2009-10-01

MeshScan: Fast and Efficient Handoff in IEEE802.11 Mesh Networks

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

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

Recommended Citation

Chen, Y. & Davis, M. (2009) MeshScan: Fast and Efficient Handoff in IEEE802.11 Mesh Networks.The *7th ACM International Symposium on Mobility Management and Wireless Access - MobiWAC* 2009, Tenerife, Canary Islands, Spain. 2009.

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

MeshScan: Fast and Efficient Handoff in IEEE802.11 Mesh Networks

Yin Chen
Communications Network Research Institute,
Dublin Institute of Technology
Ireland
yin.chen@cnri.dit.ie

Karol Kowalik
Communications Network Research Institute,
Dublin Institute of Technology
Ireland
karol.kowalik@cnri.dit.ie

Mark Davis
Communications Network Research Institute,
Dublin Institute of Technology
Ireland
mark.davis@dit.ie

ABSTRACT

Handoff delay is one of the major problems in Wireless Mesh Network (WMN) that needs to be solved in order to allow time-critical and real-time applications run continuously during handoff. We have developed a fast handoff scheme called MeshScan to provide a novel use of channel scanning latency by employing open system authentication. This scheme comprises two steps: firstly a client device takes advantage of the WMN architecture to maintain a list of active mesh nodes. Secondly when handoff is required, a client transmits Authentication Request frames to all mesh nodes (MNs) from the list instead of broadcasting Probe Request frames as in an active scan to discover the available MNs. 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

Wireless Mesh Network; Fast Handoff;.

1. INTRODUCTION

A traditional wireless network deployment involves Access Points (AP) with overlapping coverage zones where each AP has a wired network connection. WMNs only require a few of the MNs to have wired network connections and allowing all others to forward packets over multiple wireless hops. In this paper, we are concerned with an IEEE 802.11a network which operates in the 5.3 GHz frequency band.

A practical problem with WMNs occurs when a connection transition (handoff) from one MN to another MN is required for mobile client to maintain network connectivity when a mobile client moves away from one MN and closer to another one. Ideally, handoff should be completely transparent to mobile client to support real-time traffic such as interactive 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 works reported that the standard handoff incurs a latency of the order of hundreds of milliseconds to several seconds. Moreover, the discovery step accounts for more than 90% of this latency [1].

In this paper, we present an experimental fast handoff scheme called MeshScan to reduce the latency associated with node by using open system authentication where no key exchange is involved. The fast handoff scheme uses a client-side control mechanism which requires a client software upgrade. The experiments are performed on an wireless mesh testbed which uses open system key authentication. All measurements are taken from the system kernel layer to ensure 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 mesh node at client side, and exploit multiple *Authentication Request* frames to find the next mesh node within same mesh network. In this paper, we compare the theoretical 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 on Linux box and experimental results. Section V concludes the paper and outlines our future work in this area.

2. BACKGROUND

2.1 Link Layer Handoff

Link-layer handoff refers to the change of the mesh node to which a station is connected in a WMN. In the case of IEEE 802.11a 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 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 when the mobile client exchanges information and establishes a physical connection with

the MN. It involves authentication and re-association. Authentication verifies the identity between client and MN. The standard defines two algorithms: open system authentication and shared key authentication. The time required for authentication takes one round trip time (RTT) for open system or twice RTT for shared key. Re-association follows after successful authentication where the client is assigned a proper association identity and required resources by new MN. The re-association delay takes one RTT.

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 RTT using equation (1):

$$RTT = (T21 - T11) + (T22 - T12) \quad (1)$$

In this paper we assume as shown in Figure 1. (T11 is the timestamp of the probe request frame that is transmitted from Node A, T21 is the times that the request frame from Node A is received by Node B, T22 and T21 are similar to T11 and T21) RTT depends on a number of factors that includes the network load, interference and contention.

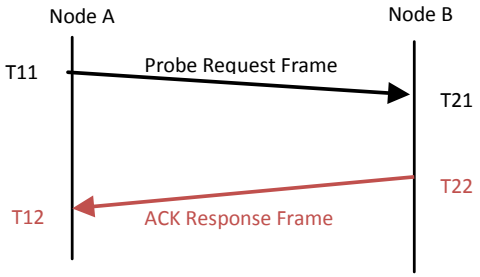

Fig. 1: Round Trip Time

2.2 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 locate 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 to scan one channel and MaxChannelTime defines the maximum time to scan one channel, the IEEE standard does not specify their values. Typical values are indicated below

1) 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.

