

Configure FlexConnect mDNS in 9800 Wireless LAN Controller

Contents

[Introduction](#)

[Prerequisites](#)

[Requirements](#)

[Components Used](#)

[Background Information](#)

[Network Diagram](#)

[Configurations](#)

[Enable mDNS Globally in the WLC](#)

[Select mDNS Services within a Service List](#)

[Merge the Service Lists with a Service Policy](#)

[Configure an mDNS Flex Profile](#)

[Configure the WLAN with mDNS Gateway Mode](#)

[Apply the mDNS Service Policy to the WLAN-Policy](#)

[Configure mDNS Flex Profile in FlexConnect Policy](#)

[Verify](#)

[WLC Show Commands](#)

[AP Show Commands](#)

[Troubleshoot](#)

[Wired Service Provider](#)

[Wireless Service Provider](#)

[Primary mDNS AP](#)

[Services Not Allowed per mDNS Service List](#)

[Custom mDNS Services Configuration in Service Definition](#)

[FlexConnect mDNS Bridging Mode](#)

[FlexConnect mDNS Drop Mode](#)

Introduction

This document describes how to configure FlexConnect Multicast Domain Name System (mDNS) Gateway in 9800 Wireless LAN Controller.

Prerequisites

Requirements

Cisco recommends you have knowledge of these topics:

- 9800 Wireless LAN Controller mDNS concepts

- FlexConnect Local Switching concepts

Components Used

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

- Catalyst 9800 Wireless Controller Series (Catalyst 9800-L), Cisco IOS® XE Cupertino 17.9.5
- Integrated Services Routers (ISR), Cisco IOS® XE Gibraltar 17.6.5
- Catalyst 3560 Series Switch, Cisco IOS® 15.2.4E10
- Access Point 9117AXI-B, Access Point 9130AXI-B

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

Multicast Domain Name System (mDNS) is a protocol that provides flexibility to discover and share services between Service Providers (SP) and Service Users (wireless clients). Service Providers are devices that provide a service such as printers, smart TV, file sharing services and more that Service Users can utilize.

The mDNS protocol is based on UDP, utilizes port 5353, Mac Address 01:00:5E:00:00:FB and IP Address 224.0.0.251 for IPv4 and FF02::FB for IPv6.

There are two modes mDNS works in the WLC: Bridging and Gateway. Bridging mode works only in the same Vlan (layer two) where the Service Provider and Service User must be in the same subnet. Gateway mode works with the Service Provider and Service User in the same or different Vlans, with the WLC or the AP doing Bonjour Gateway to cache the services from the Service Provider and share it with Service Users.

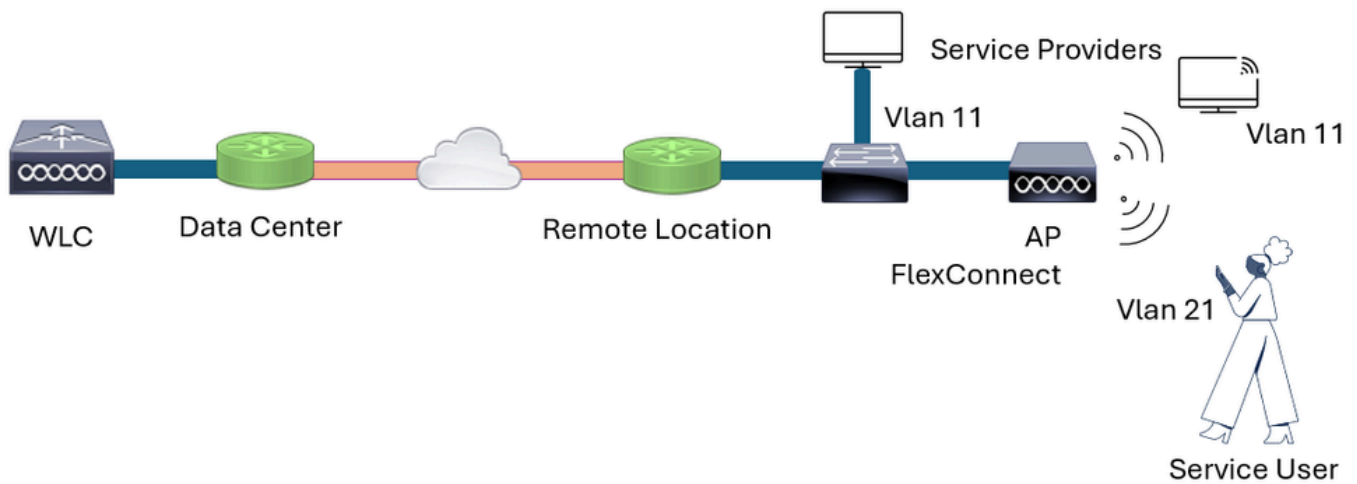
This document is based in mDNS FlexConnect Local Switching only, which in this case the AP acts as the mDNS Gateway to cache the services advertised by the Service Providers and shares these services with the Service Users.



Note: For Central Switching mDNS configuration, please refer to [Understand mDNS on Catalyst 9800 Wireless Controller](#)

Network Diagram

Wireless and wired Service Provider advertise mDNS services in a FlexConnect Local Switching environment, along with a Wireless Client (Service User) that utilizes the mDNS services.



Network Diagram

Configurations

Enable mDNS Globally in the WLC

For the AP to work as mDNS Gateway, the feature needs to be turned on by enabling mDNS Gateway globally.

WLC GUI

Navigate to **Configuration > Services > mDNS > Global > mDNS Gateway** as shown in the image.

Configuration ▾ > Services ▾ > mDNS

Global

Service Policy

mDNS Flex Profile

mDNS Gateway

ENABLED ☒

Transport

ipv4 ▾

Active-Query Timer (Minutes) *

30

mDNS-AP Service Policy

default-mdns-s...

Clear

mDNS Global Configuration

WLC CLI

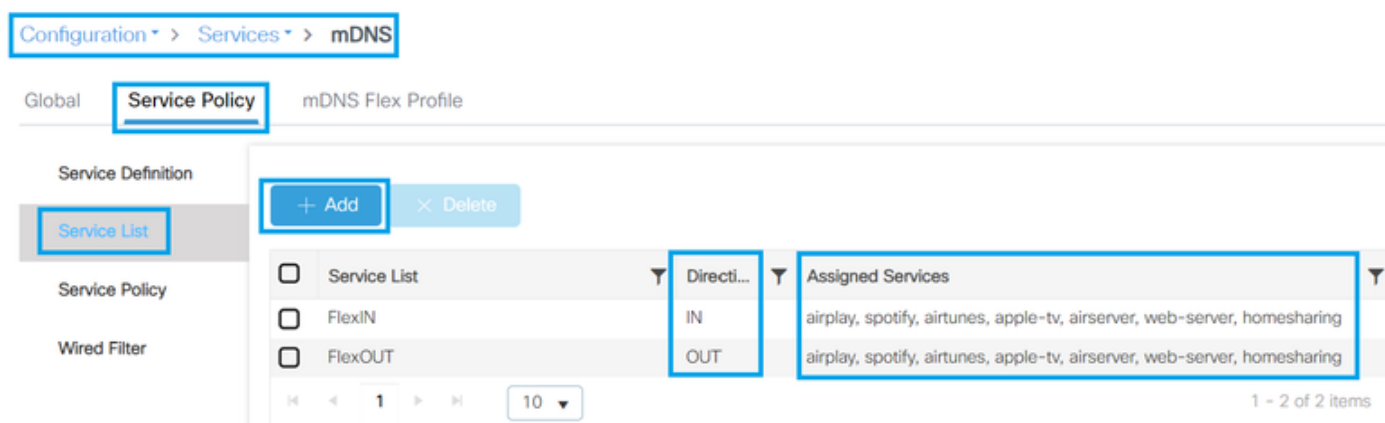
```
WLC#
WLC#conf t
WLC(config)#mdns-sd gateway
WLC(config-mdns-sd)#end
WLC#
```

