connections, however, it is reasonable to suppose that the best gateway changes during the connection. Thus, performance may not be optimal through the entire communication.

It is worth mentioning that different connections opened by one user can be routed through different gateways. We are currently working on how DynTun may explore multipath routing in order to improve WMN performance.

With DynTun, if a connection is opened from a client to the Internet through a given gateway, all the packets of this connection will be forwarded to the destination using this same gateway. As packets traverse this route, the quality of its links tends to decrease [2], [3]. In WMNs, routing protocols typically consider quality-aware metrics to deal with this problem. In this case, if the quality of the route to the Internet decreases, these protocols will change this route or even the gateway. With DynTun, the route between a client and its selected gateway changes accordingly to the criteria employed by the routing protocol. Nevertheless, even if a better route to the Internet using another gateway is found during the connection, our tool still uses a route to the original gateway.

Connections that are opened while the first connection is active may use other gateways. This can happen, for example, if the previously chosen route becomes too loaded, according to the routing metric. Quality-aware routing metrics tend to present this behavior since they use active measurements for estimating link quality. Due to this characteristic, DynTun naturally provides load balancing in the network, as the gateways usage is leveled.

A. DynTun Implementation

DynTun is developed in the context of the Remesh project [12]. Our tool is running now in two different WMN testbeds. The first one, is an outdoor WMN deployed around the Universidade Federal Fluminense (UFF) *campus* located at Praia Vermelha, Niterói, RJ, Brazil. The second one, is an indoor, WMN situated inside one of UFF buildings. The outdoor WMN, presented in Figure 3, covers an area about 250000 m^2 of the *campus* neighborhood. The indoor WMN illustrated in Figure 4 connects seven rooms, such as laboratories, class rooms, and libraries, in two different floors of one building.

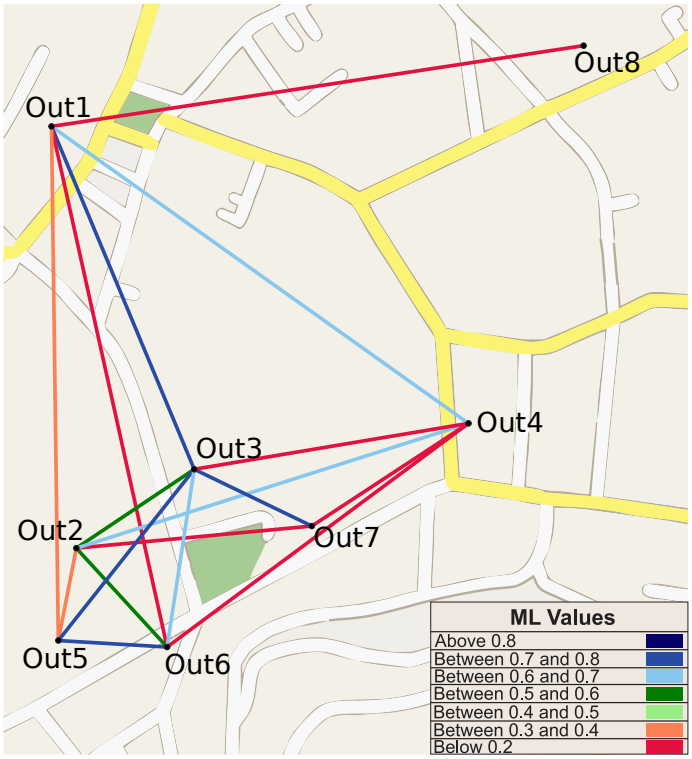

Fig. 3. WMN outdoor network.

Nodes of both networks are Linksys WRT54G routers running the OpenWrt operational system [13]. This hardware presents great restrictions [14], such as processing and memory. So, it is a good platform for the computational cost evaluation. We also consider that routers run the OLSR protocol [15]. We have implemented DynTun as a module for the OLSR protocol. We have chosen this approach in order to obtain the list of gateways in a WMN instead of implementing a new protocol for this purpose. Thus, we do not introduce additional control messages.