2) 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, we are ignoring them in this analysis.

2.3 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 reaching 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 which introduces considerable overhead into the wireless medium which will degrades the overall throughput of the network.

Mishra, Shin and Arbaugh[9] have analyzed the handoff performance in current 802.11 hardware where approximately 90% of handoff latency is attributable to the scanning phase which is used to locate the next MN. Their experiments 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 as low as 5ms. Similar approach like shared beacon channel is introduced in [11] and multiple network interface cards (NICs) are 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 and can achieve a handoff delay as low as 1.8ms and can operate under background traffic loads of up to 20 Mbps.

3. SYSTEM DESCRIPTION

3.1 Testbed Description

All experiments have been carried out using the CNRI wireless mesh testbed [13]. This testbed is a multi-purpose networking experimental platform which consists of 17 IEEE 802.11abg based mesh nodes, located around the Focus building in DIT. Each Mesh Node uses Soekris net 4521 boards as hardware platform and NETGEAR WAG511 wireless adapter cards. It runs under the Pebble Linux distribution as software platform and uses 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.

3.2 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 to.

SmartList is where the MN information stores and manages the MNs. The list is ordered where a MN's 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 in real-time.

The MN information can be easily added to and stored onto mobile client when a mobile client joins a particular WMN for the first time. The MN's RSSI is provided in real-time by listening to beacon frame from all Mesh Nodes. MeshScan does not generate any overhead during handoff and so does not produce any communication performance degradation, nor require modifications to the protocols. There is no hardware upgrade required.

3.3 Handoff Procedure

In our system, the handoff procedure is performed with the following steps. When handoff is required 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, mobile client stops transmitting Authentication Request to the rest of the MNs on the SmartList and re-associates with the MN which sends the first Authentication Response. In the case where no Authentication Response is received after Authentication Requests have been transmited to all MNs in the SmartList, the mobile client will perform active scanning to try to discover any available wireless networks. The MeshScan algorithm is shown in Figure 2.

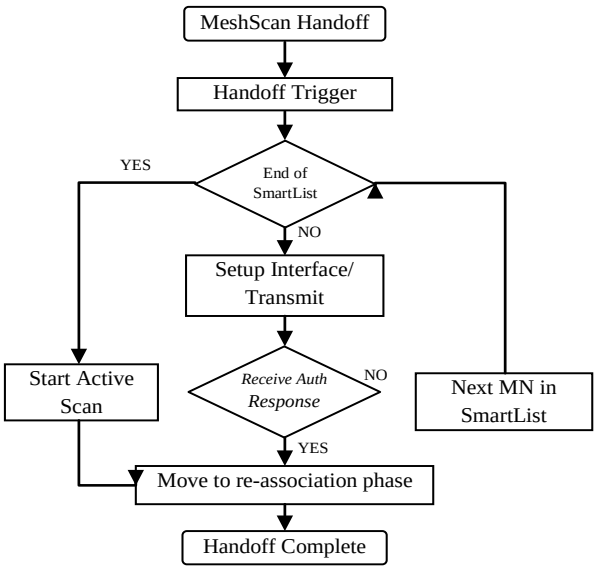

Fig.2 MeshScan Algorithm

In terms of the MeshScan algorithmic delay and assuming at least one MN is available, we have equation (2) 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 2 * RTT. The worst case will be there is no available MN and the mobile client carries out an active scanning.

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

4. IMPLEMENTION AND EXPERIMENT

4.1 Implementation Detail

We implemented the SmartList within the kernel driver (madwifi 0.9.4) [14]. The changes in madwifi driver are the minimum required to support MeshScan by using SmartList. 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 is implemented to create single linked list as SmartList. A command line input method (ioctls) is also implemented to provide a flexible way to add Mesh Node information. We created a new information state to trigger SmartList to reorder the MN position to make the best MN on top of SmartList. 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 does not get response from MN where it waits for 1 second before transmitting the next Authentication Request. They can cause significant delay so the management frame retry is set to zero and management frame backoff count is set to one millisecond when a fast handoff is triggered.

4.2 Experimental Testbed Setup

We have implemented a prototype of a MeshScan client on a Linux Box (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 modified madwifi as the wireless interface driver.

