WLANConfiguration:1
Service Template Version 1.01
For UPnP™ Version 1.0
Status: Standardized DCP
Date: October 17, 2003

This Standardized DCP has been adopted as a Standardized DCP by the Steering Committee of the
UPnP™ Forum, pursuant to Section 2.1(c)(ii) of the UPnP™ Forum Membership Agreement. UPnP™
Forum Members have rights and licenses defined by Section 3 of the UPnP™ Forum Membership
Agreement to use and reproduce the Standardized DCP in UPnP™ Compliant Devices. All such use is
subject to all of the provisions of the UPnP™ Forum Membership Agreement.

THE UPNP™ FORUM TAKES NO POSITION AS TO WHETHER ANY INTELLECTUAL
PROPERTY RIGHTS EXIST IN THE STANDARDIZED DCPS. THE STANDARDIZED DCPS ARE
PROVIDED "AS IS" AND "WITH ALL FAULTS". THE UPNP™ FORUM MAKES NO
WARRANTIES, EXPRESS, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE
STANDARDIZED DCPS, INCLUDING BUT NOT LIMITED TO ALL IMPLIED WARRANTIES OF
MERCHANTABILITY, NON-INFRINGEMENT AND FITNESS FOR A PARTICULAR PURPOSE, OF
REASONABLE CARE OR WORKMANLIKE EFFORT, OR RESULTS OR OF LACK OF
NEGLIGENCE.

© 2000-2003 Contributing Members of the UPnP™ Forum All rights Reserved.

<table>
<thead>
<tr>
<th>Authors</th>
<th>Company</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mark Yoshitake</td>
<td>Microsoft Corporation</td>
</tr>
<tr>
<td>William Lupton</td>
<td>GlobespanVirata Inc</td>
</tr>
<tr>
<td>Sungjoon Ahn</td>
<td>LG Electronics Inc</td>
</tr>
<tr>
<td>Ajay Garg</td>
<td>Intel Corporation</td>
</tr>
<tr>
<td>Trevor Freeman</td>
<td>Microsoft Corporation</td>
</tr>
</tbody>
</table>
Contents

1. OVERVIEW AND SCOPE ................................................................................................. 7

2. SERVICE MODELING DEFINITIONS ........................................................................... 7

   2.1. SERVICE TYPE ......................................................................................................... 7

   2.2. STATE VARIABLES ................................................................................................. 7

   2.2.1. InsecureOOBAccessEnabled .................................................................................. 10

   2.2.2. SSID ....................................................................................................................... 10

   2.2.3. BSSID ..................................................................................................................... 10

   2.2.4. BeaconType .......................................................................................................... 10

   2.2.5. BeaconAdvertisementEnabled .............................................................................. 11

   2.2.6. RadioEnabled ........................................................................................................ 11

   2.2.7. LocationDescription ............................................................................................ 11

   2.2.8. RegulatoryDomain .............................................................................................. 11

   2.2.9. TotalPSKFailures ................................................................................................. 12

   2.2.10. TotalIntegrityFailures ....................................................................................... 12

   2.2.11. Channel ............................................................................................................... 12

   2.2.12. PossibleChannels ............................................................................................... 12

   2.2.13. ChannelsInUse .................................................................................................... 12

   2.2.14. DeviceOperationMode ....................................................................................... 13

   2.2.15. DistanceFromRoot ............................................................................................ 13

   2.2.16. PeerBSSID ........................................................................................................... 13

   2.2.17. BasicDataTransmitRates ..................................................................................... 13

   2.2.18. OperationalDataTransmitRates ......................................................................... 13

   2.2.19. PossibleDataTransmitRates ................................................................................ 13

   2.2.20. AutoRateFallbackEnabled .................................................................................. 13

   2.2.21. TotalBytesSent .................................................................................................. 14

   2.2.22. TotalBytesReceived ........................................................................................... 14

   2.2.23. TotalPacketsSent ............................................................................................... 14

   2.2.24. TotalPacketsReceived ....................................................................................... 14

   2.2.25. TotalAssociations .............................................................................................. 14

   2.2.26. AssociatedDeviceMACAddress ........................................................................... 14

   2.2.27. AssociatedDeviceIPAddress ............................................................................... 14

   2.2.28. AssociatedDeviceAuthState ............................................................................... 14

   2.2.29. AuthenticationServiceMode ................................................................................ 14

   2.2.30. WEPKeyIndex .................................................................................................... 15

   2.2.31. WEPKey .............................................................................................................. 15

   2.2.32. KeyPassphrase ................................................................................................... 15

   2.2.33. WEPEncryptionLevel ....................................................................................... 15

   2.2.34. PreSharedKey ...................................................................................................... 16

   2.2.35. PreSharedKeyId .................................................................................................. 16

   2.2.36. BasicEncryptionModes ....................................................................................... 16

   2.2.37. BasicAuthenticationMode ................................................................................... 16

   2.2.38. WPAEncryptionModes ....................................................................................... 17

   2.2.39. WPAAuthenticationMode .................................................................................... 17

   2.2.40. IEEE11iEncryptionModes ..................................................................................... 18

   2.2.41. IEEE11iAuthenticationMode ............................................................................... 18

   2.2.42. LastRequestedUnicastCipher ............................................................................. 19

   2.2.43. LastRequestedMulticastCipher .......................................................................... 19

   2.2.44. LastPMKId .......................................................................................................... 19

   2.3. EVENTING AND MODERATION ............................................................................. 19

   2.3.1. Event Model ......................................................................................................... 19

2.4. ACTIONS ............................................................................................................................................... 19
2.4.1. SetInsecureOutOfBandAccessMode .......................................................................................... 21
2.4.2. GetInsecureOutOfBandAccessMode ......................................................................................... 22
2.4.3. SetSSID ......................................................................................................................................... 22
2.4.4. GetSSID ......................................................................................................................................... 23
2.4.5. GetBSSID ....................................................................................................................................... 23
2.4.6. SetBeaconType ............................................................................................................................ 24
2.4.7. GetBeaconType ............................................................................................................................ 24
2.4.8. SetBeaconAdvertisement ............................................................................................................ 25
2.4.9. GetBeaconAdvertisement ........................................................................................................... 25
2.4.10. SetRadioMode ........................................................................................................................... 26
2.4.11. GetRadioMode ........................................................................................................................... 26
2.4.12. SetLocationDescription ........................................................................................................... 27
2.4.13. GetLocationDescription ........................................................................................................... 27
2.4.14. SetRegulatoryDomain ............................................................................................................. 28
2.4.15. GetRegulatoryDomain ............................................................................................................. 28
2.4.16. GetFailureStatusInfo ............................................................................................................... 28
2.4.17. SetChannel ................................................................................................................................ 29
2.4.18. GetChannelInfo ......................................................................................................................... 30
2.4.19. GetChannelsInUse ..................................................................................................................... 30
2.4.20. SetDeviceOperationMode ........................................................................................................... 31
2.4.21. GetDeviceOperationMode .......................................................................................................... 32
2.4.22. SetDataTransmitRates .............................................................................................................. 32
2.4.23. GetDataTransmitRateInfo ........................................................................................................ 33
2.4.24. SetAutoRateFallbackMode ...................................................................................................... 34
2.4.25. GetAutoRateFallbackMode ...................................................................................................... 34
2.4.26. GetByteStatistics ..................................................................................................................... 35
2.4.27. GetPacketStatistics .................................................................................................................. 35
2.4.28. GetByteStatsForAssociatedDev .............................................................................................. 36
2.4.29. GetPacketStatsForAssociatedDev ............................................................................................ 36
2.4.30. GetTotalAssociations ............................................................................................................... 37
2.4.31. GetGenericAssociatedDeviceInfo .......................................................................................... 37
2.4.32. GetSpecificAssociatedDeviceInfo .......................................................................................... 38
2.4.33. GetSpecificAssociatedDevIIInfo ............................................................................................ 39
2.4.34. SetAuthenticationServiceMode .............................................................................................. 39
2.4.35. GetAuthenticationServiceMode .............................................................................................. 40
2.4.36. SetSecurityKeys ....................................................................................................................... 40
2.4.37. GetSecurityKeys ....................................................................................................................... 41
2.4.38. SetDefaultWEPKeyIndex ......................................................................................................... 42
2.4.39. GetDefaultWEPKeyIndex ......................................................................................................... 43
2.4.40. SetPreSharedKey ...................................................................................................................... 43
2.4.41. GetPreSharedKey ...................................................................................................................... 44
2.4.42. SetBasicBeaconSecurityProperties ......................................................................................... 44
2.4.43. GetBasicBeaconSecurityProperties ......................................................................................... 45
2.4.44. SetWPA BeaconSecurityProperties ......................................................................................... 46
2.4.45. GetWPA BeaconSecurityProperties ......................................................................................... 46
2.4.46. Set11i BeaconSecurityProperties ............................................................................................. 47
2.4.47. Get11i BeaconSecurityProperties ............................................................................................. 48
2.4.48. FactoryDefaultReset ................................................................................................................ 48
2.4.49. ResetAuthentication ................................................................................................................ 49
2.4.50. Non-Standard Actions Implemented by a UPnP™ Device Vendor ....................................... 49
2.4.51. Common Error Codes ............................................................................................................. 49
2.5. THEORY OF OPERATION .................................................................................................................. 50
2.5.1. Initialization of the AP Device ................................................................................................... 50
2.5.2. Operation of the AP Device ................................................................................................... 51

3. XML SERVICE DESCRIPTION ...................................................................................................... 57

4. TEST .............................................................................................................................................. 71

List of Tables

Table 1: State Variables ............................................................................................................... 8
Table 1.1: allowedValueList for BeaconType .................................................................................. 10
Table 1.2: allowedValueList for DeviceOperationMode ................................................................. 12
Table 1.3: allowedValueList for AuthenticationServiceMode ....................................................... 14
Table 1.4: allowedValueList for WPEncryptionLevel .................................................................... 15
Table 1.5: Allowed value list for BasicEncryptionLevels ............................................................... 16
Table 1.6: Allowed value list for BasicAuthenticationMode ......................................................... 16
Table 1.7: Allowed value list for WPAEncryptionModes ................................................................. 17
Table 1.8: Allowed value list for WPAAuthenticationMode ............................................................ 17
Table 1.9: Allowed value list for IEEE11iEncryptionModes ........................................................... 18
Table 1.10: Allowed value list for IEEE11iAuthenticationMode .................................................... 18
Table 2: Event Moderation ............................................................................................................. 19
Table 3: Actions ................................................................................................................................. 20
Table 4: Arguments for SetInsecureOutOfBandAccessMode ............................................................ 21
Table 5: Arguments for SetInsecureOutOfBandAccessMode ............................................................ 22
Table 6: Arguments for SetSSID ...................................................................................................... 22
Table 7: Arguments for GetSSID ..................................................................................................... 23
Table 8: Arguments for GetBSSID .................................................................................................... 23
Table 9: Arguments for SetBeaconType ........................................................................................... 24
Table 10: Arguments for GetBeaconType ......................................................................................... 24
Table 11: Arguments for SetBeaconAdvertisement .......................................................................... 25
Table 12: Arguments for GetBeaconAdvertisement ......................................................................... 25
Table 13: Arguments for SetRadioMode .......................................................................................... 26
Table 14: Arguments for GetRadioMode ......................................................................................... 26
Table 15: Arguments for SetLocationDescription .......................................................................... 27
Table 16: Arguments for GetLocationDescription .......................................................................... 27

Table 17: Arguments for SetRegulatoryDomain.......................................................................................28
Table 18: Arguments for GetRegulatoryDomain .......................................................................................28
Table 19: Arguments for GetFailureStatusInfo .........................................................................................29
Table 20: Arguments for SetChannel ...........................................................................................................29
Table 21: Arguments for GetChannelInfo ....................................................................................................30
Table 22: Arguments for GetChannelsInUse ...............................................................................................30
Table 23: Arguments for SetDeviceOperationMode .....................................................................................31
Table 24: Arguments for GetDeviceOperationMode ....................................................................................32
Table 25: Arguments for SetDataTransmitRates ............................................................................................33
Table 26: Arguments for GetDataTransmitRateInfo ...................................................................................33
Table 27: Arguments for SetAutoRateFallBackMode ..................................................................................34
Table 28: Arguments for GetAutoRateFallBackMode ..................................................................................34
Table 29: Arguments for GetByteStatistics ...............................................................................................35
Table 30: Arguments for GetPacketStatistics .............................................................................................35
Table 29: Arguments for GetByteStatsForAssociatedDev ............................................................................36
Table 30: Arguments for GetPacketStatsForAssociatedDev ........................................................................36
Table 31: Arguments for GetTotalAssociations .........................................................................................37
Table 32: Arguments for GetGenericAssociatedDeviceInfo .........................................................................37
Table 33: Arguments for GetSpecificAssociatedDeviceInfo ..........................................................................38
Table 33: Arguments for GetSpecificAssociatedDeviceInfo ..........................................................................39
Table 34: Arguments for SetAuthenticationServiceMode ............................................................................40
Table 35: Arguments for GetAuthenticationServiceMode ............................................................................40
Table 36: Arguments for SetSecurityKeys .................................................................................................41
Table 37: Arguments for GetSecurityKeys .................................................................................................41
Table 38: Arguments for SetDefaultWEPKeyIndex ....................................................................................42
Table 39: Arguments for GetDefaultWEPKeyIndex ....................................................................................43
Table 40: Arguments for SetPreSharedKey ...............................................................................................43
Table 41: Argument for GetPreSharedKey ...............................................................................................44
Table 42: Arguments for SetBasBeaconSecurityProperties .........................................................................45
Table 43: Arguments for GetBasBeaconSecurityProperties .........................................................................45
Table 44: Arguments for SetWPABeaconSecurityProperties .......................................................................46
Table 45: Arguments for GetWPABeaconSecurityProperties ................................................................. 47
Table 46: Arguments for Set11iBeaconSecurityProperties ............................................................... 47
Table 47: Arguments for Get11iBeaconSecurityProperties ............................................................... 48
Table 48: Common Error Codes for all actions .................................................................................. 49
1. Overview and Scope
This service definition is compliant with the UPnP™ Device Architecture version 1.0.

This service enables the control, monitoring and configuration of IEEE 802.11 Wireless Access Points for the unmanaged network space, namely residential and small office LANs. It focuses on the core elements required for setting up wireless networks, configuring wireless security, diagnosing and monitoring usage problems, and setting location-specific information elements.

Its intent is to simplify the setup experience, secure wireless networks and provide the framework for diagnosing and monitoring problems on wireless networks.

This service-type enables the following functions:

- Remote setup and configuration of the basic parameters of a wireless access point including: SSID, radio channel configuration, wireless access point mode and others.
- Remote setup and configuration of wireless authentication for WPA, 11i and WEP based security.
- Remote configuration and provisioning of location specific information.
- Remote diagnostics and monitoring of wireless networks.

2. Service Modeling Definitions

2.1. ServiceType

The service is REQUIRED as specified in urn:schemas-upnp-org:device:WLANAccessPointDevice:1

The following service type identifies a service that is compliant with this template: urn:schemas-upnp-org:service:WLANConfiguration:1

This service does not support the QueryStateVariable action.

