Configure MAC Authentication SSID on Catalyst 9800 Wireless Controllers

Contents

Introduction

This document describes how to set up a Wireless Local Area Network (WLAN) with MAC authentication security on Cisco Catalyst 9800 WLC.

Prerequisites

Requirement

Cisco recommends that you have knowledge of these topics:

- MAC address
- Cisco Catalyst 9800 Series Wireless Controllers
- Identity Service Engine (ISE)

Components Used

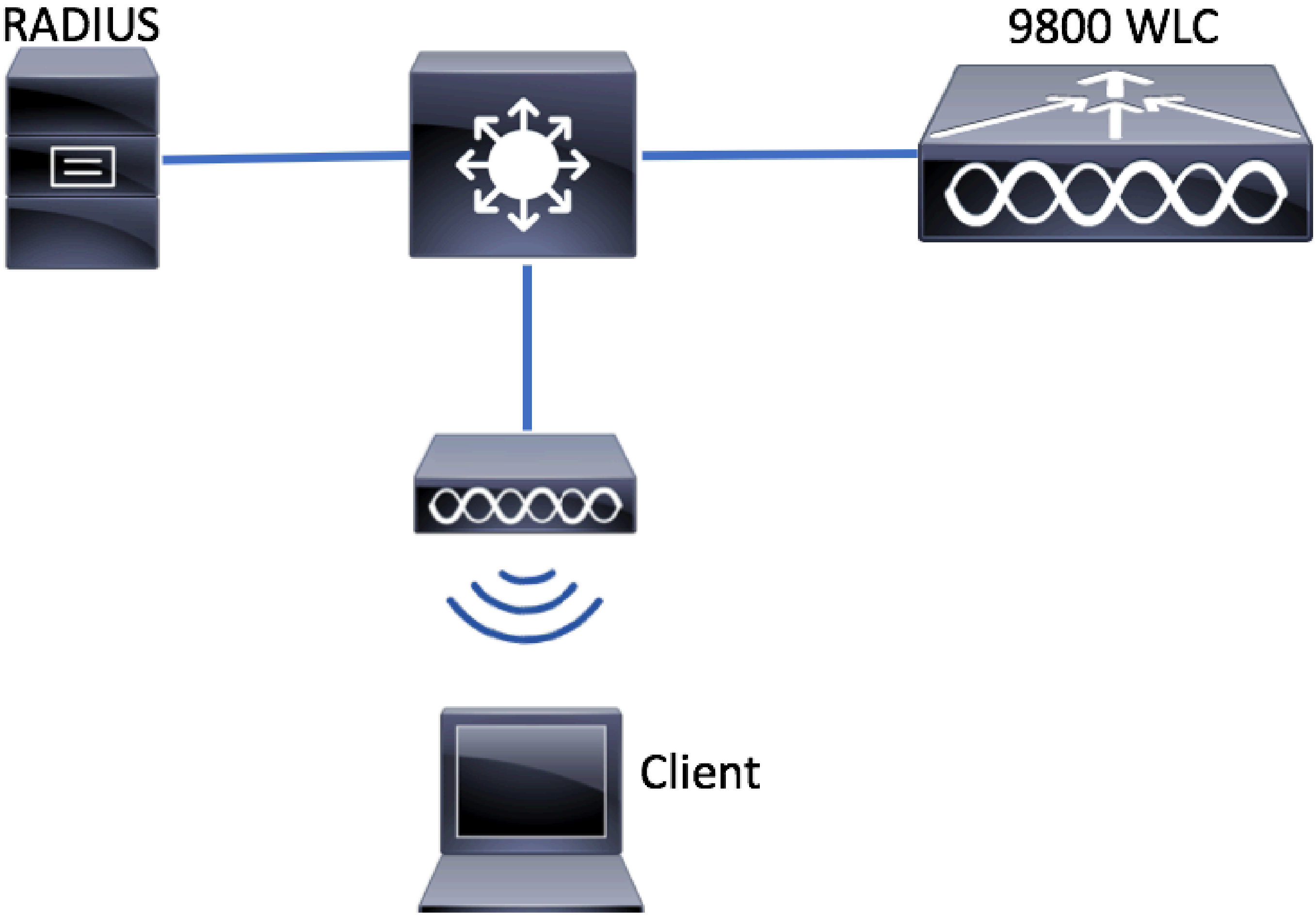
The information in this document is based on these software and hardware versions:

- Cisco IOS® XE Gibraltar v16.12
- ISE v2.2

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

Configure

Network Diagram



AAA Configuration on 9800 WLC

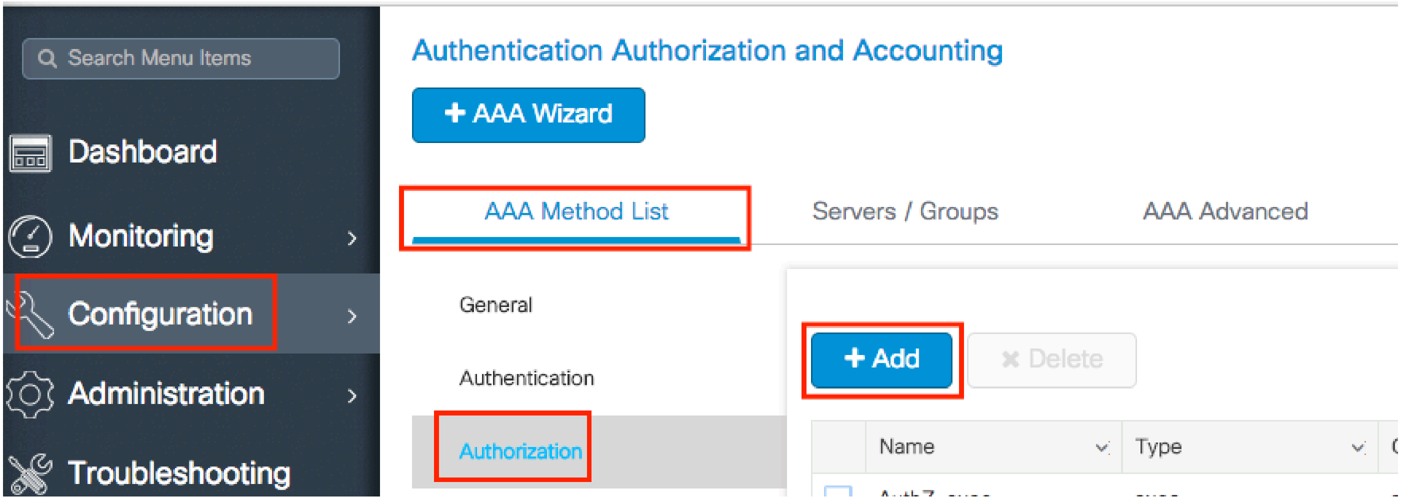
Authenticate Clients with External Server

