Troubleshoot the mDNS Gateway on Wireless LAN Controller (WLC)

Contents

Introduction

This document describes the implementation of the Bonjour protocol on the wireless controller and provides guidelines to help troubleshoot issues.

Prerequisites

Requirements

Cisco recommends that you have knowledge of these topics:

- Basic knowledge of Bonjour Protocol
- Basic knowledge of how to configure mDNS on WLC
- Basic knowledge of Multicast routing

Components Used

The information in this document is based on these software and hardware versions:

- AIR-CT2504-K9 WLC, 8.2.141.0
- WS-C3560CX-8PC-S
- AIR-CAP3702I-E-K9
- Apple TV
- Iphone5s, 10.2

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.

Background Information

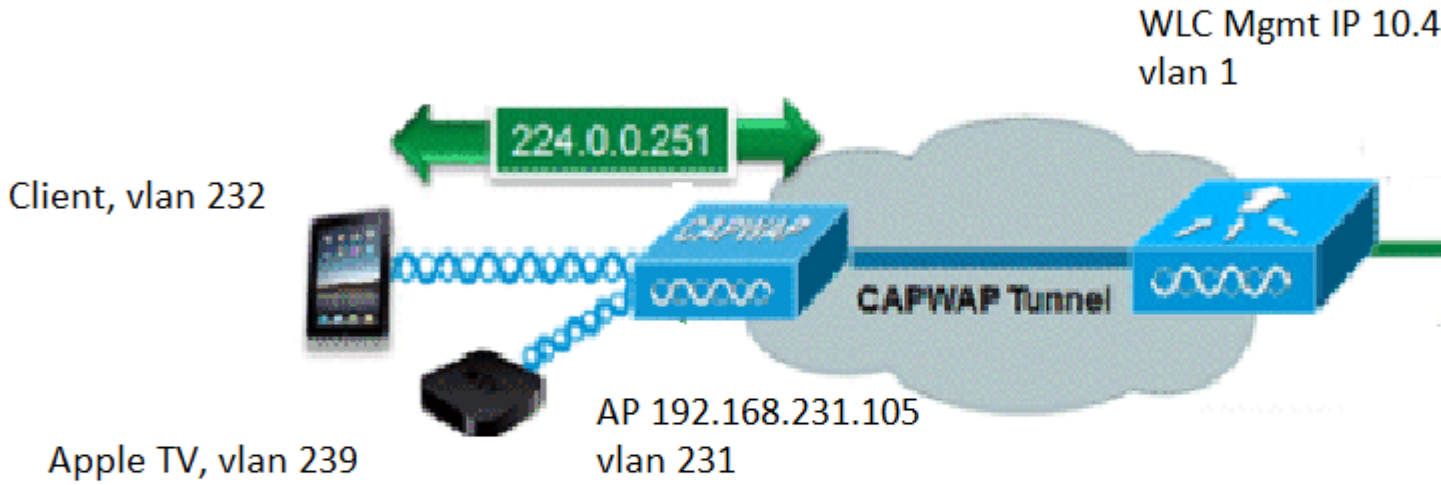
Bonjour protocol is an Apple service discovery protocol which locates devices and sevices on a local network with the use of multicast Domain Name System (mDNS) service records. The Bonjour protocol operates on service announcements and service queries. Each query or advertisement is sent to the Bonjour multicast address ipv4 224.0.0.251 (ipv6 FF02::FB). This protocal uses mDNS on UDP port 5353.

The address used by the Bonjour protocol is link-local multicast address and therefore is only forwarded to the local L2 network. Routers cannot use multicast routing to redirect the traffic because the time to live (TTL) is set to 1. This meant that all the service providers/sources (which advertise the service) and Bonjour clients(which ask for service) had to be in the same subnet. This lead to scability problems.

In order to address this issue, the Cisco Wireless LAN Controller (WLC) acts as a Bonjour Gateway. The WLC listens for Bonjour services, caches these Bonjour advertisements (AirPlay, AirPrint, and so on) from the source/host. For example, Apple TV and responds back to Bonjour clients when they ask/request for a service. This way you can have the sources and clients in different subnets.

Configure

Network Diagram



Packet Flow with Debugs

There are basic four steps which take place when mDNS runs on a Cisco WLC. These steps are described as follows:

Step 1. When You Globally Enable mDNS on the WLC

WLC listens to these default services if you does not have a customized mDNS profile created as shown in the image.

Service Name	Service String	Query Status	LSS Status	Ori
AirPrint	_ipp._tcp.local.	✔	☐	ALL
AirTunes	_raop._tcp.local.	✔	☐	ALL
AppleTV	_airplay._tcp.local.	✔	☐	ALL
HP_Photosmart_Printer_1	_universal._sub._ipp._tcp.local.	✔	☐	ALL
HP_Photosmart_Printer_2	_cups._sub._ipp._tcp.local.	✔	☐	ALL
Printer	_printer._tcp.local.	✔	☐	ALL

Each of these services has a service string associated with it. Service strings are used to match service instances to service queries. A service type always contains the service name and the protocol. Additionally, it can contain one or more subtype identifiers. AppleTV service uses: **_airplay._tcp.local.**

When mDNS is enabled globally, the controller sends mDNS queries to 224.0.0.251 for all the services on wired (management and dynamic interfaces) and wireless network.

