Qualcomm Technologies, Inc.

802.11ax Overview and Features Test Setup

Test Guide

80-WL521-105 Rev. C

August 11, 2020

For additional information or to submit technical questions, go to https://createpoint.qti.qualcomm.com

Confidential – Qualcomm Technologies, Inc. and/or its affiliated companies – May Contain Trade Secrets

Qualcomm Technologies, Inc.
5775 Morehouse Drive
San Diego, CA 92121
U.S.A.

Revision history

Revision	Date	Description
A	May 2019	Initial release
B	September 2019	Numerous updates to the content. Read the document in its entirety. ■ Added a table in Chapter 1. ■ Updated the table in section 3.3, TWT test procedures. ■ Deleted section 3.3.2 (of earlier version), Wi-Fi calling with background traffic. ■ Updated steps in the new section 3.3.2, Congestion trigger. ■ Updated section 3.3.4, TWT air packet verification.
C	August 2020	Updated sections 4.3, 4.4, 5.1.1 and 5.1.3 Added sections 4.6 and 4.7

Contents

Figures

Tables

1 Introduction to 802.11ax features and test setup

This document provides an overview of 802.11ax and the test setup to verify Qualcomm® 802.11ax features, including target wake time (TWT), 8 × 8 sounding, WPA3,and OFDMA. The document also includes test cases for reference.

The information in this document is based on Qualcomm WCN3998 and QCA639X WLAN chip solutions.

Feature	WCN3998	QCA6390
802.11ax	DL MU-MIMO 8 Stream Sounding TWT WPA3	UL/DL MU-MIMO UL/DL OFDMA Multi-BSSID and Multi-TID Spatial Reuse 8 Stream Sounding TWT WPA3

2 802.11ax overview

This chapter provides background details on the RF-related and protocol-related bottlenecks of 802.11a/b/g/n/ac, 802.11ax goals, 802.11ax PHY layer feature, and the multiuser (MU) option.

2.1 Bottlenecks of 802.11a/b/g/n/ac

The 802.11n and 802.11ac standards have improved PHY and MAC to achieve a higher data rate. However, with the number of Wi-Fi devices growing, it is observed that the radio frequency (RF) and protocol bottlenecks can cause low throughput when a device is in a high-density environment. Increasing the speed of transmission does not help to solve the issue.

2.1.1 RF-related bottlenecks

The RF-related bottlenecks are as follows:

- RF spreads evenly everywhere.
- RF transmitter sends signals to all directions and receiver tries to receive them from all directions.
- Nearby devices transmit static high-power level.
- Half-duplex cannot receive during transmission.
- The design does not consider outdoor range.

2.1.2 Protocol-related bottlenecks

The protocol-related bottlenecks are as follows:

- CCA protocol is overprotective.
- Retransmissions are inefficient and use a lot of airtime.
- Control and management traffic takes much airtime from the user data.
- Legacy device protection reduces the network capacity.
- Channel access congested with large amount of devices.

With the above drawbacks, the performance degrades in a congested environment. In a dense environment, lesser aggregation is observed, and airtime wastage occurs due to AIFS, preambles and Non-linear growth in control frames such as NDPs, ACK, and RTS/CTS.

2.2 802.11ax goal

The 802.11ax goal aims to solve the degradation of system efficiency due to a growing density of Wi-Fi devices and network with numerous small data frames.

It is designed for improving the capacity, providing better coverage, and reducing congestion to get better user experience overall. So, the 802.11ax goal aims to achieve the following capabilities:

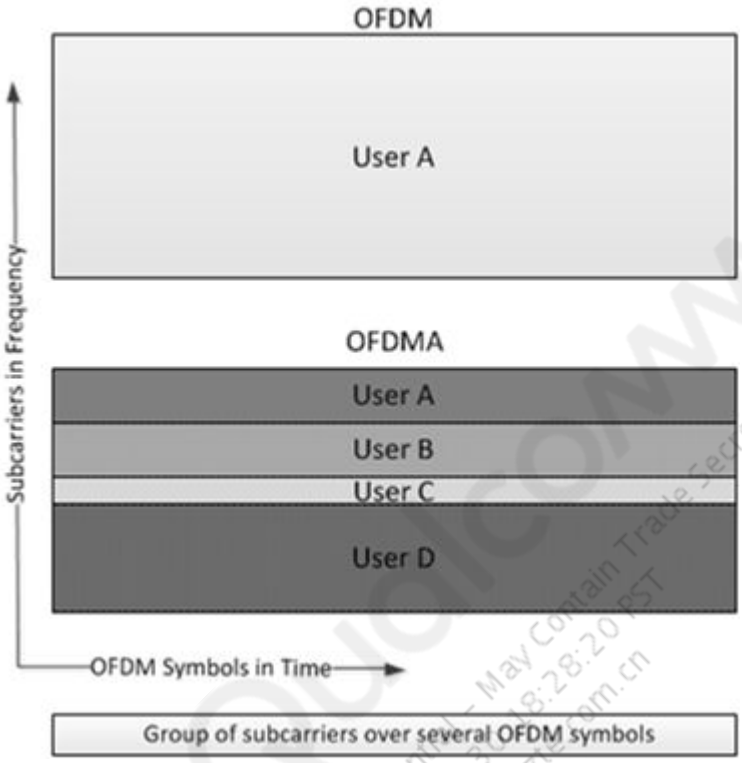
- Improve average throughput per station by four times in a high-density environment (OFDMA)
- Increase spectral frequency reuse and manage interference in a dense deployment (OFDMA)
- Increase robustness in the outdoor propagation environments and uplink transmission (OFDMA)
- Support multiple simultaneous communications in both spatial and frequency domains for both uplink and downlink (OFDMA)
- Improve power efficiency (TWT)
- Ensure backward compatibility with legacy IEEE 802.11 devices
- Operate between 1 GHz and 6 GHz
- Limit outdoor operation to pedestrian and stationary (OFDM enhancement)
- Use 1024-QAM modulation

2.3 OFDMA

OFDMA is an OFDM-based multiple access scheme where different subsets of subcarriers are allocated to different users, allowing simultaneous data transmission to or from one or more users.

It subdivides a Wi-Fi channel into small frequency allocations called resource units (RUs). The AP can communicate (uplink, downlink) with many clients assigned to specific RUs.

Similar to OFDM, OFDMA employs multiple subcarriers, but the subcarriers are divided into several groups of subcarriers where each group is denoted as an RU. With OFDMA, different transmit powers could be applied to different RUs.



2.3.1 Resource units

The resource units (RUs) defined for DL and UL transmission are as follows: 26-tone RU, 52-tone RU, 106- tone RU, 242-tone RU, 484-tone RU, 996-tone RU, and 2x996-tone RU. The maximum number of RUs in the 20 MHz, 40 MHz, 80 MHz, 160 MHz, and 80+80 MHz HE PPDU formats are defined as illustrated here.

RU type	CBW20	CBW40	CBW80	CBW160 and CBW80+80
26-subcarrier RU	9	18	37	74
52-subcarrier RU	4	8	16	32
106-subcarrier RU	2	4	8	16
242-subcarrier RU	1-SU/MU-MIMO	2	4	8
484-subcarrier RU	N/A	1-SU/MU-MIMO	2	4
996-subcarrier RU	N/A	N/A	1-SU/MU-MIMO	2
2x996 subcarrier RU	N/A	N/A	N/A	1-SU/MU-MIMO

Figure 2-1 Maximum number of RUs for each channel's width

For an OFDMA HE PPDU transmission, 20 MHz bandwidth consists of 256 subcarriers, 40 MHz bandwidth consists of 512 subcarriers, 80 MHz bandwidth consists of 1024 subcarriers. See the following diagram for details.

These subcarriers can be grouped into smaller subchannels named as resource units (RUs).

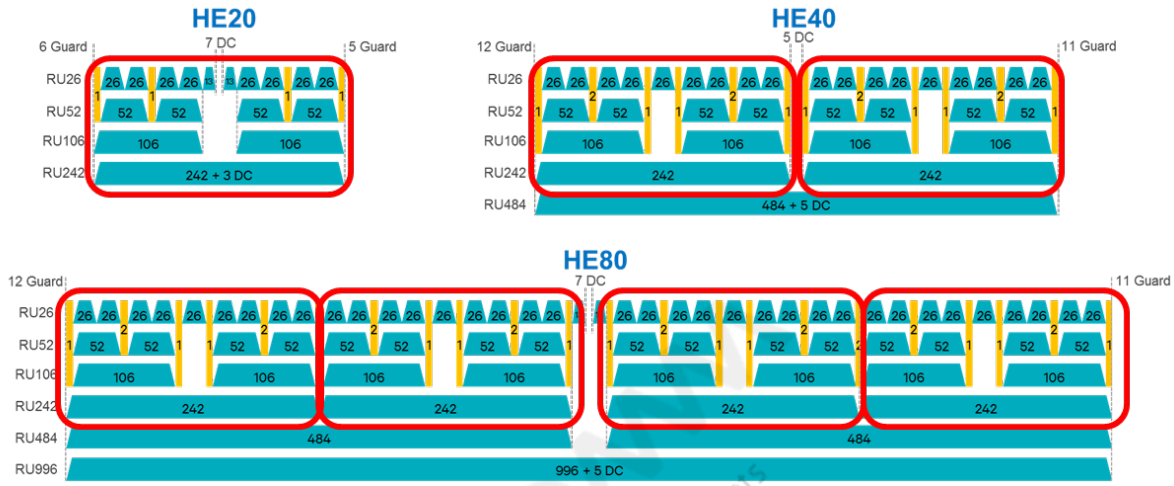


Figure 2-2 Resource Units

Using the OFDMA feature, the 802.11ax AP arranges different combinations of RUs that can be used by the client. The AP can allocate the whole channel to one client at a time or it can partition the channel to serve multiple clients simultaneously.

2.4 802.11ax PHY layer feature

IEEE 802.11ax defines the following four different efficient PPDU formats:

- High-efficiency (HE) SU PPDU
- HE MU PPDU
- HE triggers-based (TB) PPDU
- HE extended range (ER) SU PPDU

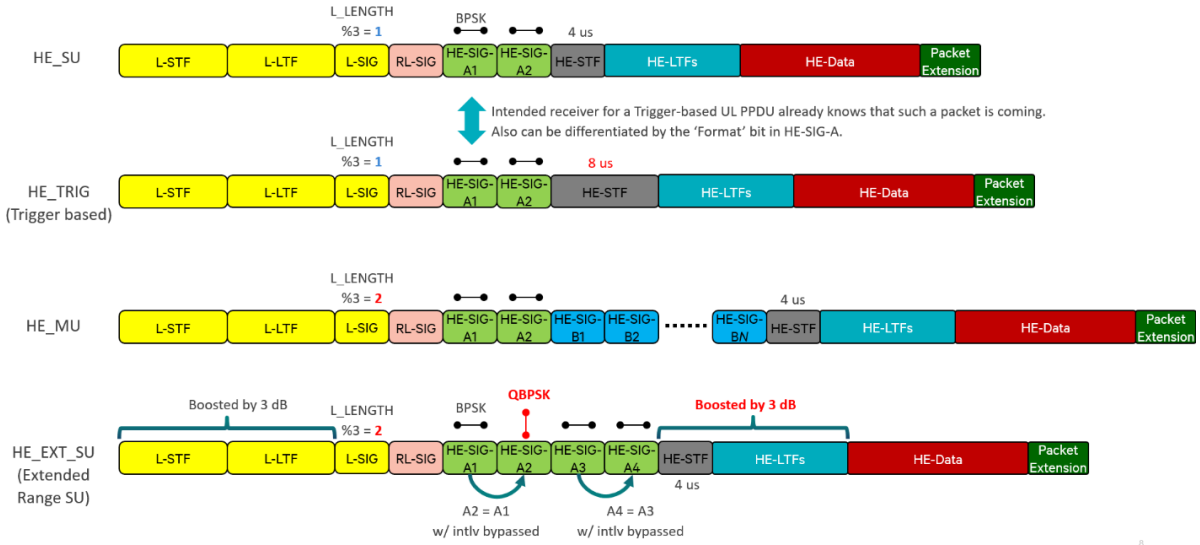


Figure 2-3 802.11ax HE PPDU

HE-SU PPDU

The single user format is the packet structure format between the AP and one single STA, and between a single STA and another single STA. Figure 2-3 shows the packet structure. Compared to IEEE 802.11ac, HE SU PPDU introduces Repeat L-SIG (RL-SIG), which is used to enhance the robustness of L-SIG and used to confirm that a PPDU is in the HE format through automatic detection. Packet extension (PE) extends the time for the receiver to process data.

HE-TRIG PPDU

It is used for triggered multiuser transmission from one or more stations. After receiving the trigger frame, multiple STAs simultaneously transmit UL frames according to the resource allocation information in the trigger frame. Compared to HE MU PPDU, this format does not have HE-SIG-B because the AP indicates the resource allocation information in the trigger frame. See Figure 2-3 to find out the packet structure.

HE-EXT-SU PPDU

To improve the transmission robustness of the outdoor scenario, IEEE 802.11ax repeats the HE-SIG-A field by directly extending two symbols to four symbols, as shown in Figure 2-3. Moreover, IEEE 802.11ax boosts the power of the traditional preamble, HE-STF and HE-LTF to further extend the transmission coverage. Finally, the transmission range of the data field can be extended through DCM narrowband RU transmission.

HE-MU PPDU

The MU format enables simultaneous transmission among MUs through OFDMA and/or MU-MIMO. Based on the single user format, as shown in Figure 2-3, the HE-SIG-B field is added to indicate the resource allocation information for multiple users.

2.5 Multiuser option

IEEE 802.11ax enhances the multiple access technology based on OFDMA and UL MU-MIMO (DL MU-MIMO is used in IEEE 802.11ac) to guarantee the MU parallel transmission in both frequency domain and spatial domain.

An 802.11ax AP can also combine MU-MIMO with the OFDMA operation. The OFDMA feature enables multiuser access by subdividing a channel. MUMIMO enables multiuser access by using different spatial streams.

The most important enhancement for the MAC layer of IEEE 802.11ax is the enhancement of MU-MAC. MUMAC is a type of high-efficiency multiple access technology, which enables multiple users to follow certain access rules to transmit UL data concurrently (known as UL MUMAC), or to transmit DL data concurrently (known as DL MUMAC) through the given network access resources (that is space domain, time domain, and frequency domain resources).

IEEE 802.11ax proposes a cascaded MUMAC, allowing the DL MU transmission and the UL MU transmission to occur alternately in a TxOP time duration, which further improves the MAC efficiency.

2.5.1 Multiuser MIMO

DL MU-MIMO was introduced by 802.11ac. An 802.11ac AP uses the beamforming techniques to direct packets simultaneously to spatially diverse users. The AP communicates with each client and steers simultaneous beams to different clients, each beam containing specific packets for its target user. Downlink MU-MIMO by steering the beams of different data streams to different clients.

802.11ax further introduces UL MU-MIMO to ensure symmetrical high-throughput for both DL and UL. IEEE 802.11ax allows up to eight STAs to transmit simultaneously through MU-MIMO, as shown in Figure 2-5.

MU-MIMO belongs to the spatial domain multiple access technology. It allocates different spatial streams for different users, and the receiver must separate the spatial streams of different users by using a signal processing technology as shown in Figure 2-4.

