2009-11-01

# MeshScan: Performance of Passive Handoff and Active Handoff

Yin Chen
*Technological University Dublin*, yin.chen@tudublin.ie

Karol Kowalik
*Technological University Dublin*, Karol.kowalik@tudublin.ie

Mark Davis
*Technological University Dublin*, mark.davis@tudublin.ie



## Recommended Citation

Chen, Y., Kowalik, K. & Davis, M. 92009) MeshScan: Performance of Passive Handoff and Active Handoff. WCSP'09: *International Conference on Wireless Communication & Signal Processing, Nanjing, China. 2009.*


Funder: Science Foundation Ireland (Grant No. RFP/ CMS704)

# MeshScan: Performance of Passive Handoff and Active Handoff

Yin Chen, Karol Kowalik, Mark Davis
Communications Network Research Institute,
Dublin Institute of Technology
Dublin, Ireland
yin.chen@cnri.dit.ie, karol.kowalik@cnri.dit.ie, mark.davis@dit.ie

*Abstract*—**A core problem of fast handoff is when handoff should perform and which Mesh Node (MN) should associated with. We have developed a fast handoff management scheme called MeshScan to provide a novel use of channel scanning latency, by employing open system authentication in both Passive Handoff and Active Handoff. This scheme comprises three steps: firstly a client device takes advantage of the Wireless Mesh Network (WMN) architecture to maintain a list of active MNs. Secondly MeshScan Handoff Sensor performs handoff when it receives a disassociation management frame from the serving MN or when the measured signal strength from the serving MN exceeds a given threshold. Thirdly when handoff is required, a client transmits Authentication Request frames to all MNs from the list instead of broadcasting Probe Request frames, as in an active scan to discover the available MNs. The handoff delay is used as criteria for system performance. Numerical results are presented to demonstrate the feasibility of MeshScan with Active Handoff algorithm. This fast handoff scheme is feasible by upgrading the software only on the client side. This paper compares the theoretical handoff latency of MeshScan with other approaches and we demonstrate the effectiveness of our scheme through experiment.**

***Keywords-component; Wireless Mesh Network; Fast Handoff; Active Handoff***

## I. INTRODUCTION

A traditional wireless network deployment involves Access Points (AP) with overlapping coverage zones where each AP has a wired network connection. Wireless mesh networks (WMNs) also consist of APs or more correctly mesh nodes (MNs) where only a few of the MNs require a wired network connection. Packets are forwarded to these wired MNs through multiple hops.

A practical problem with WMNs occurs when a connection transition (i.e. handoff) from one MN to another MN is required by a mobile client to maintain network connectivity. For instance when a mobile client moves away from one MN and closer to another one. Ideally, handoff should be completely transparent to a mobile client to support real-time traffic such as interactive Voice over IP (VoIP) or video conferencing. The handoff procedure aims to reduce this time as much as possible so that the upper layers do not notice the connectivity interruption. However, under the IEEE 802.11 WLAN standard, there are three steps involved in the handoff process: Discovery, Authentication and Re-association. Previous work has reported that the standard handoff incurs latency of the order of hundreds of milliseconds to several seconds. Moreover, the discovery step accounts for more than 90% of this latency [1].

Other important issues in handoff are when handoff should be performed and which MN should the client associate with? If the client waits too long to look for new AP then the client may incur a connectivity interruption. If the client is too eager then it may ping-pong between APs needlessly causing network overload.

In this paper, we present a practical fast handoff management scheme called MeshScan, to manage when handoff should be performed and which MN that the client should associate with. MeshScan can reduce the latency associated with handoff by using open system authentication where no key exchange is involved. The fast handoff management scheme uses a client-side control mechanism which requires a client software upgrade. An experimental analysis of MeshScan was performed on a wireless mesh testbed which uses open system key authentication. All measurements are taken from the system kernel layer to ensure the greatest accuracy. The basic idea behind MeshScan is to take advantage of the WMN architecture where all the MNs are required to cache a list of MNs at the client side. MeshScan determines when handoff is required and triggers handoff by exploiting multiple *Authentication Request* frames to find the next MN within the same mesh network. In this paper, we compare the handoff time required by MeshScan with other approaches and demonstrate the effectiveness of our system through experiment.

