Reduce 802.11 Connection Time Using Offloading and Merging of DHCP Layer to MAC Layer

Vishal Bhargava and Nallanthighal S. Raghava(✉)

Delhi Technological University, Delhi 110042, India

Abstract. With the growth of Wi-Fi by the time, performance and quality are facing quite a lot of challenges. Wi-Fi works on the principle of the IEEE 802.11 based carrier sense multiple access – collision avoidance (CSMA/CA) to transmit packets and distributed coordination function (DCF) protocol based on Inter-frame spacing used to make the gap between two frames. In this manner, an 802.11 device gets limited time in the environment to finish its activity, with a rapid increase in the number of devices in the 802.11 environments. The important operation is always a connection of Wi-Fi device with another device, another device can be an Access point or any other Wi-Fi device (ad-hoc or Wi-Fi direct mode). Several researchers work on why connection time is more important and what factors affect its most [1, 2]. In this paper, we describe a way to reduce connection time with the use of cross-layer approach via offloading DHCP work to the MAC layer.

Keywords: 802.11 connection · DHCP · Authentication · Association · Offloading · Cross-layer approach

1 Introduction

In recent years 802.11 throughput acquires tremendous growth from 802.11b/g to 802.11n to 802.11ac to 802.11ax. With the growth of the throughput, the number of Wi-Fi devices increases exponentially (Fig. 1).

More than 1 billion access points were sold last year, and the station connected to an access point can be numerous. Wi-Fi devices connection is a standard process [4] defined by the IEEE 802.11 specification. Any Wi-Fi station device which wants to connect with an access point first performs a scanning operation. Earlier devices have 2.4 GHz band support only, but now mostly devices come with 5 GHz band support also, so scanning work increases for them. To save time, devices performs an active scan in comparison of passive scan (performed for DFS channels).

Figure 2 shows the Wi-Fi connection in open security mode. In the case of security mode (WPA/WPA2), additional 4-way handshake comes after the association process. In the connection process after scan DHCP process takes maximum time [2]. Connection sub phases are scan, authentication/association and DHCP assignment.

S. Paiva et al. (Eds.): SmartCity360° 2020, LNICST 372, pp. 171–187, 2021.
https://doi.org/10.1007/978-3-030-76063-2_13

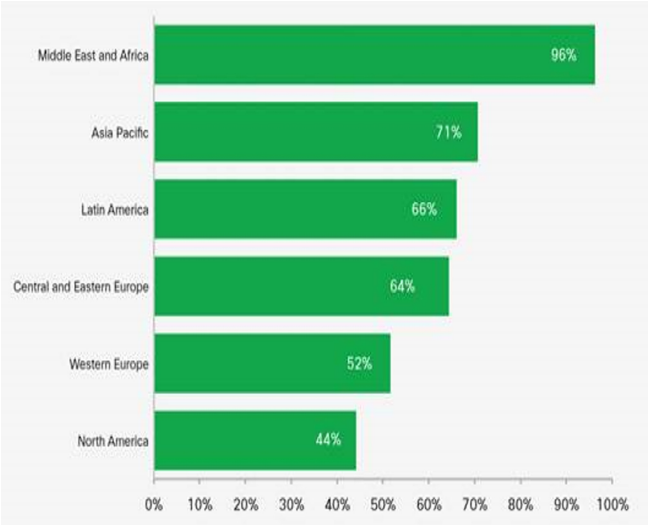

Fig. 1. Wi-Fi Data Traffic Growth in 2016 for different regions [3].

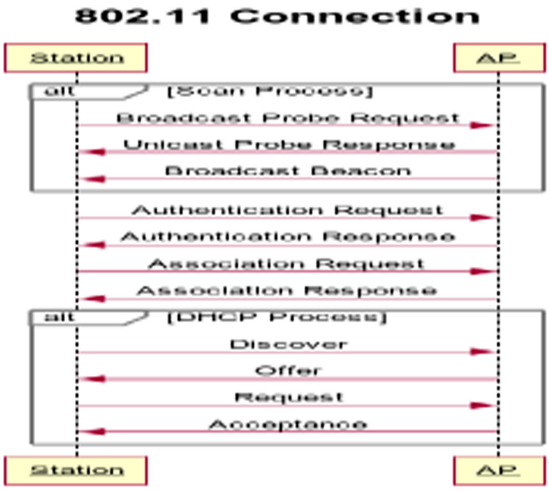

Fig. 2. Wi-Fi Connection between Station and Access Point.

