Tech Note:

ClearPass Profiling TechNote

Date	Modified By	Comments
June 2014	Danny Jump	Initial Published Version 1

Table of Contents

Table of Figures

Overview

The following guide has been produced to help educate our customers and partners in understanding ClearPass endpoint profiling.

Note: Where you see a red-chili this is to signify a **'hot'** important point and highlights that this point is to be taken as a best-practice recommendation.

ClearPass Profile

Profile is a ClearPass module that automatically classifies endpoints using attributes obtained from software components called Collectors. As an example it can be used to implement BYOD flows where access has to be controlled based on the type of the device and the identity of the user. Profile can be set up in a network with minimal amount of configuration.

Setup

To classify devices using Profile, you need to set up the following:

Select one of the CPPM nodes in the Zone as profiler. Navigate to **Administration » Server Manager » Server Configuration** as shown below in Figure1.

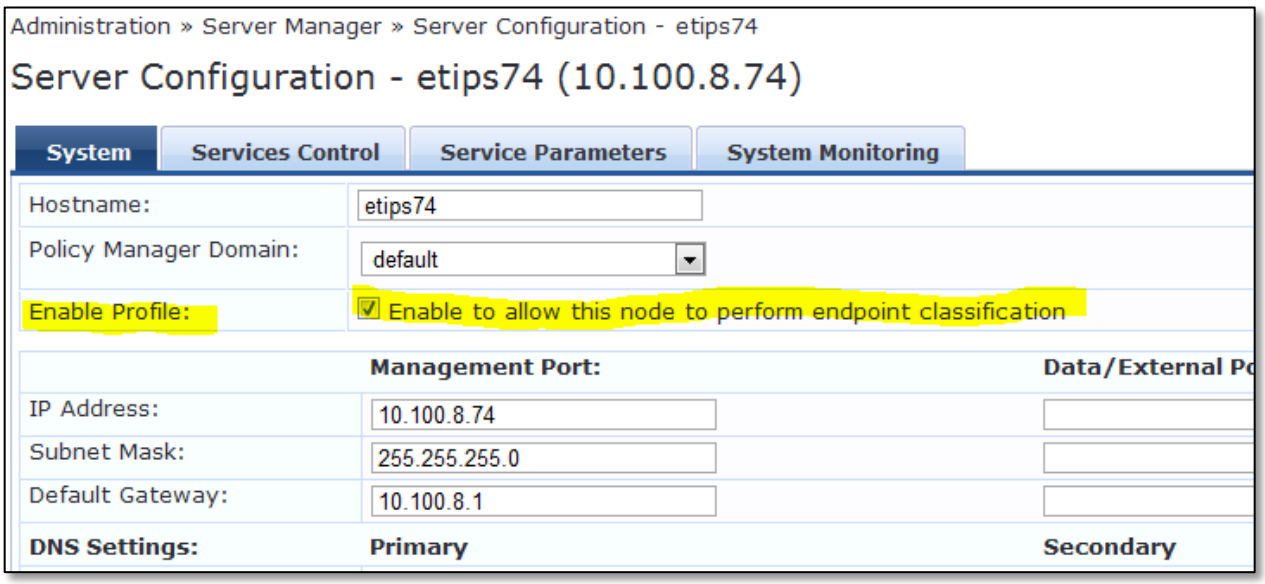

Figure 1 - Enabling 'Profiler' on a CPPM node

Once devices are classified, you can use them in policies to control access in your network. You can use the ***Authorization:[Endpoints Repository]*** attributes in the CPPM Role Mapping Policy. See section titled "Endpoint Profile Store as Authorization Source" for more information.

Device Profile

A device profile is a hierarchical model consisting of 3 elements - ***DeviceCategory***, ***DeviceFamily***, and ***DeviceName*** derived by Profile from endpoint attributes.

- ***DeviceCategory*** – This is the broadest classification of a device. It denotes the type of the device.
  Example: Computer, Smartdevice, Printer, Access Point, etc.
- ***DeviceFamily*** – This element classifies devices into a category; this is organized based on the type of OS or type of vendor.
  Example: Windows, Linux, Mac OS X are some of the families when category is Computer.
  Apple, Android are examples of DeviceFamily when category is SmartDevice.
- ***DeviceName*** - Devices in a family are further organized based on more granular details such as version.

Example: Windows 7, Windows 2008 server are device names under Windows family.

This hierarchical model provides a structured view of all endpoints accessing the network.

Apart from the these, Profile also collects and stores

- IP Address
- Hostname
- MAC Vendor
- Timestamp when device was first discovered
- Timestamp when device was last seen

Collectors

Collectors are network elements that provide data to profile endpoints. The following collectors send endpoint attributes to Profile:

- DHCP
  - DHCP snooping
  - Span ports
- ClearPass Onboard
- HTTP User-Agent
- MAC OUI – Acquired via various auth mechanisms such as 802.1X, MAC auth, etc.
- ActiveSync plugin
- CPPM OnGuard
- SNMP
- Subnet Scanner
- IF-MAP
- Cisco Device Sensor (Radius Accounting)
- MDM

DHCP

DHCP attributes such as option55 (parameter request list), option60 (vendor class) and options list from DISCOVER and REQUEST packets can uniquely fingerprint most devices that use the DHCP mechanism to acquire an IP address on the network. Switches and controllers can be configured to forward DHCP packets such as DISCOVER, REQUEST and INFORM to CPPM (DHCP Relay/ IP-Helper). These DHCP packets are decoded by CPPM to arrive at the device category, family, and name. Apart from fingerprints, DHCP also provides hostname and IP address.

DHCP Relay Agent – Aruba/Cisco

Configuring Aruba Controller and Cisco Switch to Send DHCP Traffic to CPPM

```
interface <VLAN_NAME>
ip address <IP_ADDR> <NETMASK>
ip helper-address <DHCP SERVER IP>
ip helper-address <CPPM IP>
```

Figure 2 - Adding multiple IP helpers

Notice how multiple **'ip helper-address'** can be configured to send DHCP packets to servers other than the DHCP server, i.e. a CPPM node.

DHCP SPAN

Certain networks precipitate the need to receive DHCP packets off a mirrored port, instead of relying on DHCP relays, which is used by CPPM for device profiling. In earlier release we support only dhcp relays. From 6.3 we support SPAN for receiving DHCP packets. Currently only the 25K HW appliances has additional ports beyond the MGMT/DATA interfaces where this can be utilized as a dedicated interface.

SPAN Configuration:

SPAN Port Configuration has to be done on switches where DHCP Servers (Source) and CPPM Servers (Destination) are connected.

Cisco Switch SPAN Configuration:

Local SPAN: Mirrors traffic from one or more interface on the switch to one or more interfaces on the same switch.

Configuring for Local SPAN: Local SPAN configures using "monitor session" command specifying source and destination on the same switch.

```
Switch1# configure terminal
Switch1(config)# monitor session 1 source interface fastEthernet0/2
Switch1(config)# monitor session 1 destination interface
fastEthernet0/24
Switch1(config)#end
```

Figure 3 - Configure Local Span port on older Cisco 2900/3500XL

Local SPAN configuration syntax on Cisco IOS release 12.2(33)SXH and beyond as shown below.

```
monitor session 1 type local
source int fa0/2
destination int fa0/24
```

Figure 4 - Configuring Local SPAN port IOS 12.2(33) and later (not ALL Cisco switches)

Good link for port mirroring example across different vendors….
http://www.securitywizardry.com/index.php/tools/switch-port-mirroring.html

Remote SPAN (RSPAN): An extension of SPAN called remote SPAN or RSPAN. RSPAN allows you to monitor traffic from source ports distributed over multiple switches, which means that you can centralize your network capture devices. RSPAN works by mirroring the traffic from the source ports of an RSPAN session onto a VLAN that is dedicated for the RSPAN session. This VLAN is then trunked to other switches, allowing the RSPAN session traffic to be transported across multiple switches. On the switch that contains the destination port for the session, traffic from the RSPAN session VLAN is simply mirrored out the destination port. Not all switches support remote SPAN.

Configuring RSPAN: Step1: In order to configure RSPAN you need to have a RSPAN VLAN, those VLANs have special properties and can't be assigned to any access ports. To create a VLAN for RSPAN on Cisco IOS, you must create the VLAN via the config-vlan configuration mode, as opposed to using the older VLAN database configuration mode. During the process of defining VLAN parameters, you must specify that the new VLAN is an RSPAN VLAN by configuring the remote-span VLAN configuration command.

```
Switch1# configure terminal
Switch1(config)# vlan 200
Switch1(config-vlan)# remote-span
Switch1(config-vlan)# end
Switch1# show vlan remote-span
Remote SPAN VLANs
------------------------------------------------------------------------------
200
Switch2# configure terminal
Switch2(config)# vlan 200
Switch2(config-vlan)# remote-span
Switch2(config-vlan)# end
Switch2# show vlan remote-span
Remote SPAN VLANs
------------------------------------------------------------------------------
200
```

Figure 5 - Configuring the RSPAN on 'Local' & 'Remote' Cisco switches

Step2: Next configure the RSPAN on Source switch: Unlike SPAN, where the source and destination ports exist on the same switch, the source and destination ports for an RSPAN session reside on different switches. This requires a separate RSPAN source session to be configured, as well as a separate RSPAN destination session to be configured.

```
Switch1# configure terminal
Switch1(config)# monitor session 1 source interface fastEthernet0/2 rx
Switch1(config)# monitor session 1 destination remote vlan 200
reflector-port fastEthernet0/24
Switch1# show monitor
Session 1
---------
Type              : Remote Source Session
Source Ports      :
   Rx            : Fa0/2
Reflector Port    : Fa0/24
Dest RSPAN VLAN   : 200
```

Figure 6 - Configuring the RSPAN monitor session on the 'Remote' switch

Step3: Configure the RSPAN on destination switch:

```
Switch2# configure terminal
Switch2(config)# monitor session 1 source remote vlan 200
Switch2(config)# monitor session 1 destination interface
fastEthernet0/3
Switch2(config)# exit
```

Figure 7 - Configuring the RSPAN monitor session on the 'Local' switch

Note: The RSPAN VLAN should be allowed in ALL trunks between the involved switches (Source and Destination switches in this case); if you have enabled "pruning" in your network, remove the RSPAN VLAN from the pruning, with the command: "**switchport trunk pruning vlan remove <RSPAN VLAN ID>**" under the interface configure as trunk.

Encapsulated remote SPAN (ERSPAN): Encapsulated Remote SPAN (ERSPAN), as the name says, brings generic routing encapsulation (GRE) for all captured traffic and allows it to be extended across Layer 3 domains, i.e. cross a WAN.

