# Enhanced Mobility Management in Wireless Mesh Networks

M. Bezahaf[1], L. Iannone[2], and S. Fdida[1]

[1] Université Pierre et Marie Curie - Laboratoire LIP6/CNRS
[2] Université Catholique de Louvain - IP Networking Lab (INL)

**Abstract.** Despite considerable efforts, mobility management in Wireless Mesh Networks (WMN) remains an open issue. Several high performance solutions can be found in the literature, however, they all have the same requirement that refrains them from being widely adopted: they need either modifications or additional modules into the protocol stack of users' equipment. In this paper, we investigate the mobility problem in WMNs and propose a new efficient solution, which does not rely on any modification or additional software on the client side, thus being totally transparent for end-users. In our analysis, we first show the measurements performed on the existing WMN deployed in LIP6, namely MeshDVNet, and highlight the reasons of its poor performance. Then, we describe the design of *Enhanced Mobility Management* (EMM), our proposal, which does not need any supplementary installation. EMM takes advantage of the existing Neighbor Discovery Protocol (NDP) cache to keep a track of the last client association and uses this information to trigger an update in order to re-route packets. The measurements we performed show how EMM is able to greatly improve performances.

## 1 Introduction

Mobility management is a very important issue in current networking, since users are more and more mobile due to the widespread of wireless technology. This new tendency of clients, who want to move during communications with no constraint of connectivity, no additional software to install, and where changes of network are completely transparent, has induced the researchers to design new architectures. Wireless Mesh Networks (WMN) [1] fall in this category. WMNs are an emerging class of wireless networks, able to organize and configure themselves dynamically. They take the principle of a wireless network based on multi-hops transmission, *i.e.*, the communications between two nodes is supported by several intermediate nodes (called Wireless Mesh Router (WMR)) whose role is to relay the information. Their two-tier architecture concentrate routing on a stable wireless backbone (first-tier), composed of WMRs, while allowing mobile clients (second-tier) to connect to the backbone. In this context, the challenge is to preserve client's connections whatever the type of displacement.

Mobility management in Wireless Mesh Networks is composed of two phases: localization (also detection) and handover. The first phase consists in detecting,

possibly instantaneously, moving clients and determine their position in the network at any time and with a good precision (*i.e.,* have a realistic snapshot). The second phase completes the first one by updating the routing information in order to minimize the disconnection time, to avoid packet loss, and to maintain already opened client's connections.

Several existing proposals tackle mobility management. Nevertheless, despite good performances, they struggle to be really and widely deployed due to the necessity to modify the protocol stack or at least install additional software into clients' devices.

In this paper, we propose the *Enhanced Mobility Management* (EMM) approach, in order to efficiently manage mobility without the need, for end-users, to install any software or modify their protocol stack. EMM is based on the utilization of clients' Neighbor Discovery Protocol (NDP) cache. In particular, with EMM, WMRs inject a particular entry in the cache, which is used when client moves in order to recognize where the client was previously associated and make known that the client is now associated elsewhere. Measurements we performed on the MeshDVNet ([2], [3]) test-bed deployed at LIP6 show that EMM is able to greatly improve performances by substantially reducing disconnection time and, hence, packet losses.

The remainder of the paper is organized as follows. We review main examples of mobility management solutions in section 2, before analyzing the original behavior and related issues of MeshDVNet's mobility management in section 3. We then present EMM, our approach, in section 4. We performed several measurements on the MeshDVNet test-bed, clearly showing how EMM outperforms the original MeshDVNet proposal. These are presented in section 5. Section 6 summarize our achievements and concludes the paper.

## 2 Related Works

Wireless Mesh Networks [1] are often solicited for various purposes: community networks, enterprise or home networks, and local or metropolitan area networks. Some industrials have already marketed WMNs ([4], [5], [6]). They exist also some WMNs communities like NYC wireless [7] and Quail Ridge Wireless Mesh Network [8].