2.2. State Variables

Table 1 shows all the state variables of the WLANConfiguration service. These variables are represented in the order:

- First set of variables is Access Point common beacon properties.
- Second set shows the status-related variables
- Third set represents all statistics-related parameters.
- Fourth set shows all the associated device-related parameters
- Last set represents all the security-related parameters.
Table 1: State Variables

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Req. or Opt.</th>
<th>Data Type</th>
<th>Allowed Value</th>
<th>Default Value</th>
<th>Eng. Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>InsecureOOBAccessEnabled</td>
<td>O</td>
<td>Boolean</td>
<td>0, 1</td>
<td>0</td>
<td>N/A</td>
</tr>
<tr>
<td>SSID</td>
<td>R</td>
<td>String</td>
<td>&lt;= 32 char</td>
<td>Not Specified</td>
<td>N/A</td>
</tr>
<tr>
<td>BeaconType</td>
<td>R</td>
<td>String</td>
<td>See Table 1.1, &lt;= 32 char</td>
<td>Not Specified</td>
<td>N/A</td>
</tr>
<tr>
<td>BeaconAdvertisementEnabled</td>
<td>O</td>
<td>Boolean</td>
<td>0, 1</td>
<td>1</td>
<td>N/A</td>
</tr>
<tr>
<td>RadioEnabled</td>
<td>R</td>
<td>Boolean</td>
<td>0, 1</td>
<td>1</td>
<td>N/A</td>
</tr>
<tr>
<td>LocationDescription</td>
<td>O</td>
<td>String</td>
<td>XML Location Schema, &lt;= 4096 char</td>
<td>Not specified</td>
<td>N/A</td>
</tr>
<tr>
<td>RegulatoryDomain</td>
<td>O</td>
<td>String</td>
<td>== 3 chars</td>
<td>Not Specified</td>
<td>N/A</td>
</tr>
<tr>
<td>TotalPSKFailures</td>
<td>O</td>
<td>ui4</td>
<td>&gt;=0</td>
<td>0</td>
<td>N/A</td>
</tr>
<tr>
<td>TotalIntegrityFailures</td>
<td>O</td>
<td>ui4</td>
<td>&gt;=0</td>
<td>0</td>
<td>N/A</td>
</tr>
<tr>
<td>Channel</td>
<td>R</td>
<td>ui1</td>
<td>Depends on PossibleChannels</td>
<td>Not Specified</td>
<td>N/A</td>
</tr>
<tr>
<td>PossibleChannels</td>
<td>R</td>
<td>String</td>
<td>&lt;= 1024 char</td>
<td>Not Specified</td>
<td>N/A</td>
</tr>
<tr>
<td>ChannelsInUse</td>
<td>O</td>
<td>String</td>
<td>Depends on PossibleChannels, &lt;= 1024 char</td>
<td>Not specified</td>
<td>N/A</td>
</tr>
<tr>
<td>DeviceOperationMode</td>
<td>O*</td>
<td>String</td>
<td>See Table 1.2, &lt;= 32 char</td>
<td>“Infrastructure AccessPoint”</td>
<td>N/A</td>
</tr>
<tr>
<td>DistanceFromRoot</td>
<td>O*</td>
<td>ui1</td>
<td>&gt;= 0</td>
<td>0</td>
<td>N/A</td>
</tr>
<tr>
<td>BasicDataTransmitRates</td>
<td>R</td>
<td>String</td>
<td>Depends on PossibleDataTransmitRates, &lt;=256 char</td>
<td>Not Specified</td>
<td>Megabits per sec</td>
</tr>
<tr>
<td>OperationalDataTransmitRates</td>
<td>R</td>
<td>String</td>
<td>Depends on PossibleDataTransmitRates, &lt;=256 char</td>
<td>Not Specified</td>
<td>Megabits per sec</td>
</tr>
<tr>
<td>PossibleDataTransmitRates</td>
<td>R</td>
<td>String</td>
<td>&lt;=256 char</td>
<td>Not Specified</td>
<td>Megabits per sec</td>
</tr>
<tr>
<td>AutoRateFallBackEnabled</td>
<td>R</td>
<td>Boolean</td>
<td>0, 1</td>
<td>1</td>
<td>N/A</td>
</tr>
<tr>
<td>TotalBytesSent</td>
<td>O**</td>
<td>ui4</td>
<td>&gt;=0</td>
<td>0</td>
<td>N/A</td>
</tr>
<tr>
<td>TotalBytesReceived</td>
<td>O**</td>
<td>ui4</td>
<td>&gt;=0</td>
<td>0</td>
<td>N/A</td>
</tr>
<tr>
<td>TotalPacketsSent</td>
<td>R</td>
<td>ui4</td>
<td>&gt;=0</td>
<td>0</td>
<td>N/A</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Req. or Opt.</th>
<th>Data Type</th>
<th>Allowed Value</th>
<th>Default Value</th>
<th>Eng. Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>TotalPacketsReceived</td>
<td>R</td>
<td>ui4</td>
<td>&gt;= 0</td>
<td>0</td>
<td>N/A</td>
</tr>
<tr>
<td>TotalAssociations</td>
<td>R</td>
<td>ui2</td>
<td>&gt;= 0</td>
<td>0</td>
<td>N/A</td>
</tr>
<tr>
<td>AssociatedDeviceMACAddress</td>
<td>R</td>
<td>String</td>
<td>MAC Address, “XX:XX:XX:XX:XX:XX”, case-independent, 17 char</td>
<td>Not Specified</td>
<td>N/A</td>
</tr>
<tr>
<td>AssociatedDeviceIPAddress</td>
<td>R</td>
<td>String</td>
<td>IP Address, &lt;= 64 char</td>
<td>Not Specified</td>
<td>N/A</td>
</tr>
<tr>
<td>AssociatedDeviceAuthState</td>
<td>R</td>
<td>Boolean</td>
<td>0, 1</td>
<td>Not Specified</td>
<td>N/A</td>
</tr>
<tr>
<td>AuthenticationServiceMode</td>
<td>R</td>
<td>String</td>
<td>See Table 1.3, &lt;= 32 char</td>
<td>Not Specified</td>
<td>N/A</td>
</tr>
<tr>
<td>WEPKeyIndex</td>
<td>R</td>
<td>ui1</td>
<td>0 – 3</td>
<td>Not specified</td>
<td>N/A</td>
</tr>
<tr>
<td>WEPKey</td>
<td>R</td>
<td>String</td>
<td>WEP key of correct encryption level, &lt;= 128 char</td>
<td>Empty string</td>
<td>N/A</td>
</tr>
<tr>
<td>KeyPassphrase</td>
<td>R</td>
<td>String</td>
<td>&lt;= 63 char</td>
<td>Not specified</td>
<td>N/A</td>
</tr>
<tr>
<td>WEPEncryptionLevel</td>
<td>R</td>
<td>String</td>
<td>See Table 1.4, &lt;= 32 char</td>
<td>Not specified</td>
<td>N/A</td>
</tr>
<tr>
<td>PreSharedKey</td>
<td>R</td>
<td>String</td>
<td>Key, 64 Hex char, case-independent</td>
<td>Not specified</td>
<td>N/A</td>
</tr>
<tr>
<td>PreSharedKeyIndex</td>
<td>O</td>
<td>ui1</td>
<td>1 – 9</td>
<td>Not specified</td>
<td>N/A</td>
</tr>
<tr>
<td>BasicEncryptionModes</td>
<td>R</td>
<td>String</td>
<td>See Table 1.5, &lt;= 32 char</td>
<td>Not specified</td>
<td>N/A</td>
</tr>
<tr>
<td>BasicAuthenticationMode</td>
<td>R</td>
<td>String</td>
<td>See Table 1.6, &lt;= 32 char</td>
<td>Not Specified</td>
<td>N/A</td>
</tr>
<tr>
<td>WPAEncryptionModes</td>
<td>R</td>
<td>String</td>
<td>See Table 1.7, &lt;= 32 char</td>
<td>Not specified</td>
<td>N/A</td>
</tr>
<tr>
<td>WPAAuthenticationMode</td>
<td>R</td>
<td>Boolean</td>
<td>See Table 1.8, &lt;= 32 char</td>
<td>Not specified</td>
<td>N/A</td>
</tr>
<tr>
<td>IEEE11iEncryptionModes</td>
<td>R</td>
<td>Boolean</td>
<td>See Table 1.9, &lt;= 32 char</td>
<td>Not specified</td>
<td>N/A</td>
</tr>
<tr>
<td>IEEE11iAuthenticationMode</td>
<td>R</td>
<td>Boolean</td>
<td>See Table 1.10, &lt;= 32 char</td>
<td>Not specified</td>
<td>N/A</td>
</tr>
<tr>
<td>LastRequestedUnicastCipher</td>
<td>O</td>
<td>String</td>
<td>&lt;=256 char</td>
<td>Not specified</td>
<td>N/A</td>
</tr>
<tr>
<td>LastRequestedMulticastCipher</td>
<td>O</td>
<td>String</td>
<td>&lt;=256 char</td>
<td>Not specified</td>
<td>N/A</td>
</tr>
<tr>
<td>LastPMKId</td>
<td>O</td>
<td>String</td>
<td>&lt;=256 char</td>
<td>Not specified</td>
<td>N/A</td>
</tr>
<tr>
<td>Non-standard state variables implemented by an UPnP™ device vendor go here.</td>
<td>X</td>
<td>TBD</td>
<td>TBD</td>
<td>TBD</td>
<td>TBD</td>
</tr>
</tbody>
</table>

1 R = Required, O = Optional, X = Non-standard.

2 Values listed in this column are required. To specify standard optional values or to delegate assignment of values to the vendor, you must reference a specific instance from the appropriate table below.

* DeviceOperationMode, DistanceFromRoot and PeerBSSID are conditionally optional. If the device implementer includes Set/GetDeviceOperationMode actions in the WLANConfiguration XML then these state variables also MUST be present in the WLANConfiguration XML.
**TotalBytesSent** and **TotalBytesReceived** are conditionally optional. If the device implementer includes **GetByteStatistics** or **GetByteStatsForAssociatedDev** action in the WLANConfiguration XML then these state variables also MUST be present in the WLANConfiguration XML.

### 2.2.1. InsecureOOBAccessEnabled
This state variable is optional and is relevant only if securing of UPnP™ actions is implemented in the AP device. This variable indicates whether the state variables that are accessed using secure actions on the Access point device can be set/retrieved via a mechanism that is not based on UPnP™ technology, such as Web Browser, SNMP, Telnet, etc. This is a read/write variable.

### 2.2.2. SSID
This variable represents the Service Set Identifier or network name. This is used by the client to connect to the wireless network created by the access point. There is a 32-character limit on this string. This variable is read/write.

### 2.2.3. BSSID
This variable represents the Basic Service Set Identifier. This is used to identify the physical address of the wireless access point for advanced bridging scenarios or to uniquely identify it if there are multiple UPnP™ access points. This variable is limited to valid MAC addresses. This variable is read only.

### 2.2.4. BeaconType
This variable indicates what beacon mode the access point is in. The value must be selected from Table 1.1. This variable is read/write.

<table>
<thead>
<tr>
<th>Value</th>
<th>Req. or Opt.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>R</td>
<td>Beacon is disabled</td>
</tr>
<tr>
<td>Basic</td>
<td>R</td>
<td>AP is enabled with beacon as per basic 802.11 specification</td>
</tr>
<tr>
<td>WPA</td>
<td>R</td>
<td>AP is enabled with beacon as per WPA specification</td>
</tr>
<tr>
<td>11i</td>
<td>O</td>
<td>AP is enabled with beacon as per 802.11 specification</td>
</tr>
<tr>
<td>BasicandWPA</td>
<td>O</td>
<td>AP is enabled with beacon as per basic 802.11 and WPA specifications</td>
</tr>
<tr>
<td>Basicand11i</td>
<td>O</td>
<td>AP is enabled with beacon as per basic 802.11 and 11i specifications</td>
</tr>
<tr>
<td>WPAand11i</td>
<td>O</td>
<td>AP is enabled with beacon as per WPA and 11i specifications</td>
</tr>
<tr>
<td>BasicandWPAand11i</td>
<td>O</td>
<td>AP is enabled with beacon as per basic 802.11 and WPA and 11i specifications</td>
</tr>
<tr>
<td>Vendor-defined</td>
<td>R</td>
<td>R</td>
</tr>
<tr>
<td>Vendor-defined</td>
<td>O</td>
<td>O</td>
</tr>
</tbody>
</table>

R = Required, O = Optional.

2.2.5. **BeaconAdvertisementEnabled**
This variable indicates whether or not the access point is advertising the beacon. This is a read/write variable.

2.2.6. **RadioEnabled**
This variable indicates whether or not the Wireless Radio is enabled. This is a read/write variable.

2.2.7. **LocationDescription**
This variable contains an XML description of some basic information that is useful in identifying both the access point by name and the physical location. The AP is not expected to parse this description: it will treat it as an opaque string.

The location XML schemas are still in the definition stage from multiple standards bodies. The two references are included as examples of what control points could implement.

1. OGC (Open GIS Consortium) GML 3.0 (Geography Markup Language) specification.

2. OMA Location Working Group specification (formerly LIF)

The AP does not interpret the contents of the location field. When there is no location set at the AP and a control point calls the `GetLocationDescription` action, the location XML would be an empty string. If `DeviceSecurity` is implemented, the AP device implementation must escape the XML per `DeviceSecurity` specification.

2.2.8. **RegulatoryDomain**
This variable indicates which RegulatoryDomain the access point is in. This variable is read/write.

The RegulatoryDomain state variable MUST be defined as per 802.11d specification. 802.11d specifies that, if multi-domain capability is enabled in the AP, beacons will include country information, consisting of a three octet country string and a set of one or more (start channel, number of channels, maximum transmit power) triples. The country string is defined as follows.

"This attribute identifies the country in which the station is operating. The first two octets of this string is the two character country code as described in document ISO/IEC 3166-1. The third octet shall be one of the following:

1. an ASCII space character, if the regulations under which the station is operating encompass all environments in the country,
2. an ASCII ‘O’ character, if the regulations under which the station is operating are for an Outdoor environment only, or
3. an ASCII ‘I’ character, if the regulations under which the station is operating are for an Indoor environment only."

For example, the country code of the United States is US and the country code of Japan is JP. The RegulatoryDomain state variable for setting all environments in the United States would be “US”. Another example for setting the RegulatoryDomain state variable for only inside environments in the United States would be “USI”.

2.2.9. TotalPSKFailures
This variable indicates the number of times the Pre Shared Key authentication has failed. This can give an indication of whether there were more attempts than normal to get onto the network that have failed. This is a read only variable.

2.2.10. TotalIntegrityFailures
This variable indicates the number of times the integrity check for MICHAEL has failed. This can give an indication of whether there were attempts to get onto the network. This is a read only variable.

2.2.11. Channel
This variable represents the wireless LAN channel that the AP is currently using. This could be changed programmatically to select a particular channel in a given environment. This variable is read/write.

2.2.12. PossibleChannels
This variable contains the range of WLAN channels possible. It is a comma-separated list; “n-m” ranges are permitted. For example, 802.11b, possible channels are 1-11 for North America. For 802.11a, channel numbers are in the range 0-200. This variable is read only.

2.2.13. ChannelsInUse
This variable represents the WLAN channels that the Access Point detects as currently being in use in the area. It is a comma-separated list; “n-m” ranges are permitted. It includes any channels that the AP is itself using. This variable is read only.

2.2.14. DeviceOperationMode
This variable indicates which mode the access point is in. This variable is read/write.

Table 1.2: allowedValueList for DeviceOperationMode

<table>
<thead>
<tr>
<th>Value</th>
<th>Req. or Opt.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InfrastructureAccessPoint</td>
<td>R</td>
<td>This is the default mode of the Access Point. It will provide wireless stations connectivity to the LAN.</td>
</tr>
<tr>
<td>WirelessBridge</td>
<td>O</td>
<td>In this mode, the wireless access point is configured (via BSSID) to communicate directly with another access point(s) to bridge two or more separate LANs together</td>
</tr>
<tr>
<td>WirelessRepeater</td>
<td>O</td>
<td>In this mode, the wireless access point is configured to be a point to point bridge as well as an infrastructure access point serving wireless clients.</td>
</tr>
<tr>
<td>WirelessStation</td>
<td>O</td>
<td>In this mode, the wireless access point is configured to be a wireless station connecting to another SSID.</td>
</tr>
</tbody>
</table>
### 2.2.15. DistanceFromRoot

This variable represents the number of hops from the root access point repeater or bridge when the access point is being used as a wireless repeater or bridge. The root repeater or bridge has zero as the distance value. This variable is read/write.

### 2.2.16. PeerBSSID

This variable represents the physical address of the peer AP in repeating or bridging mode. This variable is limited to valid MAC addresses. This variable is read/write.

### 2.2.17. BasicDataTransmitRates

This variable represents the maximum AP data transmission rates in megabits per second for unicast, multicast, and broadcast frames. It is a comma-separated list. This variable is read/write.

For example, `BasicDataTransmitRates` might be “1,2”, indicating that unicast, multicast and broadcast frames can be transmitted at 1Mbps and 2Mbps.

### 2.2.18. OperationalDataTransmitRates

This variable represents the maximum AP data transmission rates in megabits per second for unicast frames. `OperationalDataTransmitRates` is a superset of `BasicDataTransmitRates` (note that an AP can transmit unicast frames at higher speeds than multicast and broadcast frames). It is a comma-separated list. This variable is read/write.

Given the value of `BasicDataTransmitRates` from the previous section, “1,2”, `OperationalDataTransmitRates` might be “1,2,5,5,11”, indicating that unicast frames can in addition be transmitted at 5.5Mbps and 11Mbps.