Aruba Switch SPAN Configuration:

```
Enable vlan's used

interface-profile switching-profile "vlan6"
   access-vlan 6
!

Configure a mirroring profile, which will be the destination port where
cppm is connected.

interface-profile mirroring-profile "dhcp-span-port-4-vineeth"
   destination gigabitethernet "0/0/5"
!
Configure source port where DHCP server is connected.

interface gigabitethernet "0/0/6"
   mirroring-in-profile "dhcp-span-port-4-vineeth"
   mirroring-out-profile "dhcp-span-port-4-vineeth"
   switching-profile "vlan6"
```

Figure 8 - Configure SPAN on MAS

CPPM Log to debug :

- Enable log level to DEBUG for Async-Netd service in CPPM.

ClearPass Onboard

ClearPass Onboard collects rich and authentic device information from all devices during the onboarding process. Onboard then posts this information to Profile via the Profile API. Since the information collected is definitive, Profile directly classifies these devices into their Category, Family and Name, without having to rely on any other fingerprinting information.

HTTP User-Agent

In some cases, DHCP fingerprints alone cannot fully classify a device. A common example is the Apple family of smart devices; DHCP fingerprints cannot distinguish between an Apple iPad and an iPhone. In these scenarios, User-Agent strings sent by browsers in the HTTP protocol are useful to further refine classification results.

User-Agent strings are collected from:

- ClearPass Guest
- ClearPass Onboard
- Aruba controller through IF-MAP interface

MAC OUI

Mac OUI can be useful in some cases to better classify endpoints. An example is Android devices, where DHCP fingerprints can only classify a device as a generic Android device, but it cannot provide more detail about vendor. Combining this information with MAC OUI, Profile can classify a device as HTC Android, Samsung Android, Motorola Android, etc. MAC OUI is also useful to profile devices such as printers which may be configured with static IP addresses.

ActiveSync plugin

ActiveSync plugin is a Windows Service component (that is, it runs as a service on the Exchange server) provided by Aruba to be installed on Microsoft Exchange servers. When a device communicates with the corporate Exchange Server using the ActiveSync protocol, it provides attributes such as device type and user agent. These attributes are collected by the plugin software and are sent to CPPM Profile. Profile uses dictionaries to derive profiles from these attributes.

QA Tested Version: MicroSoft Exchange Server 2010

Configuration of ActiveSync plugin

Installation

1. The plugin is packaged as ArubaMSExchangePlugin.zip. This containes two files:
   a. setup.exe
   b. MSExchangePlugin.msi
2. Extract and copy both files on Microsoft Exchange Server 2010
3. Double click on "setup.exe" and install the Aruba MSExchange Plugin

Installation Folders

The plugin gets installed under **"C:\Program Files\ArubaNetworks\"** on 32-bit systems, and under **"C:\Program Files (x86)\ArubaNetworks\"** on 64-bit systems.

Folder structure is:

- $install_root\bin ==> Contains binaries of MSExchange Plugin
- $install_root\etc ==> Contains configuration files
- C:\ArubaNetworks\MSExchangePlugin\data ==> Contains ActiveSync plugin records which are periodically collected by the plugin
- C:\ArubaNetworks\MSExchangePlugin\var ==> Contains plugin log files

Configuration Files

1. IIS log reader configuration file

Location : $install_root\etc\iislogreader.conf

The contents of the configuration file are pasted below:

```
[iis-log-config]
  logDir=C:/inetpub/logs/LogFiles/W3SVC1
  ##################################################
  # If advanced logging is enabled then make sure you
  # specify the path for advanced logging files
  # in the logDir variable
  ##################################################
  advancedLogging=0
  ##################################################
  # Read interval in seconds
  ##################################################
  readInterval=300
  ##################################################
  # Refresh interval for active sync records
  ##################################################
  refreshInterval=14400
```

2. ActiveSync log record configuration file Location : $install_root\etc\logrecord.conf

Contents of the configuration file are pasted below: Note the section highlighted in **RED** below which refers to the CPPM node where the plugin transmits data to. The username must be a LOCAL ADMIN-USER **(Administration-> Users and Privileges->Admin Users)** user configured on the CPPM node with a role of API Administrator.

```
[log-record-config]
############################################################
# This is the data directory where the ActiveSync records
# are stored prior to sending it to Profile
############################################################
dataDir=C:/ArubaNetworks/MSExchangePlugin/var/data
[log-dispatcher-config]

############################################################
# This is the Profile URL and login credentials
############################################################
url=http://<profile-ipaddress>/async_netd/deviceprofiler/endpoints
username=<XXXXXXXXX>
password=<YYYYYYYYY>
```

3. MSExchange Plugin configuration file Location : $install_root\etc\msexchange-plugin.conf

Contents of the configuration file are pasted below:

```
[domain-controller-info]
############################################################
# AD domain controller name
############################################################
serverName=WIN2008R2DEV-AD.dev.avendasys.com
############################################################
# AD domain controller base dn
############################################################
baseDn=dc=dev,dc=avendasys,dc=com
############################################################
# AD domain authentication source name
############################################################
authSourceName=
############################################################
# AD domain bind dn
```

```
    ##########################################################
    bindDn=cn=Administrator,cn=Users,dc=dev,dc=avendasys,dc=com
    ##########################################################
    # AD domain bind password
    ##########################################################
    bindPassword=password
    ##########################################################
    # Filter configuration
    ##########################################################
    userFilter=(&(objectClass=user)(sAMAccountName=%s))
    groupFilter=(&(objectClass=group)(member=%s))

deviceFilter=(&(objectClass=top)(objectClass=msExchActiveSyncDevice))
    ##########################################################
    # Attributes to fetch
    ##########################################################

attributes=distinguishedName,msExchDeviceID,msExchDeviceModel,msExchDev
iceType,msExchDeviceUserAgent
```

Any configuration file changes above require the restart of Aruba MSExchange Plugin service.

CPPM OnGuard

ClearPass OnGuard agents perform advanced endpoint posture assessment. It collects and sends OS details from endpoints during authentication. Profile uses os_type attribute from Onguard to derive a profile. For example, a Device Name of Windows XP can be further classified as Windows XP Service Pack 3.

SNMP

Endpoint information obtained by reading SNMP MIBs of network devices is used to discover and profile static IP devices in the network. The following information read via SNMP is used:

- **sysDescr** information from RFC1213 MIB is used to profile the device. This is used both for profiling switches/controllers/routers configured in CPPM, and for profiling printers and other static IP devices discovered through SNMP or subnet scans.
- **cdpCacheTable** information read from CDP (Cisco Discovery Protocol) capable devices is used to discover neighbour devices connected to switch/controller configured in CPPM
- **lldpRemTable** information read from LLDP (Link Layer Discovery Protocol) capable devices is used to discover and profile neighbour devices connected to switch/controller configured in CPPM
-

Note: The SNMP based mechanism is only capable of profiling devices if they respond to SNMP, or if the device advertises its capability via Link Layer Discovery Protocol (LLDP). When performing SNMP reads for a device, CPPM uses SNMP Read credentials configured in Network Devices, or defaults to using SNMP v2c with the "public" community string.

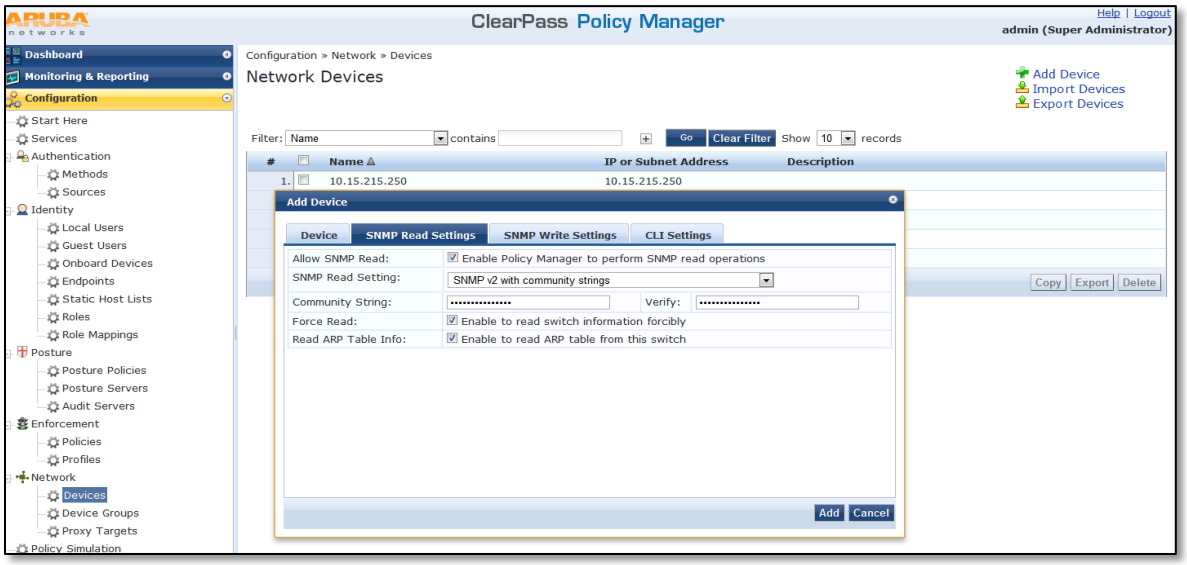

Figure 9 - Setting SNMP community attributes

Discovering endpoint with static IP address

There are two ways to discover endpoints that are statically addressed:

- ARP read
- Subnet scan

Discovery via ARP Read

ARP table read from network devices is used as a means to discover endpoints in the network.