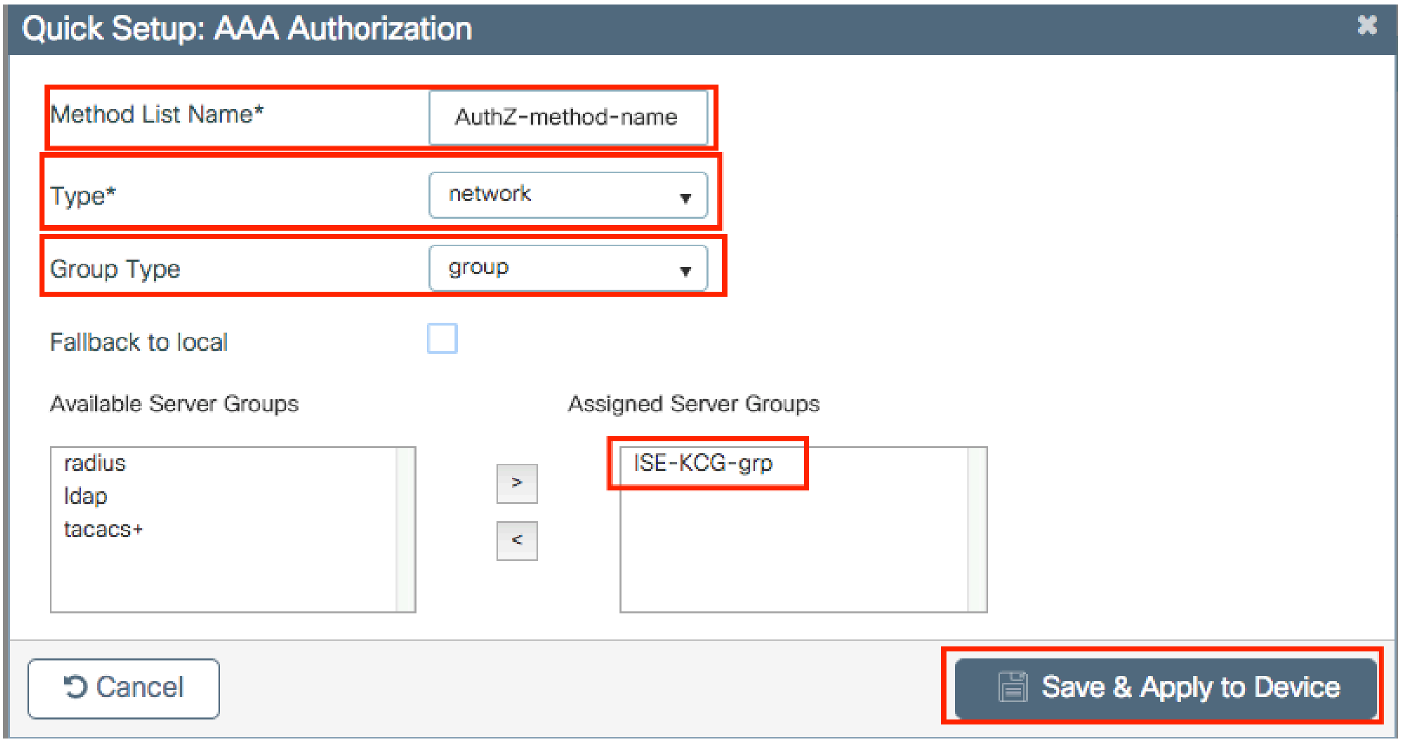
GUI:

Read Steps 1-3 of section of AAA Configuration on 9800 Series WLC.

Step 4. Create an authorization network method.

Navigate to Configuration > Security > AAA > AAA Method List > Authorization > + Add and create it.





Method List Name*	AuthZ-method-name
Type*	network
Group Type	group
Fallback to local	□

Available Server Groups	Assigned Server Groups
radius ldap tacacs+	ISE-KCG-grp

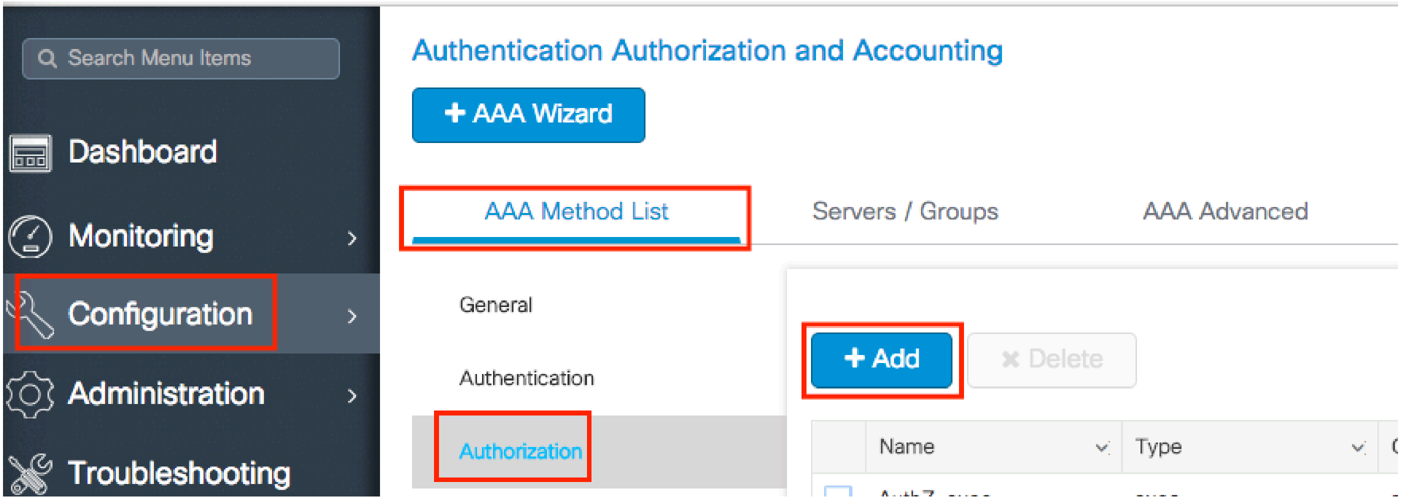
CLI:

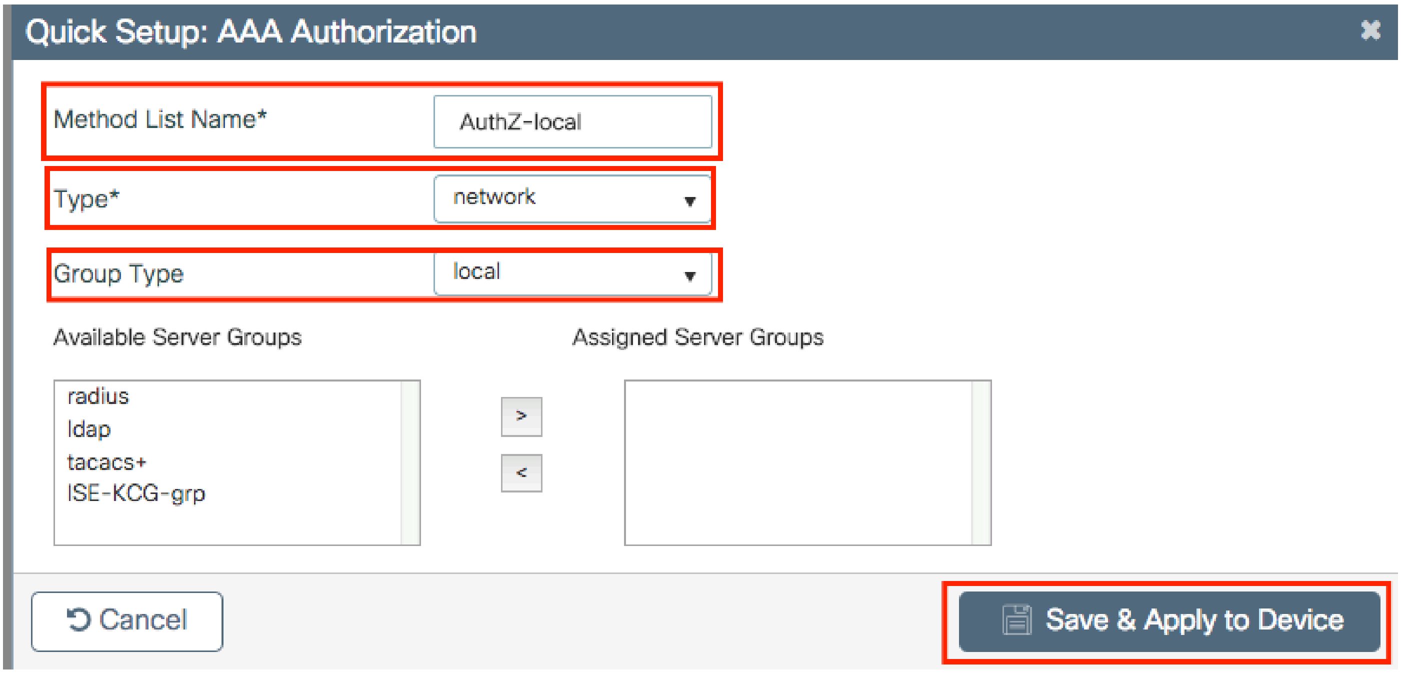
```
# config t
# aaa new-model

# radius server <radius-server-name>
# address ipv4 <radius-server-ip> auth-port 1812 acct-port 1813
# timeout 300
# retransmit 3
# key <shared-key>
# exit

# aaa group server radius <radius-grp-name>
# server name <radius-server-name>
# exit

# aaa server radius dynamic-author
# client <radius-server-ip> server-key <shared-key>
# aaa authorization network <AuthZ-method-name> group <radius-grp-name>
```

Authenticate Clients Locally

Create a local authorization network method.