1.1 Scan Phase

The scan is the process to find the desired Access point via station to which it wants to connect. There are mainly two types of scan: active scan and passive scan [4]. In active scan station device sends a frame which is called probe request frame to access point. Probe request can be unicast or broadcast. In response to the probe request, the Access point sends a probe response, which is used by a station to take connection related decision. In a passive scan, the station device wait for a special frame called beacon

frame from the access point and connection decision taken according to that. A passive scan is a power saving technique but it is time-consuming in comparison to an active scan.

1.2 Authentication/Association Phase

After scan, now it is the turn of Wi-Fi station to show its capability to access point – whether it's a real authentic device to connect, and if yes, what it supports, what not, an access point takes the decision on the basis of these points, whether station can connect to access point or not. Four important frame exchange is done sequentially in this phase: authentication request, an authentication response, the association request an association response.

In the case of WPA/WPA2 security, additional 4-way handshake perform after the above four frame exchange.

1.3 DHCP Phase

Till now Wi-Fi device and Access point only interact with the perspective of the MAC layer. Now, it's application and IP layer's turn, a 1st station needs an IP address so the application can use it for communication with other devices.

After a successful association, the station device will interact with the DHCP server which can be inside of access point or can be a different entity. As an IP address is needed for communication, this phase is also considered in connection phase, actual real data transfer after this one.

Every network device has two types of addresses to communicate, so the wireless interface card also has two address. First one is the MAC address or physical address which is unique to device and second is a logical address also called IP address. IP address work on layer 3 (networking layer) so it can be considered as layer 3 address, while the MAC address is layer 2 address.

DHCP (Dynamic Host Configuration Protocol) is used to provide an IP address to a device. This IP address is provided by the remote server. DHCP works on a client-server model, where the device which needs IP address acts as a client and server assign an IP to the client.

DHCP mechanism is a combination of 4 processes (DORA). Stated below are the message exchanges between DHCP client and server.

- Discover
- Offer
- Request
- Acknowledgement

Below diagram shows the message exchange between the DHCP client and the Server (Fig. 3):

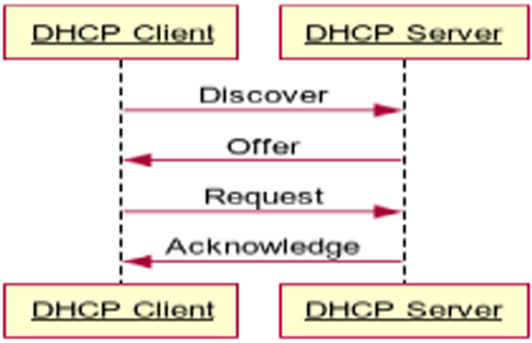

Fig. 3. DHCP Message Exchange between DHCP Client and DHCP Server.

Now let's go through every frame exchange between client & server with consideration of the OSI layer model, especially layer 2 i.e. data link layer and layer 3 i.e. networking layer.

Discover Frame or Message1

Device who is looking for an IP address sends a DHCP Discover message intended to search appropriate DHCP server. DHCP discover message is a layer 2 broadcast as well as layer 3 broadcast frame ((Figs. 4 and 5).

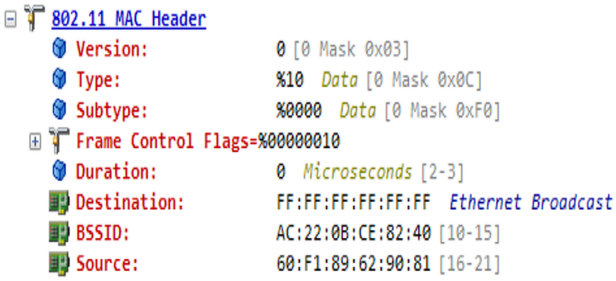

Fig. 4. Sniffer capture of DHCP Discover (MAC Layer Perspective).

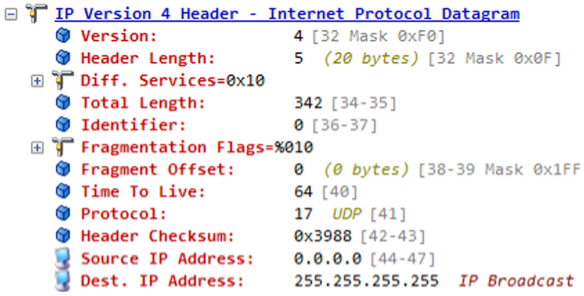

Fig. 5. Sniffer capture of DHCP Discover (IP Layer Perspective).

Offer Frame or Message 2

In response to Discover message, the DHCP server sends the DHCP offer message (Figs. 6 and 7).

