Configure Mobility Topologies on Catalyst 9800 WLCs

Contents

- Introduction
- Prerequisites
  - Requirements
  - Components Used
- Configure
  - Network Diagram
  - Guidelines and Restrictions
  - Mobility Tunnel Between Two Catalyst 9800 WLCs
    - Step 1. Collect Mobility Configuration of both 9800 WLCs
    - Step 2. Add Peer Configuration
- Mobility Tunnel Between AireOS WLC and 9800-CL controllers
  - Network Diagram
  - AireOS WLC Configuration
    - Step 1. Collect 9800 WLC Mobility Information
    - Step 2. Collect the HashValue from the 9800 WLC
    - Step 3. Add the 9800 WLC Information into the AireOS WLC
  - 9800 WLC Configuration
    - Step 1. Collect AireOS Mobility Information
    - Step 2. Add the AireOS WLC Information into the 9800 WLC
- Verify
  - AireOS WLC Verification
  - Catalyst 9800 WLC Verification
- Troubleshoot
  - AireOS WLC
  - Catalyst 9800 WLC
  - Radio Active Tracing
  - Embedded Packet Capture
- Common Troubleshoot Scenarios
  - Control and Data Path Down Due to Connectivity Issues
  - Configuration Mismatch Between WLCs
  - DTLS Handshake Issues
  - The HA SSO Scenario
- Related Information

Introduction

This document describes mobility configuration scenarios that cover topologies between Catalyst 9800 Wireless LAN Controllers (WLCs) and AireOS WLCs.

Prerequisites

Requirements

Cisco recommends knowledge of these topics:

- CLI or GUI access to the wireless controllers.

Components Used

- AireOS WLC version 8.10 MR1 or later. You can also use Inter Release Controller Mobility (IRCM) special 8.5 images
- 9800 WLC, Cisco IOS® XE v17.3.4

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, ensure that you understand the potential impact of any command.

Configure

Network Diagram



Guidelines and Restrictions

Mobility Group name on 9800 out of the box is default.

Note:
1) In cases where WLCs are in different subnets, ensure that port UDP 16666 and 16667 is open between them.
2) It is recommended that both 9800 WLCs run the same version so clients that roam across have consistent experience in both Layer 3 roam and guest anchor scenarios.

Mobility Tunnel Between Two Catalyst 9800 WLCs

This basic example describes how to set up mobility across two 9800 controllers. This is commonly used for Guest access (anchor), or to allow clients to roam across controllers and preserve client identity.

When you configure mobility on C9800, first thing to choose is the mobility group name. The prepopulated mobility group name is a default, but you can customize it to a desired value.

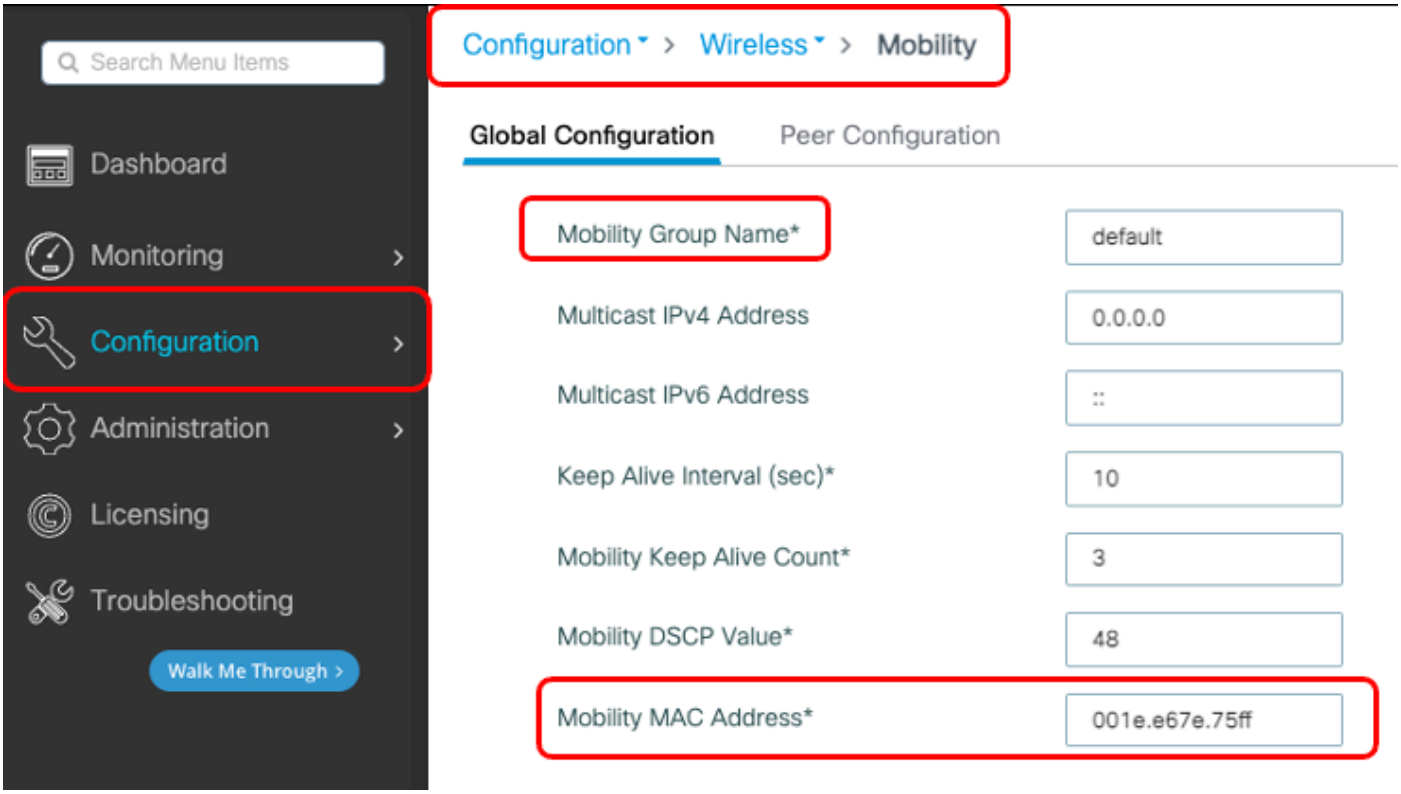
You must configure the same mobility group name across controllers when a fast layer2 roam like Fast Transition (FT) or Cisco Centralized Key Management (CCKM) is in use.

By default, the base Ethernet mac address of the chassis as seen in show version is reflected on GUI for mobility MAC Address. On CLI, by default, the mobility mac is 0000.0000.000 as seen in show run all | inc mobility mac-address.

In cases where 9800s are paired for High Availability (HA) Stateful Switchover (SSO):

If the configuration is left at default and the chassis MAC address is used to form mobility tunnel, the Active chassis and mobility tunnel fail when failover occurs. Therefore, it is mandated that a mobility MAC address be configured for C9800 HA pair.

