Campus Wireless Networks (6.x)

Security

Authors: Meggie Yao
Contributors: Syed Ahmed
Copyright Information

© Copyright 2017 Hewlett Packard Enterprise Development LP.

Open Source Code

This product includes code licensed under the GNU General Public License, the GNU Lesser General Public License, and/or certain other open source licenses. A complete machine-readable copy of the source code corresponding to such code is available upon request. This offer is valid to anyone in receipt of this information and shall expire three years following the date of the final distribution of this product version by Hewlett Packard Enterprise Company. To obtain such source code, send a check or money order in the amount of US $10.00 to:

Hewlett Packard Enterprise Company
Attn: General Counsel
3000 Hanover Street
Palo Alto, CA 94304
USA
## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>About this Guide</td>
<td>7</td>
</tr>
<tr>
<td>Scope</td>
<td>7</td>
</tr>
<tr>
<td>Typical Campus Deployment with Redundancy</td>
<td>7</td>
</tr>
<tr>
<td>Aruba Campus Logical Architecture</td>
<td>8</td>
</tr>
<tr>
<td>Policy Enforcement Firewall</td>
<td>10</td>
</tr>
<tr>
<td>Policy Enforcement Firewall</td>
<td>10</td>
</tr>
<tr>
<td>User Roles</td>
<td>10</td>
</tr>
<tr>
<td>When does a Role Get Applied?</td>
<td>11</td>
</tr>
<tr>
<td>How is a User Role Derived from Derivation Rules?</td>
<td>12</td>
</tr>
<tr>
<td>User and Server Role Derivation</td>
<td>13</td>
</tr>
<tr>
<td>Working with User-Derived VLANs and Roles</td>
<td>14</td>
</tr>
<tr>
<td>Working with Server-Derived Role</td>
<td>15</td>
</tr>
<tr>
<td>Server-Derivation Rules</td>
<td>15</td>
</tr>
<tr>
<td>ArubaVSA</td>
<td>16</td>
</tr>
<tr>
<td>Summary – When do Derived Rules Get Applied</td>
<td>20</td>
</tr>
<tr>
<td>Example Use for User Derived Rules</td>
<td>21</td>
</tr>
<tr>
<td>Example Use for Server Derived Rules</td>
<td>21</td>
</tr>
<tr>
<td>Example Use for Mixed Authentications</td>
<td>22</td>
</tr>
<tr>
<td>Role Derivation in Mixed Authentication Environment</td>
<td>23</td>
</tr>
<tr>
<td>General Role Derivation Flow</td>
<td>23</td>
</tr>
<tr>
<td>Layer 2 Role Priority</td>
<td>24</td>
</tr>
<tr>
<td>Layer 3 Roles</td>
<td>25</td>
</tr>
<tr>
<td>Expected User Role in Mixed Authentication Environment</td>
<td>26</td>
</tr>
<tr>
<td>Role Derivation Flowchart in Various Authentications</td>
<td>28</td>
</tr>
<tr>
<td>Valid-user ACL</td>
<td>29</td>
</tr>
<tr>
<td>What is a Valid-user ACL?</td>
<td>30</td>
</tr>
<tr>
<td>How it Works?</td>
<td>30</td>
</tr>
<tr>
<td>Valid-user ACL Configuration</td>
<td>31</td>
</tr>
<tr>
<td>Firewall Local-Valid-Users</td>
<td>31</td>
</tr>
<tr>
<td>Best Practices</td>
<td>32</td>
</tr>
</tbody>
</table>
Block critical network resources .................................................. 32
Firewall ACE Entries .................................................................... 33
  How many ACE entries are there? ................................................. 33
  How are ACE entries used? ......................................................... 33
  Viewing ACE Entries ................................................................. 35
  What to do when the ACE Table is full? ...................................... 35

**Authentication** ........................................................................ 37
Captive Portal Authentication ....................................................... 38
  Packet Captures on the Client ..................................................... 41
  CPPM Server Initiated Web Login .............................................. 44
  CPPM Controller Initiated Web Login ......................................... 46
  Captive Portal Configuration Flow .............................................. 47
  CP Whitelist or Walled Garden .................................................. 48
  Apple CNA Bypass .................................................................... 49
  Bypassing Captive Portal Landing Page ..................................... 49
  Tri-Session DNAT for CP User VLAN Without IP ...................... 50
  Captive Portal Configuration Recommendations ..................... 51

Captive Portal with MAC Caching via CPPM ............................... 52
  MAC Caching Authentication Flow ............................................ 52
  Guest MAC Added to Endpoints with Unknown State ............... 53
  Guest Finishes Registration with Username and Password .......... 54
  Guest Passes Captive Portal Authentication ............................. 55
  Guest MAC State Changed to Known in Endpoints .................... 56
  Captive Portal Authentication Not Required Again .................... 56
  MAC Caching Expired .............................................................. 56

MAC Authentication ..................................................................... 57
  Use Cases .................................................................................. 57
  Delimiters .................................................................................. 57
  MAC Authentication Configuration Flow .................................... 58

Guest Access with Self Registration .......................................... 59
  Self-Registration Workflow ....................................................... 60

**802.1X Authentication** .......................................................... 62
  Key Hierarchy ............................................................................ 63
  General Packets Flow of 802.1X Authentication ....................... 64
  802.11 Association .................................................................... 64
<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ports Need to be Opened</td>
<td>81</td>
</tr>
<tr>
<td>Operating Systems Supported by Latest VIA</td>
<td>82</td>
</tr>
<tr>
<td>VPN Server – The Controller Configuration for VIA</td>
<td>82</td>
</tr>
<tr>
<td>VPN Server Configuration</td>
<td>83</td>
</tr>
<tr>
<td>Configuring the VPN Server for IKEv1</td>
<td>84</td>
</tr>
<tr>
<td>Configuring VPN Server for IKEv1-PSK</td>
<td>85</td>
</tr>
<tr>
<td>L2TP and XAUTH Parameters</td>
<td>85</td>
</tr>
<tr>
<td>Address Pools</td>
<td>85</td>
</tr>
<tr>
<td>IKE Aggressive Group Name</td>
<td>86</td>
</tr>
<tr>
<td>IKE Shared Secret</td>
<td>86</td>
</tr>
<tr>
<td>Configuring VPN Server for IKEv1 Certificates</td>
<td>86</td>
</tr>
<tr>
<td>IKE Server Certificate</td>
<td>86</td>
</tr>
<tr>
<td>CA Certificate Assigned for VPN-Clients</td>
<td>87</td>
</tr>
<tr>
<td>Certificate Groups for VPN-Clients</td>
<td>87</td>
</tr>
<tr>
<td>IKEv1 Phase 2 Authentication</td>
<td>87</td>
</tr>
<tr>
<td>Configuring VPN Server for IKEv2</td>
<td>88</td>
</tr>
<tr>
<td>L2TP and XAUTH Parameters</td>
<td>89</td>
</tr>
<tr>
<td>IKE Server Certificate</td>
<td>89</td>
</tr>
<tr>
<td>CA Certificate Assigned for VPN-Clients</td>
<td>89</td>
</tr>
<tr>
<td>Check Certificate Common Name Against AAA Server</td>
<td>89</td>
</tr>
<tr>
<td>IKEv2 EAP Authentication</td>
<td>89</td>
</tr>
<tr>
<td>IKE Policies and IPsec Maps</td>
<td>90</td>
</tr>
<tr>
<td><strong>Figures</strong></td>
<td>91</td>
</tr>
<tr>
<td><strong>Symbols</strong></td>
<td>93</td>
</tr>
<tr>
<td><strong>Acronyms</strong></td>
<td>96</td>
</tr>
</tbody>
</table>
This chapter includes the following sections:

- **Scope on page 7**
- **Typical Campus Deployment with Redundancy on page 7**
- **Aruba Campus Logical Architecture on page 8**

**Scope**

The validated reference design (VRD) series focuses on particular aspects of Aruba technologies and deployment models.

Together the guides provide a structured framework to understand and deploy Aruba Wireless Local Area Networks (WLANs). The VRD series has four types of guides:

- Foundation: These guides explain the core technologies of an Aruba WLAN. The guides also describe different aspects of planning, operation, and troubleshooting deployments.
- Base Designs: These guides describe the most common deployment models, recommendations, and configurations.
- Application: These guides build on the base designs. These guides deliver specific information that is relevant to deploying particular applications such as voice, video, or outdoor campus extension.
- Specialty Deployments: These guides involve deployments in conditions that differ significantly from the common base design deployment models, such as high-density WLAN deployments.

**Figure 1 Aruba Reference Architectures**

This VRD covers the deployment of Aruba WLAN in a typical campus network, and it is considered part of the Base Designs within the VRD core technologies series. This guide covers the design recommendations for a campus deployment and it explains the various configurations needed to implement the Aruba secure, high-performance, multimedia grade WLAN solution in large campuses.

**Typical Campus Deployment with Redundancy**

An Aruba corporate campus encompasses two master controllers (MC) and all of the network services in the data center. The campus also includes distribution to corporate clients on the campus with a pair of local
controllers (LC). A fire walled demilitarized zone (DMZ) holds an additional set of master controllers (DMZ-MC) and services meant to deliver Wi-Fi to a separate domain of guest clients. As a whole, the topology makes use of guest traffic isolation, various redundancies, and master-local configuration.

**Figure 2 Campus Deployment with Redundancy**

![Campus Deployment with Redundancy](image)

**Aruba Campus Logical Architecture**

Aruba WLAN has a logical four-tier operating model that consists of these four layers:

**Management** - The management layer consists of AirWave®. AirWave provides a single point of management for the WLAN, including reporting, heat maps, centralized configuration, and troubleshooting.

**Network services** - The network services layer consists of master mobility controllers and Amigopod. Amigopod provides secure and flexible visitor management services. The master controllers provide a control plane for the Aruba WLAN that spans the physical geography of the wired network. The control plane does not directly deal with user traffic or access points (APs). Instead the control plane provides services such as whitelist coordination, valid AP lists, Control Plane Security (CPSec) certificates, Radio Frequency Protect (RFProtect™) coordination, and Remote Authentication Dial-In User Service (RADIUS) or authentication, authorization, and accounting (AAA) proxy.

**Aggregation** - The aggregation layer is the interconnect point where the AP, air monitor (AM), and spectrum monitor (SM) traffic aggregates. This layer provides a logical point for enforcement of roles and policies on centralized traffic that enters or exits the enterprise LAN.

**Network access** - The network access layer is comprised of APs, AMs, and SMs that work together with the aggregation layer controllers to overlay the Aruba WLAN.
Figure 3  Aruba Campus Logical Architecture
This chapter includes the following sections:

- Policy Enforcement Firewall on page 10
- User Roles on page 10
- When does a Role Get Applied? on page 11
- How is a User Role Derived from Derivation Rules? on page 12
- User and Server Role Derivation on page 13
- ArubaVSAs on page 16
- Summary – When do Derived Rules Get Applied on page 20
- Valid-user ACL on page 29
- Firewall ACE Entries on page 33

**Policy Enforcement Firewall**

Aruba integrates a stateful firewall – Policy Enforcement Firewall (PEF) into its wireless offering. Using PEF, organizations can enforce network access policies that specify who may access the network, which areas of the network they may access, and how much bandwidth they can have. Administrators can build a unified, integrated system for network policy enforcement by leveraging PEF’s open interfaces to external services such as content security appliances, policy engines like ClearPass, performance monitors, and authentication/authorization servers.

A firewall policy is a set of access list rules that applies to traffic that passes through the Aruba controller. A firewall policy identifies specific characteristics about a data packet passing through the Aruba controller and takes some action based on that identification. In an Aruba controller, that action can be a firewall-type action such as permitting or denying the packet, an administrative action such as logging the packet, or a quality of service (QoS) action such as setting 802.1p bits or placing the packet into a priority queue. You can apply firewall policies to user roles to give differential treatment to different users on the same network, or to physical ports to apply the same policy to all traffic through the port.

The rules and policies are processed in a top-down fashion, so the position of a rule within a policy and the position of a policy within a user-role determines the functionality of the user role. When you construct a role, you must put the rules and policies in the proper order.

**User Roles**

Every client in an Aruba user-centric network is associated with a user role, which determines the client’s network privileges, how often it must re-authenticate, and which bandwidth contracts are applicable. A policy is a set of rules that applies to traffic that passes through the Aruba controller. You specify one or more policies for a user role. Finally, you can assign a user role to clients before or after they authenticate to the system.

In other words, a user-role is a set of rights assigned to a user. The attributes that can be defined within those rights fall various categories as explained in Figure 4.
When does a Role Get Applied?

As noted in the previous section, the user-role can be thought of as a set of policies, that define access rules, VLAN the client needs to be assigned, bandwidth limitations to be applied on the client, and so on.

Figure 5 gives a snapshot of when a ‘role’ gets applied to a client (wireless or wired).

Figure 5 Applying a Role
When a wired client connects to a wired port that is untrusted* on an Aruba controller, the controller looks up what is known as a user-table.

If the user-table does not contain the MAC address of the wired client that just connected, a user-role is applied to the client. Unless defined otherwise, all such wired users are placed in a default logon role.

If an ‘aaa profile’ is attached to such untrusted wired port, the user-role defined in the aaa profile is assigned to the wired client. The AAA profile attached to a WLAN defines a role a client should be in before it authenticates, and a role it should get assigned once it authenticates successfully.

Analogous to that, when a wireless client associates to the AP, it has to successfully authenticate first, in order to get assigned the user-role, which defines a set of access rules for that client.

For a wireless client, the AAA profile attached to the Virtual AP# defines what user-role the wireless client should be placed into, either in case of successful authentication or failed authentication.

*Untrusted port is defined as a port that does not let any traffic through unless the wired client is subjected to an authentication process, following which it gets added to the desired user-role and allows traffic post authentication.

A logon role is a default role that is assigned to a wired or wireless client that has either failed to authenticate successfully, or that happens to be the default role assigned in the aaa profile for that wired port, or in the wireless case, the SSID assigned to a virtual ap profile.

#Virtual-AP - The Wireless LAN collection of profiles configure WLANs in the form of virtual AP profiles. A virtual AP profile contains an SSID profile, which defines the WLAN, the high-throughput SSID profile, and an AAA profile that defines the authentication for the WLAN.

---

How is a User Role Derived from Derivation Rules?

**Figure 6  User Role Derived**

- **How a User Role is derived?**
  - **Via User Derivation Rules**
    - Attributes of user or device used to make a role/VLAN decision
  - **Via Server Derivation Rules**
    - Attributes returned by auth server during authentication and used to make a role/VLAN decision

Defined with
- **User Attributes**
- **Server Attributes**

---

**Figure 6** above shows a snapshot of how a User Role is derived, using one of two main types of rules:

- User Derived Rules
- Server Derived Rules
There is a certain order in which these rules get applied and one may precede the other. The below (in the same order) is how a user-role is assigned.

A client is assigned a user role by one of several methods. A role assigned by one method may take precedence over one assigned by a different method.

The methods of assigning user roles are, from lowest to highest precedence:

1. The initial user role or VLAN for unauthenticated clients is configured in the AAA profile for a virtual AP.
2. The user role can be derived from user attributes upon the client’s association with an AP (this is known as a user-derived role). You can configure rules that assign a user role to clients that match a certain set of criteria. For example, you can configure a rule to assign the role VoIP-Phone to any client that has a MAC address that starts with bytes xx:yy:zz. User-derivation rules are executed before client authentication.
3. The user role can be the default user role configured for an authentication method, such as 802.1X or VPN. For each authentication method, you can configure a default role for clients who are successfully authenticated using that method.
4. The user role can be derived from attributes returned by the authentication server and certain client attributes (this is known as a server-derived role). If the client is authenticated via an authentication server, the user role for the client can be based on one or more attributes returned by the server during authentication, or on client attributes such as SSID (even if the attribute is not returned by the server). Server-derivation rules are executed after client authentication.
5. The user role can be derived from Aruba Vendor-Specific Attributes (VSA) for RADIUS server authentication. A role derived from an Aruba VSA takes precedence over any other user roles.

**User and Server Role Derivation**

The following flowchart illustrates user and server role derivation.

**Figure 7  User and Server Role Derivation**

This section includes the following topics:
• Working with User-Derived VLANs and Roles on page 14
• Working with Server-Derived Role on page 15
• Server-Derivation Rules on page 15

Working with User-Derived VLANs and Roles

Attributes derived from the client’s association with an AP can be used to assign the client to a specific role or VLAN, as user-derivation rules are executed before the client is authenticated.

You configure the user role or VLAN to be assigned to the client by specifying condition rules; when a condition is met, the specified user role or VLAN is assigned to the client. You can specify more than one condition rule; the order of rules is important as the first matching condition is applied. You can optionally add a description of the user rule.

<table>
<thead>
<tr>
<th>Rule Type</th>
<th>Condition</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>BSSID: Assign client to a role or VLAN based upon the BSSID of AP to which client is associating.</td>
<td>One of the following: contains, ends with, equals, does not equal, starts with</td>
<td>MAC address (xx:xx:xx:xx:xx:xx)</td>
</tr>
<tr>
<td>DHCP-Option: Assign client to a role or VLAN based upon the DHCP signature ID.</td>
<td>One of the following: equals, starts with</td>
<td>DHCP signature ID. NOTE: This string is not case sensitive.</td>
</tr>
<tr>
<td>DHCP-Option-77: Assign client to a role or VLAN based upon the user class identifier returned by DHCP server.</td>
<td>equals</td>
<td>string</td>
</tr>
<tr>
<td>Encryption: Assign client to a role or VLAN based upon the encryption type used by the client.</td>
<td>One of the following: equals, does not equal</td>
<td>Open (no encryption) WPA/WPA2 AES WPA-TKIP (static or dynamic) Dynamic WEP WPA/WPA2 AES PSK Static WEP xSec</td>
</tr>
<tr>
<td>Rule Type</td>
<td>Condition</td>
<td>Value</td>
</tr>
<tr>
<td>--------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>ESSID: Assign client to a role or VLAN based upon the ESSID to which the client is associated</td>
<td>One of the following: • contains • ends with • equals • does not equal • starts with • value of (does not take string; attribute value is used as role)</td>
<td>string</td>
</tr>
<tr>
<td>Location: Assign client to a role or VLAN based upon the ESSID to which the client is associated</td>
<td>One of the following: • equals • does not equal</td>
<td>string</td>
</tr>
<tr>
<td>MAC address of the client</td>
<td>One of the following: • contains • ends with • equals • does not equal • starts with</td>
<td>MAC address (xx:xx:xx:xx:xx:xx)</td>
</tr>
</tbody>
</table>

**Working with Server-Derived Role**

If the client is authenticated through an authentication server, the user role for the client can be based on one or more attributes returned by the server during authentication. You configure the user role to be derived by specifying condition rules; when a condition is met, the specified user role is assigned to the client. You can specify more than one condition rule; the order of rules is important as the first matching condition is applied. You can also define server rules based on client attributes such as ESSID, BSSID, or MAC address, even though these attributes are not returned by the server.

