



debug Commands

- [debug arp, on page 3](#)
- [debug ble, on page 4](#)
- [debug capwap client, on page 5](#)
- [debug capwap client avc, on page 6](#)
- [debug cdp, on page 7](#)
- [debug cleanair, on page 8](#)
- [debug dhcp, on page 9](#)
- [debug dot11, on page 10](#)
- [debug dot11 client datapath, on page 11](#)
- [debug dot11 client level, on page 12](#)
- [debug dot11 driver slot, on page 13](#)
- [debug dot11 firmware , on page 15](#)
- [debug dot11 sensor, on page 16](#)
- [debug dtls client, on page 17](#)
- [debug ethernet, on page 18](#)
- [debug flexconnect, on page 19](#)
- [debug lldp, on page 20](#)
- [debug memory, on page 21](#)
- [debug memory pool, on page 22](#)
- [debug memory pool alloc, on page 23](#)
- [debug memory pool free, on page 24](#)
- [debug mesh, on page 25](#)
- [debug mesh adjacency, on page 26](#)
- [debug mesh path-control, on page 27](#)
- [debug rrm neighbor, on page 28](#)
- [debug rrm reports, on page 29](#)
- [debug sip, on page 30](#)
- [debug wips, on page 31](#)
- [debug authentication interface, on page 32](#)
- [debug process memory, on page 33](#)
- [debug traffic, on page 34](#)
- [debug tunnel, on page 35](#)
- [debug client trace, on page 36](#)

- [no](#), on page 37
- [traceroute](#), on page 38
- [undebug](#), on page 39

debug arp

To enable debugging of ARP, use the **debug arp** command.

debug arp { **errors** | **events** | **packets** }

Syntax Description

errors Enable debugging of ARP errors

events Enable debugging of ARP events

packets Enable debugging of ARP Tx and Rx packets

Command Modes

Privileged EXEC (#)

Command History

Release Modification

8.1.111.0 This command was introduced.

Examples

The following example shows how to enable debugging of ARP errors:

```
cisco-ap# debug arp errors
```

debug ble

To enable debugging of Bluetooth Low Energy (BLE), use the **debug ble** command.

debug ble {critical | error | events | fastpath {rssi | scan | sync} | receive | transmit}

Syntax Description

critical	Enables debugging of BLE critical events
error	Enables debugging of BLE error events
events	Enables debugging of BLE events
fastpath {rssi scan sync}	Shows data exported to CMX. The following options are available: <ul style="list-style-type: none"> • RSSI data • Scan data • Sync data
receive	Enables debugging of BLE packet received from BLE radio
transmit	Enables debugging of BLE packet transmitted to BLE radio

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
8.7	This command was introduced.

Examples

The following example shows how to enable debugging of BLE critical events:

```
cisco-ap# debug ble critical
```

debug capwap client

To enable debugging of CAPWAP clients, use the **debug capwap client** command.

debug capwap client { **ble** | **detail** | **efficient-upgrade** | **error** | **events** | **flexconnect** | **info** | **keepalive** | **payload** | **pmtu** | **qos** | **reassembly** | **security** }

Syntax Description	
ble	Enables debugging of CAPWAP BLE detail
detail	Enables debugging of CAPWAP detail
efficient-upgrade	Enables debugging of image predownload
error	Enables debugging of CAPWAP error
events	Enables debugging of CAPWAP events
flexconnect	Enables debugging of CAPWAP FlexConnect mode event
info	Enables debugging of CAPWAP information
keepalive	Enables debugging of CAPWAP keepalive
payload	Enables debugging of CAPWAP payload
pmtu	Enables debugging of CAPWAP path MTU
qos	Enables debugging of CAPWAP QoS
reassembly	Enables debugging of CAPWAP reassembly
security	Enables debugging of CAPWAP security

Command Modes Privileged EXEC (#)

Command History	Release	Modification
	8.1.111.0	This command was introduced.