Navigate to Configuration > Security > AAA > AAA Method List > Authorization > + Add and create it.




Quick Setup: AAA Authorization

Method List Name* AuthZ-local

Type* network

Group Type local

CLI:

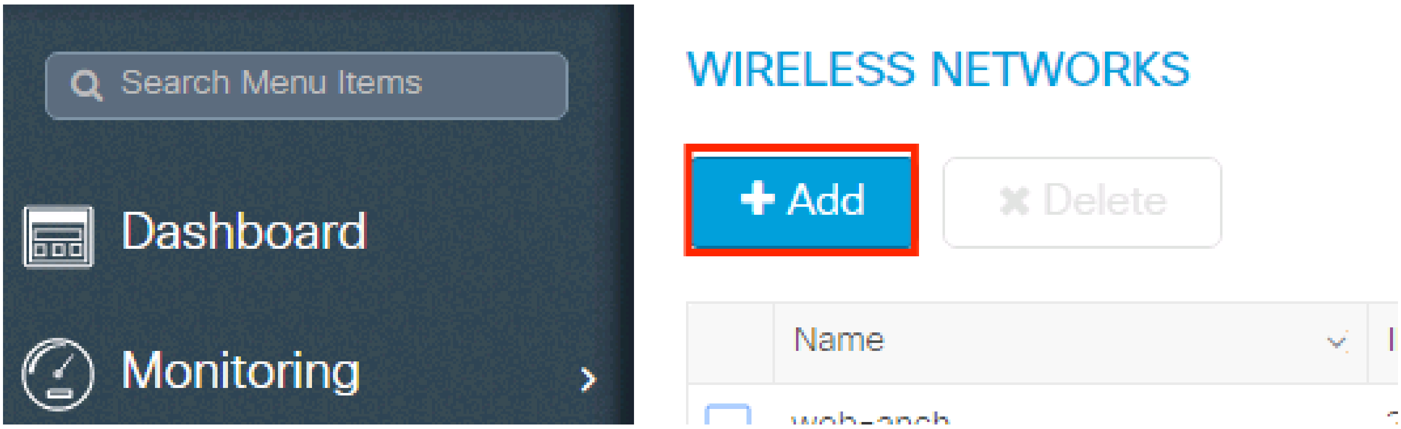
```
# config t
# aaa new-model
# aaa authorization network AuthZ-local local
```

WLAN Configuration

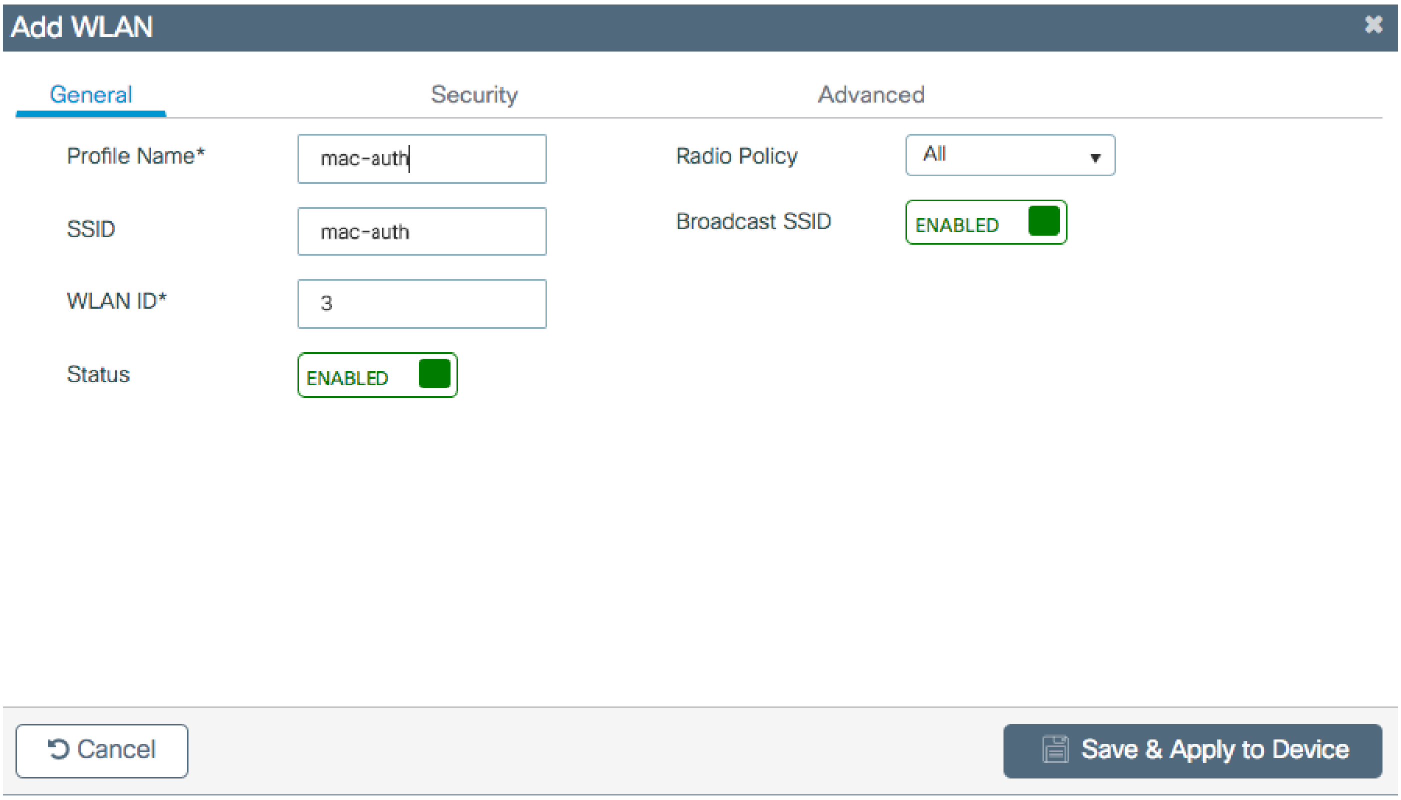
GUI:

Step 1. Create the WLAN.

Navigate to Configuration > Wireless > WLANs > + Add and configure the network as needed.