Many Network Address Server (NAS) vendors, including Aruba, use VSAs to provide features not supported in standard RADIUS attributes. For Aruba systems, VSAs can be employed to provide the user role and VLAN for RADIUS-authenticated clients, however the VSAs must be present on your RADIUS server. This involves defining the vendor (Aruba) and/or the vendor-specific code (14823), vendor-assigned attribute number, attribute format (such as string or integer), and attribute value in the RADIUS dictionary file. VSAs supported on controllers conform to the format recommended in RFC 2865, “Remote Authentication Dial In User Service (RADIUS)”.

Aruba supports Vendor Specific Attributes (VSA), custom-defined attributes and Aruba VSAs.

Dictionary files that contain Aruba VSAs are available on the Aruba support website for various RADIUS servers. Log into the Aruba support website to download a dictionary file from the Tools folder.

**Server-Derivation Rules**

When you configure a server group, you can set the VLAN or role for clients based on attributes returned for the client by the server during authentication. The server derivation rules apply to all servers in the group. The user role or VLAN assigned through server derivation rules takes precedence over the default role and VLAN configured for the authentication method.

The authentication servers must be configured to return the attributes for the clients during authentication.
The server rules are applied based on the first match principle. The first rule that is applicable for the server and the attribute returned is applied to the client and would be the only rule applied from the server rules. These rules are applied uniformly across all servers in the server group.

Table 2: Conditions for Server-Derived Rules

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Role or VLAN</td>
<td>The server derivation rules apply to either user role or VLAN assignment. With Role assignment, a client can be assigned a specific role based on the attributes returned. In VLAN assignment, the client can be placed in a specific VLAN based on the attributes returned.</td>
</tr>
<tr>
<td>Attribute</td>
<td>This is the attribute returned by the authentication server that is examined for Operation and Operand match.</td>
</tr>
</tbody>
</table>
| Operation  | This is the match method by which the string in Operand is matched with the attribute value returned by the authentication server.  
  - contains: The rule is applied if and only if the attribute value contains the string in parameter Operand.  
  - starts-with: The rule is applied if and only if the attribute value returned starts with the string in parameter Operand.  
  - ends-with: The rule is applied if and only if the attribute value returned ends with the string in parameter Operand.  
  - equals: The rule is applied if and only if the attribute value returned equals the string in parameter Operand.  
  - not-equals: The rule is applied if and only if the attribute value returned is not equal to the string in parameter Operand.  
  - value-of: This is a special condition. What this implies is that the role or VLAN is set to the value of the attribute returned. For this to be successful, the role and the VLAN ID returned as the value of the attribute selected must already be configured on the managed device when the rule is applied. |
| Operand    | This is the string to which the value of the returned attribute is matched.                                                                                                                               |
| Action     | Defines whether to assign a role or a VLAN to the user when the rule is matched.                                                                                                                     |
| Position   | Position of the condition rule. Rules are applied based on the first match principle. 1 is the top. Default: bottom                                                                                     |

ArubaVSAs

As mentioned earlier, Aruba controllers support various Aruba VSAs, as illustrated in Figure 8.
Figure 8  Aruba VSAs

Table 3 below contains a list of such VSAs and their function.

Aruba VSAs greatly simplify the user role derivation based on various VSAs.

Table 3: VSAs

<table>
<thead>
<tr>
<th>VSA Name</th>
<th>VSA Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aruba-User-Role String 1</td>
<td>This VSA returns the role, to be assigned to the user post authentication. The user will be granted access based on the role attributes defined in the role.</td>
</tr>
<tr>
<td>Aruba-User-VLAN Integer 2</td>
<td>This VSA is used to return the VLAN to be used by the user. This is an integer and ranges between 1 – 4094.</td>
</tr>
<tr>
<td>Aruba-Priv-Admin-User Integer 3</td>
<td>If this VSA is set in the RADIUS accept message, it is used to bypass the enable prompt.</td>
</tr>
<tr>
<td>Aruba-Admin-Role String 4</td>
<td>This VSA returns the management role to be assigned to the user post management authentication. This role can be seen using the command “show mgmt.-role”.</td>
</tr>
<tr>
<td>Aruba-Essid-Name String 5</td>
<td>This is “&quot;sent by ArubaOS””, to the RADIUS. This is the WLAN SSID name to which the client is connected.</td>
</tr>
<tr>
<td>Aruba-Location-Id String 6</td>
<td>Can be present in either Access Request or Access Accept packet and is used to define and identify Aruba AP name. This again is sent by the ArubaOS to the RADIUS server.</td>
</tr>
<tr>
<td>Aruba-Port-Id String 7</td>
<td></td>
</tr>
<tr>
<td>Aruba-Template-User String 8</td>
<td></td>
</tr>
<tr>
<td>Aruba-Named-User-Vlan String 9</td>
<td>This VSA returns a VLAN name for a user and this vlan-name on controller could be mapped to a user defined one or multiple vlan-id's.</td>
</tr>
<tr>
<td>Aruba-AP-Group String 10</td>
<td>Can be present in either access request to identify the AP Group to which client is connected, or in Access Accept to identify the AP Group to which AP should be associated.</td>
</tr>
</tbody>
</table>
### Table 3: VSAs

<table>
<thead>
<tr>
<th>VSA Name</th>
<th>VSA Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aruba-Framed-IPv6-Address String 11</td>
<td>This attribute is used for RADIUS accounting, for accounting the v6 user.</td>
</tr>
<tr>
<td>Aruba-Device-Type String 12</td>
<td>Sent by ArubaOS, contains the OS type of the client device as determined by ArubaOS.</td>
</tr>
<tr>
<td>Aruba-No-DHCP-Fingerprint Integer 14</td>
<td>This VSA signals the controller to not derive role and VLAN based on DHCP fingerprint.</td>
</tr>
<tr>
<td>Aruba-Mdps-Device-Udid String 15</td>
<td>Unique device identifier for an iOS device (UDID) is unique device identifier, which is used as input attribute by the Onboard application while performing the device authorization to the internal RADIUS server within the CPPM. UDID is used to check against role mappings or enforcement policies to determine if the device is authorized to be Onboarded.</td>
</tr>
<tr>
<td>Aruba-Mdps-Device-Imei String 16</td>
<td>International Mobile Equipment Identity (IMEI) is used as an input attribute by the Onboard application while performing the device authorization to the internal RADIUS server within the CPPM. IMEI is used to check against role mappings or enforcement policies to determine if the device is authorized to be Onboarded.</td>
</tr>
<tr>
<td>Aruba-Mdps-Device-Iccid String 17</td>
<td>Integrated Circuit Card Identifier (ICCID) is used as an input attribute by the Onboard application while performing the device authorization to the internal RADIUS server within the CPPM. ICCID is used to check against role mappings or enforcement policies to determine if the device is authorized to be Onboarded.</td>
</tr>
<tr>
<td>Aruba-Mdps-Max-Devices Integer 18</td>
<td>Used by Onboard as a way to define and enforce the maximum number of devices that can be provisioned by a given user.</td>
</tr>
<tr>
<td>Aruba-Mdps-Device-Name String 19</td>
<td>Device name is used as input attribute by the Onboard application while performing the device authorization to the internal RADIUS server within the CPPM. The device name is used to check against role mappings or enforcement policies to determine if the device is authorized to be Onboarded.</td>
</tr>
<tr>
<td>Aruba-Mdps-Device-Product String 20</td>
<td>Device Product is used as input attribute by the Onboard application while performing the device authorization to the internal RADIUS server within the CPPM. The Device Product is used to check against role mappings or enforcement policies to determine if the device is authorized to be Onboarded.</td>
</tr>
<tr>
<td>Aruba-Mdps-Device-Version String 21</td>
<td>Device Version is used as input attribute by the Onboard application while performing the device authorization to the internal RADIUS server within the CPPM. The Device Version is used to check against role mappings or enforcement policies to determine if the device is authorized to be Onboarded.</td>
</tr>
<tr>
<td>Aruba-Mdps-Device-Serial String 22</td>
<td>Device Serial number is used as an input attribute by the Onboard application while performing the device authorization to the internal RADIUS server within the CPPM. The Device Serial number is used to check against role mappings or enforcement policies to determine if the device is authorized to be Onboarded.</td>
</tr>
</tbody>
</table>
### Table 3: VSAs

<table>
<thead>
<tr>
<th>VSA Name</th>
<th>VSA Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aruba-AirGroup-User-Name String 24</td>
<td>Device owner – username associated with the device.</td>
</tr>
<tr>
<td>Aruba-AirGroup-Shared-User String 25</td>
<td>Contains a comma separated list of user names with whom the device is shared.</td>
</tr>
<tr>
<td>Aruba-AirGroup-Shared-Role String 26</td>
<td>Contains a comma separated list of user roles with which this particular device is shared.</td>
</tr>
<tr>
<td>Aruba-AirGroup-Device-Type Integer 27</td>
<td>1 for personal device and 2 for shared device.</td>
</tr>
<tr>
<td>Aruba-Auth-Survivability String 28</td>
<td>Used by the Authentication survivability feature for IAP to indicate CPPM server to send Aruba-AS-User-Name and Aruba-AS-Credential-Hash, attribute is just used as a flag and no specific value is used.</td>
</tr>
<tr>
<td>Aruba-AS-User-Name String 29</td>
<td>Used by the Authentication survivability feature for IAP. CPPM sends the actual username to the IAP, which can be used by IAP to authenticate the user if the CPPM server is not reachable.</td>
</tr>
<tr>
<td>Aruba-AS-Credential-Hash String 30</td>
<td>Used by the Authentication survivability feature for IAP. CPPM sends the NT Hash of the password to IAP, which can be used by IAP to authenticate the user if the CPPM server is not reachable.</td>
</tr>
<tr>
<td>Aruba-WorkSpace-App-Name String 31</td>
<td>This VSA is used by CPPM to implement per-APP-VPN and identify the application on the controller. A container VM running on iPad for example houses multiple applications; whenever a particular application initiates an Internet key exchange (IKE)/IPsec connection with the controller, provision is made to download role, VLAN, and so on, from CPPM based on the client application name.</td>
</tr>
<tr>
<td>Aruba-Mdps-Provisioning-Settings String 32</td>
<td>Used as part of the ClearPass Onboard technology, this attribute allows CPPM to signal back to the Onboard process the context of the device provisioning settings that should be applied to the device based on applied role mappings.</td>
</tr>
<tr>
<td>Aruba-Mdps-Device-Profile String 33</td>
<td>Used as part of the ClearPass Onboard technology, this attribute allows CPPM to signal back to the Onboard process the device profile that should be applied to the device based on applied role mappings.</td>
</tr>
<tr>
<td>Aruba-AP-IP-Adress Ipaddr 34</td>
<td>Sent by the RADIUS server to controller when RAP authenticates. The VSA contains the IP address, which is used as static inner IP for the RAP.</td>
</tr>
<tr>
<td>Aruba-AirGroup-Shared-Group String 35</td>
<td>Specifies the shared group list for the device, if this information is known. This is a comma-separated list of group names. This attribute is omitted if the shared group list is not set or blank.</td>
</tr>
<tr>
<td>Aruba-User-Group String 36</td>
<td>Specifies group(s) where the user is a member, if this information is known. This is a comma-separated list of group names. This attribute is omitted if the group list is not set or blank.</td>
</tr>
<tr>
<td>Aruba-Network-SSO-Token String 37</td>
<td>Specifies the single-sign on (SSO) token obtained as part of network level authentication from CPPM for a particular user.</td>
</tr>
<tr>
<td>VSA Name</td>
<td>VSA Function</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Aruba-AirGroup-Version Integer 38</td>
<td>This attribute is a 32-bit integer in network byte order. The allowable values for this attribute are: 1: for ArubaOS release before 6.4 and CPPM release before 6.3 2: for ArubaOS release 6.4 and CPPM release 6.3. New values shall be defined in future releases.</td>
</tr>
<tr>
<td>Aruba-Auth-SurvMethod Integer 39</td>
<td>It is used internally between AUTH and local SURVIVAVAL-SERVER in ArubaOS 6.4.3.x and higher to support auth-survivability. AUTH will add this VSA with other RADIUS attributes in an auth-request to Auth-Server (for example CPPM). If all servers are out-of-service (OOS) in the server-group, this auth-request with all RADIUS attributes (including VSA-39) will be sent to Local Survival-server.</td>
</tr>
<tr>
<td>Aruba-Port-Bounce-Host Integer 40</td>
<td>When this VSA is included with a DISCONNECT message, it will signal the Mobility Access Switch running ArubaOS 7.4.0.3 or higher to shut down the port for the duration specified (0 – 60 seconds).</td>
</tr>
<tr>
<td>Aruba-Calea-Server-Ip Ipaddr 41</td>
<td>Sent by the RADIUS server to RAP RADIUS client when a bridge mode client authenticates through captive-portal. The VSA contains the IP address of the Calea enforcement server to where the replicated data of the client should be sent.</td>
</tr>
</tbody>
</table>

Summary – When do Derived Rules Get Applied

Figure 9 is a summary of the order in which user derived rules and server derived rules get applied.

Figure 9  Summary - Order in which Derived Rules Get Applied

Below are examples that explain the use of user derivation rules and server derivation rules.
• Example Use for User Derived Rules on page 21
• Example Use for Server Derived Rules on page 21
• Example Use for Mixed Authentications on page 22

Example Use for User Derived Rules
In the retail and manufacturing sectors, there are:

• Two SSIDs in store, one for scanners, and one for employees.
• Scanner SSID is setup for Wired Equivalent Privacy (WEP) and restricted access, no authentication.
• Employee SSID is setup for employees’ daily applications.

Figure 10  Scanner Gun

User Derived Rule Configuration for Scanner SSID:

Rule - If user MAC address “starts-with” a specific organizationally unique identifier (OUI), set-role “Scanner”.

Example Use for Server Derived Rules
In the education sector, there is:

• One SSID, authenticating through the Web.
• Students and faculty get assigned different roles based on LDAP server group assignment.
• When a student roams to an area where no Internet access is allowed, Internet access is removed.

Figure 11  Server Derivation Rules
Server Derivation Rule Configuration:

- Rule #1 - If “groupmembership” = “faculty” set-role to “faculty”
- Rule #2 – If “location” (ap-name) = “99.99.1” set-role to “student-no-internet”
- Rule #3 – If “groupmembership” = “student” set-role to “student”

**Example Use for Mixed Authentications**

In wired security:

- Any port should support different authentication methods and access rights for guests, employees, phones, and printers.
- Employees use 802.1X.
- Guests use Web Authentication.
- Phones and printers are identified by OUI.

**Figure 12  Wired Security**

User Derived Rule Configuration for wired ports:

- Rule #1- If user MAC address “starts-with” specific phone OUI set-role “Phone”.
- Rule #2- If user MAC address “starts-with” printer OUI set-role “Printer”.

AAA profile:

- The initial-role has the captive portal profile applied; it is used for guests.
- 802.1X authentication is enabled; it is used for employees.
Role Derivation in Mixed Authentication Environment

This section includes the following topics:

- General Role Derivation Flow on page 23
- Layer 2 Role Priority on page 24
- Layer 3 Roles on page 25

**General Role Derivation Flow**

In the previous sections we looked at what comprises a user role and how the user role is derived using various types of user derived rules and/or server derived rules.

In the following sections we look at a general flow starting from authentication, clients assigned a initial role, and how pre-authentication and post-authentication roles are derived.

As summarized in Figure 13 below, each wireless client (or user) always starts with being in the 'logon' role. This means the wireless client has not yet performed Layer 2 authentication. The wired client has connected to an 'untrusted' port on the controller, and is waiting to go through Layer 2 authentication.

**Figure 13  General Role Derivation Flow**

```
System Default Role “logon”

AAA Profile Initial Role

UDR

Layer 2 Auth Role
(MAC Auth Default Role)
(Dot1x Machine Auth Role)
(Dot1x Default Role)

Layer 3 Auth role
(Captive Portal Role)
```
As previously mentioned, user derived rules, which are applicable in the pre-authentication stage of a client, get applied first. The client starts the Layer 2 authentication of any of the following types as an example: MAC Authentication and 802.1X Authentication. At the end of the Layer 2 authentication, the user is placed in the post-authentication role defined by the AAA profile for that WLAN or wired port, in case of a wired client.

In some cases, there might be another authentication step involved after the Layer 2 authentication. This would be Layer 3 authentication (typically captive portal), which will assign a Layer 3 authenticated role to the user.

**Layer 2 Role Priority**

Given there are various ways in which a role can be derived, either user-derived or server-derived, there is a certain hierarchy they follow.

- L2 role priorities are in increasing order; last one wins.
- L3 roles override L2 roles.
- Higher priority rules override lower priority rules, but not vice-versa.

**Figure 14** below shows the hierarchy of which rule overrides the other.

**Figure 14  Layer 2 Role Priority**
For example, the default logon role is always overridden by the initial role defined in the AAA profile. The role received as part of the change of attribute trigger from a RADIUS server for a given client takes priority over any previous Layer 2 role.

However, the Layer 3 role always overrides the Layer 2 role previously assigned to a user. An example of a Layer 3 role is one that is derived as part of captive portal authentication, for example.

**Layer 3 Roles**

The most commonly used Layer 3 roles are (in no particular order):

- Assigned from Captive Portal.
- Server derived rule (SDR) derived during Layer 3 authentication.
- VSA derived during Layer 3 authentication.
- Derived from VPN Authentication.
**Expected User Role in Mixed Authentication Environment**

Often in an environment multiple authentication methods are employed. For example, an administrator may like to authenticate the user’s device’s MAC address first, then perform the 802.1X authentication following that.

Hence the common question is, if MAC authentication succeeds, will the user get assigned the default role as configured in the AAA authentication profile for MAC authentication.

However, subsequently the user performs 802.1X authentication as well. Upon completion the user gets assigned a role within the 802.1X AAA profile. In this case the 802.1X role assigned is the final role in which the user is expected.

In such a mixed authentication environment, please refer to Table 4 below, which summarizes what the expected role for the user should be for various pass and fail criteria.

Table 4 includes the following abbreviations:
- c - configured
- P - pass
- F - fail

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Initial Role</th>
<th>MAC Authentication</th>
<th>802.1X User Authentication</th>
<th>Expected Role for User</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>c</td>
<td>P</td>
<td>P</td>
<td>802.1X</td>
</tr>
</tbody>
</table>
| 2        | c            | P                  | F                           | Wired user - MAC authentication default role  
                                                   Wireless user - no role, get de-authenticated |
| 3        | c            | F (L2 fail through enabled) | P                           | 802.1X default role    |
| 4        | c            | F                  | N/A                         | Initial role          |
If user derived rules (UDR) are in place in addition to a mixed authentication environment, please refer to Table 5 below for the expected final user role for the client, in case of different pass and fail criteria.