When DynTun is started, our tool creates a Generic Routing Encapsulation (GRE) tunnel [16] of Non-Broadcast Multi-Access (NBMA) type. We have chosen this kind of tunnels because they can be created without a specified destination address. Besides the tunnels, packet marking rules are created using the *conntrack* module of the iptables tool [17] tool.

DynTun works as follows. Each user connection receives a logical mark within the router kernel, and this mark will be

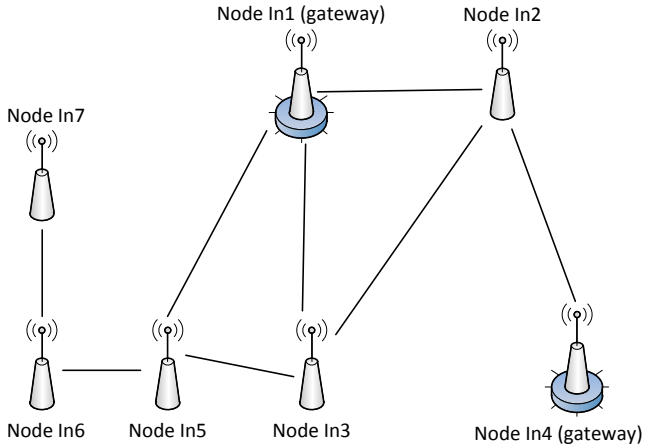

Fig. 4. WMN indoor network.

verified when our tool is choosing the gateway. The packet content is not modified. During the connection establishment, DynTun must do the following steps:

- Search for all available gateways in the WMN.
- Create one routing table for each gateway, which contains only one record (for that gateway).
- Identify the best gateway in during the connection establishment.
- Change the connection marking rule to use the mark that indicates the current preferred gateway. All packets of this connection will have the same mark used in the first packet.

DynTun performs the search for new gateways periodically. This action allows automatic detection of new gateways and the release of system resources when a gateway disappears. The gateway list is obtained during the topology processing done periodically by the OLSR protocol.

V. Performance Evaluation

We evaluate the performance of DynTun in the two WMN testbeds described in the previous section. Both networks have two nodes that can operate as gateways, depending on the current test. Figures 3 and 4 show the topologies of the outdoor WMN and the indoor WMN, respectively. Nodes Out1 and Out5 are the gateways of the outdoor WMN, and nodes In1 and In4 are the gateways of the indoor WMN.

A. Preserving connection semantics

Our first test aims at verifying the frequency of connections disruptions. The indoor topology node In2 is used as the source of several TCP flows sent to a server external to the indoor WMN. Node In2 was chosen because of it is geographically near to both available gateways. Thus, it is expected that frequent gateway changes occur because the quality of the node In2 links to each gateway is quite similar.

The experiment consists in opening 100 TCP connections, each started one minute after the previous one. The connections duration is fixed at 900 s, adding up to 114 minutes of test. These parameters were chosen to simulate connections used by file-sharing P2P applications. This connection type has longer duration, higher amount of transferred data and greater parallelism than other application types. Thus, it is more likely to suffer connection disruptions.

Over the test period, each connection was analyzed and eventual connections disruptions caused by gateway changes were observed. Without DynTun and considering 100 TCP connections, it was observed that 73 connections were not disrupted while 27 were broken. The average duration of the broken connections is 178.9 s. Table I shows the distribution of the duration of broken connections.

TABLE I
DURATION OF BROKEN CONNECTIONS.

Interval (s)	Number of broken connections
0-50	9
50-100	3
100-150	3
150-200	2
200-250	1
250-300	2
300-350	2
350-400	0
400-450	2
450-500	1
500-550	2

Results in Table I indicate that a mechanism to avoid connections disruptions caused by gateway changes is needed to provide multihoming support in WMNs. According to the Table I, more than 25% of the connections were broken and 33.3% of these connections were active during less than 50 s. With DynTun, this same set of tests is repeated and no connections disruptions are observed.

