Debugging on Cisco Wireless Controllers

Understanding Debug Client on Wireless Controllers

Use the Wireless Debug Analyzer tool to analyze the debug client output.

Deauthenticating Clients

Using the controller, you can deauthenticate clients based on their user name, IP address, or MAC address. If there are multiple client sessions with the same user name, you can deauthenticate all the client sessions based on the user name. If there are overlapped IP addresses across different interfaces, you can use the MAC address to deauthenticate the clients.

This section contains the following subsections:

Deauthenticating Clients (GUI)

Procedure

Step 1 Choose **Monitor** > **Clients**.

Step 2 On the **Clients** page, click the MAC address of the client.

Step 3 On the **Clients > Detail** page displayed, click **Remove**.

Step 4 Save the configuration.

Deauthenticating Clients (CLI)

Procedure

- **config client deauthenticate** {*mac-addr* | *ipv4-addr* | *ipv6-addr* | *user-name*}

Using the CLI to Troubleshoot Problems

If you experience any problems with your controller, you can use the commands in this section to gather information and debug issues.

- The **debug** command enables diagnostic logging of specific events. The log output is directed to the terminal session in which the debug command is entered.
- Only one debug session at a time is active. If one terminal has debugging running, and a **debug** command is entered on another terminal, the debug session on the first terminal is terminated.
- To turn off all debugs, use the **debug disable-all** command.
- To filter the debugs based on client or AP MAC addresses, use the **debug mac addr** *mac-address* command. Up to 10 MAC addresses are supported.

Procedure

- **show process cpu**: Shows how various tasks in the system are using the CPU at that instant in time. This command is helpful in understanding if any single task is monopolizing the CPU and preventing other tasks from being performed.

  The Priority field shows two values: 1) the original priority of the task that was created by the actual function call and 2) the priority of the task that is divided by a range of system priorities.

  The CPU Use field shows the CPU usage of a particular task.

  The Reaper field shows three values: 1) the amount of time for which the task is scheduled in user mode operation, 2) the amount of time for which the task is scheduled in a system mode operation, and 3) whether the task is being watched by the reaper task monitor (indicated by a “T”). If the task is being watched by the reaper task monitor, this field also shows the timeout value (in seconds) before which the task needs to alert the task monitor.

Note If you want to see the total CPU usage as a percentage, enter the **show cpu** command.

- **show process memory**: Shows the allocation and deallocation of memory from various processes in the system at that instant in time.

  In the example above, the following fields provide information:

  The Name field shows the tasks that the CPU is to perform.

  The Priority field shows two values: 1) the original priority of the task that was created by the actual function call and 2) the priority of the task that is divided by a range of system priorities.

  The BytesInUse field shows the actual number of bytes used by dynamic memory allocation for a particular task.

The BlocksInUse field shows the chunks of memory that are assigned to perform a particular task.

The Reaper field shows three values: 1) the amount of time for which the task is scheduled in user mode operation, 2) the amount of time for which the task is scheduled in system mode operation, and 3) whether the task is being watched by the reaper task monitor (indicated by a "T"). If the task is being watched by the reaper task monitor, this field also shows the timeout value (in seconds) before which the task needs to alert the task monitor.

- **show tech-support**: Shows an array of information that is related to the state of the system, including the current configuration, last crash file, CPU utilization, and memory utilization.
- **show run-config**: Shows the complete configuration of the controller. To exclude access point configuration settings, use the **show run-config no-ap** command.

Note If you want to see the passwords in clear text, enter the **config passwd-cleartext enable command**. To execute this command, you must enter an admin password. This command is valid only for this particular session. It is not saved following a reboot.

- **show run-config commands**: Shows the list of configured commands on the controller. This command shows only values that you configured. It does not show system-configured default values.

Potential Reasons for Controller Reset

This section lists all the potential reasons for a controller reset.

- User initiated reset
- Hard/Unknown reboot
- Reset due to switch-driver crash
- Reset due to DP crash
- Peer-RMI, peer-RP and management default gateway are reachable
- Both controllers are active, same timestamp, rebooting secondary controller
- Mandatory argument is missing for starting redundancy manager transport task
- Failed to create socket to communicate with peer
- Failed to create socket to communicate with peer via secondary link
- Failed bind socket to communicate with peer
- Failed bind socket to communicate with peer via secondary link
- License count was not received from primary controller
- Not reaching Hot Standby
- Standby has not received config files from Active
- Corrupted XMLs transferred from Active to Standby
- Corrupted XMLs in Active controller

- Standby TFTP failure
- New XML downloaded
- Active to Standby request
- Standby IPC failure
- Certificate installed in Standby controller
- Mandatory argument to start redundancy manager ping task is missing
- Self sanity check failed; both controllers are in maintenance state
- Self sanity check failed; in maintenance state because both controllers were active
- Self sanity check failed; current controller became Active before peer reboot
- User has initiated reset
- XML transfer was initiated but role negotiation was not done
- IPC timeout has occurred multiple times
- Role notification timeout has occurred
- Peer sanity check failed
- Active is down, Standby is not ready to take over
- Configuration out of sync
- Configuration download failure
- None of the ports is connected
- None of the local ports is connected
- Peer maintenance mode
- RF keepalive timeout
- Peer notification timeout
- Peer platform sync timeout
- Peer progression failed
- Standby default gateway is not reachable
- Active default gateway is not reachable
- Redundancy management interface and redundancy port are down
- Redundancy port is down
- Redundancy management interface is down
- Standby timeout
- Active timeout
- License count was not received from Primary controller

- XMLs were not transferred from Active to Standby
- Certificate transfer from Active to Standby failed
- Redundant pair assume same role
- Failed to create redundancy manager semaphore
- Failed to create redundancy manager keepalive task
- Failed to create redundancy manager main task
- Failed to create redundancy manager message queue
- Failed to start redundancy manager transport task
- Controller is not in proper state for more than expected time
- Mandatory argument is missing in redundancy manager main task
- Failed to create timer to send sanity messages
- Failed to create timer to send role negotiation message
- Failed to create timer to send the messages to peer
- Failed to create timer for handling max role negotiation time
- Mandatory argument to start keepalive task is missing
- Failed to create the semaphore used for sending keepalive messages
- Reset due to config download
- Watchdog reset
- Unknown reset reason