### 2.2.19. PossibleDataTransmitRates

This variable represents the possible data rates, in megabits per second for unicast frames, at which the AP will allow devices to connect (this could be useful for minimizing the use of bandwidth when users are charged per bit). `PossibleDataTransmitRates` is a subset of `OperationalDataTransmitRates`. It is a comma-separated list. This variable is read only.

Given the values of `BasicDataTransmitRates` and `OperationalDataTransmitRates` from the above two sections, `PossibleDataTransmitRates` might be “1,2,5,5”, indicating that the AP will only permit connections at 1Mbps, 2Mbps and 5.5Mbps, even though it could theoretically accept a connection at 11Mbps.

### 2.2.20. AutoRateFallBackEnabled

This variable indicates whether the data rate can automatically be reduced in the event of undue noise or contention. This variable is read/write.

---

1 R = Required, O = Optional.
2.2.21.TotalBytesSent
This variable indicates the total number of bytes sent on the wireless interface since the last bytes data reset. This is a read only variable that can be reset to 0 by AP device when user performs Reset operations. The value is expected to wrap around when it reaches maximum value.

2.2.22.TotalBytesReceived
This variable indicates the total number of bytes received on the wireless interface since the last bytes data reset. This is a read only variable that can be reset to 0 by AP device when user performs Reset operations. The value is expected to wrap around when it reaches maximum value.

2.2.23.TotalPacketsSent
This variable indicates the total number of packets sent on the wireless interface since the last packet data reset. This is a read only variable that can be reset to 0 by AP device when user performs Reset operations. The value is expected to wrap around when it reaches maximum value.

2.2.24.TotalPacketsReceived
This variable indicates the total number of packets received on the wireless interface since the last packet data reset. This is a read only variable that can be reset to 0 by AP device when user performs Reset operations. The value is expected to wrap around when it reaches maximum value.

2.2.25.TotalAssociations
This variable indicates the current number of devices associated with the access point. This is a read only variable that is evented.

2.2.26.AssociatedDeviceMACAddress
This variable indicates MAC Address of a device currently associated with the access point. This is a read only variable.

2.2.27.AssociatedDeviceIPAddress
This variable indicates the IP Address of a device currently associated with the access point. This may be an IPv4 address, an IPv6 address, or a DNS name. This is a read only variable. If the access point vendor is unable to determine this information, it should be set to an empty string.

2.2.28.AssociatedDeviceAuthState
This variable indicates the Authentication State of a device currently associated with the access point. This is a read only variable.

2.2.29.AuthenticationServiceMode
This variable indicates whether or not an authentication server is co-located with an access point. This variable is read/write.

Table 1.3: allowedValueList for AuthenticationServiceMode

<table>
<thead>
<tr>
<th>Value</th>
<th>Req. or Opt.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>R</td>
<td>This value indicates that no authentication service (for example, LinkAuthentication or RadiusClient) has been used for the wireless client authentication.</td>
</tr>
<tr>
<td>LinkAuthentication</td>
<td>O</td>
<td>This value indicates that LinkAuthentication service has been used for the wireless client authentication.</td>
</tr>
<tr>
<td>RadiusClient</td>
<td>O</td>
<td>This value indicates that RadiusClient service has been used for the wireless client authentication.</td>
</tr>
<tr>
<td>Vendor-defined</td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>Vendor-defined</td>
<td>O</td>
<td></td>
</tr>
</tbody>
</table>

1 R = Required, O = Optional.

2.2.30.WEPKeyIndex
This variable indicates the index of the WEP key.

2.2.31.WEPKey
This variable represents a WEPkey. It is used only when WEP encryption is enabled; its value consists of the appropriate number of Hex digits, with no punctuation. Case is not significant. If possible, WEP keys should be generated, by the control point, from a passphrase as specified under KeyPassphrase (section 2.2.32). This variable is read/write.

2.2.32.KeyPassphrase
This variable represents a key passphrase (or password) from which the four WEP keys and the default pre-shared key (key 0) will be derived by the control point. The passphrase will be stored by (but not used by) the AP. This variable is read/write.

The default pre-shared key will be derived from the passphrase according to the recommendations in the WPA specification, which uses PBKDF2 from PKCS #5: Password-based Cryptography Specification Version 2.0 (RFC2898).

The WEP keys, if derived from the passphrase, will use following scheme –

For a passphrase that is 5 characters long, the ASCII value of each character is used to generate a 40 bit WEP key. For e.g., "ABCDE" = "100000101000010010000110100010001000101". If the passphrase is 13 characters, a 104 bit WEP key is generated.

For a passphrase that is 10 characters long, each character is required to be in hexadecimal format and a 40 bit WEP key is generated from the hexadecimal characters. For e.g., "4142434445" = "10000101000010010000111010001100010001000101101000110001000100010100010001000101". If the passphrase is 26 characters each character is required to be in hexadecimal format and a 104 bit WEP key is generated from the hexadecimal characters.

2.2.33.WEPEndpointLevel
This variable represents the WEP encryption levels that are supported by the AP. Vendor-defined values must consist of the key length in bits, followed by non-numeric characters. This variable is read/write.

Table 1.4: allowedValueList for WEPEndpointLevel

### Value | Req. or Opt. | Description
--- | --- | ---
**Disabled** | R | WEP encryption is disabled

**40-bit** | R | WEP encryption is enabled, with a 40-bit (sometimes referred to as 64-bit) key

**104-bit** | R | WEP encryption is enabled, with a 104-bit (sometimes referred to as 128-bit) key

**Vendor-defined** | R | R

**Vendor-defined** | O | O

1 R = Required, O = Optional.

#### 2.2.34. PreSharedKey
This variable represents the 256-bit key (64 Hex digits) used when `PSKAuthenticationEnabled` is set to TRUE. This is the network-wide key that is used for WPA and 11i. The default pre-shared key (key 0) must be generated, by the control point, from a passphrase as specified under `KeyPassphrase` (section 2.2.32). This variable is read/write.

Pre-shared keys 1-9 are assumed to be supplied via other mechanisms, e.g. pre-installed on a simple wireless device.

#### 2.2.35. PreSharedKeyId
This variable indicates the index of the pre-shared key. It is used only for the non-default keys and is hence in the range 1-9.

#### 2.2.36. BasicEncryptionModes
This variable indicates the encryption modes that are available when basic 802.11 is enabled. If it is “WEPEncryption” then all wireless clients can use WEP for data encryption. This variable is read/write.

### Table 1.5: Allowed value list for BasicEncryptionModes

<table>
<thead>
<tr>
<th>Value</th>
<th>Req. or Opt.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>R</td>
<td>No encryption is available</td>
</tr>
<tr>
<td>WEPEncryption</td>
<td>R</td>
<td>WEP encryption is available</td>
</tr>
<tr>
<td>Vendor-defined</td>
<td>O</td>
<td>O</td>
</tr>
</tbody>
</table>

1 R = Required, O = Optional.

#### 2.2.37. BasicAuthenticationMode
This variable indicates whether wireless clients can use EAP to authenticate when the AP supports basic 802.11. This variable is read/write.

### Table 1.6: Allowed value list for BasicAuthenticationMode

### Table 1.7: Allowed value list for WPAEncryptionModes

<table>
<thead>
<tr>
<th>Value</th>
<th>Req. or Opt.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>R</td>
<td>There is no authentication</td>
</tr>
<tr>
<td>EAPAuthentication</td>
<td>O</td>
<td>Authentication mode is EAP (802.1x)</td>
</tr>
<tr>
<td>Vendor-defined</td>
<td>O</td>
<td>O</td>
</tr>
</tbody>
</table>

1 R = Required, O = Optional.

#### 2.2.38.WPAEncryptionModes

This variable indicates the encryption modes that are available when WPA is enabled. This variable is read/write.

### Table 1.8: Allowed value list for WPAAuthenticationMode

<table>
<thead>
<tr>
<th>Value</th>
<th>Req. or Opt.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSKAuthentication</td>
<td>R</td>
<td>Pre-shared key authentication is enabled</td>
</tr>
<tr>
<td>EAPAuthentication</td>
<td>O</td>
<td>Authentication mode is EAP (802.1x)</td>
</tr>
<tr>
<td>Vendor-defined</td>
<td>O</td>
<td>O</td>
</tr>
</tbody>
</table>

1 R = Required, O = Optional.

#### 2.2.39.WPAAuthenticationMode

This variable indicates which authentication modes wireless clients can use when WPA is enabled. This variable is read/write.
2.2.40. IEEE11iEncryptionModes
This variable indicates the encryption modes that are available when 11i is enabled. This variable is read/write.

Table 1.9: Allowed value list for IEEE11iEncryptionModes

<table>
<thead>
<tr>
<th>Value</th>
<th>Req. or Opt.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WEPEncryption</td>
<td>R</td>
<td>Only WEP encryption is available in the 11i cipher suite</td>
</tr>
<tr>
<td>TKIPEncryption</td>
<td>R</td>
<td>Only TKIP encryption is available in the 11i cipher suite</td>
</tr>
<tr>
<td>WEPandTKIPEncryption</td>
<td>R</td>
<td>WEP and TKIP encryption modes are available in the 11i cipher suite</td>
</tr>
<tr>
<td>AESEncryption</td>
<td>O</td>
<td>Only AES encryption is available in the 11i cipher suite</td>
</tr>
<tr>
<td>WEPandAESEncryption</td>
<td>O</td>
<td>WEP and AES encryption modes are available in the 11i cipher suite</td>
</tr>
<tr>
<td>TKIPandAESEncryption</td>
<td>O</td>
<td>TKIP and AES encryption modes are available in the 11i cipher suite</td>
</tr>
<tr>
<td>WEPandTKIPandAESEncryption</td>
<td>O</td>
<td>WEP, TKIP and AES encryption modes are available in the 11i cipher suite</td>
</tr>
</tbody>
</table>

Vendor-defined: O

1 R = Required, O = Optional.

2.2.41. IEEE11iAuthenticationMode
This variable indicates which authentication modes wireless clients can use when 11i is enabled. This variable is read/write.

Table 1.10: Allowed value list for IEEE11iAuthenticationMode

<table>
<thead>
<tr>
<th>Value</th>
<th>Req. or Opt.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSKAuthentication</td>
<td>R</td>
<td>Pre-shared key authentication is enabled</td>
</tr>
<tr>
<td>EAPAuthentication</td>
<td>O</td>
<td>Authentication mode is EAP (802.1x)</td>
</tr>
<tr>
<td>EAPandPSKAuthentication</td>
<td>O</td>
<td>Authentication modes are Pre-Shared Key authentication and EAP (802.1x). Wireless client have an option to authenticate itself either way.</td>
</tr>
</tbody>
</table>

Vendor-defined: O

1 R = Required, O = Optional.
2.2.42.LastRequestedUnicastCipher
This variable indicates the unicast cipher that was most recently used for a station with a specific MAC address. This is used in conjunction with action GetSpecificAssociatedDeviceInfo.

2.2.43.LastRequestedMulticastCipher
This variable indicates the multicast cipher that was most recently used for a station with a specific MAC address. This is used in conjunction with action GetSpecificAssociatedDeviceInfo.

2.2.44.LastPMKId
This variable indicates the previous pair-wise master key used between the AP and a station with a specific MAC address.

2.3. Eventing and Moderation
The WLANConfiguration service has a large number of state variables, of which only one is evented. Accordingly, non-evented variables have been omitted from the following table.

Table 2: Event Moderation

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Evented</th>
<th>Moderated Event</th>
<th>Max Event Rate(^1)</th>
<th>Logical Combination</th>
<th>Min Delta per Event(^2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TotalAssociations</td>
<td>Yes</td>
<td>No</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Non-standard state variables implemented by an UPnP™ device vendor go here.</td>
<td>TBD</td>
<td>TBD</td>
<td>TBD</td>
<td>TBD</td>
<td>TBD</td>
</tr>
</tbody>
</table>

\(^1\) Determined by \(N\), where \(\text{Rate} = \frac{\text{Event}}{\text{N secs}}\).

\(^2\) \((N) \times (\text{allowedValueRange} \ \text{Step})\).

2.3.1. Event Model
One of the state variables in the WLANConfiguration service is evented:

TotalAssociations: This state variable event helps to keep the client’s associated device list synchronized with the associated device list maintained at the AP device.

None of the events are moderated.

2.4. Actions
Table 3 lists the required and optional actions for the UPnP™ AP device. This is followed by detailed information about these actions, including short descriptions of the actions, the effects of the actions on state variables, and error codes defined by the actions.

Securing UPnP™ actions in this service is optional but highly recommended, using UPnP™ security protocols as defined by UPnP™ Security working group. If the AP implements security for UPnP™ actions, Table 3 indicates which actions MUST be secure. The others may be implemented as secure or open. Secure actions MUST be protected for both confidentiality and integrity.

Access permissions will be inherited from the containing device (e.g., WLANAccessPointDevice).

Table 3: Actions

<table>
<thead>
<tr>
<th>Name</th>
<th>Secure or Open*</th>
<th>Req. or Opt.</th>
</tr>
</thead>
<tbody>
<tr>
<td>SetInsecureOutOfBandAccessMode</td>
<td>S</td>
<td>O</td>
</tr>
<tr>
<td>GetInsecureOutOfBandAccessMode</td>
<td>S</td>
<td>O</td>
</tr>
<tr>
<td>SetSSID</td>
<td>S</td>
<td>R</td>
</tr>
<tr>
<td>GetSSID</td>
<td>O</td>
<td>R</td>
</tr>
<tr>
<td>GetBSSID</td>
<td>S</td>
<td>R</td>
</tr>
<tr>
<td>SetBeaconType</td>
<td>S</td>
<td>R</td>
</tr>
<tr>
<td>GetBeaconType</td>
<td>O</td>
<td>R</td>
</tr>
<tr>
<td>SetBeaconAdvertisement</td>
<td>S</td>
<td>O</td>
</tr>
<tr>
<td>GetBeaconAdvertisement</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>SetRadioMode</td>
<td>S</td>
<td>R</td>
</tr>
<tr>
<td>GetRadioMode</td>
<td>O</td>
<td>R</td>
</tr>
<tr>
<td>SetLocationDescription</td>
<td>S</td>
<td>O</td>
</tr>
<tr>
<td>GetLocationDescription</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>SetRegulatoryDomain</td>
<td>S</td>
<td>O</td>
</tr>
<tr>
<td>GetRegulatoryDomain</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>GetFailureStatusInfo</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>SetChannel</td>
<td>S</td>
<td>R</td>
</tr>
<tr>
<td>GetChannelInfo</td>
<td>O</td>
<td>R</td>
</tr>
<tr>
<td>GetChannelsInUse</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>SetDeviceOperationMode</td>
<td>S</td>
<td>O</td>
</tr>
<tr>
<td>GetDeviceOperationMode</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>SetDataTransmitRates</td>
<td>S</td>
<td>R</td>
</tr>
<tr>
<td>GetDataTransmitRateInfo</td>
<td>O</td>
<td>R</td>
</tr>
<tr>
<td>SetAutoRateFallBackMode</td>
<td>S</td>
<td>R</td>
</tr>
<tr>
<td>GetAutoRateFallBackMode</td>
<td>O</td>
<td>R</td>
</tr>
<tr>
<td>GetByteStatistics</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>GetPacketStatistics</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>GetByteStatsForAssociatedDev</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>GetPacketStatsForAssociatedDev</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>GetTotalAssociations</td>
<td>S</td>
<td>R</td>
</tr>
<tr>
<td>GetGenericAssociatedDeviceInfo</td>
<td>S</td>
<td>R</td>
</tr>
<tr>
<td>GetSpecificAssociatedDeviceInfo</td>
<td>S</td>
<td>R</td>
</tr>
<tr>
<td>GetSpecificAssociatedDev11iInfo</td>
<td>S</td>
<td>O</td>
</tr>
<tr>
<td>SetAuthenticationServiceMode</td>
<td>S</td>
<td>R</td>
</tr>
<tr>
<td>GetAuthenticationServiceMode</td>
<td>S</td>
<td>R</td>
</tr>
<tr>
<td>SetSecurityKeys</td>
<td>S</td>
<td>R</td>
</tr>
</tbody>
</table>

WLANConfiguration Service:1 Service Template Version 1.01

Name | Secure or Open* | Req. or Opt.
--- | --- | ---
GetSecurityKeys | S | R
SetDefaultWEPKeyIndex | S | R
GetDefaultWEPKeyIndex | S | R
SetPreSharedKey | S | O
GetPreSharedKey | S | O
SetBasBeaconSecurityProperties | O | R
GetBasBeaconSecurityProperties | O | R
SetWPABeaconSecurityProperties | S | R
GetWPABeaconSecurityProperties | O | R
Set11iBeaconSecurityProperties | S | O
Get11iBeaconSecurityProperties | O | O
FactoryDefaultReset | S | O
ResetAuthentication | S | O

1 R = Required, O = Optional, X = Non-standard.

* This column is relevant if Devicesecurity service is present in the container device.

