Contrast Continuous Application Security Plugin

Plugin Information
View Contrast Continuous Application Security on the plugin site for more information.

About

This plugin verifies vulnerability conditions by checking a build's vulnerabilities found against configured filters. The plugin also graphs history of vulnerability detection found during each projects build.

This plugin supports a post build action and a step in the pipeline build process.

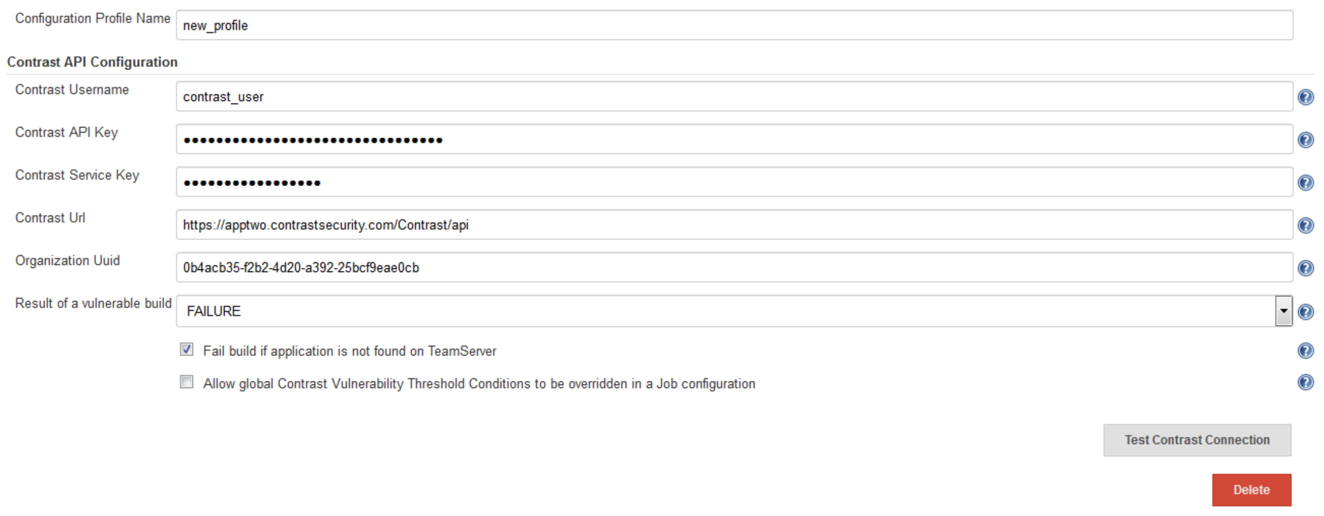
Use the Plugin

You can view the plugin code in Jenkins' Github repository. In the Jenkins dashboard, go to **Manage Jenkins** in the left sidebar, and select the **Configure System** page to find a new Contrast TeamServer profiles section.

Contrast API Settings

Contrast API settings enable the plugin to connect to Contrast and query for results. The plugin leverages these result to authenticate to Contrast and make API calls in post-build actions. Among the following requirements, you'll need a unique profile name to identify your configuration and use it in a specific job.

Parameter	Description	Since
Contrast Username	Username/email for your account in Contrast	
Contrast API Key	Log in to your Teamserver account and go to **Your Account**. Look under **YOUR KEYS**.	
Contrast Service Key	Log in to your Teamserver account and go to **Your Account**. Look under **YOUR KEYS**.	
Contrast URL	API URL to your Contrast instance Use *https://app.contrastsecurity.com/Contrast/api* if you're a SaaS customer; all others use the URL of your Contrast UI (e.g., *https://contrastserver/Contrast/api*).	
Organization UUID	Organization UUID of the configured user found in **Organization Settings**	
ignoreContrastFindings	Jenkins boolean build parameter. If set to true, builds will not be failed when Vulnerability Threshold Conditions are not met.	2.3
Result of a vulnerable build	Contrast TeamServer profile configuration parameter allowing to choose the result of a build that does not meet the Vulnerability Threshold Conditions.	2.3
Fail build if application is not found on TeamServer	This option allows to fail a build if the **application is not found** in the Contrast application.	2.4
Allow global Contrast Vulnerability Threshold Conditions to be overridden in a Job configuration	Choose if global threshold conditions can be overridden in post-build actions. (See the **Global threshold conditions** section for more details.)	2.5