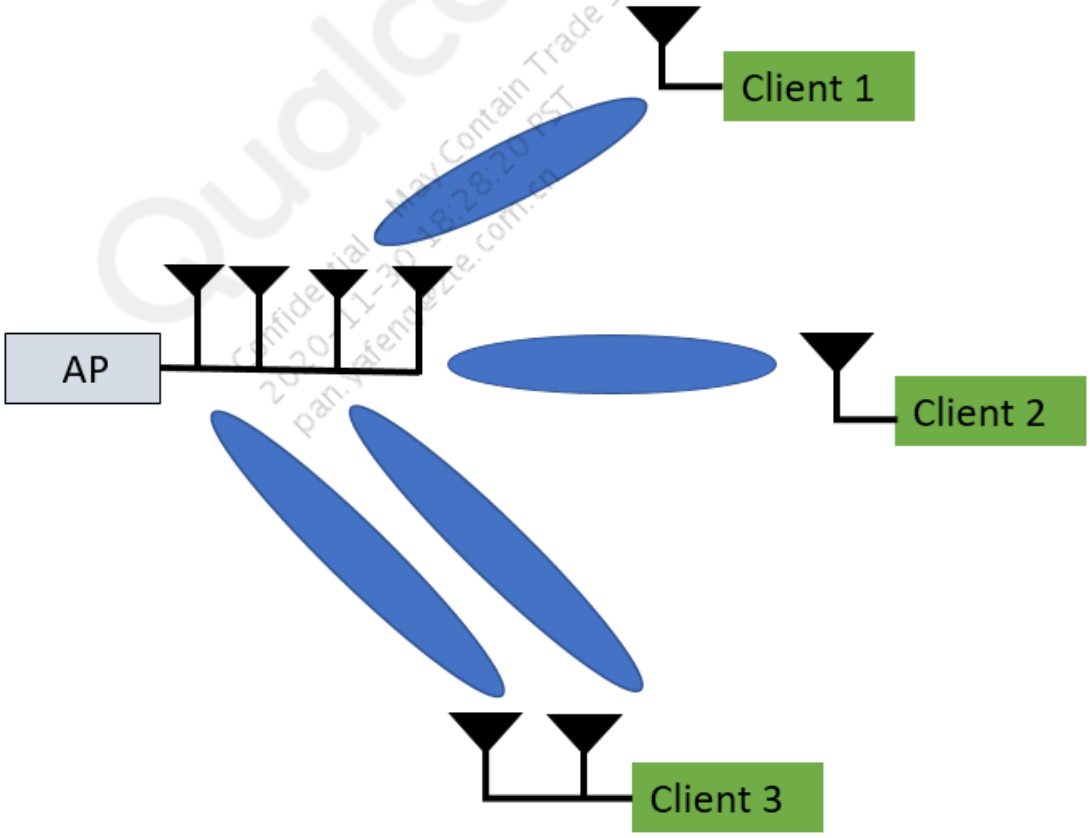


Figure 2-4 Example illustration for DL/UL MU-MIMO beamforming

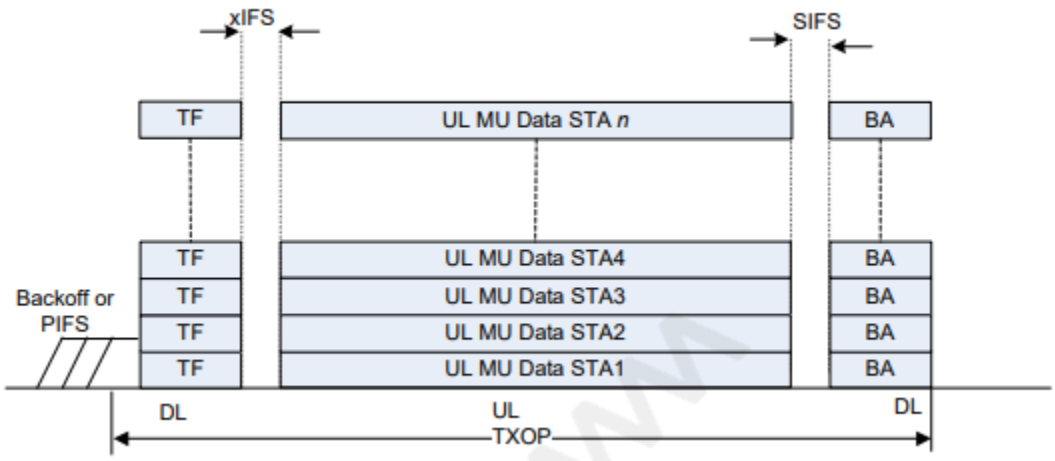


Figure 2-5 UL MU-MIMO

2.5.2 OFDMA

IEEE 802.11ax introduces OFDMA that belongs to the frequency domain multiple access technology, dividing the channel(s) into several resource units (RUs) with same or different bandwidths. Each STA is supported in one RU for UL or DL transmission. By enabling parallel transmission, OFDMA reduces the overhead and collision, and further enhances spectrum efficiency.

2.5.3 OFDMA plus MU-MIMO

802.11ax allows MU-MIMO and OFDMA to work together. Multiple STAs can parallelly send or receive frames in the same RU through MU-MIMO, which further increases transmission efficiency.

For DL MU-MIMO, the AP uses the HE MU PPDU format to support both OFDMA and MU-MIMO, which is more flexible than IEEE 802.11ac. Spatial stream and RUs allocation information are embedded in the HE-SIG-B, and then the AP simultaneously transmits frames to multiple STAs through allocated spatial streams.

For UL MU-MIMO, the AP uses trigger frame (TF) to trigger multiple STAs and allocate RUs and spatial streams. After that, the STAs triggered by the AP use the HE TB PPDU format to simultaneously transmit frames to the AP through allocated spatial streams.

3 Target wake time

This chapter details the TWT feature, the test setup required for performing TWT tests, and the TWT test procedures.

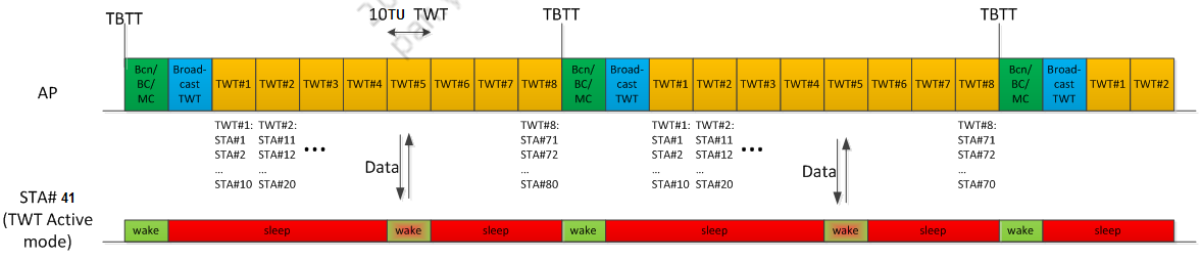
3.1 TWT introduction

The target wake time (TWT) feature is largely deployed in 802.11ax as a solution for the dense environment and it optimizes the power consumption.

TWT defines a mechanism where a specific time or set of times is used, for individual station devices (STAs), to wake up and exchange frames with the AP. TWT offers the following benefits:

- Centralize Tx/Rx for a group of STAs, and thus minimize collisions in an 802.11ax dense deployment environment.
- Because collisions in a congested environment leads to large amount of wasted power, a reduction of contention in a densely populated medium can save power.

With TWT, the aggregated throughput and per client power consumption is significantly improved in a 802.11ax dense environment. The following diagram demonstrates the basic idea about how TWT functions:



The following topics of this chapter describe the test setup and test cases for basic functionalities of (TWT) STA.

3.2 TWT test setup details

The following list of devices is used for the TWT test:

Role	Device
TWT STA (DUT)	SM8250 MTP with QCA639X RF card
TWT AP	IPQ8074 AP

Role	Device
	Build used: ■ IPQ8074.ILQ.9.0-00063-P-1 ■ WLAN.HK.1.1-00409-QCAHKSWPL_SILICONZ-1

	Open the serial connection to configure the AP (the following is an example):
	`uci set wireless.wifi0.channel='36'`
	`uci set wireless.wifi0.htmode='HT80'`
	`uci set wireless.wifi0.disabled='0'`
	`uci set wireless.@wifi-iface[0].ssid='NPRPERF5G'`
	`uci set wireless.wifi1.channel='1'`
	`uci set wireless.wifi1.htmode='HT20`
	`uci set wireless.wifi1.disabled='0'`
	`uci set wireless.@wifi-iface[1].ssid='NPRPERF2G'`
	`uci set wireless.@wifi-iface[0].encryption='psk2'`
	`uci set wireless.@wifi-iface[0].key='12345678'`
	`uci set wireless.@wifi-iface[1].encryption='psk2'`
	`uci set wireless.@wifi-iface[1].key='12345678'`
	`uci commit`
	`reboot`

3.3 TWT test procedures

This section provides details on the trigger source and trigger application type with corresponding procedures.

Table 3-1 TWT test case overview

Trigger source	Trigger application type	Description
Firmware trigger	Wi-Fi calling with bidirectional ping	In the firmware trigger mode, enable/disable VoWifi call. Check whether the TWT session is created or tear down accordingly.
	Congestion trigger	In the firmware trigger mode, see whether the TWT session is created or tear down when congestion level is greater or lesser than the maximum or minimum threshold trigger values.
	Duty cycle tuning	In the firmware trigger mode, see if TWT session is renegotiating the session parameters based on the Tx/Rx traffic.

3.3.1 Wi-Fi calling with bidirectional ping

1. Connect the DUT to the AP.
2. Enable the FW-triggered TWT on the DUT by using the following command:

   `iwpriv wlan0 setUnitTestCmd 77 3 0x2 0 100`

   NOTE This is required if the TWT is not configured in the INI configuration.
3. Enable the Wi-Fi calling mode on the DUT by using the following command:

   `iwpriv wlan0 setUnitTestCmd 35 2 201 1`

4. Run the continuous ping traffic with an interval of 30 milliseconds from both of DUT and the AP sides to mimic the Wi-Fi calling status.
5. Monitor the ping traffic status.
6. Tear down the Wi-Fi calling session to end the call by using the following command:

   ```
   iwpriv wlan0 setUnitTestCmd 35 2 201 0
   ```

7. Check if any ping packet is lost during the Wi-Fi calling session.

   NOTE When an actual Wi-Fi calling is started, the firmware detects it and the TWT automatically gets triggered, and automatically tears down when the call is terminated.

3.3.2 Congestion trigger

1. Connect the DUT to the AP.
2. Enable the FW-triggered TWT on DUT by using the following command:

   ```
   iwpriv wlan0 setUnitTestCmd 77 3 0x2 0 100
   ```

   NOTE This is required if TWT is not configured in the INI configuration.

3. A congestion trigger happens automatically if the threshold values are hit. The trigger depends on the present congestion level of the test environment. The following command configures the DUT to start the TWT session when the congestion level is higher than 60, and to tear down the TWT session when the congestion level is lower than 40:

   ```
   iwpriv wlan0 setUnitTestCmd 77 3 0x9 60 40
   ```

   Always enabled:

   ```
   iwpriv wlan0 setUnitTestCmd 77 3 0x9 0 0
   ```

   ```
   iwpriv wlan0 setUnitTestCmd 77 3 0x9 1
   ```

4. To trigger the TWT session by the congestion trigger with congestion high threshold value 60 in the preceding example, perform the following steps:
   a. In the shield room, prepare two APs in the same channel, DUT, and any STA.
   b. Start traffic VHT80 UDP UL or DL.

      The TWT session starts.

5. To stop the TWT session by the congestion trigger with congestion low threshold value 40 in the preceding example, perform the following steps:
   a. Stop the traffic between other AP and STA.

      The TWT session tears down on the DUT.

6. Check the firmware log and air packet log to confirm if the TWT session is started and tore down based on the test environment congestion status. If **yes**, then it is passed.

   The following is the example for checking firmware log messages:

   ```
   1129300 [4548/0001] MSG 00:21:29.928 WLAN MACCORE/Medium
   [                sm.c    228]
   [SM:RcvEvt] CurS:0 <-- Event:0 mod_id=77 vapid =0 make_sm = 0x1000000
   extra = 0x7 extra2 =
   ```

```
0x13005 extra3 = 0x0

1129305 [4548/0001] MSG 00:21:29.928 WLAN MACCORE/Medium
[                sm.c    367]
[SM:ChgSta] CurS:0 --> NextS:1 mod_id=77 vapid =0 make_sm = 0x100 extra =
0x407 extra2 = 0x1302d
extra3 = 0x0
```

- TWT setup happens because outside congestion increases above upper threshold. TWT session state is “INIT” -> “RUNNING” due to “SETUP_DONE” event.

```
1116412 [4548/0001] MSG 00:21:20.322 WLAN MACCORE/Medium
[                sm.c    228] [SM:RcvEvt] CurS:1 <
-Event:15 mod_id=77 vapid =0 make_sm = 0x1011500 extra = 0x10406 extra2
= 0x1306d extra3 = 0x0

1116413 [4548/0001] MSG 00:21:20.322 WLAN MACCORE/Medium
[                sm.c    367] [SM:ChgSta] CurS:1 --
> NextS:6 mod_id=77 vapid =0 make_sm = 0x10600 extra = 0x10406 extra2 =
0x1306d extra3 = 0x0

1116414 [4548/0001] MSG 00:21:20.322 WLAN MACCORE/Medium
[                sm.c    228] [SM:RcvEvt] CurS:6 <-
-Event:16 mod_id=77 vapid =0 make_sm = 0x1061601 extra = 0x10406 extra2
= 0x1306d extra3 = 0x0

1116423 [4548/0001] MSG 00:21:20.325 WLAN MACCORE/Medium
[                sm.c    228] [SM:RcvEvt] CurS:6 <-
-Event:17 mod_id=77 vapid =0 make_sm = 0x1061700 extra = 0x1010406
extra2 = 0x1306d extra3 = 0x0

1116432 [4548/0001] MSG 00:21:20.326 WLAN MACCORE/Medium
[                sm.c    367] [SM:ChgSta] CurS:6 --
> NextS:7 mod_id=77 vapid =0 make_sm = 0x60700 extra = 0x1100406 extra2
= 0x1302d extra3 = 0x0
```

- TWT tear down occurs because the congestion (outside DUT) has fallen below the lower threshold value. The TWT session state is “RUNNING” > “DELETING” due to the “DELETE_REQ” event. The TWT session state is “DELETING” > “DELETED” due to the “DELETE_TX” and “DELETE_TX_OK” events.

3.3.3 Duty cycle tuning

1. Connect the DUT to the AP.
2. Enable the FW-triggered TWT on the DUT by using the following command:

```
iwpriv wlan0 setUnitTestCmd 77 3 0x2 0 100
```

NOTE This step is required if TWT is not configured in the INI configuration.

3. Configure the congestion lower and upper thresholds to have the TWT session on the DUT triggered by the congestion level of the testing environment. The following command configures the DUT to start the TWT session when cthe ongestion level is higher than 20, and to tear down the TWT session when the congestion level is lower than 10:

   ```
   iwpriv wlan0 setUnitTestCmd 77 3 0x9 20 10
   ```

4. Ensure that no Tx/Rx traffic goes on the DUT.
5. Check the firmware log and air packet log to confirm if the TWT session is paused, because the outer congestion level is higher than the upper threshold value but there is no Tx/Rx traffic on the DUT. If the DUT does not function as per the expectation, then, it results in failure.

   The following is the example for checking firmware log messages:

   ```
   1077255 [4548/0001] MSG 00:20:52.604 WLAN MACCORE/Medium
   [                sm.c    228]
   [SM:RcvEvt] CurS:1 <-- Event:b mod_id=77 vapid =0 make_sm = 0x1010b00
   extra = 0x10406 extra2 =
   0x1306d extra3 = 0x0

   1077256 [4548/0001] MSG 00:20:52.604 WLAN MACCORE/Medium
   [                sm.c    367]
   [SM:ChgSta] CurS:1 --> NextS:3 mod_id=77 vapid =0 make_sm = 0x10300 extra
   = 0x10406 extra2 =
   0x1306d extra3 = 0x0

   1077257 [4548/0001] MSG 00:20:52.604 WLAN MACCORE/Medium
   [                sm.c    228]
   [SM:RcvEvt] CurS:3 <-- Event:c mod_id=77 vapid =0 make_sm = 0x1030c01
   extra = 0x10406 extra2 =
   0x1306d extra3 = 0x0

   1077264 [4548/0001] MSG 00:20:52.605 WLAN MACCORE/Medium
   [                sm.c    228]
   [SM:RcvEvt] CurS:3 <-- Event:d mod_id=77 vapid =0 make_sm = 0x1030d00
   extra = 0x410406 extra2 =
   0x1306d extra3 = 0x0

   1077273 [4548/0001] MSG 00:20:52.608 WLAN MACCORE/Medium
   [                sm.c    367]
   [SM:ChgSta] CurS:3 --> NextS:4 mod_id=77 vapid =0 make_sm = 0x30400 extra
   = 0x400406 extra2 =
   0x1302d extra3 = 0x0
   ```

   NOTE Due to the lack of enough traffic, the TWT session state is “INIT” > “PAUSING” due to the “PAUSE_REQ” event. Then, the state is “PAUSING” > “PAUSED” due to “PAUSE_TX” and “PAUSE_TX_OK” events.

6. Pump the traffic that is heavy enough to trigger the TWT session resume.

7. Check the firmware log and air packet log to confirm if the TWT session is resumed because there is enough traffic on the DUT. If **yes**, then, it is passed.

