

Real-world Implementation of Ad-hoc wireless Networks

Tushar G.Dangat¹, Prasanna J.Shete²

Department of Computer Engineering^{1,2}, Technical Assistant¹, Ast.Professor²

Email: tushar.dangat7@gmail.com¹, prasannashete@somaiya.edu²

ABSTRACT

An ad hoc network consists of a number of wireless hosts that temporarily form a dynamic LAN and are capable of operate without the support of any fixed infrastructure like access point wireless routers etc. effective dynamic routing is a challenge in such a networks. To enable communication between hosts, each host must function as a wireless router and potentially forward data traffic on behalf of others. During the last few years more than 30 different ad hoc routing protocols have been proposed, mostly compared based on simulation studies. The majority ad hoc routing implementation has been done using simulation tools. In simulation the code is contended in single logical component on other hand while implementing real world we required system with many components including many that have little or no documentation. For researcher must understand not only the routing protocol, but all the system components and their complex interactions also. Simulation studies are just complement not getting actual result like real world experiments. In this paper we summarized our effort and experiences to deploy real world wireless ad hoc network infrastructure .we discussed the challenges and survey the field of ad hoc routing and related real world experiment. It is useful to aid researchers for developing on-demand ad hoc routing protocols environment and assist users in determining the implementation design that best fits their needs. In this paper we discuss the socket based mechanism particularly when AODV routing daemon communicates changes to the IP route table.

Index Terms: ad hoc network, AODV, Simulation.

1. INTRODUCTION

Mobile wireless devices are getting rapidly popular due to their efficiency, portability and comfort to use. IEEE 802.11 [3] standard is providing specification for wireless connectivity for fixed, portable and moving stations within a local area. It defines over the air to support networking in local area. It is providing MAC and physical layer functionality. The IEEE 802.11b gives transmission rates up to 11Mbps and operates in the 2.4 GHz band. The IEEE 802.11 standard takes into account of power management, bandwidth, security and addressing, since these are the significant differences from wireless to wired LANs. The MAC layer speciation of the IEEE 802.11 standard provides radio channel access control functions, such as addressing, access coordination etc. The *distributed coordination function (DCF)* is the basic medium access mechanism of IEEE 802.11, and uses a *Carrier senses multiple accesses with collision avoidance (CSMA/CA)* algorithm to mediate the access to the shared medium. The standard also describes centralized, polling-based access mechanism, the *point coordination function (PCF)* which is very rarely used in practice. The DCF protocol in IEEE 802.11 standard defines how the medium is shared among stations. It includes a basic access method and an optional channel access method with *request-to-send (RTS)* and *clear-to-send (CTS)* exchanged as shown in Figure 1 and 2, respectively. First, we explain the basic access method

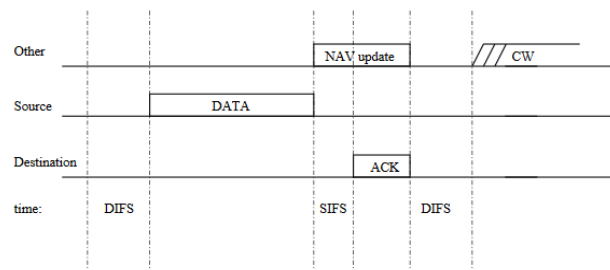


Figure1: Basic access CSMA/CA protocol

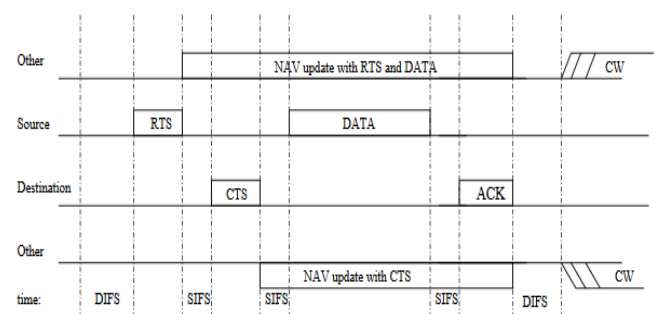


Figure 2: RTS/CTS exchange in the CSMA/CA protocol

If the channel is busy for the source, a backoff time (measured in slot times) is chosen randomly in the interval $[0, CW]$, where CW stands for the *contention window*. This timer is decreased by one as long as the channel is sensed idle for a