Select mDNS Services within a Service List

Configure a Service List to allow the mDNS services of preference. The list has to be configured in two directions which are IN and OUT, which filters what ingress and egress services are allowed by the Access Point acting as mDNS gateway.

WLC GUI

Navigate to **Configuration > Services > mDNS > Service Policy > Service List** as shown in the image.



Select the Services Needed in the Service List

WLC CLI

```
WLC#
WLC#conf t
WLC(config)#mdns-sd service-list FlexIN IN
WLC(config-mdns-sl-in)#match airplay
WLC(config-mdns-sl-in)#match spotify
WLC(config-mdns-sl-in)#exit

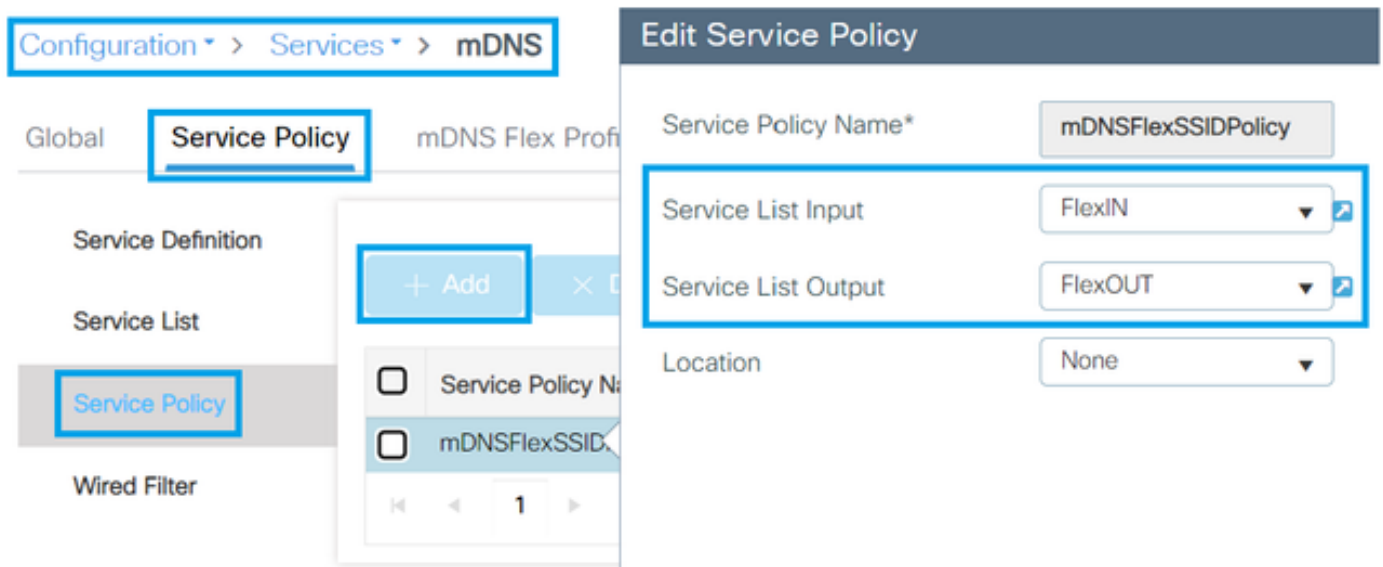
WLC(config)#mdns-sd service-list FlexOUT OUT
WLC(config-mdns-sl-out)#match airplay
WLC(config-mdns-sl-out)#match spotify
WLC(config-mdns-sl-out)#end
WLC#
```

Merge the Service Lists with a Service Policy

Once the Service List IN and OUT are configured with the needed services a Service Policy is used to merge them. Once merged this Service Policy can be used in the WLAN-Policy, FlexConnect profile and mDNS Flex Policy.

WLC GUI

Navigate to **Configuration > Services > mDNS > Service Policy > Service Policy** as shown in the image.



Merge the mDNS Service Lists in an mDNS Policy

WLC CLI

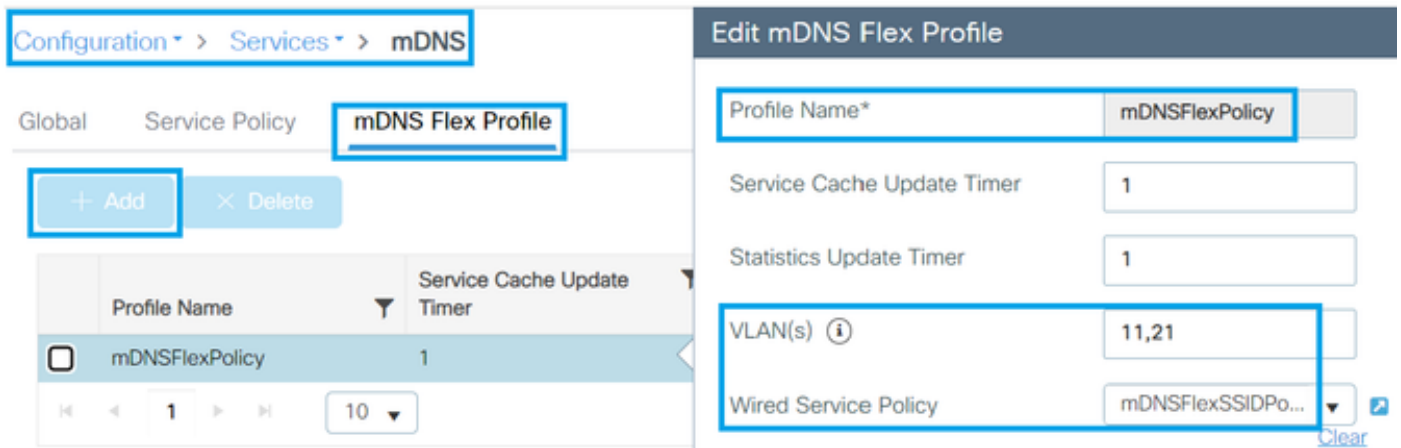
```
WLC#
WLC#conf t
WLC(config)#mdns-sd service-policy mDNSFlexSSIDPolicy
WLC(config-mdns-ser-pol)#service-list FlexIN IN
WLC(config-mdns-ser-pol)#service-list FlexOUT OUT
WLC(config-mdns-ser-pol)#end
WLC#
```

Configure an mDNS Flex Profile

In the mDNS Flex Profile, the FlexConnect Local Switching Vlans where mDNS is used need to be added to the Flex Profile, the Vlan of the Service Provider and Service User must be added to the mDNS Flex Profile, along with the mDNS Service Policy which allows to filter the services via wired.