Table 5 includes the following abbreviations:

- P - pass
- F - fail
- nc - not configured

**Table 5: Expected User Role in Mixed Authentication Environment**

<table>
<thead>
<tr>
<th>Scenario</th>
<th>UDR</th>
<th>MAC Authentication</th>
<th>Machine Authentication</th>
<th>802.1X</th>
<th>Expected Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>P/F</td>
<td>P</td>
<td>P</td>
<td>nc</td>
<td>Machine authentication default role</td>
</tr>
<tr>
<td>2</td>
<td>P/F</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>802.1X default role</td>
</tr>
<tr>
<td>3</td>
<td>P/F</td>
<td>P</td>
<td>P</td>
<td>F</td>
<td>Wired user - Machine authentication default role Wiredless user - No Role, get de-authenticated</td>
</tr>
<tr>
<td>4</td>
<td>P/F</td>
<td>P</td>
<td>F</td>
<td>nc</td>
<td>MAC authentication default role</td>
</tr>
<tr>
<td>5</td>
<td>P/F</td>
<td>P</td>
<td>F</td>
<td>P</td>
<td>Machine authentication default user role</td>
</tr>
<tr>
<td>6</td>
<td>P/F</td>
<td>P</td>
<td>F</td>
<td>F</td>
<td>Wired user - MAC authentication default role Wiredless user - No role, get de-authenticated</td>
</tr>
<tr>
<td>7</td>
<td>P/F</td>
<td>F (L2 fail through enabled)</td>
<td>P</td>
<td>nc</td>
<td>Machine authentication default role</td>
</tr>
<tr>
<td>8</td>
<td>P/F</td>
<td>F (L2 fail through enabled)</td>
<td>P</td>
<td>P</td>
<td>802.1X default role</td>
</tr>
<tr>
<td>9</td>
<td>P/F</td>
<td>F (L2 fail through enabled)</td>
<td>P</td>
<td>F</td>
<td>Wired user - Machine authentication role Wiredless user - No role, get de-authenticated</td>
</tr>
<tr>
<td>10</td>
<td>P/F</td>
<td>F</td>
<td>F</td>
<td>nc</td>
<td>Role from UDR (if UDR passes); else initial-role</td>
</tr>
<tr>
<td>11</td>
<td>P/F</td>
<td>F</td>
<td>N/A</td>
<td>nc</td>
<td>Role from UDR (if UDR passes); else initial-role</td>
</tr>
<tr>
<td>12</td>
<td>P/F</td>
<td>F</td>
<td>N/A</td>
<td>nc</td>
<td>Role from UDR (if UDR passes); else initial-role</td>
</tr>
</tbody>
</table>
Role Derivation Flowchart in Various Authentications

We have looked at the definitions of user derived rules, server derived rules, role derivation in mixed authentication and also the priorities of which role should override the previous assigned role.

The flowcharts below are a summary of role derivation through various steps from UDR to multiple authentication to final role. The flowcharts are self-explanatory and capture the essence of user role derivation on Aruba WLANs.

The flowchart sequence assumes and covers the presence of initial Logon role, UDR, MAC authentication, roaming (PMK-caching) and 802.1X authentication.

Figure 15  Initial Role and UDR

Figure 16  MAC Authentication
Valid-user ACL

It is common with some devices to use static IP or retain the IP address received on other network connections. When these devices connect to the network, they try to use the IP subnet that is not part of the current network and consume an entry in the user table. If a network has a large number of devices that exhibit such behavior, the user table of the controller will be filled with foreign IP address that can't pass traffic. An easy solution to limit the devices that appear on the controller's user table is the use of valid-user ACL.

This section includes the following topics:
What is a Valid-user ACL?

The valid-user ACL is a special ACL that is checked against all user's source IP addresses before they are allowed to be in the user-table and thus before that IP is subjected to roles and policies. By default, this ACL is allowing anything into the table. While at first this may sound fine, some adverse side effects may occur such as:

- User table clutter may occur because multiple IP addresses from the same host will appear that may or may not be in use, such as:
  - The last IP the client had before they attached to the WLAN
  - 169.254.x.x auto-config address due to DHCP timeout issues
  - VMWARE IP addresses
- Connectivity issues to internal servers due to malicious or unintentional use of such addresses by WLAN (or other untrusted users like RAP wired, S2500/3500 wired, etc) clients. This can happen if a WLAN client sources a packet with a source IP other than its own that matches that of an internal server. This will become an entry in the user table and will be subjected to firewall rules and will appear as a local address to the Aruba controller.
- User-Table exhaustion (DoS) due to pseudo-random IP addresses being sourced by a client and sent into the WLAN and filling up the user table. There are a couple of ways to deal with this:
  - In Aruba 6.1 an later, a lot of this can be resolved by using the "Enforce DHCP" option in the AAA profile to ensure a client only uses the IP address that was given to it via DHCP.
  - Manually modify the valid-user ACL to permit only the subnets that clients should be pulling IP addresses from and deny from all others, for example, to allow only client subnet 10.1.1.0/24 in the user table, do the following:

  ip access-list session validuser' no any any any permit' no ipv6 any any any permit' network 10.1.1.0 255.255.255.0 any any permit' any any any deny

  This process can be cumbersome and error prone, and it doesn't help that this is "global" in a master/local topology. Much care should be taken if this ACL is modified.

- As an alternative to manually modifying the valid-user ACL, in ArubaOS 3.4 and later, use the "Only allow local subnets in the user table" option under the Advanaced>Stateful Firewall section of the controller configuration. This will look at the locally configured subnets and automatically allow those while denying everything else. One consideration with this requires that the VLANs you are putting user traffic onto have a IP address configured on it.

How it Works?

The valid-user ACL controls the IP addresses that can enter the controller's user table through all the untrusted interfaces. Any traffic entering the system is first processed by this ACL before being processed by other ACLs and user roles. The controller denies network access to any client that uses an IP address denied by this rule. Valid-user ACL is a special kind of internal system ACL that should not be added to any user role.

The ArubaOS have a default predefined valid-user ACL that can be modified but a new valid-user ACL cannot be created. As such, the configuration parameters values acceptable in regular ACLs are not applicable in the valid-user ACL. The valid-user ACL processes traffic differently than a standard ACL, so the configuration of rules for valid-user ACL varies as follows:
Like the standard session ACL configuring a rule in the valid-user ACL requires defining the source, destination, service and action fields. However, while processing the rules, the valid-user ACL considers only the source and action field. In other words, the destination and services fields in a rule are ignored by the valid-user ACL. So, a rule "any host 10.10.3.10 tcp 80 deny" will deny all traffic from any user because only the source and action field are relevant.

They keyword "user" cannot be used in the source field of the rules in the valid-user ACL. So, a rule "user any any permit" cannot be used. Instead, use "any any any permit".

Since the destination and services fields in a rule are ignored by the valid user ACL, the keyword for these fields is "any" by default. So, a rule "any host 10.10.3.10 tcp 80 deny" will be considered as "any any any deny" by the valid-user ACL.

The keyword in the action field should only be "permit" or "deny"

**Valid-user ACL Configuration**

The default predefined valid-user ACL is:

```
ip access-list session validuser
  network 169.254.0.0 255.255.0.0 any any deny
  any any any permit
  ipv6 any any any permit
```

If a network administrator wants to deny 169.254.0.0/16 and allow only the subnets 10.10.1.0/24, 10.10.2.0/24 and 10.10.3.0/24 then the valid-user ACL should be:

```
ip access-list session validuser
  network 169.254.0.0 255.255.0.0 any any deny
  network 10.10.1.0 255.255.255.0 any any permit
  network 10.10.2.0 255.255.255.0 any any permit
  network 10.10.3.0 255.255.255.0 any any permit
```

Do not use valid-user ACL to deny traffic to certain destinations or to deny certain services. The following ACL is wrong and will deny all 10.10.1.0 users and not just the traffic to 10.10.3.10 on port 80.

```
ip access-list session validuser
  network 169.254.0.0 255.255.0.0 any any deny
  network 10.10.1.0 255.255.255.0 host 10.10.3.10 tcp 80 deny --> incorrect use (denies the entire 10.10.1.0/24 subnet and not just the traffic to 10.10.3.10 on port 80)
  any any any permit
```

**Firewall Local-Valid-Users**

Configuring valid-user ACL on controllers with hundreds of subnets can be very cumbersome. The “firewall local-valid-users” feature simplifies the configuration of valid-user ACL. When the “firewall local-valid-users” feature is enabled, only the IP subnets corresponding to L3 interfaces on the controller are considered valid. In other words, for a subnet to be considered valid by the “firewall local-valid-users”, the controller should have an L3 interface that belongs to that subnet. Only client devices belonging to the valid subnets are allowed network access.

For the “firewall local-valid-users” feature to be active, the valid-user ACL should have only an “any any any deny” rule. Remember that the presence of any other rule other than “any any any deny” in the valid-user ACL automatically disables the “firewall local-valid-users” feature. In general the configuration of valid-user ACL requires these two steps:
1. Enable the “firewall local-valid-users” feature.
   
   (config) #firewall local-valid-users

2. Modify the valid-user ACL to include only the “any any any deny” rule.
   
   ip access-list session validuser
   any any deny

The “firewall local-valid-users” feature should not be used if one or more valid subnets are defined as L2 interfaces on the controller. In these cases, use the standard valid-user ACL to define all the valid subnets.

---

**Best Practices**

**Block critical network resources**

All networks depend on communication with critical resources such as DNS servers, DHCP servers, and gateway routers. Any untrusted user that intentionally or accidentally spoofs one of these resources can severely impact network performance. The example below illustrates use of wildcard masks and aliases (netdestinations) to prevent critical resources from appearing in the user-table:

```
netdestination guest-networks
 network 10.252.0.0 255.255.0.0
 network 10.242.0.0 255.255.0.0
!
netdestination guest-gateways
 network 10.252.0.254 255.255.0.255
 network 10.242.0.254 255.255.0.255
!
netdestination internal-gateways
 network 10.0.0.1 255.0.0.255
 network 10.0.0.254 255.0.0.255
!
netdestination ACME-Approved-Public-DNS
 host 8.8.8.8
!
netdestination ACME-Internal-DNS
 host 10.2.3.4
 host 10.5.6.7
!
netdestination ACME-Internal-DHCP
 host 10.8.9.10
 host 10.11.12.13
!
ip access-list session validuser
 alias ACME-Approved-Public-DNS any any deny
 alias guest-gateways any any deny
 alias guest-networks any any permit
```
alias ACME-Internal-DNS any any deny
alias ACME-Internal-DHCP any any deny
alias internal-gateways any any deny
any any any permit
!

**Firewall ACE Entries**

In ArubaOS, an ACE entry is an internal data structure that is used to enforce firewall policies. Essentially, ACLs are translated into one or more ACE entries that are used to enforce the actual roles configured in the firewall.

This section includes the following topics:
- [How many ACE entries are there? on page 33](#)
- [How are ACE entries used? on page 33](#)
- [Viewing ACE Entries on page 35](#)
- [What to do when the ACE Table is full? on page 35](#)

**How many ACE entries are there?**

Regardless of platform, there are in all 8K (2^13=8192 total) ACE entries that one can configure. Out of these 8,192 ACE entries available, the last 512 had been allocated to cp-firewall, the rest can be used by the regular ACLs. That leaves a total of 7,680 ACE entries available in the table. This limit applies to all controller platforms up to and including all 8.0.1, and 8.1 versions of AOS.

**How are ACE entries used?**

In AOS, the nomenclature is as follows:

ACE = Access Control Entry, used per rule or policy within an ACL or for each line in an alias within an ACL

Rule/Policy = Individual firewall rule in an ACL

ACL = Access Control List

User Role = Container with one or more ACLs within it

User-role contains a list of ACLs. ACL contains a list of firewall rules. A firewall rule/policy can get expanded to multiple ACEs, depending on the type of firewall rule.

Example 1: This is a very simple ACL, which is a copy of the logon-control, but we will modify this as part of the demo.

```
ip access-list session ace-entry-demo-1
  user any udp 68 permit
  any any svc-icmp permit
  any any svc-dhcp permit
  any any svc-natt permit
  any network 169.254.0.0 255.255.0.0 any deny
  any network 240.0.0.0 240.0.0.0 any deny
!
```

Each policy within the ACL 'ace-entry-demo-1' consumes one ACE entry, so the total ACE entry consumption for the above is 6 ACE entries.

Example 2 - This is another simple ACL, but it will consume a LOT of ACE entries because of the way it’s built.
Below are two simple netdestinations, each containing four different /24 networks each (this could be any four VLANs on the network). Note that each network within each net destination will consume one ACE entry when used in an ACL.

```
netdestination ace-demo-netdest-1
    network 1.0.1.0 255.255.255.0
    network 1.0.2.0 255.255.255.0
    network 1.0.3.0 255.255.255.0
    network 1.0.4.0 255.255.255.0
! 
netdestination ace-demo-netdest-2
    network 1.1.1.0 255.255.255.0
    network 1.1.2.0 255.255.255.0
    network 1.1.3.0 255.255.255.0
    network 1.1.4.0 255.255.255.0
! 
```

The following ACL, using the netdestinations above, is built to allow HTTP, HTTPS, and SSH between one group of networks to another group of networks.

```
ip access-list session ace-entry-demo-2
    alias ace-demo-netdest-1 alias ace-demo-netdest-2 svc-http permit
    alias ace-demo-netdest-1 alias ace-demo-netdest-2 svc-https permit
    alias ace-demo-netdest-1 alias ace-demo-netdest-2 svc-ssh permit
    alias ace-demo-netdest-2 alias ace-demo-netdest-1 svc-http permit
    alias ace-demo-netdest-2 alias ace-demo-netdest-1 svc-https permit
    alias ace-demo-netdest-2 alias ace-demo-netdest-1 svc-ssh permit
!
```

Because of how the above is constructed, *each line* in the above ACL actually contains 16 ACE entries each (4 ACE in the src * 4 ACE in the dst = 16 ACE entries per line for each combination:

- Network-List-1 to Network-List-2 on HTTP = 4*4=16 ACE
- Network-List-1 to Network-List-2 on HTTPS = 4*4=16 ACE
- Network-List-1 to Network-List-2 on SSH = 4*4=16 ACE
- Network-List-2 to Network-List-1 on HTTP = 4*4=16 ACE
- Network-List-2 to Network-List-1 on HTTPS = 4*4=16 ACE
- Network-List-2 to Network-List-1 on SSH = 4*4=16 ACE

Total ACE entries are 16 ACE entries per line * 6 lines = 96 ACE entries

So looking forward, because each role has it’s own ACE entry consumption calculation, if the above ACL was used in 10 different roles, that would be 970 ACE entries consumed off the table (nearly 13% of the entire ACE Table)! This is why the number of Roles created should be as minimal as possible.

The above example is a perfect illustration of how ACE entry consumption can explode with certain construction topologies of ACL. The 'math' of the ACE entry calculation is approximately:

Number of lines in each ACL * Number of lines in each src or dst netdestination used (* number of lines in used in dst if present) + the role(s) applied (* number of roles containing the ACL) = Total ACE Entries consumed.
In the above Example 1:
6 lines in the ACL + 1 role applied to (*1 role applied in) = 7 ACE entries consumed

In the above Example 2:
6 lines in the ACL* (4 lines in the SRC * 4 lines in the DST) + 1 role applied to (*1 role applied in) = 97 ACE entries consumed

Viewing ACE Entries

show acl acl-table
AclTable
---------
ACL Type ACE Index Rule Count Ace Count Name Applied
--------- -------- -------- --------- ---------- -----------
1 session 0 0 1 global-sacl 0
2 role 1035 32 33 logon 0
3 session 887 11 12 validuser 0
4 session 1201 27 28 sdn-acl 0
5 route 684 0 1 uplink-lb-cfg-racl 0
6 route 685 0 1 uplink-lb-sys-racl 0

The top outlines what each column represents:
- ACL - the ACL number of each ACL
- Type - shows what type of ACL it is (role, session, route, ether-type, or geolocation)
- ACE Index - This shows the first ACE entry consumed by the ACL (if ACE Index 3,433 is the next available, a new ACL will start with ACE Index 3433)
- Rule Count - shows how many rules are in each ACL
- ACE Count - Total of ACE entries consumed by the ACL
- Name - Name of the ACL to be found in the config
- Applied - TBD, engineering is returning an answer, for now ignore it.

At the bottom of the output is a summary of the current ACE entries in use on the system, such as the following:
Total ACE entries in use = 509
Total free ACE entries = 7171
Free ACE entries at the bottom = 7101
Next ACE entry to use = 579 (table 1)
ACE entries reused 1 times
ACL count 77, tunnel acl 0

What to do when the ACE Table is full?
At some point if the ACE table gets full and the customer keeps loading large or complex ACLs, the steps to look at to free up entries are:
- Are ALL ACLs created on the controller actively being used in a role? If not, delete all unused roles.
- Are all ACL as efficient as they could be? If there are ACLs that use the same netdestination alias for source AND destination, that is not really an efficient ACL. In that case, the SRC could be 'user' because that role is
already applied to a specific VAP/User group, there's no need to really specify the SRC in most all cases. That change alone can free up HALF of the ACE entries in that specific case.

- Can the number of ACLs in a single role, or can the number of roles be reduced? If there are dozens of roles on the controller, but most of them are the same, reduce the number of roles and assign them appropriately to the required VAPs.
- If all else fails and the ACE table is full, and there are no reductions possible, then a new master/cluster will need to be established to address this and modify the config as needed to split the load.
When we talk about security for wireless users, there are two aspects: authentication and encryption.

Authentication is a method of validating the identity of users on the network. It ensures that users who are connecting to your network can be trusted. All Aruba mobility controllers are equipped with an integrated firewall that provides role-based access to users. Based on a user’s identity, different access privileges can be granted to users.

For example, in office environments, two SSIDs are typically advertised for end-users. There is an "Employee" SSID that authenticates users based on their 802.1X credentials. 802.1X authentication is an L2 type of authentication, meaning users need to be authenticated on the network before they can receive an IP address and access network resources. There is also a “Guest” SSID that allows users to connect to the network and receive an IP address. These guest users are then subjected to L3 authentication. A captive portal page either authenticates them by their guest credentials (these credentials may be provided to the guest via email, SMS, or by the guest’s sponsor), or allows guests to self register themselves (by inputting basic personal information and viewing/accepting terms and conditions), before they are granted Internet access. Based on their guest credentials or self-registration information, roles are applied to these guest users that determine the level of access they have to network resources. Typically, guest users are only allowed Internet access. Access to internal and corporate resources is completely blocked for these users.

This chapter includes the following sections:

- Captive Portal Authentication on page 38
- Captive Portal with MAC Caching via CPPM on page 52
- MAC Authentication on page 57
- Guest Access with Self Registration on page 59
- 802.1X Authentication on page 62
- VPN Authentication on page 79
Captive Portal Authentication

Captive portal is an L3 method of authentication supported by ArubaOS. A captive portal presents a web page that requires user action before network access is granted. The user action can be simply viewing and agreeing to an “acceptable use” policy, or entering a user ID and password that must be validated against an internal or external database of authorized users.

A captive portal can serve both new and registered guest users and is customizable.

