Reconciling a spreadhseet from Cisco

- Pre-Release Only
- Introduction
- Running the Script
- What is imported?
- Options
- Errors and Warnings (oh my!)
- Sample Spreadsheet Columns
- Mapping the spreadsheet to Open-AudIT
- New Database Columns
- Discovery and your Imported Devices
- Sample Queries
    - Devices Not Discovered
    - Devices Discovered Out of Contract
    - Discovered Inventory Report
    - Serial Numbers Not Found
    - Devices Without Serial Numbers

Pre-Release Only

As at 2020-08-13, this is currently in selective pre-release to supported customers only. Please contact Opmantek to be involved.

All of the below is subject to change in released versions of Open-AudIT, however the functionality will remain the same.

We heavily suggest you read this entire page before running **any** commands.

Introduction

So you have a Cisco supplied spreadsheet in .xlsb format that contains lots of great information like serial numbers, date of purchase, warranty details and more and you want to get that information into Open-AudIT for easy reporting. Say no more...

We are developing a script to just this very thing. This is a perl script for Linux Open-AudIT servers only.

The information to be consumed relies on the Cisco Ready SMARTNET reports that can be requested from Cisco if you have SMARTNET contracts with them.

Running the Script

First thing to do is to convert the spreadsheet to something workable. That's a CSV. Open the spreadsheet in your program of choice and save the relevant tab (usually named "Powered by Cisco Ready") as a CSV. Copy that file to your Open-AudIT server in to the /usr/local/omk/bin directory. Name it cisco.csv.

Next, log on to your Open-AudIT server and change into the directory /usr/local/omk/bin

Now run the script. Done. It's that easy.

```
cd /usr/local/omk/bin
./import_cisco_into_open-audit.pl
```

What is imported?

But which rows are imported? We have a few rules around this, namely:

- Product Type must equal CHASSIS
- A Serial Number must be present
- Covered Line Status must be either ACTIVE, EXPIRED or SIGNED
- The Last Date of Support must be after the configured value of 2019-01-01 (see below).

You'll not see much by default until it's complete. When it has done so, you'll see a summary like below.

```
mark@desktop:/usr/local/opmojo/bin$ ./import_cisco_into_open-audit.pl
Start Time: Thu Aug 13 14:13:11 2020
End Time: Thu Aug 13 14:22:05 2020
Duration: 534 seconds
Location Count: 32
Device Count: 896
mark@desktop:/usr/local/opmojo/bin$
```

Options

We have some options that you can set by directly editing the script. These *might* be settable on the command line in future. They are below.

name	default value	what it does
conf_file	/usr/local/omk/conf /opCommon.nmis	The full path to the Opmantek configuration file. Used to retrieve the UUID.
debug	0	0 - Summary output only 1 - Commands run 2 - Command outputs (usually file contents). A **lot** of information is dumped, use for troubleshooting only.
filename	cisco.csv	The filename to be imported. If it's not in the default directory, the full path including name is OK to use.
last_date_of_support	2019-01-01	The date in YYYY-MM-DD format. Devices with a column value for 'Last Date of Support' that's older than this date will not be imported.
user	admin	The name of the Open-AudIT user performing the action. 'admin' is the default user and fine to use.

Errors and Warnings (oh my!)

You may see lines beginning with DEBUG, INFO, WARNING or ERROR. Generally only ERROR lines are bad and will immediately halt the script execution.

Sample Spreadsheet Columns

Below are the columns from our current Cisco spreadsheet. If your order of these columns differs, it shouldn't matter, however your spreadsheet should contain all the columns used in the mappings below.

The dates we are assuming have a format of D/M/YYYY. These will be reformatted to YYYY-MM-DD for Open-AudIT.

Serial Number/PAK number	Coverage	Covered Line Status	Business Entity	Sub Business Entity	Product Family	Product ID	Product Description	Asset Type	Product Type	Item Quantity	Covered Line Start Date	Covered Line End Date	Covered Line End Date FY-FQ	Contract Type	Service Brand Code	Contract Number	Subscription Reference ID	Ship Date	End of Product Sale Date	End of Software Maintenance Date	Last Date of Support	LDOS FY-FQ	End Of Life Product Bulletin	Warranty Type	Warranty End Date	Install Site Customer Registry GU Name	Install Site Customer Registry Party Name	Install Site Customer Registry Party ID	Install Site Name	Install Site ID	Install Site Address 1	Install Site City	Install Site State	Install Site Country	Install Site Postal Code	Product Bill-to ID	Product Bill-to Partner Name	Product Partner BEGEO ID	POSP Partner BEGEO ID	POSP Partner BEGEO Name	Service Bill-to ID	Service Bill-to Partner Name	Service Partner BEGEO ID	Product List Price $	Default Service List Price $	Default Service Level	Existing Coverage Level List Price $	Instance ID	Patch Installed