DIFS, i.e., *distributed inter-frame space* time. DIFS is equal to $SIFS+2 \times SlotTime$, where SIFS stands for *short inter-frame space* (see values in Table 1). The timer stops when the channel is busy and resumes when the channel is idle again for at least a DIFS period. CW is an integer whose range is determined by the PHY layer characteristics: CW_{min} and CW_{max} . CW is doubled after each unsuccessful transmission, up to the maximum value equal to $CW_{max} + 1$. When the backoff timer reaches zero, the source transmits the data packet. The ACK is transmitted by the receiver immediately after a period of duration equal to SIFS. When a data packet is transmitted, all other stations hearing this transmission adjust their *net allocation vector (NAV)*. The NAV maintains a prediction of future traffic on the medium based on the duration information that is announced in Data frames (or RTS/CTS frames as will be explained in the following) prior to the actual exchange of data. In addition, whenever a node detects an erroneous frame, the node defers its transmission by a fixed duration indicated by EIFS, i.e., *extended inter-frame space* time. This time is equal to the $SIFS+ACKtime+DIFStime$. The contention window is initially set to the minimum value of CW_{min} , equal for example to 15 (see Table 1). Every time a collision occurs, this is interpreted as a high load of the network, and each station involved in the collision throttles down its transmission rate by doubling the size of its contention window. In this way, the contention window can take values equal for example to 31, 63, 127, 255, 511, up to $CW_{max} = 1023$. Larger contention windows slow down the transmission of packets and reduce the probability of collisions. In case of a successful (i.e. collision-free) transmission, the transmitting station brings the value of its contention window back to CW_{min} . The mechanism we have just described is called exponential backoff or binary exponential backoff. If the optional access method is used, an RTS frame should be transmitted by the source and the destination should accept the data transmission by sending a CTS frame prior to the transmission of the actual data packet. Note that stations in the sender's range that hear the RTS packet should update their NAVs and defer their transmissions for the duration specified by the RTS. Nodes that overhear the CTS packet update their NAVs and refrain from transmitting. In this way, the transmission of the data packet and its corresponding ACK can proceed without interference from other nodes (hidden nodes problem). Table 1 shows the important time interval between frames in different standard specification called *inter-frame space (IFS)* [7, 9, and 8]. IEEE 802.11g uses the IFS corresponding to its operating

mode.¹ The slot time is the sum of the Receiver-to-Transmitter turnaround time, MAC processing delay, and **clear channel assessment (CCA)** detect time [3]. The value of slot time for different PHY layer protocols is shown in Table 1.

Table 1: Inter frame space and CW time for different PHY layers.

Parameters	802.11a	802.11b (FH)	802.11b (DS)	802.11b (IR)	802.11b (High Rate)
Slot Time (μs)	9	50	20	8	20
SIFS (μs)	16	28	10	10	10
DIFS (μs)	34	128	50	26	50
EIFS (μs)	92.6	396	364	205 or 193	268 or 364
$CW_{min}(SlotTime)$	15	15	31	63	31
$CW_{max}(SlotTime)$	1023	1023	1023	1023	1023
Physical Data Rate (Mbps)	6 to 54	1 and 2	1 and 2	1 and 2	1, 2, 5.5, and 11

IEEE 802.11 can be used in two deferent operating modes:

- (I) Infrastructure mode (all communication goes through an access point (AP)).
- (II) Ad-Hoc mode (no AP is present; all communication is peer-to-peer).

2. BACKGROUND

A network is a collection of two or more computing devices connected by a communication medium. Figure3 shows a simple network with three computing devices. When a computing device wishes to send information to another device, it may do so by transmitting the information along a shared communication medium.

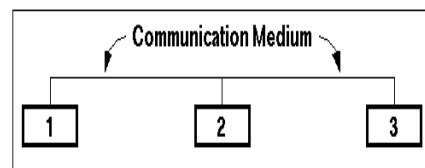


Figure3: Simple network with three nodes.

Any computing device actively participating in a network is called a *node*. Nodes are connected by a communication medium or *link*. Nodes exchange information over links in discrete blocks called *packets*.

In Figure 3, any node can communicate with any other node along the single shared link. In this case, no special steps are needed for any two nodes in the network to exchange information. If, however, nodes in a network do not share a common link, this no longer holds true. Figure4 shows a network where different nodes share different links.