Network Devices configured with SNMP Read enabled are polled periodically for updates based on the time interval configured in ***Administration -> Server Configuration -> Service Parameters -> ClearPass network services -> Device Info Poll Interval***

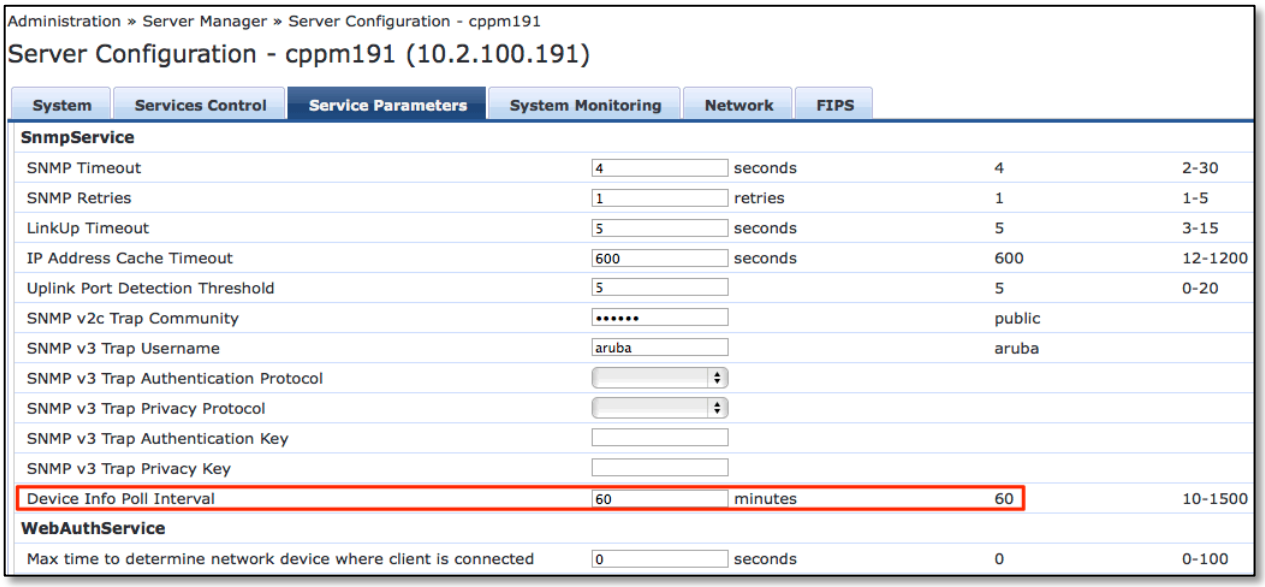

Figure 10 – Setting ARP read frequency

The following additional settings have been introduced for the ARP table read:

1. **Read ARP Table Info** – Enable this setting if this is a L3 device and you want to use the ARP table on this device as a way to discover endpoints in the network. Static IP endpoints discovered this way are further probed via SNMP to profile the device.

2. **Force Read** – Enable this to ensure all CPPM nodes in the cluster read SNMP information from this device irrespective of trap configuration on the device. This option is especially useful when demonstrating static IP based device profiling, since this does not require any trap configuration on the network device.

3. In large or geographically spread cluster deployments you do not want all CPPM nodes to probe all SNMP configured devices. The default behavior is for a CPPM node in the cluster to read network device information only for devices configured to send traps to that CPPM node.

Discovery via Subnet Scanner

Network subnet scan is used to discover IP addresses of devices in the network. We use NMAP to discover the devices and whether they have SNMP port 161 open, we then fingerprint these devices to gather additional data. The devices are probed based upon SNMP community strings configured for a SUBNET or HOST address under **Configuration-> Networks->Devices**. Configuring a device here with an IP or Subnet Address provides Profiler with the SNMP community strings it needs to gather more data. Profiler will use the most specific entry from the Devices list for its SNMP community strings, i.e. if a device is configured with a 172.16.1.0/24 and a SNMP RO string of **arubaro** but a device in this subnet is configured with an address of 172.16.1.250/32 with a SNMP RO string of **danny** then for this single device this string is used, for the rest of the subnet **arubaro** will be used.

Note: If no match if found then we will probe devices using the default community string **public** and type V2c.

When defining the device, the option to select '**Force Read**' and '**Read ARP Table Info**' is allowed. This ONLY applies to devices configured with a HOST IP address, not a SUBNET. Note that if a cluster of CPPM nodes exists, the **'Force Read'** option results in all nodes in the cluster probing the ARP table of the device which is not desired. If the **'Force Read'** option is not enabled, device ARP table is read only by the CPPM nodes that are configured as SNMP trap targets in the network device (for Cold Start/Warm Start/Link traps).

Figure 11 - Setting community string and enabling ARP-read

Subnets to scan are configured per CPPM Zone. This is particularly useful in deployments that are geographically distributed. In such deployments, it is recommended that you assign the CPPM nodes in a cluster to multiple "Zones", based on the geographical area served by that node, and enable Profile on at least one node per zone. Below we have created an additional zone '**California**' to that of '**default'** and then assigned the IP Subnets specific to that new zone as can be seen below.

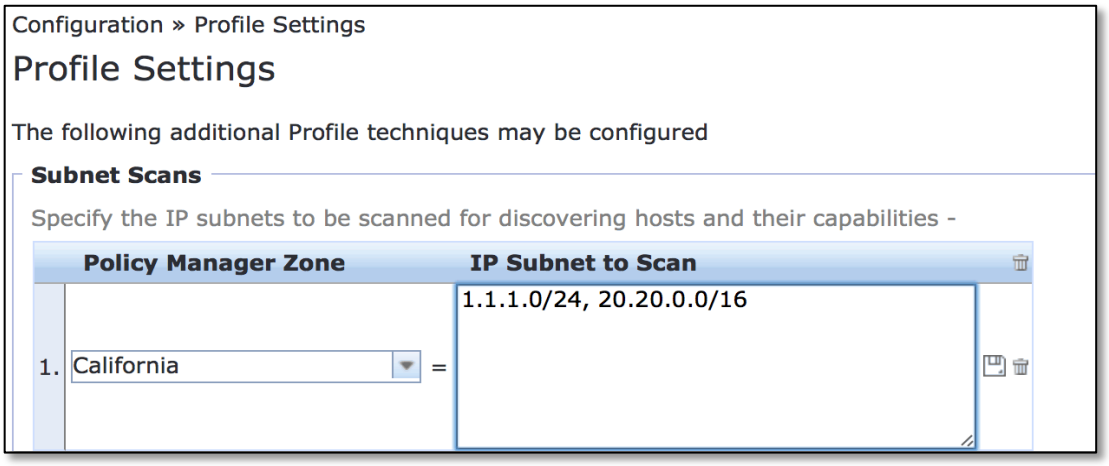

Figure 12 - Assigning IP SUBNETS in Profiler to zones

The frequency of the SUBNET scan is controlled from cluster-wide settings and by default this occurs ONCE every 24-hours.

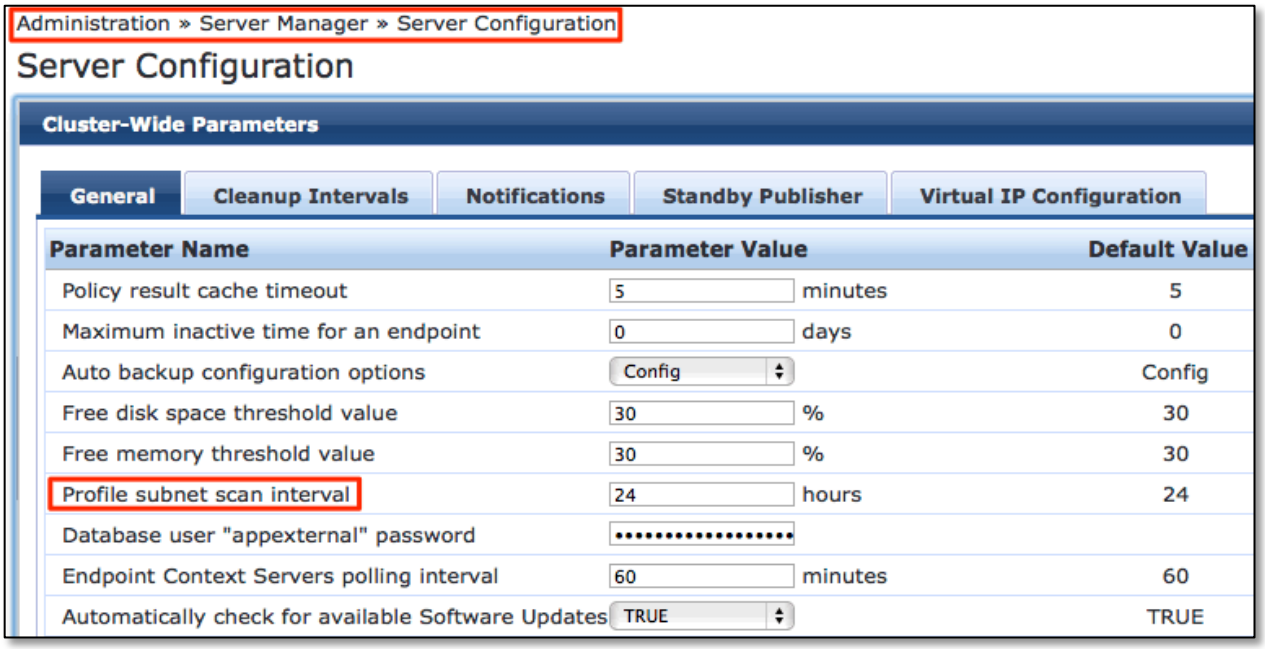

Figure 13 - Configuring SUBNET scan frequency

IF-MAP

If configured Aruba Controller (AOS 6.3 and higher) can send HTTP user-agent and DHCP packets through IF-MAP interface. IF-MAP info sent by a wireless client has mac, ip and user-agent. But wired clients can only provide ip and user-agent, hence dhcp relay has to be properly configured to populate IP-MAC table to fetch the mac address for given IP.

Configurations of IF-MAP on AOS Controller:

To enable IF-MAP on Aruba controller:

In the **GUI:** follow these steps

Go to: **Configuration -> Advanced Services > All Profile Management > Other Profiles-> CPPM IF-MAP**

Click Enable: **CPPM IF-MAP Interface** and ADD CPPM Server details as required, this will add the CPPM node.

- Host: **<CPPM IP Address> or <FQDN>**
- Port : **443**
- Username : **apiadmin**
- Password : apiadmin <password>

* Configure and user an admin user with privilege level, API Administrator or Read only administrator.