WLC GUI

Navigate to **Configuration > Services > mDNS > mDNS Flex Profile** as shown in the image.



Create an mDNS FlexConnect Profile

WLC CLI

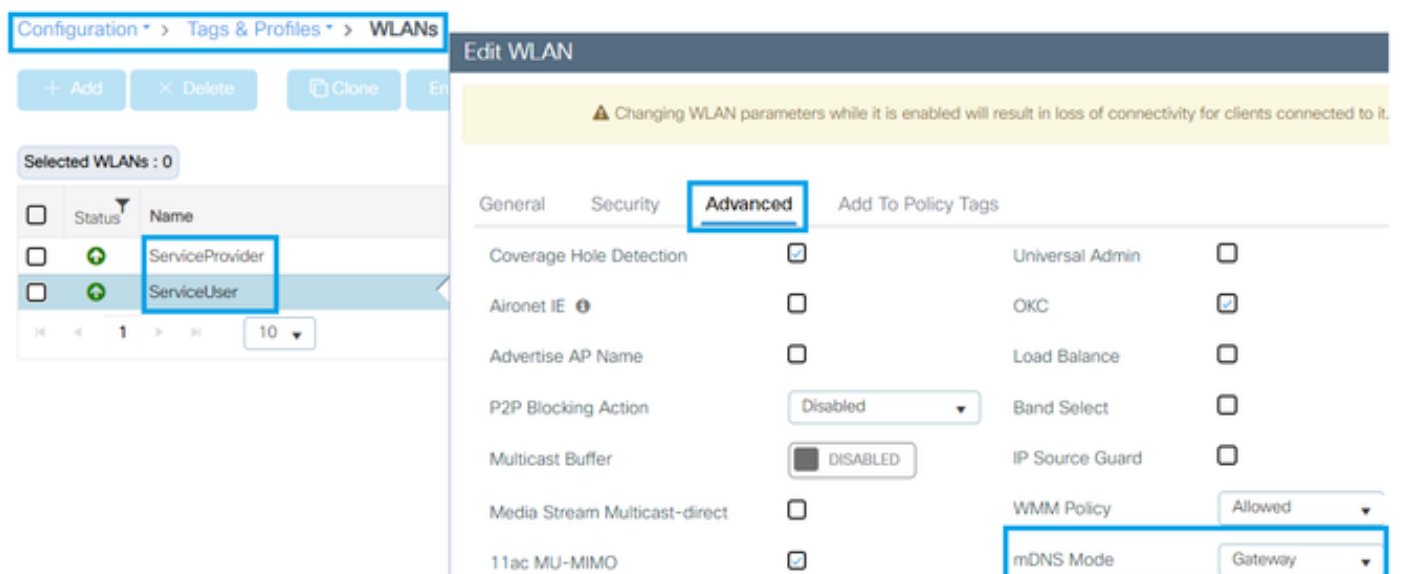
```
WLC#
WLC#conf t
WLC(config)#mdns-sd flex-profile mDNSFlexPolicy
WLC(config-mdns-flex-prof)#wired-vlan-range 11,21
WLC(config-mdns-flex-prof)#wired-service-policy mDNSFlexSSIDPolicy
WLC(config-mdns-flex-prof)#end
WLC#
```

Configure the WLAN with mDNS Gateway Mode

Every WLAN has by default the mDNS mode as Bridging. For the AP to know when to act as an mDNS Gateway for Service Providers connected via wireless and for Service Users the WLAN must be configured with mDNS as Gateway mode.

WLC GUI

Navigate to **Configuration > Tags & Profiles > WLANs** as shown in the image.



WLC CLI

```
WLC#  
WLC#conf t  
WLC(config)#wlan ServiceUser  
WLC(config-wlan)#mdns-sd-interface gateway  
WLC(config-wlan)#end  
WLC#
```



Warning: Configuration changes in the WLAN provoke connected wireless clients to drop from the SSID. Please be cautious with any configuration change in the WLANs during production time.

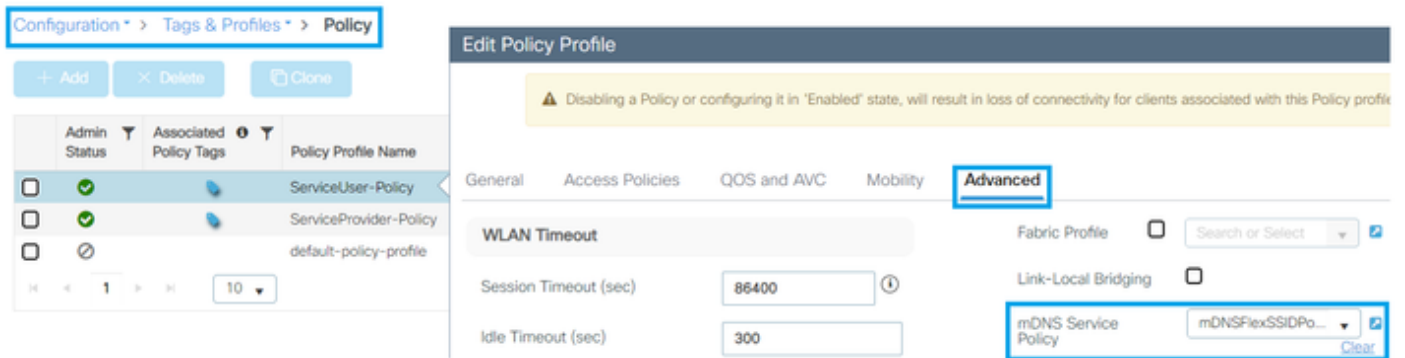
Apply the mDNS Service Policy to the WLAN-Policy

For wireless Service Providers and wireless User Providers, the mDNS services are filtered with the

mDNS Policy previously configured once it is applied to the WLAN-Policy of the WLANs.

WLC GUI

Navigate to **Configuration > Tags & Profiles > Policy** as shown in the image.



Assign the mDNS Policy

WLC CLI

```
WLC#
WLC#conf t
WLC(config)#wireless profile policy ServiceUser-Policy
WLC(config-wireless-policy)#mdns-sd service-policy mDNSFlexSSIDPolicy
WLC(config-wireless-policy)#end
WLC#
```



Warning: Configuration changes in the WLAN-Policy provoke connected wireless clients to drop from the WLAN. Please be cautious with any configuration in the WLAN-Policy during production time.