In such context, moving clients are often requested to change sub-network, consequently client's old IP address will not have any significance in the visited network. Changing client's IP address at each change of Access Point is not a good solution, because all TCP connections will be dropped, and all applications storing the IP address will suffer. To avoid this problem, different solutions have been introduced like Mobile IP ([9], [10]), HIP [11] and RendezVous mechanisms [12]. All these solutions allow clients to maintain the same identity (same IP or same Identifier) while visiting other sub-networks, also permitting their localization. Some existing works use Mobile IP to achieve mobility, like the OBAN project [13], where the backbone is built upon the existing private access points in urban areas. We can also quote experimental WMNs, which support mobility,

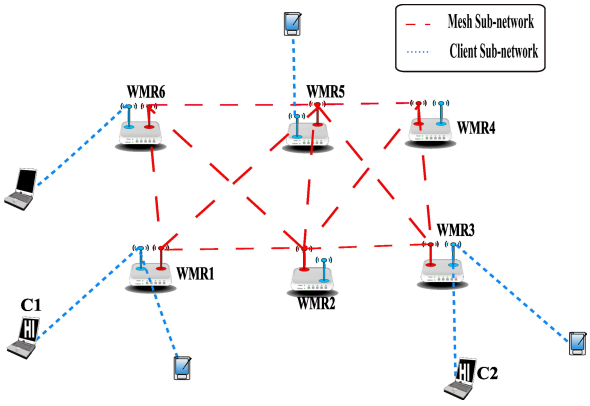



**Fig. 1.** An example of MeshDVNet deployment.

like iMesh Network ([14], [15]) or SMesh Network [16]. Nevertheless, our main objective is to have an efficient handover when clients move, with no client's pre-necessary installation, which is not the case in most existing solutions and protocols that introduce an additional software installation.

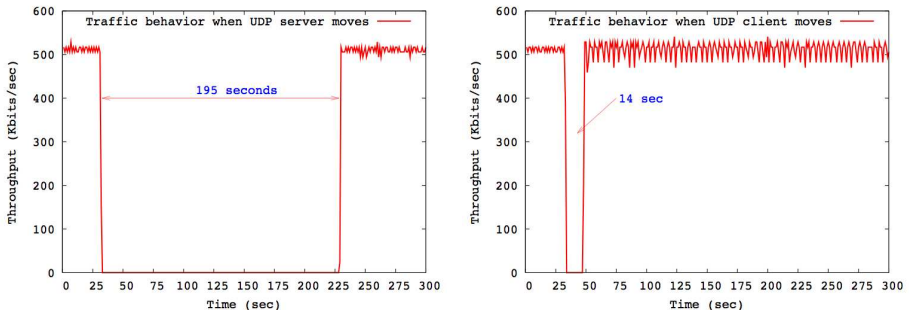
## 3 Mobility Management in MeshDVNet

To acquire a deep understanding of the existing issues in WMNs and to have a realistic study, we analyzed and performed several measurements on MeshDVNet ([2], [3]). Based on IPv6, MeshDVNet is a WMN test-bed deployed at LIP6, offering wireless connection to clients and allowing them to communicate with no pre-necessary installation. In order to have a real time snapshot of our test-bed and to monitor if WMRs are running or not, we use supervision web page, which is publicly available (only for IPv6 connections) at:

http://www.infradio-jussieu.lip6.fr/supervision/supervision-mesh-kennedy.html

MeshDVNet is decoupled in two sub-networks (Fig. 1): one formed by the set of WMRs, which constitute the backbone, and one formed by the set of clients. Named MeshDVbox, the routers (WMRs) used in MeshDVNet are Soekris net4521 running Linux (Crux distribution [17]), on which the MeshDV protocol is running. A MeshDVbox is equipped with two wireless interfaces, one for communications with others routers and one for communications with clients.

