APPENDIX A

Troubleshooting and Best Practices

This appendix identifies and explains any additional troubleshooting or best practices you may find necessary as you implement a particular function.

This appendix includes the following sections:

- Troubleshooting Cisco Compatible Extensions Version 5 Client Devices, page A-1
- Web Auth Security on WLANs, page A-3

Troubleshooting Cisco Compatible Extensions Version 5 Client Devices

Two features are designed to troubleshoot communication problems with Cisco Compatible Extension clients: diagnostic channel and client reporting.

Note These features are supported only on Cisco Compatible Extensions Version 5 Client Devices. They are not support for use with non-Cisco Compatible Extensions Version 5 Client Devices or with clients running an earlier version.

Diagnostic Channel

The diagnostic channel feature enables you to troubleshoot problems regarding client communication with a WLAN. When initiated by a client having difficulties, the diagnostic channel is a WLAN configured to provide the most robust communication methods with the fewest obstacles to communication placed in the path of the client. The client and access points can be put through a defined set of tests in an attempt to identify the cause of communication difficulties experienced by the client.

Note Only one WLAN per controller can have the diagnostic channel enabled, and all of the security on this WLAN is disabled.

Configuring the Diagnostic Channel

Follow these steps to configure the diagnostic channel.

Step 1 Choose **Configure > Controllers**.

Step 2 Click an IP address to choose a specific controller.

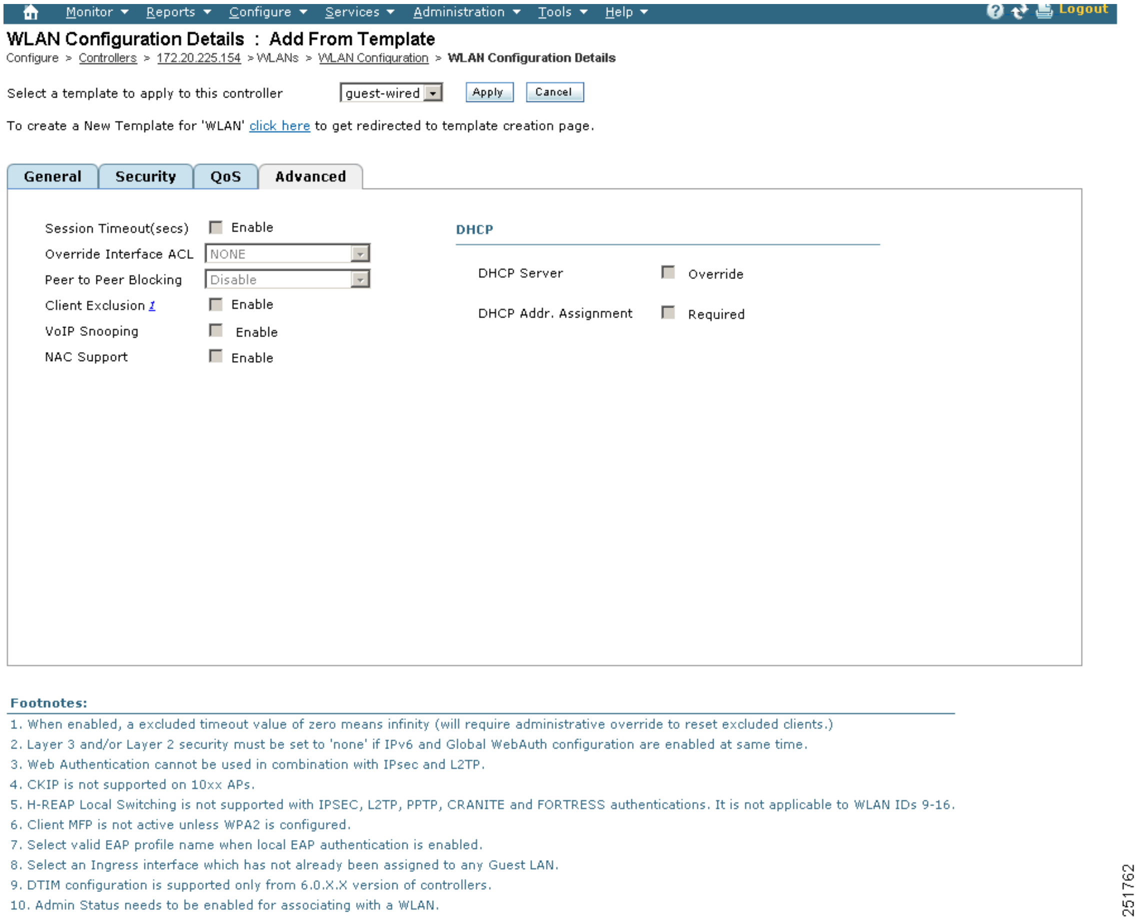
Step 3 Choose **WLAN> WLAN Configuration** from the left sidebar menu.