The flow chart in Figure 19 describes the captive portal authentication process.

Figure 19  Captive Portal Authentication Flow Chart

1. The user connects to the guest network and the controller assigns an initial guest role. The user initiates an HTTP/HTTPS request to a web server on the Internet.
2. An ACL that is part of the initial guest role redirects all of the HTTP/HTTPS traffic to the controller’s internal web server (on ports 8080 and 8081 for HTTP and HTTPS, respectively).
3. After the user gets redirected, they are presented with a captive portal login page.
4. The user logs in with a username and password, or registers themselves by providing an email address.
5. If the authentication is successful, the user is assigned a post-web-authentication user role (proceed to step #6). If authentication fails, the user is redirected back to the captive portal login page (step #3).
6. The user is either presented with the controller’s welcome page or redirected to the URL that was initially requested by the user in Step #1.

7. The user’s post-authentication user-role generally allows access to internet resources and blocks access to corporate resources.

Figure 20 describes the packet flow process when the controller’s internal captive portal is used as the landing page for guest logins.

**Figure 20 Packet Flow Process**

1. The user associates to the guest SSID, receives an IP address, and is assigned an initial user-role.

2. When the client starts browsing for a specific URL on the internet, it initially performs a DNS query to resolve the URL, and then initiates an HTTP/HTTPS GET request to that URL.

3. The DST-NAT ACL that is part of the guest’s initial user role causes the controller to intercept the HTTP/HTTPS traffic from the client and redirect the client to the controller’s login URL. A packet capture of this process shows an “HTTP GET” message from the client being responded to by an “HTTP 302 Temporarily Moved” message that includes the controller’s captive portal login URL.

4. The user performs a DNS query for the controller’s captive portal login page URL and receives a DNS response. Upon contacting the controller’s URL, the user is presented with the controller’s captive portal page (HTTPS GET/HTTPS response).

5. The user supplies guest credentials, if available. First-time users go through a self-registration process, typically by supplying basic information and accepting terms and conditions of the network. When the user submits guest credentials, the controller forwards these credentials to a RADIUS server as part of RADIUS exchange messages. The RADIUS server informs the controller upon successful authentication and the controller assigns a post-authentication user-role to the guest user.

6. The client is now either directed to a welcome page, if one is configured on the controller, or is redirected to the URL that the user was initially trying to access. The post-authentication user role should ensure that the guest can reach internet resources but cannot access any internal resources. Bandwidth contracts may also be applied to the post-authentication user-role to limit the bandwidth of guest traffic, especially if the internet uplink has limited bandwidth.
This section includes the following topics:

- Packet Captures on the Client on page 41
- CPPM Server Initiated Web Login on page 44
- CPPM Controller Initiated Web Login on page 46
- Captive Portal Configuration Flow on page 47
- CP Whitelist or Walled Garden on page 48
- Apple CNA Bypass on page 49
- Bypassing Captive Portal Landing Page on page 49
- Tri-Session DNAT for CP User VLAN Without IP on page 50
- Captive Portal Configuration Recommendations on page 51
Packet Captures on the Client

The packet capture example has the following details:

*Client IP: 10.70.25.188*

*DNS IP: 10.44.10.11*

*Destination URL IP: 68.142.68.6*

1. This packet capture (pcap) example below shows a client (10.70.25.188) trying to resolve www.brocade.com, which is the URL to which the client wants to connect. The DNS server (10.44.10.11) sends a DNS response with the IP address of www.brocade.com (68.142.68.6).

   **Figure 21 Packet Capture 1**

2. In this pcap, the client sends an HTTP GET message to the resolved URL (68.142.68.6) but the controller intercepts this traffic and responds to the client with an “HTTP 302 Temporarily Moved” message. The body of this message contains the redirect URL, to which the client needs to connect. This redirect URL is the controller’s captive portal URL (securelogin.arubanetworks.com). A more detailed pcap for this step is shown below.

   **Figure 22 Packet Capture 2**

This is an elaboration of Step 2 of the packet capture process described above.

Redirecting traffic to the controller via an HTTP 302 message:
The following events occur as illustrated in the pcap:

- The controller intercepts the HTTP/HTTPS traffic, and replaces the user’s default gateway MAC with its own MAC (see the Ethernet header).
- It sends the HTTP 302 temporarily moved message to the user (see the HTTP header).
- It redirects the traffic to the CP login page: https://securelogin.arubanetworks.com (see the HTTP header).

3. The client performs a DNS query for securelogin.arubanetworks.com and receives a response from the DNS server, that is, the IP address of the controller (10.70.25.16).
4. The user upon contacting the controller (HTTPS GET/HTTPS response) is presented with a captive portal page with a username and password field. The user enters the username and password and submits the form.

\[\text{Client IP: 10.70.25.188} \]
\[\text{Controller IP: 10.70.25.16} \]

**Figure 25 Packet Capture 4**

<table>
<thead>
<tr>
<th>376</th>
<th>10.70.25.188</th>
<th>10.70.25.16</th>
<th>TCP</th>
<th>55030 &gt; https [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=8 SACK_PERM</th>
</tr>
</thead>
<tbody>
<tr>
<td>377</td>
<td>10.70.25.188</td>
<td>10.70.25.188</td>
<td>TCP</td>
<td>https &gt; 55030 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1386 SACK</td>
</tr>
<tr>
<td>378</td>
<td>10.70.25.188</td>
<td>10.70.25.188</td>
<td>TCP</td>
<td>55030 &gt; https [ACK] Seq=1 Ack=1 Win=66304 Len=0</td>
</tr>
<tr>
<td>379</td>
<td>10.70.25.188</td>
<td>10.70.25.188</td>
<td>TLSv1.2Server Hello</td>
<td></td>
</tr>
<tr>
<td>380</td>
<td>10.70.25.188</td>
<td>10.70.25.188</td>
<td>TCP</td>
<td>https &gt; 55030 [ACK] Seq=1 Ack=518 Win=7168 Len=0</td>
</tr>
<tr>
<td>381</td>
<td>10.70.25.16</td>
<td>10.70.25.188</td>
<td>TLSv1.2Client Hello</td>
<td></td>
</tr>
<tr>
<td>382</td>
<td>10.70.25.16</td>
<td>10.70.25.188</td>
<td>TCP</td>
<td>[TCP segment of a reassembled PDU]</td>
</tr>
<tr>
<td>383</td>
<td>10.70.25.16</td>
<td>10.70.25.188</td>
<td>TCP</td>
<td>55030 &gt; https [ACK] Seq=518 Ack=2773 Win=66304 Len=0</td>
</tr>
<tr>
<td>384</td>
<td>10.70.25.16</td>
<td>10.70.25.188</td>
<td>TLSv1.2Certificate, Server Hello Done</td>
<td></td>
</tr>
<tr>
<td>385</td>
<td>10.70.25.188</td>
<td>10.70.25.16</td>
<td>TLSv1.2Client Key Exchange, Change Cipher Spec, Encrypted Handshake Mes:</td>
<td></td>
</tr>
<tr>
<td>386</td>
<td>10.70.25.16</td>
<td>10.70.25.188</td>
<td>TLSv1.2Encrypted Handshake Message, Change Cipher Spec, Encrypted Handsh</td>
<td></td>
</tr>
<tr>
<td>387</td>
<td>10.70.25.16</td>
<td>10.70.25.188</td>
<td>TLSv1.2Application Data</td>
<td></td>
</tr>
<tr>
<td>388</td>
<td>10.70.25.16</td>
<td>10.70.25.188</td>
<td>TLSv1.2Application Data, Application Data</td>
<td></td>
</tr>
<tr>
<td>389</td>
<td>10.70.25.16</td>
<td>10.70.25.188</td>
<td>TCP</td>
<td>[TCP segment of a reassembled PDU]</td>
</tr>
<tr>
<td>390</td>
<td>10.70.25.16</td>
<td>10.70.25.188</td>
<td>TCP</td>
<td>[TCP segment of a reassembled PDU]</td>
</tr>
<tr>
<td>391</td>
<td>10.70.25.16</td>
<td>10.70.25.188</td>
<td>TCP</td>
<td>55030 &gt; https [ACK] Seq=1969 Ack=7761 Win=66304 Len=0</td>
</tr>
<tr>
<td>392</td>
<td>10.70.25.16</td>
<td>10.70.25.188</td>
<td>TLSv1.2Application Data, Application Data</td>
<td></td>
</tr>
<tr>
<td>393</td>
<td>10.70.25.16</td>
<td>10.70.25.188</td>
<td>TLSv1.2Application Data, Application Data</td>
<td></td>
</tr>
<tr>
<td>394</td>
<td>10.70.25.16</td>
<td>10.70.25.188</td>
<td>TCP</td>
<td>55030 &gt; https [ACK] Seq=1969 Ack=9511 Win=66304 Len=0</td>
</tr>
<tr>
<td>395</td>
<td>10.70.25.16</td>
<td>10.70.25.188</td>
<td>TCP</td>
<td>55030 &gt; https [FIN, ACK] Seq=1969 Ack=9511 Win=66304 Len=0</td>
</tr>
<tr>
<td>396</td>
<td>10.70.25.16</td>
<td>10.70.25.188</td>
<td>TCP</td>
<td>https &gt; 55030 [ACK] Seq=9511 Ack=1970 Win=10240 Len=0</td>
</tr>
</tbody>
</table>

5. The controller, upon successfully authenticating the user (via a RADIUS server, for example) sends an “HTTP 301 Moved Permanently” message with the URL for a welcome page (if configured on the controller) or the URL that was initially attempted by the client (www.brocade.com). The user is now able to access www.brocade.com, provided this URL is permitted by the ACLs defined in the post-authentication user-role.

\[\text{Client IP: 10.70.25.188} \]

\[\text{Destination URL IP: 68.142.68.6} \]

**Figure 26 Packet Capture 5**

<table>
<thead>
<tr>
<th>465</th>
<th>10.70.25.188</th>
<th>68.142.68.6</th>
<th>TCP</th>
<th>55030 &gt; http [SYN] Seq=0 Win=6192 Len=0 MSS=1460 WS=8 SACK_PERM</th>
</tr>
</thead>
<tbody>
<tr>
<td>466</td>
<td>68.142.68.6</td>
<td>10.70.25.188</td>
<td>TCP</td>
<td>http &gt; 55036 [SYN, ACK] Seq=0 Ack=1 Win=65335 Len=0 MSS=1386 WS=</td>
</tr>
<tr>
<td>467</td>
<td>10.70.25.188</td>
<td>68.142.68.6</td>
<td>TCP</td>
<td>55036 &gt; http [ACK] Seq=1 Ack=1 Win=66304 Len=0</td>
</tr>
<tr>
<td>468</td>
<td>10.70.25.188</td>
<td>68.142.68.6</td>
<td>HTTP GET / HTTP/1.1</td>
<td></td>
</tr>
<tr>
<td>469</td>
<td>68.142.68.6</td>
<td>10.70.25.188</td>
<td>TCP</td>
<td>http &gt; 55036 [ACK] Seq=1 Ack=373 Win=66144 Len=0</td>
</tr>
<tr>
<td>470</td>
<td>68.142.68.6</td>
<td>10.70.25.188</td>
<td>HTTP GET/1.1 301 Moved Permanently</td>
<td></td>
</tr>
<tr>
<td>471</td>
<td>10.70.25.188</td>
<td>68.142.68.6</td>
<td>HTTP GET /en.html HTTP/1.1</td>
<td></td>
</tr>
<tr>
<td>472</td>
<td>68.142.68.6</td>
<td>10.70.25.188</td>
<td>TCP</td>
<td>http &gt; 55036 [ACK] Seq=179 Ack=752 Win=66144 Len=0</td>
</tr>
</tbody>
</table>
CPPM Server Initiated Web Login

CPPM server initiated web login is the recommended method of captive portal authentication since it is less impactful on the controller’s web process. The user role is assigned via the RADIUS CoA packet.

**Figure 27  CPPM Server Initiated Web Login Packets Flow**

Following are the steps in the CPPM server initiated web login packets flow:

1. The user associates to the guest SSID, receives an IP address, and is assigned an initial user-role.
2. When the client starts browsing for a specific URL on the Internet, it initially performs a DNS query to resolve the URL and then initiates an HTTP/HTTPS GET request to that URL.
3. The destination network address translation (DST-NAT) ACL that is part of the guest’s initial user role causes the controller to intercept the HTTP/HTTPS traffic from the client and redirect the client to the ClearPass server’s login URL. A packet capture of this process would show an “HTTP GET” message from the client being responded to by an "HTTP 302 Temporarily Moved" message that includes the ClearPass server’s URL.
4. The user performs a DNS query for the ClearPass URL that was provided as part of the “HTTP 302 Temporarily Moved” message and receives a DNS response.
5. Upon contacting the ClearPass URL, the user is presented with the ClearPass captive portal page (HTTPS GET/HTTPS response) with a login form.
6. The user supplies guest credentials, if available. First-time users go through a self-registration process, typically by supplying basic information and accepting terms and conditions of the network. When the user submits the guest credentials, ClearPass authenticates these credentials (or forwards the credentials to another RADIUS server, if one is configured).
7. ClearPass sends a login message to the client with a delay of 60 seconds.
8. During this 60 second period, CPPM sends a RADIUS Disconnect message to the controller, prompting the controller to send a RADIUS Request message to ClearPass (authentication request). ClearPass validates the guest credentials and responds with a RADIUS Accept message. It also sends a RADIUS CoA message requesting the controller to assign a captive portal post-authentication role to the guest user. The controller then assigns the post-authentication user-role to the guest user.
9. Post the 60 second delay, the client receives the login message and is redirected to the URL that was originally requested by the user. Since the controller has already completed the authentication process for the user, the user is placed in a post-authentication user role. This user role should make sure that the guest can reach Internet resources but cannot access any internal resources. Bandwidth contracts may also be applied to the post-authentication user-role to limit the bandwidth of guest traffic, especially if the Internet uplink has limited bandwidth.
CPPM Controller Initiated Web Login

You can also use CPPM controller initiated web login for captive portal authentication.

**Figure 28 CPPM Controller Initiated Web Login Packets Flow**

Following are the steps in the CPPM controller initiated web login packets flow:

1. The user associates to the guest SSID, receives an IP address, and is assigned an initial user-role.

2. When the client starts browsing for a specific URL on the Internet, it initially performs a DNS query to resolve the URL, and then initiates an HTTP/HTTPS GET request to that URL.

3. The destination network address translation (DST-NAT) ACL that is part of the guest’s initial user role causes the controller to intercept the HTTP/HTTPS traffic from the client and redirect the client to the ClearPass server’s login URL. A packet capture of this process shows an “HTTP GET” message from the client being responded to by an "HTTP 302 Temporarily Moved” message that includes the ClearPass server’s URL.

4. The client device performs a DNS query for the ClearPass URL that was provided as part of the “HTTP 302 Temporarily Moved” message and receives a DNS response.

5. Upon contacting the ClearPass URL, the user is presented with the ClearPass captive portal page (HTTPS GET/HTTPS response) with a login form.

6. The user provides the login details and submits the form after which ClearPass responds with a hidden JavaScript message prompting the client device to post the form to the controller at the controller’s URL. This step is transparent to the user and no user interaction is needed here.

7. The client performs a DNS query to resolve the controller’s URL.

8. The client posts the form to the controller (no user interaction needed). The controller then authenticates the client by sending a RADIUS request to ClearPass.

9. If the authentication is successful, the user is assigned a post-authentication user role and then is redirected to the initially requested URL. The post-authentication user role should make sure that the guest can reach Internet resources but cannot access any internal resources. Bandwidth contracts may also be applied to the...
post-authentication user-role to limit the bandwidth of guest traffic, especially if the Internet uplink has limited bandwidth.

**Captive Portal Configuration Flow**

The flow illustrated below shows the different elements that make up the captive portal configuration on the controller and what the output for each element looks like after configuration.

This flow can also be visualized from a top-down level where you configure an AP Group and server and CA certificates on the controller. Under the AP Group, we have the Virtual AP profile. The Virtual AP profile has the AAA and SSID profiles. The AAA profile has the guest’s initial user role, which points to a captive portal profile. The captive portal profile contains four configuration elements: the server group, login page, welcome page, and post-authentication user role. Finally, the server group holds the server(s) required to authenticate guest users.

**Figure 29 Captive Portal Configuration Flow**
CP Whitelist or Walled Garden

Captive portal profile configuration involves different fields such as post-authentication user-role, login page, welcome page, and so on. It also includes a field called Whitelist (also referred to as Walled Garden). A whitelist is optional and may contain 'netdestinations' (lists of domain names, URLs) that the client is allowed to access even before captive portal authentication. This can be beneficial in cases where access to certain Web sites needs to be allowed (for online certificate status protocol (OCSP) checking, for instance) before captive portal authentication takes place. The client is redirected to the captive portal page if it tries to access any URLs outside of the whitelist.

When a client attempts to access a URL that is whitelisted, the controller snoops the corresponding DNS requests and responses and if these URLs are present in the whitelist, the controller automatically creates an ACL at position 1 in the pre-authentication user-role. All URLs that are part of the whitelist are automatically added to this ACL as “permitted” URLs. As a result, user traffic to all whitelisted URLs is permitted instead of being redirected to the captive portal login page.

Below are examples of what whitelists and netdestinations look like. You would first create a netdestination (or multiple netdestinations) with all the permitted URLs. Then apply the netdestination(s) to the captive portal profile whitelist.

1. URLs that commonly need to be whitelisted are listed below. These URLs are accessed by the client as part of network setup and detection.
   - Google Play: android.clients.google.com – for Google Play access
   - .ggpht.com – to download an app from Google Play store
   - Amazon Market: amzdigitaldownloads.edgesuite.net

2. Do not forget to add the OCSP checking URL from the CP server certificate into the whitelist, also.

```bash
netdestination whitelist
   name ocsp.comodoca.com
!
aaa authentication captive-portal "test-captive-portal"
   white-list "whitelist"
!
user-role test-logon
   captive-portal "test-captive-portal"
   access-list session guest-control
   access-list session captiveportal
```

You can observe the dynamic ACL created by this whitelist using the 'show rights' command in the CLI:

```bash
# show rights test-logon
Captive Portal profile = test-captive-portal

access-list List
1   test-captive-portal_list_operations
2   guest-control
3   captiveportal
test-captive-portal_list_operations
            ------ ------ -------------- ------ ------ -------------- ------ ------
            1   user   whitelist svc-http permit Low
            2   user   whitelist svc-https permit Low
```
Apple CNA Bypass

Apple has historically used the Apple Captive Network Assistant (CNA) in Apple clients that access [http://www.apple.com/library/test/success.html](http://www.apple.com/library/test/success.html). But since iOS 7, this Apple CNA website is no longer being used and some new websites have been added for CNA, which may not be listed in the configured whitelist.

The “Bypass Apple CNA” feature handles the new behavior of iOS 7 devices as well as older iOS devices. No whitelist for Apple devices is needed when this feature is enabled.