Note: For general FlexConnect configuration, please refer to [Understand FlexConnect on Catalyst 9800 Wireless Controller](#)

Configure mDNS Flex Profile in FlexConnect Policy

In the FlexConnect Policy, where configuration like Vlans, ACLs and more are applied, the mDNS Flex Profile needs to be selected to apply it to the APs that belong to the FlexConnect Policy.

WLC GUI

Navigate to **Configuration > Tags & Profiles > Flex** as shown in the image.

Configuration * > Tags & Profiles * > Flex

+ Add - Delete Clone

Flex Profile Name

- ☐ mDNSFlexPolicy
- ☐ default-flex-profile

1 10

Edit Flex Profile

General Local Authentication Policy ACL VLAN DNS Layer Security

Name* mDNSFlexPolicy

Description Enter Description

Native VLAN ID 1

HTTP Proxy Port 0

HTTP-Proxy IP Address 0.0.0.0

CTS Policy

Inline Tagging ☐

Fallback Radio Shut ☐

Flex Resilient ☐

ARP Caching ☐

Efficient Image Upgrade ☐

OfficeExtend AP ☐

Join Minimum Latency ☐

IP Overlap ☐

mDNS Flex Profile mDNSFlexPolicy

Assign the mDNS Flex Profile

WLC CLI

```
WLC#
WLC#conf t
WLC(config)#wireless profile flex mDNSFlexPolicy
WLC(config-wireless-flex-profile)#mdns-sd profile mDNSFlexPolicy
WLC(config-wireless-flex-profile)#end
WLC#
```

Verify

From the WLC and AP, the configuration can be checked with these commands.

WLC Show Commands

Example of general FlexConnect mDNS configuration can be checked with these commands:

```
WLC#show run | sec mdns-sd
mdns-sd gateway
mdns-sd service-list FlexIN IN
  match airplay
  match spotify
  match airtunes
  match apple-tv
  match airserver
  match web-server
  match homesharing
mdns-sd service-list FlexOUT OUT
  match airplay
  match spotify
  match airtunes
  match apple-tv
  match airserver
  match web-server
  match homesharing
mdns-sd service-policy mDNSFlexSSIDPolicy
  service-list FlexIN IN
```

```
service-list FlexOUT OUT
mdns-sd flex-profile mDNSFlexPolicy
wired-vlan-range 11,21
wired-service-policy mDNSFlexSSIDPolicy
mdns-sd profile mDNSFlexPolicy
```

WLAN mDNS mode can be checked with this command:

```
WLC#show wlan name ServiceUser | in mDNS
  mDNS Gateway Status : Gateway
WLC#show wlan name ServiceProvider | in mDNS
  mDNS Gateway Status : Gateway
```

WLAN-Policy mDNS configuration can be checked with this command:

```
WLC#show wireless profile policy detailed ServiceUser-Policy | in mDNS
  mDNS Service Policy name : mDNSFlexSSIDPolicy
WLC#show wireless profile policy detailed ServiceProvider-Policy | in mDNS
  mDNS Service Policy name : mDNSFlexSSIDPolicy
```

AP Show Commands

Configuration related to mDNS can be checked from the AP side with these commands:

```
9130mDNSAP#show mdns profile detail
FlexIN_IN _home-sharing._tcp.local ANY
FlexIN_IN _airplay._tcp.local ANY
FlexIN_IN _airserver._tcp.local ANY
FlexIN_IN _raop._tcp.local ANY
FlexIN_IN _spotify-connect._tcp.local ANY
FlexIN_IN _http._tcp.local ANY
FlexOUT_OUT _home-sharing._tcp.local ANY
FlexOUT_OUT _airplay._tcp.local ANY
FlexOUT_OUT _airserver._tcp.local ANY
FlexOUT_OUT _raop._tcp.local ANY
FlexOUT_OUT _spotify-connect._tcp.local ANY
FlexOUT_OUT _http._tcp.local ANY
```

```
9130mDNSAP#show mdns status
Global mDNS gateway:Enabled
vap_id ssid mdns_mode
0 ServiceUser Gateway
1 ServiceProvider Gateway
Active query interval:30
vap service_list_in service_list_out location
0 FlexIN_IN FlexOUT_OUT 0
```

```
1 FlexIN_IN FlexOUT_OUT 0
Wired vlan configuration: 11 21
mdns stats timer: 1
mdns cache timer: 1
AP Sync VLAN: 10
Wired service list IN: FlexIN_IN
Wired service list OUT: FlexOUT_OUT
```

```
9130mDNSAP#show mdns ap-table
AP_ETH_MAC Last_message_time Msg_seq Is_primary_ap
3C:57:31:55:E4:28 1721178339 133 YES
0C:D0:F8:98:1B:F0 1721178339 133 NO
```

Troubleshoot

For the purposes of troubleshooting, this document explains the workflow mDNS goes through in FlexConnect Local Switching. It is important to remember the WLC does not have any role in how mDNS is managed due to the deployment mode which is FlexConnect Local Switching.

The AP itself is the mDNS Gateway device. The AP learns the services from the Service Providers and shares the services with the Services User. This while the AP, Service Provider, and Service User are placed in different Vlans.

Per Network Diagram section:

- Service Provider is in Vlan 11
- Service User is in Vlan 21
- Access Point is in Vlan 10

Wired Service Provider

The Service Provider, once it detects there is connectivity to the network, uses a mechanism called probe. It sends an mDNS query to make sure if there is any other network device that offers the same mDNS services or not. After the probe, the Wired Service provider uses an announce mechanism, it sends an mDNS type response to announce the services it supports.

Next, a packet capture taken from the mDNS Gateway AP switchport which shows the Service Provider, announces the services it supports. The packet is sourced with the MAC Address and IP Address of the Service Provider in Vlan 11 and it has a destination of the MAC Address and IP Address of mDNS, including the mDNS port 5353 over UDP, it also contains the answers which are the services supported by the Service Provider.

The answers section in next image shows the services of our interest which are airplay and spotify, later the AP cache these services and save it them in the database.