Step 4 Choose **Add a WLAN** from the Select a command drop-down menu to create a new or click the profile name of an existing.

Note Cisco recommends that you create a new WLAN on which to run the diagnostic tests.

Step 5 When the WLANs page appears, click the **Advanced** tab (see Figure A-1).

Figure A-1 WLANs Advanced Tab



WLAN Configuration Details : Add From Template

Configure > Controllers > 172.20.225.154 > WLANs > WLAN Configuration > WLAN Configuration Details

Select a template to apply to this controller: guest-wired

To create a New Template for 'WLAN' click here to get redirected to template creation page.

Session Timeout(secs) □ Enable

Override Interface ACL: NONE

Peer to Peer Blocking: Disable

Client Exclusion 1 □ Enable

VoIP Snooping □ Enable

NAC Support □ Enable

DHCP

DHCP Server □ Override

DHCP Addr. Assignment □ Required

Footnotes:

1. When enabled, a excluded timeout value of zero means infinity (will require administrative override to reset excluded clients.)
2. Layer 3 and/or Layer 2 security must be set to 'none' if IPv6 and Global WebAuth configuration are enabled at same time.
3. Web Authentication cannot be used in combination with IPsec and L2TP.
4. CKIP is not supported on 10xx APs.
5. H-REAP Local Switching is not supported with IPSEC, L2TP, PPTP, CRANITE and FORTRESS authentications. It is not applicable to WLAN IDs 9-16.
6. Client MFP is not active unless WPA2 is configured.
7. Select valid EAP profile name when local EAP authentication is enabled.
8. Select an Ingress interface which has not already been assigned to any Guest LAN.
9. DTIM configuration is supported only from 6.0.X.X version of controllers.
10. Admin Status needs to be enabled for associating with a WLAN.

Step 6 If you want to enable diagnostic channel troubleshooting on this WLAN, select the **Diagnostic Channel** check box. Otherwise, leave this check box unselected, which is the default value.

Step 7 Click **Save** to commit your changes.

Web Auth Security on WLANs

This section describes the troubleshooting and best practices procedures that are useful when implementing web auth security on WLANs.

Web-auth is a Layer 3 security feature which allows web-based authentication to users on a WLAN. It is used mainly in guest networking scenarios, although not restricted to that usage.

When a WLAN is configured with web-auth security, you are redirected to the login page after passing Layer 2 authentications (static WEP, WPA+PSK, MAC filtering, and so on). The login page is stored on the local device or an external web server, and the page can be modified to allow a customized logo, title, and so on.

After the WLAN is configured with a web-auth WLAN, the HTTP *get* request is sent by the wireless client to the requested website. The controller firewall allows the DNS resolution of the specified URL. After the resolution, the controller interrupts the HTTP packets from the wireless client and redirects to the login page. When the credentials are entered on the login page and submitted, they are authenticated against the local database. If the user is not found in the local database, the configured RADIUS servers are contacted.

Note PAP and CHAP authentication are used between the client and authentication agent. Make sure your RADIUS server supports both of these protocols so web-auth login is allowed.

Upon successful authentication, you are allowed to pass traffic. After three unsuccessful authentication attempts, the client is excluded. This excluded client cannot associate until the exclusion timeout limit is surpassed. The exclusion timeout limit is configured with aggressive load balancing, which actively balances the load between the mobile clients and their associated access points.

Web-auth WLAN is also configured with a pre-authentication access control list (ACL). This ACL is configured the same as a normal ACL but permits access to resources that the client needs prior to authentication. An administrator must use the interface section to apply an ACL to the client after authentication.