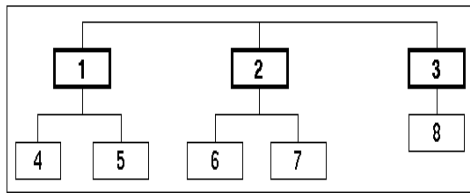


Figure4: Network with eight nodes connected by four separate links.

For example, in Figure4, for node 1 to send a packet to node 8, the packet must first be sent to node 3. Node 3 must subsequently be willing and able to forward the packet on to node 8. The links and nodes a packet traverses along its journey from source to destination are called the packet's path.

Whenever a packet is transmitted from one node to another, it is said to have made a hop. In the above example, a packet sent from node 1 to node 3 requires one hop, whereas a packet sent from node 1 to node 8 requires two hops (one hop from node 1 to node 3, and a second hop from node 3 to node 8).

In the above example, the various nodes along the packet's path from node 1 to node 8 must cooperate in order to make the information exchange successful. This cooperation process is called routing.

Routing in conventional network technologies, such as the Internet Protocol (IP) [10] or ATM [11], is simplified by designing the link topology in a predictable (usually hierarchical) manner. The topology of a network is the way in which nodes are connected to each other. In these networks, the routing protocol can take advantage of the topology structure, resulting in a simplified routing algorithm.

Conventional networks tend to change infrequently, relaxing the burden on the routing protocol to return to a stable state in response to a topology change. The process of returning to a stable state after a topology change is called convergence. The time required for a routing protocol to converge is called its convergence time. As will be seen in later sections, many routing protocols (in both ad hoc and conventional networks) can form temporary routing loops when the topology changes. These routing loops may persist until the routing protocol converges.

3. CHARACTERISTICS OF AD HOC NETWORKS OF MOBILE NODES

An ad hoc network is a collection of mobile nodes forming a temporary network without the aid of any centralized administration or standard support services regularly available

on conventional networks. In this paper, it is assumed the mobile hosts use wireless RF transceivers as their network interface, although many of the same principles will apply to infra-red and wire based networks. Some form of routing protocol is necessary in these ad hoc networks since two hosts wishing to exchange packets may not be able to communicate directly.

For example, Figure5 illustrates an ad hoc network with three wireless mobile hosts. Node 3 is not within the range of node 1's wireless transmitter (indicated by the circle around node 1) and vice versa. If node 1 and node 3 wish to exchange packets, they must enlist the services of node 2 to forward packets for them, since node 2 is within the range overlap between node 1 and node 3.

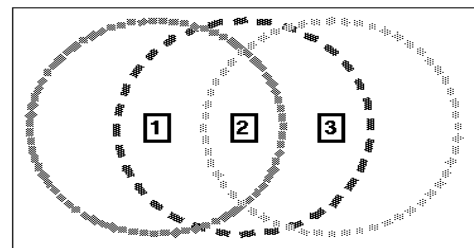


Figure5: Ad hoc network of three wireless mobile hosts.

The situation in Figure5 becomes more complicated with the addition of more nodes. The addition of just one node, as illustrated in Figure6, results in multiple paths existing between nodes 1 and 3; packets may travel the path 1 - 2 - 3, 1 - 4 - 3, or even 1 - 4 - 2 - 3. An ad hoc routing protocol must be able to decide on a single "best" path between any two nodes.

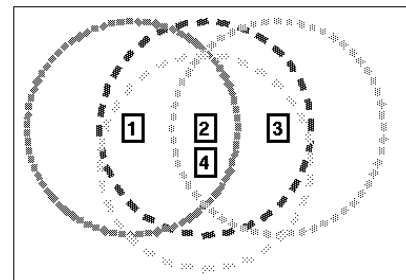


Figure6: Ad hoc network with four wireless nodes.

Another situation unique to wireless networks is illustrated in Figure7. In this example, node 1 has a large enough range to transmit packets directly to node 3. However, node 3 has a much smaller range and must enlist the help of node 2 in order to return packets to node 1. This makes the link between node

1 and node 3 appears as a one-way or unidirectional link. Most conventional routing protocols require bidirectional links.

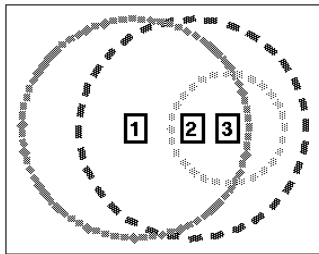


Figure7: Ad hoc network with a one-way link

Another problem with wireless network interfaces is they typically operate at significantly slower bit rates than their wire-based counterparts. Frequent flooding of packets throughout the network, a mechanism many protocols require, can consume significant portions of the available network bandwidth. Ad hoc routing protocols must minimize bandwidth overhead at the same time as they enable proper routing to take place.