B. Aggregated Throughput

The goal of the following tests is to evaluate the aggregated throughput achieved with DynTun. We compare the obtained throughput considering different connections between nodes inside the WMN network and servers placed in the Internet with and without our tool.

The first set of experiments is performed in the indoor WMN network using nodes In2 and In3 for three different configurations. For all these configurations, we measure the throughput achieved by simultaneous TCP connections from nodes In2 and In3 to a server placed externally to the WMN. In the first configuration only node In4 is a gateway to the Internet. In the second one, only node In1 is a gateway. The results of these two configurations show the impact of the physical position of gateways on the aggregated throughput. Our goal, however, is to recognize the real contribution of the DynTun tool. Thus, in the third configuration, we consider both gateways In1 and In4 and DynTun. Table II shows the obtained results.

TABLE II
INTERNAL NETWORK THROUGHPUT IN THE THREE SCENARIOS (IN MBPS).

	Only GW In1	Only GW In4	DynTun
Node In2	3.12	4.66	2.97
Node In3	3.88	0.22	4.51
Aggregated	7.00	4.88	7.48

For the first configuration, the only available gateway is node In4. Thus, node In3 needs 3 hops to reach the server in the Internet. In addition, node In2 is also placed in the path used to reach the gateway. In this case, both nodes contend for the medium. With this test, we show the unfairness in WMNs. In this case, node In2 prioritizes its own connection over the connection of node In3.

On the other hand, for the second configuration where node In1 is the only gateway, nodes In2 and In3 are 2 hops distant from the server. Thus, both nodes experience a similar throughput.

When the two gateways are used with DynTun tool, each TCP connection can be initialized towards a different gateway. According to the results, DynTun provides a 7%-growth in the aggregated throughput. The throughput obtained by In3 is twenty times higher than the one obtained by using only node In4 as a gateway.

A second set of experiments is performed in the WMN outdoor network. In these experiments, a TCP flow of 20 minutes is started and then, 10 minutes later, a second TCP flow is started while the first connection is still active. Both flows have the same source node (Out3) and the same destination (an external server in the Internet). With DynTun, the first connection is opened through node Out5 and thus the second one must use node Out1 as the gateway to the Internet.

This experiment is repeated 16 times, achieving a total of 8 hours of for each scenario (with one or two gateways). While this test is executed, we observe that with DynTun and two gateways the aggregated throughput increases.. It could also be noticed that the deviation in the two gateways scenario were remarkably lower than the deviation in the one gateway scenario. Table III shows the minimum, maximum, average and standard deviation values of this experiment.

TABLE III
MAXIMUM, MINIMUM, AVERAGE, AND STANDARD DEVIATION FOR THE EXPERIMENTS IN THE OUTDOOR WMN (KB/S).

Configuration	Max.	Min.	Avg.	Std. Dev.
One gateway	4096	1366	2591.25	846.30
Two gateways (DynTun)	4687	3752	4062.31	287.05

The difference between the maximum throughput values for both configurations is in order of 600 Kb/s. The differences between the minimum and the average throughput, however, is high. In addition, the standard deviation of the two gateways configuration is approximately three times lower than the configuration with only one gateway. These values indicate the benefits of using DynTun in order to increase the network throughput. The results present presented an increase