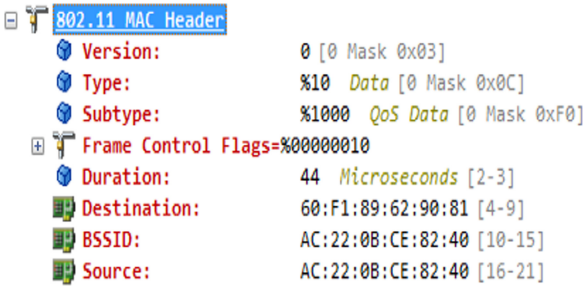

Fig. 6. Sniffer capture of DHCP Offer (MAC Layer Perspective).

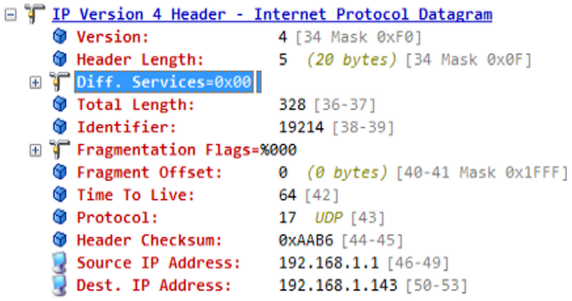

Fig. 7. Sniffer capture of DHCP Offer (IP Layer Perspective). Sniffer capture of DHCP Offer (IP Layer Perspective).

Request Frame or Message 3

After receiving the offer, DHCP client sends a DHCP Request message to the server, in which client either accept IP address given by server or it can request the new one. It

can be seen at the mac layer as a unicast frame while broadcast on the IP layer and in this frame DHCP server provides an IP to the client (Figs. 8 and 9).

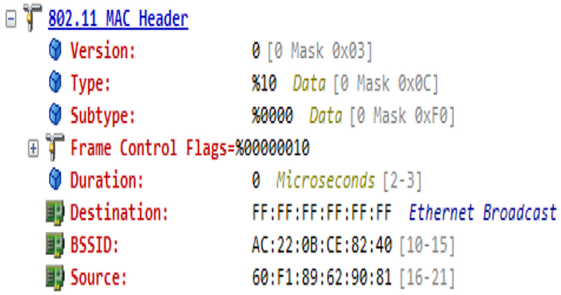

Fig. 8. Sniffer capture of DHCP Request (MAC Layer Perspective).

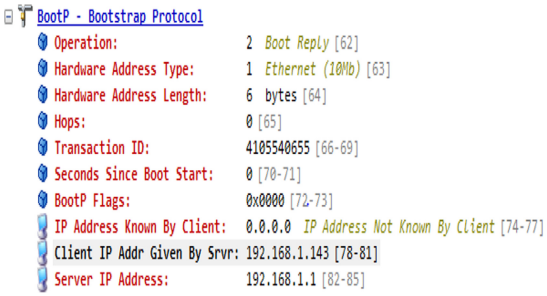

Fig. 9. Sniffer capture of DHCP Request (IP Layer Perspective).

Acceptance Frame or Message 4

After the request frame, the server sends an acceptance/acknowledge frame, whether it has accepted the client IP or denied it. The reason code is also mentioned in case of failure so a client can get information about why the DHCP server has rejected the request. This frame is unicast on both layers (Figs. 10 and 11).

Below figure shows complete sniffer capture for a station to access-point Wi-Fi connection in open security mode (Fig. 12):

The organization of this paper is as follows: Section 2 presents related work and research objectives. Our Simulation and Test Results and its conclusion is discussed in Sect. 3. Proposed work & feedback is described in Sect. 4 and finally, conclusions and future work are drawn in Sect. 5.

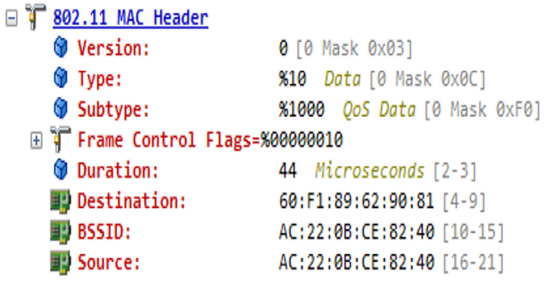

Fig. 10. Sniffer capture of DHCP Acknowledge (MAC Layer Perspective).