Product Description	description
Last Date of Support	end_of_life
End of Product Sale Date	end_of_production (new, see below)
Covered Line End Date	end_of_service
Serial Number / PAK number	hostname
End of Software Maintenance Date	maintenance_expires (new, see below)
Product ID	model
Install Site Customer Registry GU Name	owner
Product List Price $	purchase_amount
Covered Line Start Date	purchase_date
Contract Number	purchase_service_contract_number
Product Bill-to Partner Name	purchase_vendor
Serial Number / PAK number	serial
Warranty End Date	warranty_expires
Covered Line Status	warranty_status (new, see below)

In addition to the above we set a few other fields as below.

Field	Set to
last_seen_by	nmap - so we can override this by discovery, audit or the GUI.
location_id	Calculated by inserting any missing Locations as below, then associating to the device.
org_id	1 - the default Org.
type	By default, set to 'network device'. If one of the below matches, set.

```
if ($line->{'Business Entity'} eq 'Collaboration' and CORE::index($line->{'Product Description'},
'Phone') != -1) { $device->{'system'}->{'type'} = 'voip phone'; }

if ($line->{'Business Entity'} eq 'Collaboration' and CORE::index($line->{'Product Description'},
'Camera') != -1) { $device->{'system'}->{'type'} = 'webcam'; }

if ($line->{'Business Entity'} eq 'Computing Systems') { $device->{'system'}->{'type'} = 'computer'; }

if ($line->{'Business Entity'} eq 'Data Center Networking') { $device->{'system'}->{'type'} =
'switch'; }

if ($line->{'Business Entity'} eq 'Enterprise Routing') { $device->{'system'}->{'type'} = 'router'; }

if ($line->{'Business Entity'} eq 'Enterprise Switching') { $device->{'system'}->{'type'} = 'switch'; }

if ($line->{'Business Entity'} eq 'IOT') { $device->{'system'}->{'type'} = 'network device'; }

if ($line->{'Business Entity'} eq 'Other') { $device->{'system'}->{'type'} = 'network device'; }

if ($line->{'Business Entity'} eq 'Security') { $device->{'system'}->{'type'} = 'firewall'; }

if ($line->{'Business Entity'} eq 'Service Provider Routing') { $device->{'system'}->{'type'} =
'router'; }

if ($line->{'Business Entity'} eq 'Wireless') { $device->{'system'}->{'type'} = 'wap'; }
```

We also import the list of Locations mapping the fields as below. If already present, we leave the Location as is. We test for this using the spreadsheet value for Install Site ID.

Cisco Spreadsheet	Open-AudIT
Install Site Address 1	address
Install Site City	city
Install Site Country	country
Install Site ID	external_ident
Install Site Name	name
1 (the default Org)	org_id
Install Site Postal Code	postcode
Install Site State	state

New Database Columns

As at Open-AudIT 3.4.0 we need three extra columns to populate. You can add these columns as below by running these commands on the command line of your Open-AudIT server.

```
mysql -u openaudit -popenauditpassword openaudit -e "ALTER TABLE system ADD end_of_production date NOT NULL
DEFAULT '2000-01-01' AFTER end_of_service;"

mysql -u openaudit -popenauditpassword openaudit -e "ALTER TABLE system ADD maintenance_expires date NOT NULL
DEFAULT '2000-01-01' AFTER warranty_type;"

mysql -u openaudit -popenauditpassword openaudit -e "ALTER TABLE system ADD warranty_status VARCHAR(100) NOT
NULL DEFAULT '' AFTER warranty_type;"
```

NOTE - As at Open-AudIT 3.4.0, these attributes will **not** appear in the GUI. The template to enable this will ship in a future release.