   The following is the example for checking firmware log messages:

```
1112345 [4548/0001] MSG 00:21:17.346 WLAN MACCORE/Medium
[                sm.c    228]
[SM:RcvEvt] CurS:4 <-- Event:10 mod_id=77 vapid =0 make_sm = 0x1041000
extra = 0x40406 extra2 =
0x1301d extra3 = 0x1fea94fa

1112346 [4548/0001] MSG 00:21:17.346 WLAN MACCORE/Medium
[                sm.c    367]
[SM:ChgSta] CurS:4 --> NextS:5 mod_id=77 vapid =0 make_sm = 0x40500 extra
= 0x40406 extra2 =
0x1301d extra3 = 0x0

1112347 [4548/0001] MSG 00:21:17.346 WLAN MACCORE/Medium
[                sm.c    228]
[SM:RcvEvt] CurS:5 <-- Event:11 mod_id=77 vapid =0 make_sm = 0x1051101
extra = 0x40406 extra2 =
0x1301d extra3 = 0x1fea94fa

1112354 [4548/0001] MSG 00:21:17.346 WLAN MACCORE/Medium
[                sm.c    228]
[SM:RcvEvt] CurS:5 <-- Event:12 mod_id=77 vapid =0 make_sm = 0x1051200
extra = 0x840406 extra2 =
0x1301d extra3 = 0x0

1112359 [4548/0001] MSG 00:21:17.346 WLAN MACCORE/Medium
[                sm.c    367]
[SM:ChgSta] CurS:5 --> NextS:1 mod_id=77 vapid =0 make_sm = 0x50100 extra
= 0x800406 extra2 =
0x1302d extra3 = 0x0
```

   NOTE Because of the traffic on the DUT that is heavy enough to trigger TWT resume, the TWT session state is “PAUSED” > “RESUMING” due to the “RESUME_REQ” event. And then the state is “RESUMING” > “RUNNING” due to “RESUME_TX” and “RESUME_TX_OK” events.

3.3.4 TWT air packet verification

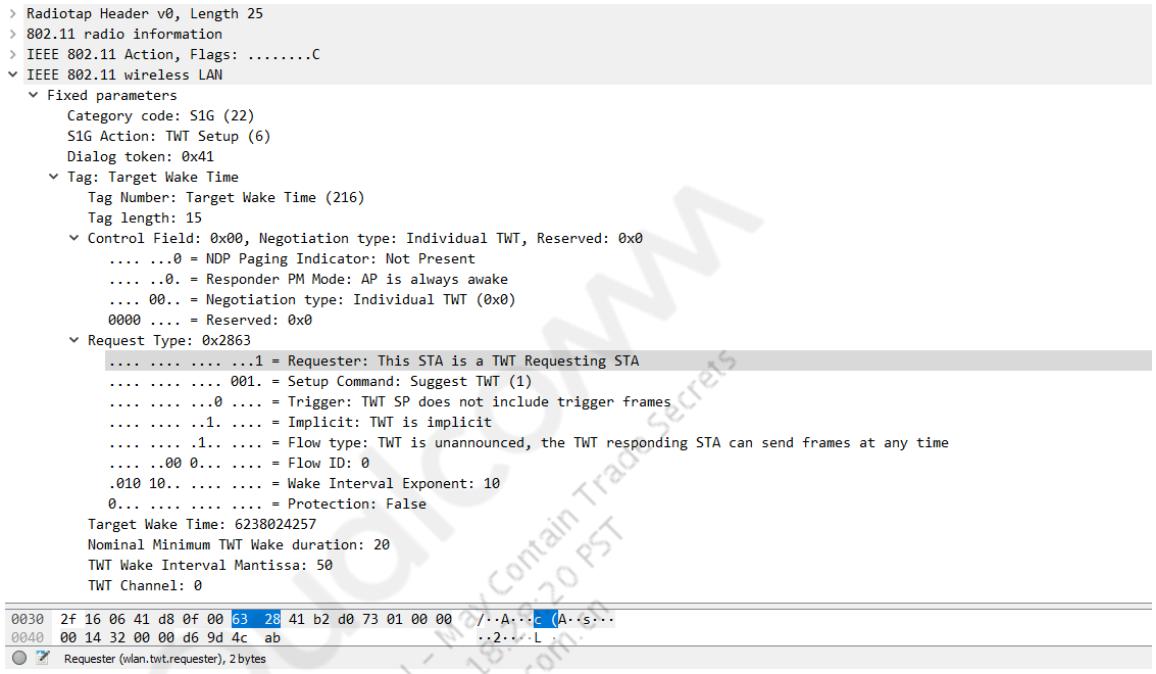
Prerequisites:

The Wireshark version later than 2.9.0 can parse the TWT packets.

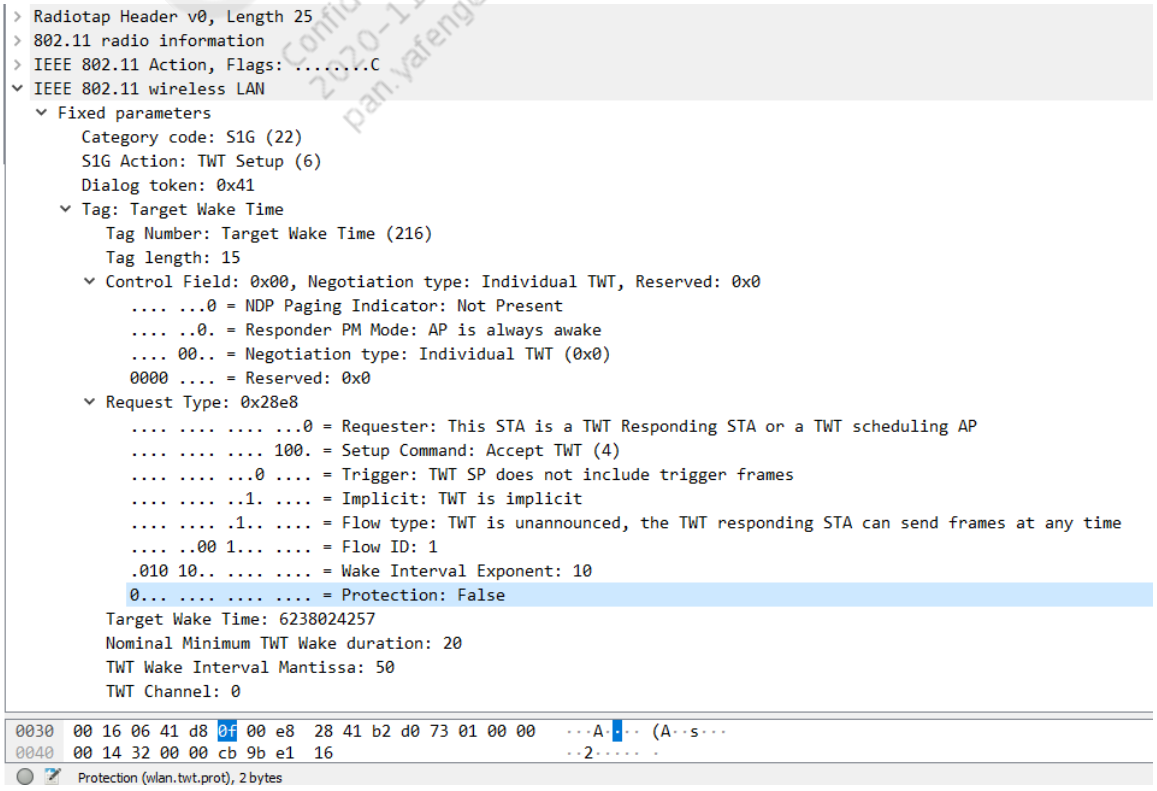
To check the TWT air packets, perform the following steps:

1. Check if the TWT setup packet exchange takes place between the DUT and AP, which uses the **TWT Setup** action type.

   Request packet:



   Response packet:



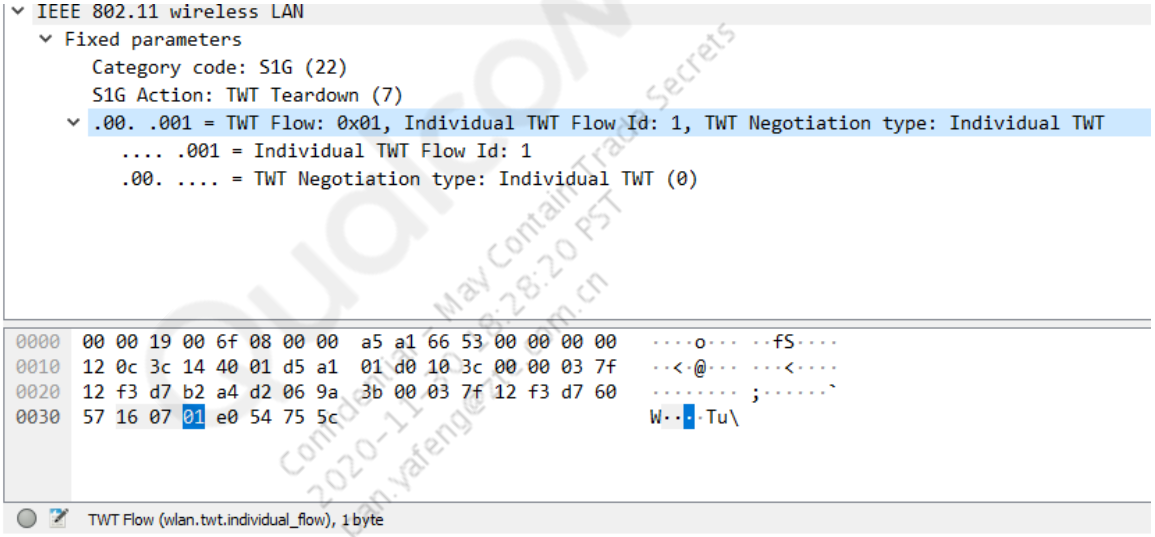
2. Check the TWT pause or resume packets—that are sent by the DUT—which use the **TWT Information** action type.

   Pause or resume packet:

```
IEEE 802.11 wireless LAN
  Fixed parameters
      Category code: S1G (22)
      S1G Action: TWT Information (11)
    Tagged parameters (1 bytes)
```

3. Check the TWT delete packets—that are sent by DUT—which use the **TWT Teardown** action type.

   Teardown packet:



4 WPA3

This chapter provides details on the following topics:

- Introduction to WPA3
- Common test setup for WPA3
- WPA3 OWE
- WPA3 SAE
- WPA3-Suite B

4.1 WPA3 introduction

WPA3 provides more robust security protection as research suggests that WPA2 is not fully secured.

WPA3 has the following technology with improved security:

- Opportunistic Wireless Encryption (OWE)–replace open unencrypted networks.
- Simultaneous Authentication of Equals (SAE)–replace WPA2-PSK to mitigate attacks against PSK.
- Suite B–stronger link-layer encryption and stronger authentication methods. Suite B uses a set of cryptographic algorithms defined by the United States NSA.

The use of open unencrypted wireless networks presents a huge security risk from passive packet capture and sniffing. The purpose of OWE is to mitigate attacks on open unencrypted wireless networks that pose the following issues:

- Present significant security threats to users.
- Provide encryption of the wireless medium but without authentication.

With OWE, the client and AP perform a Diffie-Hellman key exchange during the access procedure and use the resulting pairwise secret with the four-way handshake instead of using a shared and public PSK in the four-way handshake. When the DUT finds the OWE AP available, it triggers OWE connection.

SAE is a secure and password-based key-exchange mechanism for mutual authentication between AP and STA. SAE provides authentication and data security protection. The legacy WPA PSK provides data security protection only. SAE is resistant to off-line dictionary attacks.

The following topics in this chapter describe the test setup and test cases for basic functionalities of WPA3 features.

4.2 WPA3 common test setup

This section provides details on the network topology, test setup device requirements, AP serial console connection, and AP common configuration.

4.2.1 Network topology for WPA3

The following diagram illustrates the network topology for WPA3:

The RADIUS server is only for WPA3 Enterprise security mode. It is not required for Simultaneous Authentication of Equals (SAE) and Opportunistic Wireless Encryption (OWE).

4.2.2 Test setup devices for WPA3

See the following table for details on the test setup devices for WPA3:

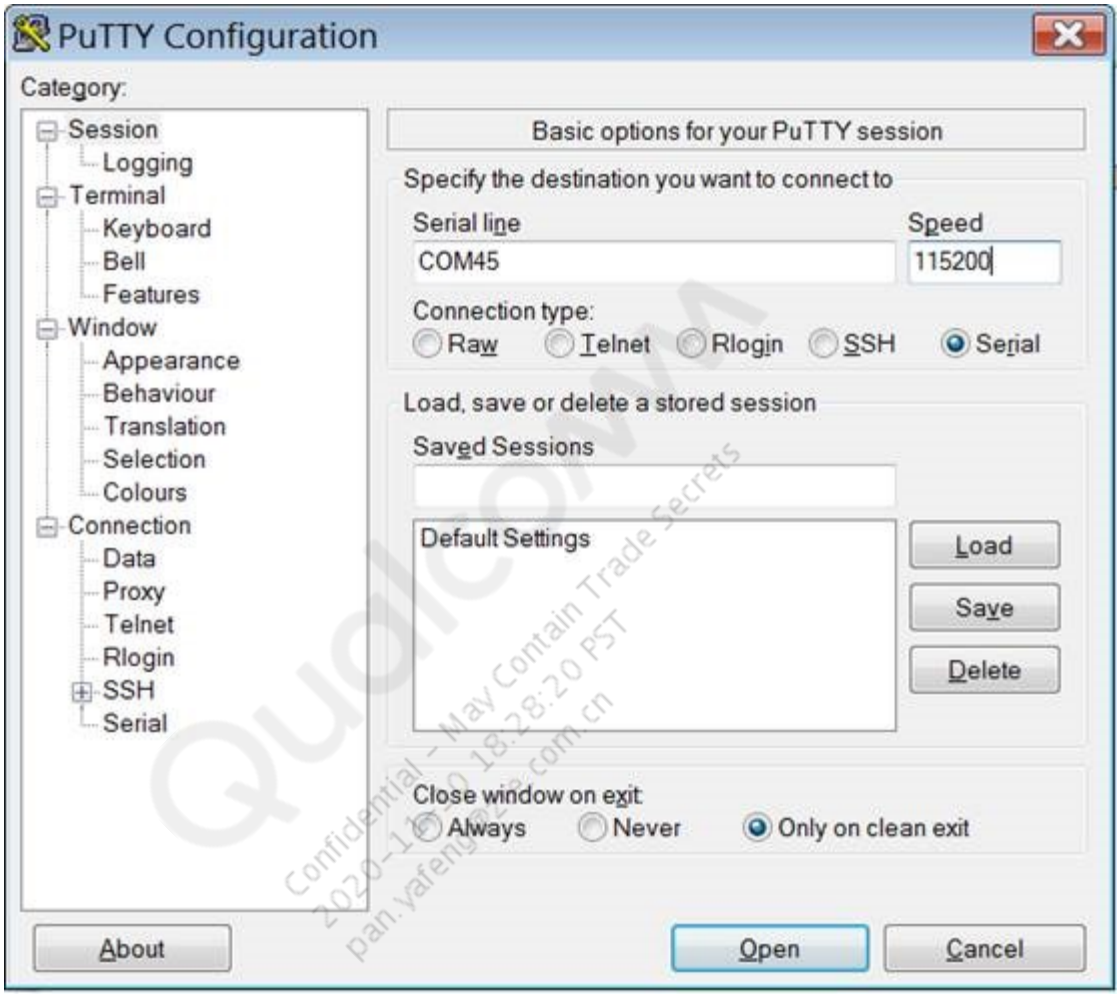
Role	Device
TWT STA (DUT)	SM8250 MTP with QCA639X card
TWT AP	QCA9984 AP
	Build used: IPQ8064.ILQ.6.1.0-00051-P-1

4.2.3 Connect AP serial console for WPA3

To provide an AP serial console connection for WPA3:

1. Connect the AP to the PC by using USB to RS232 (serial) converter for opening a serial session with the AP.
2. Connect an Ethernet cable from the PC to the AP for copying the files and network connectivity.

3. Open a serial port session to the AP by using a console application, for example, putty/Teraterm, with the COM port ID, baud rate 115200.