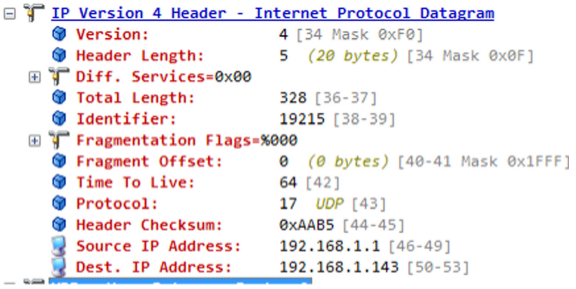

Fig. 11. Sniffer capture of DHCP Acknowledge (IP Layer Perspective).

Packet	Source	Destination	BSSID	Flags	Channel	Signal	Data Rate	Size	Relative Time	Protocol	Decode: Message Type	
1	60:F1:89:62:90:81	AC:22:0B:CE:82:40	AC:22:0B:CE:82:40	M	11	47%	1.0	45	0.000000	802.11 Auth		
3	AC:22:0B:CE:82:40	60:F1:89:62:90:81	AC:22:0B:CE:82:40	M	11	76%	1.0	45	0.001005	802.11 Auth		
4	60:F1:89:62:90:81	AC:22:0B:CE:82:40	AC:22:0B:CE:82:40	M	11	65%	1.0	90	0.009375	802.11 Assoc Req		
6	AC:22:0B:CE:82:40	60:F1:89:62:90:81	AC:22:0B:CE:82:40	M	11	76%	1.0	97	0.010746	802.11 Assoc Rsp		
8	0.0.0.0	IP Broadcast	AC:22:0B:CE:82:40		11	78%	1.0	378	0.422490	DHCP	1	Discover
9	192.168.1.1	192.168.1.143	AC:22:0B:CE:82:40		11	78%	54.0	366	0.422605	DHCP	2	Offer
10	0.0.0.0	IP Broadcast	AC:22:0B:CE:82:40		11	65%	36.0	392	0.454995	DHCP	3	Request
11	192.168.1.1	192.168.1.143	AC:22:0B:CE:82:40		11	78%	54.0	366	0.459353	DHCP	5	ACK

Fig. 12. Sniffer capture of Wi-Fi Connection in open mode.

2 Related Work and Research Objectives

At a broad level, fast operation and increased data throughput are a great challenge in a Wi-Fi environment and researchers are working to overcome these challenges. Users are increasing day by day using Wi-Fi and analyst are working on their behavior and problems [19]. Active scanning is also used to reduce connection time as shown in several research reports and also reduction of MAC layer handover time [8]. Partial scanning

(perform channel scanning in chunks, not in one go) and special scanning algorithms [9, 10, 13] also works well in the fast handoff between the station and access point. Power consumption is also a challenging area especially for battery operated device [20] and offloading of features are performed to overcome this challenge in a small manner.

To make a fast connection, researchers used caching of previous connection information [5] or do fast sub-steps like authentication [6]. To improve connection time pre-allocation of DHCP also done [7]. In the case of roaming, authentication frame uses to share 4-way handshake information [11] for a fast connection.

3 Our Work, Simulation, and Results

To perform the test, we have used our own simulator system on windows operating system. We compile our driver and application by Visual Studio 2017 and use WinDDK to build the driver [17]. To perform our test, we developed a simple command line application and a windows WDM driver [15]. Here application acts as both DHCP Client & Server application for the station and access point respectively and this application talk with a simple WDM driver [16] using IOCTL. WDM driver works as the driver itself and as a virtual firmware and hardware device for a Wi-Fi device.

Here we assume access point has an embedded DHCP server. First, we tried to offload DHCP feature, like ARP or EAPOL offload (GTK rekey) performed in case of WoWLan (Wake on Wireless LAN) [13] feature. Offloading means instead of host/driver, the feature is performed by firmware. Mainly offloading is for the reduction of power consumption of device but it has saved a little time as driver-device latency reduced [12]. Later, we merge a DHCP frame with connection frames.

DHCP mainly IP layer procedure, and it is time-consuming also. If we talk from 802.11 perspectives, DHCP related frames are data frames, while connection frame like authentication & association is management frames. In our work, we have offloaded the DHCP feature and we have combined this feature with MAC layer connection frame via adding DORA [14] support in authentication/association frame. We have already discussed "connection and DHCP both have 4 frames respectively". So, we have done one to one mapping to create new frames which are shown in Table 1.

We have moved all the things to the MAC layer and at the MAC layer, Authentication Request is a unicast frame to AP, while DHCP discover is a multicast frame. Here we are assuming that access point has embedded DHCP server, so discover frame can be unicast as well.

