Multicast Domain Name System

Introduction to mDNS Gateway

Multicast Domain Name System (mDNS) is an Apple service discovery protocol which locates devices and services on a local network with the use of mDNS service records.

The Bonjour protocol operates on service announcements and queries. Each query or advertisement is sent to the Bonjour multicast address ipv4 224.0.0.251 (ipv6 FF02::FB). This protocol 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. As, multicast DNS is limited to an L2 domain for a client to discover a service it has to be part of the same L2 domain, This is not always possible in any large scale deployment or enterprise.

In order to address this issue, the Cisco Catalyst 9800 Series Wireless Controller acts as a Bonjour Gateway. The controller then listens for Bonjour services, caches these Bonjour advertisements (AirPlay, AirPrint, and so on) from the source or host. For example, Apple TV responds back to Bonjour clients when asked or requested for a service. This way you can have sources and clients in different subnets.

By default, the mDNS gateway is disabled on the controller. To enable mDNS gateway functionality, you must explicitly configure mDNS gateway using CLI or Web UI.

Prerequisite

Since the Cisco Catalyst 9800 Series Wireless Controller will respond and advertise for services cached when acting as a Bonjour Gateway, it must have an SVI interface with a valid IP address on every VLAN where mDNS is allowed or used. This will be the source IP address of those mDNS packets that are coming out from the controller acting as mDNS Gateway.

Enabling mDNS Gateway (GUI)

Procedure

Step 1 Choose **Configuration** > **Services** > **mDNS**.

Step 2 In the **Global** section, toggle the slider to enable or disable the **mDNS Gateway**.

Step 3 From the **Transport** drop-down list, choose one of the following types:

- **ipv4**
- **ipv6**
- **both**

Step 4 Enter an appropriate timer value in **Active-Query Timer**. The valid range is between 15 to 120 minutes. The default is 30 minutes.

Step 5 From the **mDNS-AP Service Policy** drop-down list, choose an mDNS service policy.

Note Service policy is optional only if mDNS-AP is configured. If mDNS-AP is not configured, the system uses default-service-policy.

Step 6 Click **Apply**.

Enabling or Disabling mDNS Gateway (GUI)

Procedure

Step 1 Choose **Configuration** > **Services** > **mDNS** > **Global**.

Step 2 Enable or disable the **mDNS Gateway** toggle button.

Step 3 Choose **ipv4** or **ipv6** or **both** from the **Transport** drop-down list.

Step 4 Enter the **Active-Query Timer**.

Step 5 Click **Apply**.

Enabling or Disabling mDNS Gateway (CLI)

Note

- mDNS gateway is disabled by default globally on the controller.
- You need both global and WLAN configurations to enable mDNS gateway.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged **EXEC** mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**mdns-sd gateway** **Example:** `Device(config)# mdns-sd gateway`	Enables mDNS gateway.
Step 4	**transport** {**ipv4** | **ipv6** | **both**} **Example:** `Device(config-mdns-sd)# transport ipv4`	Processes mDNS message on a specific transport. Here, **ipv4** signifies that the IPv4 mDNS message processing is enabled. This is the default value.

	Command or Action	Purpose
		ipv6 signifies that the IPv6 mDNS message processing is enabled. **both** signifies that the IPv4 and IPv6 mDNS message is enabled for each network.
Step 5	**active-query timer** *active-query-periodicity* **Example:** `Device(config-mdns-sd)# active-query timer 15`	Changes the periodicity of mDNS multicast active query. **Note** An active query is a periodic mDNS query to refresh dynamic cache. Here, *active-query-periodicity* refers to the active query periodicity in Minutes. The valid range is from 15 to 120 minutes. Active query runs with a default periodicity of 30 minutes.
Step 6	**exit** **Example:** `Device(config-mdns-sd)# exit`	Returns to global configuration mode.

Creating Default Service Policy

When the mdns gateway is enabled on any of the WLANs by default, mdns-default-service-policy is associated with it. Default service policy consists of default-service-list and their details are explained in this section. You can override the default service policy with a custom service policy.

Procedure

Step 1 Create a service-definition if the service is not listed in the preconfigured services.

Step 2 Create a service list for IN and OUT by using the service-definitions.

Step 3 Use the existing service list to create a new service. For more information, refer to *Creating Service Policy* section.

Step 4 Attach the mdns-service-policy to the profile or VLAN that needs to be enforced.