In order to explain how mobility is managed in the original version of MeshDVNet, and to point out current issues, we present hereafter how a communication between two clients associated to different WMRs is carried out. Let us assume the scenario presented in Fig. 1, and let us assume that the client $C_1$, associated to $WMR_1$ is starting a connection toward the client $C_2$, which is associated to $WMR_3$. The communication is set up in the following way:




(a) When server moves (b) When client moves

**Fig. 2.** MeshDV mobility performance using UDP traffic.

**1:** Since $C_1$ and $C_2$ are in the same logical sub-network, the client $C_1$ sends first a multicast neighbor solicitation, in order to obtain $C_2$'s MAC address. $WMR_1$ receives this solicitation packet and sends a MCREQ[3] packet to other WMRs to find out where the client $C_2$ is associated.
**2:** $WMR_3$ receives the multicast request (MCREQ), and answers with a CRREP[4] packet.
**3:** When $WMR_1$ receives the CRREP message, it replies to the $C_1$'s neighbor solicitation by a neighbor advertisement containing the $C_2$'s IP address associated to the $WMR_1$'s MAC address.
**4:** Thereafter, all packets sent from the client $C_1$ to the client $C_2$ are actually captured by $WMR_1$, encapsulated in IPv6 packets and sent to $WMR_3$.
**5:** $WMR_3$ will decapsulate the packets and it will transmit them to the client $C_2$.

Clients' movements detection in WMNs can be done in two ways: either the old WMR detects its client's movements (self-detection) or the new WMR detects a new client and notifies it to the old WMR (reactive-detection). The existing MeshDV's mobility manager module uses the self-detection approach, which performs poorly. If a client moves during a communication and changes association (*i.e.* WMR), layer 2 reconnection with the new WMR is carried out quickly, but the problem consists in the old WMR where client was connected. For the latter, the client is always connected to its interface, which is not the case, and all packets destined to this client will be forwarded by the old WMR to a non-existent destination. This is due to the mobility management in MeshDV, which is based on wireless card driver detection. In particular, in the MeshVDnet testbed, WMRs uses the Madwifi driver [18], which announces clients' disconnection after a three minutes timeout. Therefore, the IP layer of the WMR holds wrong information about its local clients who have moved, taking more than three minutes to realize that.

[3] MCREQ (Multicast Client REQuest) is a multicast packet, used to search remote clients, by asking which WMR manages a certain client [3]. Note that the multicast approach is not the most effective to perform network-wide lookups, however, proposing an improved mechanism is out of the scope of this paper.
[4] CRREP (Client Request REPly) is a unicast packet, used to reply to a MCREQ [3].

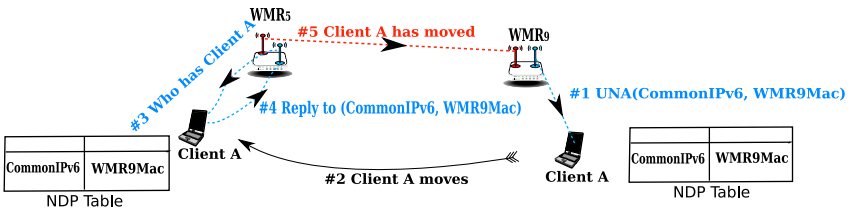



**Fig. 3.** EMM mobility management example.

We performed thorough measurements of MeshDVNet using Compaq nx7000 laptops, running Linux Fedora Core 3, as clients. We experimented different mobility scenarios that present some performance problems. We focused on the disconnection time that MeshDV suffers when a client moves. To perform that, we used three different kinds of traffic (TCP, UDP, and Ping) between two clients for 300 seconds and observed the disconnection time when one of them moves. Note that we use Iperf program [19] to generate TCP and UDP traffic. Fig. 2 shows the throughput during each second in the UDP case. When Iperf server moves during UDP traffic (Fig. 2(a)), we obtained around three minutes disconnection time (self-detection). In the case of the Iperf client moving (Fig. 2(b)), the client is instantaneously connected to a new WMR, and continues to send UDP packets with destination MAC address equals to the old WMR's MAC address. This is due to the Neighbor Discovery Protocol cache (NDP) [20], which is not refreshed immediately after the physical connection.