Configuration Profile Name	new_profile

Contrast API Configuration

Contrast Username	contrast_user
Contrast API Key	••••••••••••••••••••••••••••••••
Contrast Service Key	•••••••••••••••••
Contrast Url	https://apptwo.contrastsecurity.com/Contrast/api
Organization Uuid	0b4acb35-f2b2-4d20-a392-25bcf9eae0cb
Result of a vulnerable build	FAILURE

[✓] Fail build if application is not found on TeamServer

[ ] Allow global Contrast Vulnerability Threshold Conditions to be overridden in a Job configuration

Test the connection

When you add a Contrast profile, use the validation button to test your connection and make sure that all the fields are accurate. Contrast prompts you if the test is successful or gives an error message if it fails.

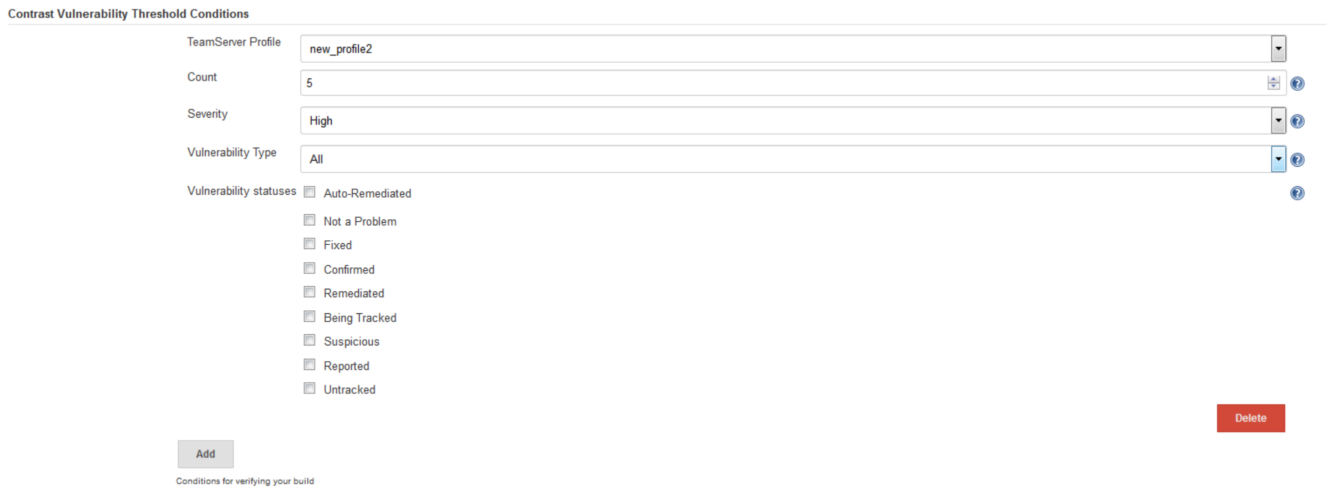
Global threshold conditions

Once a connection is made, complete the following fields for **Contrast Vulnerability Threshold Conditions**.

- Select a **Profile** from the dropdown.
- Add a **Count**. The count is exclusive; if you set a count for "5", it fails on six or more vulnerabilities. This field is **required**.
- Choose a **Severity** from the options in the dropdown menu (Note, Low, Medium, High or Critical). The plugin sets a filter in the API call for all vulnerabilities greater than or equal to this field. This field is recommended to reduce your results, but not required.
- Choose a **Vulnerability Type** (rule name) from the dropdown menu. If you specify a single rule for which to filter, the plugin checks for the number of vulnerabilities with the rule type and compares it to the count. This field is recommended to reduce your results, but not required.
- Choose from the list of **Vulnerability Statuses**. Statues aren't required, but can be helpful if you want to exclude vulnerabilities with certain statuses - for example, "Not a Problem" - from the results. If you don't select any statuses, the plugin won't filter vulnerabilities by statuses.

You can add as many rules as you like. The plugin fails on the **first** bad condition and tells you on which condition it failed.