2.4.1. SetInsecureOutOfBandAccessMode

This action is optional and is relevant only if securing of UPnP™ actions is implemented in the AP device. This action allows/disallows access to state variables (that require secure UPnP™ actions) via mechanisms that are not UPnP™ technology based. For example, if InsecureOOBAccessEnabled is set to 0, then the state variables that require secure UPnP™ actions to change their state MUST not be changed through a mechanism such as a Web browser, without using UPnP™ technology.

2.4.1.1. Arguments

Table 4: Arguments for SetInsecureOutOfBandAccessMode

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewInsecureOOBAccessEnabled</td>
<td>IN</td>
<td>InsecureOOBAccessEnabled</td>
</tr>
</tbody>
</table>

2.4.1.2. Dependency on State (if any)

2.4.1.3. Effect on State (if any)

2.4.1.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
<tr>
<td>501</td>
<td>Action Failed</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>

2.4.2. GetInsecureOutOfBandAccessMode
This action is optional and is relevant only if securing of UPnP™ actions is implemented in the AP device. This action retrieves the restriction status for mechanisms (to access state variables) that are not UPnP™ technology based.

2.4.2.1. Arguments
Table 5: Arguments for SetInsecureOutOfBandAccessMode

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewInsecureOOBAccessEnabled</td>
<td>OUT</td>
<td>InsecureOOBAccessEnabled</td>
</tr>
</tbody>
</table>

2.4.2.2. Dependency on State (if any)

2.4.2.3. Effect on State (if any)

2.4.2.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>

2.4.3. SetSSID
This action sets the SSID string. If the control point is a wireless station, in the case of successful request, it will lose the wireless connectivity and will re-establish the connectivity using a new SSID. If the device returns failure, the connectivity will be maintained.

2.4.3.1. Arguments
Table 6: Arguments for SetSSID

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewSSID</td>
<td>IN</td>
<td>SSID</td>
</tr>
</tbody>
</table>

2.4.3.2. Dependency on State (if any)

2.4.3.3. Effect on State (if any)

2.4.3.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
<tr>
<td>501</td>
<td>Action Failed</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
<tr>
<td>605</td>
<td>String Argument Too Long</td>
<td>See UPnP™ Device Architecture section on Control (proposed).</td>
</tr>
</tbody>
</table>
2.4.4. GetSSID
This action retrieves the SSID string of the access point. This action is mainly useful for the wired control points because a wireless control point already knows the SSID.

2.4.4.1. Arguments
Table 7: Arguments for GetSSID

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewSSID</td>
<td>Out</td>
<td>SSID</td>
</tr>
</tbody>
</table>

2.4.4.2. Dependency on State (if any)

2.4.4.3. Effect on State (if any)

2.4.4.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>

2.4.5. GetBSSID
This action retrieves the BSSID or the MAC address of the access point. This action is mainly useful for the wired control points because a wireless control point already knows the BSSID.

2.4.5.1. Arguments
Table 8: Arguments for GetBSSID

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewBSSID</td>
<td>IN</td>
<td>BSSID</td>
</tr>
</tbody>
</table>

2.4.5.2. Dependency on State (if any)

2.4.5.3. Effect on State (if any)

2.4.5.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>
2.4.6. SetBeaconType
This action sets the beacon type (as defined in table 1.4). The AP device will start beaoning according to the type set by this action once BeaconAdvertisementEnabled is set to TRUE.

2.4.6.1. Arguments
Table 9: Arguments for SetBeaconType

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewBeaconType</td>
<td>IN</td>
<td>BeaconType</td>
</tr>
</tbody>
</table>

2.4.6.2. Dependency on State (if any)

2.4.6.3. Effect on State (if any)

2.4.6.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>ErrorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
<tr>
<td>501</td>
<td>Action Failed</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>

2.4.7. GetBeaconType
This action retrieves the beacon type (as defined in table 1.4).

2.4.7.1. Arguments
Table 10: Arguments for GetBeaconType

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewBeaconType</td>
<td>OUT</td>
<td>BeaconType</td>
</tr>
</tbody>
</table>

2.4.7.2. Dependency on State (if any)

2.4.7.3. Effect on State (if any)

2.4.7.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>ErrorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>
2.4.8. SetBeaconAdvertisement
This action enables/disables the beacon advertisements of the AP device. In the case where BeaconAdvertisementEnabled is set to FALSE, the AP device will not beacon and wireless clients have to probe for the access point. This state is useful if end users do not want to broadcast their AP device.

2.4.8.1. Arguments
Table 11: Arguments for SetBeaconAdvertisement

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewBeaconAdvertisementEnabled</td>
<td>IN</td>
<td>BeaconAdvertisementEnabled</td>
</tr>
</tbody>
</table>

2.4.8.2. Dependency on State (if any)

2.4.8.3. Effect on State (if any)

2.4.8.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
<tr>
<td>501</td>
<td>Action Failed</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>

2.4.9. GetBeaconAdvertisement
This action determines whether or not the AP device is beacons.

2.4.9.1. Arguments
Table 12: Arguments for GetBeaconAdvertisement

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewBeaconAdvertisementEnabled</td>
<td>OUT</td>
<td>BeaconAdvertisementEnabled</td>
</tr>
</tbody>
</table>

2.4.9.2. Dependency on State (if any)

2.4.9.3. Effect on State (if any)

2.4.9.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>
2.4.10.SetRadioMode
This action enables/disables the radio of the AP device. In the case where RadioEnabled is set to FALSE, the AP device will not communicate with wireless clients. This state is useful if end users want to shut down the wireless network. Once RadioEnabled is set to FALSE, only wired control points can change RadioEnabled to TRUE.

2.4.10.1.Arguments
Table 13: Arguments for SetRadioMode

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewRadioEnabled</td>
<td>IN</td>
<td>RadioEnabled</td>
</tr>
</tbody>
</table>

2.4.10.2.Dependency on State (if any)

2.4.10.3.Effect on State (if any)

2.4.10.4.Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
<tr>
<td>501</td>
<td>Action Failed</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>

2.4.11.GetRadioMode
This action determines whether or not the AP device’s radio is on.

2.4.11.1.Arguments
Table 14: Arguments for GetRadioMode

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewRadioEnabled</td>
<td>OUT</td>
<td>RadioEnabled</td>
</tr>
</tbody>
</table>

2.4.11.2.Dependency on State (if any)

2.4.11.3.Effect on State (if any)

2.4.11.4.Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>
2.4.12.SetLocationDescription
This action sets the location description of the AP device. The LocationDescription state variable is an XML description of some basic information that is useful in identifying both the access point by name and the physical location. The AP is not expected to parse this description; it will treat it as an opaque string. The location XML schemas are still in the definition stage from multiple standards bodies.

2.4.12.1.Arguments
Table 15: Arguments for SetLocationDescription

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewLocationDescription</td>
<td>IN</td>
<td>LocationDescription</td>
</tr>
</tbody>
</table>

2.4.12.2.Dependency on State (if any)

2.4.12.3.Effect on State (if any)

2.4.12.4.Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
<tr>
<td>605</td>
<td>String Argument Too Long</td>
<td>See UPnP™ Device Architecture section on Control (proposed).</td>
</tr>
</tbody>
</table>

2.4.13.GetLocationDescription
This action retrieves the location description of the AP device. This location description may be used by the map-related applications.

2.4.13.1.Arguments
Table 16: Arguments for GetLocationDescription

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewLocationDescription</td>
<td>OUT</td>
<td>LocationDescription</td>
</tr>
</tbody>
</table>

2.4.13.2.Dependency on State (if any)

2.4.13.3.Effect on State (if any)

2.4.13.4.Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>
2.4.14. SetRegulatoryDomain
This action sets the RegulatoryDomain of the AP device.

2.4.14.1. Arguments
Table 17: Arguments for SetRegulatoryDomain

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewRegulatoryDomain</td>
<td>IN</td>
<td>RegulatoryDomain</td>
</tr>
</tbody>
</table>

2.4.14.2. Dependency on State (if any)

2.4.14.3. Effect on State (if any)

2.4.14.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
<tr>
<td>605</td>
<td>String Argument Too Long</td>
<td>See UPnP™ Device Architecture section on Control (proposed).</td>
</tr>
</tbody>
</table>

2.4.15. GetRegulatoryDomain
This action retrieves the RegulatoryDomain of the AP device.

2.4.15.1. Arguments
Table 18: Arguments for GetRegulatoryDomain

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewRegulatoryDomain</td>
<td>OUT</td>
<td>RegulatoryDomain</td>
</tr>
</tbody>
</table>

2.4.15.2. Dependency on State (if any)

2.4.15.3. Effect on State (if any)

2.4.15.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>

2.4.16. GetFailureStatusInfo
This action retrieves some diagnostic parameters. These diagnostic parameters are:

1. Total number of integrity failures since the soft reset of the AP device.
2. Total number of PSK failures since the soft reset of the AP device.

2.4.16.1. Arguments

Table 19: Arguments for GetFailureStatusInfo

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewTotalIntegrityFailures</td>
<td>OUT</td>
<td>TotalIntegrityFailures</td>
</tr>
<tr>
<td>NewTotalPSKFailures</td>
<td>OUT</td>
<td>TotalPSKFailures</td>
</tr>
</tbody>
</table>

2.4.16.2. Dependency on State (if any)

2.4.16.3. Effect on State (if any)

2.4.16.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>ErrorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>

2.4.17. SetChannel

This action sets the channel/frequency at the AP device. The control point must select the channel from the PossibleChannels list.

2.4.17.1. Arguments

Table 20: Arguments for SetChannel

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewChannel</td>
<td>IN</td>
<td>Channel</td>
</tr>
</tbody>
</table>

2.4.17.2. Dependency on State (if any)

2.4.17.3. Effect on State (if any)

2.4.17.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>ErrorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
<tr>
<td>501</td>
<td>Action Failed</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
<tr>
<td>728</td>
<td>InvalidChannel</td>
<td>The requested Channel is not specified in the PossibleChannels</td>
</tr>
</tbody>
</table>
2.4.18.GetChannelInfo

This action retrieves the channel currently used by the AP device and the list of channels the AP device can be set to use.

2.4.18.1.Arguments

Table 21: Arguments for GetChannelInfo

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewChannel</td>
<td>OUT</td>
<td>Channel</td>
</tr>
<tr>
<td>NewPossibleChannels</td>
<td>OUT</td>
<td>PossibleChannels</td>
</tr>
</tbody>
</table>

2.4.18.2.Dependency on State (if any)

2.4.18.3.Effect on State (if any)

2.4.18.4.Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>

2.4.19.GetChannelsInUse

This action retrieves the channels currently used by the nearby access points. The vendor may choose to collect this information only when the action is executed, hence the possible 501 (Action Failed) error code.

2.4.19.1.Arguments

Table 22: Arguments for GetChannelsInUse

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewChannelsInUse</td>
<td>OUT</td>
<td>ChannelsInUse</td>
</tr>
</tbody>
</table>
2.4.19.2. Dependency on State (if any)

2.4.19.3. Effect on State (if any)

2.4.19.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
<tr>
<td>501</td>
<td>Action Failed</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>

2.4.20. SetDeviceOperationMode

This action sets the operation mode of the access point device. Possible operation modes are listed in Table 1.1. When the value of the argument `NewDeviceOperationMode` is either `InfrastructureAccessPoint` or `WirelessStation`, the rest of the arguments are ignored, i.e., the related state variables are not changed. When the value of the argument `NewDeviceOperationMode` is either `WirelessBridge` or `WirelessRepeater`, the rest of the arguments are used to update the related variables accordingly.

2.4.20.1. Arguments

Table 23: Arguments for SetDeviceOperationMode

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewDeviceOperationMode</td>
<td>IN</td>
<td>DeviceOperationMode</td>
</tr>
<tr>
<td>NewSSID</td>
<td>IN</td>
<td>SSID</td>
</tr>
<tr>
<td>NewPeerBSSID</td>
<td>IN</td>
<td>PeerBSSID</td>
</tr>
<tr>
<td>NewChannel</td>
<td>IN</td>
<td>Channel</td>
</tr>
<tr>
<td>NewBasicDataTransmitRates</td>
<td>IN</td>
<td>BasicDataTransmitRates</td>
</tr>
<tr>
<td>NewOperationalDataTransmitRates</td>
<td>IN</td>
<td>OperationalDataTransmitRates</td>
</tr>
<tr>
<td>NewDistanceFromRoot</td>
<td>IN</td>
<td>DistanceFromRoot</td>
</tr>
</tbody>
</table>

2.4.20.2. Dependency on State (if any)

2.4.20.3. Effect on State (if any)

2.4.20.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>

**2.4.21.** **GetDeviceOperationMode**

This action retrieves the operation mode and related state variables of the access point device. Possible operation modes are listed in Table 1.1. When the value of the argument `NewDeviceOperationMode` is either `InfrastructureAccessPoint` or `WirelessStation`, the rest of the arguments should be returned as empty strings. When `WirelessBridge` or `WirelessRepeater` mode is returned, the rest of the arguments should have the valid values copied from the corresponding state variables.

**2.4.21.1. Arguments**

Table 24: Arguments for GetDeviceOperationMode

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>NewDeviceOperationMode</code></td>
<td>OUT</td>
<td><code>DeviceOperationMode</code></td>
</tr>
<tr>
<td><code>NewSSID</code></td>
<td>OUT</td>
<td><code>SSID</code></td>
</tr>
<tr>
<td><code>NewBSSID</code></td>
<td>OUT</td>
<td><code>PeerBSSID</code></td>
</tr>
<tr>
<td><code>NewChannel</code></td>
<td>OUT</td>
<td><code>Channel</code></td>
</tr>
<tr>
<td><code>NewBasicDataTransmitRates</code></td>
<td>OUT</td>
<td><code>BasicDataTransmitRates</code></td>
</tr>
<tr>
<td><code>NewOperationalDataTransmitRates</code></td>
<td>OUT</td>
<td><code>OperationalDataTransmitRates</code></td>
</tr>
<tr>
<td><code>NewDistanceFromRoot</code></td>
<td>OUT</td>
<td><code>DistanceFromRoot</code></td>
</tr>
</tbody>
</table>

**2.4.21.2. Dependency on State (if any)**

**2.4.21.3. Effect on State (if any)**

**2.4.21.4. Errors**

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>

**2.4.22.** **SetDataTransmitRates**

This action sets the basic and operational data transmission rates of the AP device. The data rates defined in the `NewBasicDataTransmitRates` and `NewOperationalDataTransmitRates` string are applicable by the AP only if they are also present in the `PossibleDataTransmitRates` list.
2.4.22.1. Arguments

Table 25: Arguments for SetDataTransmitRates

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewBasicDataTransmitRates</td>
<td>IN</td>
<td>BasicDataTransmitRates</td>
</tr>
<tr>
<td>NewOperationalDataTransmitRates</td>
<td>IN</td>
<td>OperationalDataTransmitRates</td>
</tr>
</tbody>
</table>

2.4.22.2. Dependency on State (if any)

2.4.22.3. Effect on State (if any)

2.4.22.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
<tr>
<td>730</td>
<td>InvalidDataTransmissionRates</td>
<td>The requested data transmission rates are not specified in the PossibleDataTransmitRates</td>
</tr>
</tbody>
</table>

2.4.23. GetDataTransmitRateInfo

This action retrieves the data transmission rates currently used by the AP device and the list of data rates the AP device can use to set the Basic and operational data rates.

2.4.23.1. Arguments

Table 26: Arguments for GetDataTransmitRateInfo

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewBasicDataTransmitRates</td>
<td>OUT</td>
<td>BasicDataTransmitRates</td>
</tr>
<tr>
<td>NewOperationalDataTransmitRates</td>
<td>OUT</td>
<td>OperationalDataTransmitRates</td>
</tr>
<tr>
<td>NewPossibleDataTransmitRates</td>
<td>OUT</td>
<td>PossibleDataTransmitRates</td>
</tr>
</tbody>
</table>

2.4.23.2. Dependency on State (if any)

2.4.23.3. Effect on State (if any)

2.4.23.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>
2.4.24. SetAutoRateFallBackMode
This action enables/disables the auto fall back rate at the AP device. If it is enabled, the AP device can pick a lower data transmission rate in the case of heavy packet losses.

