

Demonstrate Client Profiling on 9800 Wireless LAN Controller

Contents

[Introduction](#)

[Requirements](#)

[Components Used](#)

[Profiling Process](#)

[MAC Address OUI Profiling](#)

[Locally Administered MAC Addresses Issues](#)

[DHCP Profiling](#)

[HTTP Profiling](#)

[RADIUS Profiling](#)

[DHCP RADIUS Profiling](#)

[HTTP RADIUS Profiling](#)

[Configure Profiling on 9800 WLC](#)

[Local Profiling Configuration](#)

[RADIUS Profiling Configuration](#)

[Profiling Use Cases](#)

[Applying Local Policies Based on Local Profiling Classification](#)

[RADIUS Profiling for Advanced Policy Sets in Cisco ISE](#)

[Profiling in FlexConnect Deployments](#)

[Central Authentication, Local Switching](#)

[Local Authentication, Local Switching](#)

[Troubleshooting](#)

[Radioactive Traces](#)

[Packet Captures](#)

Introduction

This document describes how device classification and profiling works on Cisco Catalyst 9800 Wireless LAN Controllers.

Requirements

There are no specific requirements for this document.

Components Used

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

- 9800 CL WLC running 17.2.1 image
- 1815i Access Point

- Windows 10 Pro Wireless Client
- Cisco ISE 2.7

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.

Profiling Process

This article provides an in-depth look as to how device classification and profiling works on Cisco Catalyst 9800 Wireless LAN Controllers, describes potential use cases, configuration examples, and steps necessary to troubleshoot it.

Device profiling is a feature that offers a way to find out additional info about a wireless client that has joined the wireless infrastructure.

Once device profiling is performed, it can be used to apply different local policies or to match specific RADIUS server rules.

Cisco 9800 WLCs are capable of performing three (3) types of device profiling:

1. MAC address OUI
2. DHCP
3. HTTP

MAC Address OUI Profiling

MAC address is a unique identifier of each wireless (and wired) network interface. It is a 48-bit number usually written down in a hexadecimal format MM:MM:MM:SS:SS:SS.

First 24 bits (or 3 octets) are known as Organizationally Unique Identifier (OUI) and they uniquely identify a vendor or manufacturer.

They are purchased from and assigned by the IEEE. One vendor or manufacturer can purchase multiple OUIs.

Example:

```
<#root>
```

```
00:0D:4B
```

```
- owned by Roku, LLC
```

```
90:78:B2
```

```
- owned by Xiaomi Communications Co Ltd
```

Once a wireless client associates to the access point, the WLC performs the OUI lookup to determine the manufacturer.

In Flexconnect local switching deployments, the AP still relays relevant client information to the WLC (like DHCP packets and client mac address).

Profiling based only on OUI is extremely limited and it is possible to classify device as a specific brand, but it does not able to differentiate between a laptop and smartphone.

Locally Administered MAC Addresses Issues

Due to privacy concerns, many manufacturers started implementing mac randomization features into their devices.

Locally administered MAC addresses are randomly generated and have a second-least-significant bit of the first octet of the address set to 1.

This bit acts as a flag that announces that the mac address is actually a randomly generated one.

There are four possible formats of locally administered MAC addresses (x can be any hex value):

```
x2-xx-xx-xx-xx-xx
x6-xx-xx-xx-xx-xx
xA-xx-xx-xx-xx-xx
xE-xx-xx-xx-xx-xx
```

Android 10 devices by default uses a randomly generated locally administered MAC address each time they connect to a new SSID network.

This feature completely defeats the OUI based device classification as the controller recognizes that the address has been randomized and does not perform any lookup.

DHCP Profiling

DHCP profiling is performed by WLC through investigation of the DHCP packets wireless client is sending out.