Step 1: On GUI, navigate to **Configuration > Wireless > Mobility > Global Configuration**.



Via the CLI:

```
# config t
# wireless mobility mac-address  <AAAA.BBBB.CCCC>
# wireless mobility group name <mobility-group-name>
```

Step 1. Collect Mobility Configuration of both 9800 WLCs

For both 9800 WLCs, navigate to **Configuration > Wireless > Mobility > Global Configuration** and take note of its Mobility Group Name and Mobility MAC Address.

Via the CLI:

```
#show wireless mobility summary

Mobility Summary

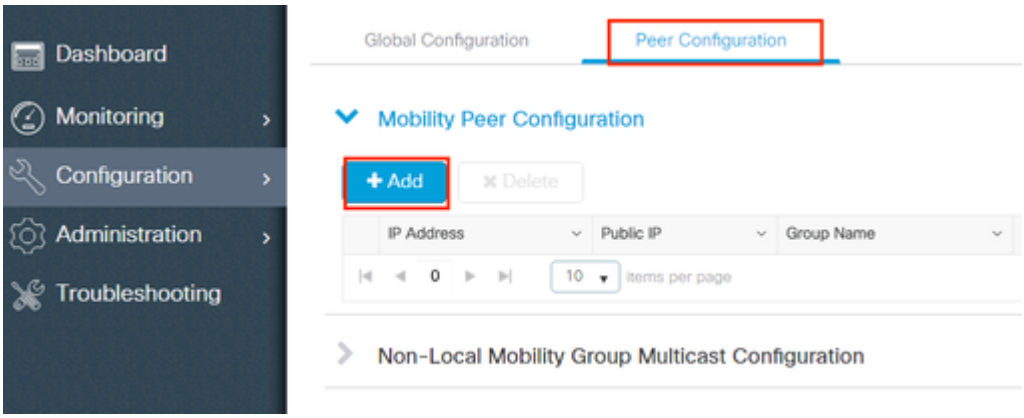
Wireless Management VLAN: 2652
```

```
Wireless Management IP Address: 172.16.51.88
Wireless Management IPv6 Address:
Mobility Control Message DSCP Value: 48
Mobility Keepalive Interval/Count: 10/3
Mobility Group Name: default
Mobility Multicast Ipv4 address: 0.0.0.0
Mobility Multicast Ipv6 address: ::
Mobility MAC Address: 001e.e67e.75ff
Mobility Domain Identifier: 0x34ac
```

Step 2. Add Peer Configuration

Navigate to **Configuration** > **Wireless** > **Mobility** > **Peer Configuration** and enter the peer controller information. Do the same for both 9800 WLCs.

Via the GUI:





Via the CLI:

```
# config t
# wireless mobility group member mac-address <peer-mac-address> ip <peer-ip-address> group <group-name>
```

Note: Optionally, you can enable Data Link Encryption.

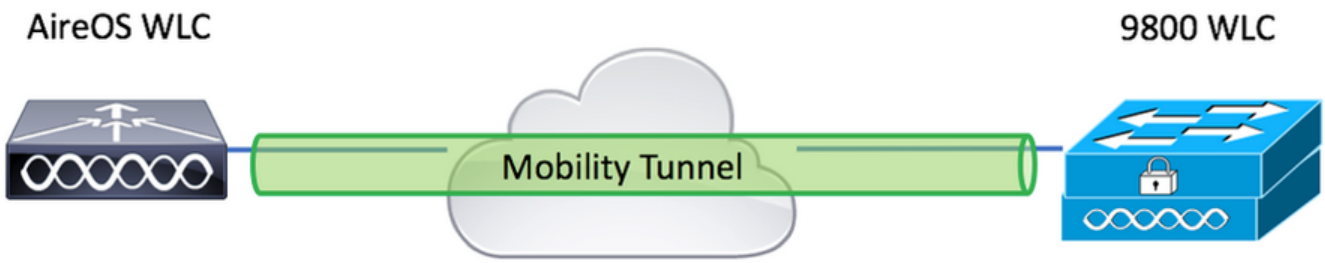
Mobility Tunnel Between AireOS WLC and 9800-CL controllers

This scenario is normal for brownfield deployments or during controller migration, where you split the network in an area of access points (APs) controlled by an AireOS controller, and another by a 9800.

It is advisable that the APs are distributed across controllers per physical or RF areas, so that clients only roam between controllers when they move across.

Avoid salt and pepper deployment. Optionally, this mobility topology also could be used for guest anchor where 9800 acts as foreign and a AireOS as anchor controller.