2.4.24.1. Arguments

Table 27: Arguments for SetAutoRateFallBackMode

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewAutoRateFallBackEnabled</td>
<td>IN</td>
<td>AutoRateFallBackEnabled</td>
</tr>
</tbody>
</table>

2.4.24.2. Dependency on State (if any)

2.4.24.3. Effect on State (if any)

2.4.24.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>

2.4.25. GetAutoRateFallBackMode
This action detects whether or not the AP device is enabled for Auto fallback rate. If it is enabled, the AP device can pick a lower data transmission rate in the case of heavy packet losses.

2.4.25.1. Arguments

Table 28: Arguments for GetAutoRateFallBackMode

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewAutoRateFallBackEnabled</td>
<td>OUT</td>
<td>AutoRateFallBackEnabled</td>
</tr>
</tbody>
</table>

2.4.25.2. Dependency on State (if any)

2.4.25.3. Effect on State (if any)

2.4.25.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>
2.4.26.**GetByteStatistics**
This action retrieves the number of incoming and outgoing bytes processed by the AP device at the wireless interface since the factory-default reset or the most recent reboot of the AP device.

2.4.26.1.**Arguments**

Table 29: Arguments for GetByteStatistics

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewTotalBytesSent</td>
<td>OUT</td>
<td>TotalBytesSent</td>
</tr>
<tr>
<td>NewTotalBytesReceived</td>
<td>OUT</td>
<td>TotalBytesReceived</td>
</tr>
</tbody>
</table>

2.4.26.2.**Dependency on State (if any)**

2.4.26.3.**Effect on State (if any)**

2.4.26.4.**Errors**

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>

2.4.27.**GetPacketStatistics**
This action retrieves the number of incoming and outgoing packets processed by the AP device at the wireless interface since the factory-default reset or the most recent reboot of the AP device.

2.4.27.1.**Arguments**

Table 30: Arguments for GetPacketStatistics

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewTotalPacketsSent</td>
<td>OUT</td>
<td>TotalPacketsSent</td>
</tr>
<tr>
<td>NewTotalPacketsReceived</td>
<td>OUT</td>
<td>TotalPacketsReceived</td>
</tr>
</tbody>
</table>

2.4.27.2.**Dependency on State (if any)**

2.4.27.3.**Effect on State (if any)**

2.4.27.4.**Errors**

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>

2.4.28.**GetByteStatsForAssociatedDev**

This action retrieves the number of incoming and outgoing bytes processed by the AP device at the wireless interface since the factory-default reset or the most recent reboot of the AP device – for a particular station specified by the MAC address parameter.

### 2.4.28.1. Arguments

Table 31: Arguments for GetByteStatsForAssociatedDev

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewAssociatedDeviceMACAddress</td>
<td>IN</td>
<td>AssociatedDeviceMACAddress</td>
</tr>
<tr>
<td>NewTotalBytesSent</td>
<td>OUT</td>
<td>TotalBytesSent</td>
</tr>
<tr>
<td>NewTotalBytesReceived</td>
<td>OUT</td>
<td>TotalBytesReceived</td>
</tr>
</tbody>
</table>

### 2.4.28.2. Dependency on State (if any)

### 2.4.28.3. Effect on State (if any)

### 2.4.28.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>

2.4.29.**GetPacketStatsForAssociatedDev**

This action retrieves the number of incoming and outgoing packets processed by the AP device at the wireless interface since the factory-default reset or the most recent reboot of the AP device – for a particular station specified by the MAC address parameter.

### 2.4.29.1. Arguments

Table 32: Arguments for GetPacketStatsForAssociatedDev

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewAssociatedDeviceMACAddress</td>
<td>IN</td>
<td>AssociatedDeviceMACAddress</td>
</tr>
<tr>
<td>NewTotalPacketsSent</td>
<td>OUT</td>
<td>TotalPacketsSent</td>
</tr>
<tr>
<td>NewTotalPacketsReceived</td>
<td>OUT</td>
<td>TotalPacketsReceived</td>
</tr>
</tbody>
</table>

2.4.29.2. Dependency on State (if any)

2.4.29.3. Effect on State (if any)

2.4.29.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>

2.4.30. GetTotalAssociations

This action retrieves the total number of associated devices.

2.4.30.1. Arguments

Table 33: Arguments for GetTotalAssociations

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewTotalAssociations</td>
<td>OUT</td>
<td>TotalAssociations</td>
</tr>
</tbody>
</table>

2.4.30.2. Dependency on State (if any)

2.4.30.3. Effect on State (if any)

2.4.30.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>

2.4.31. GetGenericAssociatedDeviceInfo

This action retrieves Associated Device Entries one entry at a time. Control points can call this action with an incrementing array index until no more entries are found on the AP. If TotalAssociations is updated during a call, the process may have to start over. Entries in the array are contiguous. As entries are deleted, the array is compacted, and the evented variable TotalAssociations is decremented. Associated Device Entries are logically stored as an array on the AP and retrieved using an array index ranging from 0 to TotalAssociations – 1.

2.4.31.1. Arguments

Table 34: Arguments for GetGenericAssociatedDeviceInfo

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewAssociatedDeviceIndex</td>
<td>IN</td>
<td>TotalAssociations</td>
</tr>
</tbody>
</table>
### 2.4.31.2. Dependency on State (if any)

### 2.4.31.3. Effect on State (if any)

### 2.4.31.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
<tr>
<td>713</td>
<td>InvalidIndex</td>
<td>The specified array index is out of bounds</td>
</tr>
</tbody>
</table>

### 2.4.32. GetSpecificAssociatedDeviceInfo

This action retrieves the Associated Device Entry corresponding to the associated device (WLAN station) with the specified MAC address.

#### 2.4.32.1. Arguments

**Table 35: Arguments for GetSpecificAssociatedDeviceInfo**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewAssociatedDeviceMACAddress</td>
<td>IN</td>
<td>AssociatedDeviceMACAddress</td>
</tr>
<tr>
<td>NewAssociatedDeviceIPAddress</td>
<td>OUT</td>
<td>AssociatedDeviceIPAddress</td>
</tr>
<tr>
<td>NewAssociatedDeviceAuthState</td>
<td>OUT</td>
<td>AssociatedDeviceAuthState</td>
</tr>
</tbody>
</table>

#### 2.4.32.2. Dependency on State (if any)

#### 2.4.32.3. Effect on State (if any)

#### 2.4.32.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
<tr>
<td>714</td>
<td>NoSuchEntryInArray</td>
<td>The specified value does not exist in the array</td>
</tr>
</tbody>
</table>
2.4.33.1.GetSpecificAssociatedDev11iInfo

This action retrieves the IEEE 802.11i related information corresponding to the associated device (WLAN station) with the specified MAC address. The AP in this case would keep information such the last requested unicast and multicast ciphers and the pair-wise master key id used for the different stations on a per MAC address basis. This information can be used for diagnosing configuration problems where there is an incompatibility between what the station is capable of and what the AP will allow. If the LastPMKId is NULL, it is assumed that AP does not have this information or does not use a RADIUS server.

2.4.33.1.1.Arguments

Table 36: Arguments for GetSpecificAssociatedDeviceInfo

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewAssociatedDeviceMACAddress</td>
<td>IN</td>
<td>AssociatedDeviceMACAddress</td>
</tr>
<tr>
<td>NewLastRequestedUnicastCipher</td>
<td>OUT</td>
<td>LastRequestedUnicastCipher</td>
</tr>
<tr>
<td>NewLastRequestedMulticastCipher</td>
<td>OUT</td>
<td>LastRequestedMulticastCipher</td>
</tr>
<tr>
<td>NewIEEE11iAuthenticationMode</td>
<td>OUT</td>
<td>IEEE11iAuthenticationMode</td>
</tr>
<tr>
<td>NewLastPMKId</td>
<td>OUT</td>
<td>LastPMKId</td>
</tr>
</tbody>
</table>

2.4.33.2.Dependency on State (if any)

2.4.33.3.Effect on State (if any)

2.4.33.4.Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
<tr>
<td>714</td>
<td>NoSuchEntryInArray</td>
<td>The specified value does not exist in the array</td>
</tr>
</tbody>
</table>

2.4.34.SetAuthenticationServiceMode

This action sets the authentication service mode of the AP device. For example, the AP device will use the LinkAuthentication service if the AuthenticationServiceMode is set to LinkAuthentication. Similarly, if the AuthenticationServiceMode is None, it means the AP device is not using the LinkAuthentication service or RadiusClient service. In this case either there is no authentication required, or the authentication service mode of the AP device is Pre Shared Key.
2.4.34.1. Arguments

Table 37: Arguments for SetAuthenticationServiceMode

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewAuthenticationServiceMode</td>
<td>IN</td>
<td>AuthenticationServiceMode</td>
</tr>
</tbody>
</table>

2.4.34.2. Dependency on State (if any)

2.4.34.3. Effect on State (if any)

2.4.34.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>ErrorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>

2.4.35. GetAuthenticationServiceMode

This action gets the authentication service mode of the AP device.

2.4.35.1. Arguments

Table 38: Arguments for GetAuthenticationServiceMode

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewAuthenticationServiceMode</td>
<td>OUT</td>
<td>AuthenticationServiceMode</td>
</tr>
</tbody>
</table>

2.4.35.2. Dependency on State (if any)

2.4.35.3. Effect on State (if any)

2.4.35.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>ErrorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>

2.4.36. SetSecurityKeys

This action sets the AP device’s four WEP keys and its default pre-shared key (i.e. key 0). The control point will have derived these keys from a single passphrase, which is stored (but not used) by the AP. The keys are used for data encryption and authentication.
WEP key encryption levels are inferred from their lengths (e.g. 10 Hex digits for 40-bit and 26 Hex digits for 104-bit). Control points can infer the supported key lengths from the allowed value list for the WEPEncryptionLevel state variable.

2.4.36.1.Arguments

Table 39: Arguments for SetSecurityKeys

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewWEPKey0</td>
<td>IN</td>
<td>WEPKey</td>
</tr>
<tr>
<td>NewWEPKey1</td>
<td>IN</td>
<td>WEPKey</td>
</tr>
<tr>
<td>NewWEPKey2</td>
<td>IN</td>
<td>WEPKey</td>
</tr>
<tr>
<td>NewWEPKey3</td>
<td>IN</td>
<td>WEPKey</td>
</tr>
<tr>
<td>NewPreSharedKey</td>
<td>IN</td>
<td>PreSharedKey</td>
</tr>
<tr>
<td>NewKeyPassphrase</td>
<td>IN</td>
<td>KeyPassphrase</td>
</tr>
</tbody>
</table>

2.4.36.2.Dependency on State (if any)

2.4.36.3.Effect on State (if any)

2.4.36.4.Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>ErrorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
<tr>
<td>731</td>
<td>InvalidWEPKey</td>
<td>WEPKey contains invalid characters or its length does not correspond to a supported encryption level.</td>
</tr>
</tbody>
</table>

2.4.37.GetSecurityKeys

This action retrieves the AP device’s four WEP keys, its default pre-shared key, and the passphrase from which they were generated.

2.4.37.1.Arguments

Table 40: Arguments for GetSecurityKeys

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>RelatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewWEPKey0</td>
<td>OUT</td>
<td>WEPKey</td>
</tr>
</tbody>
</table>
### 2.4.37.2. Dependency on State (if any)

### 2.4.37.3. Effect on State (if any)

### 2.4.37.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>

### 2.4.38. SetDefaultWEPKeyIndex

This action sets the index of the default WEP key.

#### 2.4.38.1. Arguments

Table 41: Arguments for SetDefaultWEPKeyIndex

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewDefaultWEPKeyIndex</td>
<td>IN</td>
<td>WEPKeyIndex</td>
</tr>
</tbody>
</table>

#### 2.4.38.2. Dependency on State (if any)

#### 2.4.38.3. Effect on State (if any)

#### 2.4.38.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>
2.4.39.**GetDefaultWEPKeyIndex**

This action retrieves the index of the default WEP key.

2.4.39.1.**Arguments**

Table 42: Arguments for GetDefaultWEPKeyIndex

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewDefaultWEPKeyIndex</td>
<td>OUT</td>
<td>WEPKeyIndex</td>
</tr>
</tbody>
</table>

2.4.39.2.**Dependency on State (if any)**

2.4.39.3.**Effect on State (if any)**

2.4.39.4.**Errors**

<table>
<thead>
<tr>
<th>errorCode</th>
<th>ErrorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>

2.4.40.**SetPreSharedKey**

This action sets the pre-shared key at the specified index at the AP device. After setting the pre-shared key at the AP device, wireless clients can use the PSK authentication mechanism to mutually authenticate themselves to the AP device. A UPnP™ AP device supports 10 pre-shared keys, of which key 0 is the default. This action is used only for setting the non-default keys (keys 1-9). The wireless client may specify a MAC address to be associated with the pre-shared key. If a NULL is passed for NewAssociatedDeviceMACAddress, the AP will not associate this PSK with a MAC address.

2.4.40.1.**Arguments**

Table 43: Arguments for SetPreSharedKey

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewPreSharedKeyIndex</td>
<td>IN</td>
<td>PreSharedKeyIndex</td>
</tr>
<tr>
<td>NewAssociatedDeviceMACAddress</td>
<td>IN</td>
<td>NewAssociatedDeviceMACAddress</td>
</tr>
<tr>
<td>NewPreSharedKey</td>
<td>IN</td>
<td>PreSharedKey</td>
</tr>
<tr>
<td>NewPSKPassphrase</td>
<td>IN</td>
<td>KeyPassphrase</td>
</tr>
</tbody>
</table>
2.4.40.2. Dependency on State (if any)

2.4.40.3. Effect on State (if any)

2.4.40.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>ErrorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>

2.4.41. GetPreSharedKey

This action gets the pre-shared key of the specified index. This action is used only for getting non-default keys (keys 1-9). If the AP does not support association of PSK with a MAC address, a NULL is returned for NewAssociatedDevMACAddress.

2.4.41.1. Arguments

Table 44: Argument for GetPreSharedKey

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewPreSharedKeyIndex</td>
<td>IN</td>
<td>PreSharedKeyIndex</td>
</tr>
<tr>
<td>NewAssociatedDeviceMACAddress</td>
<td>OUT</td>
<td>NewAssociatedDeviceMACAddress</td>
</tr>
<tr>
<td>NewPreSharedKey</td>
<td>OUT</td>
<td>PreSharedKey</td>
</tr>
<tr>
<td>NewPSKPassphrase</td>
<td>OUT</td>
<td>KeyPassphrase</td>
</tr>
</tbody>
</table>

2.4.41.2. Dependency on State (if any)

2.4.41.3. Effect on State (if any)

2.4.41.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>ErrorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>

2.4.42. SetBasBeaconSecurityProperties

This function does the following:

1. The BasicEncryptionModes state variable sets the encryption mode of the basic beacon type. BasicEncryptionModes has two allowed values for encryption mode, OPEN and WEPEncryption. Table 1.5 shows the allowed value list for BasicEncryptionModes.
2. The BasicAuthenticationMode state variable sets the authentication mode of basic beacon. BasicAuthenticationMode has two allowed values for authentication mode, OPEN and EAPAuthentication. UPnP™ AP devices must support OPEN authentication mode. Table 1.6 shows the allowed value list for BasicAuthenticationMode.

Note: Basic beacon does not contain an element bit for EAP Authentication. In this case, Authenticator will start 802.1x once a wireless client is associated with the Access point.