4.2.4 Configure AP common commands for WPA3

Configure the following AP common commands:

Action	Command

Delete previous config	`wifi detect > /etc/config/wireless`
	`uci commit wireless`
	`uci revert -P /var/state wireless`
	`wifi`

Delete residual interfaces	`uci export wireless`
	`uci show wireless | grep =wifi-iface`
	`uci delete wireless.@wifi-iface[1]`
	`uci delete wireless.@wifi-iface[0]`
	`uci show wireless | grep =wifi-`
	`iface`

Action	Command
Add interfaces	uci add wireless wifi-iface uci set wireless.@wifi-iface[0].network=lan uci set wireless.@wifi-iface[0].mode=ap uci set wireless.@wifi-iface[0].ssid=OpenWrt uci set wireless.@wifi-iface[0].encryption=none
Configure radio	uci set wireless.@wifi-device[0].hwmode=11ac uci set wireless.@wifi-device[0].disabled=0 uci set wireless.@wifi-device[0].htmode=HT80 uci set wireless.@wifi-device[0].channel=149 uci set wireless.@wifi-device[0].txchainmask=15 uci set wireless.@wifi-device[0].rxchainmask=15

4.3 WPA3-OWE

This section provides details on configuring the AP OWE-specific settings, DUT OWE-specific settings, enabling DUT WLAN STA connection, and WPA OWE log verification.

Irrespective of for home use, commercial use, guest access or captive portal, or device onboarding, the use of open unencrypted wireless networks presents a huge security risk from passive packet capture and sniffing. In places such as airports and hotels, internet is offered over open or unencrypted wireless networks.

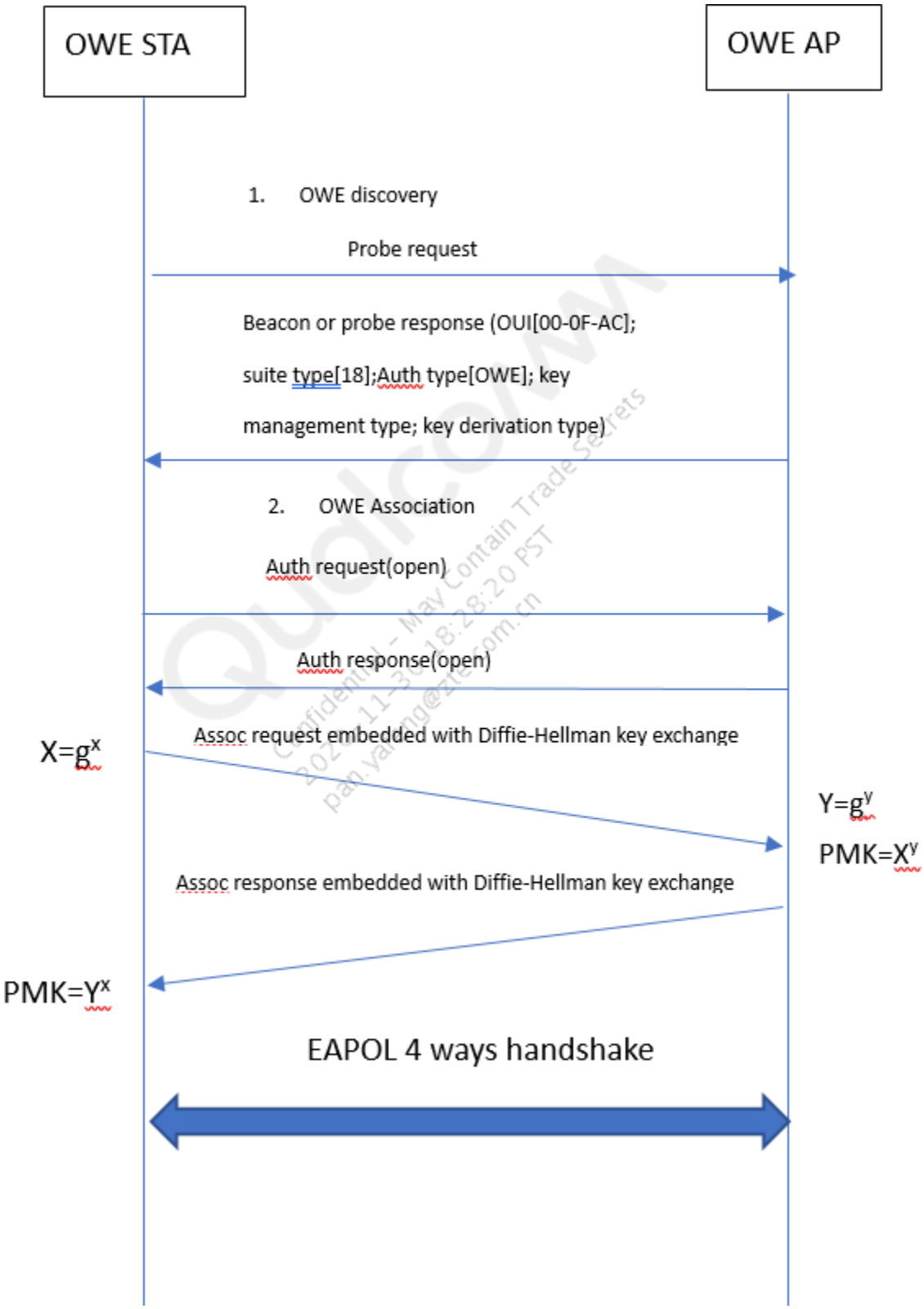
The purpose of OWE is to mitigate attacks that can pose the following risks on open unencrypted wireless networks:

- Present significant security threats to users.
- Provide encryption of the wireless medium but no authentication.

With OWE, the client and AP perform a Diffie-Hellman key exchange during the access procedure. The resulting pairwise secret is used with the four-way handshake instead of using a shared and public PSK in the four-way handshake.

There is no INI currently to enable or disable OWE from the driver. When a supplicant finds a OWE AP available, it triggers a OWE connection.

The following figure illustrates the OWE-based authentication call sequence:



4.3.1 Configure AP OWE-specific settings

To configure AP OWE-specific settings, see the following table:

Security	Command

OWE with CCMP	`uci set wireless.@wifi-iface[0].mode=ap`
	`uci set wireless.@wifi-iface[0].ssid=0000_OWE_5G`
	`uci set wireless.@wifi-iface[0].device=wifi0`
	`uci set wireless.@wifi-iface[0].owe=1`
	`uci set wireless.@wifi-iface[0].encryption=ccmp`
	`uci set wireless.@wifi-iface[0].wpa_group_rekey=60`
	`uci set wireless.@wifi-iface[0].ieee80211w='2'`
	`uci set wireless.@wifi-iface[0].nss=4`
	`uci set wireless.@wifi-iface[0].owe_group=19`
	`uci set wireless.@wifi-iface[0].group_mgmt_cipher=AES-128-CMAC`
	`uci commit`
	`wifi`

OWE with GCMP	`uci set wireless.@wifi-iface[0].mode=ap`
	`uci set wireless.@wifi-iface[0].ssid=0000_OWE_5G`
	`uci set wireless.@wifi-iface[0].device=wifi0`
	`uci set wireless.@wifi-iface[0].owe=1`
	`uci set wireless.@wifi-iface[0].encryption=gcmp`
	`uci set wireless.@wifi-iface[0].wpa_group_rekey=60`
	`uci set wireless.@wifi-iface[0].ieee80211w='2'`
	`uci set wireless.@wifi-iface[0].nss=4`
	`uci set wireless.@wifi-iface[0].owe_group=19`
	`uci set wireless.@wifi-iface[0].group_mgmt_cipher=BIP-GMAC-128`
	`uci commit`
	`wifi`

Security	Command
OWE-transition with GCMP	uci add wireless wifi-iface uci set wireless.@wifi-iface[1].network=lan uci set wireless.@wifi-iface[1].mode=ap uci set wireless.@wifi-iface[1].ssid=OpenWrt uci set wireless.@wifi-iface[1].encryption=none uci set wireless.@wifi-iface[0].mode=ap uci set wireless.@wifi-iface[0].ssid=0000_OWE_Transition_5G uci set wireless.@wifi-iface[0].device=wifi0 uci set wireless.@wifi-iface[0].encryption=gcmp uci set wireless.@wifi-iface[0].nss=4 uci set wireless.@wifi-iface[0].network=lan uci set wireless.@wifi-iface[0].hidden=1 uci set wireless.@wifi-iface[0].owe=1 uci set wireless.@wifi-iface[0].ieee80211w=2 uci set wireless.@wifi-iface[0].group_mgmt_cipher=BIP-GMAC-128 uci set wireless.@wifi-iface[0].owe_transition_ifname=ath01 uci set wireless.@wifi-iface[1].mode=ap uci set wireless.@wifi-iface[1].device=wifi0 uci set wireless.@wifi-iface[1].encryption=none uci set wireless.@wifi-iface[1].nss=4 uci set wireless.@wifi-iface[1].hidden=0 uci set wireless.@wifi-iface[1].network=lan uci set wireless.@wifi-iface[1].ssid=0000_non-OWE_Transition_5G uci set wireless.@wifi-iface[1].owe_transition_ifname=ath0 uci set wireless.@wifi- iface[1].owe_transition_ssid=0000_OWE_Transition_5G uci set wireless.@wifi-iface[1].owe_transition_bssid=8C:FD:F0:0F:94:4E uci commit wireless wifi

4.3.2 Configure DUT OWE-specific settings

1. Pull the *wpa_supplicant.conf* file on the DUT to add the network settings.
   a. For the Android O build: `adb pull/data/misc/wifi/wpa_supplicant.conf`.
   b. For the Android P build: `adb pull /data/vendor/wifi/wpa/wpa_supplicant.conf`
2. Add the necessary default settings and a network setting with OWE security in the *wpa_supplicant.conf* file as follows.

```
update_config=1
eapol_version=1
ap_scan=1
fast_reauth=1
pmf=1
p2p_add_cli_chan=1
wowlan_triggers=magic_pkt
```

```
ctrl_interface=/data/vendor/wifi/wpa/sockets
// Add this one is for Android P build.
network={
ssid="0000_OWE_ 5G"
key_mgmt=OWE
psk="1234567890"
pairwise=CCMP
group=CCMP
priority=10
ieee80211w=2
owe_groups=19 20
group_mgmt=AES-128-CMAC
}
// Change to the following network setting for OWE-GCMP Security mode.
network={
ssid="0000_OWE_ 5G"
key_mgmt=OWE
pairwise=GCMP
group=GCMP
priority=10
ieee80211w=2
owe_groups=19 20
group_mgmt=BIP-GMAC-128
}
```

3. Push the *wpa_supplicant.conf* file back to the DUT device.
   a. For the Android O build: `adb push wpa_supplicant.conf /data/misc/wifi/wpa_supplicant.conf`.
   b. For the Android P build: `adb push wpa_supplicant.conf /data/vendor/wifi/wpa/wpa_supplicant.conf`
4. Add executable permission to the *socket* folder.
   a. For the Android O build: `adb shell chmod 777 /data/misc/wifi/sockets`
   b. For Android P build: `adb shell chmod 777 /data/vendor/wifi/wpa/sockets`

4.3.3 Enable DUT WLAN STA connection

1. Load the WLAN driver manually, if not loaded by default.

   ```
   adb shell insmod /vendor/lib/modules/qca_cld3_wlan. ko
   ```

2. Assign an IP address to the DUT WLAN STA interface.

   ```
   adb shell ifconfig wlan0 192.168.1.45 up
   ```

3. Launch the **wpa_supplicant** application.
   a. For the Android O build:

   ```
   adb shell /vendor/bin/hw/wpa_supplicant -i wlan0 -Dnl80211 -dddd -c /data/misc/wifi/wpa_supplicant.conf -O /data/misc/wifi/sockets &
   ```

b. For the Android P build:

```
adb shell /vendor/bin/hw/wpa_supplicant -i wlan0 -Dnl80211 -dddd -c /
data/vendor/wifi/wpa/wpa_supplicant.conf -O /data/vendor/wifi/wpa/
sockets &
```

4. Initiate a scan to trigger the connection.

   a. For the Android O build:

   ```
   adb shell wpa_cli -i wlan0 -p /data/misc/wifi/sockets scan
   ```

   b. For the Android P build:

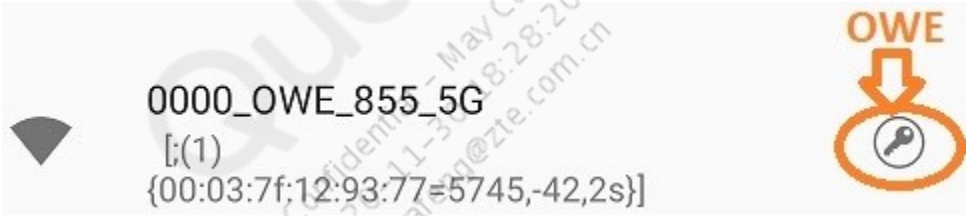
   ```
   adb shell wpa_cli -i wlan0 -p /data/vendor/wifi/wpa/sockets scan
   ```

5. For the Android P build, add the following IP rule to allow the ping traffic. This is required when establishing the connection using manual commands.

   ```
   adb shell ip rule add from all fwmark 0x0/0xffff uidrange 0-0 lookup
   main prio 23000
   ```

6. Verify if DUT WLAN STA successfully establishes the connection with the AP by checking the connection result, the key debug messages in the logs, the content of air packets, and if ping works well.

   The following image shows the icon for the OWE-supported AP in the Wi-Fi scan page, if GUI and Android framework support the OWE feature.



4.3.4 Verify log for WPA3 OWE

Check the following logs to verify the connection result:

1. Verify the WLAN driver log:

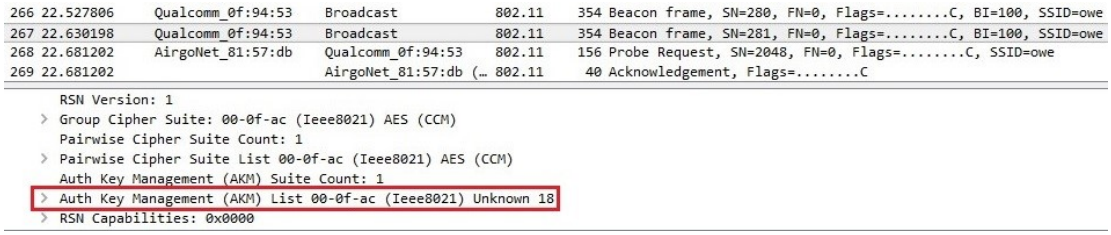
```
wlan: [4785: D:HDD] wlan_hdd_cfg80211_set_ie: 17024: Set DH EXT IE(len 37)

wlan: [4785:IL:HDD] wlan_hdd_cfg80211_connect_start: 15814: enter

wlan: [4785: D:HDD] wlan_hdd_cfg80211_connect_start: 15928: Connect to
SSID: owe operating
Channel: 0

wlan: [4785: D:HDD] hdd_translate_rsn_to_CSR_auth_type: 4812: auth_type:
22 <- Indicate OWE

wlan: [4785: E:SME] CSR_send_join_req_msg: 14827: Connecting to ssid:owe
bssid:
8c:fd:f0:0f:94:53 rssi: -40 channel: 100 country_code: US
```

```
wlan: [4786: D:PE] lim_process_mlm_auth_req: 1061: Process Auth Req
sessionID 0 Systemrole
3mlmstate 5 from: 8c:fd:f0:0f:94:53 with authtype 0

wlan: [4786: I:PE] lim_send_auth_mgmt_frame: 2196: Sending Auth seq# 1
status 0 (1) to
8c:fd:f0:0f:94:53

wlan: [4786: D:PE] lim_handle80211_frames: 945: RX MGMT - Type 0, SubType
11, seq num[527]

wlan: [4786: D:PE] lim_send_assoc_req_mgmt_frame: 2064: Sending
Association Request length 250

wlan: [4786: I:PE] lim_process_assoc_rsp_frame: 542: received Re/Assoc: 0
resp on sessionid: 0
systemrole: 3 and mlmstate: 12 RSSI: 39 from 8c:fd:f0:0f:94:53

wlan: [4786: D:HDD] hdd_association_completion_handler: 2997: sending
connect indication to
nl80211:for bssid 8c:fd:f0:0f:94:53 result:2 and Status:7
```

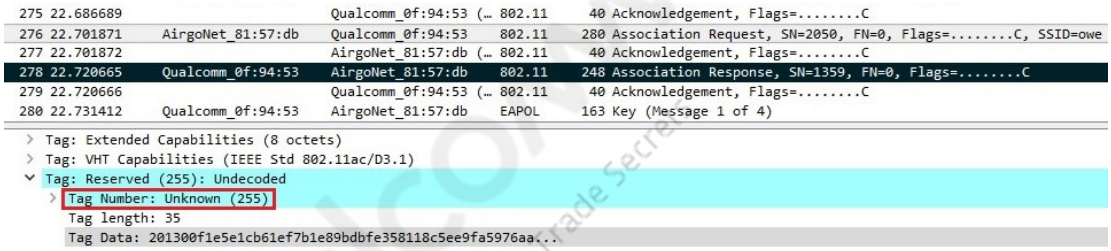
2. Verify the Logcat log:

```
12-18 23:36:24.374 D/wpa_supplicant( 4785): wlan0: RSN: using KEY_MGMT OWE
    12-18 23:36:24.375 D/wpa_supplicant( 4785): OWE: Diffie-Hellman
Parameter
    element - hexdump(len=37): ff 23 20 13 00 13 8a 9b 72 bf 6b 10 99 14
b7 79
    75 48 8e 4d 69 80 08 65 49 9c 92 7d 59 d2 f0 23 ...
    12-18 23:36:24.851 D/wpa_supplicant( 4785): OWE: DH shared secret -
hexdump(len=32): [REMOVED]
    12-18 23:36:24.852 D/wpa_supplicant( 4785): OWE: prk -
hexdump(len=32): [REMOVED]
    12-18 23:36:24.852 D/wpa_supplicant( 4785): OWE: PMK -
hexdump(len=32): [REMOVED]
    12-18 23:36:24.852 D/wpa_supplicant( 4785): OWE: PMKID -
hexdump(len=16):
    08 25 66 b9 ef 62 c0 48 62 6b 4f 89 33 83 22 19
    12-18 23:36:24.863 D/wpa_supplicant( 4785): WPA: EAPOL-Key MIC using
HMAC- SHA256 (AKM-defined - OWE)
    12-18 23:36:24.878 D/wpa_supplicant( 4785): WPA: EAPOL-Key MIC using
HMAC- SHA256 (AKM-defined - OWE)
     12-18 23:36:24.879 D/wpa_supplicant( 4785): WPA: EAPOL-Key MIC using
HMAC- SHA256 (AKM-defined - OWE)
```

3. Verify the air packets in a sniffer log:
   a. Check if the beacon packets publish OWE AKM.



   b. Check if ASSOC request and ASSOC response packets include Diffie-Hellman Parameter element whose element ID is 255.