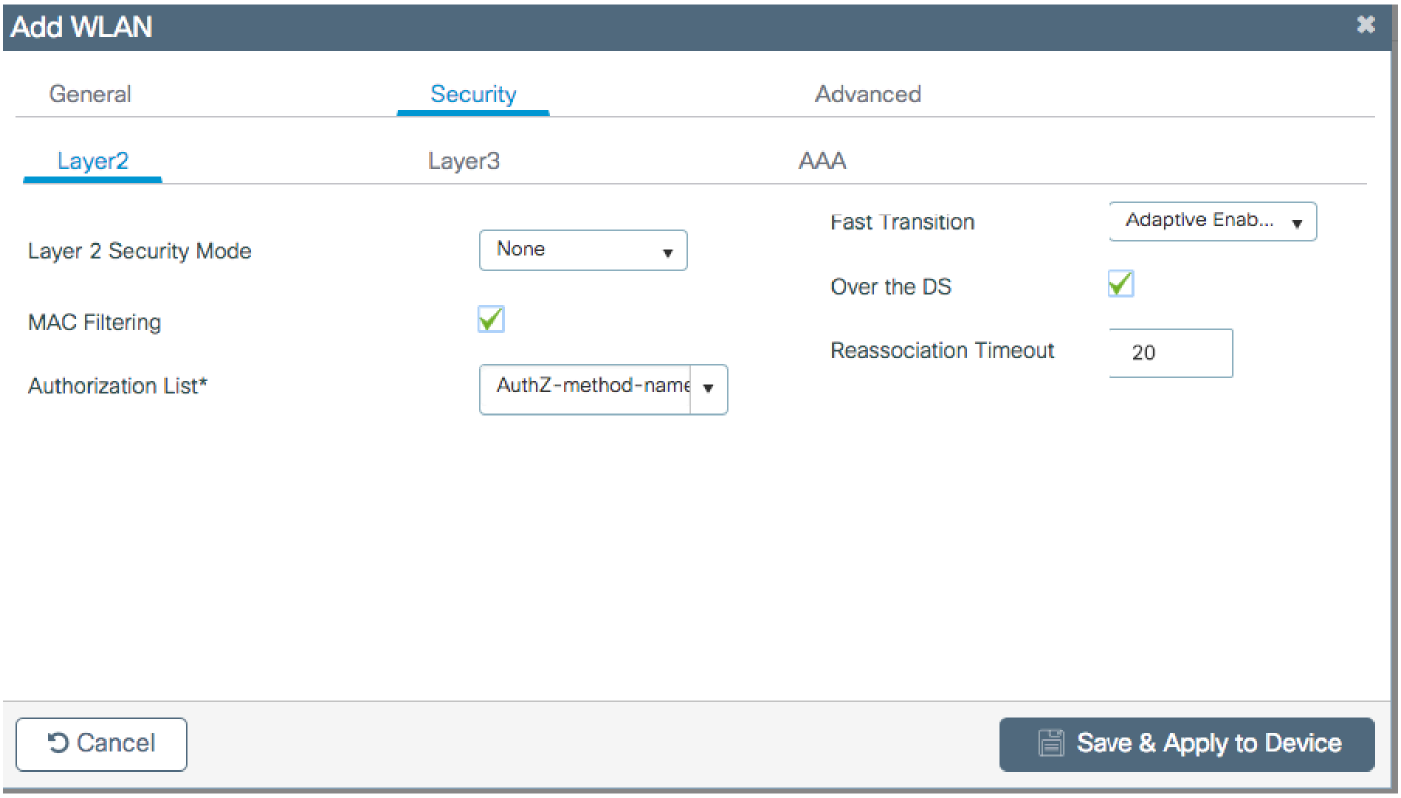


Step 2. Enter the WLAN information.



Profile Name*	mac-auth	Radio Policy	All
SSID	mac-auth	Broadcast SSID	ENABLED
WLAN ID*	3		
Status	ENABLED		

Step 3. Navigate to the Security tab and disable Layer 2 Security Mode and enable MAC Filtering. From Authorization List, choose the authorization method created in the previous step. Then click Save & Apply to Device.



Layer 2 Security Mode: None

MAC Filtering: ☑

Authorization List*: AuthZ-method-name

Fast Transition: Adaptive Enab...

Over the DS: ☑

Reassociation Timeout: 20

CLI:

```
# config t
# wlan <profile-name> <wlan-id> <ssid-name>
# mac-filtering <authZ-network-method>
# no security wpa akm dot1x
# no security wpa wpa2 ciphers aes
# no shutdown
```

Policy Profile Configuration

You must enable `aaa-override` in the policy profile to ensure that the mac-filtering per SSID works fine.

Policy Profile Configuration on 9800 WLC

Policy Tag Configuration

Policy Tag on 9800 WLC

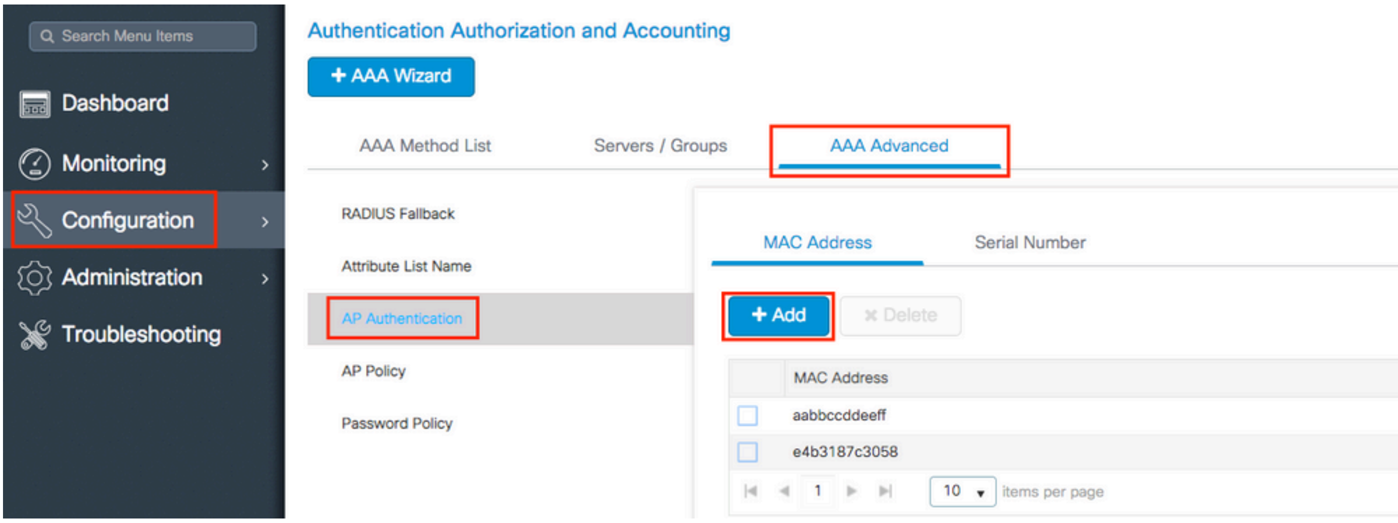
Policy Tag Assignation

Policy Tag Assignation on 9800 WLC

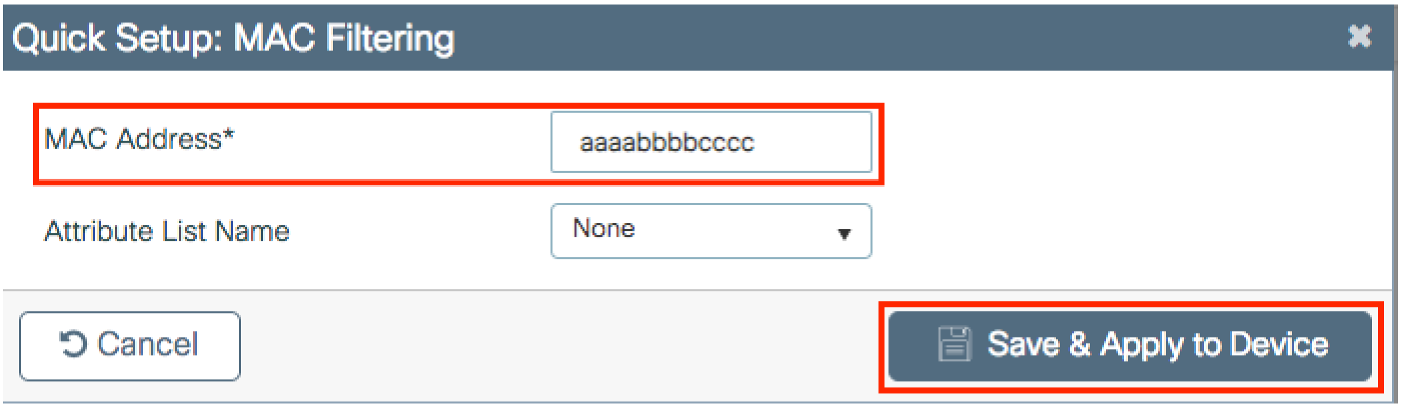
Register the allowed MAC address.

Locally Register the MAC Address on the WLC for Local Authentication

Navigate to Configuration > Security > AAA > AAA Advanced > AP Authentication > + Add.



Write the mac address in all lowercase without a separator, and click Save & Apply to Device.



