

Step 4 - Policy Guide

Contents

1	Introduction	1
2	What is a Policy?	2
2.1	Basic Policy Anatomy	2
2.2	Organizations, Applications and Inheritance	3
2.3	Summary	4
3	Importing Policies	6
3.1	Sonatype Example Policies	6
3.2	Importing a Policy to an Organization	8
3.3	Importing a Policy to an Application	10
3.4	Summary	11
4	Conclusion	12

List of Figures

3.1	Organization View with Import Button	9
3.2	Import Policy Dialog	10

[Return to the Nine Steps Main Page](#)

Chapter 1

Introduction

Making sure you can get Sonatype CLM setup and functioning quickly for your team and/or business is at the core of the Nine Steps for Open Source Governance. Up to this point, we've only notionally mentioned policy, referring to it simply as a set of rules. In this guide, we'll dig a little deeper, and introduce some of the theory as well as practice behind policy. We won't make you create or manage policies just yet. Instead we'll provide you with a sample set of policies, and provide instruction on how to import that into the organization and application you created in Step Four.

This document was published on 2014-05-22.

**Important**

This guide assumes that you have a fully installed and running Sonatype CLM Server available. It also assumes you've completed the first four steps of Ten Steps to CLM Success. Most notably, you will need to have created at least one organization and application.

Chapter 2

What is a Policy?

When we talk about policy within the paradigm of Sonatype CLM, we refer to it as a way to identify and reduce risk through a concise set of rules for component usage. These rules can be used to assist at every step of the component and development lifecycle, and can be customized for specific applications and organizations. In general, policy, within the context of Sonatype CLM, is a broad term used to encapsulate:

- Conditions
- Actions

In some ways rules as a description is a bit generic, so let's dig a bit deeper, and look at another concept you are likely familiar with, an "If/Then" statement.

2.1 Basic Policy Anatomy

One of the easiest way to break down the various elements of a policy, at least the most basic parts, is to think of a policy as an "If/Then" statement. That is, a policy simply says that if something happens, then perform a certain action. If a component meets a set of criteria, then take a certain action, or in some cases no action at all.

If it's still a bit fuzzy, an example will probably help. Let's say we have a known rule in our development organization that says if a component used in an application has a security vulnerability, the application can not be released. To do this, we tell our development team to review components before release and if a component has a security issue, we don't promote the release. Congratulations, you have formed, at least in the aether, your first policy.

Now, let's take a slightly closer look, and define the basic policy anatomy. There are actually three key parts to a policy:

Conditions

conditions are the *if* part of the *if-then* statements.

Constraints

a constraint is really just a way to organize multiple conditions (if-then statements). Our example only had one so far. Let's say we decided we wanted to add that if a security issue is found and it has a CVSS of 2 or lower, only a warning should occur, but the release should not be prohibited.

Actions

actions are simply the *then* part of the if-then statement. Basically, what you want to have happen.

The above does a good job of telling us what makes up a policy in Sonatype CLM, but you are likely thinking, not all policies should be the same, I need a way to demonstrate which policies are the most important. We thought that too, and that why in Sonatype CLM, all policies are assigned a *threat level* ranging from zero to ten (0-10). This score is completely subjective and will be unique in your organization.

OK, so now that we've opened up our concept of policy a bit, exposing the inner workings so to speak, the next question you should have is, "Where do we create policies?"

Well, as you likely recall, we can manage policy for both organizations and applications. We've learned a bit about these already, but let's go ahead and have a quick review.

2.2 Organizations, Applications and Inheritance

As a quick overview, the differences between an organization and application are as follows.

Organizations

- Require a name
- Provide an option to attach an icon
- Serve as a way to group applications.

Applications

- Require a name, application ID, and an organization.
- Provide an option to attach an icon.
- Represent a one-to-one relationship (App ID) between an actual application (or project), and Sonatype CLM.

Inheritance

Now, there is one final difference between organizations and applications, and that is inheritance, which is simply the ability for elements from an organization to transfer down to an application.

Ultimately this allows you to avoid micromanagement. For example, let's say there is a policy that says no component can have a security violation score greater than 3. This rule should apply to a number of applications that are all associated to a particular organization. Thanks to inheritance I can create a policy for the organization and all applications will inherit this policy. This also means I won't need to make changes to the policy for each application, rather I only make the change once, at the organization level, and it will affect any application attached to that organization. This policy will also apply to any additional applications I create under this organization in the future.