Network Diagram



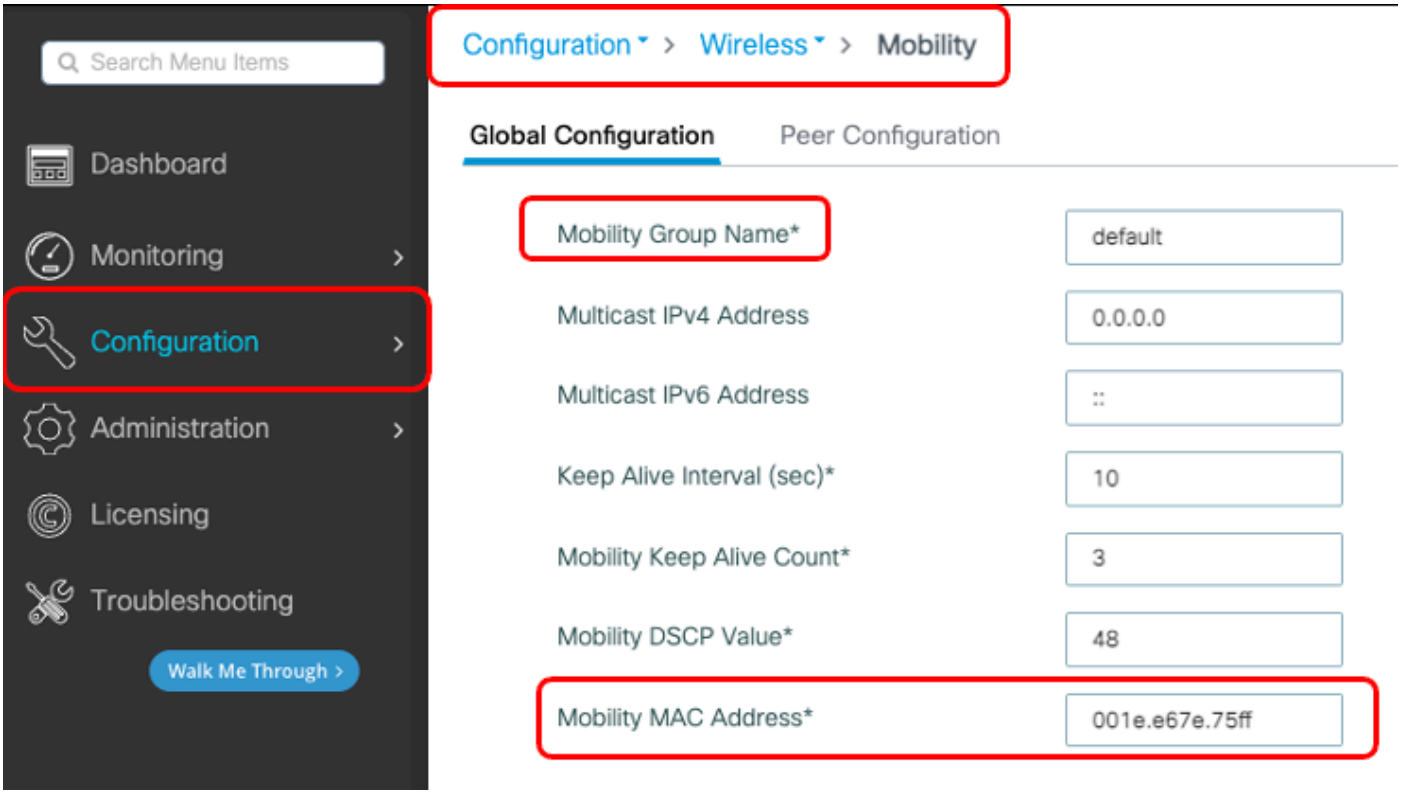
AireOS WLC Configuration

If your 9800 controllers are in High Availability, ensure you have configured the mobility MAC address.

Step 1. Collect 9800 WLC Mobility Information

Via the GUI:

Navigate to **Configuration > Wireless > Mobility > Global Configuration** and take note of its **Mobility Group Name** and **Mobility MAC Address.**



Via the CLI:

<#root>

```
#show wireless mobility summary
```

```
Mobility Summary

Wireless Management VLAN: 2652
Wireless Management IP Address: 172.16.51.88
Wireless Management IPv6 Address:
Mobility Control Message DSCP Value: 48
Mobility Keepalive Interval/Count: 10/3
Mobility Group Name: default
Mobility Multicast Ipv4 address: 0.0.0.0
Mobility Multicast Ipv6 address: ::
Mobility MAC Address: 001e.e67e.75ff
Mobility Domain Identifier: 0x34ac
```

Step 2. Collect the Hash Value from the 9800 WLC

<#root>

```
# show wireless management trustpoint
```

```
Trustpoint Name : Jay-9800_WLC_TP
Certificate Info : Available
Certificate Type : SSC
```

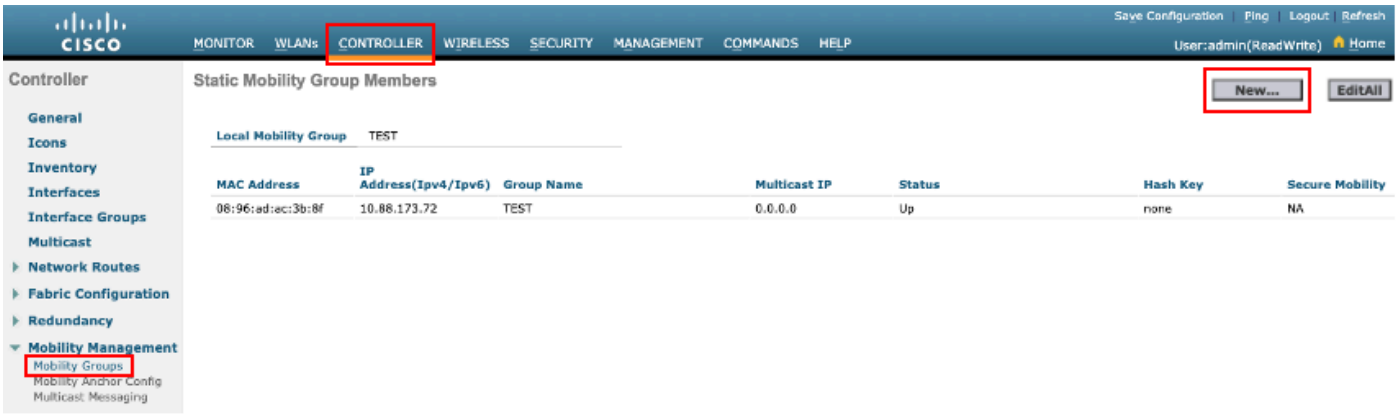
```
Certificate Hash : d7bde0898799dbfeffd4859108727d3372d3a63d
```

```
Private key Info : Available
FIPS suitability : Not Applicable
```

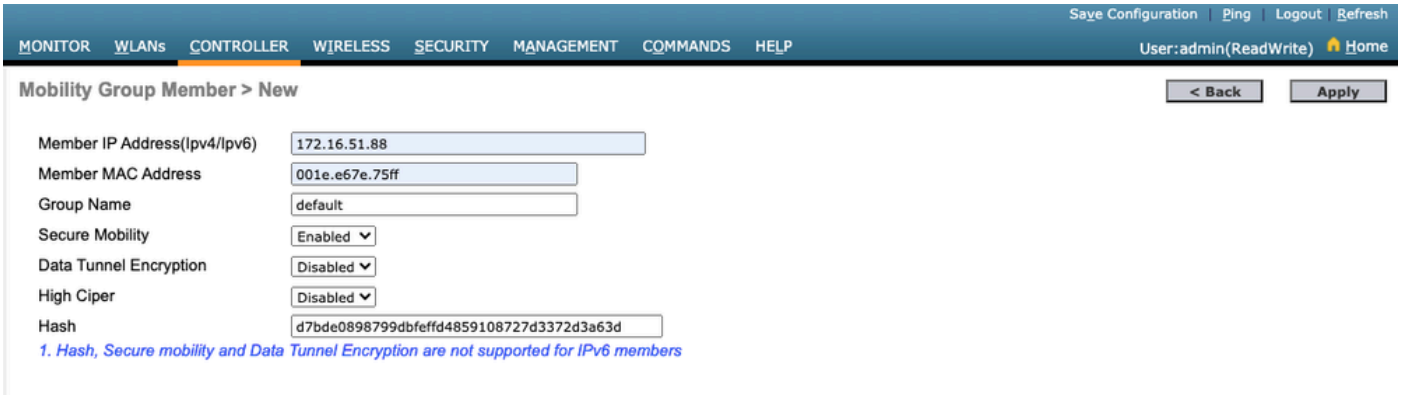
Step 3. Add the 9800 WLC Information into the AireOS WLC

Via the GUI:

Navigate to **CONTROLLER > Mobility Management > Mobility Groups > New**.



Enter the values and click **Apply**.