The rest of this paper is organized as follows. In Section II, we describe the link-layer handoff procedure in IEEE 802.11 wireless networks and present a discovery latency analysis. Section III introduces the experimental testbed, wireless interface driver and describes our experimental method to improve the efficiency of discovery. Section IV presents the details of our implementation and the experimental results. Section V concludes the paper.

## II. BACKGROUND

### A. Link-layer Handoff

Link-layer handoff refers to the change of MN to which a client is connected in a WMN. In the case of IEEE 802.11 WLANs it implies an interruption of data frame transmission. The duration of this interruption is called handoff latency. For

the purposes of this work we divide handoff into two phases: Scanning and Execution.

The scanning phase is used to acquire information about the available APs in each channel. In the IEEE 802.11 standard, there are two methods used: passive scanning and active scanning. In passive scanning, a mobile client listens for beacon frames on one channel at a time. Beacon frames are normally broadcast by a MN every 100ms. In active scanning, the mobile client broadcasts probe request frames and waits for probe response frames on each channel.

The execution phase is the phase where the mobile client exchanges information and establishes a physical connection with the MN. It involves the processes of authentication and re-association and causes four Round Trip Time (RTTs) latency. Authentication is used to verify the identity between the client and MN. The standard defines two algorithms: open system authentication and shared key authentication. The time required for authentication takes two RTTs for open system and four RTTs for shared key. Re-association follows after successful authentication where the client is assigned a proper association identity and the required resources by the new MN. The re-association delay takes two RTTs. RTT is the time corresponding to the transmission time of a probe request frame and an ACK response frame between two nodes. Four timestamps are required to calculate the RTT using equation (1).

$$RTT = (T_{21} - T_{11}) + (T_{22} - T_{12}) \qquad (1)$$

In this paper we assume $(T_{21} - T_{11}) = (T_{22} - T_{12})$ as shown in Figure 1. $T_{11}$ is the timestamp of the probe request frame that is transmitted from Node A, $T_{21}$ is the time that the request frame from Node A is received by Node B, $T_{22}$ and $T_{21}$ are similar to $T_{11}$ and $T_{21}$. RTT depends on a number of factors that includes the network load, interference and contention.

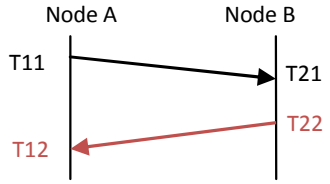


Figure 1. Round Trip Time

### B. *Discovery Latency Analysis*

The latency that arises in passive scanning is significant because the mobile client must listen to each channel of the physical medium in turn, in an attempt to determine the next MN to associate with. The latency that arises in active scanning depends on two parameters: MinChannelTime and MaxChannelTime. Both of these are measured in steps of 1024 microseconds which is called a Time Unit (TU). They control the duration of scanning in each channel where MinChannelTime defines the minimum time required to scan one channel and MaxChannelTime defines the maximum time to scan a channel. The IEEE standard does not specify their values. Typical values are indicated below

- MinChannelTime = DIFS + (aCWmin * aSlotTime). Since MinChannelTime is defined in TU, MinChannelTime will be 1 TU. DIFS = 50µsec, aCWmin = 31µsec and aSlotTime which is defined in the standard to be 20µsec in 802.11 b/g and 9µs in 802.11a.
- MaxChannelTime = 15 msec

There are also other hardware latencies that should be taken into account, e.g. the switching time between channels and the interface setup time. These delays are not considered in the analysis because they vary from manufacturer to manufacturer. Therefore, they have been omitted in this analysis.

### C. *Passive Handoff and Active Handoff*

From the mobile client's perspective, handoff can be categorized into two types: passive handoff and active handoff. In passive handoff, the client does not have control over when handoff will be performed. In active handoff, the client does have control over when handoff will be performed. Handoff performance can be improved by using active handoff in the case of poor signal strength and poor throughput.

### D. *Related work*

Extensive work has been conducted to reduce the handoff latency under IEEE802.11 wireless networks. One such commercial mesh network is Metricom's Ricochet network [2] from the mid-90s. Ricochet nodes automatically route mobile client traffic through half-duplex wireless hops until it reaches a wireline connection. Another well known modern mesh network is the MIT Roofnet project [3], [4] which uses dynamic routing based on link quality measures. Roofnet's emphasis is more on route maintainability and optimization than on handing off a client's connection from one MN to another. Other community and commercial mesh network implementations also exist [5][6][7][8], but none of them provides transparent fast handoff. Most of them use routing protocols on the mesh nodes to trigger and manage the handoff for mobile clients. This in turn introduces considerable overhead into the wireless medium which will degrade the overall throughput of the network.