In this capture at WLC switch port, packets 80, 81 and 82 show WLC sends a query to 224.0.0.251 over the wired network with source IP of the management (10.48.39.142) and dynamic interfaces(192.168.232.11 and 192.168.239.8) as shown in the image.

mdns

No.	Time	Source	Destination	Protocol	Length	Signal strength (dBm)	Cha
80	15:24:18.206675	10.48.39.142	224.0.0.251	MDNS	216		
81	15:24:18.207010	192.168.232.11	224.0.0.251	MDNS	216		
82	15:24:18.207663	192.168.239.8	224.0.0.251	MDNS	216		
83	15:24:18.208051	10.48.39.142	224.0.0.251	MDNS	292		

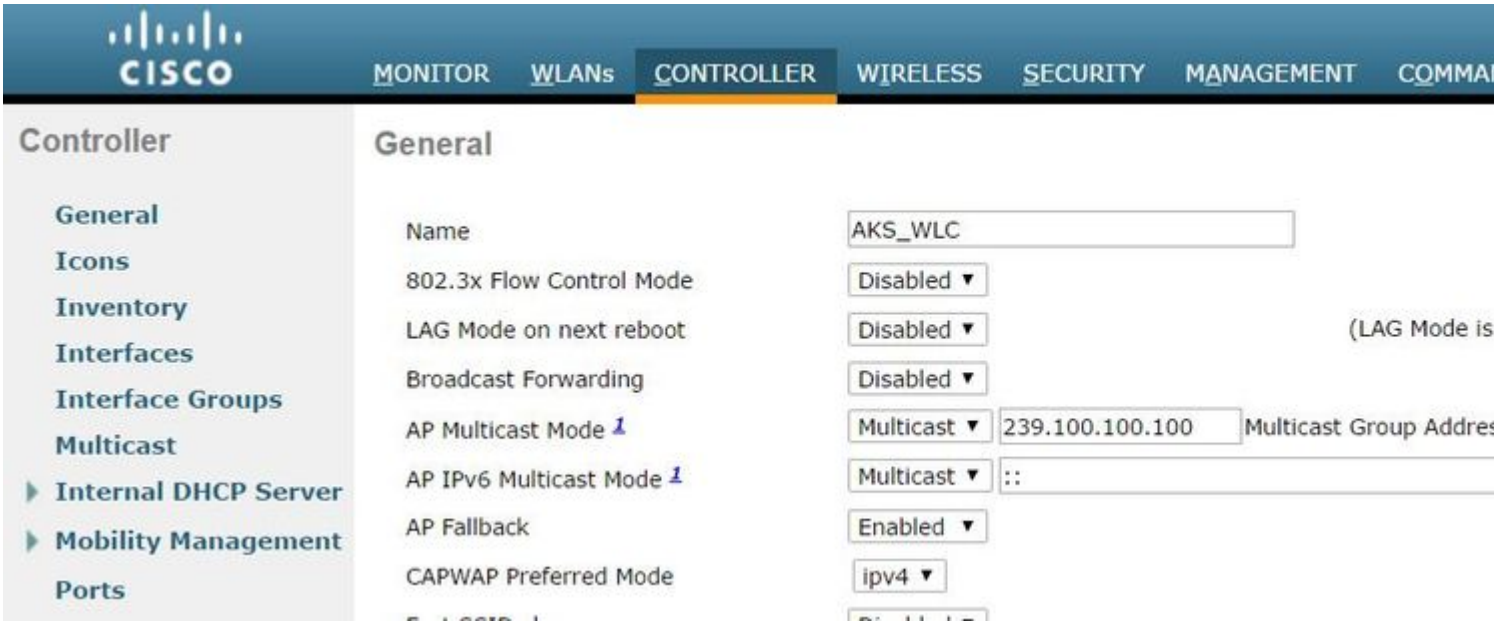
```
> Frame 80: 216 bytes on wire (1728 bits), 216 bytes captured (1728 bits) on interface 0
> Ethernet II, Src: Cisco_b9:62:60 (00:a2:89:b9:62:60), Dst: IPv4mcast_fb (01:00:5e:00:00:fb)
> Internet Protocol Version 4, Src: 10.48.39.142, Dst: 224.0.0.251
> User Datagram Protocol, Src Port: 5353, Dst Port: 5353
> Multicast Domain Name System (query)
```

Packet 83 shows that WLC sends a query over the wireless. The inner packet shows WLC query to 224.0.0.251 from management interface. Since this query is over the wireless, capwap header is added to the packet with outer source IP still to be that of management but the destination is multicast IP 239.100.100.100 as shown in the image.

```
83 16:24:18.208051 10.48.39.142    224.0.0.251    MDNS    292    Standard query
```

```
> Frame 83: 292 bytes on wire (2336 bits), 292 bytes captured (2336 bits) on interface 0
> Ethernet II, Src: Cisco_b9:62:64 (00:a2:89:b9:62:64), Dst: IPv4mcast_64:64:64 (01:00:5e:64:64:64)
> Internet Protocol Version 4, Src: 10.48.39.142, Dst: 239.100.100.100
> User Datagram Protocol, Src Port: 5247, Dst Port: 5247
> Control And Provisioning of Wireless Access Points - Data
> IEEE 802.11 Data, Flags: ......F.
> Logical-Link Control
> Internet Protocol Version 4, Src: 10.48.39.142, Dst: 224.0.0.251
> User Datagram Protocol, Src Port: 5353, Dst Port: 5353
> Multicast Domain Name System (query)
```

Now, where does this multicast IP 239.100.100.100 come from? On the WLC, Access Point (AP) multicast mode (**controller > general**) was set to multicast with multicast group address as 239.100.100.100 (it is just an example, any ip in the 239 range). The APs join this multicast group and listen on it. WLC forwards the query to this group, APs receive it and send it over the air. The address 239.100.100.100 (this is not static, this is what you configured in the next example) only appears in the capwap header between the WLC and the APs, the wireless clients never sees anything of it (but it can see the inner original mdns packet) as shown in the image.



Remember in this setup the WLC is a 2504 in vlan 1 and the AP is in vlan 231. As the devices are in different vlans, you needs to have multicast routing enabled for vlan 1 and 239 on the wired for this to work.