These two scenarios are very representative, since Ping and TCP traffic present very similar behavior [21]. In the following section, we propose our solution based on the NDP cache, which solves efficiently all the problems seen in this section.

## 4 Enhanced Mobility Management

Enhanced Mobility Management (EMM) is an improved approach to manage mobility, where clients' detection during their movements is effectively done by the new WMR (reactive-detection). Our approach is based on the NDP cache, which basically contains information about neighbors' IP addresses, their MAC addresses, and includes information about reachability state.

We use Click modular router [22] as the routing software infrastructure. In our implementation, each WMR runs Click in kernel level. Therefore, instead of processing all received packets in user level, we use Click to select the interesting packets.

In EMM, each WMR in the network at the bootstrap adds the same IPv6 address in local scope called *Common Address* to its client interface. This address is static and can be changed by configuration. Note that the *Common Address* is never used for data communication but only for mobility management. In order to update client's NDP cache, we use the Unsolicited Neighbor Advertisement

(UNA) message by introducing an entry for the *Common Address*. Presented in the RFC 2461 [20], UNA message is used by a node to inform its neighbors that its link-layer address has changed. This is the same principle used for the Web Browser Cookies mechanism, *i.e.*, clients themselves hold information about their last WMR association. In order to show how this cookie introduced in the NDP cache helps in managing mobility, let us consider the scenario of Fig. 3. Let us assume that client A initially associates to $WMR_9$ when entering the network, then it moves getting associated to $WMR_5$. In this context, EMM NDP cookie mechanism works in the following way:

---

**1:** When the Client A associates for the first time to $WMR_9$, it receives from the latter an UNA message *"associate the Common Address to the $WMR_9$ MAC address and store it in your cache"*. Therefore, Client A updates its NDP cache, holding the Common Address and the $WMR_9$'s MAC address association during its displacement.

**2:** The Client A moves from $WMR_9$ to $WMR_5$. After layer 2 connection, $WMR_5$ knows Client A's MAC address and derives its IP address[5]. Thereafter, it sends a Neighbor Solicitation (NS) packet to the Client A with IPv6 source address equal to the Common Address and MAC source address equal to the $WMR_5$'s MAC address *"Who has the Client A?"*.

**3:** The NS packet sent by $WMR_5$ is not used to obtain the Client A's MAC address, which is already known from the layer 2 association, but to know where the Client A was connected. When the Client A receives this solicitation, it checks its NDP cache and finds the Common Address cookie associated to the $WMR_9$ MAC address. Thus, it replies, using the $WMR_9$ MAC address.

**4:** Given that $WMR_5$'s wireless interface is configured in promiscuous mode, $WMR_5$ receives the reply which is then parsed by the Click software. $WMR_5$ can now extracts the $WMR_9$'s MAC address and derives the IP address of $WMR_9$ (using EUI-64). At this point, $WMR_5$ sends a CWIT[6] packet to $WMR_9$ in order to notify the Client A displacement. At the same time, $WMR_5$ sends an UNA message to the Client A *"associate the Common Address to the $WMR_5$'s MAC address in your cache"*, in order to update the cookie in the NDP cache of Client A.

**5:** If, after receiving the CWIT packet from $WMR_5$, $WMR_9$ receives a packet for Client A, it drops the packet and sends back to the WMR that issued the packet (Remote WMR) a Client ERRor (CERR) message *"The Client A is not anymore associated to me"*.

**6:** Receiving the CERR message, the remote WMR automatically sends a new MCREQ packet in order to find out to which WMR the Client A is now associated.

---

Fig. 4 shows the temporal diagram for Client mobility management mechanism using UNA packet. As we can see, the figure presents the different packets exchanged when the client $C_1$ moves from "Old WMR" to "New WMR" during its communication with the client $C_2$, connected to "Remote WMR".

[5] Note that we use the 64-bit Extended Universal Identifier (EUI-64) to obtain the IP address from a MAC address ([23], [24]).