If DHCP profiling was used to classify the device, the output of **show wireless client mac-address [MAC_ADDR] detailed** command contains:

```
<#root>
Device Type      : Microsoft-Workstation
Device Name      :
MSFT 5.0
Protocol Map     : 0x000009 (OUI, DHCP)
Protocol         :
DHCP
```

WLC inspects several DHCP Option fields in the packets sent out by wireless clients:

1. Option 12 - Hostname

This option represents clients hostname and it can be found in the DHCP Discover and DHCP Request

packets:

No.	Time	Source	Destination	Protocol	Length	Info
376	476.758338	0.0.0.0	255.255.255.255	DHCP	342	DHCP Discover - Transaction ID 0x1e60cc75

```
> Ethernet II, Src: EdimaxTe_f6:76:f0 (74:cd:a:38:f6:76:f0), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
> Internet Protocol Version 4, Src: 0.0.0.0, Dst: 255.255.255.255
> User Datagram Protocol, Src Port: 68, Dst Port: 67
v Dynamic Host Configuration Protocol (Discover)
  Message type: Boot Request (1)
  Hardware type: Ethernet (0x01)
  Hardware address length: 6
  Hops: 0
  Transaction ID: 0x1e60cc75
  Seconds elapsed: 0
  > Bootp flags: 0x0000 (Unicast)
  Client IP address: 0.0.0.0
  Your (client) IP address: 0.0.0.0
  Next server IP address: 0.0.0.0
  Relay agent IP address: 0.0.0.0
  Client MAC address: EdimaxTe_f6:76:f0 (74:cd:a:38:f6:76:f0)
  Client hardware address padding: 000000000000000000000000
  Server host name not given
  Boot file name not given
  Magic cookie: DHCP
  > Option: (53) DHCP Message Type (Discover)
  > Option: (61) Client identifier
  v Option: (12) Host Name
    Length: 15
    Host Name: DESKTOP-CLB889A
```

2. Option 60 - Vendor Class Identifier

This option is also found in the DHCP Discover and Request packets.

With this option, clients can identify themselves to the DHCP server and the servers can then be configured to only respond to the clients with specific vendor class identifier.

This option is most commonly used to identify the access points in the network and only respond to them with the option 43.

Examples of Vendor Class Identifiers

- **MSFT 5.0** for all Windows 2000 clients (and up)
- **MSFT 98** for all Windows 98 and Me clients
- **MSFT** for all Windows 98, Me and 2000 clients

Apple MacBook devices do not send out Option 60 by default.

Example packet capture from Windows 10 client:

```
Option: (60) Vendor class identifier
Length: 8
Vendor class identifier: MSFT 5.0
```

3. Option 55 - Parameter Request List

DHCP Parameter Request List option contains configuration parameters (option codes) that the DHCP client

is requesting from the DHCP server. It is a string written in comma separated notation (for example 1,15,43).

It is not a perfect solution because the data it produces is vendor-dependent and can be duplicated by multiple device types.

For example, Windows 10 devices always by default request a certain parameter list. Apple iPhones and iPads use different set of parameters on which it is possible to classify them.

Example capture from Windows 10 client:

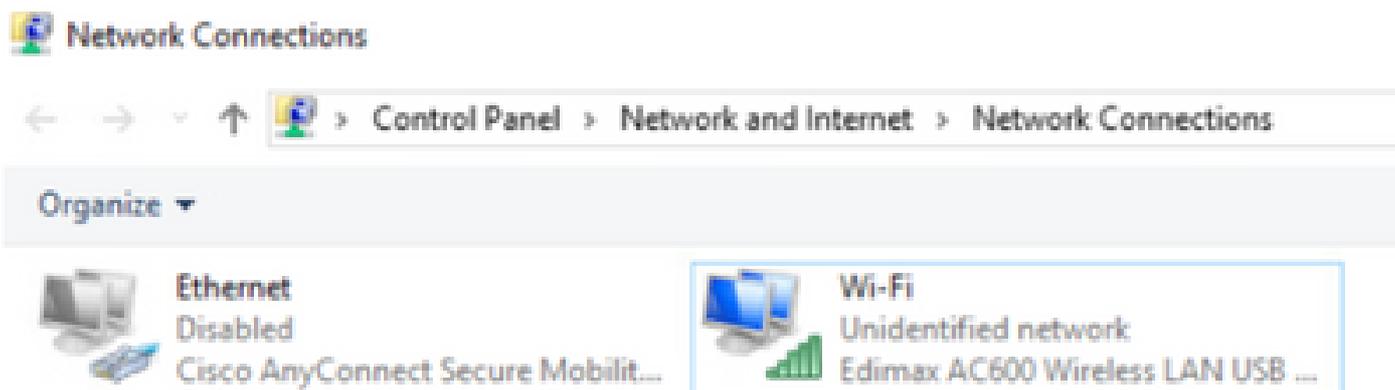
```
Option: (55) Parameter Request List
Length: 14
Parameter Request List Item: (1) Subnet Mask
Parameter Request List Item: (3) Router
Parameter Request List Item: (6) Domain Name Server
Parameter Request List Item: (15) Domain Name
Parameter Request List Item: (31) Perform Router Discover
Parameter Request List Item: (33) Static Route
Parameter Request List Item: (43) Vendor-Specific Information
Parameter Request List Item: (44) NetBIOS over TCP/IP Name Server
Parameter Request List Item: (46) NetBIOS over TCP/IP Node Type
Parameter Request List Item: (47) NetBIOS over TCP/IP Scope
Parameter Request List Item: (119) Domain Search
Parameter Request List Item: (121) Classless Static Route
Parameter Request List Item: (249) Private/Classless Static Route (Microsoft)
Parameter Request List Item: (252) Private/Proxy autodiscovery
```

