

Rules

Introduction

With the release of Open-Audit 3.2.0 we have introduced a new concept called Rules. Rules are created and run against a device when the device is discovered or an audit result is processed. Rules can be used to set a device attribute based on other attributes.

NOTE - At present we cannot delete a rule input or output that contains a /. This is because the framework is parsing the / as part of the URL and returning a 404, even before our code runs. The work-around for this is to delete the Rule itself, then recreate the inputs and outputs as required. Fortunately inputs and outputs that contain a / are rare (indeed, none exist by default).

How Does it Work?

Each time a device is discovered or an audit result is processed, all rules are retrieved from the database and run against the attributes of the specific device. Rules run against one device at a time - there is no facility to say "Run the rules against all devices" or "Run the rules against these devices". An individual rule will test one or more attributes of the device and if they match the rule, the result will be applied. Several attributes can be tested. Several attributes can be set. Think of this as an If This, Then That system for Open-Audit.

UPDATE - With the release of 3.2.2, we no longer store ~100,000 rules in the database. This was fine on my test device (a core i7, 16GB memory and Samsung 860 NVMe), but in practice was causing customers servers to choke.

As per the [Release Notes](#) for 3.2.2 -

So, that was a ride... In testing our new Rules feature worked a treat. In practice, not so much. Most servers (ie, not mine) can't cope with loading the rule set, even if we break it down to smaller chunks, when processing multiple devices. What to do? What to do? Well we've taken a small step back. Rules still exist as a feature, and they still work a treat. But instead of inserting 100,000 Rules into the database, we've split them up into four distinct files and implemented them as code only. Hence, no loading all 100,000 Rules, decoding JSON and running them against a device. Now we just load the files and run the statements. Much, much faster and more memory efficient. No load on MySQL, and hence the CPU also drops. No populating a massive record set and hence the memory drops. The not so good thing - these are no longer editable in the GUI. But it's not the end of the world. You can still make Rules as you see fit and they will be run after the "default" rules (those in code), hence you can override the "default" rules. So we don't lose much, but we gain a LOT of performance. We also added a few new Rules for Mac Models.

For those curious, the "new" files that replace the Rules are:

File	Description
/open-audit/code_igniter/application/helpers/mac_helper	Matches MAC addresses to manufacturers.
/open-audit/code_igniter/application/helpers/mac_model_helper	Matches Apple manufacturer codes to models (stored in system.manufacturer_code).
/open-audit/code_igniter/application/helpers/snmp_model_helper	Matches the device's SNMP OID to a model and type.
/open-audit/code_igniter/application/helpers/snmp_model_helper	Matches the devices's SNMP OID to the manufacturer.

And of course you are free to add or modify the rules as you see fit. If you have a device with an SNMP OID that doesn't match a model already in the file - now you can add it easily. No more waiting for us to provide a patch and add it to the code base for you.

When you create a Rules entry, you will need to provide a name and a list of inputs and outputs. The inputs and outputs are stored as JSON arrays within the database.

Inputs have a table and attribute, an operator and a value. When executing the condition, it works thus: If \$table . \$attribute \$operator \$value then apply the outputs.

A contrived example rule to match a type and set the model is below.

```
INPUTS
table = system
attribute = type
operator = eq
value = computer

OUTPUTS
table = system
attribute = model
value = My Model
value_type = string
```

Operators in Inputs can have the following values.

Name	Result
eq	Equals
ne	Does Not Equal
gt	Greater Than
ge	Greater Than or Equals
lt	Less Than
le	Less Than or Equals
st	Starts With
li	Like
nl	Not Like
in	In the (comma seperated) list
ni	Not in the (comma seperated) list

Value Types in Outputs can have the following values.

Name	Description
string	a String
integer	an Integer
timestamp	A timestamp. If the value is set, that timestamp value will be used. If the value is not set, the current timestamp will be used.

When the rules run in discovery, any matching rules will appear in the discovery log. See below for an example.

```
Hit on snmp_enterprise_id 9 eq 9
Hit on manufacturer is empty
Command: Rule Match - SNMP Enterprise Number for ciscoSystems, ID: 10
Output: {"manufacturer":"Cisco Systems","snmp_enterprise_name":"ciscoSystems"}
```

and anohter

```
Hit on snmp_oid 1.3.6.1.4.1.9.1.620 eq 1.3.6.1.4.1.9.1.620
Command: Rule Match - SNMP OID match, ID: 135661
Output: {"model":"Cisco 1841","type":"router"}
```

Create Rules Entries

Rules can be created just like any other item. Menu Manage Rules Create.