Step 5 To check the default-mdns-service list, use the following command:

show mdns-sd default-service-list

Creating Custom Service Definition (GUI)

Procedure

Step 1 Choose **Configuration** > **Services** > **mDNS**.

Step 2 In the **Service Definition** section, click **Add**.

Step 3 In the **Quick Setup: Service Definition** page that is displayed, enter a name and description for the service definition.

Step 4 Enter a service type and click + to add the service type.

Step 5 Click **Apply to Device**.

Creating Custom Service Definition

Service definition is a construct that provides an admin friendly name to one or more mDNS service types or A pointer (PTR) Resource Record Name.

By default, few built-in service definitions are already predefined and available for admin to use.

In addition to built-in service definitions, admin can also define custom service definitions.

You can execute the following command to view the list of all the service definitions (built-in and custom):

```
Device# show mdns-sd master-service-list
```

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged **EXEC** mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**mdns-sd service-definition** *service-definition-name* **Example:** `Device(config)# mdns-sd service-definition CUSTOM1`	Configures mDNS service definition. **Note** • All the created custom service definitions are added to the primary service list. • Primary service list comprises of a list of custom and built-in service definitions.

	Command or Action	Purpose
Step 4	**service-type** *string* **Example:** `Device(config-mdns-ser-def)# service-type _custom1._tcp.local`	Configures mDNS service type.
Step 5	**exit** **Example:** `Device(config-mdns-ser-def)# exit`	Returns to global configuration mode.

Creating Service List (GUI)

Procedure

Step 1 Choose **Configuration** > **Services** > **mDNS**.

Step 2 In the **Service List** section, click **Add**.

Step 3 In the **Quick Setup: Service List** page that is displayed, enter a name for the service list.

Step 4 From the **Direction** drop-down list, choose **IN** for inbound filtering or **OUT** for outbound filtering.

Step 5 From the **Available Services** drop-down list, choose a service type to match the service list.

Note To allow all services, choose the **all** option.

Step 6 Click **Add Services**.

Step 7 From the **Message Type** drop-down list, choose the message type to match from the following options:

- **any**—To allow all messages.
- **announcement**—To allow only service advertisements or announcements for the device.
- **query**—To allow only a query from the client for a service in the network.

Step 8 Click **Save** to add services.

Step 9 Click **Apply to Device**.

Creating Service List

mDNS service list is a collection of service definitions.

Procedure

	Command or Action	Purpose
Step 1	**enable**	Enables privileged **EXEC** mode.

	Command or Action	Purpose
	Example: `Device> enable`	Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Step 3	**mdns-sd service-list** *service-list-name* {**IN** | **OUT**}
	Example:
	`Device(config)# mdns-sd service-list Basic-In IN`
	`Device(config)# mdns-sd service-list Basic-Out OUT`
		Configures mDNS service list.
		• **IN**: Provides inbound filtering.
		• **Out**: Provides outbound filtering.

Step 4	**match** *service-definition-name* **message-type** {**announcement** | **any** | **query**} **Example:** `Device(config-mdns-sl-in)# match CUSTOM1 message-type query`	Matches the service to the message type. Here, *service-definition-name* refers to the names of services, such as, airplay, airserver, airtunes, and so on. **Note** To add a service, the service name must be part of the primary service list. If the mDNS service list is set to IN, you get to view the following command: **match service-definition-name message-type {announcement | any | query}**. If the mDNS service list is set to Out, you get to view the following command: **match service-definition-name**.
Step 5	**show mdns-sd service-list {direction | name }**	Displays inbound or outbound direction list of the configured service-list to classify matching service-types for service-policy. The list can be filtered by name or specific direction.
Step 6	**exit** **Example:** `Device(config-mdns-sl-in)# exit`	Returns to global configuration mode.

Creating Service Policy (GUI)

Procedure

Step 1 Choose **Configuration** > **Services** > **mDNS**.

Step 2 In the **Service Policy** section, click **Add**.

Step 3 In the **Quick Setup: Service Policy** page that is displayed, enter a name for the service policy.

Step 4 From the **Service List Input** drop-down list, choose one of the types.

Step 5 From the **Service List Output** drop-down list, choose one of the types.

Step 6 From the **Location** drop-down list, choose the location you want to associate with the service list.

Step 7 Click **Apply to Device**.