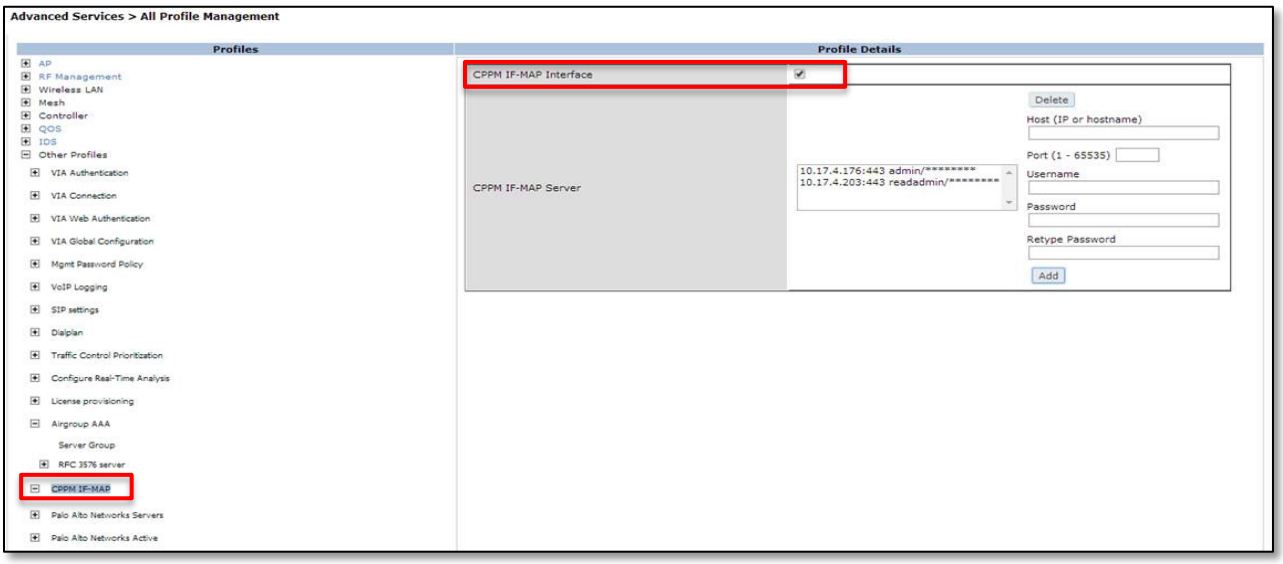

Figure 14 - Enabling Aruba Ctrl to send IF-MAP info to CPPM (GUI)

CLI:

Syntax

Parameter	Description	Default
enable	Enables the IFMAP protocol.	–
server	Configures the CPPM IF-MAP server.	–
host <host>	IP address/hostname of the CPPM IF-MAP server.	–
port <port>	Port number for the CPPM IF-MAP server. The range is 1-65535.	443
username<username>	Username for the user who performs actions on the CPPM IF-MAP server. The name must be between 1-255 bytes in length.	–
passwd <password>	Password of the user who performs actions on the CPPM IF-MAP server. The password must be between 6-100 bytes in length.	–

Example

This example configures IFMAP and enables it.

```
(host) (config) #ifmap
(host) (config) #ifmap cppm
(host) (CPPM IF-MAP Profile) #server host <host>
(host) (CPPM IF-MAP Profile) #port <port>
(host) (CPPM IF-MAP Profile) #passwd <psswd>
(host) (CPPM IF-MAP Profile) #enable
```

Figure 15 - Enabling Aruba Ctrl to send IF-MAP info to CPPM (CLI)

CPPM Logs to debug :

- Enable log level to DEBUG for IF-MAP from CPASS-Network-Service.
- Enable log level to DEBUG for async-netd service.

Cisco Device Sensor

Device Sensor feature is used to gather raw endpoint data from network devices using protocols such as Cisco Discovery Protocol (CDP), Link Layer Discovery Protocol (LLDP), DHCP and HTTP User-Agent info. All these collected information attributes are sent to CPPM using radius accounting packets. On receiving accounting packets radius server will post these inputs to profiler for device profiling. . This feature targets the information gleaned from accounting packets received in CPPM to the profiler component so that end points can be profiled without needing IP helper configuration or port SPAN.

Note: Currently this works only with Cisco devices.

Tested Versions

Cisco switch supports [Version 15.0(2)SE2] : DHCP,CDP and LLDP
Cisco controller supports [Version 7.5.102.0] : DHCP and HTTP_User_Agent

Basic Configuration needed:

1. CPPM should be configured with interim accounting packets update enabled.
2. Accounting configuration on NAD.
3. Enable IOS sensor on NAD.

Cisco switch configuration.

1. Basic radius configuration with accounting enabled.
2. Add device-sensor configuration as follows.

Configuration to enable global device sensor in Cisco switch:

```
device-sensor accounting
device-sensor notify all-changes
```

Figure 16 - Enabling device sensor on Cisco switch

Device sensor filter configuration to add what DHCP info in accounting packets.

```
device-sensor filter-list dhcp list dhcp-list
option name host-name                          [ Supported Value 1 :
dhcp option 12]
option name parameter-request-list  [ Supported Value 2 : dhcp option
55]
option name class-identifier                   [Supported Value 3 :
dhcp option 60]
!
```

Figure 17 - Configuring device sensor on Cisco switch

Device sensor filter configuration to set what LLDP TLV info is in accounting packets.

```
device-sensor filter-list lldp list lldp-list
tlv name system-description [Supported value 1 : TLV 0006 –
lldp_sys_description]
!
```

Figure 18 - Enabling device sensor LLDP TLV attributes

Device sensor filter configuration to set what CDP info is in accounting packets.

```
device-sensor filter-list cdp list cdp-list
tlv name version-type     [ Supported Value 1 : TLV 0005 –
cdp_sys_description]
tlv name platform-type  [Supported Value 2 :TLV 0006 –
cdp_cache_platform ]
```

Figure 19 - Enabling device sensor CDP TLV attributes

Configurations to enable DHCP, LLDP and CDP filter in accounting packets

```
device-sensor filter-spec dhcp include list dhcp-list
device-sensor filter-spec lldp include list lldp-list
device-sensor filter-spec cdp include list cdp-list
```

Figure 20 - Enabling device sensor filter for DHCP, LLDP & CDP

Globally enable LLDP.

```
Switch# configure terminal
Switch(config)# lldp run
Switch(config)# end
```

Figure 21 – Globally enable LLDP

Enable LLDP on an interface.

```
Switch# configure terminal
Switch(config)# interface gigabitethernet1/0/1
Switch(config-if)# lldp transmit
Switch(config-if)# lldp receive
Switch(config-if)# end
```

Figure 22 – Enable LLDP on an interface

Globally enable CDP.

```
Switch# configure terminal
Switch(config)# cdp run
Switch(config)# end
```

Figure 23 – Globally enable CDP

Cisco WLC Configuration.

1. Login to WLC
2. Configure a WLAN with DHCP profiling.
    1. Go to WLAN configuration Advanced tab.
    2. Enable DHCP Addr. Assignment Required
    3. Enable DHCP profiling and HTTP profiling under option Radius client profiling.

CPPM Log to debug:

- Enable log level to DEBUG for Radius Server.
- Enable log level to DEBUG for Async-Netd service.

Mobile Device Management (MDM)

Introduction: With the release of ClearPass Policy Manager 6.0.2 and the subsequent release of ClearPass Policy Manager, integration options are now available with the major Mobile Device Management (MDM) platforms, allowing Aruba ClearPass customers to extend the knowledge of managed device state (device type, policy compliance) down to the business rules that govern their corporate network admission policies.

For example, if the MDM platform detects that a device is jailbroken, the MDM platform only has the option to attempt to enforce the business policy at the device level. By extending this policy state to ClearPass as the network policy definition point, the jailbreak status of a device can be used to deny access or quarantine this device the next time it attempts to connect to the secure network.

TechNote: Please review the MDM TechNote for more indepth informtation about our CPPM and MDM Integration, click here to access this document.

How it works: A service running in CPPM periodically polls MDM servers using their exposed APIs. Device attributes obtained from MDM are added as endpoint tags. Profiler related attributes are send to profiler which uses these attributes to derive final profile

Below we show an example of the additional attributes that can be integrated into the ClearPass Endpoint profiler database that could be received from an MDM vendor. Not all MDM vendors expose the same level of data, but we normalize the information received and present it in a standard attribute template in the endpoint database.