Finally, ad hoc networks must deal with frequent changes in topology. By their very nature, mobile nodes tend to "wander around", changing their network location and link status on a regular basis. Furthermore, new nodes may unexpectedly join the network or existing nodes may leave or be turned off. Ad hoc routing protocols must minimize the time required to converge after these topology changes. A low convergence time is more critical in ad hoc networks because temporary routing loops can result in packets being transmitted in circles, further consuming valuable bandwidth. For the purposes of this report, an ad hoc routing protocol is considered to fill the space between two network extremes. At one extreme, the network topology is changing so rapidly the only feasible routing mechanism is for every packet to be flooded throughout the network. At the other extreme, the network topology is sufficiently permanent and static as to permit the use of conventional routing mechanisms such as those employed in the Internet. Ad hoc networks are networks which lack the support structure and permanency of traditional networks, yet change sufficiently slowly as to permit the use of a routing protocol to optimize transmission bandwidth.

4. DESIGN POSSIBILITIES

There are many ways to achieve the AODV routing protocol AODV needed to infer the event. There may be opportunities to get events: (1) snooping (2) kernel modification (3) Netfilter In the following, each possibility are described, and give their advantages and disadvantages.

1) Snooping

In order to determine the needed events is to promiscuously snoop all incoming and outgoing packets.[5] The code to perform snooping is built into the kernel and is available to user space programs. For e.g. An ARP packet is generated when a node does not know the MAC layer address of the next hop. Using this interface, if an ARP request packet is seen for an unknown destination and it is originated by the local host, then a route discovery needs to be initiated. Similarly, all other AODV events may be determined by monitoring incoming and outgoing packets. The most important advantage of this solution is it does not require any code to run in the kernel space. Hence it allows for simple installation and execution .But two disadvantages are overhead and dependence over ARP.

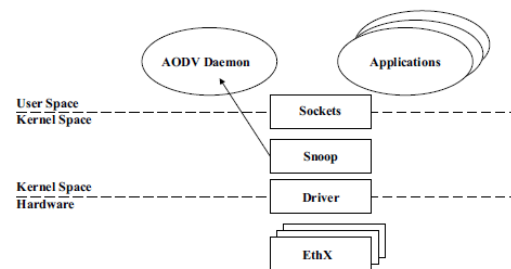


Figure8: Snooping Architecture

2) Netfilter

Netfilter [2] is a set of hooks at a various points inside the Linux protocol stack. Netfilter redirects packet flow through user defined code, which can examine, drop, discard, modify or queue the packets for user space daemon. Using Netfilter is similar to snooping method however it does not have the disadvantage of unnecessary overhead or dependence on ARP. This solution has the strength such as there is no unnecessary communication; it is highly portable, it is easy to install and user space daemon can determine all the required events. On the other hand, the disadvantage of this solution is that it requires a kernel module. However kernel module is easier than kernel modifications. A kernel module is more portable than kernel modifications because it depends only on the Netfilter interface. This interface does not change from one kernel version to next.

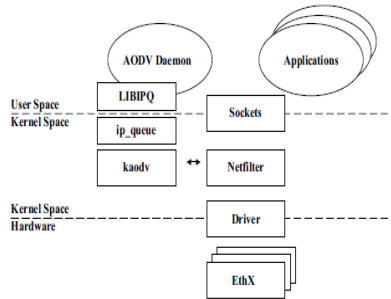


Figure9: Netfilter Architecture.

3) Kernel Modification

In order to determine the AODV events is to modify the kernel. Code can be placed in the kernel to communicate the events to an AODV user-space daemon. For example, to initiate route discovery, code is added in the kernel at the point where route lookup failures occur. Given this code in the kernel, if a route lookup failure happens, then a method is called in the user-space daemon. Figure8 shows the architecture of the AODV daemon and the required support logic. The advantages of this solution are that the events are explicitly determined and there are no wasted overhead. The main disadvantages of this solution are user installation and portability. Installation of the necessary kernel modifications requires a complete kernel recompilation. This is a difficult procedure for many users. Also, kernel patches are often not portable between one kernel version and the next. Finally, understanding the Linux kernel [14] and network protocol stack requires examining a significant amount of uncommented, complex code.