A web-auth WLAN can be configured with a session timeout value. This value defines the time the client needs to re-authenticate with the device. If the value is set to zero, which means infinity, the client never re-authenticates unless the logged out option is used. You can access the logout URL at http://<VirtualIP>/logout.html.

Note Disable all pop-up blockers on the client to see the logout page.

Web-auth can be configured in different modes under Layer 3 security. The most commonly used modes of web-auth are as follows:

- Internal Web—Redirection to an internal page using http://<virtual IP /DNS name >/login.html. Customization is available.
- External Web—Redirection to an external URL.

Debug Commands

The following debug commands are allowed:

debug client <client-mac-address>

debug pm ssh-tcp enable

debug pm ssh-appgw enable

debug pm rules enable

debug pm config enable

show client detail <client-mac-address>

debug pem event enable

Debug Strategy

Use the following strategy for web-auth configured on a WLAN without guest tunneling.

Step 1 Identify a mobile client to work with and write down its wireless MAC address. Use the command prompt > ipconfig /all for all MS Windows-based systems.

Step 2 Disable the mobile client's radio.

Step 3 Enter the following debug commands via a serial console set for high speed (115200) or SSH session to the controller's management port:

```
debug client <client-mac-address>
debug pm ssh-tcp enable
debug pm ssh-appgw enable
debug pm rules enable
debug pm config enable

show client detail <client-mac-address>

debug pem event enable
debug pem state enable
```

Step 4 Enable the radio and let the client associate. After the client is associated, enter the **show client detail** *client-mac-address* command.

```
Client Username ................................. N/A
AP MAC Address................................... 00:0b:85:09:96:10
Client State..................................... Associated
Wireless LAN Id.................................. 1
BSSID............................................ 00:0b:85:09:96:1f
Channel.......................................... 11
IP Address....................................... 10.50.234.3
Association Id................................... 1
Authentication Algorithm......................... Open System
Reason Code...................................... 0
Status Code...................................... 0
Session Timeout.................................. 0
Client CCX version............................... 3
Mirroring........................................ Disabled
QoS Level........................................ Silver
Diff Serv Code Point (DSCP)...................... disabled
```

```
802.1P Priority Tag.............................. disabled
WMM Support...................................... Disabled
Mobility State................................... Local
Internal Mobility State.......................... apfMsMmInitial
Mobility Move Count.............................. 0
--More-- or (q)uit
Security Policy Completed........................ No
Policy Manager State............................. WEBAUTH_REQD ========**
Policy Manager Rule Created...................... Yes
NPU Fast Fast Notified........................... Yes
Last Policy Manager State........................ WEBAUTH_REQD
Client Entry Create Time......................... 67733 seconds
Policy Type...................................... N/A
Encryption Cipher................................ None
Management Frame Protection...................... No
EAP Type......................................... Unknown
Interface........................................ management
VLAN............................................. 0
Client Capabilities:
      CF Pollable................................ Not implemented
      CF Poll Request............................ Not implemented
      Short Preamble............................. Implemented
      PBCC....................................... Not implemented
      Channel Agility............................ Not implemented
      Listen Interval............................ 0
Client Statistics:
      Number of Bytes Received................... 188595
      Number of Bytes Sent....................... 19229
      Number of Packets Received................. 3074
--More-- or (q)uit
      Number of Packets Sent..................... 76
      Number of Policy Errors.................... 0
      Radio Signal Strength Indicator............ -41 dBm
      Signal to Noise Ratio...................... 59 dB
Nearby AP Statistics:
        TxExcessiveRetries: 0
        TxRetries: 0
        RtsSuccessCnt: 0
        RtsFailCnt: 0
        TxFiltered: 0
        TxRateProfile: [0,0,0,0,0,0,0,0,0,0,0,0]
      ap:09:96:10(slot 1) .......................
antenna0: 48 seconds ago -45 dBm................. antenna1: 123 seconds ago -128 dBm
```