Mishra, Shin and Arbaugh [9] have analyzed the handoff performance in current 802.11 hardware and have found that approximately 90% of handoff latency is attributable to the scanning phase which is used to locate the next MN. Their experiments have also shown that the actual handoff delay varies according to the wireless network interface card and driver used.

Ramani and Savage [10] introduced the SyncScan method which uses a fast scanning mechanism to listen to all APs in range to choose the best one. This method achieved an impressive handoff time as low as 5ms. A similar approach like a shared beacon channel was introduced in [11] and multiple network interface cards (NICs) were utilized in [12]. These hardware approaches have a deployment difficulty with regard to overhead and power consumption concerns. Our approach provides a highly WMN focused handoff scheme which only requires a software update at the client. Furthermore, our

approach can achieve a handoff delay as low as 1.8ms and can operate under background traffic loads of up to 20 Mbps.

## III. SYSYTEM DESCRIPTION

### A. Testbed Description

All experiments have been carried out using the CNRI wireless mesh testbed [13]. This testbed is a multi-purpose experimental networking platform which consists of 17 IEEE 802.11abg based mesh nodes, located around the Focas building in the Dublin Institute of Technology. Each MN is utilises a Soekris net 4521 platform and a NETGEAR WAG511 wireless adapter card. Each MN runs under Pebble Linux and uses the madwifi version 0.9.4 as the wireless network interface driver. Further information about the CNRI mesh testbed can be obtained from http://mesh.cnri.dit.ie.

### B. MeshScan

MeshScan comprises two steps: First the mobile client is given a list of available mesh node information called a SmartList. Secondly, when handoff is required, the mobile client performs a unicast scan by transmitting *Authentication Request* frames to the each of the MNs on the list to discover the next MN for handoff.

The SmartList is where the MN information stores and manages the MNs. The list is ordered where a MNs position on the list depends on its Received Signal Strength Indications (RSSI) value. The MN with the highest RSSI value will be put at the top of the list in order to provide fast handoff to the best available MN.

The MN information can be easily added to and stored on mobile client when a mobile client joins a particular WMN for the first time. The MNs RSSI is provided in real-time by listening to beacon frames from all MNs. MeshScan does not generate any overhead during handoff and so does not produce any communication performance degradation, nor does it require any modifications to the existing protocols. Furthermore, there is no hardware upgrade required.

### C. Handoff Triggerring

Handoff in MeshScan can be divided into two groups: passive handoff and active handoff. Passive handoff is performed when a client receives a *disassociation management frame* or 10 consecutive *beacon frames* from a MN fail to be received [14]. In active handoff, the RSSI is used as an indicator to trigger handoff in MeshScan. An Exponential Moving Average (EMA) filter is used to obtain an average RSSI value by mitigating the impacts of interference and channel fading etc. Here *E_RSSI* is the average of RSSI over time period of *T*, with the smoothing factor α set to 0.3 in our system.

$$E\_RSSI_T = \alpha \times RSSI_T + (1-\alpha) E\_RSSI_{T-1} \qquad (2)$$

### D. Handoff Procedure

In our system, the handoff procedure is performed with the following steps. When handoff is performed the mobile client transmits an *Authentication Request* to each of the MNs on the SmartList. This is in accordance with the 802.11 standard which allows for authentication with multiple MNs. When the first *Authentication Response* frame is received, the mobile client stops transmitting *Authentication Request* frames to the rest of the MNs on the SmartList and re-associates with the MN which sent the first *Authentication Response.* In the case where no *Authentication Response* is received after all *Authentication Requests* have been transmitted to the MNs in the SmartList, the mobile client will perform active scanning to try to discover if any wireless networks are available. The MeshScan algorithm is shown in Figure 2.