Creating Service Policy

mDNS service policy is used for service filtering while learning services or responding to queries.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged **EXEC** mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**mdns-sd service-policy** *service-policy-name* **Example:** `Device(config)# mdns-sd service-policy mdns-policy1`	Enables mDNS service policy.
Step 4	**location** {**lss** | **site-tag**} **Example:**	Filters mDNS service types based on LSS or site-tag.

	Command or Action	Purpose
	`Device(config-mdns-ser-pol)# location lss`	**Note** In Location Specific Services (LSS) based filtering, the mDNS gateway responds with the service instances learnt from the neighboring APs of the querying client AP. Other service instances for the rest of APs are filtered. In Site tag based filtering, the mDNS gateway responds with the service instances that belong to the same site-tag as that of querying client. The mDNS gateway responds back with wired services even if the location based filtering is configured.
Step 5	**service-list** *service-list-name* {**IN** | **OUT**} **Example:** `Device(config-mdns-ser-pol)# service-list VLAN100-list IN`	Configures various service-list names for IN and OUT directions. **Note** If an administrator decides to create or use a custom service policy, then the custom service policy must be configured with service-lists for both directions (IN and OUT); otherwise, the mDNS Gateway will not work (will not learn services if there is no IN service-list, or will not reply or announce services learned if there is no OUT service-list).
Step 6	**exit** **Example:** `Device(config-mdns-ser-pol)# exit`	Returns to global configuration mode.

Configuring an mDNS Flex Profile (GUI)

Procedure

Step 1 Choose **Configuration** > **Services** > **mDNS**.

Step 2 In the **mDNS Flex Profile** section, click **Add**.
The **Add mDNS Flex Profile** window is displayed.

Step 3 In the **Profile Name** field, enter the flex mDNS profile name.

Step 4 In the **Service Cache Update Timer** field, specify the service cache update time. The default value is 1 minute. The valid range is from 1 to 100 minutes.

Step 5 In the **Statistics Update Timer** field, specify the statistics update timer. The default value is 1 minute. The valid range is from 1 to 100 minutes.

Step 6 In the **VLANs** field, specify the VLAN ID. You can enter multiple VLAN IDs separated by commas, or enter a range of VLAN IDs. Maximum number of VLANs allowed is 16.

Step 7 Click **Apply to Device**.

Configuring an mDNS Flex Profile (CLI)

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**mdns-sd flex-profile** *mdns-flex-profile-name* **Example:** `Device(config)# mdns-sd flex-profile mdns-flex-profile-name`	Enters the mDNS Flex Profile mode.
Step 3	**update-timer service-cache** *service-cache timer-value <1-100>* **Example:** `Device(config-mdns-flex-profile)# update-timer service-cache 60`	Configures the mDNS update service cache timer for the flex profile. The default value is 1 minute. Value range is between 1 minute and 100 minutes.
Step 4	**update-timer statistics** *statistics timer-value <1-100>* **Example:** `Device(config-mdns-flex-profile)# update-timer statistics 65`	Configures the mDNS update statistics timer for the flex profile. The default value is 1 minute. The valid range is from 1 to 100 minutes.
Step 5	**wired-vlan-range** *wired-vlan-range value* **Example:** `Device(config-mdns-flex-profile)# wired-vlan-range 10 - 20`	Configures the mDNS wired VLAN range for the flex profile. The default value is 1 minute. The valid range is from 1 minute to 100 minutes.

Applying an mDNS Flex Profile to a Wireless Flex Connect Profile (GUI)

Procedure

Step 1 Choose **Configuration** > **Tags & Profiles** > **Flex**.

Step 2 Click **Add**.
The **Add Flex Profile** window is displayed.

Step 3 Under the **General** tab, from the **mDNS Flex Profile** drop-down list, choose a flex profile name from the list.

Step 4 Click **Apply to Device**.

Applying an mDNS Flex Profile to a Wireless Flex Connect Profile (CLI)

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**wireless profile flex** *wireless-flex-profile-name* **Example:** `Device# wireless profile flex wireless-flex-profile-name`	Enters wireless flex profile configuration mode.
Step 3	**mdns-sd** *mdns-flex-profile* **Example:** `Device(config-wireless-flex-profile)# mdns-sd mdns-flex-profile-name`	Enables the mDNS features for all the APs in the profile

Associating mDNS Service Policy with Wireless Profile Policy (GUI)

Procedure

Step 1 Choose **Configuration** > **Tags & Profiles** > **Policy**.