Table 1. Proposed frames

Proposed frame	Direction	Combination
Authentication request	Station to AP	Authentication Request + DHCP Discover
Authentication response	AP to station	Authentication Response + DHCP Offer
Association request	Station to AP	Association Request + DHCP Request
Association response	AP to station	Association Response + DHCP Acceptance

Apart from discover frame, all other frames are unicast at mac layer, so it does not provide any problem.

The proposed Auth request frame have special IE (Information Element) present, which shows whether the device supports static IP or a dynamic IP and DHCP related (layer 3) information. If static set in station's auth request IE, AP's (inbuilt DHCP server) does not need to provide any IP to the station. Same in other connection frames: Auth response, Assoc request & Assoc response frame also have special IE which consists of DHCP related information.

Figure 13 shows the general device model about the layers a wifi device consist. Here firmware runs inside the device hardware, so we assume both are at the same layer (Fig. 14).

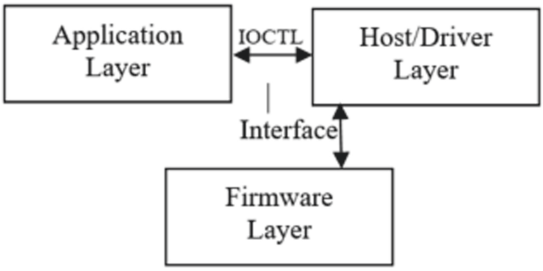

Fig. 13. General Device model.

To simulate the test, we have created two layers at user space and kernel space, Application runs at user space and driver & virtual firmware functionality runs at kernel space. The application works as a DHCP Client and DHCP server for station & access-point unit respectively. Application talks with the driver using IOCTL and a fixed interface delay gap are given between Host & Firmware function at the driver side. In a real device, an interface exists via which device is connected to the system, this interface can be USB, PCIe, SDIO or anything else, so an interface delay added between driver and firmware function.

Above figure shows our simulator model to perform the test. And below steps performed via every entity participate in this test:

1) Create connection command comes to application.
2) The application sends create connection OID to device driver with DHCP or Static IP support.
3) OID is handled by Client_Host function and it bypasses details to Client_Firmware function.
4) Now Client_Firmware sends proposed Auth Request frame to Server_Firmware function.
5) Server_Firmware process auth request frame and response back with auth response frame.
6) Client_Firmware receive Auth response frame and after successful processing, it sends back association request frame to server_firmware.

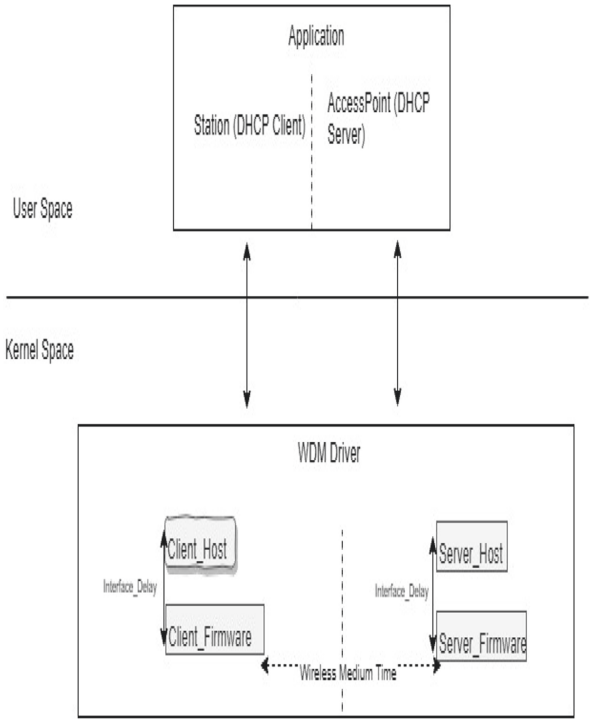

Fig. 14. Simulator Device Model in our test.

7) Finally, the association response frame is reverted back by Server_Firmware.
8) Respectively, Client_Firmware and Server_Firmware sync information with its respective application part via Client_Host&Server_Host bypass function.

In case of default behavior, DHCP works at the application layer and connection procedure works at the host or in some cases, at firmware side.

In our model, we have moved all the new frames to the firmware layer for fast execution. The application just provides required information to firmware via the host, so the new frame can be created at firmware side and send to access point as shown in Fig. 15.

Same at Access Point side, DHCP server & authentication works at firmware layer.

Below is the packet hex dump generated by our application. The color rule is followed to denote packet internally. Here Yellow color shows MAC Header, Green color shows connection Specific frame data (Auth or Association frame) and Pink color denotes DHCP Relative frame data. The default is the FCS field. In Auth Request, the red color is vendor command 0xDD followed by 0x0A, shown station support dynamic IP configuration instead of Static IP configuration.