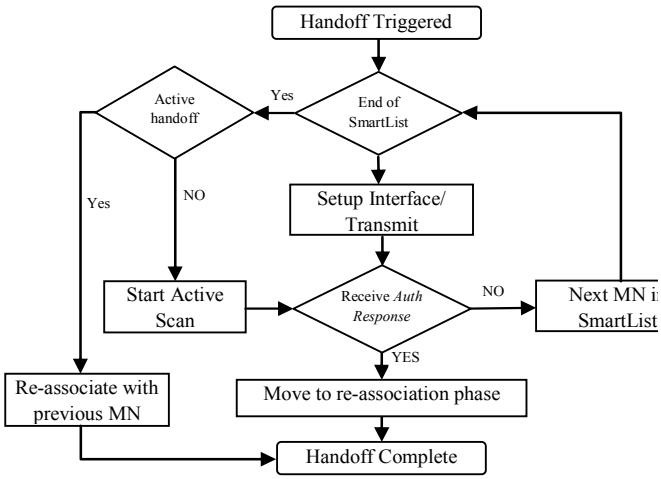


Figure 2. MeshScan Algotirhm

TABLE I. PASSIVE AND ACTIVE HANDOFF LATENCY

| | **Passive Handoff** | **Active Handoff** |
|---|---|---|
| Handoff Triggering Latency | $RRT$ | $\frac{1}{2}RRT$ |
| Authentication Latency | $(m+1)\times(\frac{1}{2}RRT) + RRT$ | $(m+1)\times(\frac{1}{2}RRT) + RRT$ |
| Association Latency | $2RRT$ | $2RRT$ |

In terms of the algorithmic delay associated with MeshScan and assuming at least one MN is available. Table 1 shows how we measure handoff latency and the latency of each step in both passive and active handoff. Equation (3) applies to passive handoff and equation (4) applies to active handoff, where *M* is the number of *Authentication Request* frames transmitted. In the best case scenario the first MN from the SmartList is the next MN to re-associate with, so the delay is $4\frac{1}{2}$ * RTT in

Passive Handoff and 5*RTT in Active Handoff. The worst case will be there is no available MN and the mobile client must carry out active scanning.

$$(M+1)\times(\frac{1}{2}RTT)+3\frac{1}{2}RTT+(M-1)\times MinChannelTime \quad (3)$$

$$(M+1)\times(\frac{1}{2}RTT)+4RTT+(M-1)\times MinChannelTime \quad (4)$$

## IV. IMPLEMENTATION AND EXPERIMENTS

### A. Implementation Details

We implemented the SmartList within the kernel driver (madwifi 0.9.4) [15]. The changes in the madwifi driver are the minimum required to support MeshScan by using SmartLis and active handoff by using a RSSI filter. All processes are carried out in kernel layer to provide stable and fast handoff in this prototype implementation.

In the madwifi driver, a new structure has been implemented to create single linked list to serve as the SmartList. A command line input method (ioctls) has also been implemented to provide a flexible way to add MN information We created a new information state to trigger SmartList to reorder the MN positions on the list. A beacon frame RSSI filter was added to provide the trigger for the active handoff mechanism The RSSI threshold has been set to 19 which indicates a poor signal [16]. The management frame retry and management frame backoff count are minimized to provide fast *Authentication Request* transmission. In the original madwifi driver, the same *Authentication Request* will be retransmitted 11 times if an *Authentication Request* fails to receive a response from the MN where it waits for 1 second before transmitting the next *Authentication Request*. These can cause significant delay so the management frame retry limit has been set to zero and management frame backoff count is set to one millisecond when fast handoff is performed.

### B. Experimental Testbed Setup

We have implemented a prototype of a MeshScan client on a Linux platform (Pentium(R) Dual-Core CPU E5200 2.5GHz, 1GB RAM) with an Atheros AR5212-based wireless interface. It runs Fedora 10 with a 2.6.27.24-170.2.68 kernel and the modified madwifi code as the wireless interface driver.