4.4 WPA3 SAE

This section details the AP SAE-specific configuration, DUT SAE-specific settings, procedure to enable the DUT WLAN STA connection, and WPA3 SAE log verification.

The following are the SAE authentication steps:

1. The STA discovers the security policy of the AP by passively monitoring the beacon frames or through active probing. After discovery, the STA performs SAE authentication using IEEE 802.11 authentication frames with the AP.
2. After successful SAE, both the STA and AP generate a PMK.
3. The PMK generated by SAE is used in a four-ways handshake using EAPOL-key frames, as with IEEE 802.1X authentication when an AS is present.

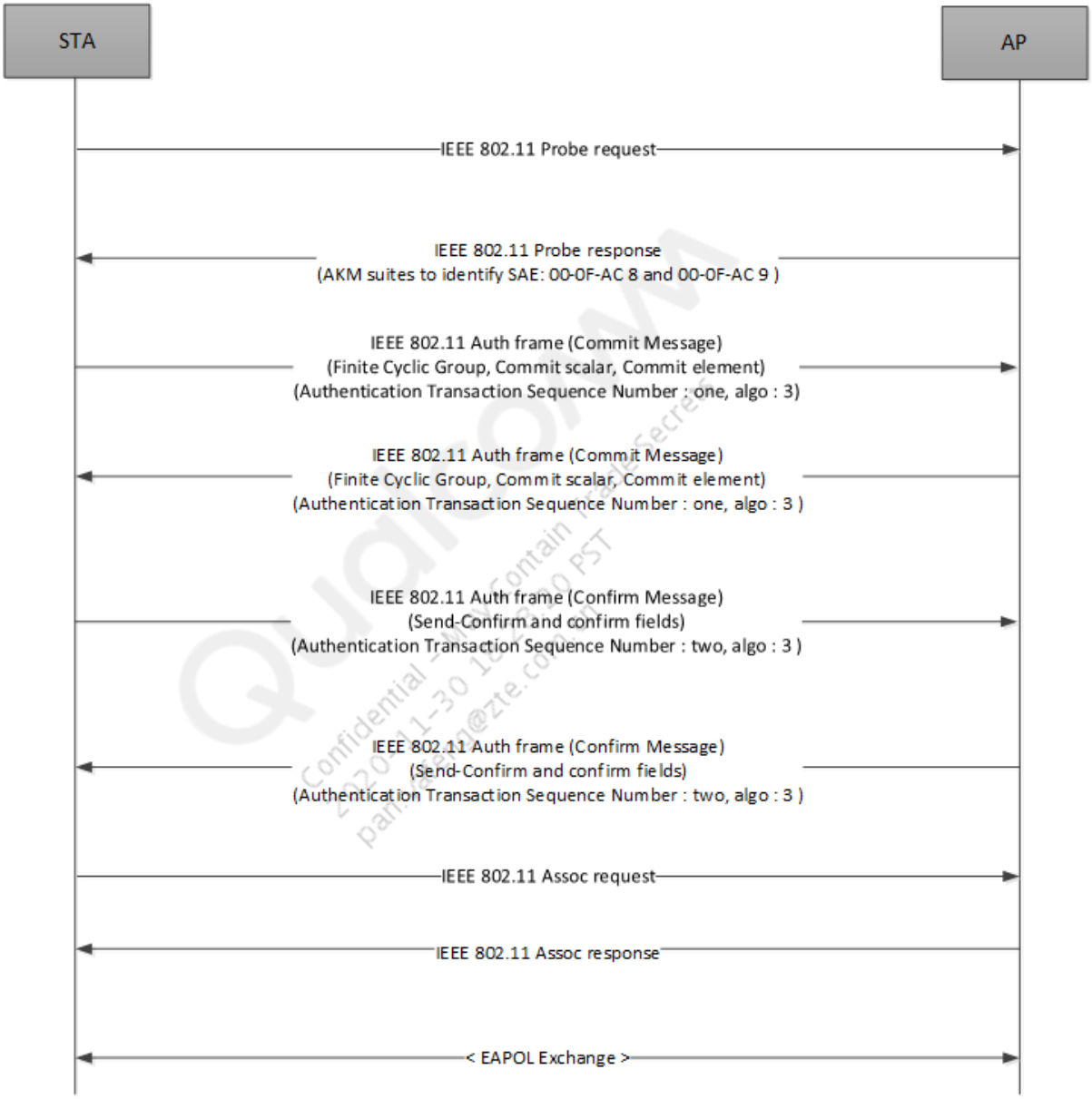
The protocol consists of two message exchanges, namely a commitment exchange and a confirmation exchange. The commitment exchange is used to force each party to the exchange to commit to a single guess of the password. The confirmation exchange is used to prove that the password guess is correct. Authentication frames are used to perform these exchanges.

Rules for message exchange

When a party has sent its message in the commit exchange, it is committed and when it has sent its message in the confirmation exchange it has confirmed. The following rules are applicable to the protocol:

- A party can commit at any time.
- A party confirms after it has committed, and its peer has committed.

- A party accepts authentication after a peer has confirmed.
- The protocol successfully terminates after each peer has accepted.



INI configuration

Table 4-1

Default configuration	Description	Default (wlan_hdd_cfg.h)	Min	Max
sae_enabled =1	sae_enabled = 1 -> enables SAEsae_enabled = 0 -> disables SAE	1	0	1

4.4.1 Configure AP SAE-specific settings

To configure AP SAE-specific settings, see the following table:

Security	Command
SAE with CCMP	uci set wireless.@wifi-iface[0].mode=ap uci set wireless.@wifi-iface[0].ssid=0000_SAE_5G uci set wireless.@wifi-iface[0].device=wifi0 uci set wireless.@wifi-iface[0].sae=1 uci set wireless.@wifi-iface[0].encryption=ccmp uci set wireless.@wifi-iface[0].wpa_group_rekey=60 uci set wireless.@wifi-iface[0].key=1234567890 uci set wireless.@wifi-iface[0].ieee80211w='2' uci set wireless.@wifi-iface[0].nss=4 uci set wireless.@wifi-iface[0].sae_group=20 uci set wireless.@wifi-iface[0].sae_anti_clogging_threshold=5 uci set wireless.@wifi-iface[0].group_mgmt_cipher=AES-128-CMAC uci commit wifi
SAE with GCMP	uci set wireless.@wifi-iface[0].mode=ap uci set wireless.@wifi-iface[0].ssid=0000_SAE_5G uci set wireless.@wifi-iface[0].device=wifi0 uci set wireless.@wifi-iface[0].sae=1 uci set wireless.@wifi-iface[0].encryption=gcmp-256 uci set wireless.@wifi-iface[0].wpa_group_rekey=60 uci set wireless.@wifi-iface[0].key=1234567890 uci set wireless.@wifi-iface[0].ieee80211w='2' uci set wireless.@wifi-iface[0].nss=4 uci set wireless.@wifi-iface[0].sae_group=20 uci set wireless.@wifi-iface[0].sae_anti_clogging_threshold=5 uci set wireless.@wifi-iface[0].group_mgmt_cipher=BIP-GMAC-256 uci commit wifi
Two AKM suites support SAE with GCMP-256 and WPA2-PSK	uci set wireless.@wifi-iface[0].mode=ap uci set wireless.@wifi-iface[0].ssid=0000_SAE_5G uci set wireless.@wifi-iface[0].device=wifi0 uci set wireless.@wifi-iface[0].sae=1 uci set wireless.@wifi-iface[0].encryption=psk2+gcmp-256 uci set wireless.@wifi-iface[0].wpa_group_rekey=60 uci set wireless.@wifi-iface[0].key=1234567890 uci set wireless.@wifi-iface[0].ieee80211w='2' uci set wireless.@wifi-iface[0].nss=4 uci set wireless.@wifi-iface[0].sae_group=20 uci set wireless.@wifi-iface[0].sae_anti_clogging_threshold=5 uci set wireless.@wifi-iface[0].group_mgmt_cipher=BIP-GMAC-256 uci commit wifi

4.4.2 Configure DUT SAE-specific settings

1. Pull the *wpa_supplicant.conf* file on the DUT to add the network settings.
   a. For the Android O build:`adb pull /data/misc/wifi/wpa_supplicant.conf`.
   b. For the Android P build: `adb pull /data/vendor/wifi/wpa/wpa_supplicant.conf`.
2. Add the necessary default settings and a network setting with OWE security in the *wpa_supplicant.conf* file as given here.

```
update_config=1
eapol_version=1
ap_scan=1
fast_reauth=1
pmf=1
p2p_add_cli_chan=1
wowlan_triggers=magic_pkt
ctrl_interface=/data/vendor/wifi/wpa/sockets
//Add this one is for Android P build.
sae_groups=19 20 21
network= {
            ssid="0000_SAE_5G"
            key_mgmt=SAE
            psk="1234567890"
            pairwise=CCMP
            group=CCMP
            priority=10
            ieee80211w=2
            group_mgmt=AES-128-CMAC
}
// Change to the following network setting for another SAE Security mode.
network={
            ssid="0000_SAE_5G"
            key_mgmt=SAE
            psk="1234567890"
            pairwise=GCMP-256
            group=GCMP-256
            priority=10
            ieee80211w=2
            group_mgmt=AES-128-CMAC
}
```

3. Push the *wpa_supplicant.conf* file back to the DUT device.
   a. For the Android O build: `adb push wpa_supplicant.conf /data/misc/wifi/wpa_supplicant.conf`.
   b. For the Android P build: `adb push wpa_supplicant.conf /data/vendor/wifi/wpa/wpa_supplicant.conf`.

4. Add executable permission to the *socket* folder.
   a. For the Android O build: `adb shell chmod 777 /data/misc/wifi/sockets`.
   b. For the Android P build: `adb shell chmod 777 /data/vendor/wifi/wpa/sockets`.

4.4.3 Enable DUT WLAN STA connection

To enable the DUT WLAN STA connection for WPA3 SAE:

1. See the Enable DUT WLAN STA connection section for making the DUT WLAN STA connection to the AP.
2. Verify if the DUT WLAN STA successfully establishes the connection with the AP by checking the connection result, the key debug messages in the logs, the content of air packets, and if ping works well.

   The following image shows the icon for SAE-supported AP in the Wi-Fi scan page, if GUI and Android framework support the SAE feature.

4.4.4 Verify log for WPA3 SAE

Check the following logs to verify the connection result:

1. Verify the WLAN driver log:

```
[00:02:12.973477] wlan: [6702: D:HDD] __wlan_hdd_cfg80211_connect: 18494:
Device_mode
QDF_STA_MODE(0)
[00:02:12.993098] wlan: [6702: D:HDD] hdd_translate_rsn_to_CSR_auth_type:
4919: auth_type: 3
 <-  SAE authentication type
[00:02:12.993137] wlan: [6702: D:HDD] hdd_set_CSR_auth_type: 5493: Set
roam Authtype to 3
[00:02:13.101823] wlan: [6702: E:SME] CSR_send_join_req_msg: 15048:
Connecting to ssid:sae
bssid: 8c:fd:f0:0f:94:53 rssi: -20 channel: 100 country_code: US
[00:02:13.172327] wlan: [6703: D:PE] lim_process_mlm_auth_req_sae: 1104:
vdev_id 0 ssid sae
8c:fd:f0:0f:94:53
[00:02:13.172447] wlan: [6703: D:PE] lim_process_mlm_auth_req: 1238:
lim_process_mlm_auth_req_sae is successful
[00:02:13.172429] wlan: [6703: D:HDD] wlan_hdd_sae_callback: 207: SAE:
sent cmd
[00:02:13.280509] wlan: [6703: I:PE] lim_process_auth_frame: 1147:
```

```
Sessionid: 0 System role: 3
limMlmState: 34: Auth response Received BSSID: 8c:fd:f0:0f:94:53 RSSI: 19
[00:02:13.280693] wlan: [6703: D:PE] lim_process_auth_frame: 1162:
auth_alg 3
[00:02:13.280787] wlan: [6703: D:PE] lim_process_sae_auth_frame: 289:
Received SAE Auth frame
type 0 subtype 11
[00:02:13.280844] wlan: [6703: D:HDD] __hdd_indicate_mgmt_frame: 913:
Frame Type = 11 Frame
Length = 128
[00:02:13.280869] wlan: [6703: D:HDD] __hdd_indicate_mgmt_frame: 995:
Indicate Frame over
NL80211 sessionid : 0, idx :23
[00:02:13.282207] wlan: [6702:IL:HDD] __wlan_hdd_mgmt_tx: 233: enter
[00:02:13.296562] wlan: [6703: I:PE] lim_process_auth_frame: 1147:
Sessionid: 0 System role: 3
limMlmState: 34: Auth response Received BSSID: 8c:fd:f0:0f:94:53 RSSI: 18
[00:02:13.298930] wlan: [6703: D:PE] lim_process_auth_frame: 1162:
auth_alg 3
[00:02:13.299055] wlan: [6703: D:PE] lim_process_sae_auth_frame: 289:
Received SAE Auth frame
type 0 subtype 11
[00:02:13.299113] wlan: [6703: D:HDD] __hdd_indicate_mgmt_frame: 913:
Frame Type = 11 Frame
Length = 64
[00:02:13.299139] wlan: [6703: D:HDD] __hdd_indicate_mgmt_frame: 995:
Indicate Frame over
NL80211 sessionid : 0, idx :23
[00:02:13.300313] wlan: [6703: D:PE] lim_process_sae_msg: 108: SAE:status
0 limMlmState 34
pePersona 0
[00:02:13.301188] wlan: [6703: D:PE] lim_send_assoc_req_mgmt_frame: 1834:
Populate HT Caps in
Assoc Request
[00:02:13.349099] wlan: [6703: D:HDD] hdd_sme_roam_callback: 4610:
****eCSR_ROAM_ASSOCIATION_COMPLETION****
[00:02:13.351152] wlan: [6703: D:HDD] hdd_conn_set_connection_state: 254:
hdd_association_completion_handler+0x6c4/0x1060 [wlan] Changed conn state
from old:1 to new:2
for dev wlan0
```