[6] CWIT (Client WIThdraw) is a unicast packet, used by the new WMR to inform the old one for its client's displacement [21].

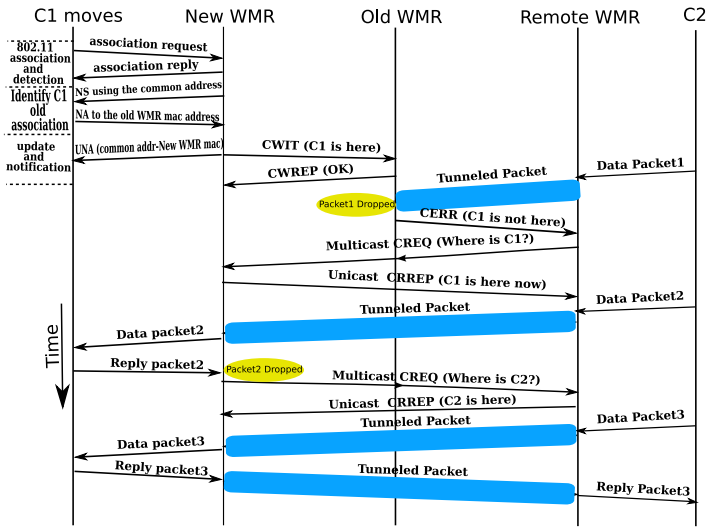



**Fig. 4.** Enhanced Client Mobility Management temporal diagram.

## 5 Enhanced Mobility Management Evaluation

In order to evaluate our work, we performed the same set of tests described in section 3. We measured the delay and evaluated the behavior of EMM when clients move and change WMR's association.

When using UDP traffic between the clients, we obtain very good results, with a disconnection time that is shorter than one second, as depicted in Fig. 5. When the client that generates UDP traffic moves, the new WMR must first detect the new client and then searches the correspondent remote client. Fig. 5 shows that the throughput between second 233 and second 234 decrease from 500 Kbits/s to 120 Kbits/s, which means that during this second, communication is disturbed over less than one second. Between second 234 and second 235 the throughput increase from 120 Kbits/s to 420 Kbits/s, which means that during this second, no packets are lost. Note that we cannot have the exact disconnection time, because the Iperf tool, which we are using, is not able to provide measurements with a granularity of less than one second.

When using TCP traffic between the clients, as Fig. 6(a) shows, we are not able to achieve the same performances like in the UDP case. Even if, compared to the original MeshDVNet disconnection time of 3 minutes, we have a great reduction, there is still a gap in the order of few seconds. This is due to the way TCP manages packets' retransmissions. Indeed, TCP uses a retransmission timer to ensure data delivery when packets are lost. The duration of this timer is referred to as RTO (Retransmission TimeOut), which is doubled each time that a TCP packet is not acknowledged during RTO seconds. When a client moves and after the layer 2 re-association to the new WMR, the latter receives a first TCP packet, however, it does not know where the correspondent client is con-

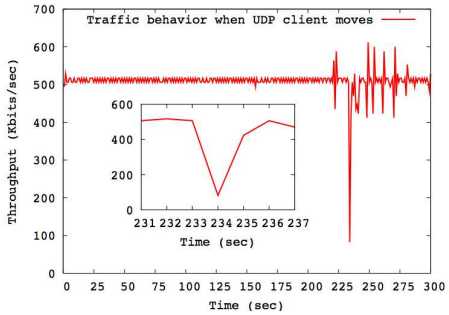
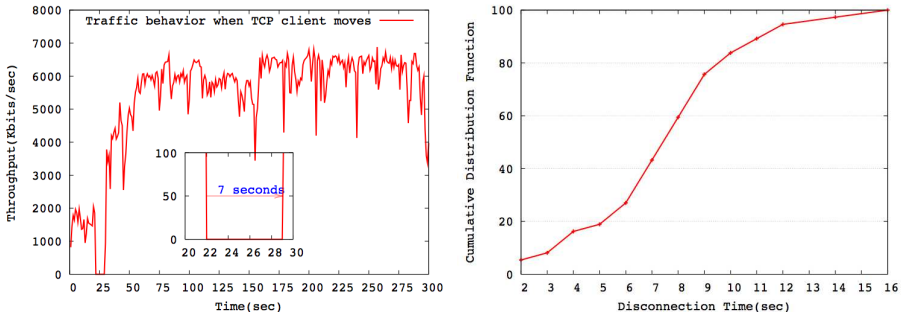