2.4.42.1. Arguments

Table 45: Arguments for SetBasBeaconSecurityProperties

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>RelatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewBasicEncryptionModes</td>
<td>IN</td>
<td>BasicEncryptionModes</td>
</tr>
<tr>
<td>NewBasicAuthenticationMode</td>
<td>IN</td>
<td>BasicAuthenticationMode</td>
</tr>
</tbody>
</table>

2.4.42.2. Dependency on State (if any)

2.4.42.3. Effect on State (if any)

2.4.42.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
<tr>
<td>732</td>
<td>NoWEPKeyIsSet</td>
<td>The complete WEP key set is empty (and EAP WEP re-keying is disabled).</td>
</tr>
<tr>
<td>734</td>
<td>NoEAPServer</td>
<td>No EAP server exists for EAPAuthentication.</td>
</tr>
</tbody>
</table>

2.4.43. GetBasBeaconSecurityProperties

This function gets the encryption and authentication suite of Basic beacon

2.4.43.1. Arguments

Table 46: Arguments for GetBasBeaconSecurityProperties

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>RelatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewBasicEncryptionModes</td>
<td>OUT</td>
<td>BasicEncryptionModes</td>
</tr>
<tr>
<td>NewBasicAuthenticationMode</td>
<td>OUT</td>
<td>BasicAuthenticationMode</td>
</tr>
</tbody>
</table>
2.4.43.2. Dependency on State (if any)

2.4.43.3. Effect on State (if any)

2.4.43.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>

2.4.44. SetWPABeaconSecurityProperties

This function does the following:

1. The WPAEncryptionModes state variable sets the cipher suite of WPA beacon. Table 1.7 shows the allowed value list for WPAAuthenticationMode. UPnP™ AP devices must support WEPEncryption and PSKEncryption in WPA beacon.
2. The WPAAuthenticationMode state variable sets the authentication mode of WPA beacon. WPAAuthenticationMode has two allowed values for authentication mode, PSKAuthentication and EAPAuthentication. At any instance only one type of authentication will be enabled. Table 1.8 shows the allowed value list for WPAAuthenticationMode.

2.4.44.1. Arguments

Table 47: Arguments for SetWPABeaconSecurityProperties

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>RelatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewWPAEncryptionModes</td>
<td>IN</td>
<td>WPAEncryptionModes</td>
</tr>
<tr>
<td>NewWPAAuthenticationMode</td>
<td>IN</td>
<td>WPAAuthenticationMode</td>
</tr>
</tbody>
</table>

2.4.44.2. Dependency on State (if any)

2.4.44.3. Effect on State (if any)

2.4.44.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
<tr>
<td>732</td>
<td>NoWEPKeyIsSet</td>
<td>The complete WEP key set is empty (and EAP WEP re-keying is disabled).</td>
</tr>
<tr>
<td>733</td>
<td>NoPSKKeyIsSet</td>
<td>The complete PSK key set is empty.</td>
</tr>
<tr>
<td>734</td>
<td>NoEAPServer</td>
<td>No EAP server exists for EAPAuthentication.</td>
</tr>
</tbody>
</table>

2.4.45. GetWPABeaconSecurityProperties

This function gets the encryption and authentication suite of WPA beacon.

2.4.45.1. Arguments

Table 48: Arguments for GetWPABeaconSecurityProperties

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>RelatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewWPAEncryptionModes</td>
<td>OUT</td>
<td>WPAEncryptionModes</td>
</tr>
<tr>
<td>NewWPAAuthenticationMode</td>
<td>OUT</td>
<td>WPAAuthenticationMode</td>
</tr>
</tbody>
</table>

2.4.45.2. Dependency on State (if any)

2.4.45.3. Effect on State (if any)

2.4.45.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>

2.4.46. Set11iBeaconSecurityProperties

This function does the following:

1. The IEEE11iEncryptionModes state variable sets the cipher suite of 11i beacon. Table 1.9 shows the allowed value list for IEEE11iEncryptionModes. UPnP™ AP devices must support WEPencryption and TKIPEncryption in 11i beacon.
2. The IEEE11iAuthenticationMode state variable sets the authentication mode of 11i beacon. Table 1.10 shows the allowed value list for IEEE11iAuthenticationMode.

2.4.46.1. Arguments

Table 49: Arguments for Set11iBeaconSecurityProperties

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>RelatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewIEEE11iEncryptionModes</td>
<td>IN</td>
<td>IEEE11iEncryptionModes</td>
</tr>
<tr>
<td>NewIEEE11iAuthenticationMode</td>
<td>IN</td>
<td>IEEE11iAuthenticationMode</td>
</tr>
</tbody>
</table>

2.4.46.2. Dependency on State (if any)

2.4.46.3. Effect on State (if any)

2.4.46.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
<tr>
<td>733</td>
<td>NoPSKKeyisSet</td>
<td>The complete PSK key set is empty.</td>
</tr>
</tbody>
</table>

734 | NoEAPServer | No EAP server exists for EAPAuthentication.

### 2.4.47. Get11iBeaconSecurityProperties

This function gets the encryption and authentication suite of 11i beacon.

#### 2.4.47.1. Arguments

**Table 50: Arguments for Get11iBeaconSecurityProperties**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>RelatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewIEEE11iEncryptionModes</td>
<td>OUT</td>
<td>IEEE11iEncryptionModes</td>
</tr>
<tr>
<td>NewIEEE11iAuthenticationMode</td>
<td>OUT</td>
<td>IEEE11iAuthenticationMode</td>
</tr>
</tbody>
</table>

#### 2.4.47.2. Dependency on State (if any)

#### 2.4.47.3. Effect on State (if any)

#### 2.4.47.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>

### 2.4.48. FactoryDefaultReset

This action resets all the state variables of the AP device to their factory default settings. The AP will disassociate all the wireless clients and the associated device list will become empty. This action also resets all the wireless sessions that were authenticated via WEP, PSK or EAP.

This action must be invoked internally if the control point calls the “FactoryDefaultReset” action of Devicesecurity (if Devicesecurity is implemented), whereas vice versa is not true i.e., resetting this service will not invoke the Devicesecurity reset. This action must internally invoke the FactoryDefaultReset functions of the LinkAuthentication and RadiusClient services if they are present inside the AP device.

#### 2.4.48.1. Arguments

None
2.4.48.2. Dependency on State (if any)

2.4.48.3. Effect on State (if any)

2.4.48.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>

2.4.49. ResetAuthentication

This action resets all wireless stations that were authenticated via WEP, PSK or EAP. Additionally this action removes all the WEP and WPA keys in the WLANConfiguration service, forcing new keys to be generated or reintroduced to the AP device. This action must internally invoke the ResetAuthentication functions of the LinkAuthentication and RadiusClient services if they are present inside the AP device.

2.4.49.1. Arguments

None

2.4.49.2. Dependency on State (if any)

2.4.49.3. Effect on State (if any)

2.4.49.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>

2.4.50. Non-Standard Actions Implemented by a UPnP™ Device Vendor

To facilitate certification, non-standard actions implemented by UPnP™ device vendors should be included in this service template. The UPnP™ Device Architecture lists naming requirements for non-standard actions (see the section on Description).

2.4.51. Common Error Codes

The following table lists error codes common to actions for this service type. If an action results in multiple errors, the most specific error should be returned.

Table 51: Common Error Codes for all actions

<table>
<thead>
<tr>
<th>ErrorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Invalid Action</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
<tr>
<td>404</td>
<td>Invalid Var</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
<tr>
<td>501</td>
<td>Action Failed</td>
<td>See UPnP™ Device Architecture section on Control.</td>
</tr>
</tbody>
</table>

### 2.5. Theory of Operation

#### 2.5.1. Initialization of the AP Device

**2.5.1.1. Setting up non-security parameters on the Access Point**

Factory default setting of the AP device can be changed using a UPnP™ control point. SSID, DeviceOperationMode, Channel, DataTransmitRates, AutoRateFailBackEnabled, and BeaconEnabled can be changed to the end user’s/application’s desired values.

**2.5.1.2. Setting up link security parameters on the Access Point**

There are some assumptions regarding the link security setup at the AP device:

1. An AP can concurrently support more than one specification (‘Basic’, ‘WPA’ and ‘11i’). In other words, an AP can concurrently beacon more than one beacon type, although it is possible that an AP vendor may not support multiple beacon modes simultaneously.
2. Per the UPnP™ AP DCP requirements, AP vendors MUST support ‘Basic’ and ‘WPA’ beacon modes, but 11i is optional.
3. Multiple encryption modes can be enabled with any beacon type. For example, ‘WPA’ Beacon can support any combination of WEPEncryption, TKIPEncryption, and AESEncryption.
4. ‘Basic’ has a support of no encryption and authentication modes. In other words, basic beacon may not have encryption and authentication modes enabled.
5. There are several encryption modes available in 802.11i – CCMP, WRAP etc. Allowing a control point to select the set of encryption modes is too much detail and outside the scope of WLANConfiguration service.
6. AP vendors MUST support WEP Encryption and TKIP encryption, but AES encryption is optional.
7. AP vendors MUST support pre-shared key authentication, but EAP is optional.
8. Unicast and Broadcast encryption modes are not independently selectable.
9. If a control point changes the beacon type completely, for example from ‘Basic’ to ‘WPA’, then all the associated wireless clients using basic beacon MUST be disassociated by the AP. If the beacon set is enhanced, for example from ‘Basic’ to ‘BasicandWPA’, the associated wireless clients using ‘Basic’ beacon will remain associated.

The WEP/WPA/11i vendor-specific beaconing can be set by the SetBeaconType action. These beacons and their encryption and authentication modes can be enabled using three actions, SetBasBeaconSecurityProperties, SetWPABeaconSecurityProperties, and Set11iBeaconSecurityProperties. The beacon will advertise the selected link security capabilities.

A UPnP™ enabled AP device allows only one authentication service. This mutual exclusion is done via the AuthenticationServiceMode variable. If the AuthenticationServiceMode is None then no other service is being used for authentication. In other words, there is no EAP authentication.

2.5.2. Operation of the AP Device

2.5.2.1. Setting up Link Security

An AP vendor has the option to provide Set11iBeaconSecurityProperties and Get11iBeaconSecurityProperties actions in the WLANConfiguration service. A control point will do the following (not necessarily in this order):

1. Set (change) the link security properties of one or more beacons based upon the options given in the allowed value list of encryption and authentication related state variables.
2. Set (change) the beacon type based upon the options given in allowed value list of BeaconType state variable.
3. Set (change) the beacon advertisement.

2.5.2.2. Setting up wireless repeaters

Currently, there is no standardized programmatic means to configure IEEE 802.11 access points into wireless repeater or bridge mode over the network. UPnP™ provides a consistent method to configure wireless repeaters and bridges. However, access points from different vendors may still not interoperate in the wireless repeater or bridge mode of operation. Inter-AP protocol is not covered in this document, and it assumes that either the access points are from the same vendor, or that the inter-AP mechanisms for interoperability will be present. In addition, SetDeviceOperationMode includes a sufficient set of arguments to set up APs from any vendor in repeater or bridge mode. The action may need to set some vendor-specific variables to their default values. But control points that configure access points into repeater or bridge mode behave the same way for access points from different vendors. Control points call the same actions with the same set of arguments. The actions turn wireless access point devices from Infrastructure mode into WirelessRepeater or WirelessBridge mode. Control points have to be granted access rights to secure actions on the access point device if UPnP™ security is implemented on the AP device. Control points are responsible for configuring wireless repeaters and bridges, i.e., they invoke actions to turn wireless access point devices into WirelessRepeater or WirelessBridge mode.

Wireless repeaters extend the radio coverage of IEEE 802.11 infrastructure networks as shown in Figure 1. The objective is to enable UPnP™ enabled wireless access points that implement the optional WirelessRepeater mode to be set up as repeaters with little user effort. The actions used for switching a wireless access point device into WirelessRepeater or WirelessBridge mode may be secured via UPnP™ security. Figure 1 depicts a situation where a user wants to set up AP2 as the wireless repeater to AP1. We assume that the AP2 is placed next to AP1 so that it is within the radio range of AP1 and “setup” control point (SCP). If UPnP™ security is implemented then please refer to the Theory of Operation in the WLANAccessPointDevice document for details on assigning access to secure actions. As far as the user is concerned, a repeater setup application needs to be executed on the SCP while making sure that the two APs are in the range of the SCP to begin with. Once notified of successful completion of the repeater setup, the user places AP2 in a preferred location within the range of AP1.
Details of the mechanism - When the repeater setup application begins, it obtains the SSID, BSSID, Channel, BasicDataTransmitRates, OperationalDataTransmitRates, and link security parameters from AP1 via UPnP™ actions. The repeater setup application then gets the SCP to disassociate from AP1 and associate with AP2. The SCP then obtains BSSID of AP2 through UPnP™ actions. The application invokes the actions to set the desired link security levels and then invokes the SetDeviceOperationMode action on AP2 with SSID, Channel, BSSID, BasicDataTransmitRates and OperationalDataTransmitRates of AP1, and 1 for DistanceFromRoot. The SCP disassociates from AP2 and then associates back to AP1. It invokes SetDeviceOperationMode on AP1 with the same SSID, Channel, OperationalDataTransmitRates, and BasicDataTransmitRates values used for SetDeviceOperationMode on AP2, but with AP2’s BSSID and 0 for DistanceFromRoot. Now AP1 operates in (root) WirelessRepeater mode and AP2 operates in (non-root) WirelessRepeater mode. The new parameter values that were used for setting up AP1 and AP2 are stored in the control point for access points that might be added to the network afterwards.

This mechanism is also applicable in setting up multiple repeaters in a linear topology as shown in Figure 2. It is assumed that APn is being added to a wireless repeater network consisting of AP1 (root), AP2, ..., AP(n-1). When APn is turned on, it is in InfrastructureAccessPoint mode. The user runs the repeater
setup application on SCP while making sure that the SCP is in the radio range of APn and AP(n-1). When the application completes its task, the user may physically move APn to a preferred location within range of AP(n-1).

Details of the mechanism - When the repeater setup application begins, it obtains the SSID, BSSID, Channel, BasicDataTransmitRates, OperationalDataTransmitRates, and link security parameters from AP(n-1). The repeater application may optimize this by storing the repeater configuration parameters when each access point has been set up. In linear topologies, only the parameters of the last access point repeater need to be stored. The setup application then gets the SCP to associate with APn. The SCP then obtains the BSSID of APn through UPnP™ actions. The application invokes the actions to set the desired link security level and then invokes the SetDeviceOperationMode action on APn with SSID, BSSID, OperationalDataTransmitRates, and BasicDataTransmitRates values of AP(n-1), and n for DistanceFromRoot. The value n is obtained from the DistanceFromRoot variable of AP(n-1) using GetDeviceOperationMode. The repeater information of APn may be stored in SCP for the possible addition of another repeater. The PeerBSSID of AP(n-1) need not be updated. AP(n-1) should keep the BSSID of AP(n-2) in its PeerBSSID state variable. The value was set when AP(n-1) was being attached to AP(n-2) as a wireless repeater. In other words, the PeerBSSID state variable stores the MAC address of the upstream access point in the linear topology of wireless repeaters.

Figure 2. Wireless repeater setup (linear topology)

Figure 3 depicts a wireless repeater setup in a star topology. We assume that APn is being added to a wireless repeater network consisting of AP1 (root), AP2, ..., AP(n-1). When APn is turned on, it is in
InfrastructureAccessPoint mode. The user runs the repeater setup application on SCP while making sure that the SCP is in the radio range of APn and AP1. When the application completes its task, the user may physically move APn to a preferred location within range of AP1.

Details of the mechanism - When the repeater setup application begins, it obtains the SSID, BSSID, Channel, DataTransmitRates, and link security parameters from AP1 (via UPnP™ actions or from its storage). The setup application then gets the SCP to associate with APn. The SCP then obtains the BSSID of APn through UPnP™ actions. The application invokes the actions to set the desired link security levels and then invokes the SetDeviceOperationMode action on APn with SSID, BSSID, Channel, OperationalDataTransmitRates, and BasicDataTransmitRates values of AP1, and 1 for DistanceFromRoot.

It is also possible to set up a network of repeaters in any topology, but the task is up to the setup application on the SCP. The setup application must use SetDeviceOperationMode and GetDeviceOperationMode with proper argument values to correctly set up the network according to the intention of the user.

![Diagram of wireless repeater setup (star topology)](image_url)

**Figure 3.** Wireless repeater setup (star topology)
2.5.2.3. Setting up wireless bridges

Wireless bridges interconnect multiple LAN segments as shown in Figure 4. UPnP™ enabled wireless access points that implement optional WirelessBridge mode can be set up as bridges. We assume that the user has a wireless SCP to configure access points into the wireless bridge mode. We also assume that the new APs are within radio range of each other and also of the SCP. The user executes the bridge setup application on the SCP, and then selects the SSIDs of both APs. In a few moments, the application informs the user of the successful bridge setup.