No.	Time	Source	Destination	Protocol	Length	Info
322	00:06:04.535225	192.168.11.54	224.0.0.251	MDNS	848	Standard query response 0x0000 TXT, cach
> Frame 322: 848 bytes on wire (6784 bits), 848 bytes captured (6784 bits) on interface \Device\NPF {CC2095A1-094C-440E-A8E1-...}						
> Ethernet II, Src: SamsungElect_45:8e:26 (e0:03:6b:45:8e:26), Dst: IPv4mcast_fb (01:00:5e:00:00:fb)						
> Internet Protocol Version 4, Src: 192.168.11.54, Dst: 224.0.0.251						
> User Datagram Protocol, Src Port: 5353, Dst Port: 5353						
> Multicast Domain Name System (response)						
> Transaction ID: 0x0000						
> [Expert Info (Warning/Protocol): DNS response retransmission. Original response in frame 320]						
> Flags: 0x8400 Standard query response, No error						
Questions: 0						
Answer RRs: 10						
Authority RRs: 0						
Additional RRs: 3						
> Answers						
> ed9583d2b239afa30d7b0e7106c3710ddcfe5769._spotify-connect._tcp.local: type TXT, class IN, cache flush						
> services.dns-sd._udp.local: type PTR, class IN, spotify-connect._tcp.local						
> _spotify-connect._tcp.local: type PTR, class IN, ed9583d2b239afa30d7b0e7106c3710ddcfe5769._spotify-connect._tcp.local						
> ed9583d2b239afa30d7b0e7106c3710ddcfe5769._spotify-connect._tcp.local: type SRV, class IN, cache flush, priority 0, we						
> Samsung CU7000 55 TV._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, Samsung CU7000 55 TV._airplay._tcp.local						
> Samsung CU7000 55 TV._airplay._tcp.local: type SRV, class IN, cache flush, priority 0, weight 0, port 7000, target lo						
> localhost.local: type A, class IN, cache flush, addr 192.168.11.54						
> localhost.local: type AAAA, class IN, cache flush, addr fe80::e203:6bff:fe45:8e26						

mDNS Service Provider Services

From the AP CLI, the wired Service Provider announcements can be seen as well, to see any mDNS information from the AP itself these debugs have to be enabled:

- AP#debug mdns events
- AP#debug mdns packets

```

Jul 17 23:51:32 kernel: [*07/17/2024 23:51:32.0403] chatter: MDNSGW-EVENT: flex mdns gw: Recieved wired
Jul 17 23:51:32 kernel: [*07/17/2024 23:51:32.0403] chatter: MDNSGW-EVENT: push: adding ptr record to c
Jul 17 23:51:32 kernel: [*07/17/2024 23:51:32.0404] chatter: MDNSGW-EVENT: mdns_ptr_db:updated TXT reco
Jul 17 23:51:32 kernel: [*07/17/2024 23:51:32.0404] chatter: MDNSGW-EVENT: mdns_ptr_db:added/updated PT
Jul 17 23:51:32 kernel: [*07/17/2024 23:51:32.0404] chatter: MDNSGW-EVENT: push: added ptr record to ca
Jul 17 23:51:32 kernel: [*07/17/2024 23:51:32.0404] chatter: MDNSGW-EVENT: push: adding ptr record to c
Jul 17 23:51:32 kernel: [*07/17/2024 23:51:32.0404] chatter: MDNSGW-EVENT: mdns_ptr_db:updated TXT reco
Jul 17 23:51:32 kernel: [*07/17/2024 23:51:32.0405] chatter: MDNSGW-EVENT: mdns_ptr_db:added/updated PT
Jul 17 23:51:32 kernel: [*07/17/2024 23:51:32.0405] chatter: MDNSGW-EVENT: push: added ptr record to ca

```

Once the AP learns the services, it saves the same in the database.

The saved services in the AP database can be checked with this command:

- AP#show mdns cache

For the purpose of this document, the next output shows the relevant information to prove the mDNS Gateway AP has in its cache the services, however, the output is longer.

Next and highlighted the services, the MAC Address of the Service Provider and the Vlan where it was learned.

```
AP#show mdns cache
```

```
----- Service Provider Records-----
service_name service_provid
  _airplay._tcp.local Samsung CU7000 5
  _spotify-connect._tcp.local ed9583d2b239afa30d7b0e7106c3710dd
```

```
Total Services: 2
```

```
Total Service Providers: 2
```

```
----- PTR Records -----
service_name client_mac ap_mac ap_ether_mac wired is_rlan is_aaa_override vlan wlan_id ttl flags client
Samsung CU7000 55 TV._airplay._tcp.local E0:03:6B:45:8E:26 00:00:00:00:00:00 00:00:00:00:00:00 true fal
ed9583d2b239afa30d7b0e7106c3710ddcf5769._spotify-connect._tcp.local E0:03:6B:45:8E:26 00:00:00:00:00:00
```

Once the wired Service Provider has announced the services and the AP has cached the services and saved in its database, as shown in previous steps, the Service User (wireless client) looks to mirror the content of the device (laptop) to the smart TV for mirror display. To accomplish the mirror display, the Service User utilizes airplay service in this example.

Since the Service User is connected via wireless an Over the Air, packet capture was needed to see the connection mDNS flow from the Service User side.

From the Over the Air captures, it can be seen how the Service User which is the wireless client in Vlan 21, sends an mDNS query with the 802.11 destination MAC Address of mDNS and from the IP Address section the IP Address of mDNS is used as well as destination, the port is UDP 5353 and within the mDNS queries airplay is requested. As source the MAC Address of the Service User was used along with its IP Address.

No.	Time	Source	Destination	Protocol	Length	Info
1954	01:59:02.016646	192.168.21.54	224.0.0.251	MDNS	264	Standard query
802.11 radio information						
IEEE 802.11 QoS Data, Flags: o.....TC						
Type/Subtype: QoS Data (0x0028)						
Frame Control Field: 0x8881						
.000 0000 0011 0000 = Duration: 48 microseconds						
Receiver address: Cisco_b3:20:af (0c:75:bd:b3:20:af)						
Transmitter address: a6:c5:15:dc:dd:57 (a6:c5:15:dc:dd:57)						
Destination address: IPv4mcast_fb (01:00:5e:00:00:fb)						
Source address: a6:c5:15:dc:dd:57 (a6:c5:15:dc:dd:57)						
BSS Id: Cisco_b3:20:af (0c:75:bd:b3:20:af)						
STA address: a6:c5:15:dc:dd:57 (a6:c5:15:dc:dd:57)						
.... 0000 = Fragment number: 0						
0000 0000 1101 = Sequence number: 13						
Frame check sequence: 0x00000000 [unverified]						
[FCS Status: Unverified]						
[WLAN Flags: o.....TC]						
Qos Control: 0x1316						
HT Control (+HTC): 0x0000b20f						
Logical-Link Control						
Internet Protocol Version 4, Src: 192.168.21.54, Dst: 224.0.0.251						
User Datagram Protocol, Src Port: 5353, Dst Port: 5353						
Multicast Domain Name System (query)						
Transaction ID: 0x0000						
Flags: 0x0000 Standard query						
Questions: 4						
Answer RRs: 0						
Authority RRs: 0						
Additional RRs: 0						
Queries						
_companion-link._tcp.local: type PTR, class IN, "QU" question						
_rdlink._tcp.local: type PTR, class IN, "QU" question						
_airplay._tcp.local: type PTR, class IN, "QU" question						
_sleep-proxy._udp.local: type PTR, class IN, "QU" question						

mDNS Service User Services Request

From the AP debug, it can be seen how the AP receives a wireless mDNS packet. The debug displays the services requested which are the same services the packet capture in the previous step showed. The mDNS debugs utilized are:

- AP#debug mdns events
- AP#debug mdns packets

```

Jul 18 02:04:45 kernel: [*07/18/2024 02:04:45.1824] chatter: MDNSGW-EVENT: flex mdns gw: Recieved wirel
Jul 18 02:04:45 kernel: [*07/18/2024 02:04:45.1824] chatter: MDNSGW-PAK: query: 0/3 '_companion-link._t
Jul 18 02:04:45 kernel: [*07/18/2024 02:04:45.1824] chatter: MDNSGW-PAK: query: 1/3 '_rdlink._tcp.local
Jul 18 02:04:45 kernel: [*07/18/2024 02:04:45.1824] chatter: MDNSGW-PAK: query: 2/3 '_sleep-proxy._udp.
Jul 18 02:04:45 kernel: [*07/18/2024 02:04:45.7442] chatter: MDNSGW-PAK: query: 0/1 '_airplay._tcp.loca

```



Note: To take Over The Air packet captures with an AP in Sniffer mode, please refer to this document [Configure Access Point in Sniffer Mode on Catalyst 9800 Wireless Controllers](#). To use a MacBook to take Over The Air packet captures, please refer to this document [Collect Packet Captures Over the Air on a MacBook](#)

Once the AP is received, the mDNS query from the Service User builds an mDNS response and sends it over wireless. The response is sourced with the Access Point MAC Add and IP Address as well, the destination is the Service User (wireless client) MAC Address but, the mDNS IP Address is used with the needed services included as answers, which means this packet goes to the Service User and it is an mDNS packet.

From the packet, it can also be seen how the AP uses its own IP Address in the IP section to source the packet towards the mDNS IP Address along with the mDNS port UDP 5353, since the AP is acting as mDNS Gateway.

No.	Time	Source	Destination	Protocol	Length	Info
6077	01:59:15.050707	192.168.10.63	224.0.0.251	MDNS	720	Standard query response
<ul style="list-style-type: none"> > AiroPeek/OmniPeek encapsulated IEEE 802.11 > 802.11 radio information ✓ IEEE 802.11 QoS Data, Flags:F.C <ul style="list-style-type: none"> Type/Subtype: QoS Data (0x0028) > Frame Control Field: 0x8802 <ul style="list-style-type: none"> .000 0000 0010 1100 = Duration: 44 microseconds Receiver address: a6:c5:15:dc:dd:57 (a6:c5:15:dc:dd:57) Transmitter address: Cisco b3:20:af (0c:75:bd:b3:20:af) Destination address: a6:c5:15:dc:dd:57 (a6:c5:15:dc:dd:57) Source address: Cisco_b5:e9:d0 (0c:75:bd:b5:e9:d0) BSS Id: Cisco_b3:20:af (0c:75:bd:b3:20:af) STA address: a6:c5:15:dc:dd:57 (a6:c5:15:dc:dd:57) 0000 = Fragment number: 0 0000 0000 0110 = Sequence number: 6 Frame check sequence: 0x00000000 [unverified] [FCS Status: Unverified] [WLAN Flags:F.C] > Qos Control: 0x0006 > Logical-Link Control > Internet Protocol Version 4, Src: 192.168.10.63, Dst: 224.0.0.251 > User Datagram Protocol, Src Port: 5353, Dst Port: 5353 ✓ Multicast Domain Name System (response) <ul style="list-style-type: none"> Transaction ID: 0x0000 > Flags: 0x8000 Standard query response, No error Questions: 0 Answer RRs: 3 Authority RRs: 0 Additional RRs: 0 ✓ Answers <ul style="list-style-type: none"> > _meta_response._mdns._gateway._ap._local: type TXT, class IN > _airplay._tcp.local: type PTR, class IN, Samsung CU7000 55 TV._airplay._tcp.local > Samsung CU7000 55 TV._airplay._tcp.local: type TXT, class IN 						

mDNS Services Response from AP

From the debug, it can be seen that the mDNS response was sent to the Service User. The way to know the mDNS response was for the specific Service User is to check the MAC Address of the Service User and the MAC Address of the Access Point in the response. They are together as seen in the highlighted part of the debug shown next, as seen from the previous step in the packet capture the MAC Address of the Service User is a6c515dcdd57 and the MAC Address of the Access Point is 0c75bdb5e9d0.

Jul 18 02:04:45 kernel: [*07/18/2024 02:04:45.7450] chatter: mdns response packet 599 | a6c515dc dd570c

The previous steps complete a successful mDNS packet flow for FlexConnect Local Switching, where the Service Provider was wired connected in Vlan 11, the AP in Vlan 10 and the Service User in Vlan 21.

Wireless Service Provider

The Wireless Service provider works exactly the same as the Wired Service Provider mechanism. It sends a probing, and also an announcement for the services, the AP caches the services and saves them in the

database. This section intends to explain how the AP doing mDNS Gateway learns the services when the Service Provider is connected via wireless.

The difference between a Wired and a Wireless Service Provider is how the packet looks over the air since 802.11 takes place. In the next packet, it is seen how the Wireless Service provider in Vlan 11 sends an mDNS packet with source its own MAC Address and IP Address, and the destination is the mDNS Mac Address and IP ADDs, over port UDP 5353 with the Services listed as answers.

No.	Time	Source	Destination	Protocol	Length	Info
6567	02:48:39.311587	192.168.11.55	224.0.0.251	MDNS	969	Standard query response 0x0000 TXT, cache
IEEE 802.11 QoS Data, Flags:TC						
Type/Subtype: QoS Data (0x0028)						
> Frame Control Field: 0x8801						
.000 0000 0011 0000 = Duration: 48 microseconds						
Receiver address: Cisco_b3:20:ae (0c:75:bd:b3:20:ae)						
Transmitter address: SamsungElect 6e:eb:0c (68:fc:ca:6e:eb:0c)						
Destination address: IPv4mcast_fb (01:00:5e:00:00:fb)						
Source address: SamsungElect_6e:eb:0c (68:fc:ca:6e:eb:0c)						
BSS Id: Cisco_b3:20:ae (0c:75:bd:b3:20:ae)						
STA address: SamsungElect_6e:eb:0c (68:fc:ca:6e:eb:0c)						
.... 0000 = Fragment number: 0						
0010 1001 0001 = Sequence number: 657						
Frame check sequence: 0x00000000 [unverified]						
[FCS Status: Unverified]						
[WLAN Flags:TC]						
> Qos Control: 0x0000						
> Logical-Link Control						
> Internet Protocol Version 4, Src: 192.168.11.55, Dst: 224.0.0.251						
> User Datagram Protocol, Src Port: 5353, Dst Port: 5353						
Multicast Domain Name System (response)						
> Transaction ID: 0x0000						
> Flags: 0x8400 Standard query response, No error						
Questions: 0						
Answer RRs: 10						
Authority RRs: 0						
Additional RRs: 3						
Answers						
> ed9583d2b239afa30d7b0e7106c3710ddcfe5769._spotify-connect._tcp.local: type TXT, class IN, cache flush						
> _services._dns-sd._udp.local: type PTR, class IN, _spotify-connect._tcp.local						
> _spotify-connect._tcp.local: type PTR, class IN, ed9583d2b239afa30d7b0e7106c3710ddcfe5769._spotify-connect._tcp.local						
> ed9583d2b239afa30d7b0e7106c3710ddcfe5769._spotify-connect._tcp.local: type SRV, class IN, cache flush, priority 0, wei						
> Samsung CU7000 55 TV._airplay._tcp.local: type TXT, class IN, cache flush						