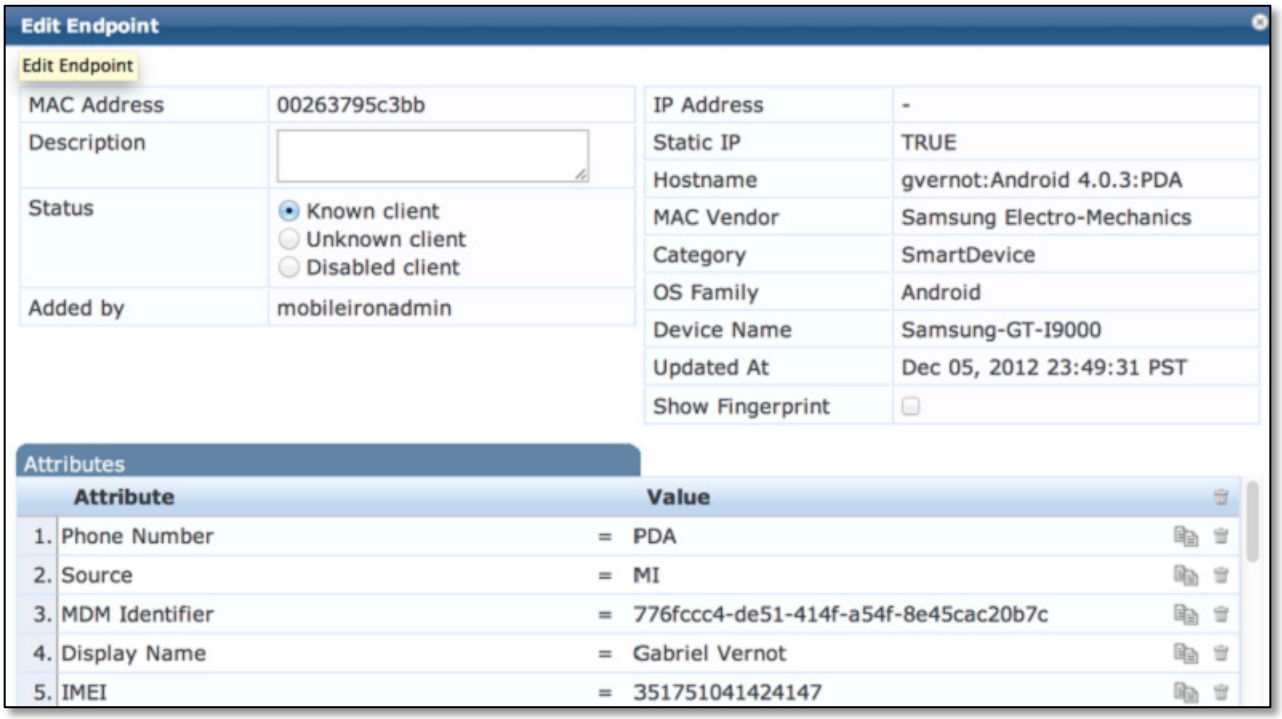

Figure 24 - Example of MDM attributes 1…

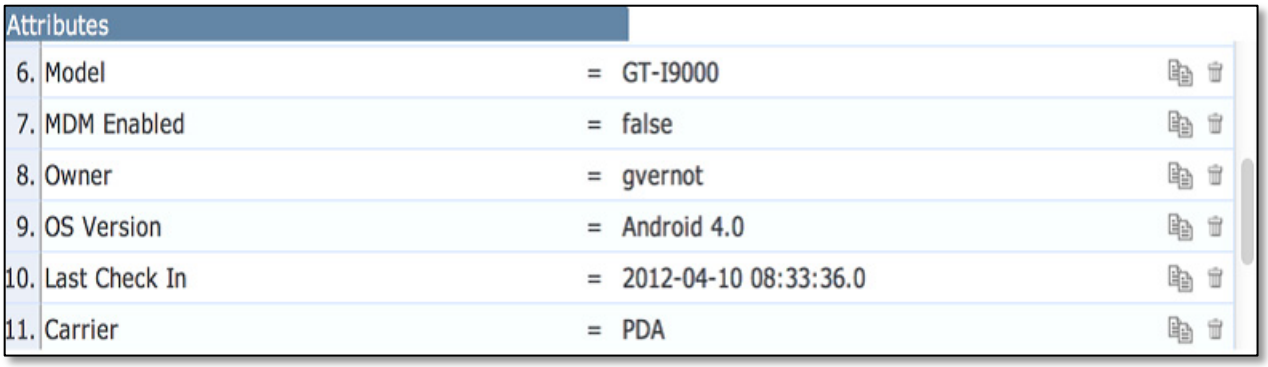

Figure 25 - Example of MDM attributes 2…

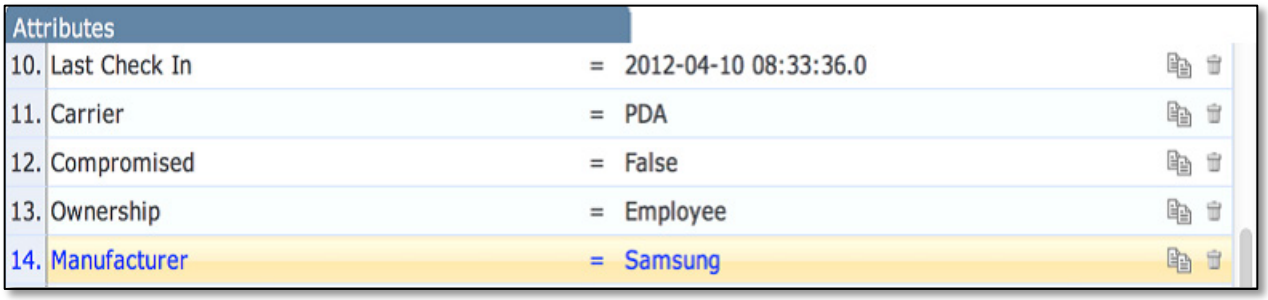

Figure 26 - Example of MDM attributes 3…

MDM Configuration Details

From the **Administration** menu of ClearPass Policy Manager, the menu option called **Endpoint Context Servers** is used to add and configure the MDM Servers.

Use "**Add**" option to add a specific type of MDM Server, the following figure shows various MDM Servers that are supported by CPPM.

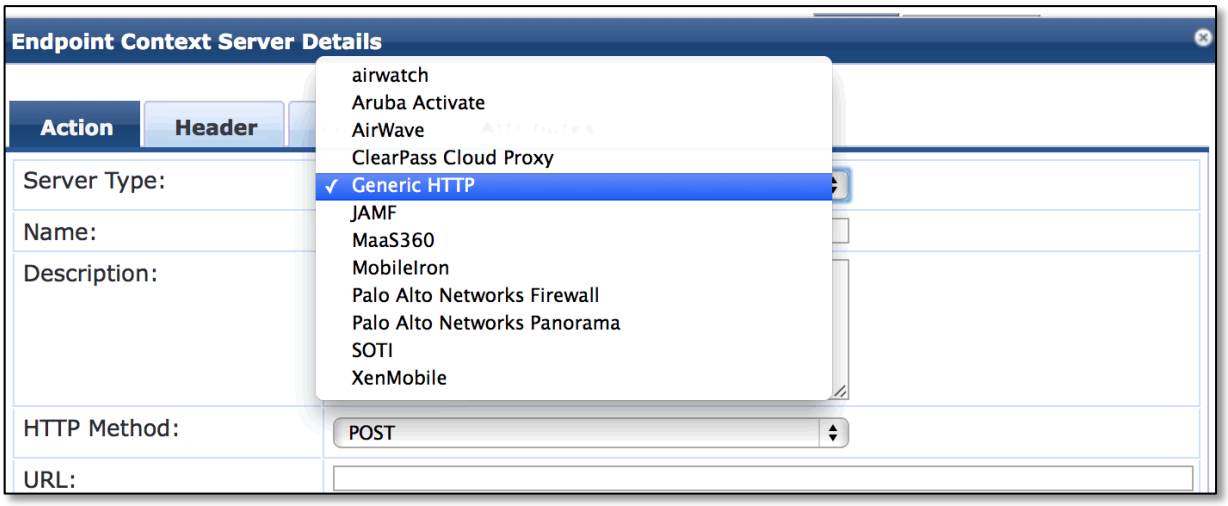

Figure 27 - Adding an MDM context server

Some minor differences exist in various types of MDM vendors with respect to some parameters for polling and fetching the details. Some of them are shown below, more are detailed in the MDM TechNote.

- Airwatch makes use of an API Key
- MaaS360 makes use of an Application Access Key, Application ID, Application Version, Platform ID and a Billing ID
- SOTI makes use of a Group ID

The polling interval for MDM Servers can be set at the cluster level from

Administration > Server Manager > Server Configuration and click on Cluster-Wide Parameters

Figure 28 - Setting Cluster Wide Parameters

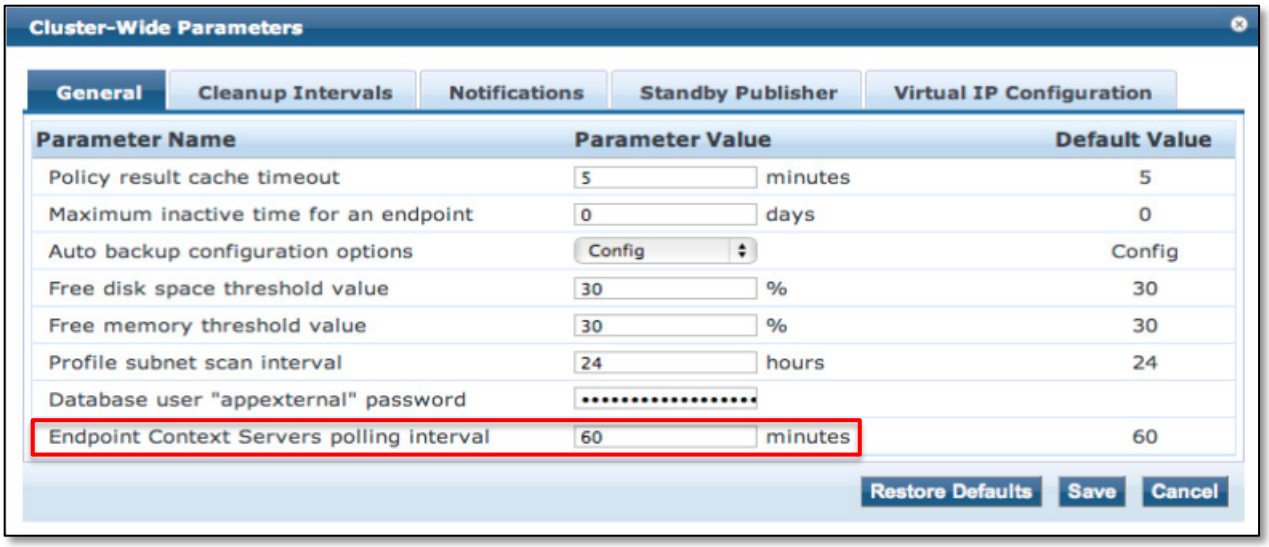

Figure 29 - Setting MDM polling frequency

Profiling

Profile uses a **two-stage** approach to classifying endpoints using input attributes.

Stage 1

Stage 1 tries to derive device profiles using static dictionary lookups. Based on the attributes available, CPPM looks up DHCP, HTTP, ActiveSync and MAC OUI dictionaries, and derives multiple matching profiles. Each attribute from a source (eg DHCP, SNMP etc) is assigned 2 weights – **reliability** and **score**.

If profiling results in multiple matches, these weights are used to find best match. All matches are sorted on (**reliability, score)** tuple and one with highest value is chosen.

Attributes	Reliability	Score
dhcp:options	98	95
dhcp:option55	98	95
dhcp:option60	99	96
snmp:sys_descr	100	97
snmp:cdp_cache_platform	100	97
snmp:device_type	98	1
snmp:name	98	2
host:os_type	100	100
host:user_agent	10	99
active_sync:device_type	100	99
active_sync:user_agent	100	99

Figure 30 - Profiling Reliability/Score

In addition to these attributes, mac_vendor and hostname are also used in Stage-2 rule evaluation.

Example: In this example an Aruba controller proxied HTTP requests from an Apple iPad. HTTP User-Agent classifies the device as an Apple iPad. SNMP collector provides sys_descr which classifies the device as Aruba controller. As **the device_category** of profile derived from these 2 inputs are different (Computer, Controller), CPPM's profiles picks the one with highest reliability and finally classifies this device as Aruba Controller.

```
curl -X POST http://localhost:6180/async_netd/deviceprofiler/endpoints \
   -H "Content-Type: application/json" -d                    \
   '[{"mac" : "000b86625750",
     "host": {
        "user_agent" : "iPad;"
     },
     "snmp": {
        "sys_descr" : "ArubaOS (MODEL: Aruba620), Version"
     }
```

Stage 2

CPPM comes pre-built with a set of rules that evaluates a device profile. CPPM uses all input attributes and device profiles from **Stage 1**. The resulting rule evaluation may or may not result in a profile. **Stage 2** is intended to refine the results of profiling.