For this test:

- AP MAC Address: 00:11:22:33:44:55

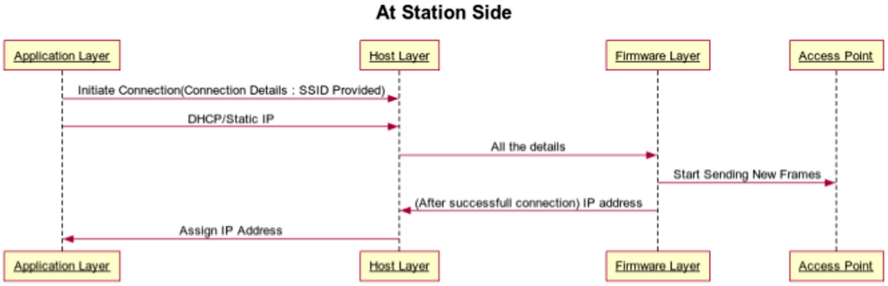

Fig. 15. Station side implementation. (Color figure online)

- Station MAC Address: 00:00:66:77:88:99

A. **Auth Request**:

B0 00 3C 00 00 11 22 33 44 55 00 00 66 77 88 99 00 11 22 33 44 55 D0 0E00 00 01 00 00 00DD 0ADD 01 00 00 00 00 00 00 00 00 00 00 00 00 15 93 DE D9

B. **Auth Response:**

B0 00 32 00 00 00 66 77 88 99 00 11 22 33 44 55 00 11 22 33 44 55 E0 9400 00 02 00 00 00DD 02 00 00 00 00 C0 A8 2B 2E C0 A8 2B BE A7 73 EB 4B

C. **Assoc Request:**

00 00 3C 00 00 11 22 33 44 55 00 00 66 77 88 99 00 11 22 33 44 55 E0 0E11 01 0A 00 00 09 56 49 53 48 41 4C 5F 35 47 01 08 8C 12 98 24 B0 48 60 6CDD 03 00 00 00 00 00 00 00 00 00 00 00 00 32 04 C0 A8 2B 2E 4C 21 96 CE

D. **Assoc Response:**

10 00 32 00 00 00 66 77 88 99 00 11 22 33 44 55 00 11 22 33 44 55 F0 9411 01 00 00 02 C0 01 08 8C 12 98 24 B0 48 60 6CDD 04 00 00 00 00 C0 A8 2B 2E C0 A8 2B BE 35 DE 27 67

After connection data, DHCP data start from 0xDD (Denotes vendor specific command). Here is the DHCP information inside packets which is exchanged between client and server.

- Message Type - 1 (Discover)

IP Address Known By Client: 0.0.0.0
Client IP Addr Given By Srvr: 0.0.0.0
Server IP Address: 0.0.0.0

- Message Type - 2 (Offer)

IP Address Known By Client: 0.0.0.0
Client IP Addr Given By Srvr: 192.168.43.46
Server IP Address: 192.168.43.179

- Message Type - 3 (Request)

IP Address Known By Client: 0.0.0.0
Client IP Addr Given By Srvr: 0.0.0.0
Server IP Address: 0.0.0.0
Requested IP Address
Option Code = 50
Option Length = 4
Address = 192.168.43.46

- Message Type - 4 (Ack)

IP Address Known By Client:0 .0.0.0
Client IP Addr Given By Srvr: 192.168.43.46
Server IP Address :192.168.43.179

It simply provides a reduction of 4 frames in the connection process. We have also performed a timing comparison between a normal connection procedure and our proposed procedure. Here is the timing variable used in the calculation.

- T_{MODE}: User Space to Kernel mode delay time
- $T_{INTERFACE}$: Interface Delay
- T_{TRANS}: Wireless Medium Time (Packet air travel time)

First, we will calculate DIFS (Distributed coordinator function Inter Frame Space), between two separate transmissions:

$$T_{SIFS} = 10\,\mu s \tag{1}$$

$$T_{SLOT} = 20\,\mu s \tag{2}$$

$$T_{SLOT} = 20\,\mu s \tag{3}$$

$$T_{DIFS} = T_{SIFS} + 2 \times T_{SLOT} = 10\,\mu s + 2 \times 20\,\mu s = 50\,\mu s \tag{4}$$

Coming on to the packet, it firstly consists of PHY header further consisting of PLCP preamble (144 bits) and header (48 bits). Here we have DSSS mode and assumed the packet transmission rate as 1Mbps. Therefore, time to transmit PHY header will be:

$$T_{PHY} = (144\,\text{bits})/(1\,\text{Mbps}) + (48\,\text{bits})/(1\,\text{Mbps}) = 192\,\mu s \tag{5}$$

Next up will be the MAC Header which is 24bytes (192 bits) which will also transfer at 1 Mbps. Therefore.

$$T_{FCS} = (32\,\text{bits})/(1\text{Mbps}) = 32\,\mu s \tag{6}$$

Now payload will vary according to the packet, so now we will calculate FCS (Frame Check Sequence) which is 4 bytes (32 bits) long.