Details of the mechanism - When the bridge setup application begins, it establishes a connection with AP1 and obtains the SSID, BSSID, Channel, BasicDataTransmitRates, OperationalDataTransmitRates, and link security parameters from AP1 via UPnP™ actions. The bridge setup application then gets the SCP to disassociate from AP1 and associate with AP2. The SCP obtains the BSSID of AP2 through UPnP™ actions. The application invokes the actions to set the desired link security levels and then invokes the SetDeviceOperationMode action on AP2 with SSID, Channel, BSSID, BasicDataTransmitRates of AP1, and 1 for DistanceFromRoot. The SCP disassociates from AP2 then associates back to AP1. It invokes SetDeviceOperationMode on AP1 with the same SSID, Channel, and BasicDataTransmitRates values used for SetDeviceOperationMode on AP2 but with AP2’s BSSID, and 0 for DistanceFromRoot. Now AP1 operates in (root) WirelessBridge mode and AP2 operates in (non-root) WirelessBridge mode.

Figure 4. Wireless bridge setup (point-to-point)
If multipoint bridges are being set up as shown in Figure 5 below, only one AP is configured into the root bridge with an empty PeerBSSID. Other (non-root) bridges are configured to have the root’s MAC address (BSSID).

![Diagram of wireless bridge setup (multipoint)](image)

Figure 5. Wireless bridge setup (multipoint)

Although the scenarios described above involve configuration based upon wireless SCP, SCPs on wired LAN segments can also perform wireless bridge setup procedures using the same SetDeviceOperationMode action. However, this may require that the information about one access point be physically transferred to the other LAN segment so that the other access point can acquire that information. For example, the user runs the SCP on a notebook computer that is plugged into one LAN segment and then is plugged into the other LAN segment.

2.5.2.4. Gathering statistics of the AP device

Control points can gather the packet and bytes statistics by retrieving the four parameters TotalPacketsSent, TotalPacketsReceived, TotalBytesSent and TotalBytesReceived. UPnP™ enabled AP devices also maintain the associated wireless client list with their authentication state, MAC address and IP address. If the AP supports it, the control points can also retrieve the number of incoming and outgoing bytes and packets processed by the AP device at the wireless interface for a particular station.
2.5.2.5. Security Diagnostics of the AP Device
TotalPSKFailures and TotalIntegrity failures give an overview of possible malicious attacks on the AP device.

3. XML Service Description

```xml
<?xml version="1.0"?>
<scpd xmlns="urn:schemas-upnp-org:service-1-0">
  <specVersion>
    <major>1</major>
    <minor>0</minor>
  </specVersion>
  <actionList>
    <action>
      <name>SetInsecureOutOfBandAccessMode</name>
      <argumentList>
        <argument>
          <name>NewInsecureOOBAccessEnabled</name>
          <direction>in</direction>
          <relatedStateVariable>InsecureOOBAccessEnabled</relatedStateVariable>
        </argument>
      </argumentList>
    </action>
    <action>
      <name>GetInsecureOutOfBandAccessMode</name>
      <argumentList>
        <argument>
          <name>NewInsecureOOBAccessEnabled</name>
          <direction>out</direction>
          <relatedStateVariable>InsecureOOBAccessEnabled</relatedStateVariable>
        </argument>
      </argumentList>
    </action>
    <action>
      <name>SetSSID</name>
      <argumentList>
        <argument>
          <name>NewSSID</name>
          <direction>in</direction>
          <relatedStateVariable>SSID</relatedStateVariable>
        </argument>
      </argumentList>
    </action>
    <action>
      <name>GetSSID</name>
      <argumentList>
        <argument>
          <name>NewSSID</name>
          <direction>out</direction>
          <relatedStateVariable>SSID</relatedStateVariable>
        </argument>
      </argumentList>
    </action>
    <action>
      <name>GetBSSID</name>
      <argumentList>
        <argument>
          <name>NewBSSID</name>
          <direction>out</direction>
          <relatedStateVariable>BSSID</relatedStateVariable>
        </argument>
      </argumentList>
    </action>
    <action>
      <name>SetBeaconType</name>
      <argumentList>
        <argument>
          <name>NewBeaconType</name>
        </argument>
      </argumentList>
    </action>
  </actionList>
</scpd>
```

<action>
<name>GetBeaconType</name>
<argumentList>
<argument>
<name>NewBeaconType</name>
<direction>out</direction>
<relatedStateVariable>BeaconType</relatedStateVariable>
</argument>
</argumentList>
</action>

<action>
<name>SetBeaconAdvertisement</name>
<argumentList>
<argument>
<name>NewBeaconAdvertisementEnabled</name>
<direction>in</direction>
<relatedStateVariable>BeaconAdvertisementEnabled</relatedStateVariable>
</argument>
</argumentList>
</action>

<action>
<name>GetBeaconAdvertisement</name>
<argumentList>
<argument>
<name>NewBeaconAdvertisementEnabled</name>
<direction>out</direction>
<relatedStateVariable>BeaconAdvertisementEnabled</relatedStateVariable>
</argument>
</argumentList>
</action>

<action>
<name>SetRadioMode</name>
<argumentList>
<argument>
<name>NewRadioEnabled</name>
<direction>in</direction>
<relatedStateVariable>RadioEnabled</relatedStateVariable>
</argument>
</argumentList>
</action>

<action>
<name>GetRadioMode</name>
<argumentList>
<argument>
<name>NewRadioEnabled</name>
<direction>out</direction>
<relatedStateVariable>RadioEnabled</relatedStateVariable>
</argument>
</argumentList>
</action>

<action>
<name>SetLocationDescription</name>
<argumentList>
<argument>
<name>NewLocationDescription</name>
<direction>in</direction>
<relatedStateVariable>LocationDescription</relatedStateVariable>
</argument>
</argumentList>
</action>

<action>
<name>GetLocationDescription</name>
<argumentList>
<argument>
<name>NewLocationDescription</name>
<direction>out</direction>
<relatedStateVariable>LocationDescription</relatedStateVariable>
</argument>
</argumentList>
</action>
<action>
  <name>SetRegulatoryDomain</name>
  <argumentList>
    <argument>
      <name>NewRegulatoryDomain</name>
      <direction>in</direction>
      <relatedStateVariable>RegulatoryDomain</relatedStateVariable>
    </argument>
  </argumentList>
</action>

<action>
  <name>GetRegulatoryDomain</name>
  <argumentList>
    <argument>
      <name>NewRegulatoryDomain</name>
      <direction>out</direction>
      <relatedStateVariable>RegulatoryDomain</relatedStateVariable>
    </argument>
  </argumentList>
</action>

<action>
  <name>GetFailureStatusInfo</name>
  <argumentList>
    <argument>
      <name>NewTotalIntegrityFailures</name>
      <direction>out</direction>
      <relatedStateVariable>TotalIntegrityFailures</relatedStateVariable>
    </argument>
    <argument>
      <name>NewTotalPSKFailures</name>
      <direction>out</direction>
      <relatedStateVariable>TotalPSKFailures</relatedStateVariable>
    </argument>
  </argumentList>
</action>

<action>
  <name>SetChannel</name>
  <argumentList>
    <argument>
      <name>NewChannel</name>
      <direction>in</direction>
      <relatedStateVariable>Channel</relatedStateVariable>
    </argument>
  </argumentList>
</action>

<action>
  <name>GetChannelInfo</name>
  <argumentList>
    <argument>
      <name>NewChannel</name>
      <direction>out</direction>
      <relatedStateVariable>Channel</relatedStateVariable>
    </argument>
    <argument>
      <name>NewPossibleChannels</name>
      <direction>out</direction>
      <relatedStateVariable>PossibleChannels</relatedStateVariable>
    </argument>
  </argumentList>
</action>

<action>
  <name>GetChannelsInUse</name>
  <argumentList>
    <argument>
      <name>NewChannelsInUse</name>
      <direction>out</direction>
      <relatedStateVariable>ChannelsInUse</relatedStateVariable>
    </argument>
  </argumentList>
</action>

<action>
  <name>SetDeviceOperationMode</name>
  <argumentList>
  </argumentList>
</action>
<argument>
  <name>NewDeviceOperationMode</name>
  <direction>in</direction>
  <relatedStateVariable>DeviceOperationMode</relatedStateVariable>
</argument>
<argument>
  <name>NewSSID</name>
  <direction>in</direction>
  <relatedStateVariable>SSID</relatedStateVariable>
</argument>
<argument>
  <name>NewBSSID</name>
  <direction>in</direction>
  <relatedStateVariable>PeerBSSID</relatedStateVariable>
</argument>
<argument>
  <name>NewChannel</name>
  <direction>in</direction>
  <relatedStateVariable>Channel</relatedStateVariable>
</argument>
<argument>
  <name>NewBasicDataTransmitRates</name>
  <direction>in</direction>
  <relatedStateVariable>BasicDataTransmitRates</relatedStateVariable>
</argument>
<argument>
  <name>NewOperationalDataTransmitRates</name>
  <direction>in</direction>
  <relatedStateVariable>OperationalDataTransmitRates</relatedStateVariable>
</argument>
<argument>
  <name>NewDistanceFromRoot</name>
  <direction>in</direction>
  <relatedStateVariable>DistanceFromRoot</relatedStateVariable>
</argument>
</argumentList>
</action>

<action>
  <name>GetDeviceOperationMode</name>
  <argumentList>
    <argument>
      <name>NewDeviceOperationMode</name>
      <direction>out</direction>
      <relatedStateVariable>DeviceOperationMode</relatedStateVariable>
    </argument>
    <argument>
      <name>NewSSID</name>
      <direction>out</direction>
      <relatedStateVariable>SSID</relatedStateVariable>
    </argument>
    <argument>
      <name>NewBSSID</name>
      <direction>out</direction>
      <relatedStateVariable>PeerBSSID</relatedStateVariable>
    </argument>
    <argument>
      <name>NewChannel</name>
      <direction>out</direction>
      <relatedStateVariable>Channel</relatedStateVariable>
    </argument>
    <argument>
      <name>NewBasicDataTransmitRates</name>
      <direction>out</direction>
      <relatedStateVariable>BasicDataTransmitRates</relatedStateVariable>
    </argument>
    <argument>
      <name>NewOperationalDataTransmitRates</name>
      <direction>out</direction>
      <relatedStateVariable>OperationalDataTransmitRates</relatedStateVariable>
    </argument>
    <argument>
      <name>NewDistanceFromRoot</name>
      <direction>out</direction>
      <relatedStateVariable>DistanceFromRoot</relatedStateVariable>
    </argument>
  </argumentList>
</action>

<action>
  <name>SetDataTransmitRates</name>
  <argumentList>
    <argument>
      <name>NewBasicDataTransmitRates</name>
      <direction>in</direction>
      <relatedStateVariable>BasicDataTransmitRates</relatedStateVariable>
    </argument>
    <argument>
      <name>NewOperationalDataTransmitRates</name>
      <direction>in</direction>
      <relatedStateVariable>OperationalDataTransmitRates</relatedStateVariable>
    </argument>
  </argumentList>
</action>

<action>
  <name>GetDataTransmitRateInfo</name>
  <argumentList>
    <argument>
      <name>NewBasicDataTransmitRates</name>
      <direction>out</direction>
      <relatedStateVariable>BasicDataTransmitRates</relatedStateVariable>
    </argument>
    <argument>
      <name>NewOperationalDataTransmitRates</name>
      <direction>out</direction>
      <relatedStateVariable>OperationalDataTransmitRates</relatedStateVariable>
    </argument>
    <argument>
      <name>NewPossibleDataTransmitRates</name>
      <direction>out</direction>
      <relatedStateVariable>PossibleDataTransmitRates</relatedStateVariable>
    </argument>
  </argumentList>
</action>

<action>
  <name>SetAutoRateFallBackMode</name>
  <argumentList>
    <argument>
      <name>NewAutoRateFallBackEnabled</name>
      <direction>in</direction>
      <relatedStateVariable>AutoRateFallBackEnabled</relatedStateVariable>
    </argument>
  </argumentList>
</action>

<action>
  <name>GetAutoRateFallBackMode</name>
  <argumentList>
    <argument>
      <name>NewAutoRateFallBackEnabled</name>
      <direction>out</direction>
      <relatedStateVariable>AutoRateFallBackEnabled</relatedStateVariable>
    </argument>
  </argumentList>
</action>

<action>
  <name>GetByteStatistics</name>
  <argumentList>
    <argument>
      <name>NewTotalBytesSent</name>
      <direction>out</direction>
      <relatedStateVariable>TotalBytesSent</relatedStateVariable>
    </argument>
    <argument>
      <name>NewTotalBytesReceived</name>
      <direction>out</direction>
      <relatedStateVariable>TotalBytesReceived</relatedStateVariable>
    </argument>
  </argumentList>
</action>
<name>GetPacketStatistics</name>
<argumentList>
<argument>
<name>NewTotalPacketsSent</name>
<direction>out</direction>
<relatedStateVariable>TotalPacketsSent</relatedStateVariable>
</argument>
<argument>
<name>NewTotalPacketsReceived</name>
<direction>out</direction>
<relatedStateVariable>TotalPacketsReceived</relatedStateVariable>
</argument>
</argumentList>
</action>

<name>GetByteStatsForAssociatedDev</name>
<argumentList>
<argument>
<name>NewAssociatedDeviceMACAddress</name>
<direction>out</direction>
<relatedStateVariable>AssociatedDeviceMACAddress</relatedStateVariable>
</argument>
<argument>
<name>NewTotalBytesSent</name>
<direction>out</direction>
<relatedStateVariable>TotalBytesSent</relatedStateVariable>
</argument>
<argument>
<name>NewTotalBytesReceived</name>
<direction>out</direction>
<relatedStateVariable>TotalBytesReceived</relatedStateVariable>
</argument>
</argumentList>
</action>

<name>GetPacketStatsForAssociatedDev</name>
<argumentList>
<argument>
<name>NewAssociatedDeviceMACAddress</name>
<direction>out</direction>
<relatedStateVariable>AssociatedDeviceMACAddress</relatedStateVariable>
</argument>
<argument>
<name>NewTotalPacketsSent</name>
<direction>out</direction>
<relatedStateVariable>TotalPacketsSent</relatedStateVariable>
</argument>
<argument>
<name>NewTotalPacketsReceived</name>
<direction>out</direction>
<relatedStateVariable>TotalPacketsReceived</relatedStateVariable>
</argument>
</argumentList>
</action>

<name>GetTotalAssociations</name>
<argumentList>
<argument>
<name>NewTotalAssociations</name>
<direction>out</direction>
<relatedStateVariable>TotalAssociations</relatedStateVariable>
</argument>
</argumentList>
</action>

<name>GetGenericAssociatedDeviceInfo</name>
<argumentList>
<argument>
<name>NewAssociatedDeviceIndex</name>
<direction>in</direction>
<relatedStateVariable>TotalAssociations</relatedStateVariable>
</argument>
<argument>
<name>NewAssociatedDeviceMACAddress</name>
</argument>
</argumentList>
</action>
<direction>out</direction>
<relatedStateVariable>AssociatedDeviceMACAddress</relatedStateVariable>
</argument>
<argument>
<name>NewAssociatedDeviceIPAddress</name>
<direction>out</direction>
<relatedStateVariable>AssociatedDeviceIPAddress</relatedStateVariable>
</argument>
<argument>
<name>NewAssociatedDeviceAuthState</name>
<direction>out</direction>
<relatedStateVariable>AssociatedDeviceAuthState</relatedStateVariable>
</argument>
</argumentList>
</action>

<action>
<name>GetSpecificAssociatedDeviceInfo</name>
<argumentList>
<argument>
<name>NewAssociatedDeviceMACAddress</name>
<direction>in</direction>
<relatedStateVariable>AssociatedDeviceMACAddress</relatedStateVariable>
</argument>
<argument>
<name>NewAssociatedDeviceIPAddress</name>
<direction>out</direction>
<relatedStateVariable>AssociatedDeviceIPAddress</relatedStateVariable>
</argument>
<argument>
<name>NewAssociatedDeviceAuthState</name>
<direction>out</direction>
<relatedStateVariable>AssociatedDeviceAuthState</relatedStateVariable>
</argument>
</argumentList>
</action>