MAC Address* aaaabbbbcccc

Attribute List Name None

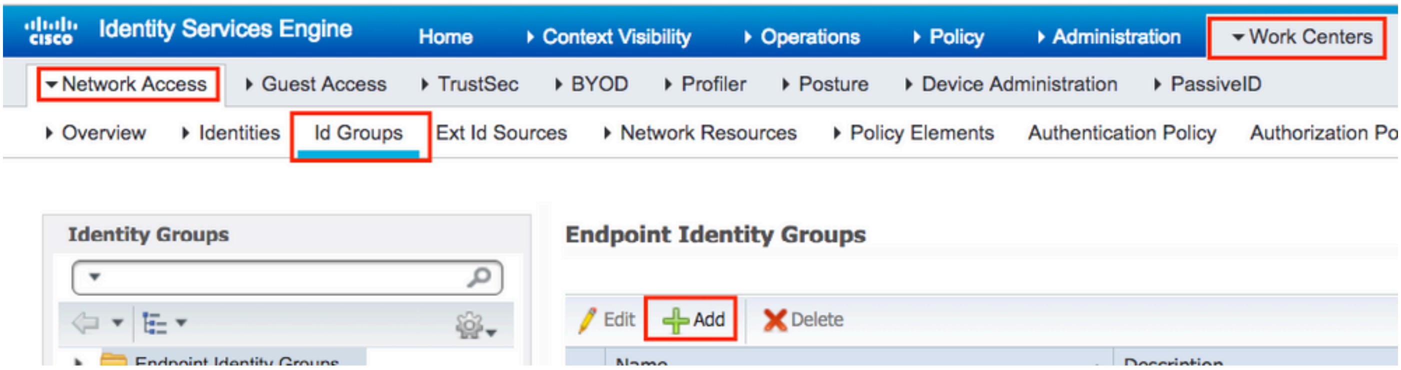
Note: In versions earlier than 17.3, the Web User Interface (UI) changed any MAC format you typed into the **no separator** format shown in the illustration. In 17.3 and later, the Web UI respects whatever design you entered and it is, therefore, essential not to enter any separator. Enhancement bug Cisco bug ID CSCvv43870 tracks the support of several formats for MAC authentication.

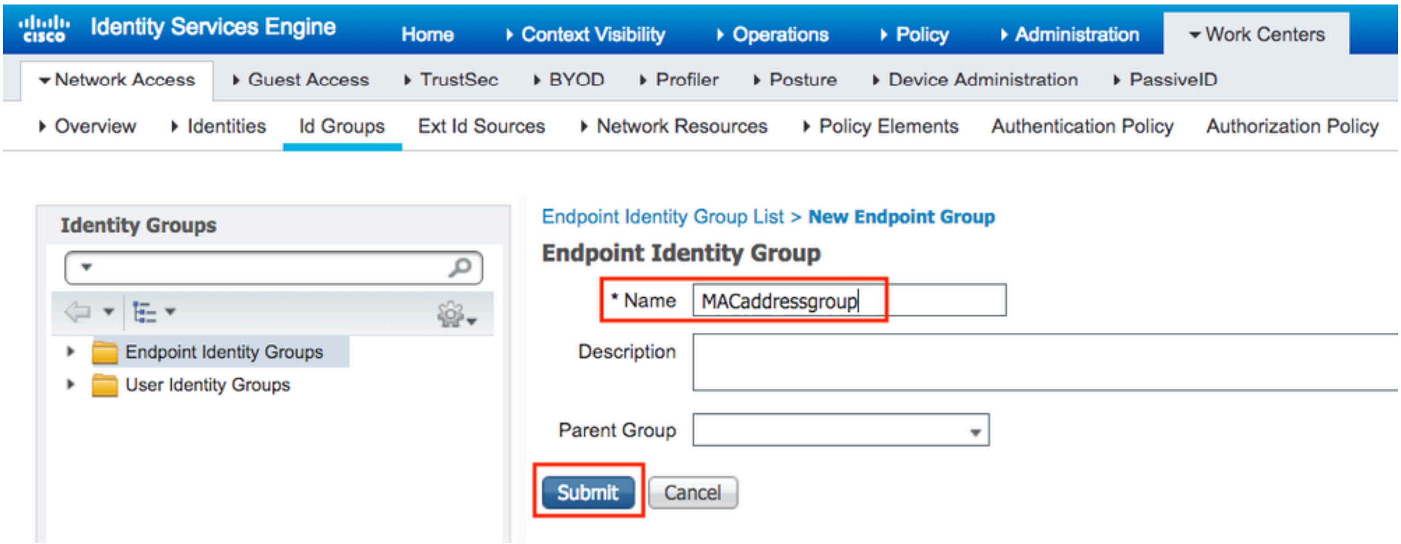
CLI:

```
# config t
# username <aabbccddeeff> mac
```

Enter the MAC Address on the ISE Endpoint Database

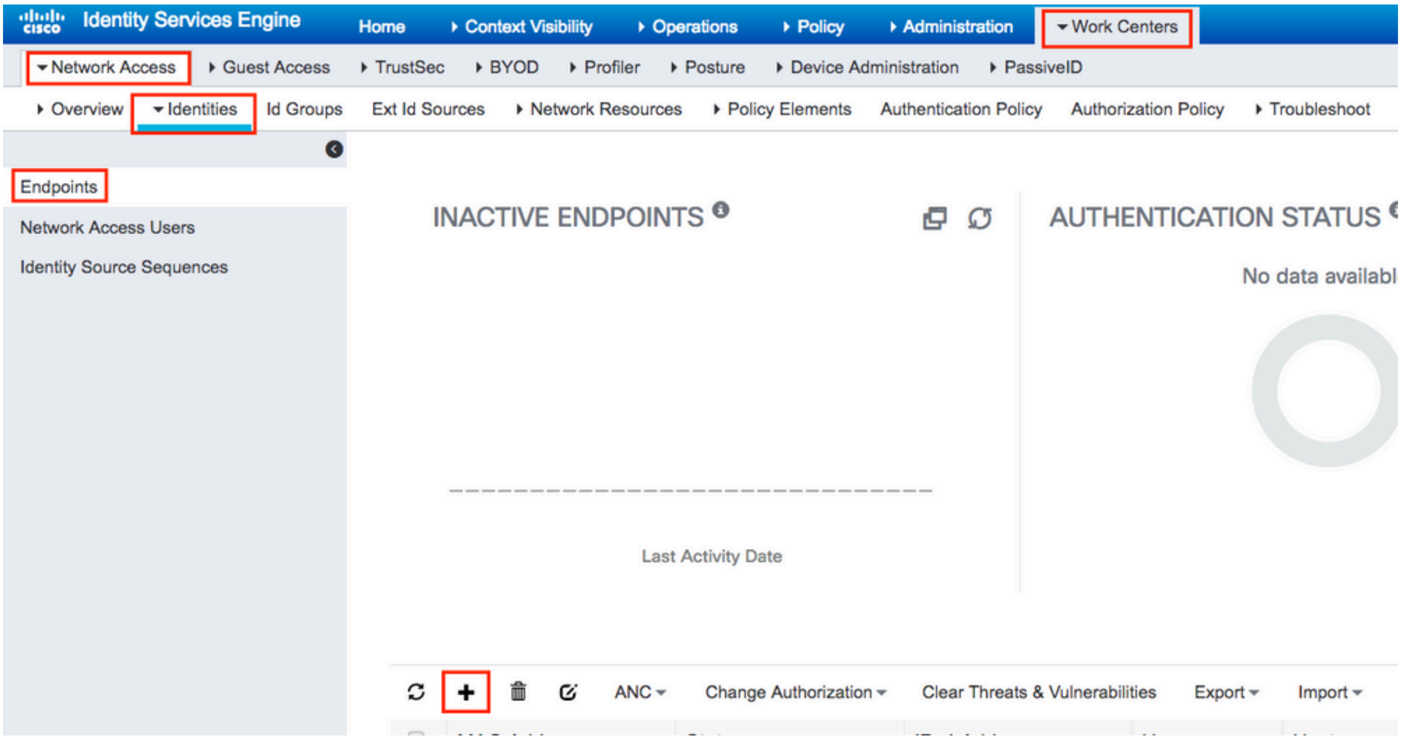
Step 1. (Optional) Create a new Endpoint group.
Navigate to Work Centers > Network Access > Id Groups > Endpoint Identity Groups > + Add.