> **Note**: If multicast routing is not enabled on the wired for wlc and AP management vlan, then AP multicast mode must be set to unicast. In this mode, the controller unicasts every multicast packet to every AP associated to the controller. This mode is very inefficient and is not recommended.

This capture is the query packet in detail as shown in the image.

mdns

No.	Time	Source	Destination	Protocol	Length	Signal strength (dBm)
80	15:24:18.206675	10.48.39.142	224.0.0.251	MDNS	216	
81	15:24:18.207010	192.168.232.11	224.0.0.251	MDNS	216	
82	15:24:18.207663	192.168.239.8	224.0.0.251	MDNS	216	
83	15:24:18.208051	10.48.39.142	224.0.0.251	MDNS	292	

```
> User Datagram Protocol, Src Port: 5247, Dst Port: 5247
> Control And Provisioning of Wireless Access Points - Data
> IEEE 802.11 Data, Flags: ......F.
> Logical-Link Control
> Internet Protocol Version 4, Src: 10.48.39.142, Dst: 224.0.0.251
> User Datagram Protocol, Src Port: 5353, Dst Port: 5353
v Multicast Domain Name System (query)
    Transaction ID: 0x0000
  > Flags: 0x0400 Standard query
    Questions: 6
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
  v Queries
    > _ipp._tcp.local: type ANY, class IN, "QU" question
    > _raop._tcp.local: type ANY, class IN, "QU" question
    > _airplay._tcp.local: type ANY, class IN, "QU" question
    > _universal._sub._ipp._tcp.local: type ANY, class IN, "QU" question
    > _cups._sub._ipp._tcp.local: type ANY, class IN, "QU" question
    > _printer._tcp.local: type ANY, class IN, "QU" question
```

The debugs reflect the same thing as seen in the captures. Here the snippet only shows query from management interface.

<#root>

```
(Cisco Controller) >
```

debug mdns all enable

```
Cisco Controller) >*emWeb: Feb 22 16:24:18.203: bgSetBonjourAccessPolicy :1192 Bonjour AccessPolicy stat
*emWeb: Feb 22 16:24:18.203: bgSetBonjourQueryInterval :1359
```

Bonjour query interval is already configured for requested value = 15

```
*Bonjour_Process_Task: Feb 22 16:24:18.215: bonjourProcessTask :
```

220 Processing message type = BONJOUR_AGGREGATED_QUERY

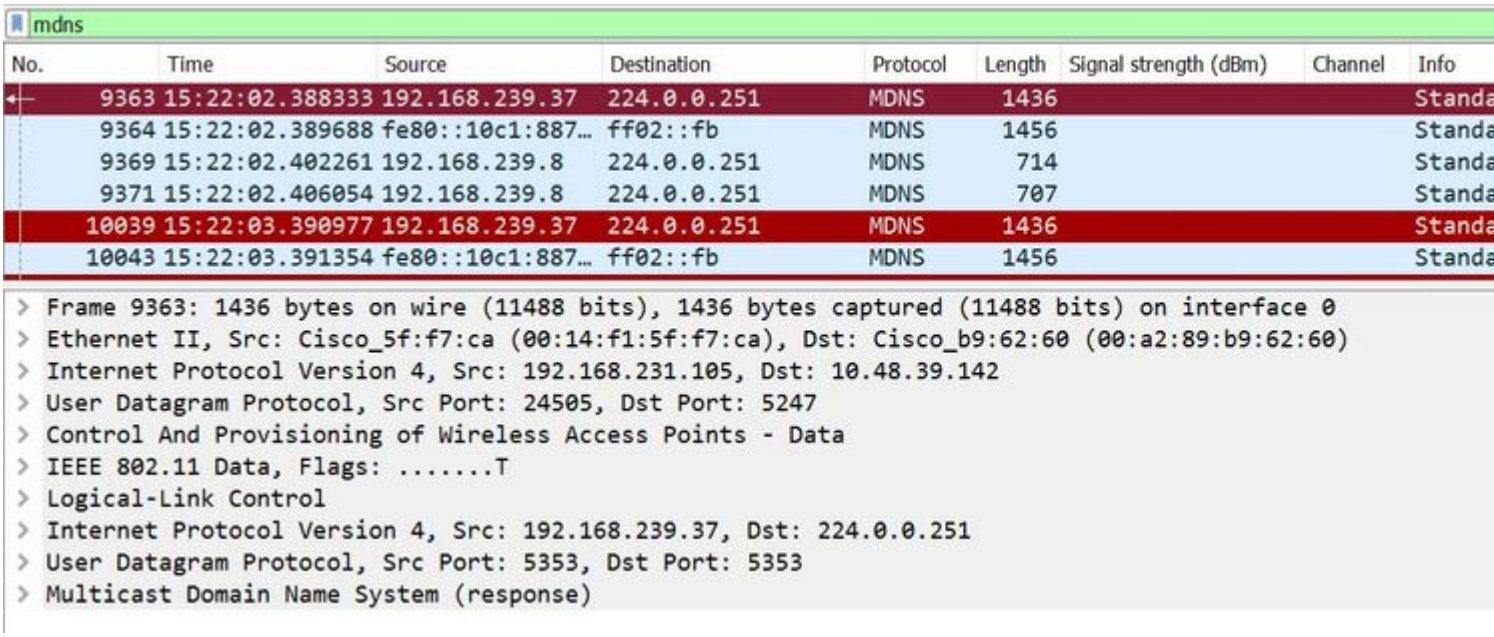
```
*Bonjour_Process_Task: Feb 22 16:24:18.215: sendBonjourPkt : 3881 sendBonjourPkt msg-type = BONJOUR_AGGR
*Bonjour_Process_Task: Feb 22 16:24:18.216: Send to Wired, All vlan is TRUE
*Bonjour_Process_Task: Feb 22 16:24:18.216: sendBonjourPacketToWired : 3652 sending aggregated query on
*Bonjour_Process_Task: Feb 22 16:24:18.216: buildBonjourPacket : 2916 Preparing for l2 Multicast send
*Bonjour_Process_Task: Feb 22 16:24:18.216: buildBonjourPacket : 2936 allVlan = 0 ,
```