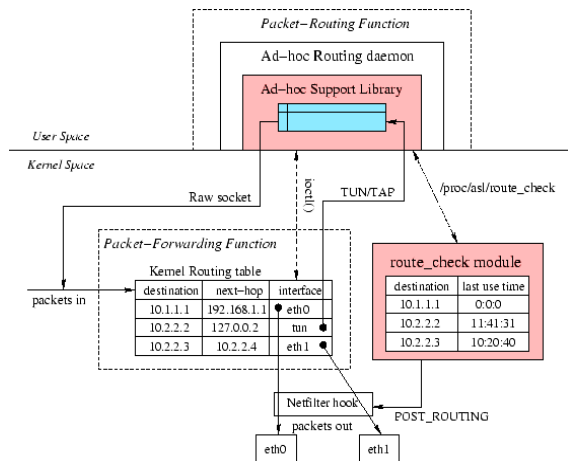


Figure10: kernel modification architecture

5. CONFIGURATION OF WIRELESS ADAPTER IN AD-HOC MODE.

Before you running “aodvd” you need to configure you wireless card (wlan0).

Try to use static IP in every node. Don’t forget to set Netmask, Mode as “ad-hoc”, ESSID, Channel, Rate.

```
stop network-manager
ifconfig wlan0 down
iwconfig wlan0 mode ad-hoc
iwconfig wlan0 channel 6
iwconfig wlan0 essid 'aodv'
ifconfig wlan0 up
ifconfig wlan0 10.0.0.4 netmask
255.255.255.0
```

Extended Service Set Identification (ESSID):

The Extended Service Set Identification (ESSID) is one of two types of Service Set Identification (SSID). In an ad hoc wireless network with no access points, the Basic Service Set Identification (BSSID) is used. In an infrastructure wireless network that includes an access point, the ESSID is used, but may still be referred to as SSID. An SSID is a 32-character (maximum) alphanumeric key identifying the name of the wireless local area network. Some vendors refer to the SSID as the network name. For the wireless devices in a network to communicate with each other, all devices must be configured with the same SSID.

Wireless Channels:

Choosing the best Wi-Fi channel on your router helps to reduce interference and improve your Wi-Fi signal. These tools will help you identify the least congested Wi-Fi channel in your area. Wi-Fi channels overlap with nearby channels. Channels 1, 6, and 11 are the most frequently used for 2.4 GHz Wi-Fi. These three channels are the only ones that don’t overlap with each other. As shown in figure 11.

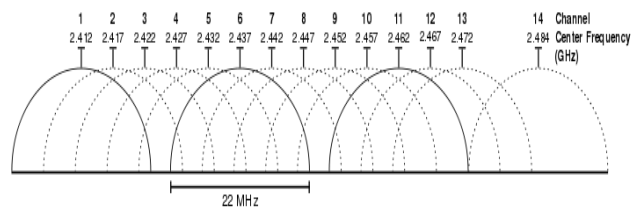
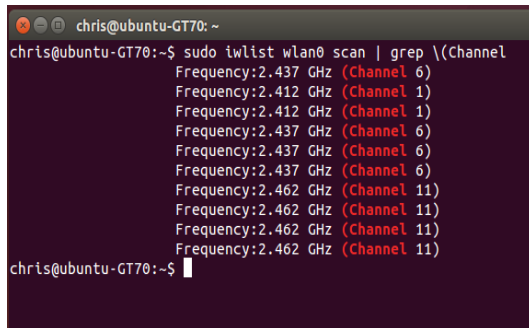


Figure11: Graphical representation of 2.4 GHz band channels overlapping

Open a Terminal and run the following command:

```
sudo iwlist wlan0 scan | grep \(\Channel
```

Read the output of the command to see which channels are the most congested and make your decision. In the screenshot below, channel 1 looks the least congested.



```
chris@ubuntu-GT70:~$ sudo iwlist wlan0 scan | grep \(\Channel
Frequency:2.437 GHz (Channel 6)
Frequency:2.412 GHz (Channel 1)
Frequency:2.412 GHz (Channel 1)
Frequency:2.437 GHz (Channel 6)
Frequency:2.437 GHz (Channel 6)
Frequency:2.437 GHz (Channel 6)
Frequency:2.462 GHz (Channel 11)
Frequency:2.462 GHz (Channel 11)
Frequency:2.462 GHz (Channel 11)
chris@ubuntu-GT70:~$
```

Figure12: Channel List screenshot

Kernel IP forwarding

"IP forwarding" is a synonym for "routing." It is called "kernel IP forwarding" because it is a feature of the Linux kernel. A router has multiple network interfaces. If traffic comes in on one interface that matches a subnet of another network interface, a router then forwards that traffic to the other network interface.