4. Option 77 - User Class

User class is an option that is most commonly not used by default and requires the client to be manually configured. For example, this option can be configured on a windows machine using the command:

```
ipconfig /setclassid "ADAPTER_NAME" "USER_CLASS_STRING"
```

Adapter name can be found in the Network & Sharing Center in control panel:



Configure DHCP option 66 for Windows 10 client in CMD (requires administrator rights):

```
C:\Windows\system32>ipconfig /setclassid "Wi-Fi" "test_user_class"
Windows IP Configuration
Successfully set the DHCPv4 class id for adapter Wi-Fi.
```

Due to the Windows implementation of option 66, wireshark is not able to decode this option and part of the packet coming after option 66 shows up as malformed:

```
  ▾ Option: (77) User Class Information
    Length: 15
    ▾ Instance of User Class: [0]
      User Class Length: 116
  ▾ [Malformed Packet: DHCP/BOOTP]
    ▾ [Expert Info (Error/Malformed): Malformed Packet (Exception occurred)]
      [Malformed Packet (Exception occurred)]
      [Severity level: Error]
      [Group: Malformed]
```

HTTP Profiling

HTTP profiling is the most advanced way of profiling 9800 WLC supports and it offers the most detailed device classification. For a client to be HTTP profiled, it needs to be in a Run state and perform an HTTP GET request. WLC intercepts the request and looks into the **User-Agent** field in HTTP header of the packet. This field contains additional information about the wireless client that can be used to classify it.

By default, almost all manufacturers have implemented a feature where a wireless client tries to perform internet connectivity check. This check is also used for automatic guest portal detection. If a device receives an HTTP response with status code 200 (OK), that means the WLAN is not secured with webauth. If it is, the WLC then performs interception necessary to perform the rest of the authentication. This initial HTTP GET is not the only one WLC can use to profile the device. Every subsequent HTTP request is inspected by the WLC and it possibly results with even more detailed classification.

Windows 10 devices use the domain **msftconnecttest.com** to perform this test. Apple devices use **captive.apple.com**, while Android devices usually use **connectivitycheck.gstatic.com**.

Packet captures of the Windows 10 client performing this check can be found below. The User Agent field is populated with **Microsoft NCSI**, which results in client being profiled on the WLC as **Microsoft-Workstation**:

No.	Time	Source	Destination	Protocol	Length	Info
32	11.200752	10.40.99.235	64.102.6.247	DNS	83	Standard query 0x66d1 AAAA www.msftconnecttest.com
40	11.204057	64.102.6.247	10.40.99.235	DNS	349	Standard query response 0x66d1 A www.msftconnecttest.com CNAME vlcw
55	11.354877	10.40.99.235	13.107.4.52	HTTP	165	GET /connecttest.txt HTTP/1.1
79	11.370009	13.107.4.52	10.40.99.235	HTTP	624	HTTP/1.1 200 OK (text/plain)

```

> Frame 55: 165 bytes on wire (1320 bits), 165 bytes captured (1320 bits) on interface \Device\NPF_{764D00B3-0A27-4F08-B712-000446000000}, id 0
> Ethernet II, Src: E81aa07a_f0:76:f9 [76da:00:f9:76:f9], Dst: Cisco_20:41:e1 (24:7e:12:10:41:e1)
> Internet Protocol Version 4, Src: 10.40.99.235, Dst: 13.107.4.52
> Transmission Control Protocol, Src Port: 56815, Dst Port: 80, Seq: 1, Ack: 1, Len: 111
* Hypertext Transfer Protocol
  GET /connecttest.txt HTTP/1.1/r/ta
  [Expert Info (Chat/Sequence): GET /connecttest.txt HTTP/1.1/r/ta]
    Request Method: GET
    Request URI: /connecttest.txt
    Request Version: HTTP/1.1
    Connection: close/r/ta
    User-Agent: Microsoft MSIE/r/ta
    Host: www.msftconnecttest.com/r/ta
    /r/ta
    [Full request URI: http://www.msftconnecttest.com/connecttest.txt]
    [HTTP request 1/1]
    [Response in frame 79]