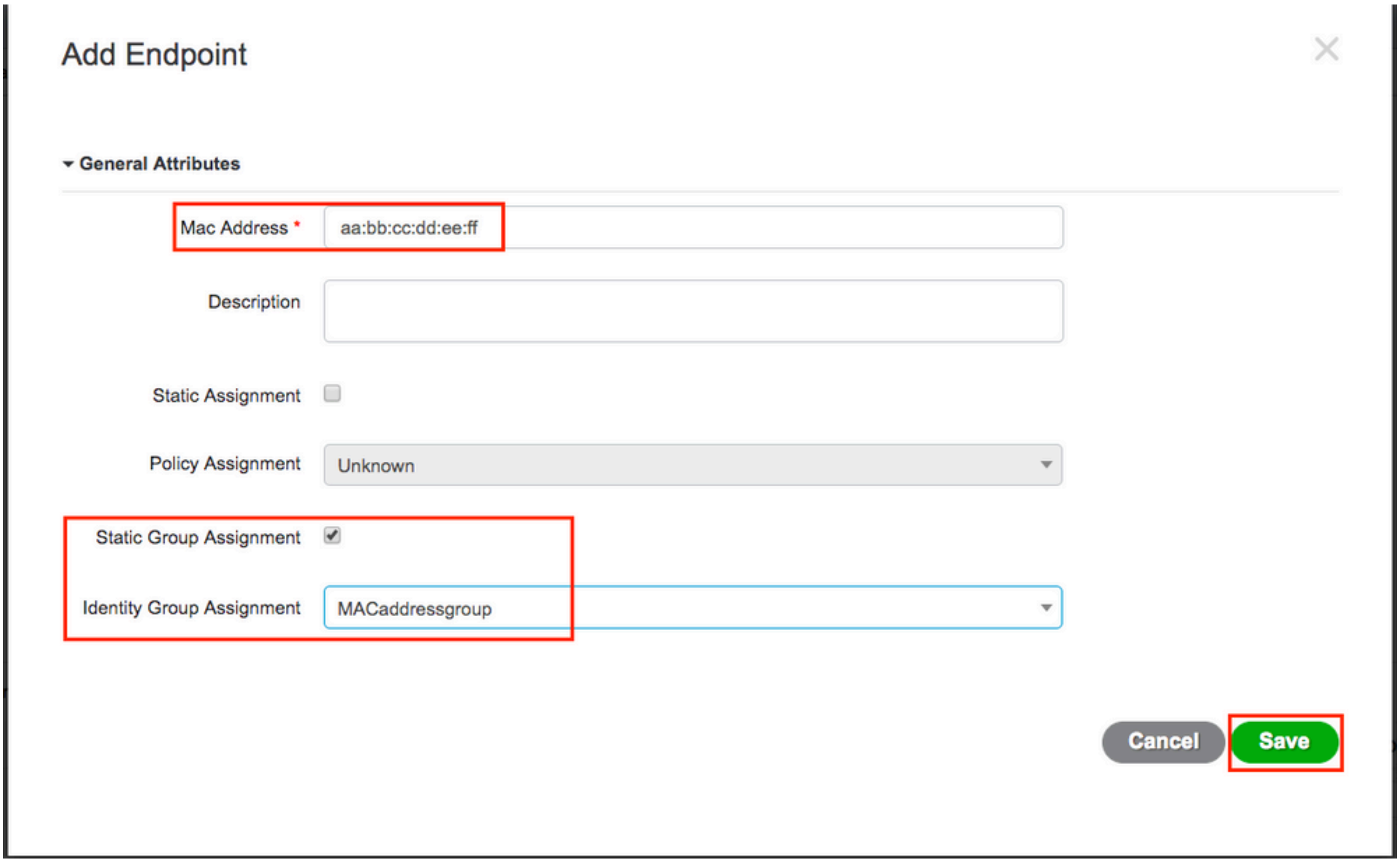




* Name MACaddressgroup

Step 2. Navigate to Work Centers > Network Access > Identities > Endpoints > +Add.





Add Endpoint

▾ General Attributes

Mac Address * aa:bb:cc:dd:ee:ff

Description

Static Assignment □

Policy Assignment Unknown

Static Group Assignment ☑

Identity Group Assignment MACaddressgroup

Cancel Save

ISE Configuration

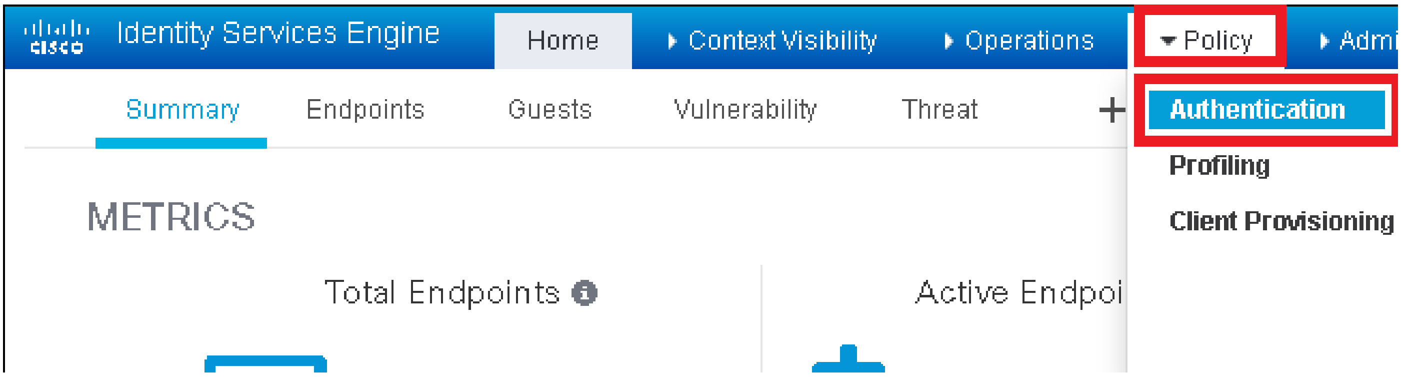
Add 9800 WLC to ISE.

Read the instructions in this link: Declare WLC to ISE.

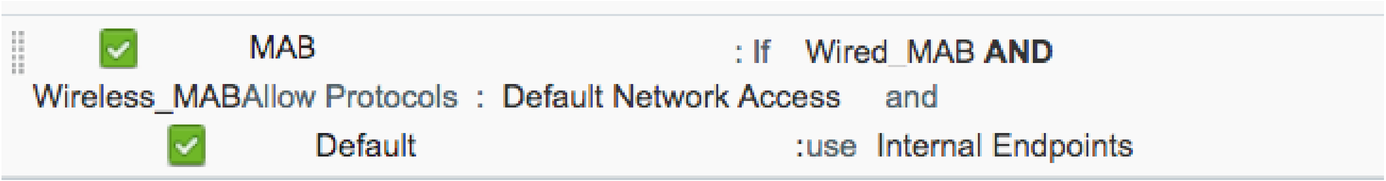
Create an Authentication Rule

Authentication rules are used to verify if the credentials of the users are right (verify if the user really is who it says it is) and limit the authentication methods that are allowed to be used by it.

Step 1. Navigate to Policy > Authentication as shown in the image.
Confirm that the default MAB rule exists on your ISE.