vlanId = 0

```
*Bonjour_Process_Task: Feb 22 16:24:18.216: buildBonjourPacket : 2948 simInterfaceMacAddrGet(

management

 ) =  00:A2:89:B9:62:60
*Bonjour_Process_Task: Feb 22 16:24:18.216: Inside buildBonjourAggregatedQuery, available len = 1458
*Bonjour_Process_Task: Feb 22 16:24:18.216: fillBonjourAggregatedQuery : 7339 Sending mDNS AGGREGATED qu
*Bonjour_Process_Task: Feb 22 16:24:18.216: fillBonjourAggregatedQuery : [ 1 ] Including SRV = AirPrint
*Bonjour_Process_Task: Feb 22 16:24:18.216: fillBonjourAggregatedQuery : [ 2 ] Including SRV = AirTunes

*Bonjour_Process_Task: Feb 22 16:24:18.216: fillBonjourAggregatedQuery : [ 3 ] Including SRV = AppleTV i


*Bonjour_Process_Task: Feb 22 16:24:18.216: fillBonjourAggregatedQuery : [ 4 ] Including SRV = HP_Photos
*Bonjour_Process_Task: Feb 22 16:24:18.216: fillBonjourAggregatedQuery : [ 5 ] Including SRV = HP_Photos
*Bonjour_Process_Task: Feb 22 16:24:18.216: fillBonjourAggregatedQuery : [ 6 ] Including SRV = Printer i
*Bonjour_Process_Task: Feb 22 16:24:18.216: ----------------------------------------
*Bonjour_Process_Task: Feb 22 16:24:18.216

: fillBonjourAggregatedQuery : PACKET-1 mDNS-QUERY sent for [ 6 ] services



*Bonjour_Process_Task: Feb 22 16:24:18.216: fillBonjourAggregatedQuery : mDNS-QUERY sent for all service


*Bonjour_Process_Task: Feb 22 16:24:18.216: ----------------------------------------
*Bonjour_Process_Task: Feb 22 16:24:18.216: buildBonjourPacket : 3054 BONJOUR_AGGREGATED_QUERY: buildBon

*Bonjour_Process_Task: Feb 22 16:24:18.216: buildBonjourPacket MCAST-DST-IP ADDR = 224.0.0.251
```

Step 2. WLC Caches Bonjour Services (Apple TV Advertisement)

In this packet, the Apple TV (192.168.239.37) sends advertisements to 224.0.0.251 . Since in this case Apple TV is wireless , you can see the advertisement sent over capwap. The WLC takes note only once of the mDNS service response, however, that cache entry has a TTL and keepalives are necessary to maintain it as shown in the image.



The detailed response from Apple TV is as shown in the image.

mdns

No.	Time	Source	Destination	Protocol	Length	Signal strength (dBm)	Channel	Info
9363	15:22:02.388333	192.168.239.37	224.0.0.251	MDNS	1436			Standar
9364	15:22:02.389688	fe80::10c1:887…	ff02::fb	MDNS	1456			Standar
9369	15:22:02.402261	192.168.239.8	224.0.0.251	MDNS	714			Standar
9371	15:22:02.406054	192.168.239.8	224.0.0.251	MDNS	707			Standar
10039	15:22:03.390977	192.168.239.37	224.0.0.251	MDNS	1436			Standar
10043	15:22:03.391354	fe80::10c1:887…	ff02::fb	MDNS	1456			Standar

```
    [Request In: 9327]
    [Time: 0.040960000 seconds]
    Transaction ID: 0x0000
  > Flags: 0x8400 Standard query response, No error
    Questions: 0
    Answer RRs: 21
    Authority RRs: 0
    Additional RRs: 8
  v Answers
    > 70-35-60-63.1 Wireless Team (4)._sleep-proxy._udp.local: type TXT, class IN, cache flush
    > _services._dns-sd._udp.local: type PTR, class IN, _sleep-proxy._udp.local
    > _sleep-proxy._udp.local: type PTR, class IN, 70-35-60-63.1 Wireless Team (4)._sleep-proxy._udp.local
    > 70-35-60-63.1 Wireless Team (4)._sleep-proxy._udp.local: type SRV, class IN, cache flush, priority 0
    > Wireless Team (4)._airplay._tcp.local: type TXT, class IN, cache flush
    > _services._dns-sd._udp.local: type PTR, class IN, _airplay._tcp.local
    > _airplay._tcp.local: type PTR, class IN, Wireless Team (4)._airplay._tcp.local
    > Wireless Team (4)._device-info._tcp.local: type TXT, class IN
    > 18EE6911DC61@Wireless Team._raop._tcp.local: type TXT, class IN, cache flush
```

These debugs show Apple TV in respond to the WLCâ€™s queries . In this scenario, Apple TV responded with 21 services out of which only Airplay service is of interest.

