

Simulation of Handoff in Wireless Wi-Fi Networks

*Chris J. Friesen
Stetson University
November 26, 2003*

Abstract

Since its debut in the late 1990s, wireless technology has become as much a necessity as the wired technology. Each new iteration of wireless technology brings with it new features like speed increases and security enhancements. The technology is at the point where it is being used for real time applications, like video streaming and voice chat. As these applications develop, a problem has risen. On a wireless network, it is not acceptable to require a roaming client to stay within the limited range of a single access point (AP) and the delays incurred from jumping between APs causes issues when the client is partaking in a real time conversation or chat. In this paper, a way of improving mobility of wireless clients will be researched. The focus will be on the building of a wireless simulator with special attention on the connection of a client to an access point. The process by which clients communicate to the AP will also be stressed. Next, an explanation of how clients can hop between APs will be explained along with an implementation of how to keep this hop as clean and errorless as possible, on a TCP network. The simulation will be written in a Linux environment using the standard C++ networking libraries.

I. Introduction

Wireless technology has grown in popularity exponentially. The development of faster wireless technology is empowering users to do the things they were only permitted to while sitting at a terminal on a wired network. Today, the technology is good, but it is not perfect. There are still many issues facing the wireless community that need to be worked out, like better security and handling of error prone networks. Another larger problem is the use of TCP to communicate between the clients and their access points (AP). TCP was designed with specific requirements to prevent faults in the connecting, sending, receiving, and disconnecting processes of a client. There is a lot of overhead in these requirements which can be a burden to a wireless client who might be on a slow link, is trying to move between different APs, or trying to utilize the network for real time applications. In order to use real time applications like voice communication or data streaming, the rate of error must be kept at a minimum, but there must be a steady rate of information sent at all times. On a wired network, this is not a large issue, because the line speed is generally high. With a wireless network, the speed of the medium is dependent on factors such as signal strength and the number and type of clients trying to use the medium. This can cause the line speed to fluctuate or just be slow. Also, the more clients that are connected to a given AP, the harder it is for the medium to keep up with all of the clients and for each client to get fair use of the medium.

II. Background

One of the main issues with wireless mobility is the latency involved when a client moves out of the range of one AP and into the range of another, called a handoff [1]. Handoff can cause jittering in the connection which interrupts the stream of data being sent to the client.

A. Process

The handoff process involves two steps:

1. Discovery – this is when the client scans the network by looking for the beacon messages that each access point broadcasts. Also, the client can build a priority list of APs, depending on its signal strength.
2. Reauthentication – Using the priority list built during discovery, the client is synchronizing itself with the best AP in its list. It involves having authorization credentials and state information sent from the original AP to the new one.

B. Delays

The two steps required for a successful handoff introduce latency issues. These issues are as follows:

- Probe Delay – this is the amount of time it takes the client to complete a scan of available networks and to build its priority list. It is required to send somewhere between 3 to 11 messages in order to complete this task.
- Authentication Delay – this is the amount of time it takes for the client to reauthenticate to the AP it chose from its priority list. Depending on the type of authentication, either 2 or 4 packets need to be exchanged.
- Reassociation Delay – this is the amount of time it takes for the client to signal the AP that the handoff is complete. It is required that a minimum of 2 packets be exchanged.

III. Related Work

Much research has been done in attempts to improving the handoff procedure for wireless Asynchronous Transfer Mode Networks (WATM-N) [2]. ATM networks are designed to provide high speed communication for roaming clients, which makes it a good candidate to use when real time applications are involved. Current ATM handoff optimization techniques are broken up into four categories [6]:

- connection reestablishment – requires that the client establishes a new connection each time a handoff is required.
- route argumentation – extends the clients current connection to the new AP.
- partial rerouting – searches for the shortest route between APs.
- connection prediction – attempt to predict where the client will go.

Each of the above techniques has its disadvantages. In connection reestablishment, the amount of time it takes for a client to recreate a connection is too high and becomes a waste of network resources. Sangheon Pack and Yanghee Choi [5] found a possible solution by

requiring a roaming client to be authenticated to multiple APs around it, based on patterns in its movement. Route argumentation has the potential to result in long looping routes. One way to overcome this is to use a partial rerouting algorithm to compute the optimal route between two APs. Although it can provide better performance, it requires resources to compute the route. One such algorithm that has been researched is called Nearest Common Node Rerouting (NCNR) [2]. It attempts to reroute a client using the least amount of bandwidth by eliminating connections that are not necessary. This is calculated in as efficient manner as possible, saving time and resources. Li-Yun Chiang [4] has also solved the routing issue by using vectors to represent the shortest path between APs. The final technique, connection prediction, requires that a tree of the network be maintained and searched when a handoff is required. This creates much more overhead than is acceptable [6].