<action>
<name>GetSpecificIEEE11iInfo</name>
<argumentList>
<argument>
<name>NewAssociatedDeviceMACAddress</name>
<direction>in</direction>
<relatedStateVariable>AssociatedDeviceMACAddress</relatedStateVariable>
</argument>
<argument>
<name>NewLastRequestedUnicastCipher</name>
<direction>out</direction>
<relatedStateVariable>LastRequestedUnicastCipher</relatedStateVariable>
</argument>
<argument>
<name>NewLastRequestedMulticastCipher</name>
<direction>out</direction>
<relatedStateVariable>NewLastRequestedMulticastCipher</relatedStateVariable>
</argument>
<argument>
<name>NewIEEE11iAuthenticationMode</name>
<direction>out</direction>
<relatedStateVariable>IEEE11iAuthenticationMode</relatedStateVariable>
</argument>
<argument>
<name>NewLastPMKId</name>
<direction>out</direction>
<relatedStateVariable>LastPMKId</relatedStateVariable>
</argument>
</argumentList>
</action>

<action>
<name>SetAuthenticationServiceMode</name>
<argumentList>
<argument>
<name>NewAuthenticationServiceMode</name>
<direction>in</direction>
<relatedStateVariable>AuthenticationServiceMode</relatedStateVariable>
</argument>
</argumentList>
</action>
<action>
  <name>GetAuthenticationServiceMode</name>
  <argumentList>
  </argumentList>
</action>
<action>
  <name>SetSecurityKeys</name>
  <argumentList>
    <argument>
      <name>NewWEPKey0</name>
      <direction>in</direction>
      <relatedStateVariable>WEPKey</relatedStateVariable>
    </argument>
    <argument>
      <name>NewWEPKey1</name>
      <direction>in</direction>
      <relatedStateVariable>WEPKey</relatedStateVariable>
    </argument>
    <argument>
      <name>NewWEPKey2</name>
      <direction>in</direction>
      <relatedStateVariable>WEPKey</relatedStateVariable>
    </argument>
    <argument>
      <name>NewWEPKey3</name>
      <direction>in</direction>
      <relatedStateVariable>WEPKey</relatedStateVariable>
    </argument>
    <argument>
      <name>NewPreSharedKey</name>
      <direction>in</direction>
      <relatedStateVariable>PreSharedKey</relatedStateVariable>
    </argument>
    <argument>
      <name>NewKeyPassphrase</name>
      <direction>in</direction>
      <relatedStateVariable>KeyPassphrase</relatedStateVariable>
    </argument>
  </argumentList>
</action>
<action>
  <name>GetSecurityKeys</name>
  <argumentList>
    <argument>
      <name>NewWEPKey0</name>
      <direction>out</direction>
      <relatedStateVariable>WEPKey</relatedStateVariable>
    </argument>
    <argument>
      <name>NewWEPKey1</name>
      <direction>out</direction>
      <relatedStateVariable>WEPKey</relatedStateVariable>
    </argument>
    <argument>
      <name>NewWEPKey2</name>
      <direction>out</direction>
      <relatedStateVariable>WEPKey</relatedStateVariable>
    </argument>
    <argument>
      <name>NewWEPKey3</name>
      <direction>out</direction>
      <relatedStateVariable>WEPKey</relatedStateVariable>
    </argument>
    <argument>
      <name>NewPreSharedKey</name>
      <direction>out</direction>
      <relatedStateVariable>PreSharedKey</relatedStateVariable>
    </argument>
  </argumentList>
</action>
<action>
  <name>GetAuthenticationServiceMode</name>
  <argumentList>
    <argument>
      <name>NewAuthenticationServiceMode</name>
      <direction>out</direction>
      <relatedStateVariable>AuthenticationServiceMode</relatedStateVariable>
    </argument>
  </argumentList>
</action>
<argument>
  <name>NewKeyPassphrase</name>
  <direction>out</direction>
  <relatedStateVariable>KeyPassphrase</relatedStateVariable>
</argument>
</argumentList>
</action>

<action>
  <name>SetDefaultWEPKeyIndex</name>
  <argumentList>
    <argument>
      <name>NewDefaultWEPKeyIndex</name>
      <direction>in</direction>
      <relatedStateVariable>WEPKeyIndex</relatedStateVariable>
    </argument>
  </argumentList>
</action>

<action>
  <name>GetDefaultWEPKeyIndex</name>
  <argumentList>
    <argument>
      <name>NewDefaultWEPKeyIndex</name>
      <direction>out</direction>
      <relatedStateVariable>WEPKeyIndex</relatedStateVariable>
    </argument>
  </argumentList>
</action>

<action>
  <name>SetPreSharedKey</name>
  <argumentList>
    <argument>
      <name>NewPreSharedKeyIndex</name>
      <direction>in</direction>
      <relatedStateVariable>PreSharedKeyIndex</relatedStateVariable>
    </argument>
    <argument>
      <name>NewAssociatedDeviceMACAddress</name>
      <direction>in</direction>
      <relatedStateVariable>AssociatedDeviceMACAddress</relatedStateVariable>
    </argument>
    <argument>
      <name>NewPreSharedKey</name>
      <direction>in</direction>
      <relatedStateVariable>PreSharedKey</relatedStateVariable>
    </argument>
    <argument>
      <name>NewPSKPassphrase</name>
      <direction>in</direction>
      <relatedStateVariable>KeyPassphrase</relatedStateVariable>
    </argument>
  </argumentList>
</action>

<action>
  <name>GetPreSharedKey</name>
  <argumentList>
    <argument>
      <name>NewPreSharedKeyIndex</name>
      <direction>in</direction>
      <relatedStateVariable>PreSharedKeyIndex</relatedStateVariable>
    </argument>
    <argument>
      <name>NewPreSharedKeyIndex</name>
      <direction>out</direction>
      <relatedStateVariable>PreSharedKeyIndex</relatedStateVariable>
    </argument>
    <argument>
      <name>NewPreSharedKey</name>
      <direction>out</direction>
      <relatedStateVariable>PreSharedKey</relatedStateVariable>
    </argument>
    <argument>
      <name>NewPSKPassphrase</name>
      <direction>out</direction>
      <relatedStateVariable>PreSharedKey</relatedStateVariable>
    </argument>
  </argumentList>
</action>
<action name="SetBasBeaconSecurityProperties">
    <argumentList>
        <argument>
            <name>NewBasicEncryptionModes</name>
            <direction>in</direction>
            <relatedStateVariable>BasicEncryptionModes</relatedStateVariable>
        </argument>
        <argument>
            <name>NewBasicAuthenticationMode</name>
            <direction>in</direction>
            <relatedStateVariable>BasicAuthenticationMode</relatedStateVariable>
        </argument>
    </argumentList>
</action>

<action name="GetBasBeaconSecurityProperties">
    <argumentList>
        <argument>
            <name>NewBasicEncryptionModes</name>
            <direction>out</direction>
            <relatedStateVariable>BasicEncryptionModes</relatedStateVariable>
        </argument>
        <argument>
            <name>NewBasicAuthenticationMode</name>
            <direction>out</direction>
            <relatedStateVariable>BasicAuthenticationMode</relatedStateVariable>
        </argument>
    </argumentList>
</action>

<action name="SetWPABeaconSecurityProperties">
    <argumentList>
        <argument>
            <name>NewWPAEncryptionModes</name>
            <direction>in</direction>
            <relatedStateVariable>WPAEncryptionModes</relatedStateVariable>
        </argument>
        <argument>
            <name>NewWPAAuthenticationMode</name>
            <direction>in</direction>
            <relatedStateVariable>WPAAuthenticationMode</relatedStateVariable>
        </argument>
    </argumentList>
</action>

<action name="GetWPABeaconSecurityProperties">
    <argumentList>
        <argument>
            <name>NewWPAEncryptionModes</name>
            <direction>out</direction>
            <relatedStateVariable>WPAEncryptionModes</relatedStateVariable>
        </argument>
        <argument>
            <name>NewWPAAuthenticationMode</name>
            <direction>out</direction>
            <relatedStateVariable>WPAAuthenticationMode</relatedStateVariable>
        </argument>
    </argumentList>
</action>

<action name="SetIEEE11iBeaconSecurityProperties">
    <argumentList>
        <argument>
            <name>NewIEEE11iEncryptionModes</name>
            <direction>in</direction>
            <relatedStateVariable>IEEE11iEncryptionModes</relatedStateVariable>
        </argument>
        <argument>
            <name>NewIEEE11iAuthenticationMode</name>
            <direction>in</direction>
        </argument>
    </argumentList>
</action>

<action name="GetIEEE11iBeaconSecurityProperties">
    <argumentList>
        <argument>
            <name>NewIEEE11iEncryptionModes</name>
            <direction>out</direction>
            <relatedStateVariable>IEEE11iEncryptionModes</relatedStateVariable>
        </argument>
        <argument>
            <name>NewIEEE11iAuthenticationMode</name>
            <direction>out</direction>
        </argument>
    </argumentList>
</action>
<relatedStateVariable>IEEE11iAuthenticationMode</relatedStateVariable>
</argumentList>
</argument>
</action>
<action>
<name>Get11iBeaconSecurityProperties</name>
<argumentList>
<argument>
<name>NewIEEE11iEncryptionModes</name>
<direction>out</direction>
<relatedStateVariable>IEEE11iEncryptionModes</relatedStateVariable>
</argument>
<argument>
<name>NewIEEE11iAuthenticationMode</name>
<direction>out</direction>
<relatedStateVariable>IEEE11iAuthenticationMode</relatedStateVariable>
</argument>
</argumentList>
</action>
<action>
<name>FactoryDefaultReset</name>
</action>
<action>
<name>ResetAuthentication</name>
</action>
</actionList>
</serviceStateTable>
<stateVariable sendEvents="no">
<name>InsecureOOBAccessEnabled</name>
<dataType>boolean</dataType>
</stateVariable>
<stateVariable sendEvents="no">
<name>SSID</name>
<dataType>string</dataType>
</stateVariable>
<stateVariable sendEvents="no">
<name>BSSID</name>
<dataType>string</dataType>
</stateVariable>
<stateVariable sendEvents="no">
<name>BeaconType</name>
<dataType>string</dataType>
<allowedValueList>
<allowedValue>None</allowedValue>
<allowedValue>Basic</allowedValue>
<allowedValue>WPA</allowedValue>
<allowedValue>11i</allowedValue>
<allowedValue>BasicandWPA</allowedValue>
<allowedValue>Basicand11i</allowedValue>
<allowedValue>WPAand11i</allowedValue>
<allowedValue>BasicandWPAand11i</allowedValue>
</allowedValueList>
</stateVariable>
<stateVariable sendEvents="no">
<name>BeaconAdvertisementEnabled</name>
<dataType>boolean</dataType>
</stateVariable>
<stateVariable sendEvents="no">
<name>RadioEnabled</name>
<dataType>boolean</dataType>
</stateVariable>
<stateVariable sendEvents="no">
<name>LocationDescription</name>
<dataType>string</dataType>
</stateVariable>
<stateVariable sendEvents="no">
<name>RegulatoryDomain</name>
<dataType>string</dataType>
</stateVariable>
<stateVariable sendEvents="no">
<name>TotalPSKFailures</name>
<dataType>ui4</dataType>
</stateVariable>
<stateVariable sendEvents="no">
</stateVariable>
<name>TotalIntegrityFailures</name>
<dataType>ui4</dataType>
</stateVariable>

<stateVariable sendEvents="no">
  <name>Channel</name>
  <dataType>ui1</dataType>
</stateVariable>

<stateVariable sendEvents="no">
  <name>PossibleChannels</name>
  <dataType>string</dataType>
</stateVariable>

<stateVariable sendEvents="no">
  <name>ChannelsInUse</name>
  <dataType>string</dataType>
</stateVariable>

<stateVariable sendEvents="no">
  <name>DeviceOperationMode</name>
  <dataType>string</dataType>
  <allowedValueList>
    <allowedValue>InfrastructureAccessPoint</allowedValue>
    <allowedValue>WirelessBridgePointToPoint</allowedValue>
    <allowedValue>WirelessBridgePointToMultipoint</allowedValue>
    <allowedValue>WirelessRepeater</allowedValue>
    <allowedValue>WirelessSTA</allowedValue>
  </allowedValueList>
</stateVariable>

<stateVariable sendEvents="no">
  <name>DistanceFromRoot</name>
  <dataType>ui1</dataType>
</stateVariable>

<stateVariable sendEvents="no">
  <name>PeerBSSID</name>
  <dataType>string</dataType>
</stateVariable>

<stateVariable sendEvents="no">
  <name>BasicDataTransmitRates</name>
  <dataType>string</dataType>
</stateVariable>

<stateVariable sendEvents="no">
  <name>OperationalDataTransmitRates</name>
  <dataType>string</dataType>
</stateVariable>

<stateVariable sendEvents="no">
  <name>PossibleDataTransmitRates</name>
  <dataType>string</dataType>
</stateVariable>

<stateVariable sendEvents="no">
  <name>AutoRateFallBackEnabled</name>
  <dataType>boolean</dataType>
</stateVariable>

<stateVariable sendEvents="no">
  <name>TotalBytesSent</name>
  <dataType>ui4</dataType>
</stateVariable>

<stateVariable sendEvents="no">
  <name>TotalBytesReceived</name>
  <dataType>ui4</dataType>
</stateVariable>

<stateVariable sendEvents="no">
  <name>TotalPacketsSent</name>
  <dataType>ui4</dataType>
</stateVariable>

<stateVariable sendEvents="no">
  <name>TotalPacketsReceived</name>
  <dataType>ui4</dataType>
</stateVariable>

<stateVariable sendEvents="yes">
  <name>TotalAssociations</name>
  <dataType>ui2</dataType>
</stateVariable>

<stateVariable sendEvents="no">
  <name>AssociatedDeviceMACAddress</name>
  <dataType>string</dataType>
</stateVariable>

<stateVariable sendEvents="no">
  <name>AssociatedDeviceIPAddress</name>
  <dataType>string</dataType>
</stateVariable>

<stateVariable sendEvents="no">
  <name>AssociatedDeviceAuthState</name>
  <dataType>boolean</dataType>
</stateVariable>

<stateVariable sendEvents="no">
  <name>AuthenticationServiceMode</name>
  <dataType>string</dataType>
  <allowedValueList>
    <allowedValue>None</allowedValue>
    <allowedValue>LinkAuthentication</allowedValue>
    <allowedValue>RadiusClient</allowedValue>
  </allowedValueList>
</stateVariable>

<stateVariable sendEvents="no">
  <name>WEPKeyIndex</name>
  <dataType>ui1</dataType>
</stateVariable>

<stateVariable sendEvents="no">
  <name>WEPKey</name>
  <dataType>string</dataType>
</stateVariable>

<stateVariable sendEvents="no">
  <name>KeyPassphrase</name>
  <dataType>string</dataType>
</stateVariable>

<stateVariable sendEvents="no">
  <name>WEPEncryptionLevel</name>
  <dataType>string</dataType>
  <allowedValueList>
    <allowedValue>Disabled</allowedValue>
    <allowedValue>40-bit</allowedValue>
    <allowedValue>104-bit</allowedValue>
  </allowedValueList>
</stateVariable>

<stateVariable sendEvents="no">
  <name>PreSharedKey</name>
  <dataType>string</dataType>
</stateVariable>

<stateVariable sendEvents="no">
  <name>PreSharedKeyIndex</name>
  <dataType>ui1</dataType>
</stateVariable>

<stateVariable sendEvents="no">
  <name>BasicEncryptionModes</name>
  <dataType>string</dataType>
  <allowedValueList>
    <allowedValue>None</allowedValue>
    <allowedValue>WEPEncryption</allowedValue>
  </allowedValueList>
</stateVariable>

<stateVariable sendEvents="no">
  <name>BasicAuthenticationMode</name>
  <dataType>string</dataType>
  <allowedValueList>
    <allowedValue>None</allowedValue>
    <allowedValue>EAPAuthentication</allowedValue>
  </allowedValueList>
</stateVariable>

<stateVariable sendEvents="no">
  <name>WPAEncryptionModes</name>
  <dataType>string</dataType>
  <allowedValueList>
    <allowedValue>WEPEncryption</allowedValue>
    <allowedValue>TKIPIEncryption</allowedValue>
    <allowedValue>WEPandTKIPIEncryption</allowedValue>
    <allowedValue>AESEncryption</allowedValue>
    <allowedValue>WEPandAESEncryption</allowedValue>
    <allowedValue>TKIPandAESEncryption</allowedValue>
    <allowedValue>WEPandTKIPandAESEncryption</allowedValue>
  </allowedValueList>
</stateVariable>
</serviceStateTable>
</scp>
4. Test

*No semantic tests have been defined for this service.*