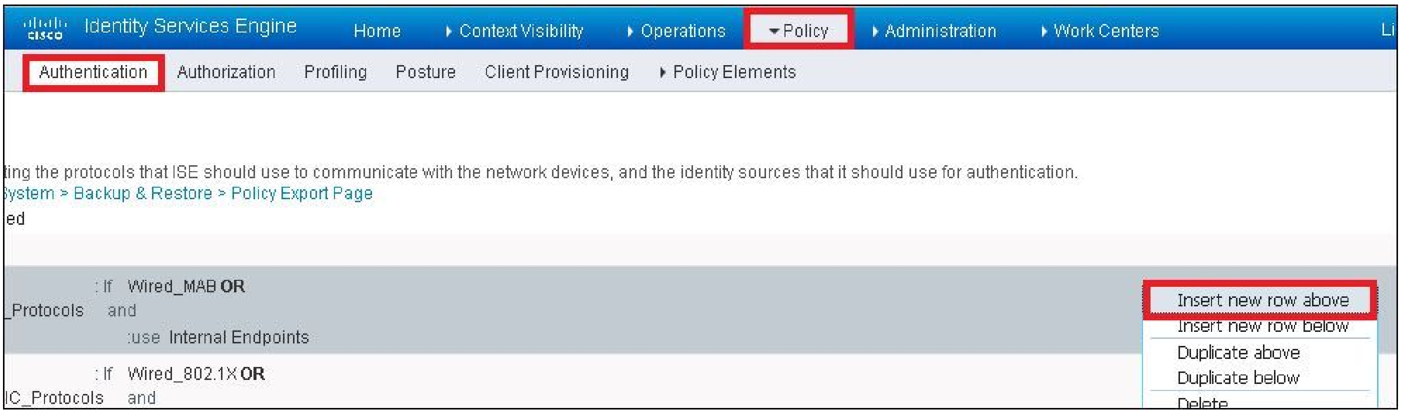
Step 2. Verify that the default authentication rule for MAB already exists:



MAB : If Wired_MAB **AND** Wireless_MABAllow Protocols : Default Network Access and

Default :use Internal Endpoints

If not, you can add a new one when you click Insert new row above.



ing the protocols that ISE should use to communicate with the network devices, and the identity sources that it should use for authentication.
ystem > Backup & Restore > Policy Export Page

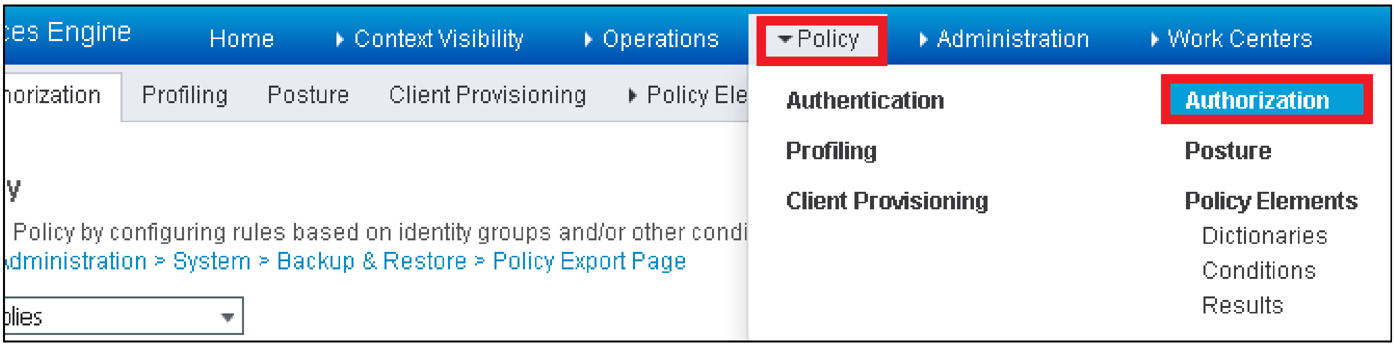
: If Wired_MAB **OR** _Protocols and :use Internal Endpoints

: If Wired_802.1X **OR** IC_Protocols and

Authorization Rule Creation

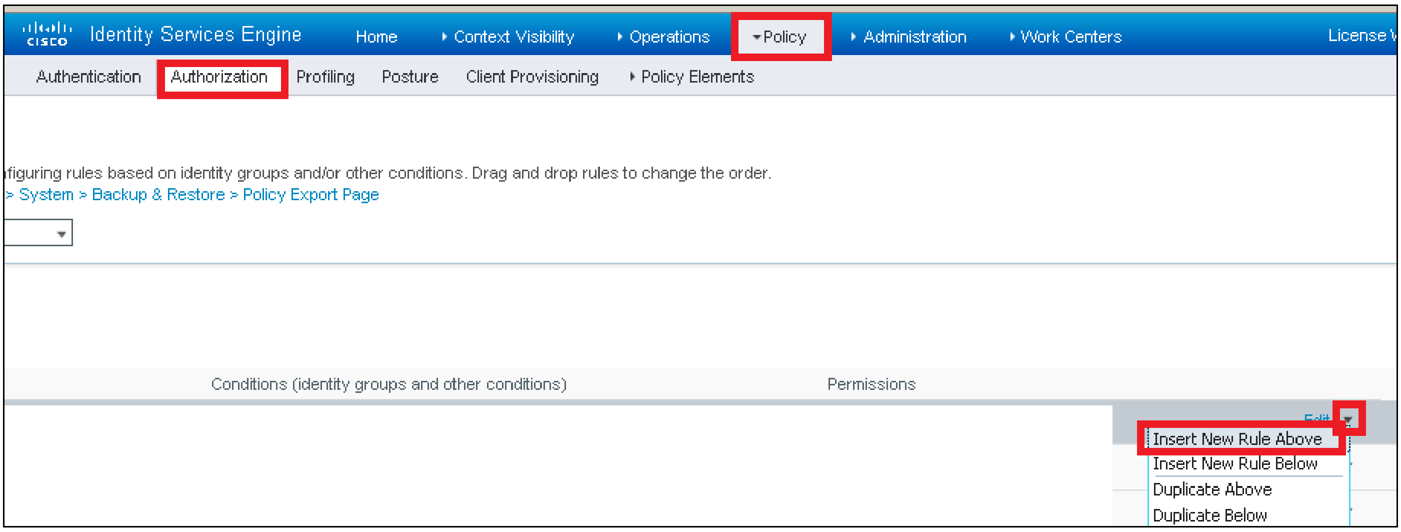
The authorization rule is the one in charge to determine which permissions (which authorization profile) result is applied to the client.

Step 1. Navigate to Policy > Authorization as shown in the image.



Policy by configuring rules based on identity groups and/or other condi
dministration > System > Backup & Restore > Policy Export Page

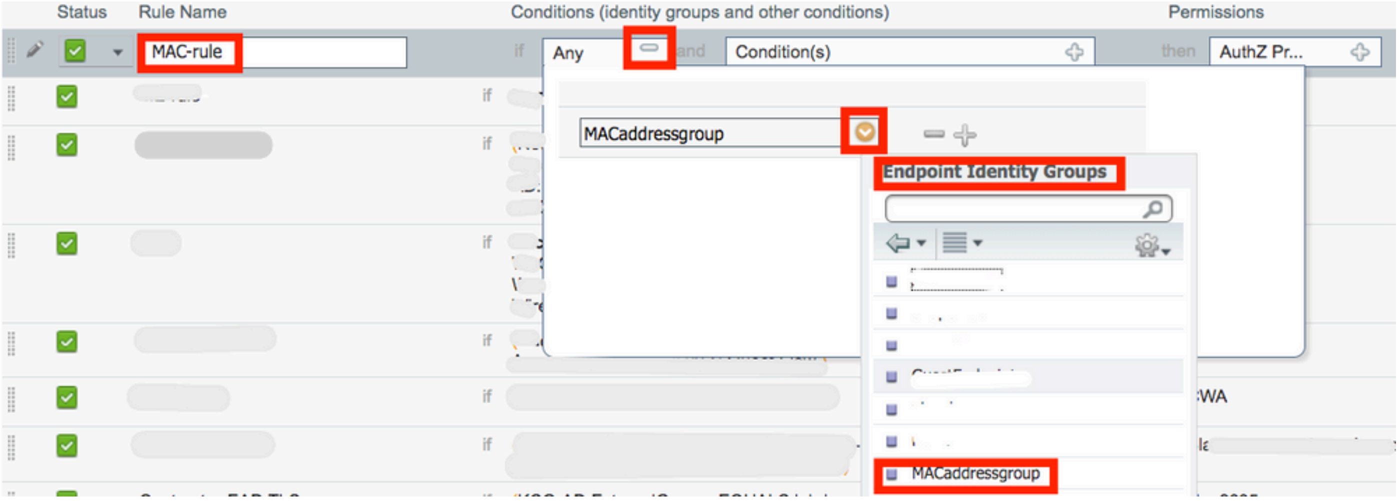
Step 2. Insert a new rule as shown in the image.



figuring rules based on identity groups and/or other conditions. Drag and drop rules to change the order.
> System > Backup & Restore > Policy Export Page

Step 3. Enter the values.