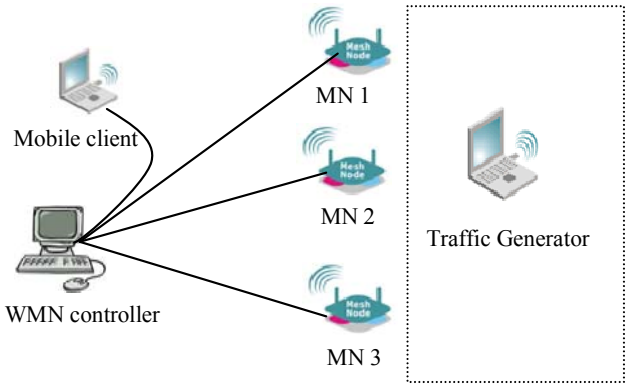


Figure 3. Experimental Setup

We evaluated the performance of our prototype by measuring handoff delay across ESSs with the same ESSID. All MNs operate using 802.11a, channel 60 and use open system key authentication. The testbed is shown in Fig.3. The mobile client and three MNs have fixed positions to allow repeatable experimental work. The WMN controller uses SSH to control mobile client and three MNs. An automated script runs on WMN controller to force mobile client handoff among three MNs by turning the MN interfaces on and off for passive handoff and by adjusting MN interfaces' transmit power for active handoff. We use the D-ITG [16] tool to generate background traffic.

### C. Experimental Results

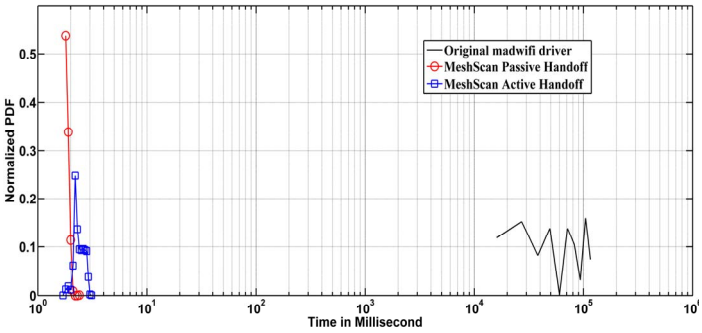


Figure 4. Handoff Latency for MeshScan and Madwifi driver

Fig.4 shows handoff latency for the original madwifi, MeshScan Passive Handoff and MeshScan Active Handoff. The x-axis shows the time in millisecond and the y-axis shows the normalized frequency of handoff latency. From the figure it can be seen that the handoff latency in the original madwifi driver appear to be random and widely distributed from $10^4$ ms to over $10^5$ ms which does not provide fast handoff. It can also be seen that the handoff latency associated with our MeshScan approach decreases dramatically both in Passive and Active Handoffs where the lowest handoff was just 1.8ms in both Passive and Active Handoff. In most of the cases, handoff latencies were between 1.8ms to 3ms by using MeshScan.

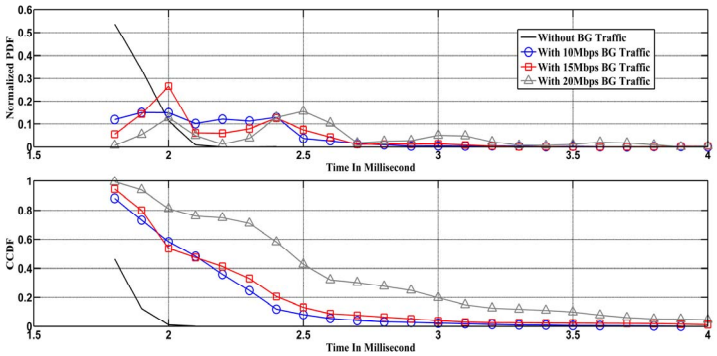


Figure 5. MeshScan Passive Handoff Latency in Different Background Traffic Load

In Fig.5, we have compared the latency of MeshScan Passive Handoff under different background loads of 10Mbps 15Mbps and 20Mbps. From PDF chart, it can be observed that

when the background traffic load was up to 15Mbps, over 97% of the handoff was completed within 3ms and 98% of the handoff finished in 4ms when background traffic load was 20Mbps. In the CCDF below, we show that the average handoff latency increases according to background traffic as expected. The increase in handoff latency when background traffic load was added is because the RTT was increased due to network interference and traffic load.