Wireless Service Provider mDNS Services

From the AP debugs, it can be seen how the AP gets a wireless mDNS packet and add the services learned to the database.

```
Jul 18 02:42:01 kernel: [*07/18/2024 02:42:01.7785] chatter: MDNSGW-EVENT: flex mdns gw: Recieved wirel
Jul 18 02:42:01 kernel: [*07/18/2024 02:42:01.7786] chatter: MDNSGW-EVENT: push: added ptr record to ca
Jul 18 02:42:01 kernel: [*07/18/2024 02:42:01.7786] chatter: MDNSGW-EVENT: push: adding ptr record to c
Jul 18 02:42:01 kernel: [*07/18/2024 02:42:01.7786] chatter: MDNSGW-EVENT: push: adding ptr record to c
Jul 18 02:42:01 kernel: [*07/18/2024 02:42:01.7787] chatter: MDNSGW-EVENT: mdns_ptr_db:updated TXT reco
Jul 18 02:42:01 kernel: [*07/18/2024 02:42:01.7787] chatter: MDNSGW-EVENT: mdns_ptr_db:added/updated PT
Jul 18 02:42:01 kernel: [*07/18/2024 02:42:01.7787] chatter: MDNSGW-EVENT: push: added ptr record to ca
```

Once the AP caches the services, the database is built and it shows some differences compared to the Wired Services provider services, since the Wireless Service provider database in the AP shows details like SSID name, site name (site TAG) and more highlighted shown next.

```
AP#show mdns cache
```

```
----- Service Provider Records -----
service_name                                     service_provider
  _airplay._tcp.local                            Samsung CU7000 55 TV._airp
 _spotify-connect._tcp.local                    ed9583d2b239afa30d7b0e7106c3710ddcfe576
```

```
Total Services: 2
```

```
Total Service Providers: 2
```

```
----- PTR Records -----
service_name client_mac ap_mac ap_ether_mac wired is_rlan is_aaa_override vlan wlan_id ttl flags client_mac
Samsung CU7000 55 TV._airplay._tcp.local 68:FC:CA:6E:EB:0C 0C:75:BD:B3:20:A0 0C:75:BD:B5:E9:D0 false f
ed9583d2b239afa30d7b0e7106c3710ddcfe5769._spotify-connect._tcp.local 68:FC:CA:6E:EB:0C 0C:75:BD:B3:20:A
```

The mDNS User Service query and the AP mDNS Gateway answer are exactly the same already explained in the Wired Service Provider section, the Service User sends an mDNS query and the AP mDNS acts as a Gateway and sends a response to the Service User with the needed services details.

Primary mDNS AP

There is only one Primary mDNS AP per Site Tag and it is in charge of two jobs:

1. Keep all the APs mDNS database updated as long as they belong to the same Site TAG, so the mDNS database of each AP is the same and there are no missing mDNS services.
2. Inform the WLC about the mDNS services learned at the remote location (this is just informational and for management purposes only, the WLC cannot use these services).

Primary AP informs update from a non-Primary AP perspective, keep in mind all the APs are in Vlan 10 in this site:

```
Jul 18 03:26:25 kernel: [*07/18/2024 03:26:25.4852] chatter: MDNSGW-EVENT: flex mdns gw: Recieved wired
Jul 18 03:26:25 kernel: [*07/18/2024 03:26:25.4853] chatter: MDNSGW-EVENT: Received _heartbeat record.
Jul 18 03:26:25 kernel: [*07/18/2024 03:26:25.4853] chatter: seq=355
Jul 18 03:26:25 kernel: [*07/18/2024 03:26:25.4853] chatter: is_primary_ap=true
Jul 18 03:26:25 kernel: [*07/18/2024 03:26:25.4854] chatter: MDNSGW-EVENT: Calculated digest=f7adbb063c
Jul 18 03:26:25 kernel: [*07/18/2024 03:26:25.4854] chatter: MDNSGW-EVENT: Verified meta message
Jul 18 03:26:25 kernel: [*07/18/2024 03:26:25.4854] chatter: MDNSGW-EVENT: [0C:75:BD:B5:E9:D0] Verified
Jul 18 03:26:25 kernel: [*07/18/2024 03:26:25.4854] chatter: MDNSGW-EVENT: New pkt from 3C:57:31:55:E4:
Jul 18 03:26:25 kernel: [*07/18/2024 03:26:25.4854] chatter: MDNSGW-EVENT: mdns_gw_ap_mgr :: MdnsGwApMg
```

```
9130mDNSAP#show mdns ap-table
```

```
AP_ETH_MAC Last_message_time Msg_seq Is_primary_ap
```

```
3C:57:31:55:E4:28 1721273666 363 YES
```

```
9130mDNSAP#
```

Primary mDNS AP informing the other APs about the services learned in the Site TAG and network the Primary AP belongs to. Once the mDNS informational packet reaches the other APs in the same site tag, the mDNS cache database is updated in the APs if new services are learned:

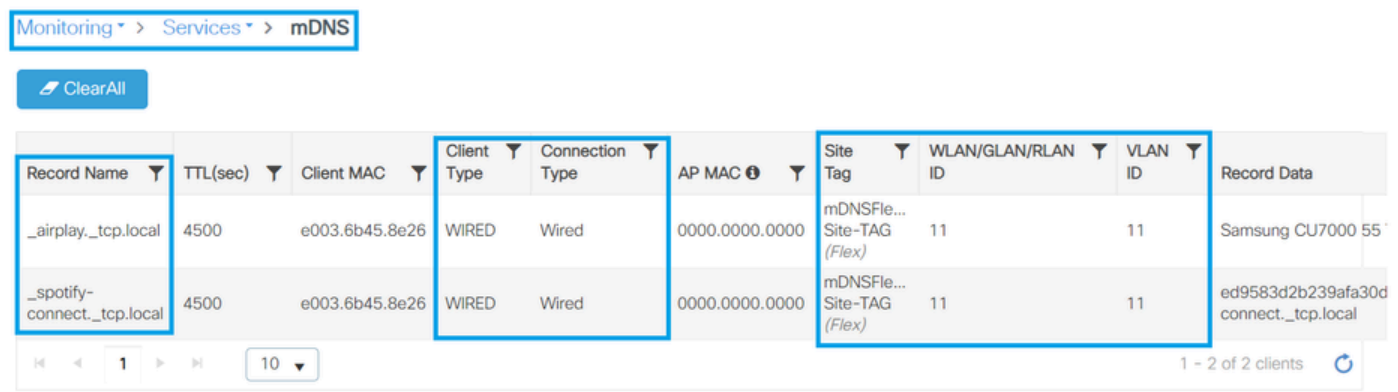
```
Jul 18 03:41:26 kernel: [*07/18/2024 03:41:26.1021] chatter: MDNSGW-EVENT: forward_packet: sending pack
Jul 18 03:41:26 kernel: [*07/18/2024 03:41:26.1023] chatter: send meta packet 177 | 01005e00 00fb3c57 3
```