Examples

The following example shows how to enable debugging of CAPWAP client detail:

```
cisco-ap# debug capwap client detail
```

debug capwap client avc

To enable debugging of CAPWAP client AVC, use the **debug capwap client avc** command.

debug capwap client avc { **all** | **detail** | **error** | **event** | **info** | **netflow** { **all** | **detail** | **error** | **event** | **packet** } | **numflows** }

Syntax Description

all	Enables debugging of all CAPWAP client AVC
detail	Enables debugging of CAPWAP AVC detail
error	Enables debugging of CAPWAP AVC error
event	Enables debugging of CAPWAP AVC event
info	Enables debugging of CAPWAP AVC information
netflow	Enables debugging of CAPWAP client AVC NetFlow
netflow all	Enables debugging of all CAPWAP client AVC NetFlow
netflow detail	Enables debugging of CAPWAP client AVC NetFlow detail
netflow error	Enables debugging of CAPWAP client AVC NetFlow error
netflow event	Enables debugging of CAPWAP client AVC NetFlow event
netflow packet	Enables debugging of CAPWAP client AVC NetFlow packet
numflows	Enables debugging of CAPWAP client AVC numflows

Command Modes

Privileged EXEC (#)

Command History

Release Modification

8.1.111.0 This command was introduced.

Examples

The following example shows how to enable debugging of all CAPWAP client AVC:

```
cisco-ap# debug capwap client avc all
```

debug cdp

To enable debugging of controller discovery protocol (CDP), use the **debug cdp** command.

debug cdp { **adjacency** | **events** | **ilp** | **packets** }

Syntax Description	
adjacency	Enables debugging of CDP neighbors
events	Enables debugging of CDP events
ilp	Enables debugging of inline power
packets	Enables debugging of CDP packets

Command Modes Privileged EXEC (#)

Command History	Release	Modification
	8.1.111.0	This command was introduced.

Examples

The following example shows how to enable debugging of CDP events:

```
cisco-ap# debug cdp events
```

debug cleanair

To configure debugging of CleanAir, use the **debug cleanair** command.

debug cleanair {bringup | event | logdebuglow | major | nsi | offchan {0 | 1}}

Syntax Description

bringup	Enables debugging of CleanAir port or bringups
events	Enables debugging of normal CleanAir events
logdebug	Logs CleanAir debug output to a logfile
low	Enables debugging of hex dump of some messages
major	Enables debugging of major CleanAir events
nsi	Enables debugging of NSI messages
offchan 0 1	Enables debugging of CleanAir MSMT requests. You have to specify the radio slot as either 0 or 1

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
8.1.111.0	This command was introduced.

Examples

The following example shows how to enable debugging of major CleanAir events:

```
cisco-ap# debug cleanair major
```


debug dhcp

To configure debugging of DHCP, use the **debug dhcp** command.

```
debug dhcp {errors | events | packets}
```

Syntax Description	
errors	Enables debugging of DHCP errors
events	Enables debugging of DHCP events
packets	Enables debugging of DHCP packets

Command Modes	
	Privileged EXEC (#)

Command History	Release	Modification
	8.1.111.0	This command was introduced.

Examples

The following example shows how to enable debugging of DHCP errors:

```
cisco-ap# debug dhcp errors
```

debug dot11

To enable debugging of 802.11, use the **debug dot11** command.

debug dot11 { **critical** | **errors** | **events** | **info** }

Syntax Description	
critical	Enables 802.11 critical level debugging
errors	Enables 802.11 error level debugging
events	Enables 802.11 event level debugging
info	Enables 802.11 information level debugging

Command Modes Privileged EXEC (#)

Command History	Release Modification
	8.1.111.0 This command was introduced.