Example: DHCP option55 classifies device as Android. Stage 2 rules reclassifies the device as HTC Android by combining mac-vendor information.

```
curl -X POST http://localhost:6180/async_netd/deviceprofiler/endpoints \
    -H "Content-Type: application/json" -d                    \
    '{"mac" : "00092d112233",
     "hostname" : "myandroid.domain.com",
     "dhcp" : {
         "options" : ["53,55,57,61,51"],
         "option55" : ["1,121,33,3,6,12,15,28,51,58,59,119", ""]
      }
   }';;
```

Post Profile Actions

After profiling an endpoint, profile can be configured to perform RADIUS Change of Authorization (CoA) on the NAD to which an endpoint is connected. Post profile rules are configured in the CPPM Service configuration wizard. Make sure you turn on **"Profile Endpoints"** from the Service tab:

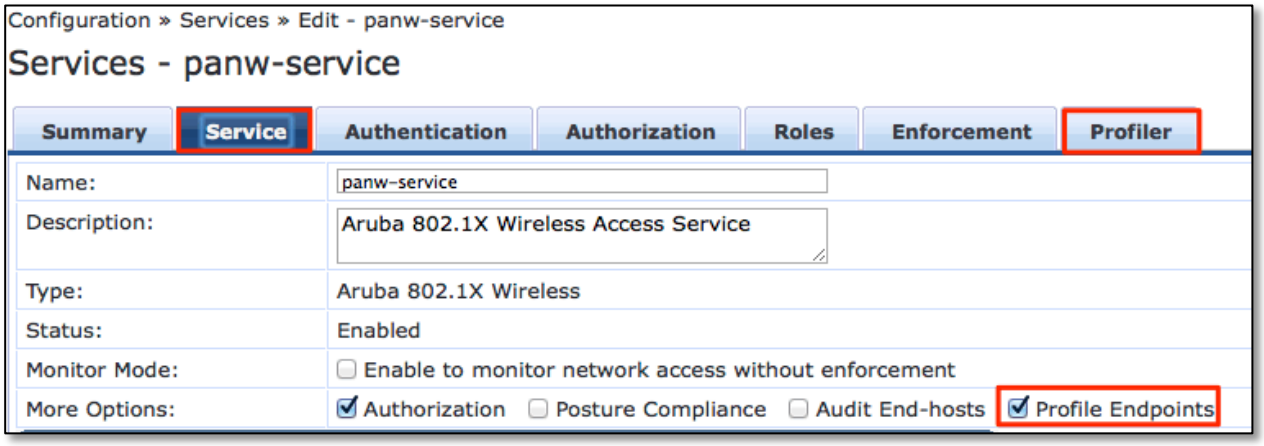

Figure 31 - Enabling Profiler on a service

Configure **[Endpoints Repository]** as Authorization Source. Endpoint profile attributes derived by Profile are available through the **'[*Endpoint Repository*]'** authorization source. These attributes can be used in role-mapping or enforcement policies to control network access. Available attributes are:

- ***Authorization:[Endpoints Repository]:MAC Vendor***
- ***Authorization:[Endpoints Repository]:Category***
- ***Authorization:[Endpoints Repository]:OS Family***
- ***Authorization:[Endpoints Repository]:Name***

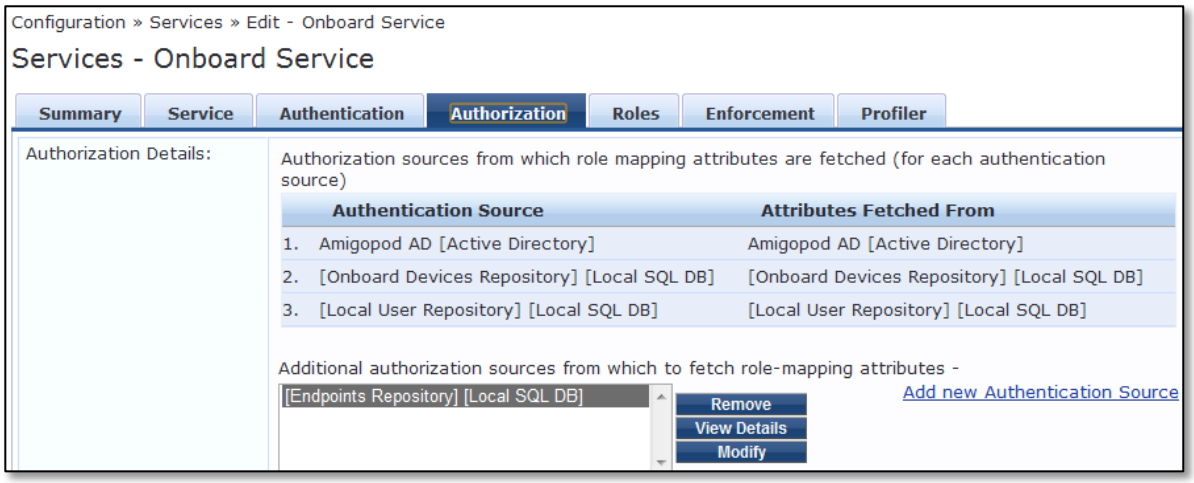

Figure 32 – Using [Endpoints Repository] as Authorization Source

You can select a set of categories and a CoA profile to be applied when the profile matches one of the selected categories. CoA is triggered using the selected CoA profile. ANY option from ***'Endpoint Classification'*** can be used to invoke CoA on a change of any one of the fields (category, family, and name).

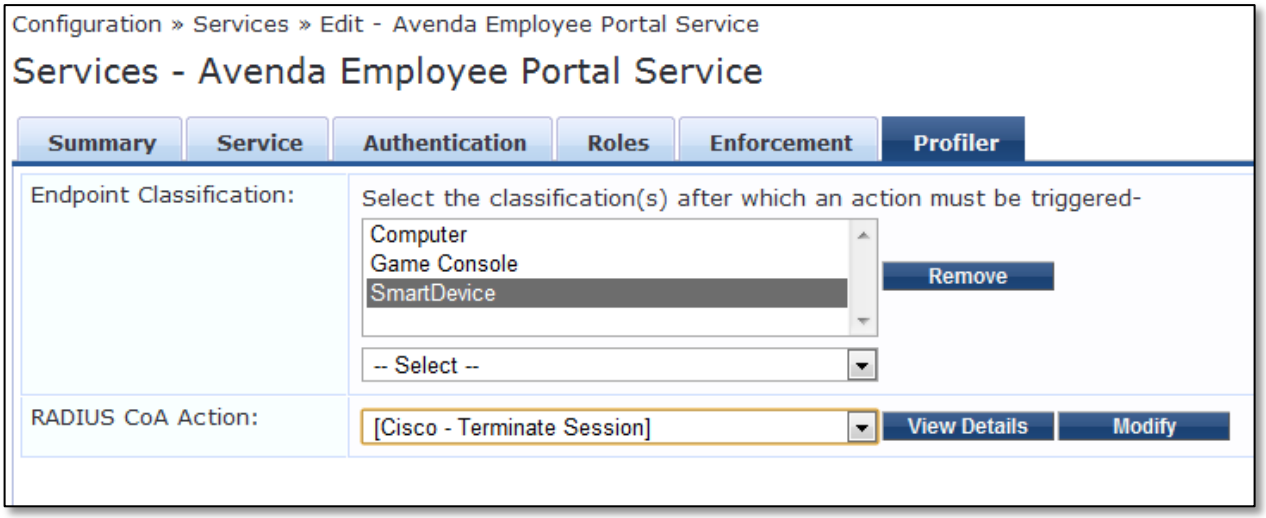

Figure 33 - Send CoA based upon endpoint classification

Use profiled endpoint attributes in Role Mapping Rules

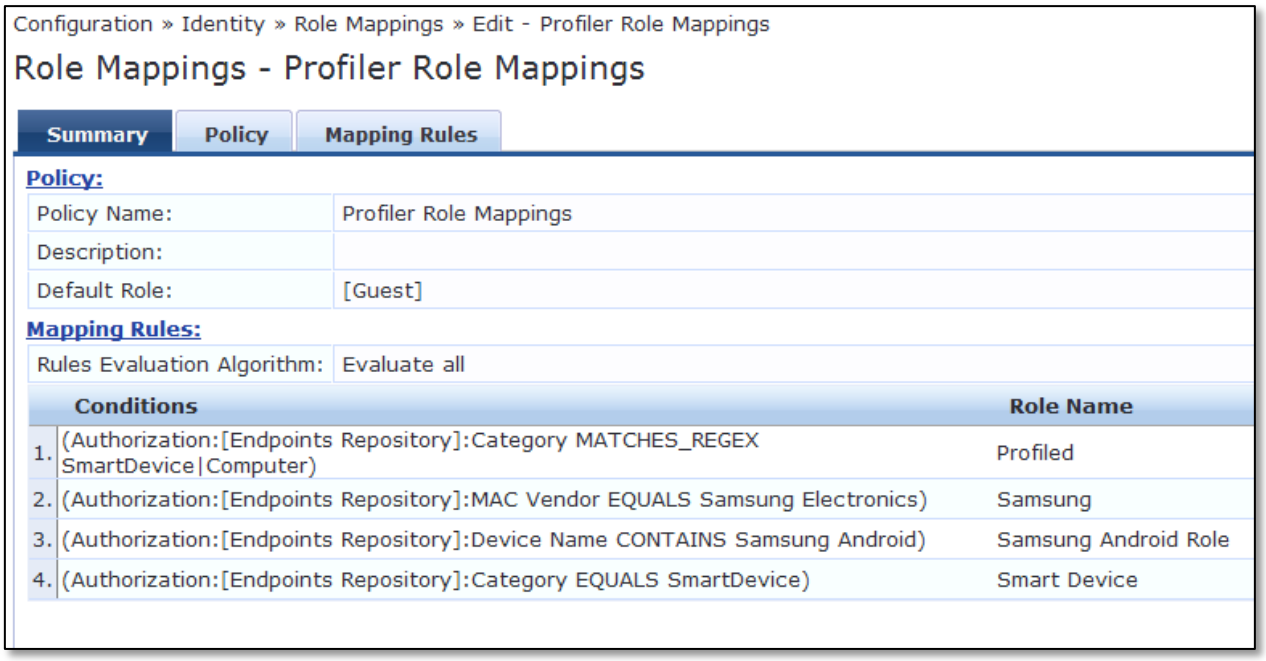

Figure 34 - Example of using Profiled info in role-mapping

Fingerprint Dictionaries

CPPM uses a set of dictionaries and built-in rules to perform device fingerprinting. Listed below are the dictionaries used by CPPM.