Note: Hash is only required in cases where the 9800 uses a self-signed certificate such as the C9800-CL. Hardware Appliances have a SUDI certificate and do not need a hash (for example, a 9800-40, 9800-L, and so on).

Via the CLI:

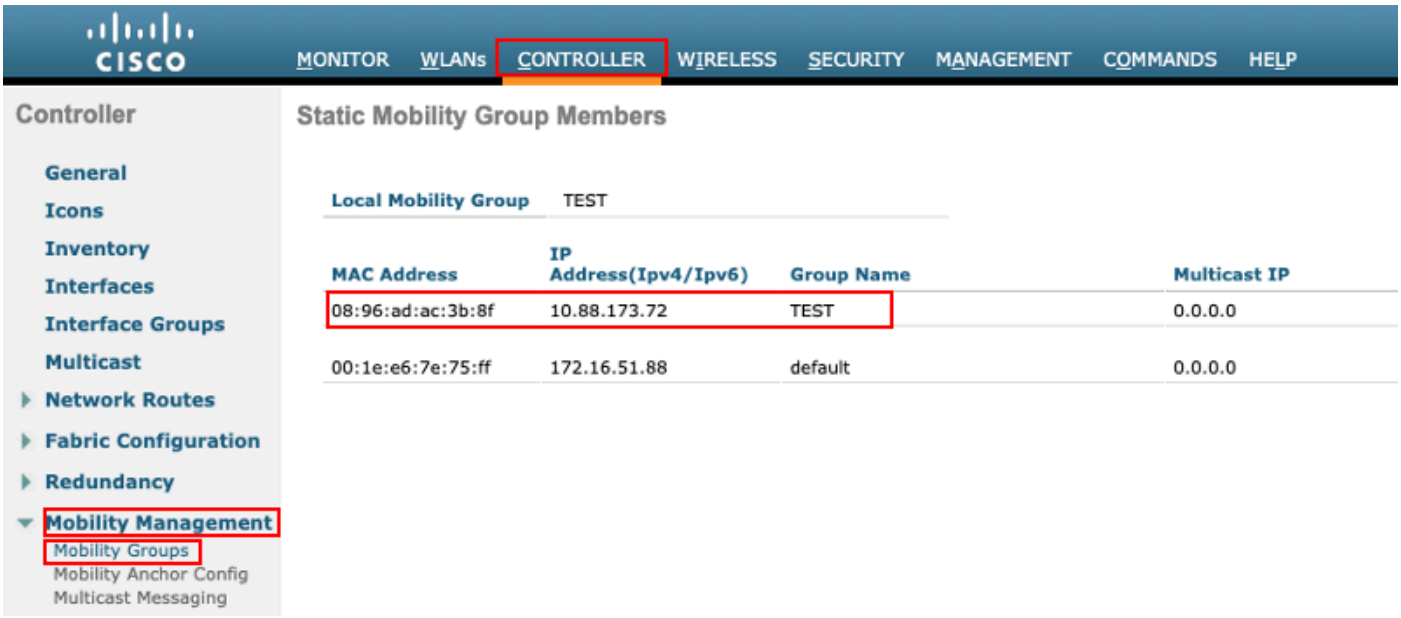
```
>config mobility group member add <9800 mac-address> <9800 WLC-IP> <group-name> encrypt enable
>config mobility group member hash <9800 WLC-IP> <9800 WLC-Hash>
>config mobility group member data-dtls <9800 mac-address> disable
```

9800 WLC Configuration

Step 1. Collect AireOS Mobility Information

Via the GUI:

Log in to AireOS GUI and navigate to **CONTROLLER > Mobility Management > Mobility Groups** and take note of MAC Address, IP Address, and Group Name.



Via the CLI:

```
>show mobility summary

Mobility Protocol Port.......................... 16666
Default Mobility Domain......................... TEST
Multicast Mode ................................. Disabled
Mobility Domain ID for 802.11r.................. 0x6ef9
Mobility Keepalive Interval..................... 10
Mobility Keepalive Count........................ 3
Mobility Group Members Configured............... 2
Mobility Control Message DSCP Value............. 48

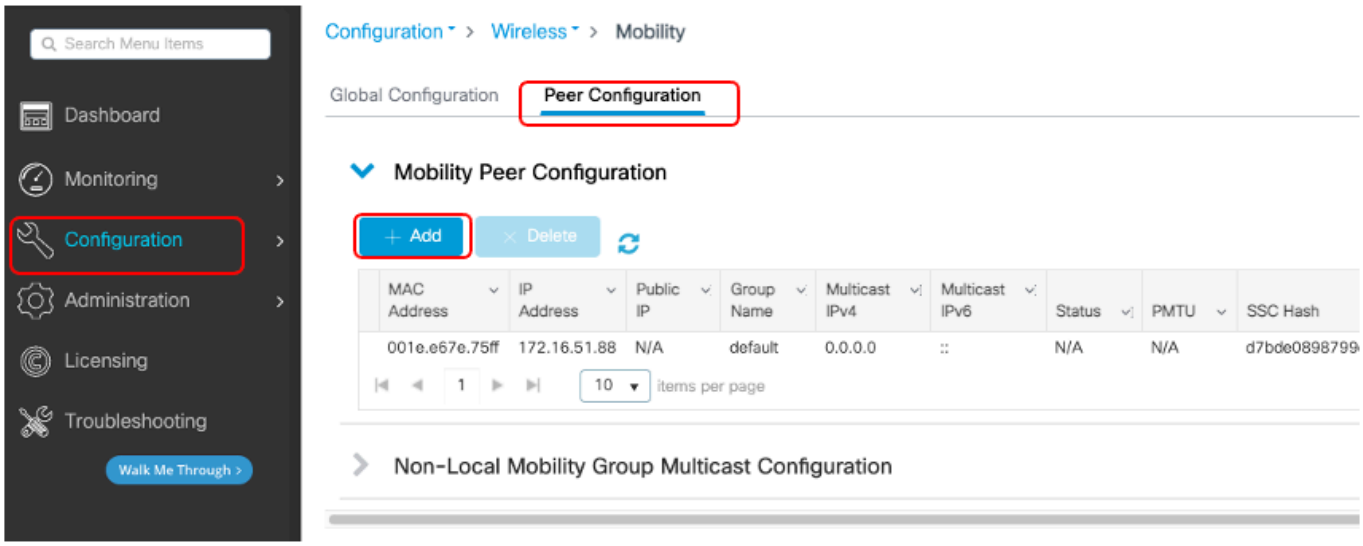
Controllers configured in the Mobility Group

 MAC Address        IP Address         Group Name                  Multicast IP
08:96:ad:ac:3b:8f   10.88.173.72         TEST                        0.0.0.0
```