Step 5 Make sure the client's pemstate is WEBAUTH_REQD. Open the browser page on the client and look for the following messages:

```
Wed Mar  7 17:59:15 2007: ************* sshpmAddWebRedirectRules: POLICY SEMAPHORE LOCKED
*************
Wed Mar  7 17:59:15 2007: sshpmAddWebRedirectRules: mobile station addr is 10.50.234.3
Wed Mar  7 17:59:15 2007: sshpmAddWebRedirectRules: RuleID for ms 10.50.234.3 is 44
Wed Mar  7 17:59:15 2007: sshpmAddWebRedirectRules: using HTTP-S for web auth (addr:
10.50.234.15).
Wed Mar  7 17:59:15 2007: sshpmAddWebRedirectRules: inbound local http rule created for ms
10.50.234.3 local 1.1.1.1.
Wed Mar  7 17:59:15 2007: sshpmAddWebRedirectRules: inbound http redirect rule created.
Wed Mar  7 17:59:15 2007: sshpmRuleIndexInsert: adding rule for RuleID 44
Wed Mar  7 17:59:15 2007: sshpmRuleIndexInsert: computed raw hash index 02ad3271 for rule
id 0000002c
Wed Mar  7 17:59:15 2007: sshpmRuleIndexInsert: computed adjusted index 00000c32 for rule
id 0000002c
Wed Mar  7 17:59:15 2007: sshpmAddWebRedirectRules: committing rules for ms 10.50.234.3
Wed Mar  7 17:59:15 2007: ************* sshpmPolicyCommitCallback: POLICY SEMAPHORE
UNLOCKED - [unconditionally] ************
```

```
Wed Mar  7 17:59:15 2007: sshpmPolicyCommitCallback: called; ContextPtr: 0x2c; Success: 1
Wed Mar  7 17:59:15 2007: ************* sshpmPolicyCommitCallback: POLICY SEMAPHORE
UNLOCKED - [unconditionally] ************
Wed Mar  7 18:02:32 2007: SshPmAppgw/pm_appgw.c:1234/ssh_pm_appgw_request: New application
gateway request for `alg-http@ssh.com': 10.50.234.3.1153 > 10.50.234.1.80 (nat:
10.50.234.1.80) tcp ft=0x00000000 tt=0x00000000
Wed Mar  7 18:02:32 2007: SshPmAppgw/pm_appgw.c:1239/ssh_pm_appgw_request: Packet
attributes: trigger_rule=0x4ecb, tunnel_id=0x0, trd_index=0xddffffff,
prev_trd_index=0xddffffff
Wed Mar  7 18:02:32 2007: SshPmAppgw/pm_appgw.c:1240/ssh_pm_appgw_request: Packet:
Wed Mar  7 18:02:32 2007: 00000000: 4500 0030 0308 4000 8006 0f57 0a32 ea03
E..0..@....W.2..
Wed Mar  7 18:02:32 2007: 00000010: 0a32 ea01 0481 0050 2f42 e3a4 0000 0000
.2.....P/B......
Wed Mar  7 18:02:32 2007: 00000020: 7002 4000 42fe 0000 0204 05b4 0101 0402
p.@.B...........
Wed Mar  7 18:02:32 2007: SshPmStAppgw/pm_st_appgw.c:403/ssh_pm_st_appgw_start: Calling
redirection callback
Wed Mar  7 18:02:32 2007: SshPmAppgw/pm_appgw.c:155/ssh_appgw_redirect: Application
gateway redirect: 10.50.234.1.80 -> 10.50.234.1.80
Wed Mar  7 18:02:32 2007: SshPmStAppgw/pm_st_appgw.c:445/ssh_pm_st_appgw_mappings:
Creating application gateway mappings: 10.50.234.3.1153 > 10.50.234.1.80 (10.50.234.1.80)
Wed Mar  7 18:02:32 2007: SshPmStAppgw/pm_st_appgw.c:102/ssh_pm_appgw_mappings_cb: appgw
connection cached: init flow_index=5967 resp flow_index=5964 event_cnt=718
Wed Mar  7 18:02:32 2007: SshPmStAppgw/pm_st_appgw.c:493/ssh_pm_st_appgw_mappings_done:
NAT on initiator side