Examples

The following example shows how to enable debugging of 802.11 error level:

```
cisco-ap# debug dot11 errors
```

debug dot11 client datapath

To enable debugging of 802.11 client datapath, use the **debug dot11 client datapath** command.

```
debug dot11 client datapath {{arp | dhcp | eapol} {addr {mac-addr1 | mac-addr2 | mac-addr3
| mac-addr4} | all}} | dns-acl}
```

Syntax Description		
arp		Enables client datapath ARP debugging
dhcp		Enables client datapath DHCP debugging
eapol		Enables client datapath EAPOL debugging
{ addr all }		Option to specify MAC address of specific clients or all clients
{ <i>mac-addr1</i> <i>mac-addr2</i> <i>mac-addr3</i> <i>mac-addr4</i> }		MAC addresses of clients that you have to enter
dns-acl		Enables client datapath DNS-ACL debugging

Command Modes Privileged EXEC (#)

Command History **Release** **Modification**

8.1.111.0 This command was introduced.

Examples

The following example shows how to enable debugging of client datapath ARP:

```
cisco-ap# debug dot11 client datapath arp
```

debug dot11 client level

To enable 802.11 client debugging level, use the **debug dot11 client level** command.

```
debug dot11 client level {critical | errors | events | info} {addr {mac-addr1 | mac-addr2 |
mac-addr3 | mac-addr4} | all}
```

Syntax Description		
critical		Enables client critical level debugging
errors		Enables client error level debugging
events		Enables client event level debugging
info		Enables client information level debugging
{addr all}		Option to specify MAC address of specific clients or all clients
	<i>{mac-addr1 mac-addr2 mac-addr3 mac-addr4}</i>	MAC addresses of clients that you have to enter

Command Modes Privileged EXEC (#)

Command History	Release Modification
	8.1.111.0 This command was introduced.

Examples

The following example shows how to enable debugging of all clients at the event level:

```
cisco-ap# debug dot11 client level events all
```

debug dot11 driver slot

To enable debugging of 802.11 drivers, use the **debug dot11 driver slot** command.

```
debug dot11 driver slot {0 | 1} {all | {cac {info | metrics}} | chd | save-acnt-data |
save-on-failure [extended] | stop-on-failure | tsm | vim | type { all | assoc | auth |
dhcp | eap | icmp | probe } mac-addr1 | mac-addr2 | mac-addr3 | mac-addr4
```

Syntax Description

slot {0 1}	Enables 802.11 driver debugs per radio
all	Enables all 802.11 driver debugs
cac	Enables 802.11 CAC debugs
cac info	Enables 802.11 CAC info level debugs
cac metrics	Enables debugging of 802.11 CAC metrics
chd	Enables 802.11 CHD debugs
save-acnt-data	Saves the radio accounting data
save-on-failure	Saves the radio crash information upon radio failure
save-on-failure extended	Saves extended information on radio failure
stop-on-failure	Stops the AP from reboot on radio failure
tsm	Enables 802.11 traffic stream metric debugs
vim	Enables 802.11 video metric debugs
type	Enables the debug types.
all	Enables the all type debugging.
assoc	Enables the assoc debugging.
auth	Enables the auth debugging.
dhcp	Enables the dhcp debugging.
eap	Enables the eap debugging.
icmp	Enables the icmp debugging.
probe	Enables the probe debugging.
<i>mac-addr</i>	MAC addresses of the clients. You can enter upto four MAC addresses.

Command Modes

Privileged EXEC (#)

Command History	Release	Modification
	8.1.111.0	This command was introduced.
	8.5.140.0 and 8.8	This command was enhanced by adding the type parameter.

Examples

The following example shows how to enable debugging of CAC at the information level:

```
cisco-ap# debug dot11 driver slot cac info
```

debug dot11 firmware

To debug the 802.11 firmware, use the **debug dot11 firmware** command.

```
debug dot11 firmware slot slot_ID level { all | critical | emergency | error | info }
addressmac-addr1 | mac-addr2 | mac-addr3 | mac-addr4
```

Syntax Description

<i>slot_ID</i>	Enables 802.11 driver debugs per radio
all	Enables all the debug levels.
critical	Enables critical level debugs.
emergency	Enables emergency level debugs.
error	Enables error level debugs.
info	Enables info level debugs.
address	To add client address for driver/firmware debugging.
<i>mac-addr</i>	MAC addresses of the clients. You can enter upto four MAC addresses.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
8.5.140.0 and 8.8	This command was introduced.