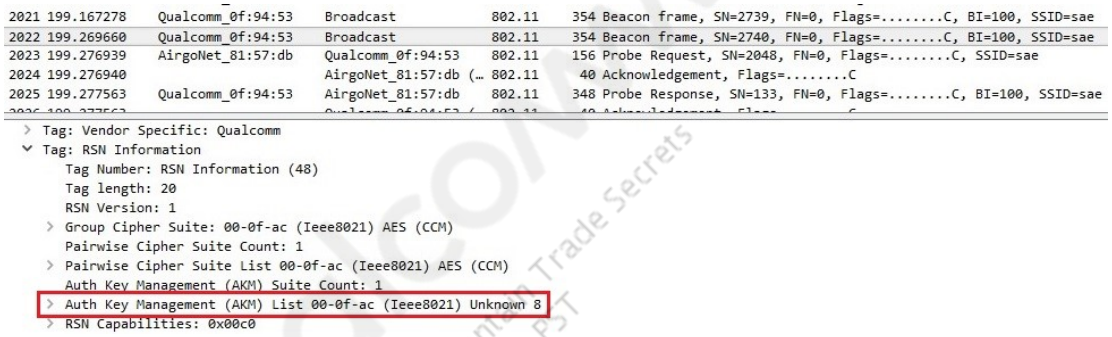
2. Verify the Logcat log:

```
01-27 00:02:12.969 D/wpa_supplicant( 6702): wlan0: selected BSS
8c:fd:f0:0f:94:53 ssid='sae'
01-27 00:02:12.969 D/wpa_supplicant( 6702): wlan0: Considering connect
request: reassociate: 0
selected: 8c:fd:f0:0f:94:53  bssid: 00:00:00:00:00:00  pending:
00:00:00:00:00:00  wpa_state:
```

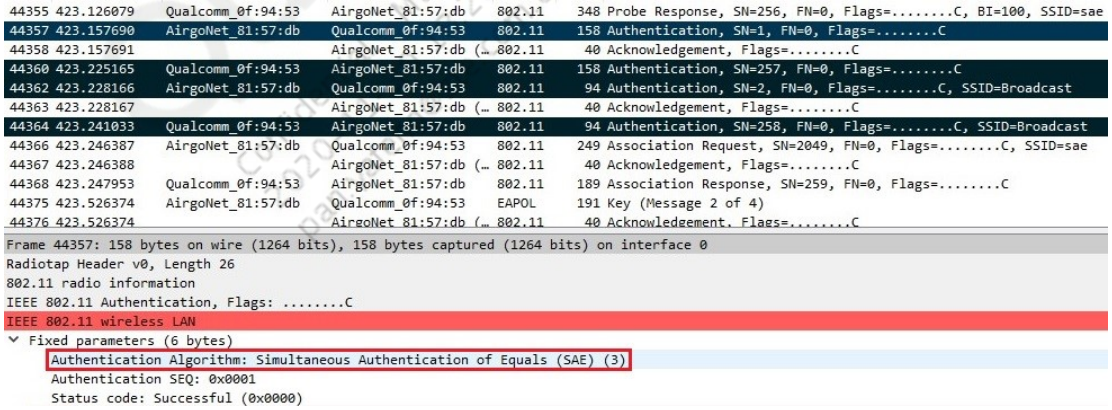
```
SCANNING  ssid=0x72d3c7d000  current_ssid=0x0
01-27 00:02:12.969 D/wpa_supplicant( 6702): wlan0: Request association
with 8c:fd:f0:0f:94:53
01-27 00:02:12.969 D/wpa_supplicant( 6702): wlan0: RSN: using KEY_MGMT SAE
01-27 00:02:13.174 D/wpa_supplicant( 6702): nl80211: Drv Event 123
(NL80211_CMD_EXTERNAL_AUTH)
received for wlan0
01-27 00:02:13.174 D/wpa_supplicant( 6702): nl80211: external auth action:
0, AKM: 0x8ac0f00
01-27 00:02:13.174 D/wpa_supplicant( 6702): wlan0: Event EXTERNAL_AUTH
(54) received
01-27 00:02:13.174 D/wpa_supplicant( 6702): wlan0: SME: Selected SAE group
19
01-27 00:02:13.209 D/wpa_supplicant( 6702): SAE: own commit-scalar -
hexdump(len=32): 2f 41 df
29 dc 28 f5 2a 9e ee cf 80 2d 38 55 02 91 be f3 70 e3 d6 c7 31 22 24 f1 6d
bf 12 7a 04
01-27 00:02:13.209 D/wpa_supplicant( 6702): SAE: own commit-element(x) -
hexdump(len=32): 40 b8
62 62 60 aa f0 fd 33 2c 94 e6 6b 97 fa 1b d9 bd 0f 9e 09 2f ec 8b 2d ad b3
0f ed cf 04 5e
01-27 00:02:13.209 D/wpa_supplicant( 6702): SAE: own commit-element(y) -
hexdump(len=32): 4f d5
49 54 08 93 0a 4f 4f 9d 98 f9 06 4a 3b 5f e2 5c b1 4b ee 4c a3 02 3e fe c8
18 c4 2c 54 91
01-27 00:02:13.281 D/wpa_supplicant( 6702): wlan0: Event RX_MGMT (18)
received
01-27 00:02:13.281 D/wpa_supplicant( 6702): wlan0: SME: SAE authentication
transaction 1 status
code 0
01-27 00:02:13.281 D/wpa_supplicant( 6702): wlan0: SME SAE commit
01-27 00:02:13.281 D/wpa_supplicant( 6702): SAE: Peer commit-scalar -
hexdump(len=32): 67 63 4e
7b fa 1a 87 c9 fc 66 f4 c5 1f 67 c8 92 8c 98 63 94 25 95 af 2b 02 37 56 d7
a1 cb d0 91
01-27 00:02:13.281 D/wpa_supplicant( 6702): SAE: Peer commit-element(x) -
hexdump(len=32): 96 9a
75 74 0b a6 67 0f a1 5d 24 4a a8 2c fb 77 f9 92 17 73 78 b9 c5 58 cc e9 13
e7 a9 d2 18 70
01-27 00:02:13.281 D/wpa_supplicant( 6702): SAE: Peer commit-element(y) -
hexdump(len=32): 7c 42
31 2d 3a 32 84 74 ab b5 ce bf a5 32 bb 14 fa 5b 64 f5 37 c0 49 53 b1 69 b0
46 d2 07 a8 8e
01-27 00:02:13.299 D/wpa_supplicant( 6702): wlan0: Event RX_MGMT (18)
received
01-27 00:02:13.299 D/wpa_supplicant( 6702): wlan0: SME: SAE authentication
transaction 2 status
code 0
```

```
01-27 00:02:13.299 D/wpa_supplicant( 6702): wlan0: SME SAE confirm
01-27 00:02:13.299 D/wpa_supplicant( 6702): SAE: peer-send-confirm 0
01-27 00:02:13.299 D/wpa_supplicant( 6702): wlan0: nl80211: external auth
status : 0
01-27 00:02:13.299 D/wpa_supplicant( 6702): SME: SAE completed - setting
PMK for 4-way handshake
01-27 00:02:13.395 D/wpa_supplicant( 6702): nl80211: Associated with
8c:fd:f0:0f:94:53
```

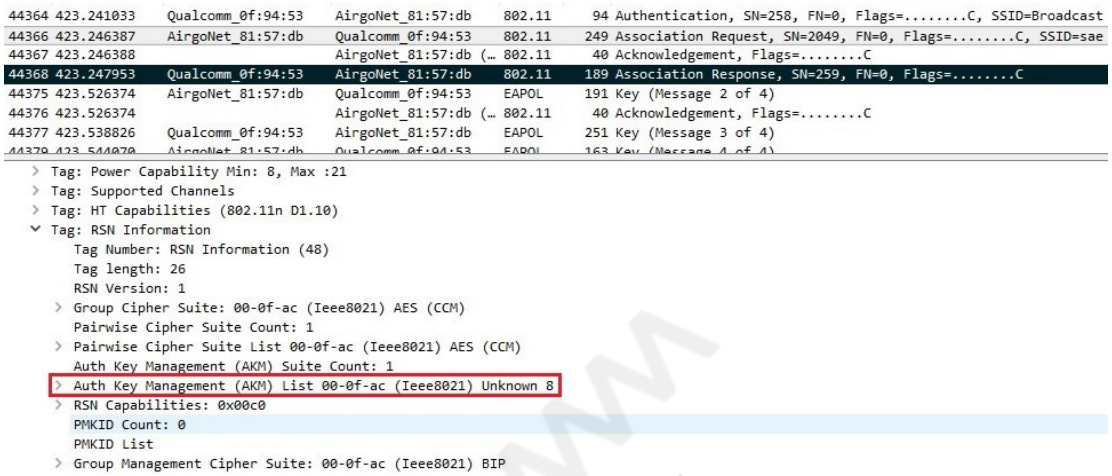
3. Verify the air packets in a sniffer log:
   a. Check if the beacon packets publish SAE AKM.



   b. Check the AUTH request and AUTH response packets.



c. Check the ASSOC request and ASSOC response packets.



4.5 WPA3-Suite B

This section provides detailed procedures to configure the RADIUS server, AP, and STA.

4.5.1 Configure RADIUS server

Prerequisites:

For the WPA3-Suite B Enterprise security mode, a Linux machine installed with Ubuntu 16.04 version and a 64-bit operating system is required for acting as RADIUS server for the network. The hostapd RADIUS server must be configured and set up for the network.

The hostapd RADIUS server package, (EC + RSA 3K) client certificates, hostapd certificates, all configuration files (EC + RSA3K) are available at the *SuiteB-04_03* package. Contact the QTI WLAN SW CE team to access it and copy this package to the Linux machine.

To configure the RADIUS server:

1. Configure the EAP users. Note that the EAP username must be specified. File to be used for the configuration is in SuiteB-04_03\hostapd\eap_user.conf.
2. Configure the RADIUS clients or AP IP address, so that the RADIUS server accepts the RADIUS requests from those APs only. Also, specify the shared secret to be used between the AP and RADIUS server. The configuration file is in SuiteB-04_03\hostapd\radius_clients.conf.

   The RADIUS client IP address is configured as 0.0.0.0, which accepts RADIUS request from all access points. The shared secret as “12345678901234567890123456789012345678901234567890123456789012345678901234”, the same shared secret code can be used or can be configured to any values as required.

The predefined RADIUS server configuration files are available in *SuiteB-04_03\hostapd*. The configurations are different for the two Suite B EAP methods, where the server-side certificates, the RADIUS clients file to be used, the EAP user's file to be used, and other settings are specified.

Serial no.	Suite B EAP method	RADIUS server configuration file to be used
1	TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384	suiteb-ec.conf
2	TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384	suite-rsa-ecdhe.conf

3. As the certificate names to be specified in the RADIUS server configuration file are the same for both the Suite B EAP methods, at a time RADIUS server can support only one Suite B EAP method.
4. To start the RADIUS server for Suite B EAP methods:
   a. Go to the location on the Linux machine where the folder is copied.
   b. Execute `cd SuiteB-04_03\hostapd`
   c. Run the RADIUS server start command as given in the following table:

Serial no.	Suite B EAP method	Commands to start RADIUS server
1	TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384	./hostapd -dd suiteb-ec.conf
2	TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384	./hostapd -dd suiteb-rsa-echdhe.conf

4.5.2 Configure AP

The Suite B AP configuration is the same for both the EAP methods. The two EAP methods are as follows:

- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384

To configure the AP, use the following commands:

```
uci set wireless.@wifi-iface[0].mode=ap
uci set wireless.@wifi-iface[0].ssid=0000_Suite-B_5G_infinity
uci set wireless.@wifi-iface[0].device=wifi0
uci set wireless.@wifi-iface[0].suite_b=192
uci set wireless.@wifi-iface[0].wpa_group_rekey=300
uci set wireless.@wifi-iface[0].ieee80211w='2'
uci set wireless.@wifi-iface[0].server=192.168.1.182
uci set wireless.@wifi-iface[0].port=1812
uci set wireless.@wifi-
iface[0].auth_secret=12345678901234567890123456789012345678901234567890123456
5678901234
uci set wireless.@wifi-iface[0].nss=4
uci set wireless.@wifi-iface[0].group_mgmt_cipher=BIP-GMAC-256
uci commit;wifi
```

4.5.3 Configure STA and connect

The STA configuration for Suite B consists of the following three steps while connecting using CLI commands:

1. Install the required certificates.
2. Configure the STA for the Suite B profile.
3. Establish the connection.

Install certificates for Suite B EAP methods

For the Suite B EAP method, the following certificate files are required:

- ca_cert (CA certificate)
- User certificate
- User key

These certificates must be generated and provisioned on the client device. These certificates are already available at *SuiteB-04_03\client_certs_for_WPA3*. Use the following commands for provisioning required certificates on the device.

1. For the EAP method TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384, use the following commands to push the EC-related certificates to the device:

```
adb shell mkdir /data/cert
adb push ec2-ca.pem /data/cert/ec2-ca.pem
adb push ec2-user.pem /data/cert/ec2-user.pem
adb push ec2-user.key /data/cert/ec2-user.key
```

2. For the EAP method TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384, use the following commands to push the RSA-related certificates to the device:

```
adb shell mkdir /data/cert
adb push rsa3072-ca.pem /data/cert/rsa3072-ca.pem
adb push rsa3072-user.pem /data/cert/rsa3072-user.pem
adb push rsa3072-user.key /data/cert/rsa3072-user.key
```

Configure STA

1. Load the WLAN driver manually, if it is not loaded by default.

```
adb shell insmod /vendor/lib/modules/qca_cld3_wlan. ko
```

2. Pull the *wpa_supplicant.conf* file to add the network settings specific to the security modes

```
adb pull /data/vendor/wifi/wpa/wpa_supplicant.conf
```

3. Update *wpa_supplicant.conf* as follows:

Security	wpa_supplicant.conf

ECDSA	update_config=1
	eapol_version=1
	ap_scan=1
	fast_reauth=1
	pmf=1
	p2p_add_cli_chan=1
	ctrl_interface=/data/vendor/wifi/wpa/sockets
	
	network={
	ssid="0000_Suite-B_5G_infinity"
	proto=RSN
	key_mgmt=WPA-EAP-SUITE-B-192
	pairwise=GCMP-256
	group=GCMP-256
	eap=TLS
	priority=10
	ieee80211w=2
	group_mgmt=BIP-GMAC-256
	openssl_ciphers="SUITEB192"
	identity="Client"
	ca_cert="/data/cert//ec2-ca.pem"
	client_cert="/data/cert//ec2-user.pem"
	private_key="/data/cert//ec2-user.key"
	private_key_passwd="wifi"
	}

RSA	update_config=1
	eapol_version=1
	ap_scan=1
	fast_reauth=1
	pmf=1
	p2p_add_cli_chan=1
	ctrl_interface=/data/vendor/wifi/wpa/sockets
	
	network={
	ssid="0000_Suite-B_5G_infinity"
	proto=RSN
	key_mgmt=WPA-EAP-SUITE-B-192
	pairwise=GCMP-256
	group=GCMP-256
	eap=TLS
	priority=10
	ieee80211w=2
	group_mgmt=BIP-GMAC-256
	phase1="tls_suiteb=1"
	identity="Client"
	ca_cert="/data/cert//rsa3072-ca.pem"
	client_cert="/data/cert//rsa3072-user.pem"
	private_key="/data/cert//rsa3072-user.key"
	private_key_passwd="wifi"
	}

Establish connection in any security mode

1. Push the *wpa_supplicant.conf* file to the device.

   ```
   adb push wpa_supplicant.conf /data/vendor/wifi/wpa/wpa_supplicant.conf
   ```

2. Add executable permission to the pushed *wpa_supplicant.conf* file.

   ```
   adb shell chmod 777 /data/vendor/wifi/wpa/sockets
   ```

3. Assign an IP to the DUT.

   ```
   adb shell ifconfig wlan0 192.168.1.45 up
   ```

4. Run wpa_supplicant.

   ```
   adb shell /vendor/bin/hw/wpa_supplicant -i wlan0 -Dnl80211 -c
   /data/vendor/wifi/wpa/wpa_supplicant.conf -O
   /data/vendor/wifi/wpa/sockets&
   ```

5. Using wpa_cli, initiate a scan to connect.

   ```
   adb shell wpa_cli -i wlan0 -p /data/vendor/wifi/wpa/sockets SCAN
   ```

6. Add the following IP rule to allow the ping traffic on Android-P builds. This is required when establishing a connection using CLI commands.

   ```
   adb shell ip rule add from all fwmark 0x0/0xffff uidrange 0-0 lookup
   main prio 23000
   ```

4.6 Support for GMAC-256

When management frame protection is negotiated, GMAC provides integrity protection of MC/BC robust management frames. The driver calculates MIC when it receives protected MC/BC frames and use the calculated MIC to validate the received frame.