IV. Implementation

As stated above, NCNR has been tested on a wireless ATM network, but it has not been tested on TCP networks. In order to accurately test the effects of NCNR on a TCP network, a good simulation will be needed. Fortunately, not the whole 802.11 specification is required, only the connection procedure and the process by which the clients communicate with the access points they are connected to, called Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA).

A. Connection

In order for a client to connect to an AP, a simple handshaking process must be completed. To initiate the connection, the client wishing to connect sends a request to the AP with the following information [3]:

- Authentication Algorithm Identification – this is used to determine if both parts of the network are using the same algorithm to process their information. The two types are *Shared Key* and *Open System*. Shared Key uses Wired Equivalent Security (WEP) and will not be used in this project. Open system is transmitting on an insecure line without the worry of the data being intercepted.
- Station Identity Assertion (SSID) – this is the unique identification of the access point being connected to. This must be present or authentication will fail.
- Authentication transaction sequence number – at this time, this field will be set to 1 to signal the beginning of the process.
- Authentication algorithm dependent information – this section is left blank if Open System authentication is used.

When the AP receives a request, it processes it and sends a message back to the client with the state of its authentication. This message contains the following information:

- Authentication Algorithm Identification – this field has the same function as the first step of authentication process.
- Authentication transaction sequence number – this is set to 2 to signal to the client that it is the second phase of the authentication process.
- Authentication algorithm dependent information – is left blank for the same reasons as above.

- Result of authentication – this is a code sent back to the client specifying the state of its authentication as shown in *Table 1*.

Status Code	Meaning
0	Successful
1	Unspecified Failure
2 – 9	Reserved
10	Cannot support all requested capabilities in the Capability Information field
11	Reassociation denied due to inability to confirm that association exists
12	Association denied due to reason outside the scope of this standard
13	Responding station does not support the specified authentication algorithm
14	Received an Authentication frame with authentication transaction sequence number out of expected sequence
15	Authentication rejected because of challenge failure
16	Authentication rejected due to timeout waiting for next frame in sequence.
17	Association denied because AP is unable to handle additional associated stations
18	Association denied due to requesting station not supporting all of the data rates in the BSSBasicRateSet parameter
19 – 65 535	Reserved

Table 1: a list of the codes that can be sent to a client awaiting authentication

B. Carrier Sense Multiple Access with Collision Avoidance

CSMA/CA is a distributed coordination function (DCF) that is used to check to see if a medium is being used, because a client can not transmit if the medium is being used by another. This process is achieved in two different ways:

1. Physical carrier sense – this information is provided by the physical layer (PHY). The PHY sends messages to the MAC layer with details about the state of the medium.
2. Virtual carrier sense – this uses a network allocation vector (NAV) to maintain predictions of future traffic, based on the duration information transmitted by a client before it begins exchanging data with the AP.

It takes only one of these mechanisms to set the medium as being busy. When they both suggest that the line is free, any connected client is free to attempt to transmit. It is at this point where the most number of collisions will occur, so upon the completion of transmission by one client, each of the other clients choose a random backoff time before attempting to use the medium.

C. Nearest Common Node Rerouting

Handoff, as defined previously, is the procedure of predicting and rerouting packets for a wireless client as it hops between different APs [2]. The originating AP will be labeled **A** and the candidate AP will be labeled **B**.

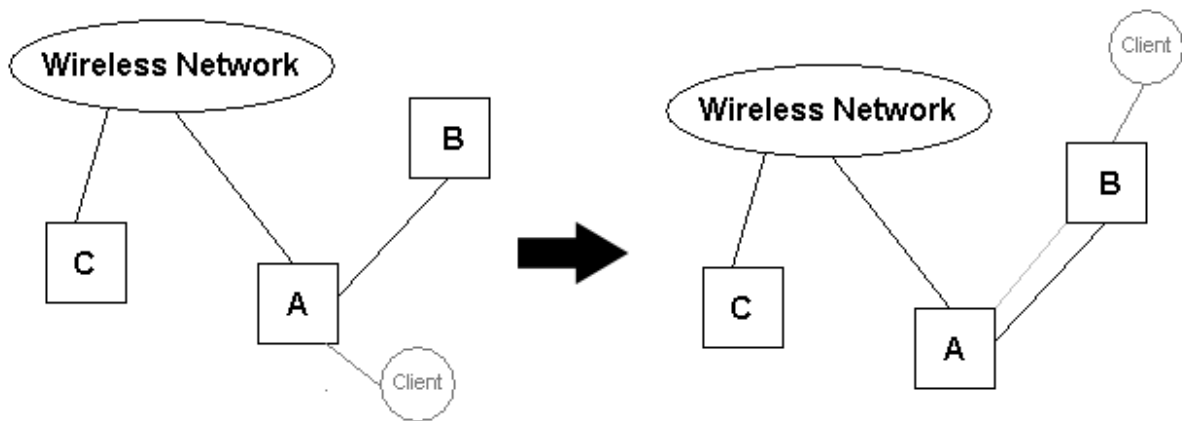


Figure 1: A has a direct link to B

The first step in using the NCNR algorithm is for **A** to check if it has a direct physical connection to **B** (Figure 1). If it does, **A** will check to see if it is the parent of **B**. If it is, it sends a message to **B** telling it of the connection and acts as a relay between the client and **B** until the transmission is stable. If **B** is the parent of **A**, then **A** sends a message to **B** saying a handoff is necessary. **B** is then the anchor between the client and **A** until the transmission is stable. In this state, the client is actually connected to both **A** and **B** and can send and receive messages to and from either one.