```

Example output of **show wireless client mac-address [MAC_ADDR] detailed** for a client that is profiled via HTTP:

```
<#root>
```

```
Device Type      : Microsoft-Workstation
Device Name     : MSFT 5.0
Protocol Map    : 0x000029 (OUI, DHCP, HTTP)
Device OS      :
```

```
Windows NT 10.0; Win64; x64; rv:76.0
```

```
Protocol      :
```

```
HTTP
```

RADIUS Profiling

When it comes to methods used to classify the device, there is no difference between **Local** and **RADIUS Profiling**.

If **RADIUS Profiling** is enabled, the WLC forwards the information it learned about the device through a specific set of vendor specific RADIUS attributes to the RADIUS server.

DHCP RADIUS Profiling

Information obtained through **DHCP Profiling** is sent over to the RADIUS server inside the accounting request as a vendor-specific RADIUS AVPair **cisco-av-pair: dhcp-option=<DHCP option>**.

Example of an accounting request packet showing AVPairs for DHCP option 12, 60 and 55, respectively sent from WLC to RADIUS server (option 55 value possibly appears as corrupted due to Wireshark decoding):

No.	Time	Source	Destination	Protocol	Length	Source Port	Destination Port	Info
829	9.120996	10.40.79.112	10.40.71.92	RADIUS	763	64189	1813	Accounting-Request id=282
830	9.120995	10.40.71.92	10.40.79.112	RADIUS	62	1813	64189	Accounting-Response id=282
838	9.120995	10.40.71.92	10.40.79.112	RADIUS	62	1813	64189	Accounting-Response id=282, Duplicate Response


```

Frame 829: 763 bytes on wire (6104 bits), 763 bytes captured (6104 bits)
Ethernet II, Src: 00:00:00:00:00:00 (00:00:00:00:00:00), Dst: 00:00:00:00:00:00 (00:00:00:00:00:00)
Datagram Protocol Version 6, Src: 10.40.79.112, Dst: 10.40.71.92
User Datagram Protocol, Src Port: 64189, Dst Port: 1813
RADIUS Protocol
Code: Accounting-Request (4)
Packet Identifier: 282
Length: 763
Authenticator: 2a2d44424e79e2718f5602420276c1
[The response to this request is in frame 830]
Attribute Value Pairs
- AVPair: c-Vendor-Specific (28) [0-45] vendor-ciscoSystems (R)
- AVPair: c-Vendor-Specific (28) [0-10] vendor-ciscoSystems (R)
- AVPair: c-Vendor-Specific (28) [0-12] vendor-ciscoSystems (R)
- AVPair: c-Vendor-Specific (28) [0-38] vendor-ciscoSystems (R)
- AVPair: c-Vendor-Specific (28) [0-38] vendor-ciscoSystems (R)
- AVPair: c-Vendor-Specific (28) [0-39] vendor-ciscoSystems (R)
- AVPair: c-Vendor-Specific (28) [0-39] vendor-ciscoSystems (R)
  Type: 28
  Length: 39
  Vendor ID: ciscoSystems (R)
  - Value: c-Class-IPAddr (1) [0-01] val=0000-0000-0000-0000-0000-0000-0000-0000
- AVPair: c-Vendor-Specific (28) [0-02] vendor-ciscoSystems (R)
  Type: 28
  Length: 33
  Vendor ID: ciscoSystems (R)
  - Value: c-Class-IPAddr (1) [0-01] val=0000-0000-0000-0000-0000-0000-0000-0000
- AVPair: c-Vendor-Specific (28) [0-08] vendor-ciscoSystems (R)
  Type: 28
  Length: 38
  Vendor ID: ciscoSystems (R)
  - Value: c-Class-IPAddr (1) [0-01] val=0000-0000-0000-0000-0000-0000-0000-0000