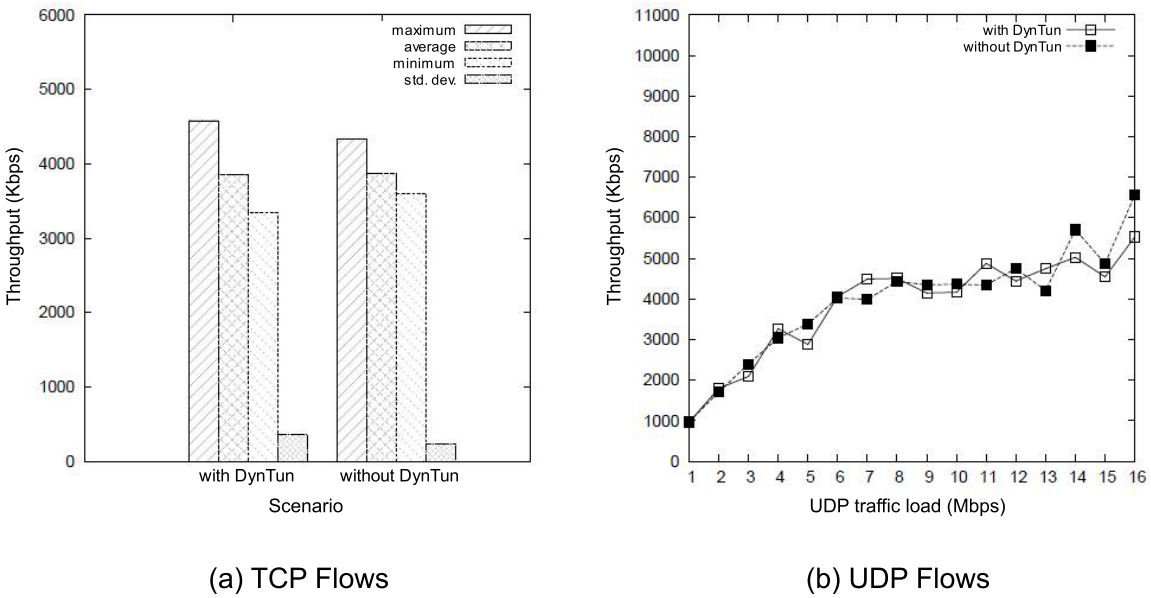

Fig. 5. Demonstration of the low impact of the solution over different data flow types

of 56.77% on average.

C. Processing Overhead

The fact that routers need to maintain the connections state can compromise the performance of DynTun. In order to evaluate the impact of our tool in a real environment, two similar experiments is performed. On the first experiment, a TCP data flow is created from node Out3 placed in the outdoor WMN to a server placed in the Internet. In this test, node Out1 is disabled and thus the network has only one gateway (node Out5). In this configuration, the experiment is performed without using the DynTun tool. After that, the experiments are repeated with DynTun. The obtained results is shown in Figure 5a. The average value refers to 12 runs of 5 minutes for each configuration. It is clear that the use of DynTun tool does not affect the performance in this scenario. For both situations, the average values are quite similar.

In the second set of experiments, we consider UDP flows and varies the transmission rate from 1 Mb/s to 6 Mb/s. In this case, the experiments have the duration of 5 minutes. Figure 5b summarizes the results.

Again, the two configurations achieved similar results to all evaluated rates. DynTun causes an increase in processing done by routers. This increase, however, is not significant in the evaluated topology if compared with the channel performance. It is worth mentioning that experiments are performed by using resource-constrained routers [14]. The processor has a 216 MHz clock and the available RAM is only 16 MB.

VI. Conclusion

Mesh networks can use the multihoming technique to avoid performance loss caused by the high number of hops between users and network gateways. As shown in Section V, the simple combination of the multihoming and NAT techniques can result in a negative effect that is the high number of connection breaks. This problem certainly affects the network quality perceived by users. Therefore, a mechanism that avoids it is required.

This paper proposes the DynTun solution, a practical and real implementation that solves the multihoming-NAT interaction problem, preserving user connection semantics. Although focusing on avoiding connection breaks, Section V also shows an increase in network capacity when using DynTun. Particularly, one can see an increase in aggregate bandwidth and also a significant growth in the performance of users that are farther (in number of hops) from gateways.

Section IV describes DynTun's functionalities and various tasks could have a negative impact in network performance, because of the additional processing capacity needed to manage the tunnels. However, the third test in Section V showed the low impact that the DynTun implementation had in network performance. The DynTun implementation, although deployed in the Remesh network, is also applicable to other networks. Two items that may need adaptation, when deploying DynTun in other networks, are the gateway discovery process and the route quality measurement.