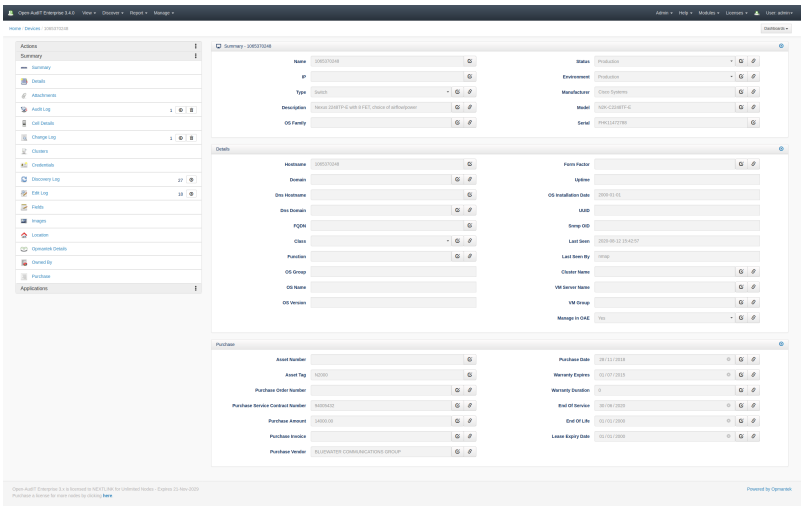
Discovery and your Imported Devices

So you may already have some of these devices in Open-AudIT, you may not. How does Open-AudIT know what to do here? It all depends on the serial number. The spreadsheet doesn't contain the usual items we need to match devices (think hostname, ip, type, UUID, etc), but it **does** contain the device serial number. So you'll need to ensure your configuration item named 'match_serial' is set to 'y' (which is the default). As long as that is set, you can import and run discoveries in whichever order you like. Obviously discovery must be working and able to retrieve the serial from the device to perform the match. If you don't have working discovery (usually SNMP based for Cisco network equipment), you'll likely end up with duplicate devices - you have been warned! Just remember, Bulk Edit is your friend.

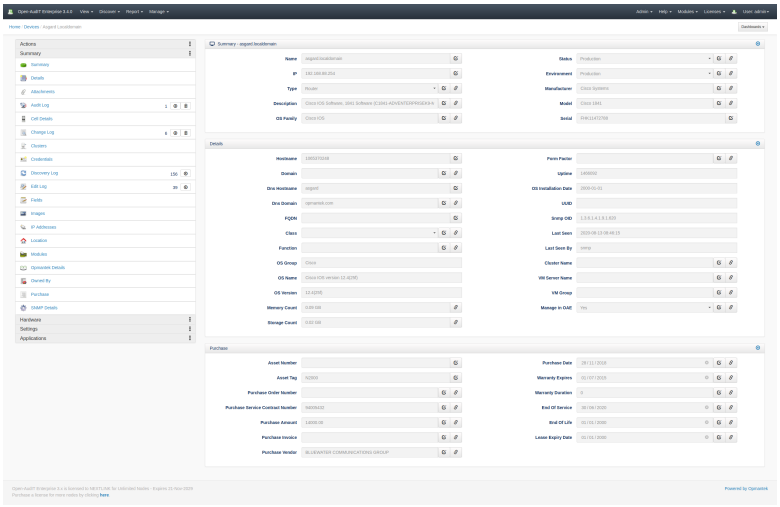
If you import a device, then run a discovery some fields may be overwritten. Model is likely the first one you'll notice. This is expected. Two screenshots are below of a device after import, then after discovery.

Note that this was an example only as the import was a Cisco Nexus and the discovered device was a Cisco 1841 router. I manually changed the serial to cause them to match.

After import.

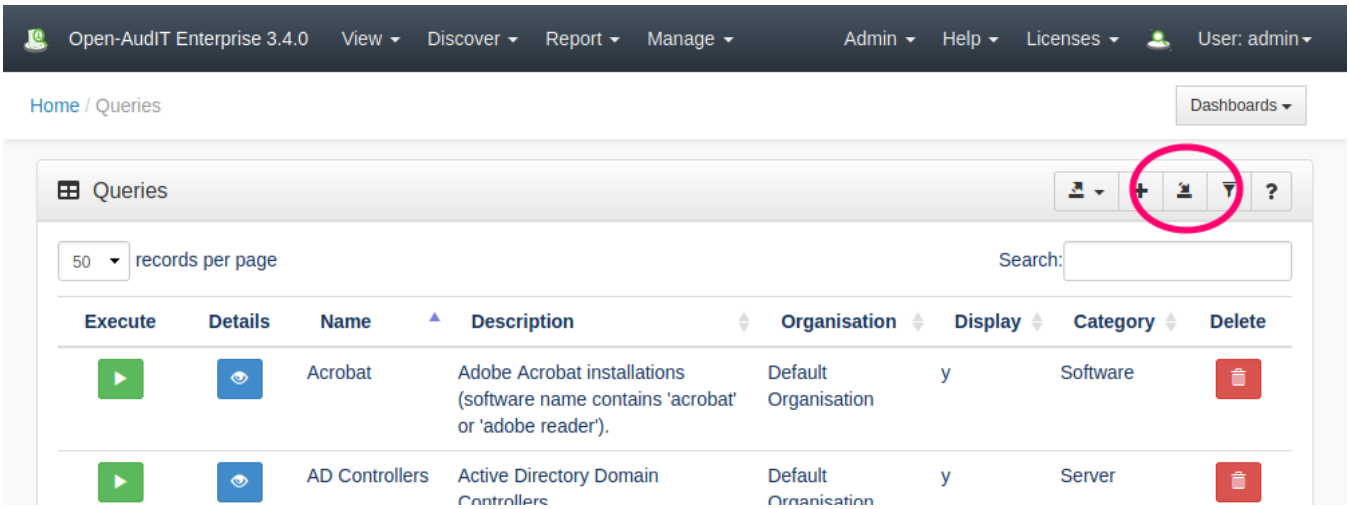


After Discovery.