It is recommended to enable Bypass Apple CAN if there are Apple devices connecting to the captive portal SSID.

Bypassing Captive Portal Landing Page

When clients, especially smartphones and tablets, are connected to a captive portal enabled SSID and are initially placed, in a pre-authentication role, a number of non-browser based applications on these devices repeatedly request the captive portal login page from the controller. This impacts the number of browser-based logins the controller can handle per second and thereby delays the process of loading the captive portal and completing authentication.

Enabling the “Bypass Captive Portal Landing Page” feature allows the controller to send a "200 OK" status code message to non-browser based apps so that they stop sending repeated requests to the controller. This in turn helps reduce the load on the httpd process on the controller.

This feature is enabled by default on the controller.
Tri-Session DNAT for CP User VLAN Without IP

This feature allows a three-way session when performing destination NAT during scenarios where the controller is not the default gateway for wireless clients and the default gateway is actually behind the controller.

By default, the controller can only send and forward Syn Ack packet back to the client where it receives Syn packet.

Tri-session with DNAT is disabled by default. If the user’s gateway is the router and the controller does not have an IP address for the user VLAN, enable the “Tri-session DNAT” feature under the firewall.

Figure 30  Tri-Session DNAT
### Captive Portal Configuration Recommendations

Table 6 describes the captive portal configuration recommendations for each listed feature.

**Table 6: Captive Portal Configuration Recommendations**

<table>
<thead>
<tr>
<th>Feature</th>
<th>Default</th>
<th>Recommendation</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>CP server certificate</td>
<td>Controller default certificate</td>
<td>Customer's own certificate</td>
<td>Built-in certificate is for demonstration purposes only.</td>
</tr>
<tr>
<td>Bypassing Captive Portal Landing Page</td>
<td>Disable</td>
<td>Enable</td>
<td>Non-web based smart phones and tablets are almost in every Wi-Fi network. If captive portal is configured, enabling this feature helps to ease the CPU usage.</td>
</tr>
<tr>
<td>Tri-session DNAT</td>
<td>Disable</td>
<td>Enable for CP authentication if the controller is not the gateway and there is no IP on the CP user VLAN.</td>
<td>Without this feature enabled, CP does not work if there is no IP on the CP user VLAN on the controller.</td>
</tr>
<tr>
<td>Whitelist</td>
<td>N/A</td>
<td>Need to add OCSP and CNA URL to the whitelist if OCSP is enabled or CNA enabled in the users’ browser.</td>
<td>Without whitelist, CP authentication may not work on some clients.</td>
</tr>
<tr>
<td>Bypass Apple Captive Network Assistant</td>
<td>Disable</td>
<td>Enable if iOS devices are running CP authentication.</td>
<td>With this feature enabled, Apple CNA URL is not needed in the whitelist.</td>
</tr>
</tbody>
</table>
Captive Portal with MAC Caching via CPPM

Captive portal authentication with MAC caching is another method of guest authentication where users need to complete captive portal authentication only once. During this time the MAC address of the user is cached on ClearPass. Subsequent client re-associations on the WLAN do not require the user to undergo captive portal authentication again. Instead, the user is subjected to MAC authentication and allowed to get on the network.

During MAC authentication, access to users can be restricted based on the day of the week, a bandwidth limit or the number of unique devices per user.

How long MAC addresses of users are cached is configurable. Posture checks are optional and can be enabled to validate the client devices for anti-virus, anti-spyware, firewall status, etc.

The following flows describe the process of captive portal authentication with MAC caching.

- MAC Caching Authentication Flow on page 52
- Guest MAC Added to Endpoints with Unknown State on page 53
- Guest Finishes Registration with Username and Password on page 54
- Guest Passes Captive Portal Authentication on page 55
- Guest MAC State Changed to Known in Endpoints on page 56
- Captive Portal Authentication Not Required Again on page 56
- MAC Caching Expired on page 56

MAC Caching Authentication Flow

A new guest fails MAC authentication for the first time and is redirected to the Captive Portal login page for Guest Registration.

**Figure 31  MAC Caching Authentication Flow**

![Diagram of MAC Caching Authentication Flow](image)

The flow includes the following steps:

1. When a new user associates to the SSID, the controller forwards the user’s MAC address to ClearPass.
2. Since this is the first time ClearPass has seen this client, the MAC authentication fails and the controller redirects the user to the ClearPass captive portal page.

3. The user completes the self-registration form and submits it.

**Guest MAC Added to Endpoints with Unknown State**

The guest user MAC address is added into the Endpoints Repository with an Unknown state in the ClearPass Guest server. Figure 32 shows how the user’s MAC address is stored on the ClearPass’ Endpoints repository as an unknown endpoint.

**Figure 32  Guest MAC Added to Endpoints with Unknown State**
**Guest Finishes Registration with Username and Password**

The guest user finishes registration, gets a valid username and password, and the user entry is added into the Guest User Repository.

**Figure 33  Guest Finishes Registration with Username and Password**

The flow includes the following steps:

1. When a new user associates to the SSID, the controller forwards the user’s MAC address to ClearPass.
2. Since this is the first time ClearPass has seen this client, the MAC authentication fails and the controller redirects the user to the ClearPass captive portal page.
3. The user downloads the self-registration page.
4. The user completes the self-registration form and submits it to ClearPass.
5. After the user submits its form, ClearPass creates a user account and displays a receipt page to the user that contains login details for the user.
Guest Passes Captive Portal Authentication

The guest user passes captive portal authentication.

Figure 34  Guest Passes Captive Portal Authentication

The flow includes the following steps:

1. When a new user associates to the SSID, the controller forwards the user’s MAC address to ClearPass.
2. Since this is the first time ClearPass has seen this client, the MAC authentication fails and the controller redirects the user to the ClearPass captive portal page.
3. The user downloads the self-registration page.
4. The user completes the self-registration form and submits it to ClearPass.
5. After the form is submitted, a user account is created on ClearPass and a receipt page is displayed to the user that contains login details for the user.
6. The user clicks Login, provides the login details and submits the form.
7. ClearPass responds with a hidden JavaScript message prompting the client device to post the form to the controller at the controller’s URL. This step is transparent to the user and no user interaction is needed here.
8. The client performs a DNS query to resolve the controller’s URL.
9. The client posts the form to the controller (no user interaction needed). The controller then authenticates the client by sending a RADIUS request to ClearPass.
10. If the authentication is successful, the user is assigned a post-authentication user role and then redirected to the initially requested URL. The controller-initiated web login process is now complete.
11. The post-authentication user role should make sure that the guest can reach internet resources but cannot access any internal resources. Bandwidth contracts may also be applied to the post-authentication user-role to limit the bandwidth of guest traffic, especially if the internet uplink has limited bandwidth.
Guest MAC State Changed to Known in Endpoints

After the successful captive portal authentication, the Guest user MAC address status changes to “Known” in the Endpoint Repository of the ClearPass Guest server.

Figure 35  Guest MAC State Changed to Known in Endpoints

Captive Portal Authentication Not Required Again

After a guest user completes captive portal registration, ClearPass creates the user’s account. It is valid for up to 8 hours, by default. Guest disconnects and returns back one hour later. At this time, the user’s MAC address becomes a “Known” endpoint on ClearPass.

When the user disconnects and re-associates on the WLAN, only MAC authentication is performed. There is no need for captive portal authentication again. The user is then granted network access.

Figure 36  Captive Portal Authentication Not Required Again

MAC Caching Expired

If the same guest user returns and re-associates on the WLAN two days later, MAC authentication for that user fails since the cached MAC on ClearPass has expired. In this case, the user needs to undergo captive portal authentication again.

Figure 37  MAC Caching Expired
MAC Authentication

MAC authentication is a Layer 2 authentication method that authenticates devices based on their MAC address. If enabled, MAC authentication always takes place before any other L2 or L3 authentication.

In MAC authentication, the MAC address of the device is used both as the username and as the password.

This type of authentication is not really secure by itself, but by working together with captive portal authentication, adds an extra layer of security and flexibility to authenticate devices.

Use Cases

MAC authentication is typically used in the following scenarios:

1. Captive portal authentication with MAC caching via CPPM: MAC authentication provides flexibility for captive portal users to avoid multiple captive portal logins as long as MAC caching has not expired. Both CP authentication and MAC authentication are enabled per guest SSID.

2. MAC authentication may also individually help in providing security to devices that may not support advanced encryption and authentication (for example, 802.1X).

Delimiters

By default, the controller sends the device’s MAC address to authentication servers without any delimiters. Some authentication servers may support only colon, dash, or OUI-NIC delimiters.

So make sure the MAC syntax in the MAC authentication profile matches the MAC address that the authentication server can recognize.

```
#aaa authentication mac mac-prof
(campus1-A7010) (MAC Authentication Profile "mac-prof") #delimiter ?
colon Send MAC as XX:XX:XX:XX:XX:XX
dash Send MAC as XX-XX-XX-XX-XX-XX
none Send MAC as XXXXXXXXXX (default)
oui-nic Send MAC as XXXXXXX-XXXXX
```
MAC Authentication Configuration Flow

The following illustration (Figure 38) shows the different elements that make up the MAC authentication configuration on the controller and what the output for each element looks like after configuration.

**Figure 38 MAC Authentication Configuration Flow**
Guest Access with Self Registration

The self-registration feature allows new guests to create their own user accounts via the “Create Account” option on the captive portal login page. Users can then register themselves by providing information such as name, email address, contact number, sponsor name, and so on.

Once a guest completes registration, the login details are typically delivered to the user via email or SMS, which the guest can then use to complete captive portal authentication.

Figure 39  Guest Access with Self Registration

![Guest Access with Self Registration](image)
Self-Registration Workflow

Figure 40, Figure 41, and illustrate the self-registration workflow.

Figure 40  Self-Registration Workflow - Part 1

The self-registration workflow is as follows:

1. The client device associates to the guest SSID.
2. The guest is assigned a “captive portal logon” initial role.
3. The user browses a website and is redirected to the captive portal page.
4. Since the guest user does not have an active guest account, “Create Account” is selected.
5. The guest user completes the registration by filling in the requested information.
6. The guest user receives login credentials immediately via a registration receipt and/or via email/SMS.

7. The guest user then logs in with these credentials, and upon successful authentication is assigned a “guest” role.

**Figure 42  Self-Registration Complete Workflow**
802.1X Authentication

This section is an overview of 802.11x authentication over Wi-Fi. After the weaknesses exposed in Wired Equivalent Privacy (WEP), 802.11i is the preferred authentication methods for enterprises today.

802.11i supersedes the previous security specification, WEP, which had security vulnerabilities. Wi-Fi Protected Access (WPA) had previously been introduced by the Wi-Fi Alliance as an intermediate solution to WEP insecurities. WPA implemented a subset of a draft of 802.11i. The Wi-Fi Alliance refers to their approved, interoperable implementation of the full 802.11i as WPA2, also called Robust Security Network (RSN). 802.11i makes use of the Advanced Encryption Standard (AES) block cipher, whereas WEP and WPA use the RC4 stream cipher.

The initial authentication process is carried out either using a pre-shared key (PSK), or following an EAP exchange through 802.1X (known as EAPoL, which requires the presence of an authentication server). This process ensures that the client station (STA) is authenticated with the access point (AP). After the PSK or 802.1X authentication, a shared secret key is generated, called the Pairwise Master Key (PMK). The PSK is derived from a password that is put through PBKDF2-SHA1 as the cryptographic hash function. In a pre-shared-key network, the PSK is actually the PMK. We are however focusing on the 802.1X authentication method.

If an 802.1X EAP exchange was carried out, the PMK is derived from the EAP parameters provided by the authentication server.

In summary, 802.1x authentication over Wi-Fi entails:

- A framework that allows EAP over Wireless 802.11 or Ethernet 802.3.
- Port-based network access control to authenticate and authorize client.
- Allows choice of authentication methods using EAP, chosen by the client and the Authentication server.
- Encryption keys are generated dynamically.
- Provides mutual authentication between the client and the server.

This section includes the following topics:

- Key Hierarchy on page 63
- General Packets Flow of 802.1X Authentication on page 64
- EAP 4-Way Key Exchange on page 66
- EAP-PEAP (MSCHAPv2) on page 68
- EAP-TLS Authentication on page 70
- EAP Termination on the Controller on page 73
- 802.1X Authentication Configuration Flow on page 74
- 802.1X User Authentication and Machine Authentication on page 75
- 802.1X Configuration Recommendation on page 78
Key Hierarchy

When reviewing the 802.1x authentication over 802.11, it is important to understand the various key types involved in securing the communication between the Wi-Fi client and AP.

Below are various keys and their definitions. The later sections will detail how these various keys are derived or in some cases from where they originate.

**Figure 43  Key Hierarchy**

- **Master Session Key (MSK)** - Key information that is jointly negotiated between the Supplicant and Authentication Server. This key information is transported via a secure channel from Authenticating Server to Authenticator.

- **Pairwise Master Key (PMK)** - PMK is derived from MSK seeding material. PMK is first 256bits (0-255) of MSK. It can be derived from an EAP method or directly from a Preshared Key (PSK).

- **Group Master Key (GMK)** - GMK is randomly created on Authenticator. Refresh it in a configured time interval to reduce the risk of GMK being compromised.

- **Pairwise Transient Key (PTK)** - PTK is a value derived from PMK, Authenticator Nonce (ANonce), Supplicant Nonce (SNonce), Authenticator Address, and Supplicant Address. This is used to encrypt all unicast transmission between client and an AP. PTK consists of five different keys.
**General Packets Flow of 802.1X Authentication**

802.1X packet exchange is familiar on the wired ethernet. In Figure 44 we take a look at the packet flow as it would look when a Wi-Fi client is configured to perform 802.1X over wireless, through an AP or controller.

The wireless client uses a 802.1X supplicant, the AP (or controller, in case of tunnel mode AP) acts as the authenticator and the RADIUS server performs the authentication.

**Figure 44** General Packets Flow 802.1X Authentication

<table>
<thead>
<tr>
<th>Client - 802.1X Supplicant</th>
<th>Controller or AP - Authenticator</th>
<th>Radius server</th>
</tr>
</thead>
<tbody>
<tr>
<td>802.11 association</td>
<td>Access Blocked</td>
<td></td>
</tr>
<tr>
<td>EAPoL-start</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EAP-request / Identity</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EAP-response / identity</td>
<td>Forward user Identity via RADIUS access request</td>
<td></td>
</tr>
<tr>
<td>EAP-request</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EAP-response (credentials)</td>
<td>Forward user credentials via RADIUS access request</td>
<td></td>
</tr>
<tr>
<td>EAP-success</td>
<td>RADIUS access accept (MS-MPPE-Send-Key &amp; MS-MPPE-Recv-key)</td>
<td></td>
</tr>
<tr>
<td>EAP 4-Way Key Exchanges</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**802.11 Association**

In Figure 44, the association phase involves the wireless clients sending a probe request, and receiving a probe response from the AP.

Following that the client sends an association request, and the AP sends an association response back.

Following the association is the EAP exchange; this is the authentication phase. Note here that the authenticator uses an uncontrolled port to allow authentication traffic from the wireless client.

Any non-authentication traffic is blocked from the client in this state.

**EAP Packet Exchange**

The client then sends an Extensible Authentication Protocol over LAN (EAPoL) start packet.

The AP and controller send an EAP request to the client requesting its identity.

The client responds with an EAP response specifying its identity to the AP. The AP and controller acting as authenticator forward the EAP response to the RADIUS server.

The RADIUS server sends an access challenge destined to the client, via the authenticator. This access challenge could be certificate based or username and password based.

\[\text{NOTE}\]

This section talks about a general flow for 802.1x over wireless and not specifically about any authentication method such as EAP-TLS, EAP-PEAP, and so on.

The wireless client responds to the access challenge with its credentials, which is relayed to the RADIUS server via the authenticator.
The RADIUS server verifies the credentials and responds with either an Access Accept or an Access Reject. If credentials mismatch, the RADIUS server sends an Access Reject to the client and the authentication process restarts with the EAP exchange triggered again.

If credentials match, the RADIUS server relays an Access Accept packet to the wireless client, which contains a 'Master Session Key'. The contents of the MSK are as follows:

\[ MSK (Master\ Session\ Key) = MS - MPPE - Send\ Key + MS - MPPE - Recv\ Key + 32\ bytes\ zeros\ (padding) \]

This Master Session Key will be used to derive the Pairwise Master Key (PMK).

At the end of the EAP exchange, the client now has the seeding material to derive the PMK.

The AP and controller would have derived a PMK and a GMK at this stage.

This step popularly known as the EAP-4-way handshake is explained in the following section.
EAP 4-Way Key Exchange

In the previous section we looked at the general EAP exchange. This is a review of the EAPOL 4-way Handshake.

After the client receives the RADIUS access accept, the encryption keys needs to be derived.

Figure 45  EAP 4-Way Key Exchange

Message 1 (EAPOL-Key Message 1)
- The authenticator sends an EAPOL-Key frame containing an Authenticator Nonce (ANonce) to the supplicant.
- With this information, the supplicant has all of the necessary input to generate PTK using the pseudo-random function (PRF).

The wireless client at this stage has derived the PMK using the MSK. The controller and AP have derived the PMK and GMK as well using the MSK.

The AP and controller initiate EAPOL-Key Message 1 that contains an ANonce (in simple terms this serves as the building block for deriving the PTK at the client’s end).

Message 2 (EAPOL-Key Message 2)
- Supplicant sends an EAPOL-Key frame containing Supplicant Nonce (SNonce) to the authenticator.
- Now authenticator has all the inputs to create PTK.
- Supplicant also sent RSN IE capabilities to the authenticator and message integrity check (MIC).
- Authenticator derives the PTK and validates the MIC as well.

Once the client receives EAPOL-Key Message 1, it uses the ANonce and PMK to generate a PTK, and SNonce. The SNonce and MIC are then transported to the authenticator, which is EAPOL-Key Message 2.

SNonce can again be thought as the building material for PTK at the authenticator.

Message 3 (EAPOL-Key Message 3)
- If necessary, the authenticator derives GTK from GMK.
- The authenticator sends EAPOL-Key frame containing ANonce, RSN-IE, and a MIC.
- GTK will be delivered (encrypted with PTK) to supplicant.
Message to supplicant to install temporal keys.

Authenticator derives PTK and a GTK (if required). The encrypted GTK is sent along with a MIC to the client as EAPOL-Key Message 3.

---

**NOTE**

Group Temporal Key (GTK) - GTK is used to encrypt all broadcast and multicast transmission between an AP and multiple client stations. GTK is derived on Authenticator and sent to supplicant during 4-Way Handshake (M3).

---

**Message 4 (EAPOL-Key Message 4)**

- Supplicant sends final EAPOL-Key frame to the authenticator to confirm temporal keys have been installed.

From this point forward the MAC service data unit (MSDU) will be encrypted using PTK or GTK (depending upon unicast or multicast and broadcast frame).
EAP-PEAP (MSCHAPv2)

The previous topic reviewed the general EAP exchange and the EAPOL 4-way handshake. This topic covers the EAP-PEAP (MSCHAPv2) method of 802.1x over wireless.