Example

The following example shows how to enable debugging of 802.11 emergency level:

```
cisco-wave2-ap# debug dot11 firmware slot 1 emergency address 92:FB:D6:B3:7A:6C
```

debug dot11 sensor

To enable debugging of 802.11 sensors, use the **debug dot11 sensor** command.

```
debug dot11 sensor { dns | file-transfer | mail-server | ping | radius | ssh | telnet | web-server }
```

Syntax Description

dns	Enables debugging of 802.11 sensor DNS
file-transfer	Enables debugging of 802.11 sensor file transfer
mail-server	Enables debugging of 802.11 sensor mail server
ping	Enables debugging of 802.11 sensor ping
radius	Enables debugging of 802.11 sensor radius
ssh	Enables debugging of 802.11 sensor SSH
telnet	Enables debugging of 802.11 sensor Telnet.
web-server	Enables debugging of 802.11 sensor web server

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
8.1.111.0	This command was introduced.

Examples

The following example shows how to enable debugging of 802.11 sensor file transfer:

```
cisco-ap# debug dot11 sensor file-transfer
```


debug dtls client

To configure DTLS client error and event debugging, use the **debug dtls client** command.

```
debug dtls client {error | event [detail]}
```

Syntax Description	error	Configures debugging of DTLS client errors
	event [detail]	Configures debugging of DTLS client events

Command Modes	Privileged EXEC (#)
----------------------	---------------------

Command History	Release	Modification
	8.1.111.0	This command was introduced.

Examples

The following example shows how to enable debugging of DTLS client events:

```
cisco-ap# debug dtls client event
```

debug ethernet

To configure Ethernet debugging, use the **debug ethernet** command.

debug ethernet *interface-number* { **both** | **rcv** | **xmt** }

Syntax Description

<i>interface-number</i>	Interface number that you have to enter as either 0 or 1
both	Enables debugging of both transmission and reception
rcv	Enables debugging of reception
xmt	Enables debugging of transmission

Command Modes

Privileged EXEC (#)

Command History

Release Modification

8.1.111.0 This command was introduced.

Examples

The following example shows how to enable debugging of transmission for interface 0:

```
cisco-ap# debug ethernet 0 xmt
```

debug flexconnect

To debug FlexConnect features, use the **debug flexconnect** command.

```
debug flexconnect {acl | cckm | dot11r | event | multicast {igmp | traffic} | pmk | proxy-arp | vsa | wlan-vlan | wsastats}
```

Syntax	Description
acl	Configures debugging of FlexConnect ACL
cckm	Configures debugging of CCKM
dot11r	Configures debugging of 802.11r
event	Configures debugging of wireless control protocol (WCP) events
multicast igmp	Configures debugging of Multicast IGMP
multicast traffic	Configures debugging of Multicast traffic
pmk	Configures debugging of opportunistic key caching (OKC) or pairwise master key caching
vsa	Configures debugging of AAA vendor specific attributes (VSA)
wlan-vlan	Configures debugging of WLAN-VLAN mapping
wsastats	Configures debugging of RADIUS or DHCP wireless service assurance statistics

Command Modes Privileged EXEC (#)

Command History	Release	Modification
	8.1.111.0	This command was introduced.

Examples

The following example shows how to enable debugging of FlexConnect ACL:

```
cisco-ap# debug flexconnect acl
```

debug lldp

To debug LLDP, use the **debug lldp** command.

debug lldp { **errors** | **events** | **packet** }

Syntax Description	
errors	Debugs LLDP errors
events	Debugs LLDP events
packet	Debugs LLDP packets

Command Modes	
	Privileged EXEC (#)

Command History	Release	Modification
	8.1.111.0	This command was introduced.