Wed Mar  7 18:02:32 2007:
SshPmStAppgw/pm_st_appgw.c:583/ssh_pm_st_appgw_tcp_responder_stream_done:
ssh_pm_st_appgw_tcp_responder_stream_done: conn->context.responder_stream=0x0
Wed Mar  7 18:02:32 2007:
SshPmStAppgw/pm_st_appgw.c:624/ssh_pm_st_appgw_tcp_responder_stream_done: Opening
initiator stream 10.50.234.1:61611 > 10.76.108.121:2024
Wed Mar  7 18:02:32 2007: SshPmStAppgw/pm_st_appgw.c:154/ssh_pm_appgw_i_flow_enabled:
Initiator flow mode has now been set.
Wed Mar  7 18:02:32 2007: SshPmAppgw/pm_appgw.c:507/ssh_appgw_tcp_listener_callback: New
initiator stream: src=10.50.234.1:61611, dst=10.76.108.121:2024
Wed Mar  7 18:02:32 2007:
SshPmStAppgw/pm_st_appgw.c:646/ssh_pm_st_appgw_tcp_open_initiator_stream: Initiator stream
opened
Wed Mar  7 18:02:32 2007: SshAppgwHttp/appgw_http.c:531/ssh_appgw_http_conn_cb: New TCP
HTTP connection 10.50.234.3.1153 > 10.50.234.1.80
Wed Mar  7 18:02:32 2007: SshAppgwHttp/appgw_http.c:535/ssh_appgw_http_conn_cb: Responder
sees initiator as `10.50.234.15.1153'
Wed Mar  7 18:02:32 2007: SshAppgwHttp/appgw_http.c:539/ssh_appgw_http_conn_cb: Initiator
sees responder as `10.50.234.1.80'
Wed Mar  7 18:02:32 2007: SshAppgwHttp/appgw_http.c:99/ssh_appgw_http_st_wait_input:
entering state st_wait_input: (i) reading_hdr 1 nmsgs 0
Wed Mar  7 18:02:32 2007:
SshAppgwHttpState/appgw_http_state.c:2077/ssh_appgw_http_handle_state: handling: 0 bytes:
Wed Mar  7 18:02:32 2007: SshAppgwHttp/appgw_http.c:136/ssh_appgw_http_st_wait_input: read
-1 bytes (offset 0 data 0)
Wed Mar  7 18:02:32 2007: SshAppgwHttp/appgw_http.c:99/ssh_appgw_http_st_wait_input:
entering state st_wait_input: (r) reading_hdr 1 nmsgs 0
Wed Mar  7 18:02:32 2007:
SshAppgwHttpState/appgw_http_state.c:2077/ssh_appgw_http_handle_state: handling: 0 bytes:
Wed Mar  7 18:02:32 2007: SshAppgwHttp/appgw_http.c:132/ssh_appgw_http_st_wait_input:
appgw_http.c.132: io->src is NULL
Wed Mar  7 18:02:32 2007: SshAppgwHttp/appgw_http.c:136/ssh_appgw_http_st_wait_input: read
-1 bytes (offset 0 data 0)
Wed Mar  7 18:02:32 2007: SshAppgwHttp/appgw_http.c:99/ssh_appgw_http_st_wait_input:
entering state st_wait_input: (i) reading_hdr 1 nmsgs 0
Wed Mar  7 18:02:32 2007:
SshAppgwHttpState/appgw_http_state.c:2077/ssh_appgw_http_handle_state: handling: 0 bytes:
```

```
Wed Mar  7 18:02:32 2007: SshAppgwHttp/appgw_http.c:136/ssh_appgw_http_st_wait_input: read
-1 bytes (offset 0 data 0)
Wed Mar  7 18:02:32 2007: SshAppgwHttp/appgw_http.c:99/ssh_appgw_http_st_wait_input:
entering state st_wait_input: (r) reading_hdr 1 nmsgs 0
Wed Mar  7 18:02:32 2007:
SshAppgwHttpState/appgw_http_state.c:2077/ssh_appgw_http_handle_state: handling: 0 bytes:
Wed Mar  7 18:02:36 2007: SshAppgwHttp/appgw_http.c:132/ssh_appgw_http_st_wait_input:
appgw_http.c.132: io->src is NULL
Wed Mar  7 18:02:36 2007: SshAppgwHttp/appgw_http.c:136/ssh_appgw_http_st_wait_input: read
-1 bytes (offset 0 data 0)