So, let's say you have two NICs, one (NIC 1) is at address 192.168.2.1/24, and the other (NIC 2) is 192.168.3.1/24. If forwarding is enabled, and a packet comes in on NIC 1 with a "destination address" of 192.168.3.8, the router will resend that packet out of the NIC 2. It's common for routers functioning as gateways to the Internet to have a default route whereby any traffic that doesn't match any NICs will go through the default route's NIC.

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

6. AODV (Adhoc On Demand Vector)

AODV is one of the reactive (on-demand) routing protocol that developed by Uppsala University dan University of Basel. Before installation, you'd better check for 'development tools' and kernel configuration on your Linux OS. If both of them are not installed yet, you should installed them first. AODV-UU compilation is performed by 'make' utility from Linux OS. The make file will create an aodv daemon (aodvd) and a kernel modul (kaodv.o). The installation is as follows:

1. First, we must have installed build-essential correctly. If not, use this command:
`apt-get install build-essential`
2. After that, download the setup file.[13]
3. And then extract the file.
4. Enter to the file using:
`cd aodv-uu-0.9.x`
5. Install it using command:
`make && make install`
6. After that, load the kaodv modul on Ubuntu, using:
`modprobe kaodv`
7. After all the step finish, activate aodv using command:
`aodvd -l -d -D -r 3`

Command -l mean write a output of this command in the /var/log/aodvd.log

Command -D mean non-activate waiting time (15 second) as a reboot delay when the protocol fail to run

Command -d mean run the protocol in background

Command -r 3 mean save the routing table in the /var/log/aodvd.log every 3 second

And finally, aodv has ran. And then we must determine the IP address statically.

To check that AODV is successfully run, try to ping other machines in the established ad hoc network. Since in our lab all the machines are within the reception range of each other (i.e., any pair of machines can communicate directly with each other), we will apply the following technique to simulate multihop communication.

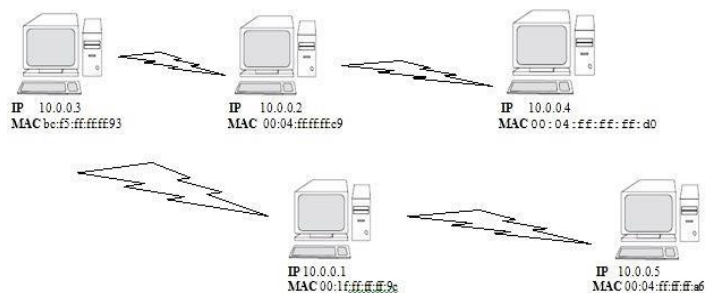


Figure 13: Multihop wireless network

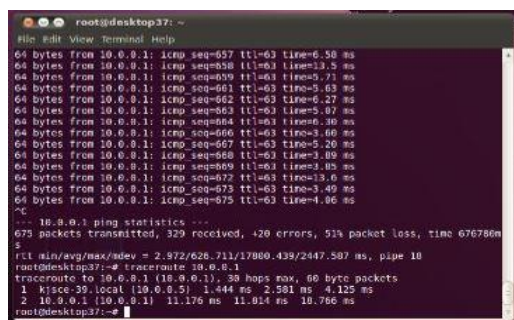
Assume that we want to simulate a situation in which machines with MAC addresses bc:f5:ff:ff:ff:93 and 00:04:ff:ff:ff:d0 cannot transmit directly to each other (see the figure above). However, both machines can hear machine 00:04:ff:ff:ff:c9. To achieve this we will use the iptables utility [2]. Iptables gives us the functionality of packet filtering. This is done on the network layer. The command to use to drop packets arriving from a specific machine is:

```
#iptables -A INPUT -m mac --mac-source bc:f5:ff:ff:ff:93 -j DROP
```

Where bc: f5:ff:ff:ff:93 is the MAC address of the node of which messages should be dropped. To see the list of MAC addresses that are blocked, type: iptables -L, for help iptables --help. In our example, on machine bc:f5:ff:ff:ff:93, we have to execute the following command:

```
#iptables -A INPUT -m mac --mac-source 00:04:ff:ff:ff:d0 -j DROP
```