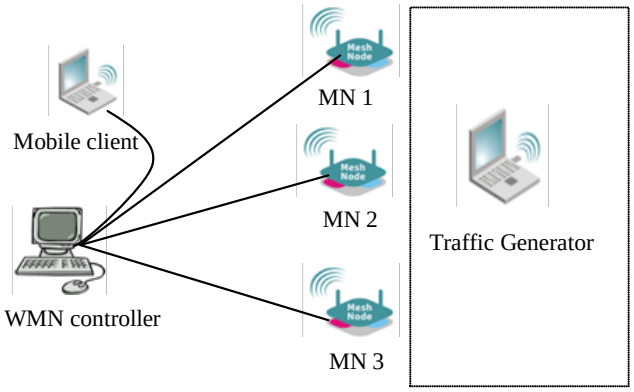

Fig. 3 Experimental Setup

We evaluated the performance of our prototype by measuring handoff delay across ESSs with the same ESSID, all MNs operate in 802.11a, channel 60 and use open system key authentication. The testbed is shown in figure 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 turn the MN interfaces on and off to force mobile client handoff among three MNs. We use the D-ITG tool [15] to generate background traffic.

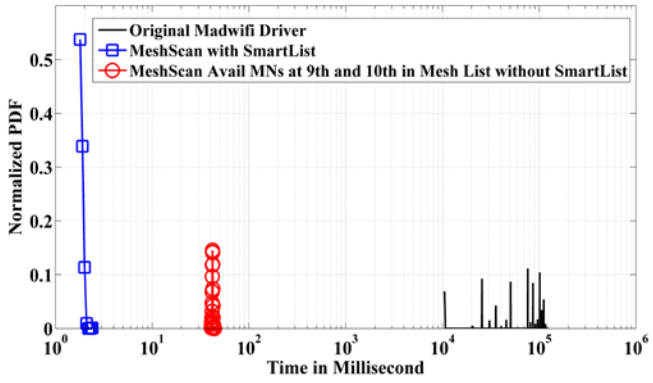

Fig.4 Handoff Latency for MeshScan and Madwifi driver

4.3 Experimental Results

Figure 4 shows handoff latency for the original madwifi, MeshScan without SmartList while available MNs were at 9th and 10th position in Mesh List, and MeshScan with SmartList. 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 and is not suitable for time critical application like VoIP during the handoff. It was observed that MeshScan could store up to 10 MNs in the MN list to provide fast handoff within 50ms without SmartList being used. It can also be seen that the handoff latency associated with our MeshScan approach decreases dramatically where the lowest handoff was just 1.8ms. In most of the cases, handoff latencies were between 1.8ms to 2.4ms by using MeshScan.

In Figure 5, we have compared the handoff latency of MeshScan under different background traffic loads of 10Mbps, 15Mbps and 20Mbps. It can be observed that when 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. The increase in handoff latency when background traffic load was added is because the RTT was increased due to network interference and traffic load.

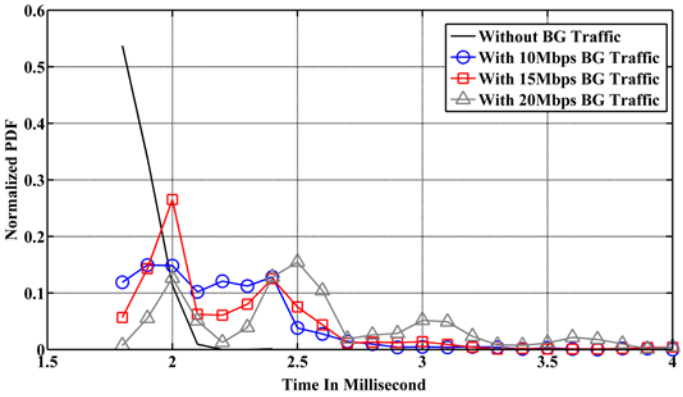

Fig. 5 MeshScan Handoff Latency in different Background Traffic Load

5. CONCLUSION AND FUTRE WORK

We have developed a fast handoff scheme, called MeshScan to improve handoff within WMN, by using 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 frame to discover available MN and perform handoff instead of broadcasting probe request frame.

In this paper we have shown a significant reduction in handoff latency of our approach through implementation and experiments. Our future work will be concerned with implementing real-time list of MNs at the MN side, and the MN provides accurate mesh node list and management method to reduce the number of MNs to check when using fast handoff.

6. ACKNOWLEDGMENTS

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

7. 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 - a Linux kernel device driver for Wireless LAN chipsets from Atheros. http://www.madwifi-project.org