<#root>

```
*Bonjour_Msg_Task: Feb 23 16:22:02.372:
```

18:ee:69:11:dc:60

Parsing 21 Bonjour Answers.

```
*Bonjour_Msg_Task: Feb 23 16:22:02.374: bgProcessServiceAdvRsp : 1562 aStringNameStr = Wireless Team (4)
*Bonjour_Msg_Task: Feb 23 16:22:02.374: bgProcessServiceAdvRsp : 1579 RR: Wireless Team (4)._airplay._tc
*Bonjour_Msg_Task: Feb 23 16:22:02.374: bgProcessServiceAdvRsp : 1581 aStringNameStr : Wireless Team (4)
*
```

Bonjour_Msg_Task: Feb 23 16:22:02.374: Found Service Name:_airplay._tcp.local., Service Provider Name:Wi

```
*Bonjour_Msg_Task: Feb 23 16:22:02.374: bgServiceAllowedInMsalDb : 181 srv_str = _airplay._tcp.local. ty
*Bonjour_Msg_Task: Feb 23 16:22:02.374: bgServiceAllowedInMsalDb : 195 Incoming Service Advertisement st
```

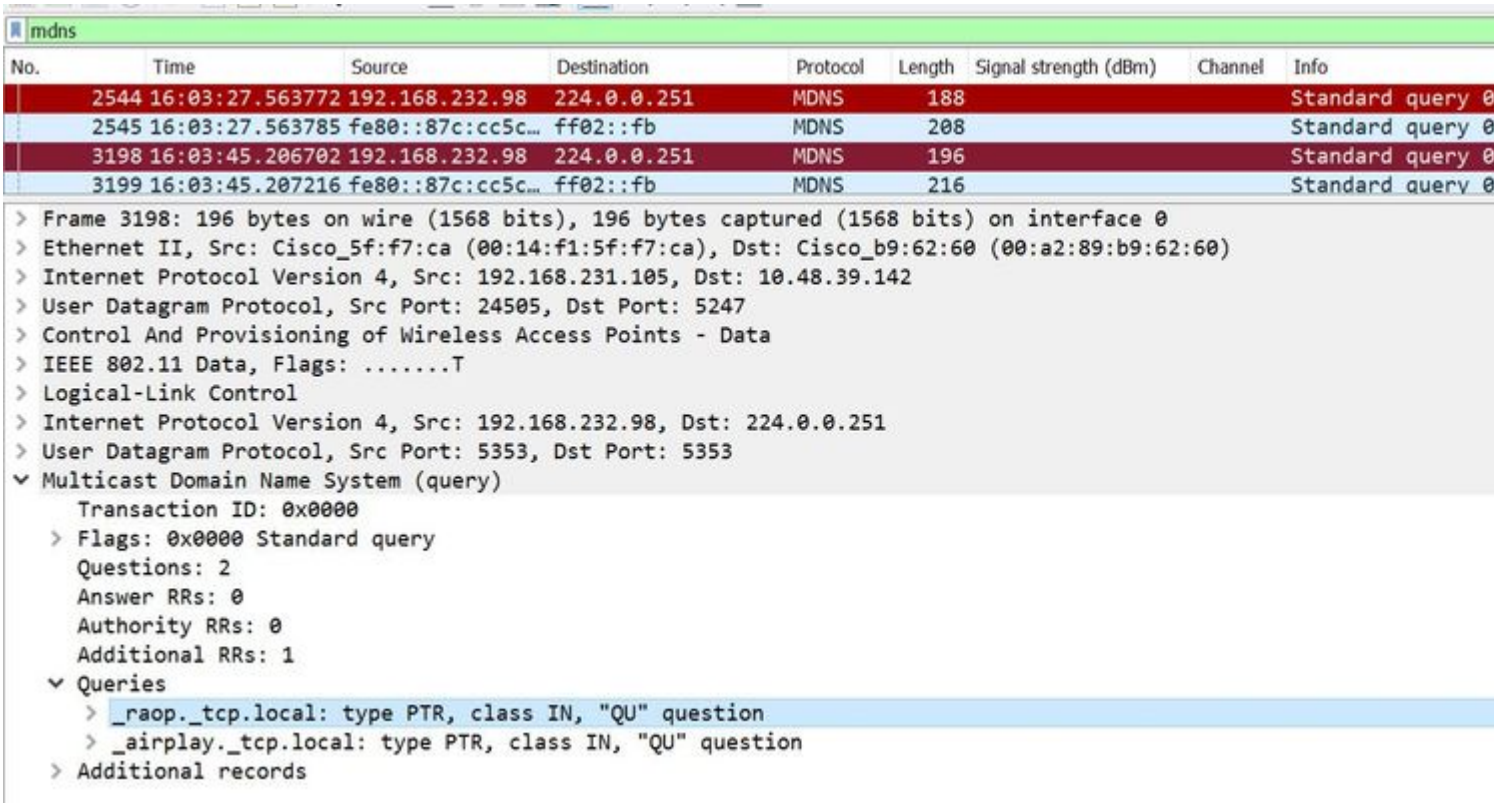
***Bonjour_Msg_Task: Feb 23 16:22:02.374: Service-Name = AppleTV Service-String = _airplay._tcp.local. Typ**

```
<<< Airplay service registered in WLC DB >>
*Bonjour_Msg_Task: Feb 23 16:22:02.374: Service Name:_airplay._tcp.local. is supported in Master-service
*Bonjour_Msg_Task: Feb 23 16:22:02.374: aDataLen: 2, aSrPtrRecord.aSrvProName.size: 39
*Bonjour_Msg_Task: Feb 23 16:22:02.374: Updating updateBonjourSrPtrDb:
*Bonjour_Msg_Task: Feb 23 16:22:02.374: aType: 12, aClass: 1, aTTL: 4500, aDataLen: 2, ptr: 0x327a9d93,
*Bonjour_Msg_Task: Feb 23 16:22:02.374:           bgProcessServiceAdvRsp : .. < SP-SR_PTR PKT >...
```

```
*Bonjour_Msg_Task: Feb 23 16:22:02.374:          bgProcessServiceAdvRsp : SERVICE NAME ......... = Appl

*Bonjour_Msg_Task: Feb 23 16:22:02.374:          bgProcessServiceAdvRsp : SERVICE STRING ....... = _air

*Bonjour_Msg_Task: Feb 23 16:22:02.374:          bgProcessServiceAdvRsp : SERVICE PROVIDER ..... = Wire

.
*Bonjour_Msg_Task: Feb 23 16:22:02.374:          bgProcessServiceAdvRsp : aTTL ................. = 4500
*Bonjour_Msg_Task: Feb 23 16:22:02.374: bgProcessServiceAdvRsp : 1546 msg : 0x327a9bda, ptr : 0x327a9d93
```

Step 3. WLC Listens to Client Queries For Services

Later on, at any point in time, the wireless client (192.168.232.98) sends a query which asks for airplay service over the air (typically when the client opens up an application that is airplay capable) as shown in the image.