Step 2. Add the AireOS WLC Information into the 9800 WLC

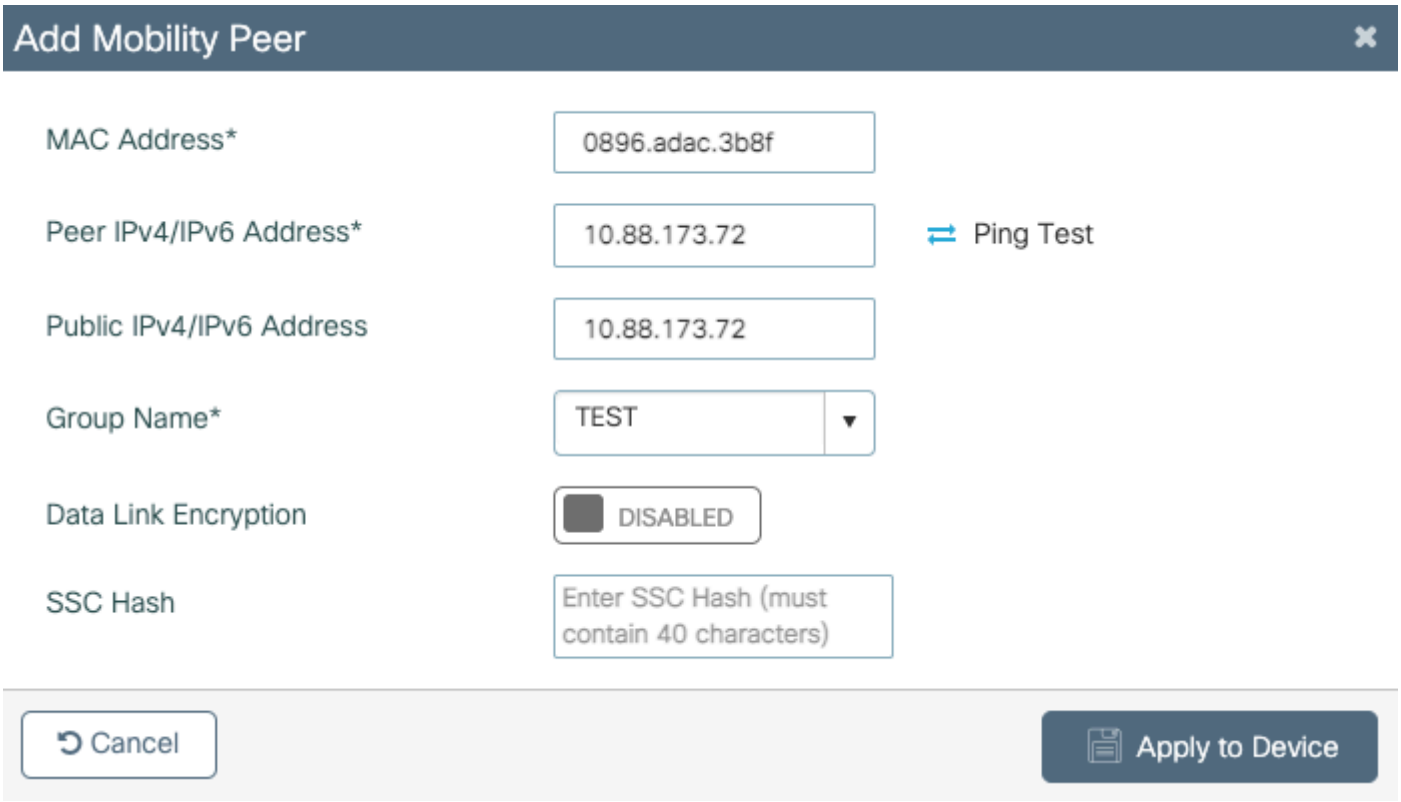
Via the GUI:

Navigate to **Configuration > Wireless > Mobility > Peer Configuration > Add**.



Enter the AireOS WLC information.

Note: On the 9800 WLC, control plane encryption is always enabled, which means that you need to have secure mobility enabled on the AireOS side. However, data link encryption is optional. If you enable it on the 9800 side, enable it on AireOS with: **config mobility group member data-dtls enable**.



Via the CLI:

```
# config t
```

```
# wireless mobility group member mac-address <peer-mac-address> ip <ip-address> group <group-name>
```

Verify

Use this section in order to confirm that your configuration works properly.

AireOS WLC Verification

```
>show mobility summary

Mobility Protocol Port..... 16666
Default Mobility Domain..... TEST
Multicast Mode ..... Disabled
Mobility Domain ID for 802.11r..... 0x6ef9
Mobility Keepalive Interval..... 10
Mobility Keepalive Count..... 3
Mobility Group Members Configured..... 2
Mobility Control Message DSCP Value..... 48

Controllers configured in the Mobility Group
 MAC Address        IP Address                                      Group Name                      M
 00:1e:e6:7e:75:ff  172.16.51.88                                     default
 08:96:ad:ac:3b:8f  10.88.173.72                                     TEST
```

Catalyst 9800 WLC Verification

```
#show wireless mobility summary
Mobility Summary

Wireless Management VLAN: 2652
Wireless Management IP Address: 172.16.51.88
Mobility Control Message DSCP Value: 48
Mobility Keepalive Interval/Count: 10/3
Mobility Group Name: mb-kcg
Mobility Multicast Ipv4 address: 0.0.0.0
Mobility Multicast Ipv6 address: ::
Mobility MAC Address: 001e.e67e.75ff

Controllers configured in the Mobility Domain:

 IP               Public Ip         Group Name                      Multicast IPv4    Multicast IPv6
-------------------------------------------------------------------------------------------------------
172.16.51.88       N/A               default                           0.0.0.0           ::
```

```
10.88.173.72      10.88.173.72        TEST                              0.0.0.0          ::
```

Troubleshoot

This section provides information used to troubleshoot your configuration.

To troubleshoot the mobility tunnel implementation, use these commands to debug the process:

AireOS WLC

Step 1. Enable the mobility debugs.

```
debug mobility handoff enable
debug mobility error enable
debug mobility dtls error enable
debug mobility dtls event enable
debug mobility pmtu-discovery enable
debug mobility config enable
debug mobility directory enable
```

Step 2. Reproduce the configuration and verify the output

Example of a successful mobility tunnel creation on a AirOS WLC.