Examples

The following example shows how to enable debugging of LLDP errors:

```
cisco-ap# debug lldp errors
```

debug memory

To debug memory, use the **debug memory** command.

```
debug memory {clear | save}
```

Syntax Description

clear Removes memory debug upon boot-up

save Saves current debug level and applies it upon following boots

Command Modes

Privileged EXEC (#)

Command History

Release Modification

8.1.111.0 This command was introduced.

Examples

The following example shows how to remove memory debug upon boot-up:

```
cisco-ap# debug memory clear
```

debug memory pool

To debug memory pool, use the **debug memory pool** command.

```
debug memory pool { diff | realtime interval 1-1000000-seconds | start }
```

Syntax Description		
diff		Shows memory pool debug difference in detail
realtime interval <i>1-1000000-seconds</i>		Configures realtime interval for the memory pool
start		Starts the debug for the memory pool

Command Modes	
	Privileged EXEC (#)

Command History	Release	Modification
	8.1.111.0	This command was introduced.

Examples

The following example shows how to configure realtime interval of 180 seconds for the memory pool:

```
cisco-ap# debug memory pool realtime interval 180
```

debug memory pool alloc

To debug memory pool allocation calls, use the **debug memory pool alloc** command.

```
debug memory pool alloc {all | name pool-name} {diff | realtime interval 1-1000000-seconds | start}
```

Syntax Description		
all		Configures debug for all memory pool allocation calls
name <i>pool-name</i>		Configures debug for a specific memory pool's allocation call
diff		Shows memory pool debug allocation call difference in detail
realtime interval <i>1-1000000-seconds</i>		Configures realtime interval for the memory pool allocation calls
start		Starts the debug for the memory pool allocation calls

Command Modes Privileged EXEC (#)

Command History	Release	Modification
	8.1.111.0	This command was introduced.

Examples

The following example shows how to configure the start of the debug for all memory pool allocation calls:

```
cisco-ap# debug memory pool alloc all start
```

debug memory pool free

To debug memory pool free calls, use the **debug memory pool free** command.

```
debug memory pool free {all | name pool-name} {diff | realtime interval 1-1000000-seconds | start}
```

Syntax Description		
all		Configures debug for all memory pool free calls
name <i>pool-name</i>		Configures debug for a specific memory pool's free call
diff		Shows memory pool debug free call difference in detail
realtime interval <i>1-1000000-seconds</i>		Configures realtime interval for the memory pool free calls
start		Starts the debug for the memory pool free calls

Command Modes Privileged EXEC (#)

Command History	Release	Modification
	8.1.111.0	This command was introduced.

Examples

The following example shows how to configure the start of the debugging of all memory pool free calls:

```
cisco-ap# debug memory pool free all start
```


debug mesh

To configure debugging of mesh networks, use the **debug mesh** command.

debug mesh {**channel** | **clear** | **convergence** | **events** | **forward-mcast** | **forward-packet** | **forward-table** | **linktest** | **path-control** | **port-control** | **security** | **trace**}

Syntax Description	channel	Configures debugging of mesh channel
	clear	Resets all mesh debugs
	convergence	Configures debugging of mesh convergence
	events	Configures debugging of mesh events
	forward-mcast	Configures debugging of mesh forwarding Multicast
	forward-packet	Configures debugging of mesh forwarding packets
	forward-table	Configures debugging of mesh forwarding table
	linktest	Configures debugging of mesh linktest
	port-control	Configures debugging of mesh port control
	security	Configures debugging of mesh security
	trace	Configures debugging of mesh trace

Command Modes Privileged EXEC (#)

Command History	Release	Modification
	8.1.111.0	This command was introduced.