The authenticated wireless access design based on Protected Extensible Authentication Protocol Microsoft Challenge Handshake Authentication Protocol version 2 (PEAP-MS-CHAPv2) utilizes the user account credentials (user name and password) stored in Active Directory Domain Services to authenticate wireless access clients, instead of using smart cards or user and computer certificates for client authentication.

**Figure 46** Packet Flow of EAP-PEAP Authentication

In reference to the packet flow in Figure 46, the exchange can be broken down into various stages:

- EAPOL Start on page 68
- Active Directory on page 69
- EAPOL on page 69
- Transport Layer Security Tunnel Setup on page 69
- Inner EAP MSCHAPv2 on page 69
- Active Directory on page 69
- Inner EAP MSCHAPv2 on page 70
- AD, Inner EAP MSCHAPv2, and EAPOL 4-way Handshake on page 70

**EAPOL Start**

1. The authenticator sends an EAP-Request for the identity of the connecting supplicant (client device).
2. The supplicant responds to the authenticator with an EAP Identity Response that contains the identity (username) used for authentication. This is referred to as the "Outer Identity."
3. The authenticator forwards the EAP Identity Response with the identity of the user to the authentication server (CPPM).
Active Directory
4. The authentication server performs an Lightweight Directory Access Protocol (LDAP) lookup against its configured Active Directory authentication sources to try to find the user’s name in the directory, along with some basic LDAP attributes, such as sAMAccountName.
5. The LDAP server responds to the authentication server’s LDAP search request with the appropriate answers to the LDAP lookup.

EAPOL
6. The authentication server responds to the supplicant through the authenticator with an EAP-Request message indicating that it would like to initiate EAP-PEAP.
7. The authenticator passes the EAP-Request message to the supplicant.

Transport Layer Security Tunnel Setup
8. The supplicant sends a Transport Layer Security (TLS) "Client Hello" message within an EAP-response message through the authenticator to the authentication server.
9. The authenticator passes the EAP-Response message containing the TLS Client Hello message to the authentication server.
11. The authenticator forwards the TLS handshake messages between the authentication server and the supplicant inside of EAP Request (server) and EAP Response (supplicant) messages.
12. Steps 10 and 11 repeat until the authentication server has transmitted all of its handshake messages. This may take several steps due to having to dismantle the certificates into fragments that fit within the size limits of an EAP message.
14. The authenticator sends this EAP-Response to the authentication server.
15. The authentication server responds to the authenticator with an EAP-Request for the supplicant that contains the message types "Change Cipher Spec" and "Server Finished."
16. The authenticator passes the EAP message to the supplicant.
17. The supplicant sends an EAP-Response for the authentication server to the authenticator.
18. The authenticator sends the EAP-Response to the authentication server.

Inner EAP MSCHAPv2
19. Inside the TLS tunnel, the EAP process starts again with the authentication server sending an EAP Identity Request to the supplicant requesting the client’s identity.
20. The authenticator sends the EAP Identity Request message to the supplicant requesting the client’s identity.
21. The supplicant responds with an EAP Identity Response containing its identity to the authenticator.
22. The authenticator forwards this EAP Identity Response to the authentication server.

Active Directory
23. The authentication server performs an LDAP lookup against its configured Active Directory authentication sources to try to find the user’s name in the directory, along with some basic LDAP attributes, such as sAMAccountName.
24. The LDAP server responds to the LDAP search request with the appropriate answers to the query.
Inner EAP MSCHAPv2

25. The authentication server sends an EAP request to the supplicant containing an MS-CHAPv2 challenge.
26. The authenticator forwards the EAP request to the supplicant.
27. The supplicant responds with an EAP Identity Response containing its identity to the authenticator.
28. The authenticator forwards this EAP Identity Response to the authentication server.

AD, Inner EAP MSCHAPv2, and EAPOL 4-way Handshake

29. The authentication server takes the username and the MSCHAPv2 response from the supplicant and combines it with the MSCHAPv2 challenge and the NetBIOS name of the Active Directory domain and submits this set of information to the Active Directory domain controller for authentication. This is done via NT LAN Manager (NTLM).
30. The Active Directory domain controller lets the authentication server know that the authentication was successful.
31. The authentication server sends an EAP-Request message for the supplicant with an MSCHAPv2 success message and an authenticator response string from the Active Directory Domain Controller to the authenticator.
32. The authenticator passes the EAP-Request with an MSCHAPv2 success message and the authenticator response to the supplicant.
33. The supplicant sends an EAP-Response message for the authentication server with an MSCHAPv2 success message to the authenticator.
34. The authenticator sends the EAP-Response message from the supplicant with the MSCHAPv2 success message to the authentication server.
35. The authentication server sends an EAP-Request message to the authenticator indicating that the Inner EAP method was successful.
36. The authenticator forwards this EAP-Request to the supplicant.
37. The supplicant sends an EAP-Response to the authentication server, acknowledging that the Inner EAP method was successful.
38. The authenticator forwards the EAP-Response from the supplicant to the authentication server.
39. The authentication server sends a RADIUS access-accept message to the authenticator with an EAPOL success message along with the key material.
40. The authenticator sends an EAPOL success message to the supplicant.
41. The authenticator and supplicant complete a four-way handshake to start the flow of encrypted wireless traffic.

EAP-TLS Authentication

When using EAP-TLS with smart cards or TLS with certificates, both the client and the server use certificates to verify their identities to each other.
Figure 47  Packet Flow of EAP-TLS Authentication

The packet flow of EAP-TLS authentication includes the following steps:

1. The wireless client gets associated with the AP.
2. AP does not permit the client to send any data at this point and sends an authentication request.
3. Client’s screen displays a logon screen. The supplicant then responds with an EAP-Response Identity with user-Id back to the authentication server.
4. The RADIUS server responds back to the client with an EAP-TLS Start Packet. The EAP-TLS conversation starts at this point.
5. The peer sends an EAP-Response back to the authentication server, which contains a client_hello handshake message, a cipher that is set for NULL that will remain this value until change_cipher_spec are negotiated, and the TLS version number.
6. The server presents its certificate to the client as well as requests a valid one from the client. The authentication server responds with an EAP-Request packet that contains the following:
   - TLS server_hello
   - handshake message
   - certificate
   - server_key_exchange
   - certificate request
   - server_hello_done
7. Client responds with a EAP-Response message that contains the following:
   - Certificate  Server can validate to verify that it is trusted
   - client_key_exchange
   - certificate_verify - verifies that the server is trusted
   - change_cipher_spec
   - TLS finished
8. After the client authenticates successfully, the EAP server responds with an EAP-Request, which contains the change_cipher_spec and finished handshake message. The finished handshake message contains the authentication response from the server. Upon receiving, the client verifies the hash in order to authenticate the EAP server. A new encryption key is dynamically derived from the master secret during the TLS handshake.

9. At this point the EAP-TLS enabled wireless client can access the wireless network.
EAP Termination on the Controller

The previous section reviewed detailed packet exchange in EAP-PEAP and EAP-TLS.

The HPE-Aruba controller supports an additional mode of operation called the EAP-Termination mode. In summary:

- It removes the requirement for external authentication servers to be 802.1X-capable.
- There are only two RADIUS messages exchanged; it increases the authentication server scalability and drops delay greatly.

To make good use of CPPM’s functions, we do not recommend enabling EAP-termination when CPPM is the server.

This feature has been present in the Aruba controller OS for a long time. We have looked at detailed flows for a couple of EAP methods.

EAP Termination

Definition

The EAP-TLS offload feature consists of a termination of the EAP-TLS tunnel at the controller rather than at the RADIUS server.

The purpose of this feature is to drastically reduce the number of packets that are exchanged between the wireless client and the RADIUS server. The load on the RADIUS server is reduced and the intranet delay with the controller is minimized.

Requirements

For the EAP-TLS offload feature to work and for the controller to terminate the EAP-TLS tunnel, the controller must have a valid server certificate and the root CA that signed the client certificates. The server certificate and the client certificate do not need to be signed by the same CA.

Figure 48  EAP Termination on the Controller
How It Works

1. When the client initiates a connection to the wireless network using EAP-TLS, both the server and client certificates are exchanged.
2. Upon successful certificate verifications and TLS negotiations, the tunnel gets established.
3. The Aruba controller does not currently support certificate revocation list (CRL) verification of the client certificate, so the controller parses the client certificate and checks if the "principal name" field exists in the "Subject alternative name" option.
4. If that field does not exist, the controller picks up the common name (CN) name.
5. The controller sends an "Authorize only" RADIUS request if the server configured in the associated aaa profile is a RADIUS server.
6. The controller does an LDAP search if the configured server is an LDAP server. In both cases, the controller checks if the principal name or CN name exists in the backend database.
7. If the server returns a successful response, the controller declares the EAP-TLS connection successful, moves to the dynamic key exchange phase, and places the user in the configured 802.1X role.

802.1X Authentication Configuration Flow

Figure 49 illustrates the 802.1X authentication configuration flow.
802.1X User Authentication and Machine Authentication

In addition to user authentication, it is possible to require the machine (the user’s laptop or wireless client) to authenticate itself as a machine that is allowed on the network.

This is mainly possible on a Windows OS machine. The Windows computer name can be authenticated against a Windows Active Directory.

About Machine Authentication

When a Windows device boots, it logs onto the network domain using a machine account. Within the domain, the device is authenticated before computer group policies and software settings can be executed; this process is known as machine authentication. Machine authentication ensures that only authorized devices are allowed on the network.

Enabling the Enforce Machine Authentication Option

You can configure 802.1X authentication for both user and machine authentication (for Windows environments only). This strengthens the authentication process further since both the device and user need to be authenticated.

Select the **Enforce Machine Authentication** option to enforce machine authentication before user authentication.

When selected, either the Machine Authentication Default Role or the User Authentication Default Role is assigned to the user, depending on which authentication is successful. This option is disabled by default.

The example CLI snippets below show the machine authentication options available within an aaa profile.

Machine Authentication Example

The following roles can be assigned in the machine authentication example:

- Machine authentication default machine role – assigned when only user passes machine authentication, but not dot1x user authentication.
- Machine authentication default user role – assigned when user passes dot1x user authentication, but not machine authentication.

```
#show aaa authentication dot1x dot1x-prof

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max authentication failures</td>
<td>0</td>
</tr>
<tr>
<td>Enforce Machine Authentication</td>
<td>Enabled</td>
</tr>
<tr>
<td>Machine Authentication: Default Machine Role</td>
<td>machine-role</td>
</tr>
<tr>
<td>Machine Authentication Cache Timeout</td>
<td>24 hr(s)</td>
</tr>
<tr>
<td>Blacklist on Machine Authentication Failure</td>
<td>Disabled</td>
</tr>
<tr>
<td>Machine Authentication: Default User Role</td>
<td>machine-user-role</td>
</tr>
</tbody>
</table>
```
802.1X Authentication Example

The 802.1X authentication default role can be assigned when:

- Both 802.1X user authentication and machine authentication are enabled, and the user passes both authentications.
- Only 802.1X user authentication is enabled, and the user passes the authentication.

#show aaa profile default

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial role</td>
<td>logon</td>
</tr>
<tr>
<td>MAC Authentication Profile</td>
<td>N/A</td>
</tr>
<tr>
<td>MAC Authentication Default Role</td>
<td>guest</td>
</tr>
<tr>
<td>MAC Authentication Server Group</td>
<td>default</td>
</tr>
<tr>
<td>802.1X Authentication Profile</td>
<td>dot1x-prof</td>
</tr>
<tr>
<td>802.1X Authentication Default Role</td>
<td>allow-all</td>
</tr>
<tr>
<td>802.1X Authentication Server Group</td>
<td>cppm</td>
</tr>
</tbody>
</table>

Machine Authentication Cache - Local Controller

Each machine entry is cached in the Authentication Manager process on the local controller.

#show dot1x machine-auth-cache

Machine Auth Cache Table

<table>
<thead>
<tr>
<th>MAC</th>
<th>Expiry</th>
</tr>
</thead>
<tbody>
<tr>
<td>9c:4e:36:9a:3c:5c</td>
<td>May 23 15:05:18</td>
</tr>
</tbody>
</table>

Machine Auth Cached Entries: 1

Local UserDB Cache - Master Controller

Each machine entry is also cached in the Local_UserDB of the Master controller by default, and it is queried when local cache query fails.

#show local-userdb | include cache

<table>
<thead>
<tr>
<th>Name</th>
<th>Password</th>
<th>Role</th>
<th>E-Mail</th>
<th>Expiry</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>maccache</em> 9c:4e:36:9a:3c:5c</td>
<td>********</td>
<td>mach_role</td>
<td>5/22/2016 10:54</td>
<td></td>
</tr>
</tbody>
</table>

Local UserDB Cache - Local Controller

If use-local-switch option for ‘internal’ server is configured, each machine entry is cached in the Local_UserDB of the local controller instead of the master.

(local-1)(config)#aaa authentication-server internal use-local-switch
Basic Workflow of Machine Authentication

1. A laptop normally boots and the user has to wait for at least 30 seconds before logging in. This delay allows the laptop enough time to search the wireless network and perform machine authentication.

2. The laptop attempts the 802.1X authentication with client machine certificate (for EAP-TLS) or uses A/D computer account and SID as password (for PEAP MSCHAPv2).

3. If the 802.1X authentication is successful, the controller keeps the client MAC address in the local-userdb as cached evidence that a good machine authentication has occurred. This client MAC address is kept for a certain amount of time, which is based on the "caching period". Also, if there is already a record, the lifetime is extended. (Step 6 and 7 explain why the cache is kept instead of looking into the user-table.)

4. After some random time, the user logs into the laptop. User login usually triggers Windows WZC to switch user-id and attempt another 802.1X authentication while it is using user account this time. This is known as user authentication.

5. The Aruba controller sees a successful user authentication, with "enforce machine authentication", and it also queries the local-userdb for the machine authentication history. If a record is cached, this client device has done "mach-user". Otherwise, it is only a user authentication.

6. After the user has logged in, Windows never attempts another machine authentication. When the user logs out, Windows can attempt it. For WPA-TKIP, a full 802.1X user authentication is attempted only on every roaming among the AP.

7. Similarly, if the user's laptop has gone into sleep mode with user logged in, Windows does not attempt another machine authentication. If the laptop has been in sleep mode for 1 hour or so, the user-table normally clears the user record. When the user begins to use the laptop again, only a user authentication is attempted (because the user has not logged out). The user authentication relies on the cache of client MAC addresses in local-userdb.

Figure 50 shows a flowchart of how user authentication and machine authentication are conducted when both are enabled.

Figure 50 Role Derivation Flowchart in 802.1X Authentication
802.1X Configuration Recommendation

Table 7 lists our configuration recommendation for each listed feature.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Default</th>
<th>Recommendation</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Opportunistic Key Caching (OKC)</td>
<td>Enable</td>
<td>Enable</td>
<td>Avoids full dot1x key exchange during roaming by only doing 4 key exchange. NOTE: Mac OS and iOS devices do not support OKC.</td>
</tr>
<tr>
<td>Validate pairwise master key identification (PMKID)</td>
<td>Enable</td>
<td>Enable</td>
<td>Validate if association packet has PMKID, if not, the AP will initiate full 802.1X authentication instead of 4 key exchange. It helps to prevent non-OKC compatible clients from getting stuck at 4 key exchange process.</td>
</tr>
<tr>
<td>Termination</td>
<td>Disable</td>
<td>Enable only when it is needed and CPPM is not the server.</td>
<td>To make good use of CPPM's powerful access control policies, no eap-termination.</td>
</tr>
<tr>
<td>TLS Guest Access</td>
<td>Disable</td>
<td>Enable only when eap-termination is enabled.</td>
<td>It skips certificate common name lookup. When eap-termination is enabled, the controller cannot do certificate common name checkup.</td>
</tr>
<tr>
<td>CA Certificate and Server Certificate</td>
<td>N/A</td>
<td>Replace the controller default certificates with the customer’s own certificates when eap-termination is enabled.</td>
<td>The controller’s certificate should be used for demonstration, only.</td>
</tr>
</tbody>
</table>
VPN Authentication

The virtual intranet access (VIA) provides:

- Secure corporate access via IPsec or secure sockets layer (SSL) to employee laptops and smartphones from anywhere, for example, mobile hotspots.
- Ease-of-use for the end users and network administrators.
- Zero-touch end-user experience, automatically configures WLAN settings on client devices.
- VIA client is available for Windows computers, Apple Mac OS X, Android, and Apple iOS devices. It is not device specific but OS specific, that is, VIA can run on Amazon tablets running Android but not Amazon Kindle Fire.

This section includes the following topics:

- VIA Bootstrap on page 79
- Recommended Deployment on page 81
- License Requirements on page 81
- Ports Need to be Opened on page 81
- Operating Systems Supported by Latest VIA on page 82
- Controller Configuration for VIA on page 82

VIA Bootstrap

The VIA client must be installed on the user device, then the VIA bootstrap process occurs.
Figure 51  VIA Bootstrap Process

1. Prompts for controller IP or FQDN and user credentials
2. Retrieves VIA web auth list, allows user to select VIA auth profile for configuration download
3. Makes an HTTPS POST request to controller to authenticate the user
4. Is Auth successful?
   - Yes: Download the VIA configuration which is tied to the user role assigned in auth process
   - No: Certificate provisioned in the configuration?
      - Yes: Check the CA cert
      - No: Try to establish IPsec connection, succeed?
         - Yes: Is SSL fallback enabled?
             - Yes: If auto upgrade is enabled and new image is available, the VIA client downloads the new image and upgrade
             - No: Fail
         - No: Fail

**Recommended Deployment**

*Figure 52* illustrates the recommended VIA deployment.

*Figure 52  Recommended VIA Deployment*

**License Requirements**

VIA has the following license requirements:

- Only the PEFV license is required for VIA termination on the controller.
- Without the PEFV license installed, there is no VIA configuration available.
- The PEFNG license is not required.

**Ports Need to be Opened**

VIA requires the following ports to be opened:

- TCP 443 - required for initial setup or SSL fallback.
- UDP 4500 – used for IPsec connection.
Operating Systems Supported by Latest VIA

Table 8 lists the operating systems supported by the latest VIA release. This is an ongoing list; a newer version of VIA will support more OS releases.

Table 8: Operating Systems Supported by Latest VIA

<table>
<thead>
<tr>
<th>Operating System</th>
<th>OS Release</th>
</tr>
</thead>
<tbody>
<tr>
<td>Windows OS</td>
<td>Windows (32/64 bit) Vista, Windows 7, Windows 8, Windows 10</td>
</tr>
<tr>
<td>Apple MAC OS X</td>
<td>MAC OS 10.7, 10.8, 10.9, 10.10</td>
</tr>
<tr>
<td>Apple iOS</td>
<td>iOS 4.x, 5.x, 6.x, 7.x, 8.0, 8.1, 9, 9.1</td>
</tr>
<tr>
<td>Android</td>
<td>Android 4.x, 5.0</td>
</tr>
<tr>
<td>Linux</td>
<td>Linux (32/64 bit) Ubuntu 12.04, 14.4, RHEL 6, CentOS 6</td>
</tr>
</tbody>
</table>

VPN Server – The Controller Configuration for VIA

Figure 53 illustrates the VPN server - controller configuration for VIA.