Step 2 Click the **policy profile** name.

Step 3 In the **Advanced** tab, choose the mDNS service policy from the **mDNS Service Policy** drop-down list.

Step 4 Click **Update & Apply to Device**.

Associating mDNS Service Policy with Wireless Profile Policy

Note You must globally configure the mDNS service policy before associating it with the wireless profile policy.

A default mDNS service policy is already attached once the wireless profile policy is created. You can use the following commands to override the default mDNS service policy with any of your service policy:

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**wireless profile policy** *profile-policy* **Example:** `Device(config)# wireless profile policy default-policy-profile`	Configures wireless profile policy. Here, *profile-policy* refers to the name of the WLAN policy profile.
Step 3	**mdns-sd service-policy** *custom-mdns-service-policy* **Example:** `Device(config-wireless-policy)# mdns-sd service-policy custom-mdns-service-policy`	Associates an mDNS service policy with the wireless profile policy. The default mDNS service policy name is **default-mdns-service-policy**.

	Command or Action	Purpose
		Note The **default-mdns-profile-policy** uses **default-mdns-service-list** configuration for filtering mDNS service announcement and queries. In wireless network, the mDNS packets are consumed by the mDNS gateway and clients or device is deprived of learning this service. To share the service with the device and provide ease of configuration to the administrator, a list of few standard service types are shared by default on the wireless network. The list of such standard service types is termed as default service policy that comprises a set of service types. The table covers a sample service list in the default service policy. ***Table 1: Default Name and mDNS Service Type*** **Default Name** — **mDNS Service Type** Apple HomeSharing — _home-sharing_tcp.local Printer-IPPS — _ipps._tcp.local Google-chromecast — _googlecast_tcp.local **Note** • Location would be disabled on mDNS default service policy. • You cannot change the contents of the mDNS default service policy. However, you can create separate mDNS service policies and associate them under the wireless policy profile.

	Command or Action	Purpose
Step 4	**exit** **Example:** `Device(config-wireless-policy)# exit`	Returns to global configuration mode.

Enabling or Disabling mDNS Gateway for WLAN (GUI)

Procedure

Step 1 Choose **Configuration** > **Tags & Profiles** > **WLANs**.

Step 2 Click on the WLAN.

Step 3 In the **Advanced** tab, choose the mode in **mDNS Mode** drop-down list.

Step 4 Click **Update & Apply to Device**.

Enabling or Disabling mDNS Gateway for WLAN

Note Bridging is the default behaviour. This means that the mDNS packets are always bridged.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**wlan** *profile-name wlan-id ssid-name* **Example:** `Device(config)# wlan test 24 ssid1`	Specifies the WLAN name and ID. • *profile-name* is the WLAN name which can contain 32 alphanumeric characters • *wlan-id* is the wireless LAN identifier. The valid range is from 1 to 512. • *ssid-name* is the SSID which can contain 32 alphanumeric characters. **Note** Global configuration must be in place for mDNS gateway to work.

	Command or Action	Purpose

Step 3	**mdns-sd-interface** {**gateway** | **drop**}
	Example:
	`Device(config-wlan)# mdns-sd gateway`
	`Device(config-wlan)# mdns-sd drop`
		Enables or disables mDNS gateway and bridge functions on WLAN.

Step 4	**exit** **Example:** `Device(config-wlan)# exit`	Returns to global configuration mode.
Step 5	**show wlan name** *wlan-name* | **show wlan all** **Example:** `Device# show wlan name test | show wlan all`	Verifies the status of mDNS on WLAN.
Step 6	**show wireless profile policy** **Example:** `Device# show wireless profile policy`	Verifies the service policy configured in WLAN.

mDNS Gateway with Guest Anchor Support and mDNS Bridging

When mDNS Gateway is enabled on both Anchor and Foreign controller, the mDNS gateway functionality is supported in guest anchor deployment where clients on guest LAN or WLAN with guest anchor enabled will be responded with any services or cache from export foreign controller itself. All advertisements received on Guest LAN or WLAN on export foreign are learnt on the export foreign itself. All queries received on guest LAN or WLAN are responded by the export foreign itself.