Examples

The following example shows how to enable debugging of mesh channel:

```
cisco-ap# debug mesh channel
```

debug mesh adjacency

To debug mesh adjacency, use the **debug mesh adjacency** command.

debug mesh adjacency { **child** | **clear** | **dfs** | **message** | **packet** | **parent** }

Syntax Description	
adjacency	Debug mesh adjacency
child	Debug mesh adjacency child
clear	Debug clear mesh adjacency
dfs	Debug mesh DFS
message	Debug mesh adjacency messages
packet	Debug mesh adjacency packet
parent	Debug mesh adjacency parent

Command Modes Privileged EXEC (#)

Command History	Release Modification
	8.1.111.0 This command was introduced.

Examples

The following example shows how to enable debugging of mesh adjacency parent:

```
cisco-ap# debug mesh adjacency parent
```

debug mesh path-control

To configure debugging of mesh path control, use the **debug mesh path-control** command.

debug mesh path-control {**error** | **events** | **packets** }

Syntax Description	
error	Configures debugging of mesh path control errors
events	Configures debugging of mesh path control events
packets	Configures debugging of mesh path control packets

Command Modes Privileged EXEC (#)

Command History	Release	Modification
	8.1.111.0	This command was introduced.

Examples

The following example shows how to enable debugging of mesh path control errors:

```
cisco-ap# debug mesh path-control error
```

debug rrm neighbor

To enable RRM neighbor debugging, use the **debug rrm neighbor** command.

debug rrm neighbor {**tx** | **rx** | **detail**}

Syntax Description	
tx	Enable RRM neighbor Tx debugging
rx	Enable RRM neighbor Rx debugging
detail	Enable RRM neighbor detail debugging

Command Modes Privileged EXEC (#)

Command History	Release	Modification
	8.1.111.0	This command was introduced.

Examples

The following example shows how to enable debugging of RRM neighbor transmissions:

```
cisco-ap# debug rrm neighbor tx
```

debug rrm reports

To enable RRM reports debugging, use the **debug rrm reports** command.

debug rrm reports

Syntax Description	reports Enables RRM report debugging
Command Modes	Privileged EXEC (#)
Command History	Release Modification 8.1.111.0 This command was introduced.

Examples

The following example shows how to enable debugging of RRM reports:

```
cisco-ap# debug rrm reports
```

debug sip

To enable session initiation protocol (SIP) debugging, use the **debug sip** command.

debug sip { **all** | **tx** | **rx** }

Syntax Description	all Enabling SIP transmission and reception debugging
	tx Enabling SIP transmission debugging
	rx Enabling SIP reception debugging
Command Modes	Privileged EXEC (#)
Command History	Release Modification
	8.1.111.0 This command was introduced.

Examples

The following example shows how to enable debugging of SIP transmissions and reception:

```
cisco-ap# debug sip all
```

debug wips

To enable wIPS debugging, use the **debug wips** command.

```
debug wips {errors | events | critical}
```

Syntax Description

errors Enable wIPS error level debugging

events Enable wIPS event level debugging

critical Enable wIPS critical level debugging

Command Modes

Privileged EXEC (#)

Command History

Release Modification

8.1.111.0 This command was introduced.

Examples

The following example shows how to enable wIPS error level debugging:

```
cisco-ap# debug wips errors
```

debug authentication interface

To enable FlexConnect radio interface debugging, use the **debug authentication interface** command.

debug authentication interface *interface-name* { **all** | **dot11** | **dot1x** | **driver** | **others** | **radius** | **wpa** }

Syntax Description

<i>interface-name</i>	Name of the interface to debug
all	Enable all parts debugging
dot11	Enable 802.11 module debugging
dot1x	Enable 802.1x module debugging
driver	Enable driver module debugging
others	Enable other non-module parts debugging
radius	Enable RADIUS module debugging
wpa	Enable WPA module debugging

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
8.1.111.0	This command was introduced.

Examples

The following example shows how to enable debugging of WPA module:

```
cisco-ap# debug authentication interface management wpa
```


debug process memory

To process memory debugging, use the **debug process memory** command.

```
debug process memory {diff | realtime [interval interval-in-seconds ] | start}
```

Syntax Description	
diff	Process memory debug show diff
realtime	Process memory real time debug
<i>interval</i>	Update interval; valid range 1 to 1000000 seconds
start	Process memory debug start

Command Modes Privileged EXEC (#)

Command History	Release	Modification
	8.1.111.0	This command was introduced.