<#root>

```
*capwapPingSocketTask: Feb 07 09:53:38.507: Client initiating connection on 172.16.0.5:16667 <-> 172.16.
```

```
*capwapPingSocketTask: Feb 07 09:53:38.507: Sending packet to 172.16.0.21:16667
*capwapPingSocketTask: Feb 07 09:53:38.508: Received DTLS packet from mobility peer 172.16.0.21 bytes:
*capwapPingSocketTask: Feb 07 09:53:38.508: mm_dtls2_process_data_rcv_msg:1207 rcvBufLen 48 clr_pkt_len
*capwapPingSocketTask: Feb 07 09:53:38.508: Record    : type=22, epoch=0, seq=0
*capwapPingSocketTask: Feb 07 09:53:38.508:    Hndshk : type=3, len=23 seq=0, frag_off=0, frag_len=23
*capwapPingSocketTask: Feb 07 09:53:38.508: Handshake in progress for link 172.16.0.5:16667 <-> 172.16.
*capwapPingSocketTask: Feb 07 09:53:38.508: Sending packet to 172.16.0.21:16667
*capwapPingSocketTask: Feb 07 09:53:38.508: DTLS consumed packet from mobility peer 172.16.0.21 bytes:
!
!<--output-omited-->
!
```

```
*capwapPingSocketTask: Feb 07 09:53:38.511: dtls2_cert_verify_callback: Forcing Certificate validation a
```

```
*capwapPingSocketTask: Feb 07 09:53:38.511: Peer certificate verified.

*capwapPingSocketTask: Feb 07 09:53:38.511: Handshake in progress for link 172.16.0.5:16667 <-> 172.16.
*capwapPingSocketTask: Feb 07 09:53:38.511: Nothing to send on link 172.16.0.5:16667 <-> 172.16.0.21:16
*capwapPingSocketTask: Feb 07 09:53:38.511: DTLS consumed packet from mobility peer 172.16.0.21 bytes:
*capwapPingSocketTask: Feb 07 09:53:38.511: Received DTLS packet from mobility peer 172.16.0.21 bytes:
*capwapPingSocketTask: Feb 07 09:53:38.511: mm_dtls2_process_data_rcv_msg:1207 rcvBufLen 56 clr_pkt_len
*capwapPingSocketTask: Feb 07 09:53:38.511: Record    : type=22, epoch=0, seq=6
*capwapPingSocketTask: Feb 07 09:53:38.511:    Hndshk : type=13, len=6 seq=3, frag_off=0, frag_len=6
*capwapPingSocketTask: Feb 07 09:53:38.523: Handshake in progress for link 172.16.0.5:16667 <-> 172.16.
*capwapPingSocketTask: Feb 07 09:53:38.523: Sending packet to 172.16.0.21:16667
*capwapPingSocketTask: Feb 07 09:53:38.523: Sending packet to 172.16.0.21:16667
*capwapPingSocketTask: Feb 07 09:53:38.523: Sending packet to 172.16.0.21:16667
*capwapPingSocketTask: Feb 07 09:53:38.523: Sending packet to 172.16.0.21:16667
*capwapPingSocketTask: Feb 07 09:53:38.523: Sending packet to 172.16.0.21:16667
*capwapPingSocketTask: Feb 07 09:53:38.524: Sending packet to 172.16.0.21:16667
*capwapPingSocketTask: Feb 07 09:53:38.524: Sending packet to 172.16.0.21:16667
*capwapPingSocketTask: Feb 07 09:53:38.524: DTLS consumed packet from mobility peer 172.16.0.21 bytes:
*capwapPingSocketTask: Feb 07 09:53:38.527: Received DTLS packet from mobility peer 172.16.0.21 bytes:
*capwapPingSocketTask: Feb 07 09:53:38.527: mm_dtls2_process_data_rcv_msg:1207 rcvBufLen 91 clr_pkt_len
*capwapPingSocketTask: Feb 07 09:53:38.527: Record    : type=20, epoch=0, seq=8
*capwapPingSocketTask: Feb 07 09:53:38.527: Connection established for link 172.16.0.5:16667 <-> 172.16
*capwapPingSocketTask: Feb 07 09:53:38.527: ciperspec 1
*capwapPingSocketTask: Feb 07 09:53:38.527: Nothing to send on link 172.16.0.5:16667 <-> 172.16.0.21:16
*capwapPingSocketTask: Feb 07 09:53:38.527: DTLS consumed packet from mobility peer 172.16.0.21 bytes:

*mmMobility: Feb 07 09:53:38.527: DTLS Action Result message received
*mmMobility: Feb 07 09:53:38.527:  Key plumb succeeded

*mmMobility: Feb 07 09:53:38.527: mm_dtls2_callback: Connection established with 172.16.0.21:16667
*mmMobility: Feb 07 09:53:38.527: mm_dtls2_db_status_up:895 Connections status up for entry 172.16.0.21
*mmMobility: Feb 07 09:53:38.527: mm_dtls2_callback: DTLS Connection established with 172.16.0.21:16667
```

Catalyst 9800 WLC

By default, the 9800 controllers continuously log process information without the need of any special debug procedure.

Simply connect to the controller and retrieve the logs associated with any wireless component for troubleshoot purposes. The logs could span days; that depends on how busy the controller is.

To simplify analysis, pull the logs with a time range or for the last number of minutes (default time is set to 10 minutes) and you can filter by IP or MAC addresses.

Step 1. Check the current on the controller time so you can track the logs in the time back to when the issue happened.

```
# show clock
```

Step 2. Collect the controller logs in case there is any information at Cisco IOS level that could be related to the problem.

```
# show logging
```

Step 3. Collect the always-on notice level traces for a specific address. You can use the mobility peer IP or MAC to filter.

```
# show logging profile wireless filter ipv4 to-file bootflash:ra-AAAA.BBBB.CCCC.txt
```

This command generates logs for the last 10 minutes. It is possible to adjust this time with command **show logging profile wireless last 1 hour filter mac AAAA.BBBB.CCCC to-file bootflash:ra-AAAA.BBBB.CCCC.txt**.

You can either display the content on the session or copy the file to an external TFTP server.

```
# more bootflash:always-on-<FILENAME.txt>

or

# copy bootflash:always-on-<FILENAME.txt> tftp://a.b.c.d/path/always-on-<FILENAME.txt>
```

Radio Active Tracing

If the always-on logs do not provide enough information to know what triggered issues during tunnel configuration, you can enable conditional debugs and capture Radio Active (RA) traces, which give a more detailed process activity.

Step 1. Verify there are no debug conditions already enabled.

```
# show debugging
IOSXE Conditional Debug Configs:

Conditional Debug Global State: Stop


IOSXE Packet Tracing Configs:
```

```
Packet Infra debugs:

Ip Address                                              Port
------------------------------------------------------|----------
```

If you see any condition that is not related to the address that you want to monitor, disable it.

To remove a specific address:

```
# no debug platform condition feature wireless { mac <aaaa.bbbb.cccc> | ip <a.b.c.d> }
```

To remove all the conditions (recommended way):

```
# clear platform condition all
```

Step 2. Add the debug condition for an address that you want to monitor.

```
# debug platform condition feature wireless ip <a.b.c.d>
```

Note: If you want to monitor more than one mobility peer at the same time, use a **debug platform condition feature wireless mac** command per MAC address.

Step 3. Have the 9800 WLC to start monitor of the specified address activity.

```
# debug platform condition start
```

Note: Output of the mobility activity is not displayed as everything is buffered internally to be collected later.