> ***Note***: *Even if your build succeeds, the plugin fails the overall build if the test finds a bad condition.*



Contrast Vulnerability Threshold Conditions

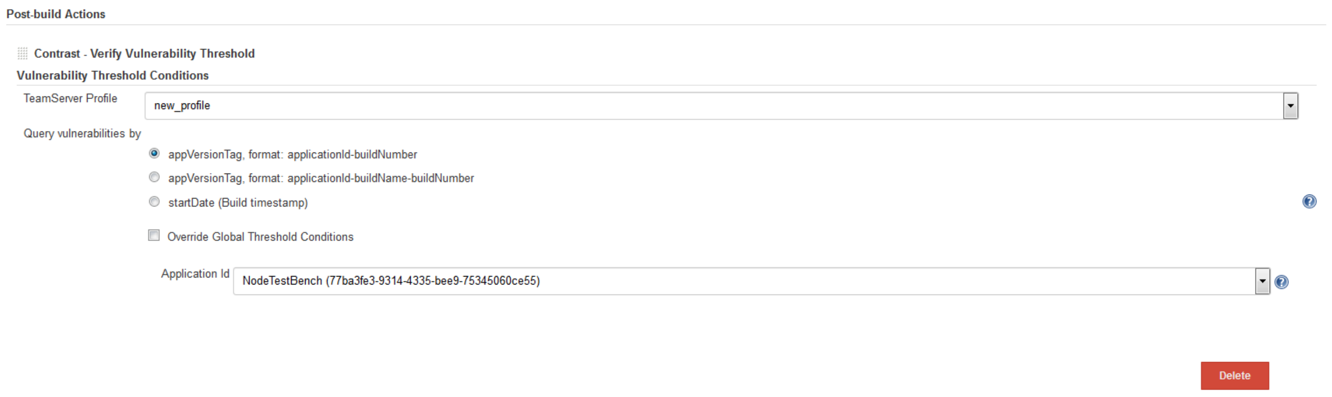
TeamServer Profile	new_profile2
Count	5
Severity	High
Vulnerability Type	All

Vulnerability statuses
[ ] Auto-Remediated
[ ] Not a Problem
[ ] Fixed
[ ] Confirmed
[ ] Remediated
[ ] Being Tracked
[ ] Suspicious
[ ] Reported
[ ] Untracked

Threshold conditions in a post-build action

Complete the following fields for **Post-Build Actions**.

- Select a **Profile** from the dropdown.
- Select **Query vulnerabilities by**. By default, the plugin uses the first option: "appVersionTag, format: applicationId-buildNumber".
- If the profile is configured to allow the global threshold conditions to be overridden, you can choose to do so.
- Select the **Application Id** from the dropdown menu. This field is **required**.
- If you chose to override the global threshold conditions, fill in the rest of the fields, including **Count**,**Severity**, **Vulnerability Type**, and **Vulnerabilit y Statuses** similarly to the global threshold conditions described above.


Post-build Actions

Contrast - Verify Vulnerability Threshold

Vulnerability Threshold Conditions

TeamServer Profile: new_profile

Query vulnerabilities by

- [✓ appVersionTag, format: applicationId-buildNumber]
- appVersionTag, format: applicationId-buildName-buildNumber
- startDate (Build timestamp)

Override Global Threshold Conditions

Application Id: NodeTestBench (77ba3fe3-9314-4335-bee9-75345060ce55)

Threshold conditions in a Pipeline step

When you add a Pipeline step with the name `contrastVerification`, it follows the same principles as the post-build action but in a newer format for Jenkins 2.0 improvements.

Pipeline configuration:

```
contrastVerification applicationId: '1e6ad9c6-89d4-4f06-bdf6-92c569ec89de', count: 1, profile: 'new-profile',
queryBy: 3, rule: 'cache-controls-missing', severity: 'High'
```

Test for Vulnerabilities

For the Jenkins plugin to get accurate information, you must add a unique identifier built from the Jenkins CI configuration as an agent property. The corresponding property for the Java agent is `contrast.override.appversion`. For example, when starting Contrast agent add the following property: "-Dcontrast.appname=${applicationName}".

The plugin can use either the unique identifier `appVersionTag` or the `startDate` to filter vulnerabilities and check conditions. You can change the format used by the plugin to create `appVersionTag` or set the plugin to use `startDate` using `queryBy` pipeline parameter. Three options are available:

- appVersionTag, format: `applicationId-${BUILD_NUMBER}` (default)
- appVersionTag, format: `applicationId-${JOB_NAME}-${BUILD_NUMBER}`
- `startDate` (Build timestamp)

Both `JOB_NAME` and `BUILD_NUMBER` are available as Jenkins environment properties.