**Fig. 5.** Enhanced Mobility Management performance using UDP traffic.




(a) When server moves (b) CDF of client disconnection time

**Fig. 6.** Enhanced Mobility Management performance using TCP traffic.

nected, hence, it sends a MCREQ on the backbone while dropping the packet. This means that RTO doubles. When client sends the second packet, the correspondent receives it and acknowledges it normally; however, the correspondent's WMR (remote WMR) forwards the reply to the old WMR instead the new one. The old WMR drops this packet and notifies to the remote WMR that its client has moved by sending a CERR message, which means the RTO doubles once again. All this mechanism leads to a gap in the order of seconds. Actually, the clients' wireless card that we used in our tests, sometimes performs a complete scan of wireless channels, which increases the IEEE 802.11 handover latency to approximately 6.9 seconds. In this case we can obtain up to thirty-six seconds of disconnection time even if the detection and the WMRs tables update are practically instantaneous after the physical connection [21].

Fig. 6(b) shows the cumulative distribution of client disconnection time during TCP traffic with a fast IEEE 802.11 handover (*i.e.* without the complete channel scanning). From the graph, we can see that the maximum of disconnection time is 16 seconds and in 95% of the cases, the disconnection time is less than 12 seconds. We remark also that in some cases the disconnection can be short (2 to 5 seconds).

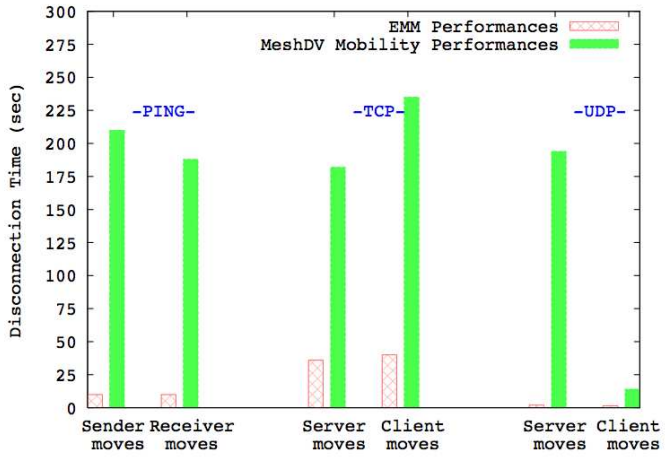



**Fig. 7.** Comparison of Disconnection Time between MeshDV mobility management and our approach.

In order to have a visual comparison between EMM and the old version of mobility management in MeshDV, we plot a histogram (Fig. 7) to show the improvements introduced by our solution. We decrease disconnection time from three minutes to a few seconds and in some cases less than one second, which is not a negligible improvement.

## 6 Conclusions

Existing solutions for mobility management in Wireless Mesh Networks, while, on the one hand, offer good performances, on the other hand, have difficulties to get widely deployed in real networks. The reason of such a shortfall can be found in the requirement of specific installation on end-user devices. The work presented in this paper represents our efforts in solving the mobility management issue without any impact whatsoever on end-user commercial equipment. Our proposal, called *Enhanced Mobility Management* (EMM), is the result of a thorough analysis that we performed on the MeshDVNet test-bed deployed in the LIP6 laboratory. EMM takes advantage of the NDP cache, present on all standard protocol stacks, by introducing a particular entry for a common IP address. Such an entry is used to retrieve the old association of a wandering client, allowing to re-route packets and thus to maintain ongoing communication with relatively small disconnection time. Compared to the original MeshDVNet proposal, EMM greatly improves performance for all types of traffic and mobility pattern, as our measurements clearly show. For instance, in the case of TCP traffic, disconnection time is reduced to less than 20% of the original one. Even more, in the case of UDP traffic and moving server, the disconnection time is reduced to less than 0.5% of the original value.