```

HTTP RADIUS Profiling

Information obtained through HTTP Profiling (**User-Agent** field from the header of HTTP GET request) sends over to the RADIUS server inside the accounting request as a vendor specific RADIUS AVPair **cisco-av-pair: http-tlv=User-Agent=<user-agent>**

Initial connectivity check HTTP GET packet does not contain much information in the **User-Agent** field, only Microsoft NCSI. Example of an accounting packet forwarding this simple value to RADIUS server:

No.	Time	Source	Destination	Protocol	Length	Source Port	Destination Port	Info
4447	25.83.845994	10.40.79.112	10.40.71.92	RADIUS	708	57997	1813	Accounting-Request id=185
4454	25.83.875000	10.40.71.92	10.40.79.112	RADIUS	62	1813	57997	Accounting-Response id=185
4455	25.83.875000	10.40.71.92	10.40.79.112	RADIUS	62	1813	57997	Accounting-Response id=185, Duplicate Response


```

User Datagram Protocol, Src Port: 57997, Dst Port: 1813
RADIUS Protocol
Code: Accounting-Request (4)
Packet Identifier: 185
Length: 708
Authenticator: 00000000000000000000000000000000
[The response to this request is in frame 4454]
Attribute Value Pairs
- AVPair: c-Vendor-Specific (28) [0-04] vendor-ciscoSystems (R)
- AVPair: c-Vendor-Specific (28) [0-07] vendor-ciscoSystems (R)
- AVPair: c-Vendor-Specific (28) [0-08] vendor-ciscoSystems (R)
- AVPair: c-Vendor-Specific (28) [0-20] vendor-ciscoSystems (R)
- AVPair: c-Vendor-Specific (28) [0-38] vendor-ciscoSystems (R)
- AVPair: c-Vendor-Specific (28) [0-39] vendor-ciscoSystems (R)
- AVPair: c-Vendor-Specific (28) [0-39] vendor-ciscoSystems (R)
  Type: 28
  Length: 35
  Vendor ID: ciscoSystems (R)
  - Value: c-Class-IPAddr (1) [0-01] val=0000-0000-0000-0000-0000-0000-0000-0000

```

Once the user starts browsing the internet and creates some additional HTTP GET requests, it is possible to gain more information about it. WLC sends additional accounting packet to the ISE if it detects new **User-Agent** values for this client. In this example, it is possible to see that the client is using Windows 10 64 bit and Firefox 76:

Configuration > Wireless > **Wireless Global**

Default Mobility
Domain *

default



RF Group
Name*

default

Maximum Login
Sessions Per
User*

0

Management Via
Wireless

Device
Classification

AP LAG Mode

Additionally, under Policy configuration you can enable **HTTP TLV Caching** and **DHCP TLV Caching**. WLC performs profiling even if without them.

With these options enabled, the WLC then cache previously learned information about this client and avoid the need to inspect additional packets generated by this device.

Edit Policy Profile

General

Access Policies

QOS and AVC

Mobility

Advanced

RADIUS Profiling

HTTP TLV Caching

DHCP TLV Caching

WLAN Local Profiling

Global State of Device Classification

Enabled ⓘ

Local Subscriber Policy Name

BlockPolicy

RADIUS Profiling Configuration

In order for RADIUS Profiling to work, besides globally enabling device classification (like mentioned in **Local Profiling** configuration), it is necessary to:

1. Configure the **AAA Method List > Accounting** with type **identity** pointing towards the RADIUS server:

Configuration > Security > AAA

+ AAA Wizard

Servers / Groups

AAA Method List

AAA Advanced

Authentication

Authorization

Accounting

Name	Type	Group1	Group2	Group3	Group4
AccMethod	identity	RADIUS	N/A	N/A	N/A

20 items per page 1 of 1 items

2. Accounting method needs to be added under **Configuration > Tags & Profiles > Policy > [Policy_Name] > Advanced**:

Edit Policy Profile

General

Access Policies

QoS and AVC

Mobility

Advanced

WLAN Timeout

Session Timeout (sec)

Idle Timeout (sec)

Idle Threshold (bytes)

Client Exclusion Timeout (sec)

Guest LAN Session Timeout

DHCP

IPv4 DHCP Required

DHCP Server IP Address

[Show more >>>](#)