Wed Mar  7 18:02:41 2007: SshAppgwHttp/appgw_http.c:99/ssh_appgw_http_st_wait_input:
entering state st_wait_input: (i) reading_hdr 1 nmsgs 0
Wed Mar  7 18:02:41 2007:
SshAppgwHttpState/appgw_http_state.c:2077/ssh_appgw_http_handle_state: handling: 0 bytes:
Wed Mar  7 18:02:41 2007: SshAppgwHttp/appgw_http.c:136/ssh_appgw_http_st_wait_input: read
283 bytes (offset 0 data 0)
Wed Mar  7 18:02:41 2007:
SshAppgwHttpState/appgw_http_state.c:2077/ssh_appgw_http_handle_state: handling: 283
bytes:
Wed Mar  7 18:02:41 2007: 00000000: 4745 5420 2f20 4854 5450 2f31 2e31 0d0a  GET /
HTTP/1.1..
Wed Mar  7 18:02:41 2007: 00000010: 4163 6365 7074 3a20 696d 6167 652f 6769  Accept:
image/gi
Wed Mar  7 18:02:41 2007: 00000020: 662c 2069 6d61 6765 2f78 2d78 6269 746d  f,
image/x-xbitm
Wed Mar  7 18:02:41 2007: 00000030: 6170 2c20 696d 6167 652f 6a70 6567 2c20  ap,
image/jpeg,
Wed Mar  7 18:02:41 2007: 00000040: 696d 6167 652f 706a 7065 672c 2061 7070  image/pjpeg,
app
Wed Mar  7 18:02:41 2007: 00000050: 6c69 6361 7469 6f6e 2f78 2d73 686f 636b
lication/x-shock
Wed Mar  7 18:02:41 2007: 00000060: 7761 7665 2d66 6c61 7368 2c20 2a2f 2a0d  wave-flash,
*/*.
Wed Mar  7 18:02:41 2007: 00000070: 0a41 6363 6570 742d 4c61 6e67 7561 6765
.Accept-Language
Wed Mar  7 18:02:41 2007: 00000080: 3a20 656e 2d75 730d 0a41 6363 6570 742d  :
en-us..Accept-
Wed Mar  7 18:02:41 2007: 00000090: 456e 636f 6469 6e67 3a20 677a 6970 2c20  Encoding:
gzip,
Wed Mar  7 18:02:41 2007: 000000a0: 6465 666c 6174 650d 0a55 7365 722d 4167
deflate..User-Ag
Wed Mar  7 18:02:41 2007: 000000b0: 656e 743a 204d 6f7a 696c 6c61 2f34 2e30  ent:
Mozilla/4.0
Wed Mar  7 18:02:41 2007: 000000c0: 2028 636f 6d70 6174 6962 6c65 3b20 4d53   (compatible;
MS
Wed Mar  7 18:02:41 2007: 000000d0: 4945 2036 2e30 3b20 5769 6e64 6f77 7320  IE 6.0;
Windows
Wed Mar  7 18:02:41 2007: 000000e0: 4e54 2035 2e31 3b20 5356 3129 0d0a 486f  NT 5.1;
SV1)..Ho
Wed Mar  7 18:02:41 2007: 000000f0: 7374 3a20 3130 2e35 302e 3233 342e 310d  st:
10.50.234.1.
Wed Mar  7 18:02:41 2007: 00000100: 0a43 6f6e 6e65 6374 696f 6e3a 204b 6565  .Connection:
Kee
Wed Mar  7 18:02:41 2007: 00000110: 702d 416c 6976 650d 0a0d 0a              p-Alive....
Wed Mar  7 18:02:41 2007:
SshAppgwHttpState/appgw_http_state.c:985/ssh_appgw_parse_request_line: parsing request
line GET / HTTP/1.1
Wed Mar  7 18:02:41 2007:
SshAppgwHttpState/appgw_http_state.c:1018/ssh_appgw_parse_request_line: internal http
version 3
Wed Mar  7 18:02:41 2007: SshAppgwHttpState/appgw_http_state.c:1155/ssh_appgw_add_method:
caching method 2 for reply 0
Wed Mar  7 18:02:41 2007: SshAppgwHttpState/appgw_http_state.c:1604/ssh_appgw_check_msg:
examining request using service id 34
```

```
Wed Mar  7 18:02:41 2007:
SshAppgwHttpState/appgw_http_state.c:594/ssh_appgw_http_get_dst_host: destination host:
10.50.234.1
Wed Mar  7 18:02:41 2007:
SshAppgwHttpState/appgw_http_state.c:1474/ssh_appgw_inject_reply: injecting 404 reply as