Primary mDNS AP database update to the WLC:

```
Jul 18 03:35:26 kernel: [*07/18/2024 03:35:26.3127] chatter: MDNSGW-EVENT: mdns_gw_visibility :: MdnsGw
Jul 18 03:35:26 kernel: [*07/18/2024 03:35:26.3128] chatter: MDNSGW-PAK: mdns_gw_visibility :: MdnsGwVi
Jul 18 03:35:26 kernel: [*07/18/2024 03:35:26.3130] chatter: MDNSGW-EVENT: mdns_gw_visibility :: MdnsGw
Jul 18 03:35:26 kernel: [*07/18/2024 03:35:26.3131] chatter: MDNSGW-EVENT: mdns_gw_visibility :: MdnsGw
```

The services informed by the Primary AP to the WLC provide information that contains the services learned. If the services are learned via Wired or Wireless by the APs (in this example is a Wired Service Provider), the Site TAG and Vlan were learned from, and the Service Provider name. For the Wireless Service Provider, the WLAN ID reflects the WLAN the Service Provider is connected to.

Navigate to **Monitoring > Services > mDNS** as shown in the image.



mDNS Services Monitoring from WLC GUI

Services Not Allowed per mDNS Service List

The mDNS service list and policies allow it to have control of the mDNS services permitted in the network. Here, an example of how mDNS services not allowed in the Service List IN and OUT are filtered.

To see the services being advertised or queried, but not allowed, please enable this debug in the AP:

- AP#debug mdns errors

These mDNS services

- _airplay-bds._tcp.local
- _wake._tcp.local

are not allowed since they are not configured and selected in the Service List configured in the Select mDNS Services.

```
Jul 18 03:46:41 kernel: [*07/18/2024 03:46:41.6986] chatter: MDNSGW-ERROR: Handle query: service_string
Jul 18 03:46:53 kernel: [*07/18/2024 03:46:53.7270] chatter: MDNSGW-ERROR: Handle query: service_string
```


Custom mDNS Services Configuration in Service Definition

In case a special service list is needed, the same needs to be added to the Service Definition section in the mDNS configuration in the WLC. Once the services are added as a service in the WLC, and selected in the Service List IN and OUT, they are pushed to the FlexConnect APs through the mDNS Service Policy.

To do it, you need to know the exact service needed, and from the Service Definition Section, add a custom name for the service and the service string.

In this example, the two services were added that were filtered by the mDNS Gateway APs in the section Services not allowed per mDNS Service List.

Navigate to **Configuration > Services > mDNS > Service Policy > Service Definition** as shown in the image.

Configuration > Services > mDNS

Global **Service Policy** mDNS Flex Profile

Service Definition

Service List

Service Policy

Wired Filter

+ Add - Delete

<input type="checkbox"/>	Service Definition	Description	Services
<input type="checkbox"/>	AirPlayBDS		_airplay-bds._tcp.local

Quick Setup: Service Definition

Service Definition Name* WAKE

Description

Service Type

_wake._tcp.local

Cancel Apply to Device

mDNS Custom Service Definition

FlexConnect mDNS Bridging Mode

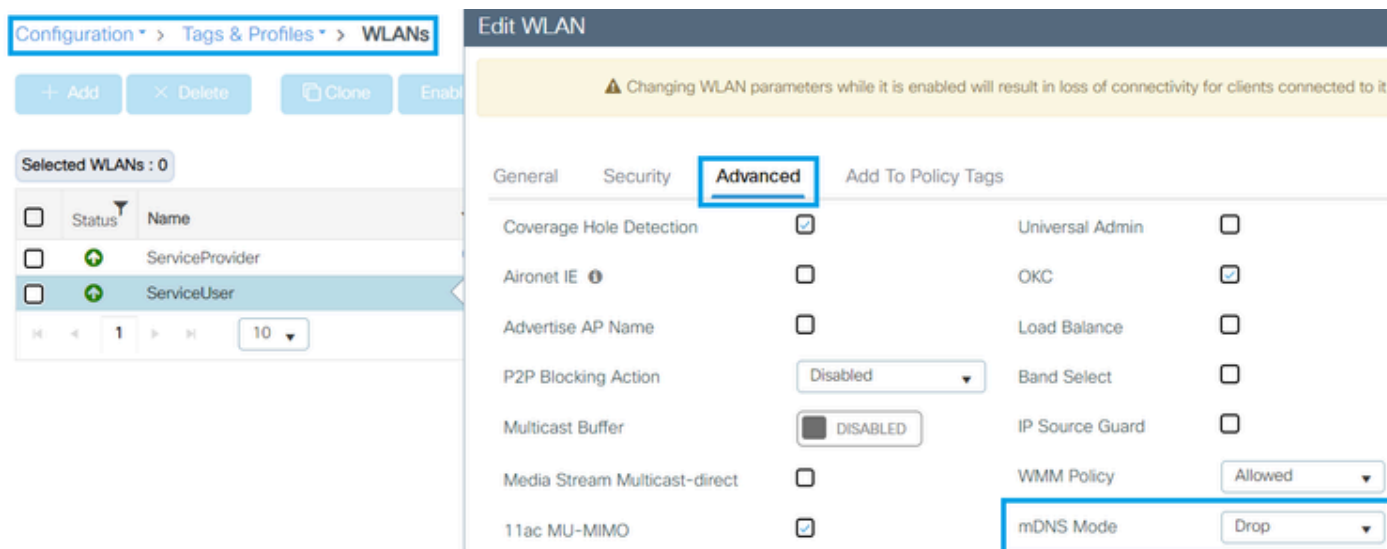
This document does not cover mDNS bridging mode due to the fact that this mDNS mode is treated as regular data traffic from the AP perspective in FlexConnect Local Switching. When bridging mode is enabled for mDNS in FlexConnect Local Switching, the AP simply forwards the mDNS packets received from the wired or wireless. These packets are forwarded only in the same Vlan, which means that the Service Provider and the Service User must be in the same Vlan for mDNS to work. mDNS Bridging does

not work across Vlans.

FlexConnect mDNS Drop Mode

If mDNS is not desired in some WLANs, but it is indeed needed in other WLANs, the mDNS mode drop can be configured per WLAN. Once mDNS drop is enabled, mDNS does not go through the devices connected to the WLAN.

Navigate to **Configuration > Tags & Profiles > WLANs** as shown in the image.



mDNS Drop Mode