AAA Policy

Allow AAA Override

NAC State

NAC Type

Policy Name

Accounting List

Fabric Profile

mDNS Service Policy [Clear](#)

Hotspot Server

User Private Network

Status

Drop Unicast

Umbrella

Umbrella Parameter Map [Clear](#)

Flex DHCP Option for DNS **ENABLED**

DNS Traffic Redirect **IGNORE**

WLAN Flex Policy

VLAN Central Switching

Split MAC ACL

Air Time Fairness Policies

3. Finally, **RADIUS Local Profiling** checkbox needs to be ticked under **Configuration > Tags & Profiles > Policy**. This checkbox enables both **HTTP** and **DHCP RADIUS Profiling** (old AireOS WLCs had 2 separate checkboxes):

Edit Policy Profile

General

Access Policies

QoS and AVC

Mobility

Advanced

RADIUS Profiling

HTTP TLV Caching

DHCP TLV Caching

WLAN Local Profiling

Global State of Device Classification

Enabled ⓘ

Local Subscriber Policy Name

BlockPolicy

Profiling Use Cases

Applying Local Policies Based on Local Profiling Classification

This sample configuration demonstrates configuration of **Local Policy** with **QoS** profile blocking **YouTube** and **Facebook** access that is applied only to devices profiled as **Windows-Workstation**.

With slight changes, this configuration can be modified to, for example, set specific DSCP marking for only wireless phones.

Create a QoS profile by navigating to **Configuration > Services > QoS**. Click **Add** to create new policy:



Specify the policy name and add a new class map. From the available protocols, select the ones that need to be blocked, DSCP marked or bandwidth limited.

In this example, **YouTube** and **Facebook** are blocked. Make sure not to apply this **QoS** profile to any of the **Policy Profiles** at the bottom of the **QoS** window:

Configuration > Security > Local Policy

Service Template

Policy Map

+ Add

- Delete

Service Template Name	Source
<input type="checkbox"/> webauth-global-inactive	
<input type="checkbox"/> DEFAULT_CRITICAL_DATA_TEMPLATE	
<input type="checkbox"/> DEFAULT_CRITICAL_VOICE_TEMPLATE	
<input type="checkbox"/> DEFAULT_LINKSEC_POLICY_MUST_SECURE	
<input type="checkbox"/> DEFAULT_LINKSEC_POLICY_SHOULD_SECURE	

1 - 5 of 5 items

Specify **Ingress QoS** and **Egress QoS** profiles that were created in the previous step. An access list can also be applied in this step. If no VLAN change is necessary, leave the **VLAN ID** field empty:

Create Service Template

Service Template Name*	BlockTemplate
VLAN ID	1-4094
Session Timeout (secs)	1-65535
Access Control List	None
Ingress QoS	block
Egress QoS	block
mDNS Service Policy	Search or Select



Cancel

Apply to Device

Navigate to the **Policy Map** tab and click **Add**:

Service Template **Policy Map**

+ Add **< Delete**

Policy Map Name
<input type="checkbox"/> BUILTIN_AUTOCONF_POLICY

1 - 1 of 1 items

Set the **Policy Map** name and add new criteria. Specify the **Service Template** that was created in the previous step and select the **Device Type** that this template is applied to.

In this case, **Microsoft-Workstation** is used. If multiple policies are defined, the first match is used.

One other common use case would be to specify OUI based match criteria. If a deployment has a large number of scanners or printers of the same model, they usually have the same MAC OUI.

This can be used to apply specific QoS DSCP marking or an ACL:

Create Policy Map Configuration

Policy Map Name *

Match Criteria List

+ Add **< Delete** **Move To** **+ Move Up** **+ Move Down**

Device Type(Match Criteria)	User Role(Match Criteria)	User Name(Match Criteria)	OUI(Match Criteria)	MAC Address(Match Criteria)	Service Template
No items to display					

Add Match Criteria

Service Template *

Device Type

User Role

User Name

OUI

MAC Address

Add Criteria **Cancel**

Cancel **Apply to Device**

In order for WLC to be able to recognize the **YouTube** and **Facebook** traffic, **Application Visibility** needs to be turned on.