- DHCP
- HTTP User-Agent
- ActiveSync attributes
- SNMP attributes
- MAC OUI

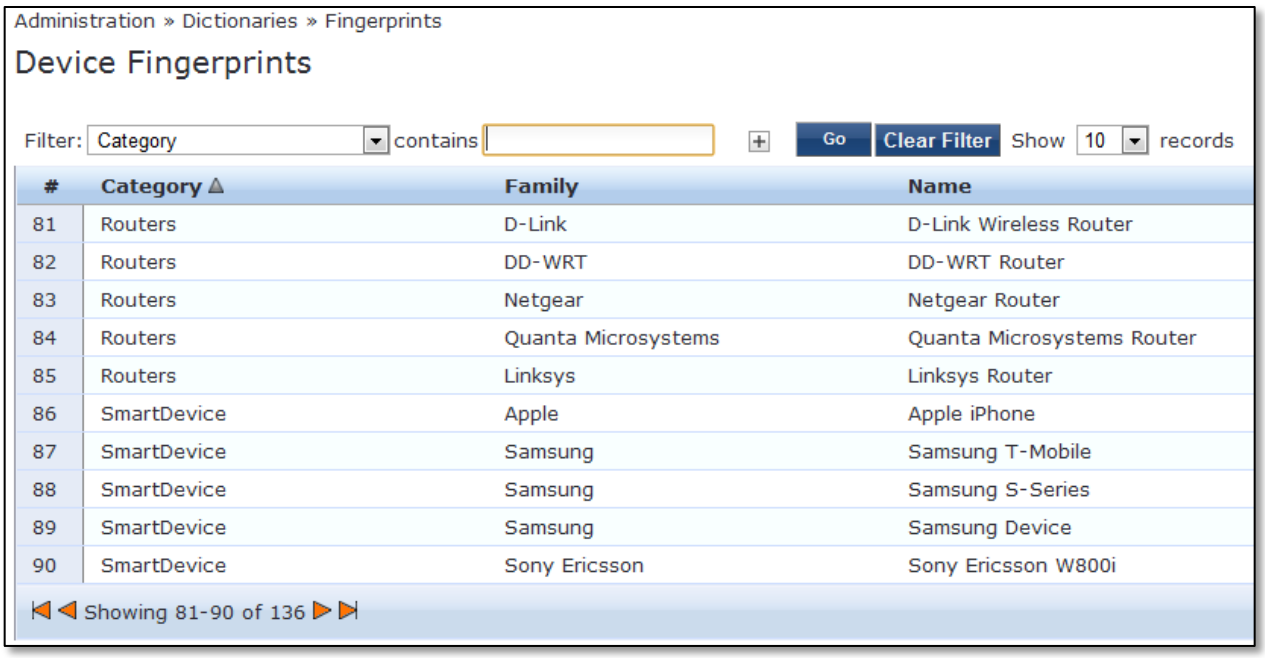

Figure 35 - Example set of Device Fingerprint Dictionaries in CPPM

As these dictionaries can change frequently, CPPM provides a way to automatically update fingerprints from an Aruba hosted portal. If external access cannot be provided to CPPM, the fingerprints file can be downloaded and imported through CPPM admin. The following screenshots show the configuration details for online and manual fingerprint updates.

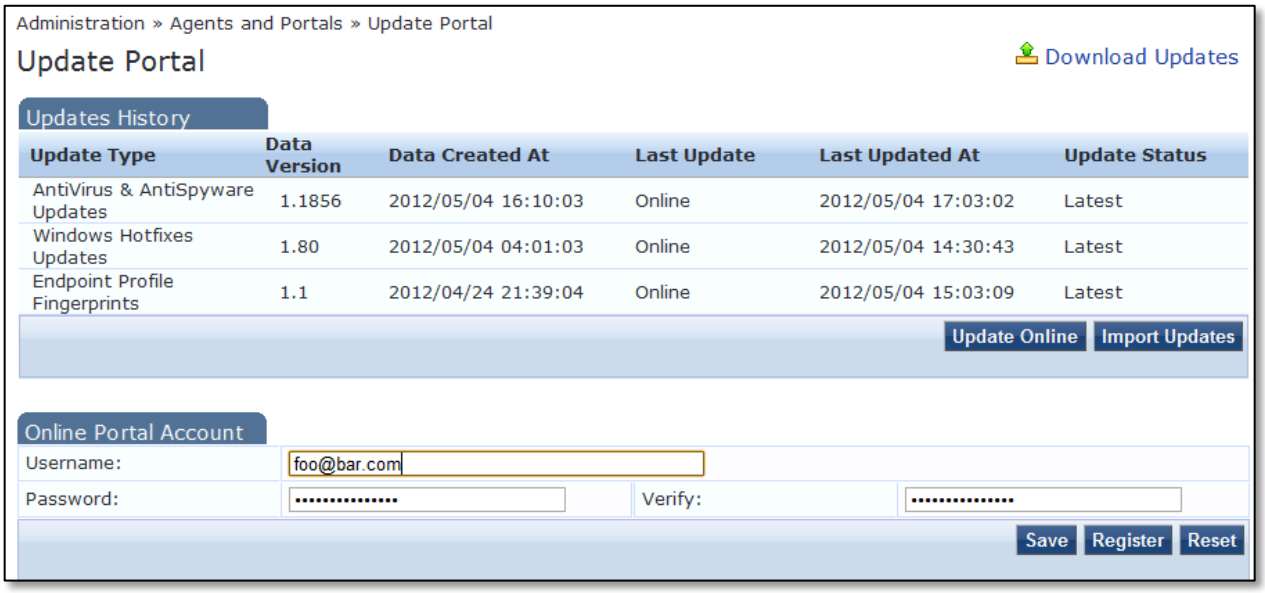

Figure 36 - CPPM WEB s/w Update

Profile Redundancy

If profiling is enabled on multiple nodes within a zone, they will form a cluster which provides redundancy and load balancing. The node with lowest UUID assumes an active role. All other nodes proxy endpoint attributes to active profiler. Active profiler periodically sends heartbeats to peers. If active node goes down, heartbeats will be lost and next peer with lowest UUID assumes master role.

When failed node comes back, it will start sending heartbeats and assumes master role. If any peer has assumed master role, it will change to passive role on receiving heartbeats from original master.

Profiler Load Balancing

Collectors can run on any node and can proxy extracted attributes to active profiler. This property of profiler helps to spread load across multiple CPPM nodes.

Example: DHCP relay or span is configured to a CPPM node which is not enabled as profiler. This node can perform required packet processing, extract mac, ip, hostname, option55, option60 and send to active profiler.

Profile UI

CPPM provides user interfaces to search and view profiled endpoints. It also provides basic statistics on the profiled endpoints.