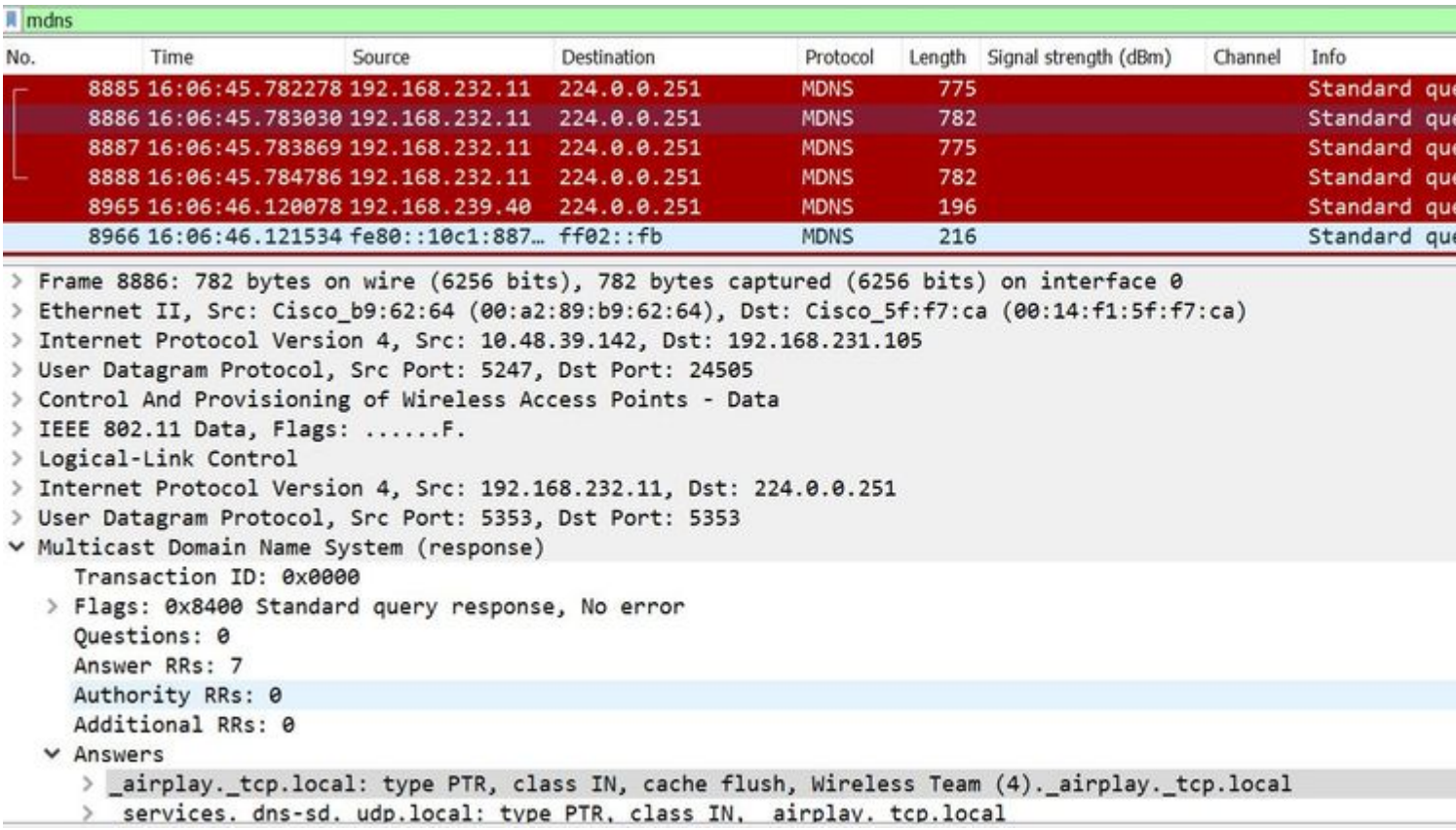


<#root>

```
*Bonjour_Msg_Task: Feb 27 17:03:15.603: 00:6d:52:5d:5a:7d Parsing 2 bonjour questions

*Bonjour_Msg_Task: Feb 27 17:03:15.603: 00:6d:52:5d:5a:7d Query Service Name: _airplay._tcp.local., RR-T

*Bonjour_Msg_Task: Feb 27 17:03:15.603: processBonjourPacket : 1017 qNameStr : _airplay._tcp.local., bon

*Bonjour_Msg_Task: Feb 27 17:03:15.603: Service Name : AppleTV  Service String : _airplay._tcp.local. is
*Bonjour_Msg_Task: Feb 27 17:03:15.603: 00:6d:52:5d:5a:7d SRV : _airplay._tcp.local. is supported by cli
```

Step 4. WLC Sends Unicast Response to Client Queries for Bonjour Services

The WLC responds with cached service Wireless Team (4)._airplay._tcp.local. The source IP of the inner packet is the dynamic interface of the client vlan, in this case 192.168.232.11 as shown in the image.