Navigate to **Configuration > Services > Application Visibility** enable visibility for the **Policy Profile** of your WLAN:

Configuration > Services > Application Visibility

Enable AVC 0 0 0

Define Policy

- Relaxed
- Consistent
- Default

Drag and Drop, double click or click on the button from Selected Profiles to add/remove Profiles

Apply

Search

Available (11)

Profiles

- 11 bandwidth
- 11 mobility
- 11 profiling
- 33 nps
- Capwap 1
- default-policy-profile

Enabled (1)

Profiles	Visibility	Collector Address
11 override	<input checked="" type="checkbox"/>	Local <input checked="" type="checkbox"/> External <input type="checkbox"/>

Enable All up down administratively down Disable All

Verify that under the **Policy Profile** the **HTTP TLV Caching**, **DHCP TLV Caching**, **Global device Classification** are enabled and that **Local Subscriber Policy Name** is pointing to the local policy map that was created in one of the previous steps:

Edit Policy Profile

General **Access Policies** QOS and AVC Mobility Advanced

RADIUS Profiling

HTTP TLV Caching

DHCP TLV Caching

WLAN Local Profiling

Global State of Device Classification **Enabled**

Local Subscriber Policy Name **BlockPolicy**

WLAN

VLAN/VLAN Group **VLAN0039**

Multicast VLAN **Enter Multicast VLAN**

WLAN ACL

IPv4 ACL **Search or Select**

IPv6 ACL **Search or Select**

URL Filters

Pre Auth **Search or Select**

Post Auth **Search or Select**

After the client connects, it is possible to check if the local policy has been applied and test if **YouTube** and **Facebook** are actually blocked. Output of the show wireless client mac-address [MAC_ADDR] detailed contains:

```
<#root>
```

```
Input Policy Name :
```

```
block
```

```
Input Policy State : Installed
```

```
Input Policy Source : Native Profile Policy
```

```
Output Policy Name :
```

```
block
```

```
Output Policy State : Installed
```

```
Output Policy Source : Native Profile Policy
```

```
Local Policies:
```

```
Service Template : BlockTemplate (priority 150)
```

```
Input QoS :
```

```
block
```

```
Output QoS :
```

```
block
```

```
Service Template : wlan_svc_11override_local (priority 254)
```

```
VLAN : VLAN0039
```

```
Absolute-Timer : 1800
```

```
Device Type :
```

```
Microsoft-Workstation
```

```
Device Name :
```

```
MSFT 5.0
```

```
Protocol Map : 0x000029 (OUI, DHCP, HTTP)
```

```
Protocol :
```

```
HTTP
```

RADIUS Profiling for Advanced Policy Sets in Cisco ISE

With RADIUS profiling enabled, the WLC forwards profiling information to the ISE. Based on this info, it is possible to create advanced authentication and authorization rules.

This article does not cover ISE configuration. Please refer to the [Cisco ISE Profiling Design Guide](#) for more information.

This workflow usually requires the use of CoA, so make sure it is enabled on the 9800 WLC.

Profiling in FlexConnect Deployments

Central Authentication, Local Switching

In this setup, both Local and RADIUS profiling continues to work exactly like described in previous chapters. If AP goes into standalone mode (AP loses connection to the WLC), device profiling stops working and no new clients are able to connect.

Local Authentication, Local Switching

If AP is in connected mode (AP joined to the WLC), profiling continues to work (AP sends a copy of client DHCP packets to the WLC to perform the profiling process).

Despite profiling working, since authentication is performed locally on the AP, profiling information cannot be utilized for any Local Policy configuration or RADIUS profiling rules.

Troubleshooting

Radioactive Traces

The easiest way to troubleshoot client profiling on the WLC is via radioactive traces. Navigate to **Troubleshooting > Radioactive Trace**, enter the client wireless adapter MAC address and click **Start**:

Troubleshooting > Radioactive Trace

Conditional Debug Global State: **Started**



	MAC/IP Address	Trace file	
<input type="checkbox"/>	74da.38f6.76f0	debugTrace_74da.38f6.76f0.txt ↓	▶ Generate

⏪ ⏩ 1 ⏪ ⏩ 20 items per page 1 - 1 of 1 items

Connect the client to the network and wait until it reaches run state. Stop the traces and click **Generate**. Make sure that Internal Logs are enabled (this option only exists in 17.1.1 and later releases):

Enter time interval



Enable Internal Logs



Generate logs for last

10 minutes

30 minutes

1 hour

since last boot

0-4294967295

seconds



Cancel

Apply to Device



Relevant snippets from the radioactive trace can be found next.