Step 4. Reproduce the issue or the behavior that you want to monitor.

Step 5. Stop the debugs.

```
# debug platform condition stop
```

Step 6. Collect the output of the address activity.

```
# show logging profile wireless filter ipv4 to-file bootflash:ra-AAAA.BBBB.CCCC.txt
```

This command generates logs for the last 10 minutes. It is possible to adjust this time with command **show logging profile wireless last 1 hour filter mac AAAA.BBBB.CCCC to-file bootflash:ra-AAAA.BBBB.CCCC.txt**.

You can either copy the FILENAME.txt to an external server or display the output directly on the screen.

Copy the file to an external server:

```
# copy bootflash:FILENAME.txt tftp://a.b.c.d/ra-FILENAME.txt
```

Display the content:

```
# more bootflash:ra-FILENAME.txt
```

Step 7. If you are still not able to find the reason for a failure, collect the internal level of logs. (You do not need to debug the client again. Use the logs that were already stored internally, but collect a wider range of them).

```
# show logging profile wireless internal filter ipv4 to-file bootflash:raInternal-AAAA.BBBB.CCCC.txt
```

You can either copy the FILENAME.txt to an external server or display the output directly on the screen.

Copy the file to an external server:

```
# copy bootflash:FILENAME.txt tftp://a.b.c.d/ra-FILENAME.txt
```

Display the content:

```
# more bootflash:ra-FILENAME.txt
```

Step 8. Remove the debug conditions.

```
# clear platform condition all
```

Note: Always remove the debug conditions after a troubleshoot session.

Example of a successful mobility tunnel creation on a 9800 WLC.

<#root>

```
2021/09/28 10:20:50.497612 {mobilityd_R0-0}{1}: [errmsg] [26516]: (info): %MM_NODE_LOG-6-MEMBER_ADDED: A


2021/09/28 10:20:52.595483 {mobilityd_R0-0}{1}: [mm-client] [26516]: (debug): MAC: 0000.0000.0000 Sendi
2021/09/28 10:20:52.595610 {mobilityd_R0-0}{1}: [mm-pmtu] [26516]: (debug): Peer IP: 172.16.55.28 PMTU
2021/09/28 10:20:52.595628 {mobilityd_R0-0}{1}: [mm-client] [26516]: (debug): MAC: 0000.0000.0000 Sendi
2021/09/28 10:20:52.595686 {mobilityd_R0-0}{1}: [mm-keepalive] [26516]: (note): Peer IP: 172.16.55.28 k
2021/09/28 10:20:52.595694 {mobilityd_R0-0}{1}: [mm-keepalive] [26516]: (note): Peer IP: 172.16.55.28 k
2021/09/28 10:21:02.596500 {mobilityd_R0-0}{1}: [mm-client] [26516]: (debug): MAC: 0000.0000.0000 Sendi
2021/09/28 10:21:02.596598 {mobilityd_R0-0}{1}: [mm-keepalive] [26516]: (note): Peer IP: 172.16.55.28 k
2021/09/28 10:21:02.598898 {mobilityd_R0-0}{1}: [mm-client] [26516]: (debug): MAC: 001e.e68c.5dff Recei
2021/09/28 10:21:12.597912 {mobilityd_R0-0}{1}: [mm-client] [26516]: (debug): MAC: 0000.0000.0000 Sendi
2021/09/28 10:21:12.598009 {mobilityd_R0-0}{1}: [mm-keepalive] [26516]: (note): Peer IP: 172.16.55.28 D

2021/09/28 10:21:12.598361 {mobilityd_R0-0}{1}: [errmsg] [26516]: (note): %MM_NODE_LOG-5-KEEP_ALIVE: Mob


! !<--output-omitted--> !

2021/09/28 10:21:22.604098 {mobilityd_R0-0}{1}: [ewlc-infra-evq] [26516]: (debug): DTLS record type: 22

2021/09/28 10:21:22.604099 {mobilityd_R0-0}{1}: [ewlc-infra-evq] [26516]: (info): DTLS client hello


2021/09/28 10:21:22.611477 {mobilityd_R0-0}{1}: [ewlc-infra-evq] [26516]: (debug): DTLS record type: 22
2021/09/28 10:21:22.611555 {mobilityd_R0-0}{1}: [ewlc-infra-evq] [26516]: (debug): DTLS record type: 22
2021/09/28 10:21:22.611608 {mobilityd_R0-0}{1}: [ewlc-infra-evq] [26516]: (debug): DTLS record type: 22
2021/09/28 10:21:22.611679 {mobilityd_R0-0}{1}: [ewlc-infra-evq] [26516]: (debug): DTLS record type: 22
2021/09/28 10:21:22.611933 {mobilityd_R0-0}{1}: [mm-dtls] [26516]: (note): Peer IP: 172.16.55.28 Port:

2021/09/28 10:21:22.612163 {mobilityd_R0-0}{1}: [ewlc-dtls-sessmgr] [26516]: (info): Remote Host: 172.16


! !<--output-omitted--> !

2021/09/28 10:21:52.603200 {mobilityd_R0-0}{1}: [mm-keepalive] [26516]: (note): Peer IP: 172.16.55.28 C
```

```
2021/09/28 10:21:52.604109 {mobilityd_R0-0}{1}: [errmsg] [26516]: (note): %MM_NODE_LOG-5-KEEP_ALIVE: Mob
```

Embedded Packet Capture

Most times, it is very useful to check packets exchanged between WLCs. It is especially useful to filter captures with Access Control Lists (ACLs) in order to limit captured traffic.

This is a configuration template for embedded captures on CLI.

Step 1. Create the filter ACL:

```
conf t
ip access-list extended <ACL_NAME>
10 permit ip host <WLC_IP_ADDR> host <PEER_WLC_IP_ADDR>
20 permit ip host <PEER_WLC_IP_ADDR>host <WLC_IP_ADDR>
end
```

Step 2. Define the capture parameters:

```
monitor capture <CAPTURE_NAME> access-list <ACL_NAME> buffer size 10 control-plane both interface <INTE
```

Note: Select management interface for INTERFACE_NAME parameter.

Step 3. Start the capture:

```
monitor capture <CAPTURE_NAME> start
```

Step 4. Stop the capture:

```
monitor capture <CAPTURE_NAME> stop
```

Step 5. Navigate to **Troubleshooting > Packet Capture** on GUI to collect packet capture file.

Common Troubleshoot Scenarios

The next examples consist of tunnels formed between 9800 WLCs.