Limitations

The following are the limitations of GMAC-256:

- Power is impacted as MIC calculation is done in the driver.
- GMAC offload cannot be supported in certain cases.
- Supported only for STA interfaces.
- Roaming cannot be supported for GMAC group management cipher suite.

4.7 Device provisioning protocol

The Device Provisioning Protocol (DPP) provides a simple and secure method to add Wi-Fi devices, which have limited or no display.

This program defines new configuration methods and specifies requirements for several services: rekeying, configuration using an intermediate device, and multiple enrolments at the same time.

In DPP, public keys are used to identify and authenticate all devices. The private key associated with a public key must be generated within each device and protected from disclosure. Devices use public key cryptographic techniques to authenticate peer devices and establish shared keys for further secured communications. This security architecture simplifies the establishment of secure connectivity

between devices and provides a foundation for improved usability in provisioning and connecting devices.

4.7.1 Device role

The following are the device roles:

- Configurator
- Enrollees

A configurator supports the setup of enrollees. As part of the setup, the configurator and the enrollee engage in the DPP authentication protocol.

After the authentication completes, the configurator provisions the enrollee for device-to-device communication or infrastructure communication. As part of this provisioning, the configurator enables the enrollee to establish secure associations with other peers in the network. During provisioning, the configurator delegates the capabilities of provisioning other devices to the enrollee.

Devices that the configurator configures are called peers. Special types of peers, such as AP or GO, are referred to as aggregators.

4.7.2 Authentication roles

The following are the authentication roles:

- Initiator
- Responder

In DPP, the device that starts the DPP authentication protocol plays the role of initiator. The device, which responds to an initiator request plays the role of responder. The DPP authentication protocol provides authentication of a responder to an initiator, and optionally authentication of the initiator to the responder. This assumes that the initiator has obtained the bootstrapping key of the responder to perform unidirectional authentication, and both parties have obtained the bootstrapping keys of each other to optionally perform mutual authentication.

In DPP, parties involved in the authentication protocol are the configurator and the enrollee. Depending on the bootstrapping scenario either of configurator or enrollee takes the roles of initiator or responder, respectively.

Message flows for infrastructure connectivity

The message flows for infrastructure connectivity are given in the following sections show a message flow where a configurator provisions an enrollee with a connector. The STA enrollee then uses the connector to establish a secure connection to an AP via the network introduction protocol as shown in the following section.

DPP message flow for DPP provisioning of an enrollee STA

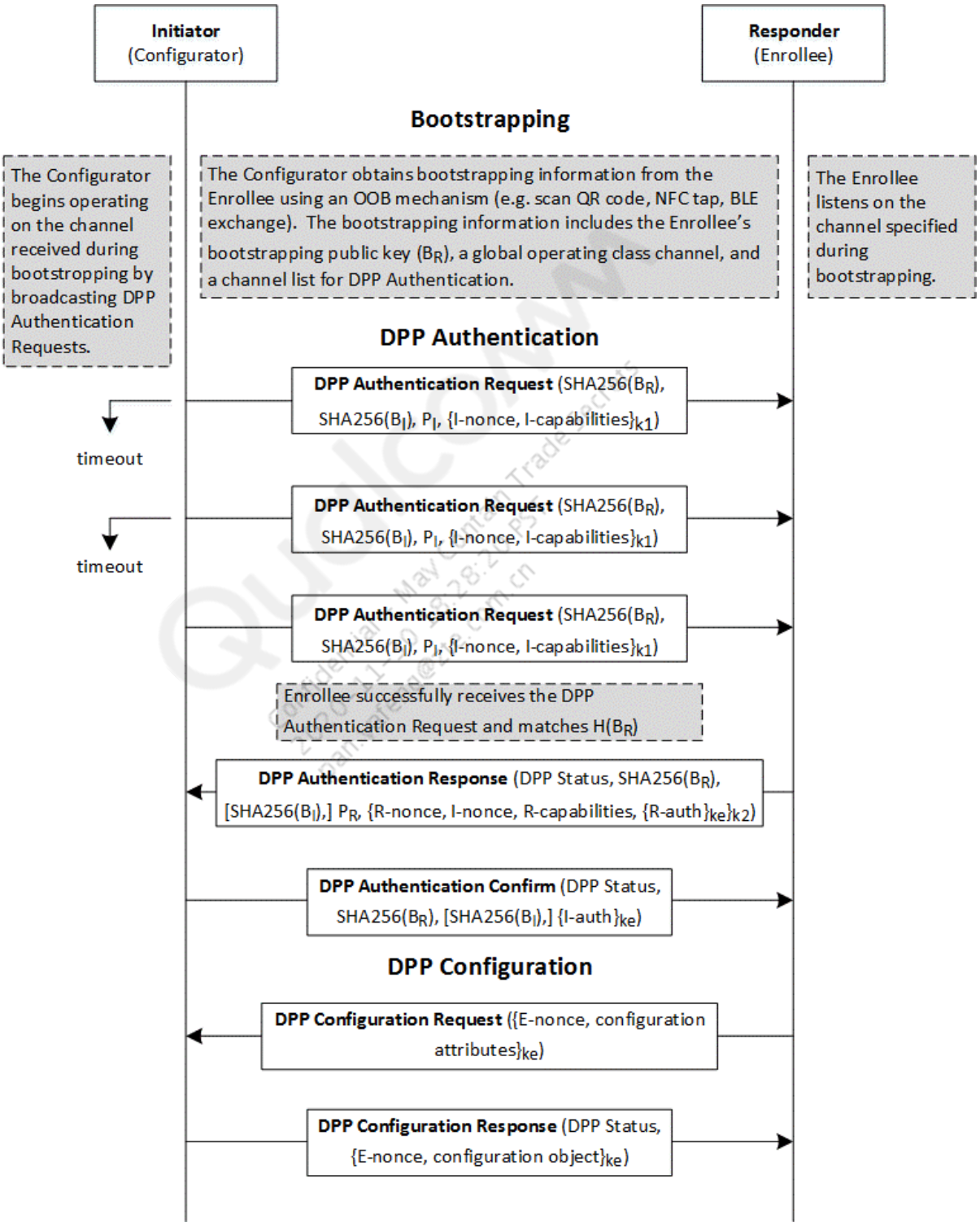
The DPP authentication requires either the enrollee or the configurator to take on the role of an initiator regarding the DPP authentication protocol. In the example, the configurator takes on the role of initiator and obtains bootstrapping information for the enrollee. The bootstrapping information includes the public bootstrap key and a global operating class and a channel number list that the STA enrollee listens on to perform the DPP authentication protocol.

The initiator issues DPP authentication request frames on channels in the channel list and waits for a response from the responder. After successfully receiving a response, the initiator validates the result

and transmits a DPP authentication confirm frame to complete the DPP authentication. After successful completion of these frame exchanges, a secure channel between the initiator/configurator and responder/enrollee is established.

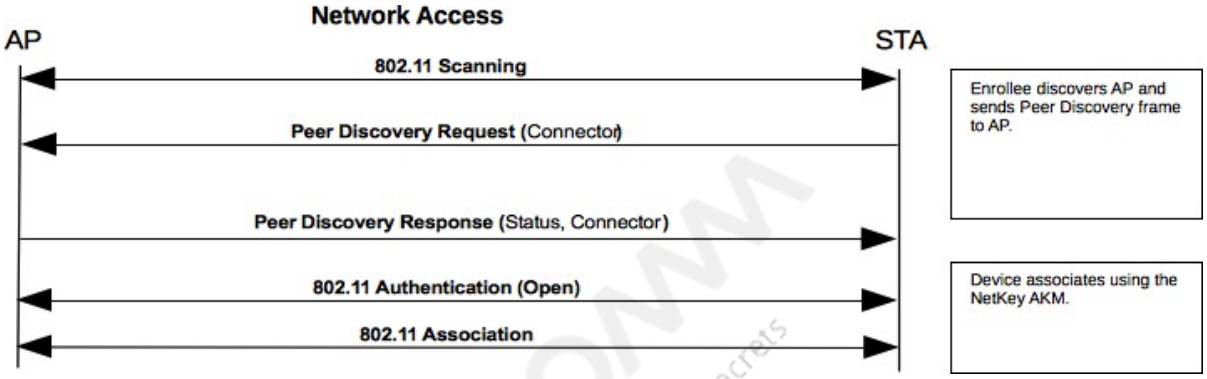
The enrollee initiates the provisioning phase by transmitting a DPP configuration request frame and is provisioned with configuration information in a DPP configuration response frame. After successfully

receiving the DPP configuration response frame, the enrollee is provisioned with the information required to establish secure access to the AP.



Message flow for network access using connector

After the client is provisioned with configuration and a connector to use as credentials, it discovers the access point; transmits a peer discovery request frame; and then waits for a peer discovery response frame. Upon successful validation of the peer discovery frames, the STA and AP mutually derive a PMK and PMKID, and the STA follows the normal IEEE 802.11 procedures.



4.7.3 Establishing P2P group using DPP

Establishing a P2P group using DPP requires the following steps:

1. Bootstrapping of trust

   The public keys of the devices are transferred from one device to another. This places a trust that the public key belongs to the peer device. This is performed using either the out-of-band (OOB) mechanism for devices with limited capability or using in-band bootstrapping (PKEX frames) for devices that have a user interface.

2. P2P discovery

   P2P discovery is performed either in-band or out-of-band. P2P discovery information is acquired via probe request and probe response. Alternatively, P2P discovery information is included in bootstrapping of OOB data.

3. P2P group owner negotiation

   The role of a P2P device, group ownership, and characteristics are determined.

4. DPP discovery

   DPP discovery is triggered by the P2P group owner (as initiator) to ensure that the P2P client to be provisioned via DPP is still active or ready for provisioning.

5. DPP authentication

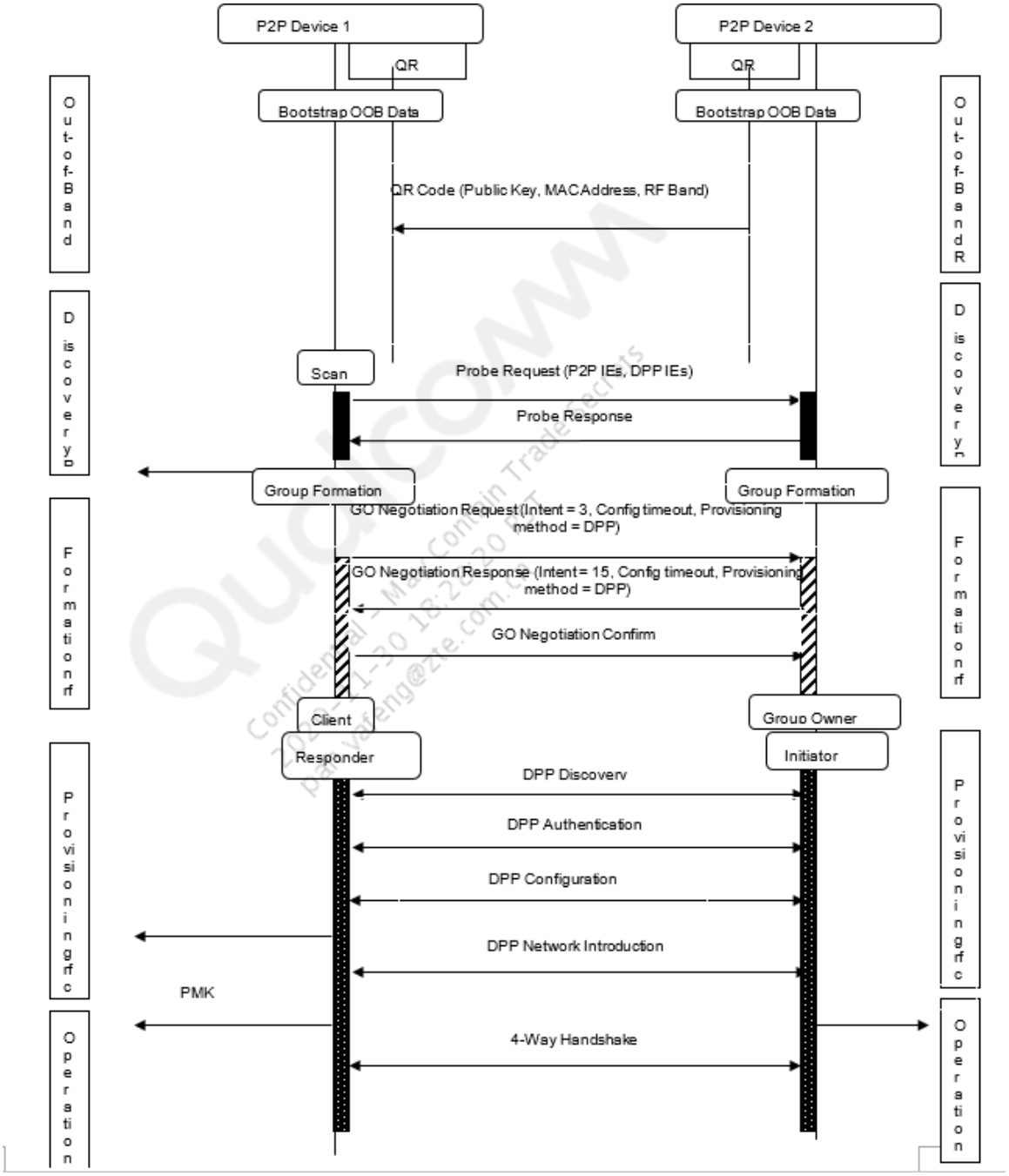
   DPP authentication verifies the public key that is acquired during bootstrapping phase via encrypted action frames.

6. DPP configuration

   DPP configuration verifies the attributes of the device and provides the connector to be used so as to gain access to the network.

7. DPP network introduction

   The DPP network introduction allows communication to devices using the same connector broadcast by the provisioned client.



4.7.4 WPA3 transition

WPA3 transition is Introduced by Google. It upgrades PSK profiles to SAE profiles while connection attempt based on latest scan results.

- Conditions for upgrading PSK to SAE connection
  - Manually triggered connection attempt
    - Latest scan results shall not have Legacy WPA2 PSK network with same SSID.
    - Latest scan results shall have either SAE (OR) SAE+PSK transition mode network with same SSID
  - Network selection triggered connection attempt
    - Selected scan result candidate should support SAE/SAE+PSK AKM
    - Latest scan results shall not have PSK only network with same SSID
- Persistent network config in “WifiConfigStore.xml” won’t be modified on auto SAE upgrade. i.e “WifiConfigStore.xml” will have PSK profile only.
- SAE auto upgrade feature can be controlled by below RRO’s
  - R.bool.config_wifiSaeUpgradeEnabled – RRO to enable the feature
    - Default value is “true”
  - R.bool.config_wifiSaeUpgradeOffloadEnabled – RRO to offload SAE upgrade decision to driver/firmware
    - Default value is “false”

5 MU-MIMO and OFDMA DL/UL tests

This chapter details the test setup for MU-MIMO and OFDMA DL/UL tests.

5.1 Test setup for MU-MIMO and OFDMA DL/UL tests

QCA639X is still at early development phase. The OFDMA feature is not ready yet until CS. The current setup is with QCA669X for the OFDMA test. This document will be updated in a future revision for QCA639X client when the OFDMA feature is ready.

5.1.1 Prerequisites for MU-MIMO and OFDMA DL/UL tests

The test equipment list for MU-MIMO and OFDMA DL/UL tests are given in the following table:

Device	Quantity	Use/Note	SW version
IPQ8087 + QCA6694 AP V2.0	1	Access Point	CI_IPQ8074.ILQ.10.0-00185-P-1
IPQ8087 + QCA6694 AP V2.0	1	Sniffer	CI_IPQ8074.ILQ.10.0-00185-P-1
SDM8150 + QCA6390	8	Clients	CI_SM8150.LA.1.0.0.1-02191-STD.INT-45
Intel i7 Windows10 laptop + Chariot server	1	Traffic Generator	–

5.1.2 AP configuration

This section details the following AP configurations:

1. 802.11 settings
2. 8 × 8 sounding
3. SU, MU-MIMO, and multiclient settings

Detailed test procedure is provided at the end of this section.