Client getting profiled by WLC as Microsoft-Workstation:

```
<#root>
```

```
2020/06/18 10:46:41.052366 {wncd_x_R0-0}{1}: [auth-mgr] [21168]: (info): [74da.38f6.76f0:capwap_9000000
```

```
Microsoft-Workstation
```

```
and old device-type not classified earlier &Device name for the session is detected as
```

```
MSFT 5.0
```

```
and old device-name not classified earlier & Old protocol map 0 and new is 41
```

```
2020/06/18 10:46:41.052367 {wncd_x_R0-0}{1}: [auth-mgr] [21168]: (debug): [74da.38f6.76f0:capwap_9000000
```

```
updating device type Microsoft-Workstation
```

```
, device name
```

```
MSFT 5.0
```

WLC caching the device classification:

(debug): [74da.38f6.76f0:unknown] Updating cache for mac [74da.38f6.76f0] device_type: Microsoft-Workst

WLC finding the device classification inside the cache:

(info): [74da.38f6.76f0:capwap_90000004] Device type found in cache Microsoft-Workstation

WLC applying local policy based on classification:

<#root>

(info): device-type filter: Microsoft-Workstation required, Microsoft-Workstation set -

match for 74da.38f6.76f0

/ 0x9700001A

(info): device-type Filter evaluation succeeded

(debug): match device-type eq "

Microsoft-Workstation

" :success

WLC sending accounting packets containing DHCP and HTTP Profiling attribute:

<#root>

[caaa-acct] [21168]: (debug): [CAAA:ACCT:c9000021] Accounting session created

[auth-mgr] [21168]: (info): [74da.38f6.76f0:capwap_90000004] Getting active filter list

[auth-mgr] [21168]: (info): [74da.38f6.76f0:capwap_90000004]

Found http

[auth-mgr] [21168]: (info): [74da.38f6.76f0:capwap_90000004]

Found dhcp

[aaa-attr-inf] [21168]: (debug): Filter list http-tlv 0

[aaa-attr-inf] [21168]: (debug): Filter list dhcp-option 0

[aaa-attr-inf] [21168]: (debug): Get acct attrs dc-profile-name 0 "

Microsoft-Workstation

"

[aaa-attr-inf] [21168]: (debug): Get acct attrs dc-device-name 0 "

MSFT 5.0

"

[aaa-attr-inf] [21168]: (debug): Get acct attrs dc-device-class-tag 0 "

Workstation:Microsoft-Workstation

"

```
[aaa-attr-inf] [21168]: (debug): Get acct attrs dc-certainty-metric 0 10 (0xa)  
[aaa-attr-inf] [21168]: (debug): Get acct attrs
```

```
dhcp-option 0 00 0c 00 0f 44 45 53 4b 54 4f 50 2d 4b 4c 52 45 30 4d 41
```

```
[aaa-attr-inf] [21168]: (debug): Get acct attrs
```

```
dhcp-option 0 00 3c 00 08 4d 53 46 54 20 35 2e 30
```

```
[aaa-attr-inf] [21168]: (debug): Get acct attrs
```

```
dhcp-option 0 00 37 00 0e 01 03 06 0f 1f 21 2b 2c 2e 2f 77 79 f9 fc
```

```
### http profiling sent in a separate accounting packet
```

```
[aaa-attr-inf] [21168]: (debug): Get acct attrs http-tlv 0 00 01 00 0e 4d 69 63 72 6f 73 6f 66 74 20 4e
```

Packet Captures

In a centrally switched deployment, packet captures can be performed on the WLC itself. Navigate to **Troubleshooting > Packet Capture** and create a new capture point on one of the interfaces that are in use by this client.

It is required to have SVI on the VLAN in order to perform capture on it, otherwise take the capture on the physical port itself

Troubleshooting > Packet Capture

+ Add - Delete

Capture Name	Interface	Monitor Control Plane	Buffer Size	Filter by	Limit	Status	Action
0	20	items per page				No items to display	

Create Packet Capture

Capture Name* capture

Filter* any

Monitor Control Plane

Buffer Size (MB)* 10

Limit by* Duration 3600 secs == 1.00 hour

Available (4)

- GigabitEthernet1
- GigabitEthernet2
- GigabitEthernet3
- Vlan1

Selected (1)

- Vlan39

Cancel Apply to Device

Related Information

- [Cisco Technical Support & Downloads](#)