You can add and remove inputs and outputs as required.

Open-Audit Enterprise 3.3.2 View Discover Report Manage Admin Help Modules Licenses User: rms

Home Rules Dashboards

Rules

Name

Org ID

Description

Weight

If

system

asset_number

+

os

Then

system

asset_number

+

String

Submit

About

Rules are used to apply attribute details to a device, depending on it's other attributes

For more detailed information, check the [Open-Audit Knowledge Base](#).

Viewing Rules Details

As usual, go to menu Manage Rules List Rules.

Because there are so many rules (near one hundred thousand), paging through them is unrealistic. We still retrieve the default number of entries as per the configuration item, however there is a search box at the top of the panel. Use this to search through the name, description, inputs and outputs to refine the list and find what you're looking for. There is also a button on the panel header that will show you all the rules you have created or edited. See below.

Open-Audit Enterprise 3.3.2 View Discover Report Manage Admin Help Modules Licenses User: rms

Home Rules Dashboards

Rules

Search Submit

10 records per page

Search

Details	Name	Organisation	Description	Input	Output	Edited By	Edited Date	Delete
	Android devices	Default	Set attributes based hostname contains android.	system.name & android	Set os_group to Android Set os_family to Android Set os_name to Android Set type to android	system	2001-01-01 00:00:00	
	Apple IOS devices	Default	Set attributes based on Nmap port 62078.	nmap.port eq 62078	Set manufacturer to Apple, Inc. Set os_group to Apple IOS Set os_family to Apple IOS Set os_name to Apple IOS	system	2001-01-01 00:00:00	
	Apple iPad devices	Default	Set attributes based on Nmap port 62078 and hostname contains ipad.	system.name & ipod nmap.port eq 62078	Set model to Apple iPad Set type to ipod	system	2001-01-01 00:00:00	
	Apple iPhone devices	Default	Set attributes based on Nmap port 62078 and hostname contains iphone.	system.name & iphone nmap.port eq 62078	Set model to Apple iPhone Set type to iphone	system	2001-01-01 00:00:00	
	Apple iPod devices	Default	Set attributes based on Nmap port 62078 and hostname contains ipod.	system.name & ipod nmap.port eq 62078	Set model to Apple iPod Set type to ipod	system	2001-01-01 00:00:00	
	Apresia switches default	Default	Set type as a default based on OID.	system.snmp_oid st 1.3.6.1.4.1.278.1.27 system.model eq	Set type to switch Set model to Apresia	system	2001-01-01 00:00:00	
	Apresia WAPs default	Default	Set type as a default based on OID.	system.snmp_oid st 1.3.6.1.4.1.278.1.19 system.model eq	Set type to wap Set model to Apresia	system	2001-01-01 00:00:00	
	Coragon Wireless Link default	Default	Set type as a default based on manufacturer.	system.manufacturer & Coragon system.type eq	Set type to wireless link	system	2001-01-01 00:00:00	
	Class based on Form Factor and OS (Physical Server)	Default	Set the class based on the form factor and OS.	system.form_factor ne Virtual system.os_group ne Windows system.os_group ne system.class eq system.manufacturer ne Ubiquiti system.type eq computer	Set class to server	system	2001-01-01 00:00:00	
	Class based on Form Factor and OS (Physical Windows Server)	Default	Set the class based on the form factor and OS.	system.form_factor ne Virtual system.os_group eq Windows system.os_name & Server system.os_group ne system.class eq	Set class to server	system	2001-01-01 00:00:00	
	Class based on Form Factor and OS (Virtual Server)	Default	Set the class based on the form factor and OS.	system.form_factor eq Virtual system.os_group ne Windows system.os_group ne system.class eq	Set class to virtual server	system	2001-01-01 00:00:00	

Database Schema

The database schema can be found in the application is the user has database::read permission by going to menu: Admin -> Database -> List Tables, then clicking on the details button for the table.

API / Web Access

You can access the collection using the normal Open-Audit JSON based API. Just like any other collection. Please see [The Open-Audit API documentation](#) for further details.

Default Items

Shipped are a set of default items. These can be found by going to menu: [Help](#) [Defaults](#) [Rules](#).