The important thing here, is to start thinking about policy more holistically. This is even before you begin to create policy in Sonatype CLM, you should think about where your policies will be created, and what applications will share similar policies. If nothing else, start writing out policies you would like to experiment with. Communicate those to your teams and start incorporating a proper feedback loop.

2.3 Summary

This section was all about policy. It's primarily theory, but theory that is quite important for your practical implementation. Sonatype CLM will build on everything discussed here. As a recap, here's what you should walk away with:

- Policy is an aggregation of rules that are basically If/Then statements - your policies
 - Each policy consists of one or many conditions
 - Multiple conditions together form a constraint
-

- When all conditions are met a policy results in the execution of an action
- Applications inherit policies from the organization they are associated to

Chapter 3

Importing Policies

Setting up policies can be quite complex and labor intensive. To make the process easier and give you a head start we have created some sample policies and provide an import feature.

We actually recommend you don't begin using Sonatype CLM by creating a bunch of policies right out of the gate. Instead, we've created a set of policies, which include other policy elements such as labels and license threat groups, that you can import into your Sonatype CLM installation.

Eventually, and there is a very short time between now and eventually, you will need to create, or at least modify, policies. For now, we'll want to focus on populating your organizations and applications with policies provided by Sonatype.

3.1 Sonatype Example Policies

The easiest way to establish policies for your applications is to use one of the policies packages provided by Sonatype. While these are not meant to be a perfect match for every business, they have been created with our extensive experience working with customers and developing policy for our own internal practices.

The policy packages can be downloaded here:

- [Sonatype-Audit-Policy.json](#)
- [Sonatype-Enforcement-Policy.json](#)

Note

The import files are simple JSON files and are only compatible with the latest version of the Sonatype CLM Server. Please review the [Archives to access Downloads for your version of Sonatype CLM](#).

Alternatively you can find them in the documentation archive in the `resources` folder.

Let's take a look at the various policies available.

Audit Policy Package

This audit policy package is an example of managing components for security, licensing, and architectural issues. It also introduces the detection of unknown and patched components used in building your applications. The audit policy package can be used to gather information about the components used to build your applications without warnings and failures occurring in the developer, continuous build, or Nexus environments. This is the perfect policy package to use in order to gather information and understand how policy management will work for your environment, without potentially distracting the people who are building and delivering your applications.

Note

This policy package includes several preset tags. The tags have been used in the Application Matching area for several of the included policies. Policies using the tags will be indicated by a special tag icon. In order to utilize the policies, you must have applied the corresponding tag to your application(s). For more information on tags, please see the [Policy Elements section of our Policy Management Guide](#).

Enforce Policy Package

The enforcement policy package includes the same set of policies as the audit policy package, with the addition of enforcement actions. It also includes suggested enforcement actions based on the severity of issues being detected. If you plan to use the policy package, policy notification actions should be added to notify interested parties of policy violations. The notifications will not overwhelm the inbox as the system tracks which notifications have been sent and will not send duplicate notifications. If you are looking for a good starting point to ensure the components being used in your applications meet the defined policy before being released, you will want to use this policy package.

Note

This policy package includes several preset tags. The tags have been used in the Application Matching area for several of the included policies. Policies using the tags will be indicate by a special tag icon. In order to utilize the policies, you must have applied the corresponding tag to your application(s). For more information on tags, please see the [Policy Elements section of our Policy Management Guide](#).

Note

This policy package includes three preset email addresses for notifications. You will want to open the JSON file and find and replace the following addresses before importing (ProjectLead@changeme.sonatype.com, LicenseTeam@changeme.sonatype.com, SecurityTeam@changeme.sonatype.com). This can be edited with in Sonatype CLM, but will be a more manual process.