First, choose a name for the rule and the Identity group where the endpoint is stored (**MACaddressgroup**) as shown in the image.



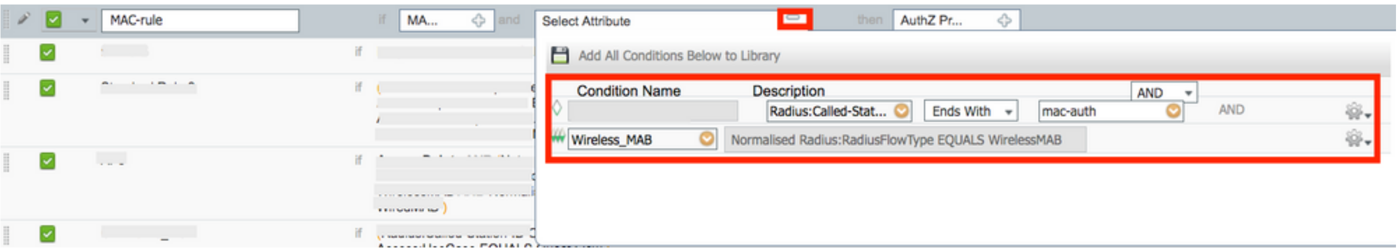
Status	Rule Name	Conditions (identity groups and other conditions)	Permissions
☑	MAC-rule	if Any and Condition(s)	then AuthZ Pr...

MACaddressgroup

Endpoint Identity Groups

MACaddressgroup

After that, choose other conditions that do the authorization process to fall into this rule. In this example, the authorization process hits this rule if it uses Wireless MAB and its called station ID (the name of the SSID) ends with mac-auth as shown in the image.

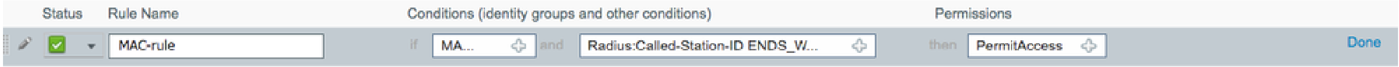


☑ MAC-rule if MA... and Select Attribute then AuthZ Pr...

Add All Conditions Below to Library

Condition Name	Description			
	Radius:Called-Stat...	Ends With	mac-auth	AND
Wireless_MAB	Normalised Radius:RadiusFlowType EQUALS WirelessMAB			

Finally, choose the Authorization profile that is assigned, in this case, PermitAccess to the clients that hit that rule. Click Done and save it.



Status	Rule Name	Conditions (identity groups and other conditions)	Permissions	
☑	MAC-rule	if MA... and Radius:Called-Station-ID ENDS_W...	then PermitAccess	Done

Verify

You can use these commands to verify the current configuration:

```
# show wlan { summary | id | name | all }
# show run wlan
# show run aaa
# show aaa servers
# show ap config general
# show ap name <ap-name> config general
# show ap tag summary
# show ap name <AP-name> tag detail
# show wlan { summary | id | name | all }
# show wireless tag policy detailed <policy-tag-name>
# show wireless profile policy detailed <policy-profile-name>
```

Troubleshoot

WLC 9800 provides ALWAYS-ON trace capabilities. This ensures all client connectivity-related errors, warnings, and notice-level messages are constantly logged and you can view logs for an incident or failure condition after it has occurred.

Note: Although it depends on the volume of logs generated, you can go back a few hours to several days.

In order to view the traces that 9800 WLC collected by default, you can connect via SSH/Telnet to the 9800 WLC and read these steps (ensure you log the session to a text file).

Step 1. Check the current time of the controller so you can track the logs from the time back to when the issue occurred.

```
# show clock
```

Step 2. Collect syslogs from the controller buffer or the external syslog as dictated by the system configuration. This provides a quick view into the health and errors of the system if any.

```
# show logging
```

Step 3. Verify if any debug conditions are enabled.

```
# show debugging
IOSXE Conditional Debug Configs:

Conditional Debug Global State: Stop

IOSXE Packet Tracing Configs:


Packet Infra debugs:

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

Note: If you see any condition listed, it means the traces are logged up to debug level for all the processes that encounter the enabled conditions (mac address, IP address, and so on). This increases the volume of logs. Therefore, it is recommended to clear all conditions when not actively debugging.

Step 4. If the MAC address under the test was not listed as a condition in Step 3., collect the always-on notice level traces for the specific mac address.

```
# show logging profile wireless filter { mac | ip } { <aaaa.bbbb.cccc> | <a.b.c.d> } to-file always-on-
```

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

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

Conditional Debugging and Radio Active Tracing

If the always-on traces do not give you enough information to determine the trigger for the problem under investigation, you can enable conditional debugging and capture Radio Active (RA) trace, which provides debug-level traces for all processes that interact with the specified condition (client mac address in this case). In order to enable conditional debugging, read these steps.

Step 5. Ensure there are no debug conditions enabled.

```
# clear platform condition all
```

Step 6. Enable the debug condition for the wireless client mac address that you want to monitor.

These commands start to monitor the provided mac address for 30 minutes (1800 seconds). You can optionally increase this time to up to 2085978494 seconds.

```
# debug wireless mac <aaaa.bbbb.cccc> {monitor-time <seconds>}
```

Note: In order to monitor more than one client at a time, run debug wireless mac <aaaa.bbbb.cccc> command per mac address.

Note: You do not see the output of the client activity on the terminal session, as everything is buffered internally to be viewed later.

Step 7. Reproduce the issue or behavior that you want to monitor.

Step 8. Stop the debugs if the issue is reproduced before the default or configured monitor time is up.

```
# no debug wireless mac <aaaa.bbbb.cccc>
```

Once the monitor time has elapsed or the debug wireless has been stopped, the 9800 WLC generates a local file with the name: ra_trace_MAC_aaaabbbbcccc_HHMMSS.XXX_timezone_DayWeek_Month_Day_year.log

Step 9. Collect the file of the mac address activity. You can either copy the ra trace .log to an external server or display the output directly on the screen.

Check the name of the RA traces file:

```
# dir bootflash: | inc ra_trace
```

Copy the file to an external server:

```
# copy bootflash:ra_trace_MAC_aaaabbbbcccc_HHMMSS.XXX_timezone_DayWeek_Month_Day_year.log tftp://a.b.c.
```

Display the content:

```
# more bootflash:ra_trace_MAC_aaaabbbbcccc_HHMMSS.XXX_timezone_DayWeek_Month_Day_year.log
```

Step 10. If the root cause is still not obvious, collect the internal logs which are a more verbose view of debug-level logs. You do not need to debug the client again as you only take a further detailed look at debug logs that have already been collected and internally stored.

```
# show logging profile wireless internal filter { mac | ip } { <aaaa.bbbb.cccc> | <a.b.c.d> } to-file r
```

Note: This command output returns traces for all logging levels for all processes and is quite voluminous. Contact Cisco TAC to help you parse through these traces.

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

Copy the file to an external server:

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

Display the content:

```
# more bootflash:ra-internal-<FILENAME>.txt
```

Step 11. Remove the debug conditions.

```
# clear platform condition all
```

Note: Ensure that you always remove the debug conditions after a troubleshooting session.

Related Information

- Cisco Technical Support & Downloads