Configure IPQ807X AP 802.11 settings (HE80/20)

Run the following commands:

```
uci set wireless.wifi0.channel='44'
uci set wireless.wifi0.htmode='HT80'
uci set wireless.wifi0.disabled='0'
uci set wireless.@wifi-iface[0].ssid='HKNPRPERF5G'
```

```
uci set wireless.wifi1.channel='6'
uci set wireless.wifi1.htmode='HT20'
uci set wireless.wifi1.disabled='0'
uci set wireless.@wifi-iface[1].ssid='HKNPRPERF2G'
uci set wireless.@wifi-iface[0].encryption='psk2'
uci set wireless.@wifi-iface[0].key='12345678'
uci set wireless.@wifi-iface[1].encryption='psk2'
uci set wireless.@wifi-iface[1].key='12345678'
uci commit
wifi
```

8 × 8 sounding—SU and DL MU-MIMO test

Consider the following points for 8 × 8 sounding test:

- 8ss Beamformee (BF) support is only for the DUT station mode.
- 8ss Beamformer capability expected from the AP for this feature to work.
- *gTxBFCsnValue* +1 is assumed to be the default firmware-supported BF antennas, if the firmware supports eight antennas in Rx-ready event. Therefore, *gTxBFCsnValue* =7 supports 8ss Beamformee feature on STA.
- CSN of the beamformer antenna in VHT caps is one less than the maximum number of spatial streams that the STA can receive in VHT NDP (that is, CSN in VHT caps = maximum space time streams STA can receive - 1). For example, if STA can receive a maximum of four space time streams, it sets the value in VHT caps as 3.

Configure AP for SU, MU-MIMO, and multiclient SU

This section provides details on the AP configuration for SU, MU-MIMO, and multiclient SU tests.

Configure AP for SU test:

Create the following file with the contents listed here. Each time if any ath* interface is brought up, MU-MIMO, DL OFDMA, and UL OFDMA are disabled automatically:

Create file

1. In the AP terminal, execute the following commands:

   ```
   cd /etc/hotplug.d/net
   cat > 99-disable-mu
   vi 99-disable-mu
   ```

2. Update the following lines in `99-disable_mu`. This disables the MU-MIMO and OFDMA features.

   NOTE Use this only for SU KPI testing.

   ```
   root@OpenWrt:~# cat /etc/hotplug.d/net/99-disable_mu
   #!/bin/sh
   if [ "${ACTION}" = "add" ]; then
               case "$INTERFACE" in
                   ath*)
                           iwpriv "${INTERFACE}" he_dlofdma 0
   ```

```
                        iwpriv "${INTERFACE}" he_ulofdma 0
                        iwpriv "${INTERFACE}" he_mubfer 0
                        ;;
            esac
fi
```

3. Before running the SU throughput, double-check if the script is taking effect with the following commands:

```
root@OpenWrt:/# iwpriv ath0 get_he_ulofdma
ath0      get_he_ulofdma:0
root@OpenWrt:/# iwpriv ath0 get_he_dlofdma
ath0      get_he_dlofdma:0
root@OpenWrt:/# iwpriv ath0 g_he_mubfer
ath0      g_he_mubfer:0
```

Configure AP for MU-MIMO test:

1. Use the following commands on the AP to enable MIMO.

```
iwpriv ath0 vhtmubfer 1; iwpriv ath0 he_mubfer 1 //enable MIMO
*iwpriv ath0 vhtmubfer 0; iwpriv ath0 he_mubfer 0 //diable MIMO
```

2. For 802.11ac cases, make sure to disable MCN 10/11 from the AP using the following command:

   AP (limit MCS to 9)

```
wifitool ath0 setUnitTestCmd 0xa 2 2 1
```

3. Disable DL OFDMA by running the following command:

```
iwpriv ath0 he_dlofdma 0
```

4. Disable UL OFDMA by using the following command:

```
iwpriv ath0 he_ulofdma 0
```

Confgure AP for multiclient SU test:

1. Disable MU-MIMO on the AP.

```
iwpriv ath0 vhtmubfer 0;
iwpriv ath0 he_mubfer 0
```

2. For 802.11ac cases, make sure to disable MCS 10/11 from the AP and STA sides using the following commands:

   AP (limit MCS to 9)

```
wifitool ath0 setUnitTestCmd 0xa 2 2 1
```

   STA (limit MCS to 9)

```
iwpriv wlan0 setUnitTestCmd 0xa 2 2 1
```

Stats to check for OFDMA/MU-MMIO status on IPQ807X AP:

```
wifistats wifiX 9      ---------DL OFDMA/MUMIMO DL stats/SU DL
wifistats wifiX 10    ------------UL OFDMA /SU UL
wifistats wifiX 17    ------------OFDMA/MUMIMO packet counts
wifistats wifiX 12     -------------sounding/trigger frames counts/errors
**X- 0 is 5G and 1 is 2G**
```

Test procedure for MU-MIMO and OFDMA DL/UL

The test procedure for MU-MIMO and OFDMA DL/UL include the following steps:

1. Scan
2. Connect
3. Check SU TxBF capability
4. Check MU TxBF capability

Scan

To find APs with 8ss support and decode IE contents correctly:

1. Turn On the AP with 8ss support.
2. Turn On WLAN on the DUT and scan for APs.

All APs including 8ss APs should be found by the DUT. The AP capability with 8ss support can be correctly decoded by DUT.

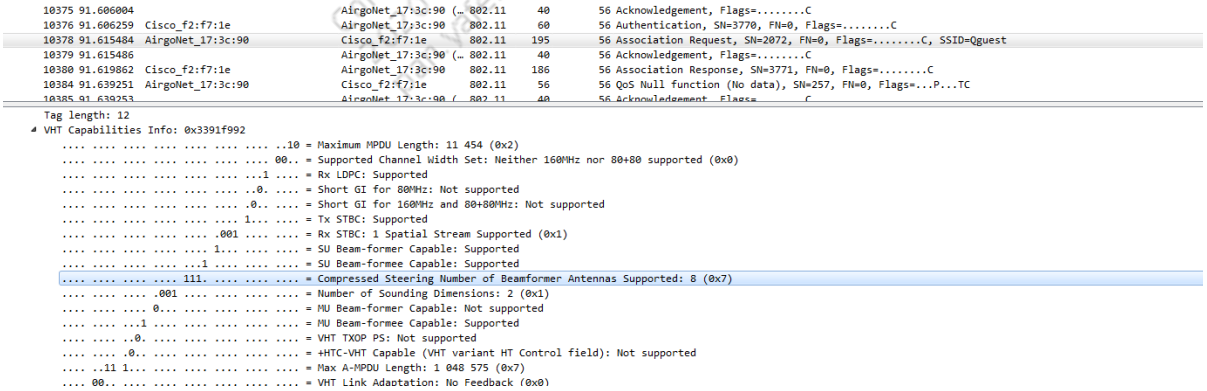
Connect

To connect to 8ss AP:

1. Turn On AP with 8ss support.
2. On the DUT, find the 8ss AP and set up connection to the 8ss APs.

Connection must be set up with the correct capability announced by the DUT to the AP.

- Data exchange (ping and so on) between the DUT and AP can go through.
- Sniffer: Check if probe/assoc frames have CSNs set to 7 or the Action frame contains VHT MIMO Control IE with Nc = no of DUT Rx ant and Nr = no.



Check SU TxBF capability

To exchange sounding packets and receive as TxBF'ee using the correct Nss from the AP:

1. Turn On the AP with 8ss support.
2. On the DUT, find the 8ss AP and set up connection to the 8ss APs.
3. Start high-throughput DL traffic from the AP to the DUT/client:

   8 x 2 CBF is exchanged during sounding.

Check MU TxBF capability

To form MU group with the 8ss AP, using 8ss sounding packets with all 8ss-capable clients:

1. Turn On AP with 8ss support.
2. On the DUT, find the 8ss AP and set up connection to the 8ss APs.
3. Repeat step 2 with other 8ss-capable clients.
4. Start high-throughput DL traffic from the AP to all the clients simultaneously.
   MU group forming with 8 x 2 CBF is exchanged during sounding.

5.1.3 DL/UL OFDMA test instructions

This section explains the AP configuration to test the following features:

- UL OFDMA
- DL OFDMA

The details to run the OFDMA test scripts are also provided.

Configure AP to test DL/UL OFDMA

This section details the required AP configuration to test DL/UL OFDMA.

UL OFDMA

Push the scripts to the AP root folder and provide write permissions to all scripts.

1. For 2 GHz UL OFDMA, push the following scripts:

```
./ disable_txbf.sh 1
./ enable_ulofdma.sh  1
./ set_muedca_paramD.sh 1
```

2. For 5 GHz UL OFDMA, push the following scripts:

```
./ disable_txbf.sh 0
./ enable_ulofdma.sh  0
./ set_muedca_paramD.sh 0
```

Test scripts

disable_txbf.sh

```
iwpriv 'ath'$1 novap_reset 1;
iwpriv 'ath'$1 he_mubfer 0;iwpriv 'ath'$1 he_subfer 0;iwpriv 'ath'$1
vhtsubfer 0;iwpriv 'ath'$1 vhtmubfer 0
```

enable_ulofdma.sh

```
    iwpriv 'ath'$1 novap_reset 1
    cfg80211tool 'ath'$1 he_ul_ofdma 1
#   cfg80211tool 'ath'$1 he_mu_edca 1
#   cfg80211tool 'ath'$1 he_ul_mcs  5
#   cfg80211tool 'ath'$1 he_ul_nss  2
    wifitool 'ath'$1 setUnitTestCmd 0x47 2 92 1
#   cfg80211tool 'ath'$1 he_ar_gi_ltf 0x0804
```

set_muedca_paramD.sh

```
echo "BE: 8 9 10"
cfg80211tool ath$1 set_muedcaparams 3 0 8
cfg80211tool ath$1 set_muedcaparams 1 0 9
cfg80211tool ath$1 set_muedcaparams 2 0 10

echo "BK: 15 9 10"
cfg80211tool ath$1 set_muedcaparams 3 1 15
cfg80211tool ath$1 set_muedcaparams 1 1 9
cfg80211tool ath$1 set_muedcaparams 2 1 10

echo "VI: 5 5 7"
cfg80211tool ath$1 set_muedcaparams 3 2 5
cfg80211tool ath$1 set_muedcaparams 1 2 5
cfg80211tool ath$1 set_muedcaparams 2 2 7

echo "VO: 5 5 7"
cfg80211tool ath$1 set_muedcaparams 3 3 5
cfg80211tool ath$1 set_muedcaparams 1 3 5
cfg80211tool ath$1 set_muedcaparams 2 3 7
```

On STA (DUT), check UL OFDMA stats with this command

```
iwpriv wlan0 txrx_stats 3
```

DL OFDMA

Push the scripts to the AP root folder and provide write permissions to all scripts.

1. For 2 GHz DL OFDMA, , push the following scripts:

   ```
   ./ enable_dlofdma_MU_BAR.sh 1
   ./ enable_ulofdma.sh 1
   ./ set_muedca_paramD.sh 1
   ```

2. For 5 GHz DL OFDMA, push the following scripts:

```
./ enable_dlofdma_MU_BAR.sh 0
./ enable_ulofdma.sh 0
./ set_muedca_paramD.sh 0
```

Test scripts

enable_dlofdma_MU_BAR.sh

```
echo 'ath'$1
iwpriv 'ath'$1 novap_reset 1
iwpriv 'ath'$1 vhtmubfer 0
iwpriv 'ath'$1 vhtsubfer 0
iwpriv 'ath'$1 he_mubfer 0
iwpriv 'ath'$1 he_subfer 0
iwpriv 'ath'$1 he_dlofdma 1;
wifitool 'ath'$1 setUnitTestCmd 0x47 2 8 0;
wifitool 'ath'$1 setUnitTestCmd 0x47 2 29 0;
wifitool 'ath'$1 setUnitTestCmd 0x47 2 11 1000000;
```

```
wifitool 'ath'$1 setUnitTestCmd 0x47 2 17 1000000;
wifitool 'ath'$1 setUnitTestCmd 0x4b 2 2 1
wifitool 'ath'$1 setUnitTestCmd 0x47 2 64 0
#cfg80211tool 'wifi'$1 ppdu_duration 2000
#wifitool 'ath'$1 setUnitTestCmd 0x48 2 167 2000
#wifitool 'ath'$1 setUnitTestCmd 0x4b 2 12 0
#wifitool 'ath'$1 setUnitTestCmd 0x4b 2 13 0
#wifitool 'ath'$1 setUnitTestCmd 0x48 3 132 3 20000
#wifitool 'ath'$1 setUnitTestCmd 0x48 2 120 1
```

Check the sniffer

To check the sniffer, use the following command:

```
wifitool ath0 setUnitTestCmd 0x49 3 12 [Band] [Association ID]
```

Where, `Band` for 5G = `0` and 2G = `1`. The Association ID (AID) can be collected from non-11ax AP. For example, the following command checks the sniffer of 5 GHz and AID = 1.

```
wifitool ath0 setUnitTestCmd 0x49 3 12 0 1
```

enable_ulofdma.sh

```
    iwpriv 'ath'$1 novap_reset 1
    cfg80211tool 'ath'$1 he_ul_ofdma 1
#  cfg80211tool 'ath'$1 he_mu_edca 1
#  cfg80211tool 'wifi'$1 he_ul_ppdu_dur 1000
#  cfg80211tool 'wifi'$1 he_ul_ppdu_dur 2000
#  cfg80211tool 'ath'$1 he_ul_mcs  5
#  cfg80211tool 'ath'$1 he_ul_nss  2
   wifitool 'ath'$1 setUnitTestCmd 0x47 2 92 1
#  cfg80211tool 'ath'$1 he_ar_gi_ltf 0x0804
```

set_muedca_paramD.sh

```
echo "BE: 8 9 10"
cfg80211tool ath$1 set_muedcaparams 3 0 8
cfg80211tool ath$1 set_muedcaparams 1 0 9
cfg80211tool ath$1 set_muedcaparams 2 0 10

echo "BK: 15 9 10"
cfg80211tool ath$1 set_muedcaparams 3 1 15
cfg80211tool ath$1 set_muedcaparams 1 1 9
cfg80211tool ath$1 set_muedcaparams 2 1 10

echo "VI: 5 5 7"
cfg80211tool ath$1 set_muedcaparams 3 2 5
cfg80211tool ath$1 set_muedcaparams 1 2 5
cfg80211tool ath$1 set_muedcaparams 2 2 7

echo "VO: 5 5 7"
cfg80211tool ath$1 set_muedcaparams 3 3 5
cfg80211tool ath$1 set_muedcaparams 1 3 5
cfg80211tool ath$1 set_muedcaparams 2 3 7
```

OFDMA test script

1. Turn On the AP and configure with OFDMA UL or DL support.
2. On the DUT, run `scan` and find the OFDMA AP.
3. Connect the OFDMA AP.
4. Start high-throughput DL/UL traffic between the APs.

A Glossary of terms

Refer to this table for the list of acronyms, abbreviations, and terms with their definitions:

Table A-1 Glossary of terms

Term	Definition
ACK	Acknowledgement - a response packet the receiver sends to acknowledge receipt of a packet.
AP	Access point
AUTH	Authorization
BF	Beamformee
CCA	Clear channel assessment
CCMP	Counter-Mode/CBC-MAC Protocol (IEEE 802.11I encryption algorithm)
CLI	Command line interface
COM	Communication (ports)
CSN	Connect supply net
CTS	Clear to send
DCM	Dual subcarrier modulation
DL	Downlink
DUT	Device under test
EAP	Extensible Authentication Protocol
ER	Extended range
FW	Firmware
HE	High efficiency
IEEE	Institute of Electrical and Electronics Engineers
KPI	Key performance indicators
L-SIG	Legacy signal
MAC	Media access control
MIMO	Multiple input, multiple output
MU	Multiuser
NDP	Null data packet
OFDM	Orthogonal frequency division multiplexing
OFDMA	Orthogonal frequency division multiple access
OWE	Opportunistic Wireless Encryption

Table A-1 Glossary of terms (cont.)

Term	Definition
PHY	Physical layer
PPDU	Physical-layer protocol data unit
QAM	Quadrature amplitude modulation
RADIUS	Remote authentication dial-in user service
RF	Radio frequency
RL-SIG	Repeat L-SIG
RTS	Request to send
RU	Resource unit
Rx	Receive
SAE	Simultaneous Authentication of Equals
STA	Station
SU	Single user
TF	Trigger frame
TWT	Target wake time
Tx	Transmit
TxBF	Transmit beamforming
UL	Uplink
VHT	Very high throughput
VoWiFi	Voice over Wi-Fi
WLAN	Wireless local area network
WPA2-PSK	Wi-Fi Protected Access (v2) - phase shift key
WPA3	Wi-Fi Protected Access (v3)