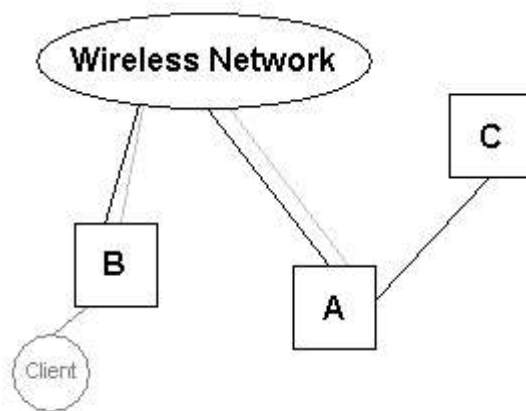


Figure 2: A does not have a direct link to B

If **A** does not have a direct connection to **B** (Figure 2), **A** sends a message to the end point for the user connection by sending it a handoff start message. This message contains the address of node **A**, node **B**, and the location of the client. This message is sent over the network from **A** for each node to check if it can service the handoff request. If another AP can service the request, it flips a bit on the handoff start message, otherwise it leaves it alone and passes it on. This AP is then labeled as the nearest common node (NCN). The NCN sends a reroute message to all APs between **B** and itself to make sure there are resources open for the handoff. If there are, a routing table is built and the handoff takes place. Otherwise, a message is sent to the affected nodes that handoff has failed. Upon receiving the reroute message by **B**, it sends a message to **A** acknowledging that the handoff was successful (Figure 3).

```

if ( A contains a direct link to B )
{
    if( A is the parent of B )
    {
        A notifies B
        A sets up a bridge with B until the handoff is complete
    }
    else if ( B is the parent of A )
    {
        B notifies A that a handoff is necessary
        B sets up a bridge with A until the handoff complete
    }
}
else
{
    A puts a handoff request message out to the network

    if ( some node receives handoff request )
    {
        node checks if it can service the request

        if ( it can service the request )
        {
            becomes labeled the NCN
            the node flips a bit on the message
            NCN sends a reroute message to all nodes between itself and B

            if ( all nodes have available resources )
            {
                built rerouting table
                handoff procedure can continue
            }
            else
            {
                notify involved parties that handoff failed
            }
        }
        else
        {
            pass request onto the next node on the network
        }
    }
}

```

}

Figure 3: Basic NCNR handoff process.

V. Future Work

In this project, we intend to write a simulator using the necessary elements we described in Section IV to accurately simulate a wireless TCP network. Once this is accomplished, implementation and testing of the NCNR algorithm will be done on the network. We will be testing how accurately TCP transmissions between clients and access points can be predicted and rerouted in order to keep the rate of error for the client at a minimum. With enough time, comparisons of the NCNR algorithm against the builtin handoff procedure will be examined. The simulator will be written in a Linux environment using the standard C++ networking libraries.

References

- [1] Arunesh Mishra, Minh Shin, and William Arbaugh. “An Empirical Analysis of the IEEE 802.11 MAC Layer Process”. University of Maryland. 2002. <http://www.cs.umd.edu/%7Ewaa/pubs/handoff-lat-acm.pdf>
- [2] Bora A. Akyol and Donald C. Cox. “Rerouting for Handoff in a Wireless ATM Network”. Cambridge, MA. 1996. <http://wireless.stanford.edu/~akyol/icupc.ps>
- [3] LAN MAN Standards Committee of the IEEE Computer Society. “Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications”. 1999. <http://standards.ieee.org/getieee802/download/802.11-1999.pdf>
- [4] Li-Yun Chiang. “An Efficient Handoff Algorithm in Wireless ATM Networks” 2002. <http://athena.cs.ccu.edu.tw/advisees/thesis/ms89/jly89.pdf>
- [5] Sangheon Pack and Yanghee Choi. “Pre-Authenticated Fast Handoff in a Public Wireless LAN Based on IEEE 802.1x Model” 2002. http://mmlab.snu.ac.kr/research/publication/docs/pwc2002_shpack.pdf
- [6] Sirisha R. Medidi and Forouzan Golshani. “Handoff In Mobile ATM Networks: Optimized Rerouting.” 2002. <http://wireless.netlab.uky.edu/SeminarArchive2001-2002/HandoffMobileATMNetworks.pdf>