3.2 Importing a Policy to an Organization

Once you have acquired a policy file to import, you can follow these steps:

1. Log into your Sonatype CLM server with a user account that has proper permissions to import policy for a specific organization (at least a member of the owner group for the organization would be required).
 2. Next, click the *Organizational Design* icon  to access the **Organizational Design** area.
 3. Click on *Organizations* in the left menu, and then click the organization you wish to import the policy to.
 4. Click the *Import* button in the top right corner of the organization view displayed in Figure 3.1.
 5. Click the *Choose File* button in the **Import Policy** dialog displayed in Figure 3.2 and select the policy JSON file in the file browser.
 6. Click the *Import* button in the **Import Policy** dialog.
 7. Confirm that the list of policies contains the imported policies.
-

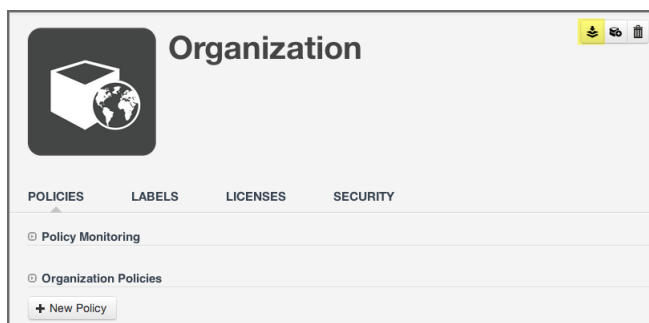


Figure 3.1: Organization View with Import Button

If you are importing to an organization, that already has some policies, labels, license threat groups, and/or tags set up, consider the following rules:

- Existing policies will be deleted during the import procedure.
- Importing policies also includes an import of associated policy elements (labels, license threat groups, and tags). The following logic will be used for Policy Elements:
 - Labels - the CLM server attempts to match labels against existing ones in a case-insensitive manner. This allows for updating the description or color of existing labels, while preserving any triage effort already done to apply these labels to components. If your import contains labels that aren't already present in the system, they will be created.
 - License Threat Groups - the CLM server will delete all existing license threat groups, and then import the new ones.
 - Tags - the CLM Server attempts to match tags against existing ones in a case-insensitive manner. This allows for updating the description or color of existing tags, while preserving any current matching of tags between policies and applications.

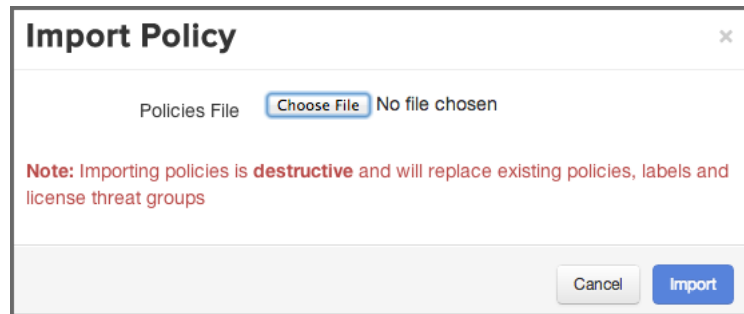


Figure 3.2: Import Policy Dialog

3.3 Importing a Policy to an Application

An application inherits policies from the organization. However it can be useful to have additional policies for fine grained control.

1. Log into your Sonatype CLM server with a user account with an **Administrator** role or as an **Owner** of the application you wish to import policy to.
2. Next, click the *Organizational Design* icon  to access the **Organizational Design** area.
3. Two columns will be displayed on the left. Click on *Applications*, and then click the application you chose to import the policy to.
4. Click the *Import* button in the top right corner of the application view, which is identical to the organization view displayed in Figure 3.1.
5. Click the *Choose File* button in the **Import Policy** dialog displayed in Figure 3.2 and select the policy JSON file in the file browser.
6. Click the *Import* button in the **Import Policy** dialog.
7. Confirm that the list of policies contains the imported policies.

The policy information will be imported, and the following rules will be applied:

- Duplication of organization policies is invalid, so you will not be able to import the same policy file into an organization and then into an application associated to it.

- When a policy is imported, any existing application policies will be deleted and replaced with the imported configuration.
- For label imports, the same logic as during imports at the organization level described in [Section 3.2](#) applies.
- Attempting to import policies that contain tags will cause the entire import to fail.

3.4 Summary

If you are having trouble coming up with your own, custom policies, importing any of our sample policies can be a great way to get started. While this may not be an exact fit, in most situations it provides a good baseline for improving the health of your applications. Better yet, if you want to modify the policies after import, you can do that as well.

Chapter 4

Conclusion

By now, you should have successfully imported a sample policy to an organization and an application. This will prepare you for scanning your applications, as well as managing policy. If you are following along with our Nine Steps for Open Source Governance you can move on to [Step 5 - Scanning Applications](#).