Sample Queries

Below are some sample queries we may refine and include in newer releases of Open-AudIT. To import them, copy the JSON contents and go to menu Manage Queries List Queries and click the Import button in the top right of the table panel.



Devices Not Discovered

```
{
   "name" : "Devices Not Discovered",
   "sql" : "SELECT system.id AS `system.id`, system.type AS `system.type`, system.manufacturer AS `system.
manufacturer`, system.model AS `system.model`, system.serial AS `system.serial`, system.owner AS `system.
owner`, locations.name AS `locations.name`, locations.id AS `locations.id` FROM `system` LEFT JOIN `locations`
ON system.location_id = locations.id WHERE @filter AND system.ip = ''",
   "description" : "Devices Imported but Never Seen by Discovery",
   "menu_display" : "y",
   "org_id" : 1,
   "menu_category" : "Device",
   "link" : ""
}
```

Devices Discovered Out of Contract

```
{
   "org_id" : 1,
   "menu_category" : "Device",
   "menu_display" : "y",
   "link" : "",
   "sql" : "SELECT system.id AS `system.id`, system.type AS `system.type`, system.ip AS `system.ip`, system.
manufacturer AS `system.manufacturer`, system.model AS `system.model`, system.serial AS `system.serial`, DATE
(system.last_seen) AS `system.last_seen`, system.end_of_service AS `system.end_of_service`, system.owner AS
`system.owner`, locations.name AS `locations.name`, locations.id AS `locations.id` FROM `system` LEFT JOIN
`locations` ON system.location_id = locations.id WHERE @filter AND system.ip != '' AND DATE(system.last_seen) >
system.end_of_service",
   "name" : "Devices Discovered Out of Contract",
   "description" : "Devices that have been seen by discovery but whose end_of_service has expired."
}
```

Discovered Inventory Report

```
{
   "link" : "",
   "org_id" : 1,
   "menu_category" : "Device",
   "menu_display" : "y",
   "description" : "Inventory details for discovered devices",
   "sql" : "SELECT system.id AS `system.id`, system.type AS `system.type`, system.ip AS `system.ip`, system.
manufacturer AS `system.manufacturer`, system.model AS `system.model`, system.serial AS `system.serial`, DATE
(system.last_seen) AS `system.last_seen`, system.end_of_service AS `system.end_of_service`, system.end_of_life
AS `system.end_of_life`, system.end_of_production AS `system.end_of_production`, system.maintenance_expires AS
`system.maintenance_expires`, system.warranty_expires AS `system.warranty_expires`, system.owner AS `system.
owner`, locations.name AS `locations.name`, locations.id AS `locations.id` FROM `system` LEFT JOIN `locations`
ON system.location_id = locations.id WHERE @filter AND system.ip != ''",
   "name" : "Discovered Inventory Report"
}
```

Serial Numbers Not Found

```
{
   "link" : "",
   "org_id" : 1,
   "menu_category" : "Device",
   "menu_display" : "y",
   "description" : "Devices imported with a serial number that have not been discovered",
   "sql" : "SELECT system.id AS `system.id`, system.type AS `system.type`, system.manufacturer AS `system.
manufacturer`, system.model AS `system.model`, system.serial AS `system.serial`, system.end_of_service AS
`system.end_of_service`, system.end_of_life AS `system.end_of_life`, system.end_of_production AS `system.
end_of_production`, system.maintenance_expires AS `system.maintenance_expires`, system.warranty_expires AS
`system.warranty_expires`, system.owner AS `system.owner`, locations.name AS `locations.name`, locations.id AS
`locations.id` FROM `system` LEFT JOIN `locations` ON system.location_id = locations.id WHERE @filter AND
system.ip = ''",
   "name" : "Serial Numbers Not Found"
}
```

Devices Without Serial Numbers

```
{
   "link" : "",
   "menu_category" : "Device",
   "org_id" : 1,
   "menu_display" : "y",
   "description" : "Devices discovered that have no reported serial",
   "sql" : "SELECT system.id AS `system.id`, system.icon AS `system.icon`, system.type AS `system.type`, system.
ip AS `system.ip`, system.manufacturer AS `system.manufacturer`, system.model AS `system.model`, system.
identification AS `identification`, locations.name AS `locations.name`, locations.id AS `locations.id` FROM
`system` LEFT JOIN `locations` ON system.location_id = locations.id WHERE @filter AND system.serial = ''",
   "name" : "Devices Without Serial Numbers"
```

Script