Figure 53  Controller Configuration for VIA
VPN Server Configuration

VIA clients connect to the controller through the public Internet. This communication between VIA clients and the controller across the public Internet is secured using the VPN technology. In the VIA solution, the controllers act as the VPN servers and the VIA clients that are installed on the end-user devices behave as the VPN clients. Secure communication between the controller and VIA clients is achieved using IPsec. As described earlier, the authentication mechanism and IKE versions used for creating the IPsec tunnel varies depending on the VIA version.

- By default, IKEv1 VIA v1 uses XAUTH with IPsec tunnel mode to establish secure VPN connections to the controller.
- VIA v1 for Mac OS uses the built-in IPsec stack to establish IPsec connection. The IPsec stack in Mac OS does not use XAUTH. Instead, it uses Point-to-Point Protocol (PPP) authentication within an L2TP tunnel to authenticate the users.
- IKEv1 phase 1 can use PSK or a certificate.
- The latest version (v2 or v3) of the VIA client uses IKE v2 to create an IPSEC tunnel to the controller.

Following are configuration examples:

IKEv1-PSK for non MAC OS VIA 1.0 VPN clients:
crypto-local isakmp xauth crypto isakmp groupname "changeme"
crypto isakmp key ***** address "0.0.0.0" netmask "0.0.0.0"
vpdn group 12tp
  enable
  client configuration dns 8.8.8.8
ip local pool "via-pool" 172.16.100.4 172.16.100.250

IKEv1-Cert (MAC OS VIA 1.0 VPN client does not support it):
crypto-local isakmp xauth crypto isakmp groupname "changeme"
crypto isakmp key ***** address "0.0.0.0" netmask "0.0.0.0"
vpdn group 12tp
  enable
  client configuration dns 8.8.8.8
ip local pool "via-pool" 172.16.100.4 172.16.100.250

ikev1-Cert
no crypto-local isakmp xauth crypto isakmp groupname "changeme"
crypto isakmp key ***** address "0.0.0.0" netmask "0.0.0.0"
vpdn group 12tp
  enable
  client configuration dns 8.8.8.8
ip local pool "via-pool" 172.16.100.4 172.16.100.250

IKEv2-Cert
No crypto-local isakmp xauth crypto isakmp groupname "changeme"
crypto isakmp key ***** address "0.0.0.0" netmask "0.0.0.0"
vpdn group 12tp
  enable
  client configuration dns 8.8.8.8
ip local pool "via-pool" 172.16.100.4 172.16.100.250
The section includes the following topics:

- Configuring the VPN Server for IKEv1 on page 84
- Configuring VPN Server for IKEv1-PSK on page 85
- L2TP and XAUTH Parameters on page 85
- Address Pools on page 85
- IKE Aggressive Group Name on page 86
- IKE Shared Secret on page 86
- Configuring VPN Server for IKEv1 Certificates on page 86
- IKE Server Certificate on page 86
- CA Certificate Assigned for VPN-Clients on page 87
- Certificate Groups for VPN-Clients on page 87
- IKEv1 Phase 2 Authentication on page 87
- Configuring VPN Server for IKEv2 on page 88
- L2TP and XAUTH Parameters on page 89
- IKE Server Certificate on page 89
- CA Certificate Assigned for VPN-Clients on page 89
- Check Certificate Common Name Against AAA Server on page 89
- IKEv2 EAP Authentication on page 89
- IKE Policies and IPsec Maps on page 90

**Configuring the VPN Server for IKEv1**

IKEv1 protocol defines a two-phase method for providing Internet security. Phase 1 involves the creation of an ISAKMP tunnel and phase 2 involves the creation of a secure IPsec tunnel. The IPsec tunnel created in phase 2 is used to secure user data. The initial ISAKMP tunnel ensures that the negotiations for the establishing the IPsec tunnel happen within a secure channel. IKEv1 for VIA has two authentication phases. Phase 1 authentication of IKEv1 can be implemented using PSK or X.509 certificates. The phase 2 authentication, which is implemented using XAUTH, requires a username and password. So, the VPN server configuration for IKEv1-PSK varies from that of IKEv1-Certs.
Configuring VPN Server for IKEv1-PSK

At a minimum, these parameters should be configured in the VPN server of the controller for VIA deployments using IKEv1 with PSK:

- L2TP and XAUTH parameters
- address pools
- IKE aggressive group name
- IKE shared secret

L2TP and XAUTH Parameters

The following L2TP and XAUTH parameters should be configured for IKEv1 VIA deployments:

- **Enable XAUTH** - By default, IKEv1 VIA deployments use XAUTH with IPsec tunnel mode to establish secure VPN connections to the controller. So, the XAUTH knob under the L2TP and XAUTH parameters settings should be enabled for IKEv1 VIA deployments.

- **Configure DNS information** - The DNS server options under the L2TP and XAUTH parameters settings must also be configured, with the appropriate corporate DNS servers, for use by VIA clients that connect to the controller. Without the DNS server information, VIA cannot resolve the DNS queries for tunneled networks.

- **Enable L2TP** - VIA for Mac OS uses the built-in IPsec stack of the Mac OS for establishing IPsec connection. The IPsec stack in Mac OS does not use XAUTH. Instead, it uses PPP authentication within an L2TP tunnel to authenticate the users. The L2TP tunnel is also used for exchange of IP information related to the IPsec tunnel. For deployments that support VIA for Mac OS, the L2TP parameter should be enabled. Remember that the L2TP tunnel is built within the secure IPsec tunnel, so all the exchanges are secure.

- **Authentication protocols** - This parameter defines the PPP authentication protocol that should be used to authenticate the credentials presented by the Mac OS VIA users. The various options available are PAP, EAP, CHAP, MSCHAP, and MSCHAPv2. For deployments that support VIA for Mac OS, select an authentication method that suits your network policy. Aruba recommends that you choose a strong authentication method, such as MSCHAPv2, rather than PAP.

Address Pools

Every VPN client (RAPs, third-party VPN clients, and VIA) that successfully authenticates to the VPN server module of the controller is given a valid inner IP address and DNS server information. This inner IP address is issued from the address pool that is configured in the VPN server. More than one pool can be configured and there is no need to assign more addresses in the pool than the number of VPN clients that terminate on that controller. DHCP services are not required for the subnets used in the VPN address pool. However, it is necessary to define a VLAN for the subnet used in the VPN address pool and ensure that this VLAN is routable from the corporate network. If only a single pool is configured, all the VPN clients that terminate on that controller are issued an inner IP address from the same pool. When multiple address pools are configured, the controller can be configured to use distinct VPN pools for RAPs, VIA, and third-party VPN clients. This configuration can be achieved by appending a VPN pool to the role assigned to the RAPs, VIA, and third-party VPN clients.

When distinct VPN pools are not defined, the controller automatically uses the first pool in the VPN address pool. When this pool expires, the next pool in the list is used and so on. Remember that if the VPN address pool is exhausted, new VIA clients cannot establish the IPsec tunnel until the required number of IP addresses are added to the pool.
It is essential that the addresses used in the VPN address pool for VIA are routable from the internal corporate network. If not, the VIA clients cannot connect to the corporate resources and vice-versa. Alternatively, you can implement Network Address Translation (NAT) on the VLAN used for the VPN address pool. Remember that NAT might cause issues with certain applications such as file transfer protocol (FTP).

Like the VLAN and IP parameters, the VPN address pools are not synchronized from the active controller to the backup controller during database synchronization. Create VPN address pools individually on the active and standby master controllers. The VPN pools used on the active and the backup controller are not required to be the same.

**IKE Aggressive Group Name**

The IKE aggressive group name is a feature used by certain legacy VPN clients that require an aggressive mode group name. This parameter is not used by VIA. However, this field cannot be empty and requires a value. The default value is “changeme”.

**IKE Shared Secret**

For VIA deployments that use IKEv1 with PSK, a part of the IPsec process requires the VPN client to present a shared secret. Aruba allows you to configure keys that are specific to a subnet or you can specify a global key. To make the IKE key global, specify 0.0.0.0 for the subnet and subnet mask length fields. Remember, for VIA deployments using IKEv1-PSK, the IKE shared secret should be configured for the IPsec tunnel to be established. From a security perspective, it is very important to make sure that the IKE pre-shared key is long and complex. Aruba recommends no fewer than 16 characters.

**Configuring VPN Server for IKEv1 Certificates**

At a minimum, these parameters should be configured in the VPN server of the controller for VIA deployments using IKEv1 with certificates:

- L2TP and XAUTH parameters
- Address pools
- IKE aggressive group name
- IKE server certificate
- CA certificate assigned for VPN-clients
- Certificate groups for VPN-clients (optional)

VIA 1.0 for Mac OS does not support IKEv1 certificates.

**IKE Server Certificate**

For VIA deployments that use IKEv1 with certificate, the VPN server on the controller and the VIA client present a certificate to each other as a part of phase 1 authentication of IKEv1. The certificate that should be presented by the VPN server module to the VIA client should be selected as the IKE server certificate.
CA Certificate Assigned for VPN-Clients

For clients that use certificates, the certificate presented during phase 1 authentication of IKEv1 is considered valid only if it is signed by a trusted CA. The CA certificate of the trusted CAs that signed the client certificates must be added to the CA Certificate Assigned for VPN-Clients parameter list. Client authentication fails if the presented client certificate is not signed by the CAs in the CA Certificate Assigned for VPN-Clients parameter list. Aruba controller can be configured as an Online Certificate Status Protocol (OCSP) client to validate the revocation state of the certificates presented by the clients. Support for OCSP requires ArubaOS version 6.1 or later.

Certificate Groups for VPN-Clients

Introduced in ArubaOS 6.1, the certificate groups for VPN-clients parameter allows the use of unique server certificates for different clients. This new parameter enables the pairing of IKE server certificates with trusted CA certificates. The controller uses this list to present the appropriate IKE server certificate to the client. The server certificate presented to the clients depends on the CA cert used to sign the client certificate. With this feature, VPN clients using RSA certificates and Suite B clients using Elliptic Curve Digital Signature Algorithm (ECDSA) certificates can be terminated on the same controller.

In ArubaOS 6.0 and earlier, only a single certificate can be used as IKE server certificate.

IKEv1 Phase 2 Authentication

After the IKEv1 phase 1 is complete using PSK or certificates, a secure ISAKMP tunnel (also known as ISAKMP SA) is formed. When phase 1 is complete, the phase 2 negotiations take place and a secure IPsec tunnel (also known as IPSec SA) is formed. This IPsec tunnel is used to secure the user data. As per the IKEv1 standard, after the initial phase 1 authentication, no additional authentication is needed to complete the phase 2. IKEv1 authenticates the IPsec devices or VPN clients but does not include any mechanism to authenticate the remote VPN user. However, if desired, the XAUTH mechanism can be used to force a VPN user to authenticate using a username and password or token cards (two-factor authentication) to a VPN gateway before the IKEv1 phase 2. This authentication provides an additional layer of security. XAUTH is not a part of the IKEv1 standard, but it is rather an extension to IKEv1 phase 1. XAUTH takes place after the successful completion of phase 1, and IKEv1 phase 2 negotiations occur only after the successful completion of XAUTH. By default, VIA uses XAUTH for IKEv1, which requires the VIA user to present valid credentials to establish a secure connection to the corporate resources. The credentials provided by the user during XAUTH are validated against the specified authentication server. Either the internal database or any other authentication server type available on ArubaOS can be used as the authentication server. If an external RADIUS server is used to authenticate IKEv1 VIA users, then it must support PAP authentication. For more information on authentication server requirements and configuring an authentication server, see Configuring a VIA Server Group for Authenticating VIA Users on page 36 in Chapter 6: Configuring VIA Profiles.
Configuring VPN Server for IKEv2

Like IKEv1, IKEv2 also forms two tunnels or SAs to secure the sensitive data. However, IKEv2 is lighter and much faster than IKEv1. IKEv1 is complex and takes up to nine messages to establish a secure IPsec tunnel, but IKEv2 requires just four messages to establish the IPsec tunnel. As a result, IKEv2 significantly reduces the bandwidth requirements. IKEv2 is also more resilient to DOS attacks than IKEv1. IKEv2 also supports EAP authentication and does not require the use of XAUTH. IKEv2 has enhancements such as liveness checks, which make it more reliable than IKEv1.

IKEv2 does not have two phases of authentication, only a single phase. The IKEv2 authentication methods that are supported for VIA clients on ArubaOS are these:

- User authentication with X.509 certificates
  - The VIA client authenticates the controller certificate.
  - The controller authenticates the user certificate. No EAP methods are involved.
- User authentication with EAP-TLS
  - The VIA client authenticates the controller certificate.
  - The controller authenticates the user certificate using EAP-TLS over IKEv2. The controller just acts as an EAP pass-through to an external EAP-compliant server. EAP termination on the controller is not supported for VIA clients.
- User authentication with EAP-PEAP
  - The VIA client authenticates the controller certificate.
  - The controller validates the user credentials (username and password) with an external server. The controller just acts as an EAP pass-through to an external EAP-compliant server. EAP termination is not supported for VIA clients, so the internal database of the controller cannot be used to validate user credentials.

EAP-TLS and EAP-MSCHAPv2 are supported for IKEv2. However, EAP termination and other EAP types are not supported for IKEv2.

**NOTE**

ArubaOS does not support the use of IKEv2 with PSK for VIA. However, site to site IKEv2 VPN links can be configured to use PSK.

At a minimum, these parameters should be configured in the VPN server of the controller for VIA deployments using IKEv2:

- L2TP and XAUTH parameters
- Address pools
- IKE aggressive group name
- IKE server certificate
- CA certificate assigned for VPN clients. (Required only for IKEv2 authentication with X.509 certificates and not for EAP authentications.)
- Certificate groups for VPN clients. (Optional for IKEv2 authentication with X.509 certificates and is not required for EAP authentications.)
L2TP and XAUTH Parameters
As described earlier, IKEv2 does not use XAUTH, so the XAUTH parameter need not be enabled for IKEv2 VIA deployments. However, the L2TP and XAUTH parameters setting that must be configured for IKEv2 VIA deployments is this one:

- **Configure DNS information** - The DNS server options under the L2TP and XAUTH parameters settings must be configured, with the appropriate corporate DNS servers, for use by VIA clients that connect to the controller. Without the DNS server information, VIA cannot resolve the DNS queries for tunneled networks. Remember that the intranet hostnames cannot be resolved if you use a public DNS server in this field.

IKE Server Certificate
IKEv2 supports asymmetric authentication, which means that both peers do not have to use the same authentication method. For instance, one peer can use certificates and the other can use EAP-MSCHAPv2. For VIA deployments that use IKEv2, the VPN server on the controller always uses a certificate for IKEv2 authentication phase. However, the clients can use certificates, EAP-MSCHAPv2, or EAP-TLS. The certificate that should be presented by the VPN server module to the VIA client should be selected as the IKE server certificate.

CA Certificate Assigned for VPN-Clients
For clients that use certificates, the certificate that is presented during the IKEv2 authentication phase is considered valid only if it is signed by a trusted CA. The CA certificate of the trusted CAs that signed the client certificates must be added to the CA Certificate Assigned for VPN-Clients parameter list. Client authentication fails if the presented client certificate is not signed by the CAs in the CA Certificate Assigned for VPN-Clients parameter list. Aruba controller can be configured as an OCSP client to validate the revocation state of the certificates presented by the clients. Support for OCSP requires ArubaOS 6.1 or later. To configure the Aruba controller as OCSP client, see the Aruba 6.1 User Guide available at the Aruba support site.

Check Certificate Common Name Against AAA Server
In IKEv2 VIA deployments using certificates, the user certificate presented by the VIA clients can be further scrutinized by validating the certificate common name (CN) against an authentication server. This can be achieved by enabling the “check certificate common name against AAA server” parameter available in the default VPN authentication profile. If this option is enabled, the CN that is present in the client certificate is authorized against the specified server. The controller captures and sends the certificate CN name as an authorization string to the specified authentication server. If the authentication server authorizes the CN, the client is authenticated by the controller. These criteria must be satisfied to pass authentication when the “check certificate common name against AAA server” parameter is enabled:

- The client certificate must be signed by a trusted CA.
- The client certificate CN should be authorized by the authentication server.

If the “check certificate common name against AAA server” option is disabled, client authentication is only based on whether the client certificate is signed by a trusted CA or not. Either the internal database on the controller or an external authentication server can be used for authorizing the CN. If the internal database is used, add all certificate CNs to the internal database of the controller on which the VIA clients terminate. When you add the user name to the internal database, you must add a password for each user. Add a dummy password because this password does not influence the authorization of CN by the internal database.

IKEv2 EAP Authentication
For IKEv2 EAP-TLS and EAP-PEAP supported by VIA, an EAP-compatible external authentication server is needed to authenticate the credentials provided by the user during the IKEv2 process.
Ensure that your authentication server supports authorization services using only the username because not all authentication servers support this feature. ClearPass has support for authorizing based on just the username. If a RADIUS server is used for authorization, the controller will send the certificate CN as a RADIUS “authorize only” attribute using PAP. So, a RADIUS server used for the certificate CN authorization should support the RADIUS “authorize-only” attribute. An LDAP server can also be used for authorization.