Some future works include accounting each gateway capacity and utilization load, offering QoS support, avoiding that same application connections be tunneled to different gateways (because some applications may have this restriction) and adding cryptography in the tunnels whit the objective to protect users data.

References

[1] M. Campista, P. Esposito, I. Moraes, L. Costa, O. Duarte, D. Passos, C. de Albuquerque, D. Saade, and M. Rubinstein, "Routing metrics and

protocols for wireless mesh networks," *IEEE NETWORK*, vol. 22, no. 1, p. 6, 2008.

[2] D. Passos, D. Teixeira, D. Muchaluat-Saade, L. Magalhães, and C. Albuquerque, "Mesh network performance measurements," in *International Information and Telecommunicatios Technologies Symposium (I2TS)*, 2006, pp. 48–55.

[3] D. Couto, D. Aguayo, J. Bicket, and R. Morris, "A high-throughput path metric for multi-hop wireless routing," *Wireless Networks*, vol. 11, no. 4, pp. 419–434, 2005.

[4] M. Campista, L. Costa, and O. Duarte, "Privileging long-life packets in multihop wireless networks," in *2007 9th IFIP International Conference on Mobile Wireless Communications Networks*, 2007, pp. 136–140.

[5] Q. Dong, S. Banerjee, and B. Liu, "Throughput optimization and fair bandwidth allocation in multi-hop wireless LANs," in *IEEE Conference on Computer Communications (INFOCOM)*. Citeseer, 2006, pp. 1–12.

[6] U. Ashraf, S. Abdellatif, and G. Juanole, "Gateway selection in backbone wireless mesh networks," in *Proceedings of the 2009 IEEE conference on Wireless Communications & Networking Conference*. Institute of Electrical and Electronics Engineers Inc., The, 2009, pp. 2548–2553.

[7] J. Ernst and M. Denko, "Fair Scheduling with Multiple Gateways in Wireless Mesh Networks," in *Proceedings of the 2009 International Conference on Advanced Information Networking and Applications-Volume 00*. IEEE Computer Society, 2009, pp. 106–112.

[8] D. Nandiraju, L. Santhanam, N. Nandiraju, and D. Agrawal, "Achieving load balancing in wireless mesh networks through multiple gateways," in *2006 IEEE International Conference on Mobile Adhoc and Sensor Systems (MASS)*, 2006, pp. 807–812.

[9] L. Suciu, J. Bonnin, K. Guillouard, and T. Ernst, "Multiple network interfaces management for mobile routers," in *Proceedings of the International Conference on Intelligent Transportation Systems Telecomunications (ITST), Brest*, 2005.

[10] T. Kniveton, J. Malinen, V. Devarapalli, and C. Perkins, "Mobile router tunneling protocol," *Internet Draft, Internet Engeneering Task Force (IETF)*, 2002.

[11] J. Shin, H. Lee, J. Na, A. Park, and S. Kim, "Gateway discovery and routing in ad hoc networks with nat-based internet connectivity," in *Vehicular Technology Conference*, 2004.

[12] J. Duarte, D. Passos, R. Valle, E. Oliveira, D. Muchaluat-Saade, and C. Albuquerque, "Management issues on wireless mesh networks," in *5th Latin American Network Operations and Management Symposium (LANOMS 2007)*, 2007.

[13] OpenWRT. (2010) Openwrt.org. [Online]. Available: http://openwrt.org/

[14] BCM4712 Hardware Specification. (2010) Broadcom products. [Online]. Available: http://www.broadcom.com/products/Wireless-LAN/802.11-Wireless-LAN-Solutions/BCM4712

[15] T. Clausen, P. Jacquet, C. Adjih, A. Laouiti, P. Minet, P. Muhlethaler, A. Qayyum, and L. Viennot, "Optimized link state routing protocol (OLSR)," 2003.

[16] D. Farinacci, T. Li, S. Hanks, D. Meyer, and P. Traina, "Generic routing encapsulation (GRE)," RFC 2784, 2000.

[17] Iptables and Netfilter. (2010) Iptables tool. [Online]. Available: http://www.iptables.org/