Control and Data Path Down Due to Connectivity Issues

Enable **Always-On-Logs** and **Embedded packet captures** to provide additional information to troubleshoot:

<#root>

```
2021/09/28 09:54:22.490625 {mobilityd_R0-0}{1}: [mm-client] [26516]: (debug): MAC: 0000.0000.0000

Sending keepalive_ctrl_req of XID

 (80552) to (ipv4: 172.16.55.28 )
2021/09/28 09:54:22.490652 {mobilityd_R0-0}{1}: [mm-keepalive] [26516]: (note): Peer IP: 172.16.55.28 k
2021/09/28 09:54:22.490657 {mobilityd_R0-0}{1}: [mm-keepalive] [26516]: (note): Peer IP: 172.16.55.28

keepalive ctrl packet misssed, total missed packet = 10


2021/09/28 09:54:32.491952 {mobilityd_R0-0}{1}: [mm-client] [26516]: (debug): MAC: 0000.0000.0000

Sending keepalive_data of XID

 (0) to (ipv4: 172.16.55.28 )
2021/09/28 09:54:32.492127 {mobilityd_R0-0}{1}: [mm-keepalive] [26516]: (note): Peer IP: 172.16.55.28

keepalive data packet misssed, total missed packet = 30
```

Packet captures are useful to confirm behavior.

```
 90 2021-09-28 12:33:52.924939 172.16.51.88    172.16.55.28    116 Mobi-Control - PingReq[Malformed Packet]
 91 2021-09-28 12:34:02.925946 172.16.51.88    172.16.55.28    172 Mobi-Data Keep-Alive - Mobility CAPWAP Ping Request
 92 2021-09-28 12:34:12.925946 172.16.51.88    172.16.55.28    172 Mobi-Data Keep-Alive - Mobility CAPWAP Ping Request
 93 2021-09-28 12:34:22.927945 172.16.51.88    172.16.55.28    172 Mobi-Data Keep-Alive - Mobility CAPWAP Ping Request
 94 2021-09-28 12:34:22.927945 172.16.51.88    172.16.55.28    116 Mobi-Control - PingReq[Malformed Packet]
 95 2021-09-28 12:34:32.927945 172.16.51.88    172.16.55.28    172 Mobi-Data Keep-Alive - Mobility CAPWAP Ping Request
 96 2021-09-28 12:34:42.929944 172.16.51.88    172.16.55.28    172 Mobi-Data Keep-Alive - Mobility CAPWAP Ping Request
 97 2021-09-28 12:34:52.930951 172.16.51.88    172.16.55.28    172 Mobi-Data Keep-Alive - Mobility CAPWAP Ping Request
```

Notice that both debug and WLC show that there is no response to the Control or Data pings. A common scenario shows IP connectivity is allowed but ports 16666 or 16667 are not allowed to communicate across the network.

Configuration Mismatch Between WLCs

In this case, you confirmed connectivity for all ports between WLCs, but continue to notice keepalives miss.

Enable **Always-On-Logs** and **Embedded packet captures** to provide additional information to troubleshoot:

<#root>

```
2021/09/28 11:34:22.927477 {mobilityd_R0-0}{1}: [mm-client] [26516]: (debug): MAC: 0000.0000.0000 Sendi
2021/09/28 11:34:22.928025 {mobilityd_R0-0}{1}: [mm-pmtu] [26516]: (debug): Peer IP: 172.16.55.28 PMTU
2021/09/28 11:34:22.928043 {mobilityd_R0-0}{1}: [mm-client] [26516]: (debug): MAC: 0000.0000.0000 Sendi
2021/09/28 11:34:22.928077 {mobilityd_R0-0}{1}: [mm-keepalive] [26516]: (note): Peer IP: 172.16.55.28
```

keepalive data packet misssed

```
, total missed packet = 8
2021/09/28 11:34:22.928083 {mobilityd_R0-0}{1}: [mm-keepalive] [26516]: (note): Peer IP: 172.16.55.28
```

keepalive ctrl packet misssed

```
, total missed packet = 3
```

Internal logs on peer 172.16.55.28 help you confirm configuration mismatch

<#root>

```
2021/09/28 17:33:22.963 {mobilityd_R0-0}{1}: [mm-keepalive] [27081]: (ERR): Peer IP: 172.16.51.88
```

Failed to validate endpoint: Invalid argument

```
2021/09/28 17:33:22.963 {mobilityd_R0-0}{1}: [errmsg] [27081]: (ERR): %MM_NODE_LOG-3-PING_DROPPED: Drop
```

Common configuration mismatch include: incorrect group name, mismatch on Data Link Encryption and incorrect Mobility mac address.

Group mismatch log:

<#root>

```
2021/09/28 17:33:22.963 {mobilityd_R0-0}{1}: [errmsg] [27081]: (ERR): %MM_INFRA_LOG-3-MSG_PROC_FAILED_G
```

Failed to validate endpoint. reason: Group name hash mismatch.

MAC address mismatch log:

<#root>

```
2021/09/28 19:09:33.455 {mobilityd_R0-0}{1}: [errmsg] [27081]: (ERR): %MM_INFRA_LOG-3-MSG_PROC_FAILED_M
```

Failed to validate endpoint. reason: MAC address mismatch.

DTLS Handshake Issues

This kind of issue is related with DTLS tunnel establishments between WLCs. It could be the case Data path is UP but Control path remains DOWN.

Enable **Always-On-Logs** and **Embedded packet captures** to provide additional information to troubleshoot:

<#root>

```
2021/09/28 19:30:23.534 {mobilityd_R0-0}{1}: [mm-msg] [27081]: (ERR): Peer IP: 172.16.51.88 Port: 16666
2021/09/28 19:30:23.534 {mobilityd_R0-0}{1}: [errmsg] [27081]: (warn): %MM_NODE_LOG-4-DTLS_HANDSHAKE_FA
```

Mobility DTLS Ctrl handshake failed for 172.16.51.88 HB is down

```
, need to re-initiate DTLS handshake
2021/09/28 19:30:23.534 {mobilityd_R0-0}{1}: [ewlc-capwapmsg-sess] [27081]: (ERR): Source IP:172.16.51.
```

Use **show wireless management trustpoint** and **show crypto pki trustpoints** commands to verify your certificate information.

The HA SSO Scenario

If you have controllers in High Availability SSO pair, there is an important catch to know. The mobility MAC address is not configured by default and it can cause the mobility tunnel to go down if a failover happens.

The **show wireless mobility summary** gives you the current mobility MAC in use, but it is not necessarily configured. Check if the configuration has the mobility MAC configured with **show run | i mobility**.

If the mobility mac is not configured in the running configuration, it changes upon failover to the standby WLC and this causes mobility tunnels to fail.

The simple solution is to navigate to the **Configuration > Wireless > Mobility** web UI page and

select **apply**. This saves the current mobility MAC to the configuration. The MAC then stays the same upon failover and mobility tunnels are preserved.

This issue mainly happens if you do your mobility configuration through the command line and forget to configure the mobility MAC address. The web UI automatically saves a mobility MAC address when you apply the settings.

Related Information

- Configure WLAN Anchor Mobility Feature on Catalyst 9800
- Technical Support & Documentation - Cisco Systems