## References


1. I. Akyildiz, X. Wang, and W. Wang, "Wireless mesh networks: a survey," *Computer Networks - Elsevier Science*, no. 47, Jan. 2005.
2. L. Iannone and S.Fdida, "Meshdv: A distance vector mobility-tolerant routing protocol for wireless mesh networks," *IEEE ICPS Workshop on Multi-hop Ad hoc Networks: from theory to reality (RealMAN'06)*, July 2005.
3. L. Iannone, "Meshdv: Implementation draft," *Technical Report*, May 2005.
4. "Nortel." [Online]. Available: http://nortel.com
5. "Cisco systems." [Online]. Available: http://cisco.com
6. "Strixsystems." [Online]. Available: http://strixsystems.com
7. "Nyc wireless." [Online]. Available: http://nycwireless.net
8. P. Mohapatra, D. Wu, and D. Gupta, "Quail Ridge Wireless Mesh Network: Experiences, Challenges and Findings," *International Conference on Testbeds and Research Infrastructures for the Development of Networks & Communities*, Dec. 2007.
9. C. Perkins, *IP Mobility Support for IPv4, RFC 3344*, IETF, Aug. 2002.
10. D. Johnson, C. Perkins, and J. Arkko, *Mobility support in ipv6, RFC 3775*, IETF, 2004.
11. T. Henderson, *End-host mobility and multihoming with the host identity protocol, draft-ietf-hip-mm-05*, IETF, Mar. 2007.
12. L. Eggert and M. Liebsch, *Host Identity Protocol (HIP) Rendezvous Mechanisms, draft-eggert-hip-rendezvous*, IETF, Jul. 2004.
13. "The open broadband access network (oban) project." [Online]. Available: http://oban.prz.tu-berlin.de/html/presentations___papers.html
14. V. Navda, A. Kashyap, and S. Das, "Design and evaluation of imesh: an infrastructure-mode wireless mesh network," *IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WOWMOM)*, Jun. 2005.
15. V. Navda, S. Ganguly, K. Kim, A. Kashyap, D. Niculescu, R. Izmailov, S. Hong, and S. Das, "Performance optimizations for deploying voip services in mesh networks," *IEEE Journal on Selected Areas in Communication (JSAC)*, Nov. 2006.
16. Y. Amir, C. Danilov, M. Hilsdale, R. Musăloiu-Elefteri, and N. Rivera, "Fast handoff for seamless wireless mesh networks," *ACM Press*, pp. 83–95, 2006.
17. "The crux operation system." [Online]. Available: http://crux.nu
18. "Madwifi home page." [Online]. Available: http://madwifi.org/
19. A. Tirumala, F. Qin, J. Dugan, J. Ferguson, and K. Gibbs, "Iperf-the tcp/udp bandwidth measurement tool," 2005.
20. T. Narten, E. Nordmark, and W. Simpson, *Neighbor Discovery for IP Version 6 (IPv6), RFC 2461*, IETF, Dec. 1998.
21. M. Bezahaf, "Fast mobility in wireless mesh networks," Master's thesis, University Pierre et Marie Curie (Paris 6), 2007.
22. E. Kohler, R. Morris, B. Chen, J. Jannotti, and F. Kaashoek, "The click modular router," Aug. 2000.
23. IEEE, "Guidelines for 64-bit global identifier (eui-64) registration authority," IEEE Standards tutorials., 1997.
24. T. Narten, "Neighbor discovery and stateless autoconfiguration in ipv6," *IEEE Internet Computing*, vol. 3, no. 4, pp. 54–62, 1999.