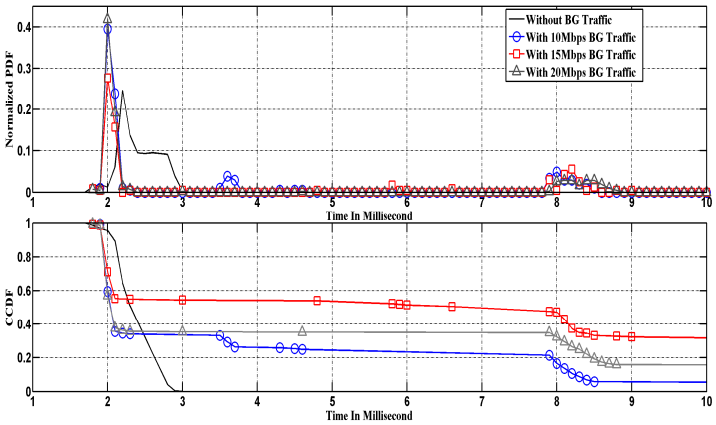


Figure 6. MeshScan Active Handoff Latency in Different Background Traffic Load

In Fig.6, we have compared the latency of MeshScan Active Handoff under different background loads of 10Mbps 15Mbps and 20Mbps. The PDF of handoff latency shows that the probability of low latency around 2ms is higher when we introduce background traffic, compared to when no background traffic is present (2ms< latency <3ms). Furthermore, the probability of low latency increases as we increase the background traffic. There is a small surge at 8ms which indicates the successful retry of handoff. This can also be clearly seen in the CCDF in Fig.6. The CCDF also shows a stepped characteristic in the background latency curves. This is due to the occurrence of retries when active handoff fails.

If we compare between Fig.5 and Fig.6, it can be seen that the average handoff latency in passive handoff was less than active handoff. Furthermore, passive handoff was more stable than active handoff. This is due to handoff triggering time (computing time) being measured in Active Handoff at the client side, but there was no handoff triggering time in passive handoff at client side by its nature.

## V. CONCLUSION

We have developed a fast handoff management scheme, called MeshScan which makes decisions when to perform handoff and which MN to associate with in order to improve handoff within a WMN in both Passive and Active Handoff. MeshScan uses a novel usage of the open system authentication scan to reduce channel scanning latency. MeshScan maintains a list of MNs in SmartList and performs unicast scanning by transmitting authentication request frames to discover available MN. It then performs handoff instead of broadcasting probe request frames. In this paper we have shown a significant reduction in handoff latency in our approach through implementation and experiments.

## ACKNOWLEDGMENT

This work was conducted with the support of Science Foundation Ireland under Grant No. RFP/CMS704.

## REFERENCES

[1] A. Mishra, et al.: "An Empirical Analysis of the IEEE 802.11 MAC Layer Handoff Process, " ACM SIGCOMM Computer Communication Review,

[2] Diane Tang and Mary Baker, "Analysis of a Metropolitan-Area Wireless Network," ACM/Kluwer Wireless Networks. Special issue: Selected Papers from Mobicom'99, vol. 8, no. 2/3, pp. 107-120, 2002.

[3] B. Chambers, "The grid roofnet: a rooftop ad hocwireless network," 2002. h

[4] John C. Bicket, Daniel Aguayo, Sanjit Biswas, and Robert Morris, "Architecture and evaluation of an unplanned 802.11b mesh network.," in MOBICOM, 2005, pp. 31-42.

[5] "Locusworld," http://locustworld.com

[6] "Tropos networks," http://www.tropos.com

[7] "Extricom," http://www.extricom.com

[8] "Cisco," http://www.cisco.com/en/

[9] Arunesh Mishra, Minho Shin, and William Arbaugh, "An empirical analysis of the IEEE 802.11 MAC layer hando_ process," SIGCOMM Comput. Commun. Rev., vol. 33, no. 2, pp. 93-102, 2003.

[10] Ishwar Ramani and Stefan Savage, "Syncscan: Practical Fast Hando_ for 802.11 Infrastructure Networks," in Proc. of IEEE INFOCOM, march 2005.M.

[11] V. Brik, et al.: "Eliminating handoff latencies in 802.11 WLANs using multiple radios: Applications, experience, and evaluation, " ACM/USENIX IMC 2005, USA, October 2005.

[12] M. Jeong, et al.: "Fast active scan for measurement and handoff, "Technical report, DoCoMo USA Labs, May 2003.

[13] CNRI Wireless Mesh Testbed. Available at http://www.cnri.dit.ie/research.mesh.testbed.html

[14] Madwifi Driver Summary http://mesh.calit2.net/whzhao/madwifi_summary.pdf

[15] "MadWifi - a Linux kernel device driver for Wireless LAN chipsets from Atheros. " [Online]. Available: http://madwifi.org

[16] D-ITG Tool http://www.grid.unina.it/software/ITG/