msg 0
Wed Mar  7 18:02:41 2007: SshAppgwHttp/appgw_http.c:284/ssh_appgw_http_st_write_data:
entering state st_write_data
Wed Mar  7 18:02:41 2007: SshAppgwHttp/appgw_http.c:99/ssh_appgw_http_st_wait_input:
entering state st_wait_input: (i) reading_hdr 1 nmsgs 1
Wed Mar  7 18:02:41 2007:
SshAppgwHttpState/appgw_http_state.c:2077/ssh_appgw_http_handle_state: handling: 0 bytes:
Wed Mar  7 18:02:41 2007: SshAppgwHttp/appgw_http.c:136/ssh_appgw_http_st_wait_input: read
-1 bytes (offset 0 data 0)
Wed Mar  7 18:02:41 2007: SshAppgwHttp/appgw_http.c:99/ssh_appgw_http_st_wait_input:
entering state st_wait_input: (r) reading_hdr 1 nmsgs 0
Wed Mar  7 18:02:41 2007:
SshAppgwHttpState/appgw_http_state.c:1851/ssh_appgw_http_is_inject: next inject is msg# 0
current msg# 0
Wed Mar  7 18:02:41 2007: SshAppgwHttp/appgw_http.c:207/ssh_appgw_http_st_inject: entering
state st_inject (r): msgs 0
Wed Mar  7 18:02:41 2007: SshAppgwHttp/appgw_http.c:259/ssh_appgw_http_st_inject: closing
connection after inject
Wed Mar  7 18:02:41 2007: SshAppgwHttp/appgw_http.c:400/ssh_appgw_http_st_terminate:
entering state st_terminate (r): teardown 0 terminate i: 1 r: 1
Wed Mar  7 18:02:45 2007: SshAppgwHttp/appgw_http.c:99/ssh_appgw_http_st_wait_input:
entering state st_wait_input: (i) reading_hdr 1 nmsgs 1
Wed Mar  7 18:02:45 2007:
SshAppgwHttpState/appgw_http_state.c:2077/ssh_appgw_http_handle_state: handling: 0 bytes:
Wed Mar  7 18:02:45 2007: SshAppgwHttp/appgw_http.c:400/ssh_appgw_http_st_terminate:
entering state st_terminate (i): teardown 0 terminate i: 1 r: 1
Wed Mar  7 18:02:45 2007:
SshAppgwHttp/appgw_http.c:732/ssh_appgw_http_connection_terminate: service HTTP-REDIR: TCP
HTTP connection 10.50.234.3.1153 > 10.50.234.1.80 terminated
Wed Mar  7 18:02:45 2007: SshPmStAppgw/pm_st_appgw.c:1094/ssh_pm_st_appgw_terminate:
terminating appgw instance
```

Step 6 If you do not see the HTTP GET message, the HTTP packet has not reached the controller. After the client completes the redirection, enter your login and submit it.

Step 7 Look at the client’s entry in NPUdevshell hapiMmcDebugScbInfoShow (‘client mac address’). If the PEM state is not moved from WEBAUTH_REQD to RUN, a credential problem exists. Check the credentials in the local or RADIUS database (where ever they were configured).

Step 8 When the RUN state appears on the client, perform a check from the client to the gateway and see if traffic is being passed.

Best Practices

If the client is not redirected to the login page and you want to avoid DNS resolution in the network, enter **http://***controller-mgmt-ip*. If a redirection occur, the issue is not network related.

Enter **config network web-auth-port** *Port* to define the ports on the controller other than the standard HTTP port (80). The controller does not interrupt secure HTTP or HTTPS (443) even if the port is configured for interrupt.