Also after each packet we have ACK (Acknowledgement frame) sent by the receiver. The MAC header of ACK frame is.

10 bytes (80 bits) long which will take 80 μs to transmit. Therefore ACK transmission time will be:

Using Eqs. (4) and (6).

$$T_{ACK} = T_{PHY} + 80\,\mu s + T_{FCS} = 304\,\mu s \tag{7}$$

As discussed, payload transmission time will be our variable and total time for complete transmission of any packet would be:

$$T_{TRANS} = T_{PHY} + T_{MAC} + T_{PAYLOAD} + T_{ACK} \tag{8}$$

Now we are ready to calculate the total time for each packet present in normal and our proposed model.

Let us first see the present model. T_{N_TOTAL} is total transmission time in normal scenario (Tables 2, 3, 4 and 5).

Table 2. Transmission time in normal scenario.

Payload	Length (bytes)	$T_{Payload}$ (μs)	T_{Trans} (μs)
Authentication Request	48	48	736
Authentication Response	48	48	736
Association Request	200	200	888
Association Response	128	128	816
DHCP Discover	2400	2400	3088
DHCP Offer	2700	2700	3408
DHCP Request	2400	2400	3088
DHCP ACK	2700	2700	3408
T_{N_TOTAL}			**16,168**

Now the proposed model: In our proposed model we have additional bytes to replace DHCP 4 packets exchange. These additional bytes will take additional time and is to be added in the total transmission time of each packet. T_{P_TOTAL} is total transmission time in proposed scenario.

To calculate the time between user mode & kernel mode we have used Dbgview.exe [18] and using print we got it.

Table 3. Transmission time in proposed scenario.

Payload	Length (bits)	Additional Length (bits)	$T_{Payload}$ (μs)	T_{Trans} (μs)
Authentication request	48	112	112	848
Authentication response	48	112	112	848
Association request	200	160	160	1048
Association response	128	112	112	928
$\mathbf{T_{P\_TOTAL}}$				**3,672**

For interface delay calculation assumption is interface equivalent to USB2.0, and practically USB 2.0 speed is around 40 megabytes per second (MBps) [21]. According to this, 112 byte takes around 2.46 μs.

Here is the value:

- $T_{INTERFACE}$ – 2.46 μs.
- T_{MODE} – 93 μs.

$$\text{Total time in all 9 steps} = 8 * T_{INTERFACE} + 5 * T_{MODE} + T_{P\_TOTAL} \quad (9)$$

Table 4. Default behavior steps

S. No.	Step Detail	Time
1	Connection Request	T_{MODE}
2	Auth Request via station driver	$T_{INTERFACE} + T_{TRANS}$
3	Auth Response via AP driver	$T_{INTERFACE} + T_{TRANS}$
4	Assoc Request via station driver	$T_{INTERFACE} + T_{TRANS}$
5	Assoc Response via AP driver	$T_{INTERFACE} + T_{TRANS}$
6	DHCP Discover from application (Client side)	$T_{MODE} + T_{INTERFACE} + T_{TRANS}$
7	DHCP Offer from Application (Server side)	$T_{MODE} + T_{INTERFACE} + T_{TRANS}$
8	DHCP Request from application (Client side)	$T_{MODE} + T_{INTERFACE} + T_{TRANS}$
9	DHCP Acceptance from Application (Server side)	$T_{MODE} + T_{INTERFACE} + T_{TRANS}$

$$\text{Total time in all 6 steps} = 2 * T_{INTERFACE} + 2 * T_{MODE} + T_{N\_TOTAL} \quad (10)$$

Table 5. Proposed behavior steps

S. No.	Step Detail	Time
1	Connection Request	$T_{MODE} + T_{INTERFACE}$
2	Auth Request via station driver	T_{TRANS}
3	Auth Response via AP driver	T_{TRANS}
4	Assoc Request via station driver	T_{TRANS}
5	Assoc Response via AP driver	T_{TRANS}
6	DHCP Discover from application (Client side)	$T_{INTERFACE} + T_{MODE}$

Total time in default behavior from Eq. 9.

$$= 8 * 2.46 + 5 * 93 + 16168 = 18463.24\,\text{ms}.$$

Total time in proposed behavior from Eq. 10.

$$= 2 * 2.46 + 2 * 93 + 3672 = 4315.56\,\text{ms}.$$

It is clearly visible that the proposed model saves connection time in the form of – reduced interface & IOCTL delay and change from 8 frame transmission to 4 frame transmission and it is around 1/4rd of default behavior.