Examples

The following example shows how to enable the start of debugging of process memory:

```
cisco-ap# debug process memory start
```

debug traffic

To enable traffic debugging, use the **debug traffic** command.

```
debug traffic {host {icmpv6 | ip | ipv6 | tcp | udp { verbose}} | wired {ip | tcp | udp {verbose}}}
```

Syntax Description

host	Enabling host traffic debugging
wired	Enabling wired traffic debugging
verbose	Display verbose output
icmpv6	Enabling host ICMPv6 traffic dump
ip	Enabling host IP traffic dump
ipv6	Enabling host IPv6 traffic dump
tcp	Enabling TCP traffic dump
udp	Enabling UDP traffic dump

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
8.1.111.0	This command was introduced.

Examples

The following example shows how to enable debugging of host IP traffic dump:

```
cisco-ap# debug traffic host ip
```

debug tunnel

To configure debugging of tunnel, use the **debug tunnel** command.

debug tunnel eogre

Syntax Description	eogre Configures debugging of EoGRE tunnel
---------------------------	---

Command Modes	Privileged EXEC (#)
----------------------	---------------------

Command History	Release	Modification
	8.1.111.0	This command was introduced.

Examples

The following example shows how to enable debugging of EoGRE tunnel:

```
cisco-ap# debug tunnel eogre
```

debug client trace

To enable client trace debugging, use the **debug client trace** command.

```
debug client trace {all | address mac-address | enable | filter {assoc | auth | dhcp | eap | icmp | mgmt | probe | proto}}
```

Syntax Description		
	all	Configure all clients tracing
	address	Configure address(es) to trace
	<i>mac-address</i>	MAC address to trace
	enable	Enable tracing
	filter	Configure trace filter
	assoc	Trace Association packets
	auth	Trace Authentication packets
	dhcp	Trace DHCP packets
	eap	Trace EAP packets
	icmp	Trace ICMP packets
	mgmt	Trace probe, assoc, auth, EAP packets
	probe	Trace probe packets
	proto	Trace DHCP, ICMP packets

Command Modes Privileged EXEC (#)

Command History	Release	Modification
	8.1.111.0	This command was introduced.

Examples

The following example shows how to enable tracing of all clients:

```
cisco-ap# debug client trace all
```

no

To negate a command or set to its defaults, use the **no** command.

no

Command Modes	Privileged EXEC (#)
----------------------	---------------------

Command History	Release Modification
------------------------	------------------------------------

8.1.111.0	This command was introduced.
-----------	------------------------------

To negate a command or set to its defaults, use this command:

```
cisco-ap# no debug
```

tracert

To view the routes followed by packets traveling in the network, use the **tracert** command.

tracert *destination-address*

Syntax Description	<i>destination-address</i> IP address of the destination of the packets				
Command Modes	Privileged EXEC (#)				
Command History	<table border="1"> <thead> <tr> <th>Release</th> <th>Modification</th> </tr> </thead> <tbody> <tr> <td>8.1.111.0</td> <td>This command was introduced.</td> </tr> </tbody> </table>	Release	Modification	8.1.111.0	This command was introduced.
Release	Modification				
8.1.111.0	This command was introduced.				

Examples

The following example shows how to view the routes followed by packets traveling in the network, with a destination IP address specified:

```
cisco-ap# tracert 209.165.200.224
```

undebug

To disable debugging on the access point, use the **undebug** command.

undebug [all]

Syntax Description	a Disables all debugging messages.
---------------------------	---

Command Modes	Privileged EXEC (#)
----------------------	---------------------

Command History	Release	Modification
	8.1.111.0	This command was introduced.

Examples

The following example shows how to disable all debugging messages:

```
cisco-ap# undebug all
```