When mDNS Gateway is enabled on Anchor and Disabled on Foreign controller [Bridging Mode], the mDNS gateway functionality is supported in guest anchor deployment where clients on guest LAN or WLAN with guest anchor enabled will be responded with any services or cache from export Anchor even though the clients are connected on Foreign. All advertisements received on guest LAN or WLAN on export foreign is forwarded to Anchor and the cache is stored on the Anchor itself. All queries received on guest LAN or WLAN are responded by the export Anchor itself.

Note

- You must configure the guest-LAN to a wireless profile policy which is configured with the required mDNS service-policy.
- To configure non guest LAN mDNS gateway, see the mDNS Gateway chapter.

Configuring mDNS Gateway on Guest Anchor

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**guest-wlan profile-name** *guest-lan-profile-name guest-wlan-id* **Example:** `Device(config)# guest-wlan profile-name g-lanpro 2`	Configures the guest WLAN profile with a wired VLAN.
Step 3	**mdns-sd gateway** **Example:** `Device(config-guest-wlan)# mdns-sd gateway`	Enables mDNS gateway on the guest WLAN.

Configuring mDNS Gateway on Guest Foreign (Guest WLAN)

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**guest-wlan profile-name** *guest-lan-profile-name guest-wlan-id* **wired-vlan** *vlan-id* **Example:** `Device(config)# guest-wlan profile-name g-lanpro 2 wired-vlan 230`	Configures guest WLAN profile with a wired VLAN. **Note** Configure the wired VLAN only for the Guest Foreign controller.
Step 3	**mdns-sd gateway** **Example:** `Device(config-guest-wlan)# mdns-sd gateway`	Enables mDNS gateway on the guest WLAN.
Step 4	**exit** **Example:**	Returns to global configuration mode.

	Command or Action	Purpose
	`Device(config-wireless-policy)# exit`	

Verifying mDNS Gateway Configurations

To verify the mDNS summary, use the following command:

```
Device# show mdns-sd summary
mDNS Gateway: Enabled
Active Query: Enabled
  Periodicity (in minutes): 30
Transport Type: IPv4
```

To verify the mDNS cache, use the following command:

```
Device# show mdns-sd cache
------------------------------------------------------------ PTR Records
----------------------------------------
RECORD-NAME                      TTL      WLAN   CLIENT-MAC       RR-RECORD-DATA

-----------------------------------------------------------------------------------------------------
_airplay._tcp.local              4500     30     07c5.a4f2.dc01   CUST1._airplay._tcp.local

_ipp._tcp.local                  4500     30     04c5.a4f2.dc01   CUST3._ipp._tcp.local2

_ipp._tcp.local                  4500     15     04c5.a4f2.dc01   CUST3._ipp._tcp.local4

_ipp._tcp.local                  4500     10     04c5.a4f2.dc01   CUST3._ipp._tcp.local6

_veer_custom._tcp.local          4500     10     05c5.a4f2.dc01
CUST2._veer_custom._tcp.local8
```

To verify the mDNS cache from wired service provider, use the following command:

```
Device# show mdns-sd cache wired
------------------------------------------------------------ PTR Records
----------------------------------------
RECORD-NAME                                TTL      VLAN      CLIENT-MAC       RR-RECORD-DATA

-----------------------------------------------------------------------------------------------------
_airplay._tcp.local                          4500     16        0866.98ec.97af
wiredapple._airplay._tcp.local
_raop._tcp.local                             4500     16        0866.98ec.97af
086698EC97AF@wiredapple._raop._tcp.local

------------------------------------------------------------ SRV Records
----------------------------------------
RECORD-NAME                                TTL      VLAN      CLIENT-MAC       RR-RECORD-DATA

-----------------------------------------------------------------------------------------------------
wiredapple._airplay._tcp.local               4500     16        0866.98ec.97af   0 0 7000
wiredapple.local
086698EC97AF@wiredapple._raop._tcp.local     4500     16        0866.98ec.97af   0 0 7000
wiredapple.local

------------------------------------------------------------ A/AAAA Records
----------------------------------------
RECORD-NAME                                TTL      VLAN      CLIENT-MAC       RR-RECORD-DATA

-----------------------------------------------------------------------------------------------------
```

```
wiredapple.local                          4500     16        0866.98ec.97af
2001:8:16:16:e5:c446:3218:7437

------------------------------------------------------------ TXT Records
-----------------------------------------
RECORD-NAME                              TTL      VLAN      CLIENT-MAC       RR-RECORD-DATA

-----------------------------------------------------------------------------------------------------------
wiredapple._airplay._tcp.local            4500     16        0866.98ec.97af
[343]'acl=0''deviceid=08:66:98:EC:97:AF''features=
086698EC97AF@wiredapple._raop._tcp.local   4500     16        0866.98ec.97af
[193]'cn=0,1,2,3''da=true''et=0,3,5''ft=0x5A7FFFF7
```