After the establishing setup we try to trace rout from IP 10.0.0.4 to 10.0.0.1 the output screenshot shown in below figure14.



```
root@desktop37: ~
File Edit View Terminal Help
64 bytes from 10.0.0.1: icmp_seq=557 ttl=63 time=6.50 ms
64 bytes from 10.0.0.1: icmp_seq=558 ttl=63 time=13.5 ms
64 bytes from 10.0.0.1: icmp_seq=559 ttl=63 time=5.71 ms
64 bytes from 10.0.0.1: icmp_seq=560 ttl=63 time=3.63 ms
64 bytes from 10.0.0.1: icmp_seq=561 ttl=63 time=6.27 ms
64 bytes from 10.0.0.1: icmp_seq=562 ttl=63 time=5.27 ms
64 bytes from 10.0.0.1: icmp_seq=563 ttl=63 time=5.07 ms
64 bytes from 10.0.0.1: icmp_seq=564 ttl=63 time=3.30 ms
64 bytes from 10.0.0.1: icmp_seq=565 ttl=63 time=3.60 ms
64 bytes from 10.0.0.1: icmp_seq=566 ttl=63 time=5.20 ms
64 bytes from 10.0.0.1: icmp_seq=567 ttl=63 time=3.09 ms
64 bytes from 10.0.0.1: icmp_seq=568 ttl=63 time=2.85 ms
64 bytes from 10.0.0.1: icmp_seq=569 ttl=63 time=3.0 ms
64 bytes from 10.0.0.1: icmp_seq=570 ttl=63 time=3.49 ms
64 bytes from 10.0.0.1: icmp_seq=571 ttl=63 time=1.66 ms
^C
--- 10.0.0.1 ping statistics ---
673 packets transmitted, 329 received, 420 errors, 51% packet loss, time 6767800 s
rtt min/avg/max/mdev = 2.972/626.711/17860.439/2447.587 ms, pipe 10
root@desktop37:~# traceroute 10.0.0.1
traceroute to 10.0.0.1 (10.0.0.1): 30 hops max, 60 byte packets
 1 10.0.0.4 local (10.0.0.5) 1.444 ms 2.581 ms 4.125 ms
 2 10.0.0.1 (10.0.0.1) 11.176 ms 11.814 ms 10.766 ms
root@desktop37:~#
```

Figure14: Traceroute 10.0.0.4 to 10.0.0.1 via 10.0.0.5

CONCLUSIONS

In this paper we discussed the experimental setup to establish the AODV-UU Protocol on Ubuntu 10.04 Linux kernel 2.4 version. And we successfully analyses multihop routing with traceroute command result.

REFERENCES:

[1] C. Perkins, E. Belding-Royer, S. Das, RFC Draft for experimental AODV. <https://www.ietf.org/rfc/rfc3561.txt>.

[2] Iptables. <https://www.netfilter.org/>.

[3] LAN/MAN Standards Committee. *ANSI/IEEE Std 802.11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*. IEEE Computer Society, 1999.

[4] C. Perkins and E. Royer. "Ad-Hoc On-Demand Distance Vector Routing". In *Proceedings of IEEE WMCSA*, 1999.

[5] C. Perkins, E. Royer, and S. Das. *Ad hoc On-Demand Distance Vector Routing*. IETF MANET Draft, 2002.

[6] Mario Cagalj, Jean-Pierre Hubaux and Imad Aad "Hands-on exercises: IEEE 802.11b standard "October 12, 2007

[7] "IEEE 802.11 WG part 11a. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications, High-speed Physical Layer in the 5 GHz Band," 1999.

[8] "IEEE 802.11 WG part 11g. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications, Further Higher Speed Physical Layer Extension in the 2.4 GHz Band,"2003

[9] "IEEE 802.11 WG part 11b. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications, Higher Speed PHY Layer Extension in the 2.4 GHz Band," 1999.

[10] "Internet Protocol," RFC 791, Information Sciences Institute, September 1981.

[11] Harry J. R. Dutton, Peter Lenhard, *Asynchronous Transfer Mode*, Prentice Hall, 1995.

[12] B. Cameron Lesiuk "Routing in Ad Hoc Networks of Mobile Hosts", <http://ghost.lesiuk.org/AdHoc/adhoc/#E7E2> December , 1998

[13] "AODV-UU" <https://sourceforge.net/projects/aodvuu/>

[14] <http://www.kernel.org>