Dashboard widget showing basic distribution of device types

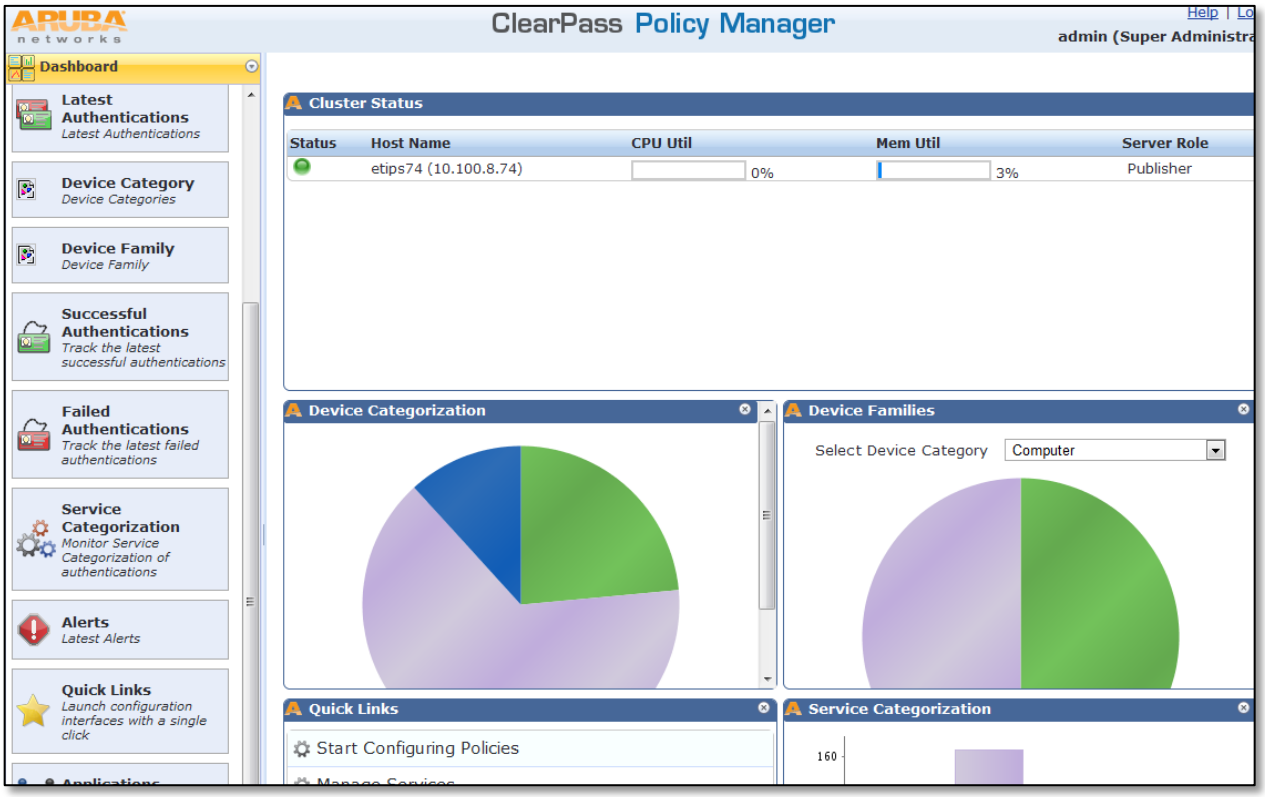

Figure 37 - Dashboard Widgets for profiling

Detailed device distribution and list of endpoints

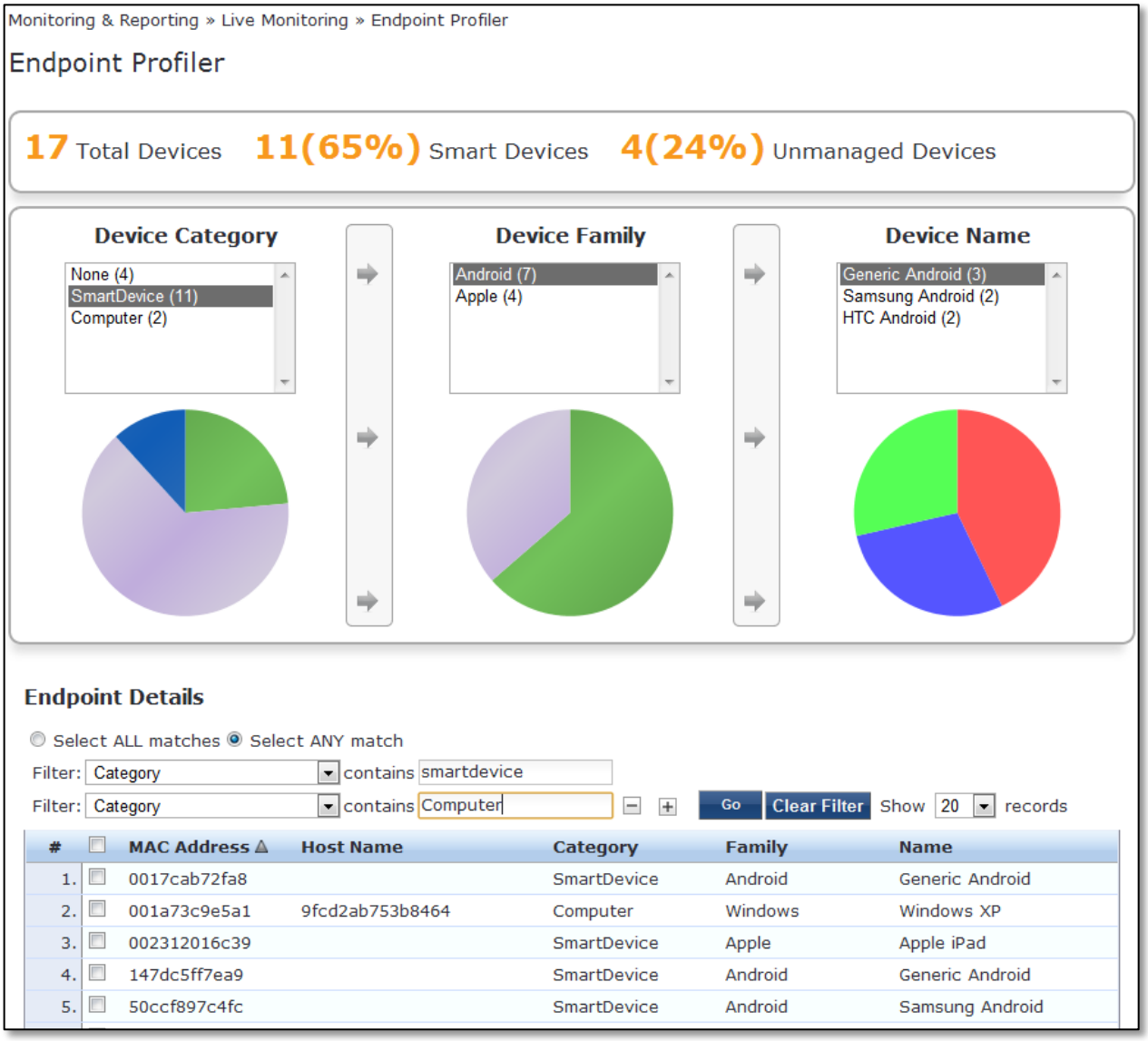

Figure 38 – Summary of Profiler Endpoint Information

Profile details of an endpoint

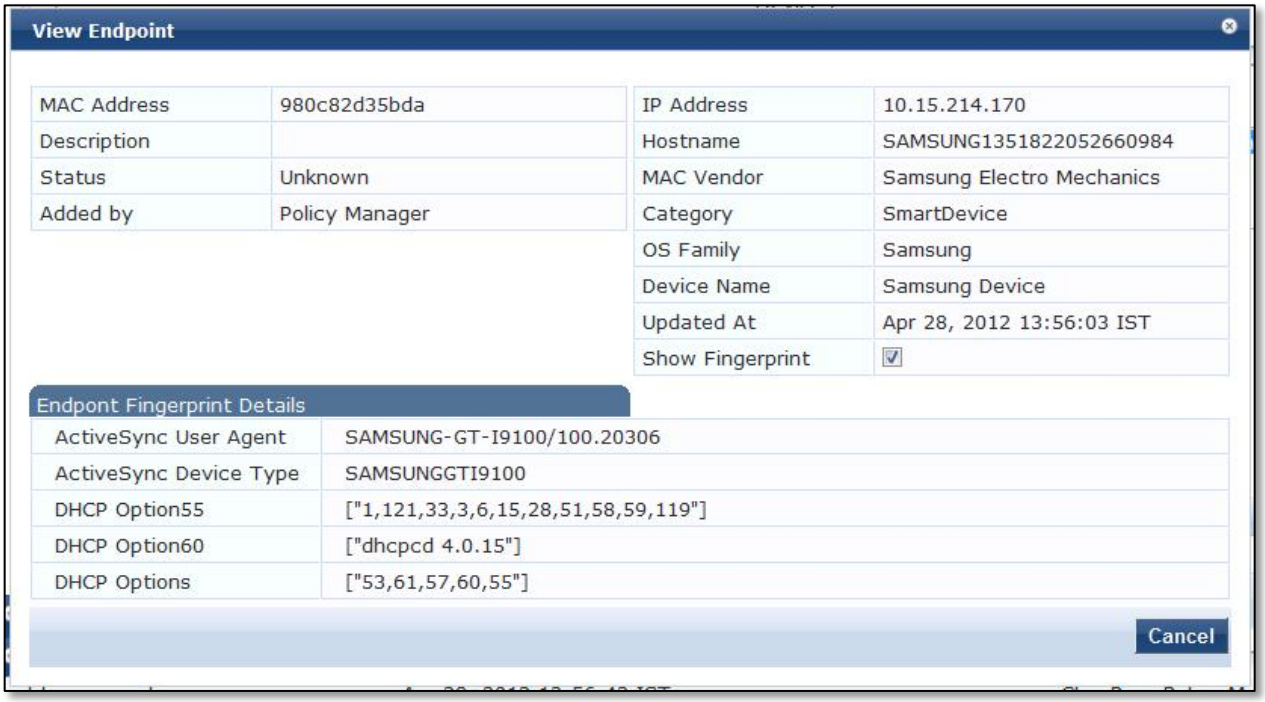

Figure 39 - Detailed Profiler endpoint information

Search endpoint profiles based on category/family/name, etc.

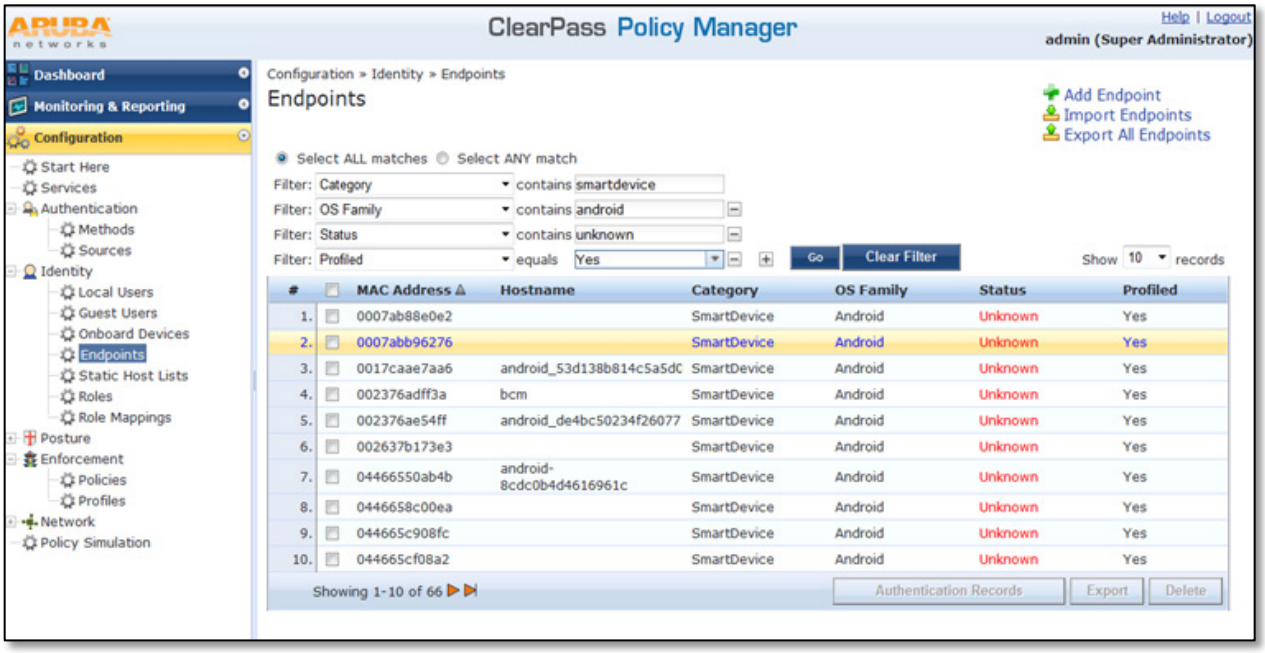

Figure 40 - Complex search of endpoint DB based upon Profiler attributes

Profile APIs

Profile exposes a set of REST APIs to receive endpoint attributes and to provide results of profiling. Basic HTTP authentication using CPPM admin user/passwords are required for the APIs. Third-party products can easily integrate with ClearPass Profile by writing to these APIs.

Post endpoint attributes for profiling

Attributes for a single or multiple endpoints can be POSTed to the following URL; this triggers profiling. MAC or IP address has to be present as the key. Other attributes are optional. If IP address is used as the key, Profile should have received MAC-IP binding from other sources such as DHCP. If ***device***:{*category*, *family*, *name*} is posted, profiler will ignore other inputs and considers this as authentic profile.

- URL: https://{host}/async_netd/deviceprofiler/endpoints
- Method: POST
- Content-Type: application/json
- Input: Single or list of endpoint attributes

```
{
```

```
  mac:
  ip:
  dhcp : {
     option55:
     option60:
     options:
  }
  hostname:
  active_sync : {
    device_type:
    user_agent:
  }
  host: {
    os_type:
    user_agent:
  }
  snmp: {
    sys_descr:
    device_type:
    cdp_cache_platform:
  }
  device: {
       category:
       family:
       name:
  }
 }
```

Output:

- 200 OK on success
- 400 Bad Request - If input data is incorrect.
- 500 Internal Error - on service internal errors

Get endpoint by MAC or IP address

- URL: https://device-profiler/async_netd/deviceprofiler/endpoints/{mac/ip}
- Method: GET
- Output:
- 200 OK - Success with json encoded endpoint details

```
{
```

```
  ip:                 => endpoint ipaddress
  hostname:           => endpoint hostname
  device_category : , => Computer, SmartDevice, Printer etc
  device_family: ,    => Android, Apple, Windows etc
  device_name: ,      => Samsung Android, Apple iPad etc
  added_at: ,         => as unix timestamp in seconds
  updated_at: ,       => as unix timestamp in seconds
}
```

- 404 Not Found - if endpoint with given MAC or IP address does not exist.
- 500 Internal Error - on service internal errors