To verify the mdns-sd type PTR, use the following command:

```
Device# show mdns-sd cache type {PTR | SRV | A-AAA | TXT}
RECORD-NAME                                   TTL      WLAN      CLIENT-MAC
RR-Record-Data
-------------------------------------------------------------------------------------------------------
_custom1._tcp.local                           4500     2         c869.cda8.77d6
service_t1._custom1._tcp.local
_custom1._tcp.local                           4500     2         c869.cda8.77d6
vk11._custom1._tcp.local
_ipp._tcp.local                               4500     2         c869.cda8.77d6
service-4._ipp._tcp.local
```

To verify the mdns-sd cache for a client MAC, use the following command:

```
Device# show mdns-sd cache {ap-mac <ap-mac> | client-mac <client-mac> | wlan-id <wlan-id>
| wired}
RECORD-NAME                                   TTL      WLAN      CLIENT-MAC
RR-Record-Data
-------------------------------------------------------------------------------------------------------
_custom1._tcp.local                           4500     2         c869.cda8.77d6
service_t1._custom1._tcp.local
_custom1._tcp.local                           4500     2         c869.cda8.77d6
vk11._custom1._tcp.local
_ipp._tcp.local                               4500     2         c869.cda8.77d6
service-4._ipp._tcp.local

------------------------------------------------------------ SRV Records
-------------------------------------------------------------
RECORD-NAME                                   TTL      WLAN      CLIENT-MAC
RR-Record-Data
-------------------------------------------------------------------------------------------------------
service-4._ipp._tcp.local                    4500     2         c869.cda8.77d6   0 0 1212
 mDNS-Client1s-275.local
vk11._custom1._tcp.local                      4500     2         c869.cda8.77d6   0 0 987
 mDNS-Client1s-275.local
service_t1._custom1._tcp.local                4500     2         c869.cda8.77d6   0 0 197
 mDNS-Client1s-275.local

------------------------------------------------------------ A/AAAA Records
-------------------------------------------------------------
RECORD-NAME                                   TTL      WLAN      CLIENT-MAC
RR-Record-Data
-------------------------------------------------------------------------------------------------------
mDNS-Client1s-275.local                      4500     2         c869.cda8.77d6   120.1.1.33


------------------------------------------------------------ TXT Records
-------------------------------------------------------------
RECORD-NAME                                   TTL      WLAN      CLIENT-MAC
RR-Record-Data
-------------------------------------------------------------------------------------------------------
```

```
service-4._ipp._tcp.local                      4500     2         c869.cda8.77d6   'CLient1'

vk11._custom1._tcp.local                        4500     2          c869.cda8.77d6
'txtvers=11'
service_t1._custom1._tcp.local                  4500     2          c869.cda8.77d6
'txtvers=12'
```

To verify the mdns-sd cache in detail, use the following command:

```
Device# show mdns-sd cache detail

Name: _custom1._tcp.local
  Type: PTR
  TTL: 4500
  WLAN: 2
  WLAN Name: mdns120
  VLAN: 120
  Client MAC: c869.cda8.77d6
  AP Ethernet MAC: 7069.5ab8.33d0
  Expiry-Time: 09/09/18 21:50:47
  Site-Tag: default-site-tag
  Rdata: service_t1._custom1._tcp.local
```