**IKE Policies and IPsec Maps**

The ArubaOS has a predefined list of IKE and IPsec policies (also known as IPsec maps) for different IKE versions. Based on the proposal of the VPN client, the controller dynamically chooses the most appropriate IKE and IPsec policy. Aruba recommends the use of the predefined IKE and IPsec policies for establishing secure IPsec connection to the VPN clients. In addition to the pre-defined policies, custom IKE and IPsec policies can be created on the ArubaOS. To create a custom IKE and IPsec policy you have to define a number of variables such as the IKE version, encryption type, hashing algorithm, life time, and Diffie-Hellman group. Aruba recommends that you have a good understanding of these variables and their implication before you create custom policies. For information on creating custom IKE and IPsec policies for VPN clients, see the Aruba 6.1 User Guide available at the Aruba support site.
Figure 1  Aruba Reference Architectures ......................................................... 7
Figure 2  Campus Deployment with Redundancy .................................................. 8
Figure 3  Aruba Campus Logical Architecture .................................................... 9
Figure 4  User Roles ......................................................................................... 11
Figure 5  Applying a Role .................................................................................. 11
Figure 6  User Role Derived ............................................................................. 12
Figure 7  User and Server Role Derivation ......................................................... 13
Figure 8  Aruba VSAs ...................................................................................... 17
Figure 9  Summary - Order in which Derived Rules Get Applied ......................... 20
Figure 10 Scanner Gun ..................................................................................... 21
Figure 11 Server Derivation Rules ..................................................................... 21
Figure 12 Wired Security .................................................................................. 22
Figure 13 General Role Derivation Flow ............................................................ 23
Figure 14 Layer 2 Role Priority ......................................................................... 24
Figure 15 Initial Role and UDR ......................................................................... 28
Figure 16 MAC Authentication ........................................................................ 28
Figure 17 PMK Caching and OKC Scenario - in the case of roaming ................ 29
Figure 18 802.1X Authentication ..................................................................... 29
Figure 19 Captive Portal Authentication Flow Chart .......................................... 38
Figure 20 Packet Flow Process ......................................................................... 39
Figure 21 Packet Capture 1 ............................................................................... 41
Figure 22 Packet Capture 2 ............................................................................... 41
Figure 23 Packet Capture 2 - Details ................................................................. 42
Figure 24 Packet Capture 3 ............................................................................... 42
Figure 25 Packet Capture 4 ............................................................................... 43
Figure 26 Packet Capture 5 ............................................................................... 43
Figure 27 CPPM Server Initiated Web Login Packets Flow ............................... 44
Figure 28 CPPM Controller Initiated Web Login Packets Flow ....................... 46
Figure 29 Captive Portal Configuration Flow ..................................................... 47
Figure 30 Tri-Session DNAT ............................................................................. 50
Figure 31 MAC Caching Authentication Flow .................................................... 52
Figure 32 Guest MAC Added to Endpoints with Unknown State .................. 53
Figure 33 Guest Finishes Registration with Username and Password .............. 54
Figure 34 Guest Passes Captive Portal Authentication ..................................... 55
Figure 35 Guest MAC State Changed to Known in Endpoints ......................... 56
Figure 36 Captive Portal Authentication Not Required Again .......................... 56
Figure 37 MAC Caching Expired ................................................................. 56
Figure 38 MAC Authentication Configuration Flow ........................................ 58
Figure 39 Guest Access with Self Registration .............................................. 59
Figure 40 Self-Registration Workflow - Part 1 ............................................... 60
Figure 41 Self-Registration Workflow - Part 2 ............................................... 61
Figure 42 Self-Registration Complete Workflow ............................................ 61
Figure 43 Key Hierarchy .............................................................................. 63
Figure 44 General Packets Flow 802.1X Authentication ................................ 64
Figure 45 EAP 4-Way Key Exchange ............................................................ 66
Figure 46 Packet Flow of EAP-PEAP Authentication ..................................... 68
Figure 47 Packet Flow of EAP-TLS Authentication ....................................... 71
Figure 48 EAP Termination on the Controller .............................................. 73
Figure 49 802.1X Authentication Configuration Flow ................................... 74
Figure 50 Role Derivation Flowchart in 802.1X Authentication ...................... 77
Figure 51 VIA Bootstrap Process ............................................................... 80
Figure 52 Recommended VIA Deployment ................................................... 81
Figure 53 Controller Configuration for VIA .................................................. 82
The following table lists the symbols used in the figures in this document.

**Table 9: Symbols**

<table>
<thead>
<tr>
<th>Description</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wireless Controller</td>
<td><img src="image1.png" alt="Symbol" /></td>
</tr>
<tr>
<td>Access Point</td>
<td><img src="image2.png" alt="Symbol" /></td>
</tr>
<tr>
<td>Layer 2 Switch</td>
<td><img src="image3.png" alt="Symbol" /></td>
</tr>
<tr>
<td>Layer 3 Switch</td>
<td><img src="image4.png" alt="Symbol" /></td>
</tr>
</tbody>
</table>
### Table 9: Symbols

<table>
<thead>
<tr>
<th>Description</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Router</td>
<td>![Router Symbol]</td>
</tr>
<tr>
<td>Servers/PBX</td>
<td>![Servers/PBX Symbol]</td>
</tr>
<tr>
<td>Wired Client - Desktop Computer</td>
<td>![Wired Client Symbol]</td>
</tr>
<tr>
<td>Description</td>
<td>Symbol</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>Wireless Client - Laptop</td>
<td>![Laptop Symbol]</td>
</tr>
<tr>
<td>Wireless Client - Smart Phone</td>
<td>![Smart Phone Symbol]</td>
</tr>
</tbody>
</table>
The following table describes the acronyms used in this guide.

**Table 10: Acronyms**

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>AAA</td>
<td>Authentication, Authorization, and Accounting</td>
</tr>
<tr>
<td>ACK</td>
<td>Acknowledgment</td>
</tr>
<tr>
<td>ACL</td>
<td>Access Control List</td>
</tr>
<tr>
<td>ACR</td>
<td>Advanced Cryptography</td>
</tr>
<tr>
<td>AD</td>
<td>Active Directory</td>
</tr>
<tr>
<td>ADP</td>
<td>Aruba Discovery Protocol</td>
</tr>
<tr>
<td>AES</td>
<td>Advanced Encryption Standard</td>
</tr>
<tr>
<td>ALE</td>
<td>Analytics and Location Engine</td>
</tr>
<tr>
<td>ALG</td>
<td>Application-Level Gateway</td>
</tr>
<tr>
<td>AM</td>
<td>Air Monitor</td>
</tr>
<tr>
<td>A-MPDU</td>
<td>Aggregated-Media Access Control (MAC) Packet Data Unit</td>
</tr>
<tr>
<td>A-MSDU</td>
<td>Aggregated-MAC Service Data Unit</td>
</tr>
<tr>
<td>AP</td>
<td>Access Point</td>
</tr>
<tr>
<td>ARM</td>
<td>Adaptive Radio Management</td>
</tr>
<tr>
<td>ARP</td>
<td>Address Resolution Protocol</td>
</tr>
<tr>
<td>AWO</td>
<td>All Wireless Office</td>
</tr>
<tr>
<td>BCMC</td>
<td>Broadcast and Multicast</td>
</tr>
<tr>
<td>BPDU</td>
<td>Bridge Protocol Data Unit</td>
</tr>
<tr>
<td>BSS</td>
<td>Basic Service Set</td>
</tr>
<tr>
<td>BSSID</td>
<td>Basic Service Set Identifier</td>
</tr>
<tr>
<td>BTM</td>
<td>BSS Transition Management</td>
</tr>
<tr>
<td>BW</td>
<td>Bandwidth</td>
</tr>
<tr>
<td>BYOD</td>
<td>Bring Your Own Device</td>
</tr>
<tr>
<td>CAP</td>
<td>Campus Access Point</td>
</tr>
</tbody>
</table>
### Table 10: Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>CN</td>
<td>Common Name</td>
</tr>
<tr>
<td>CNA</td>
<td>Captive Network Assistant</td>
</tr>
<tr>
<td>CoA</td>
<td>Change of Authorization</td>
</tr>
<tr>
<td>CP</td>
<td>Captive Portal</td>
</tr>
<tr>
<td>CPPM</td>
<td>ClearPass Policy Manager</td>
</tr>
<tr>
<td>CPSec</td>
<td>Control Plane Security</td>
</tr>
<tr>
<td>CPU</td>
<td>Central Processing Unit</td>
</tr>
<tr>
<td>CRL</td>
<td>Certificate Revocation List</td>
</tr>
<tr>
<td>DAD</td>
<td>Duplicate Address Detection</td>
</tr>
<tr>
<td>DFS</td>
<td>Dynamic Frequency Selection</td>
</tr>
<tr>
<td>DHCP</td>
<td>Dynamic Host Configuration Protocol</td>
</tr>
<tr>
<td>DLNA</td>
<td>Digital Living Network Alliance</td>
</tr>
<tr>
<td>DMO</td>
<td>Dynamic Multicast Optimization</td>
</tr>
<tr>
<td>DMZ</td>
<td>Demilitarized Zone</td>
</tr>
<tr>
<td>DNS</td>
<td>Domain Name System</td>
</tr>
<tr>
<td>DRM</td>
<td>Digital Rights Management</td>
</tr>
<tr>
<td>DST-NAT</td>
<td>Destination Network Address Translation</td>
</tr>
<tr>
<td>D-Tunnel</td>
<td>Decrypt-Tunnel</td>
</tr>
<tr>
<td>EAP</td>
<td>Extensible Authentication Protocol</td>
</tr>
<tr>
<td>EAPoL</td>
<td>Extensible Authentication Protocol over LAN</td>
</tr>
<tr>
<td>ECDSA</td>
<td>Elliptic Curve Digital Signature Algorithm</td>
</tr>
<tr>
<td>EIRP</td>
<td>Effective Isotropic Radiated Power</td>
</tr>
<tr>
<td>ESS</td>
<td>Extended Service Set</td>
</tr>
<tr>
<td>EVDO</td>
<td>Evolution Data Optimized</td>
</tr>
<tr>
<td>FFT</td>
<td>Fast Fourier Transform</td>
</tr>
<tr>
<td>FQDN</td>
<td>Fully Qualified Domain Name</td>
</tr>
<tr>
<td>FQLN</td>
<td>Fully Qualified Location Name</td>
</tr>
<tr>
<td>FT</td>
<td>Fast Transition</td>
</tr>
<tr>
<td>Acronym</td>
<td>Definition</td>
</tr>
<tr>
<td>---------</td>
<td>------------</td>
</tr>
<tr>
<td>FTP</td>
<td>File Transfer Protocol</td>
</tr>
<tr>
<td>GARP</td>
<td>Gratuitous ARP</td>
</tr>
<tr>
<td>GMK</td>
<td>Group Master Key</td>
</tr>
<tr>
<td>GRE</td>
<td>Generic Routing Encapsulation</td>
</tr>
<tr>
<td>HA</td>
<td>High Availability</td>
</tr>
<tr>
<td>HD</td>
<td>High Density</td>
</tr>
<tr>
<td>IAP</td>
<td>Instant Access Point</td>
</tr>
<tr>
<td>ICCID</td>
<td>Integrated Circuit Card Identifier</td>
</tr>
<tr>
<td>IDS</td>
<td>Intrusion Detection System</td>
</tr>
<tr>
<td>IE</td>
<td>Information Element</td>
</tr>
<tr>
<td>IGMP</td>
<td>Internet Group Management Protocol</td>
</tr>
<tr>
<td>IKE</td>
<td>Internet Key Exchange</td>
</tr>
<tr>
<td>IMEI</td>
<td>International Mobile Equipment Identity</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>IPS</td>
<td>Intrusion Prevention System</td>
</tr>
<tr>
<td>IPsec</td>
<td>Internet Protocol Security</td>
</tr>
<tr>
<td>LACP</td>
<td>Link Aggregation Control Protocol</td>
</tr>
<tr>
<td>LAN</td>
<td>Local Area Network</td>
</tr>
<tr>
<td>LDAP</td>
<td>Lightweight Directory Access Protocol</td>
</tr>
<tr>
<td>LIC-AP</td>
<td>Access Point License</td>
</tr>
<tr>
<td>LLDP</td>
<td>Link Layer Discovery Protocol</td>
</tr>
<tr>
<td>LMS</td>
<td>Local Management Switch</td>
</tr>
<tr>
<td>LPV</td>
<td>Large Public Venue</td>
</tr>
<tr>
<td>MAC</td>
<td>Media Access Control</td>
</tr>
<tr>
<td>MC</td>
<td>Master Controller</td>
</tr>
<tr>
<td>MDIE</td>
<td>Mobility Domain Information Element</td>
</tr>
<tr>
<td>MDNS</td>
<td>Multicast Domain Name System</td>
</tr>
<tr>
<td>MIC</td>
<td>Message Integrity Check</td>
</tr>
<tr>
<td>Acronym</td>
<td>Definition</td>
</tr>
<tr>
<td>---------</td>
<td>------------</td>
</tr>
<tr>
<td>MPLS</td>
<td>Multiprotocol Label Switching</td>
</tr>
<tr>
<td>MRO</td>
<td>Multicast Rate Optimization</td>
</tr>
<tr>
<td>MSCHAPv2</td>
<td>Microsoft Challenge Handshake Authentication Protocol version 2</td>
</tr>
<tr>
<td>MSDU</td>
<td>MAC Service Data Unit</td>
</tr>
<tr>
<td>MSK</td>
<td>Master Session Key</td>
</tr>
<tr>
<td>MTU</td>
<td>Maximum Transmission Unit</td>
</tr>
<tr>
<td>MU-MIMO</td>
<td>Multi-User Multiple Input, Multiple Output</td>
</tr>
<tr>
<td>NAC</td>
<td>Network Access Control</td>
</tr>
<tr>
<td>NAS</td>
<td>Network Address Server</td>
</tr>
<tr>
<td>NAT</td>
<td>Network Address Translation</td>
</tr>
<tr>
<td>NTLM</td>
<td>NT LAN Manager</td>
</tr>
<tr>
<td>OCSP</td>
<td>Online Certificate Status Protocol</td>
</tr>
<tr>
<td>OKC</td>
<td>Opportunistic Key Caching</td>
</tr>
<tr>
<td>OOS</td>
<td>Out-of-Service</td>
</tr>
<tr>
<td>OUI</td>
<td>Organizationally Unique Identifier</td>
</tr>
<tr>
<td>PAPI</td>
<td>Proprietary Access Protocol Interface</td>
</tr>
<tr>
<td>PAT</td>
<td>Port Address Translation</td>
</tr>
<tr>
<td>PCAP</td>
<td>Packet Capture</td>
</tr>
<tr>
<td>PEAP</td>
<td>Protected Extensible Authentication Protocol</td>
</tr>
<tr>
<td>PEFNG</td>
<td>Policy Enforcement Firewall Next Generation</td>
</tr>
<tr>
<td>PEFV</td>
<td>Policy Enforcement Firewall Virtual Private Network</td>
</tr>
<tr>
<td>PoE</td>
<td>Power over Ethernet</td>
</tr>
<tr>
<td>PMK</td>
<td>Pairwise Master Key</td>
</tr>
<tr>
<td>PMKID</td>
<td>Pairwise Master Key Identification</td>
</tr>
<tr>
<td>PPP</td>
<td>Point-to-Point Protocol</td>
</tr>
<tr>
<td>PRF</td>
<td>Pseudo-Random Function</td>
</tr>
<tr>
<td>PSK</td>
<td>Pre-shared Key</td>
</tr>
<tr>
<td>PTK</td>
<td>Pairwise Transient Key</td>
</tr>
<tr>
<td>Acronym</td>
<td>Definition</td>
</tr>
<tr>
<td>---------</td>
<td>------------</td>
</tr>
<tr>
<td>QoS</td>
<td>Quality of Service</td>
</tr>
<tr>
<td>RA</td>
<td>Router Advertisement</td>
</tr>
<tr>
<td>RADIUS</td>
<td>Remote Authentication Dial-In User Service</td>
</tr>
<tr>
<td>RAP</td>
<td>Remote Access Point</td>
</tr>
<tr>
<td>RF</td>
<td>Radio Frequency</td>
</tr>
<tr>
<td>RFP</td>
<td>RFProtect</td>
</tr>
<tr>
<td>RRM IE</td>
<td>Radio Resource Management Information Element</td>
</tr>
<tr>
<td>RS</td>
<td>Router Solicitation</td>
</tr>
<tr>
<td>RSN</td>
<td>Robust Security Network</td>
</tr>
<tr>
<td>RSSI</td>
<td>Received Signal Strength Indication</td>
</tr>
<tr>
<td>RTSP</td>
<td>Real Time Streaming Protocol</td>
</tr>
<tr>
<td>SCCP</td>
<td>Skinny Client Control Protocol</td>
</tr>
<tr>
<td>SDR</td>
<td>Server Derived Rule</td>
</tr>
<tr>
<td>SIP</td>
<td>Session Initiation Protocol</td>
</tr>
<tr>
<td>SLAAC</td>
<td>Stateless Address Auto Configuration</td>
</tr>
<tr>
<td>SM</td>
<td>Spectrum Monitor</td>
</tr>
<tr>
<td>SMS</td>
<td>Simple Messaging Service</td>
</tr>
<tr>
<td>SNR</td>
<td>Signal to Noise Ratio</td>
</tr>
<tr>
<td>SSDP</td>
<td>Simple Service Discovery Protocol</td>
</tr>
<tr>
<td>SSH</td>
<td>Secure Shell</td>
</tr>
<tr>
<td>SSID</td>
<td>Service Set Identifier</td>
</tr>
<tr>
<td>SSL</td>
<td>Secure Sockets Layer</td>
</tr>
<tr>
<td>SSO</td>
<td>Single Sign-On</td>
</tr>
<tr>
<td>STA</td>
<td>Station</td>
</tr>
<tr>
<td>SVP</td>
<td>SpectraLink Voice Priority</td>
</tr>
<tr>
<td>TFTP</td>
<td>Trivial File Transfer Protocol</td>
</tr>
<tr>
<td>TLS</td>
<td>Transport Layer Security</td>
</tr>
<tr>
<td>TSM</td>
<td>Traffic Stream Measurement</td>
</tr>
</tbody>
</table>
### Table 10: Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>TU</td>
<td>Time Unit</td>
</tr>
<tr>
<td>TX</td>
<td>Transmission</td>
</tr>
<tr>
<td>UDID</td>
<td>Unique Device Identifier for an iOS Device</td>
</tr>
<tr>
<td>UDP</td>
<td>User Datagram Protocol</td>
</tr>
<tr>
<td>UDR</td>
<td>User Derived Rule</td>
</tr>
<tr>
<td>UPnP</td>
<td>Universal Plug and Play</td>
</tr>
<tr>
<td>VAP</td>
<td>Virtual Access Point</td>
</tr>
<tr>
<td>VBR</td>
<td>Virtual Beacon Report</td>
</tr>
<tr>
<td>VHD</td>
<td>Very High Density</td>
</tr>
<tr>
<td>VIA</td>
<td>Virtual Intranet Access</td>
</tr>
<tr>
<td>VIP</td>
<td>Virtual IP</td>
</tr>
<tr>
<td>VLAN</td>
<td>Virtual Local Area Network</td>
</tr>
<tr>
<td>VPN</td>
<td>Virtual Private Network</td>
</tr>
<tr>
<td>VRD</td>
<td>Validated Reference Design</td>
</tr>
<tr>
<td>VRIP</td>
<td>Virtual IP Address</td>
</tr>
<tr>
<td>VRRP</td>
<td>Virtual Router Redundancy Protocol</td>
</tr>
<tr>
<td>VSA</td>
<td>Vendor Specific Attributes</td>
</tr>
<tr>
<td>WebCC</td>
<td>Web Content Classification</td>
</tr>
<tr>
<td>WEP</td>
<td>Wired Equivalent Privacy</td>
</tr>
<tr>
<td>WIDS</td>
<td>Wireless Intrusion Detection System</td>
</tr>
<tr>
<td>WLAN</td>
<td>Wireless Local Area Network</td>
</tr>
<tr>
<td>WMS</td>
<td>WLAN Management System</td>
</tr>
<tr>
<td>WPA</td>
<td>Wi-Fi Protected Access</td>
</tr>
<tr>
<td>xSec</td>
<td>Extreme Security</td>
</tr>
</tbody>
</table>