Snippet from debug

<#root>

BONJOUR_AGGREGATED_QUERY_RESPONSE

```
*Bonjour_Process_Task: Feb 27 17:03:45.229: buildBonjourQueryResponsePld : SRV-NAME ..... : AppleTV

*Bonjour_Process_Task: Feb 27 17:03:45.229: buildBonjourQueryResponsePld : SP-NAME....... :

*Bonjour_Process_Task: Feb 27 17:03:45.229: buildBonjourQueryResponsePld : SEND TO ...... : BONJOUR_PKT_

*Bonjour_Process_Task: Feb 27 17:03:45.229: buildBonjourQueryResponsePld : VLAN ......... : 232

*Bonjour_Process_Task: Feb 27 17:03:45.229: buildBonjourQueryResponsePld : IS MCAST ..... : NO

*Bonjour_Process_Task: Feb 27 17:03:45.230: buildBonjourQueryResponsePld : DST-MAC ...... :  00:6D:52:5D
```

```
*Bonjour_Process_Task: Feb 27 17:03:45.230: buildBonjourQueryResponsePld : DST-IP ....... : 192.168.232.

*Bonjour_Process_Task: Feb 27 17:03:45.230: buildBonjourQueryResponsePld : ALL mDNS-AP .. : 0
*Bonjour_Process_Task: Feb 27 17:03:45.230: buildBonjourQueryResponsePld : TTL COUNTER .. : TIMEOUT_RESE
*Bonjour_Process_Task: Feb 27 17:03:45.230: buildBonjourQueryResponsePld : RESTART TIME . : 0
*Bonjour_Process_Task: Feb 27 17:03:45.230: buildBonjourQueryResponsePld : SNOOP STATUS . : 0
*Bonjour_Process_Task: Feb 27 17:03:45.230: buildBonjourQueryResponsePld : LSS STATUS ... : DISABLED
*Bonjour_Process_Task: Feb 27 17:03:45.230: buildBonjourQueryResponsePld : RSP SRV NAME . : AppleTV
*Bonjour_Process_Task: Feb 27 17:03:45.230: buildBonjourQueryResponsePld : MSG-ID ....... : 0
*Bonjour_Process_Task: Feb 27 17:03:45.230: buildBonjourQueryResponsePld : POLICY STATUS  : DISABLED

*Bonjour_Process_Task: Feb 27 17:03:45.230: buildBonjourQueryResponsePld INCLUDING SpData : Wireless Tea

*Bonjour_Process_Task: Feb 27 17:03:45.233: VALID SR-PTR RR FOUND, attaching.....
*Bonjour_Process_Task: Feb 27 17:03:45.233: VALID SD-PTR RR FOUND, attaching.....
*Bonjour_Process_Task: Feb 27 17:03:45.233: VALID SRV RR FOUND, attaching.....
*Bonjour_Process_Task: Feb 27 17:03:45.233: VALID TXT RR FOUND, attaching.....
*Bonjour_Process_Task: Feb 27 17:03:45.233: VALID NSEC RR FOUND, attaching.....
*Bonjour_Process_Task: Feb 27 17:03:45.233: VALID DOMAIN RR FOUND, attaching.....
*Bonjour_Process_Task: Feb 27 17:03:45.233: fillBonjourDomain : 6055 : attaching SP-DOMAIN RR
*Bonjour_Process_Task: Feb 27 17:03:45.233: VALID DOMAIN-NSEC RR FOUND, attaching.....
*Bonjour_Process_Task: Feb 27 17:03:45.233: buildBonjourPacket DST-IP ADDR = 192.168.232.98

*Bonjour_Process_Task: Feb 27 17:03:45.233: Transmitting bonjour Pkt to STA: 00:6D:52:5D:5A:7D

*Bonjour_Process_Task: Feb 27 17:03:45.233: Unicast Packet sent to client  00:6D:52:5D:5A:7D  success.
```

Verification and Troubleshooting

This section provides information you can use in order to confirm and troubleshoot your configuration.

In order to identify and isolate issues in mdns requires the configuration to be correct and thus requires few basic checks.

Step 1. mDNS must be enabled globally.

From GUI navigate **Controller > mDNS** as shown in the image.

From CLI:

<#root>

```
show network summary

 (snippet)

mDNS snooping.............................. Enabled
mDNS Query Interval........................ 15 minutes
```

Step 2. If you use a custom mDNS profile ensure that all the required services are added to it.

Step 3. Ensure that mDNS is enabled under the SSID and the correct mdns profile is mapped to the SSID.

From GUI navigate to **WLAN > WLAN ID > Advanced** as shown in the image.

From CLI:

<#root>

```
show wlan <ID>

 (snippet)

mDNS Status....................................... Enabled
mDNS Profile Name................................. default-mdns-profile
```

Step 4. Verify whether mDNS service provider is listed in the mDNS domains services. This lists the domain names (Apple TV, airprinters) of the services that have been cached by the WLC.

From GUI, navigate to **Controller > mDNS > mDNS Domain Name IP> Summary** as shown in the image.