4 Future Work and Conclusion

4.1 Future Work

Although the suggested approach significantly can reduce the wi-fi connection time, it still has some points which are to be taken care of in the future.

1) The current approach is tested in a simulator way, not in real time environment. So real-time implementation needs to done by Wi-Fi chip vendors.
2) This feature needs support from both side – station and access point, so backward compatibility case needs to tackle by both (Station & AP).
3) Every access point does not have an embedded DHCP server, so in that case, this feature development would be tough.
4) Auth Response and Assoc response consists of failure reason in case anything goes wrong from connection perspective. In case of DHCP failure more reason code needs tos be added in Auth Response and Assoc response specification.
5) In Simulation only success case is covered, it would be interesting to see if connection wise environment is fine but not for data exchange.

5 Conclusion

In this paper, through simulation and results, a new way is proposed for a fast Wi-Fi connection; Things moved from application to driver/firmware level and the number of frames reduced. The proposed solution shown is only for IPV4 but the same can be applied for IPV6 as well. Currently, the IEEE 802.11 specification does not define any of this type of combination frame as described in the paper. It is shown that the cross-layer and merging of the frame can significantly improve the Wi-Fi environment. In case of power saving of device, DHCP lease offloading can also be performed using the approach presented in the paper.

References

1. Seneviratne, S., et al.: Characterizing wifi connection and its impact on mobile users: practical insights. In: WiNTECH, pp. 81–88 (2013)
2. Pei, C., et al.: Why it Takes so Long to Connect to a Wi-Fi Access Point?, Cornell University Library, January 2017
3. Cisco vnireprot 2016. https://goo.gl/eqy2s2
4. Ieee standard 802.11. IEEE Std, 802:11 (2016)
5. US Patent US9204473B2 "Method and apparatus for accelerated link setup (2015)
6. Syahputri, R., Sriyanto, S.: Fast and secure authentication in IEEE 802.11i wireless LAN. In: 2012 2nd International Conference on in Uncertainty Reasoning and Knowledge Engineering (URKE), pp. 158–161 (2012)
7. Zuquete, A., Frade, C.: Pre-allocation of DHCP leases: a cross-layer approach. In: 4th IFIP International Conference on New Technologies, Mobility and Security https://doi.org/10.1109/NTMS.2011.5720663
8. Velayos, H., et al.: Techniques to reduce IEEE 802.11b mac layer handover time. Technical report (2003)
9. Ashraf, F., Kravets, R.H.: Making dense networks work for you. In: 2015 24th International Conference on Computer Communication and Networks (ICCCN), pp. 1–8 (2015)
10. Cicconetti, C., Galeassi, F., Mambrini, R.: Network-assisted handover for heterogeneous wireless networks. In: 2010 IEEE GLOBECOM Workshops (GC Wkshps), pp. 1–5 (2010)
11. Ieee standard 802.11. IEEE Std, 802:11r
12. Jia, J., Liu, G., Han, D., wang, J.: A Novel Packets Transmission Scheme Based on Software Defined Open Wireless Platform. Digital Object Identifier https://doi.org/10.1109/ACCESS.2018.2813007.
13. whitepaper Intel® Centrino® Mobile Technology Wake on Wireless LAN (WoWLAN) Feature
14. https://learningnetwork.cisco.com/thread/118328
15. Wu, A.H.: The Development of WDM Device Drivers Under Windows 2000/XP. Electronic Industry Press, Beijing (2005)
16. Walter, O.: Programming the Microsoft Windows Driver Model. Microsoft Press, America (1999)
17. Microsoft Windows 10 driver development kit documentation (2017)
18. https://docs.microsoft.com/en-us/sysinternals/downloads/debugview
19. Balachandran, A., Voelker, G., Bahl, P., Rangan, V.: Characterizing user behavior and network performance in a public wireless LAN. In Proceedings of ACM SIGMETRICS, Marina Del Rey, CA, June 2002

20. Balasubramanian, N., Balasubramanian, A., Venkataramani, A.: Energy consumption in mobile phones: a measurement study and implications for network applications. In: Proceedings of ACM IMC (2009)
21. https://www.cypress.com/file/139866/download