To verify the mdns-sd statistics, use the following command:

```
Device# show mdns-sd statistics

-------------------------------------------------------
Consolidated mDNS Packet Statistics
-------------------------------------------------------
mDNS stats last reset time: 03/11/19 04:17:35
mDNS packets sent: 61045
  IPv4 sent: 30790
    IPv4 advertisements sent: 234
    IPv4 queries sent: 30556
  IPv6 sent: 30255
    IPv6 advertisements sent: 17
    IPv6 queries sent: 30238
  Multicast sent: 57558
    IPv4 sent: 28938
    IPv6 sent: 28620
mDNS packets received: 72796
  advertisements received: 13604
  queries received: 59192
  IPv4 received: 40600
    IPv4 advertisements received: 6542
    IPv4 queries received: 34058
  IPv6 received: 32196
    IPv6 advertisements received: 7062
    IPv6 queries received: 25134
mDNS packets dropped: 87

-------------------------------------------------------
Wired mDNS Packet Statistics
-------------------------------------------------------
mDNS stats last reset time: 03/11/19 04:17:35
mDNS packets sent: 61033
  IPv4 sent: 30778
    IPv4 advertisements sent: 222
    IPv4 queries sent: 30556
  IPv6 sent: 30255
    IPv6 advertisements sent: 17
    IPv6 queries sent: 30238
  Multicast sent: 57558
    IPv4 sent: 28938
```

```
    IPv6 sent: 28620
mDNS packets received: 52623
  advertisements received: 1247
  queries received: 51376
  IPv4 received: 32276
    IPv4 advertisements received: 727
    IPv4 queries received: 31549
  IPv6 received: 20347
    IPv6 advertisements received: 520
    IPv6 queries received: 19827
mDNS packets dropped: 63

-----------------------------------------------------
mDNS Packet Statistics, for WLAN: 2
-----------------------------------------------------
mDNS stats last reset time: 03/11/19 04:17:35
mDNS packets sent: 12
  IPv4 sent: 12
    IPv4 advertisements sent: 12
    IPv4 queries sent: 0
  IPv6 sent: 0
    IPv6 advertisements sent: 0
    IPv6 queries sent: 0
  Multicast sent: 0
    IPv4 sent: 0
    IPv6 sent: 0
mDNS packets received: 20173
  advertisements received: 12357
  queries received: 7816
  IPv4 received: 8324
    IPv4 advertisements received: 5815
    IPv4 queries received: 2509
  IPv6 received: 11849
    IPv6 advertisements received: 6542
    IPv6 queries received: 5307
mDNS packets dropped: 24
```

To verify the default service list details, use the following command:

```
Device# show mdns-sd default-service-list

-------------------------------------------
        mDNS Default Service List
-------------------------------------------

Service Definition: airplay
Service Names: _airplay._tcp.local

Service Definition: airtunes
Service Names: _raop._tcp.local

Service Definition: homesharing
Service Names: _home-sharing._tcp.local

Service Definition: printer-ipp
Service Names: _ipp._tcp.local

Service Definition: printer-lpd
Service Names: _printer._tcp.local

Service Definition: printer-ipps
Service Names: _ipps._tcp.local

Service Definition: printer-socket
Service Names: _pdl-datastream._tcp.local
```

```
Service Definition: google-chromecast
Service Names: _googlecast._tcp.local

Service Definition: itune-wireless-devicesharing2
Service Names: _apple-mobdev2._tcp.local
```

To verify the primary service list details, use the following command:

```
Device# show mdns-sd master-service-list

-------------------------------------------
        mDNS Master Service List
-------------------------------------------

Service Definition: fax
Service Names: _fax-ipp._tcp.local

Service Definition: roku
Service Names: _rsp._tcp.local

Service Definition: airplay
Service Names: _airplay._tcp.local

Service Definition: scanner
Service Names: _scanner._tcp.local

Service Definition: spotify
Service Names: _spotify-connect._tcp.local

Service Definition: airtunes
Service Names: _raop._tcp.local

Service Definition: airserver
Service Names: _airplay._tcp.local
               _airserver._tcp.local

.
.
.

Service Definition: itune-wireless-devicesharing2
Service Names: _apple-mobdev2._tcp.local
```

To verify the mDNS-AP configured on the controller and VLAN(s) associated with it, use the following command:

```
Device# show mdns-sd ap

Number of mDNS APs................................ 1

AP Name   Ethernet MAC   Number of Vlans   Vlanidentifiers
-------------------------------------------------------------------------------------------------
AP3600-1   7069.5ab8.33d0          1      300
```

Further Debug

To debug mDNS further, use the following procedure:

1. Run this command at the controller:

   ```
   set platform software trace wncd <0-7> chassis active R0 mdns debug
   ```

2. Reproduce the issue.

3. Run this command to gather the traces enabled:

```
show wireless loadbalance ap affinity wncd 0

AP MAC   Discovery Timestamp  Join Timestamp                Tag    Vlanidentifiers
-------------------------------------------------------------------------------------
0cd0.f894.0600      06/30/21 12:39:48    06/30/21 12:40:021 default-site-tag     300
```