From CLI:

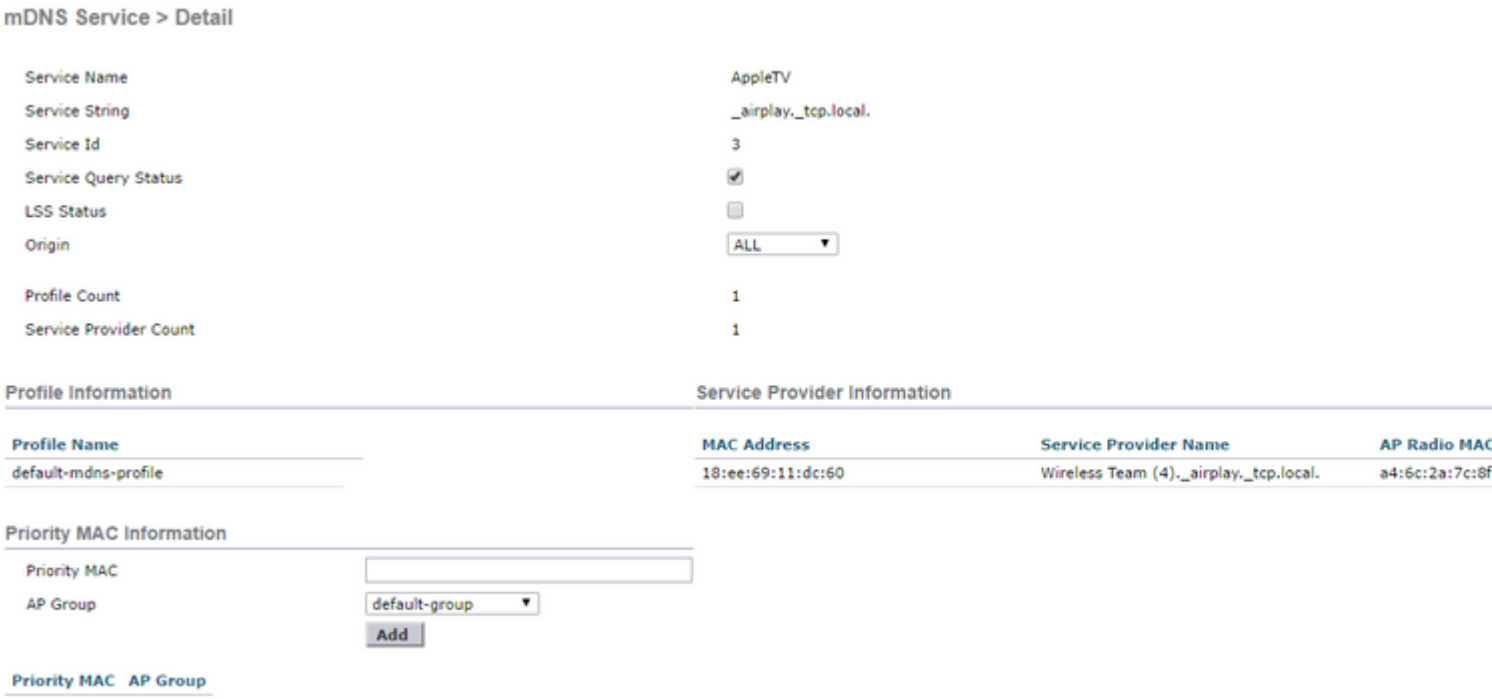
<#root>

```
show mdns domain-name-ip summary
```

```
Number of Domain Name-IP Entries................ 1
DomainName             MAC Address       IP Address       Vlan Id Type     TTL Time left (sec) (sec)
--------------------   ----------------  ---------------- ------- ------   ------ ------

Wireless-Team-3.local. 18:ee:69:11:dc:60 192.168.239.37   239     Wireless 4725   4163
```

Step 5. Verify whether the service provider is also listed under the specific service itself.

From GUI, navigate to **Controller** > **mDNS** > **General** > **Service Name** as shown in the image.



From CLI:

<#root>

```
show mdns service detailed AppleTV
```

```
Service Name.................................... AppleTV
Service String.................................. _airplay._tcp.local.
Service Id...................................... 3
Service query status............................ Enabled
Service LSS status.............................. Disabled
Service learn origin............................ Wireless and Wired
Number of Profiles.............................. 1
Profile......................................... default-mdns-profile
Number of Service Providers ..................... 1
Number of priority MAC addresses ................ 0

ServiceProvider MAC Address AP Radio MAC Vlan Id Type TTL Time left(sec) (sec)
------------------- ---------------- ---------------- ------- ------ ----- ---------
```

```
Wireless Team (4)._airplay._tcp.local. 18:EE:69:11:DC:60 A4:6C:2A:7C:8F:80 239 Wireless 4500 3841
```

Step 6. If the service is not discovered by the WLC, then check if it is to be learnt under bonjour browser (Controller>>mDNS>>mDNS browser). Bonjour browser is a cache of all the service advertisements seen at the WLC and not discovered because configuration did not allow to learn. You can select and add services from the Bonjour browser, this comes in handy when you test and implement a new service.

Step 7. These are the commands to debug Bonjour:

<#root>

```
debug mdns error enable

debug mdns message enable

debug mdns detail enable

debug mdns all enable
```

Bonjour browser and **show mdns service not-learnt** could be used as a debug tool as well.

Step 8. As mentioned before, If WLC and AP are in different subnets and AP multicast mode is set to multicast, then ensure that multicast routing is enabled on the wired network between the two vlans. In this setup, vlans are vlan 1 (WLC) and vlan 231 (AP).

```
Conf t
!
interface Vlan1
ip pim sparse-dense-mode

!
interface Vlan231
 ip pim sparse-dense-mode
!
```

Multicast routing at play:

<#root>

```
Gateway#sh ip mroute 239.100.100.100
IP Multicast Routing Table
```

```
-------snippet--------------------

(*, 239.100.100.100), 2w4d/stopped, RP 10.48.39.5, flags: SJC
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Vlan231, Forward/Sparse-Dense, 2w0d/00:02:10
    Vlan232, Forward/Sparse-Dense, 2w4d/00:02:11

(

10.48.39.142

, 239.100.100.100), 2w4d/00:02:50, flags: T

 Incoming interface: Vlan1

, RPF nbr 0.0.0.0, RPF-MFD
  Outgoing interface list:


Vlan231, Forward/Sparse-Dense, 2w0d/00:02:10, H
```

Along with these checklist, the key is to understand the packet flow when mDNS runs on WLC. The packet flow and the debugs help deep dive into areas where the previous

verification commands fall short.