[Jenkins General Configuration](https://wiki.ercot.com/display/CIA/Jenkins+General+Configuration)

[Skip to end of metadata](https://wiki.ercot.com/display/CIA/Jenkins+General+Configuration#page-metadata-end)

* Created by [Johnson, Brandon](https://wiki.ercot.com/display/~bjohnson), last modified on [Sep 22, 2020](https://wiki.ercot.com/pages/diffpagesbyversion.action?pageId=133108145&selectedPageVersions=1&selectedPageVersions=2)

[Go to start of metadata](https://wiki.ercot.com/display/CIA/Jenkins+General+Configuration#page-metadata-start)

Overview

The Jenkins instances maintained by the IT Tools Development group are configured using the JCasC plugin. The "configuration as code" approach ensures that in the in the event of an incident the group can recover quickly and with minimal disruption to customer teams. That means all core configuration for the Jenkins instances is stored in source control. The IT Tools team strongly recommends that all other teams follow the same practice. Do not configure jobs manually though Jenkins' UI. Such configuration is not easily repeatable and may be lost in the case of a destructive incident.

General Configuration

Jenkins has a number of features that enable the programmatic configuration and maintenance but the most important tool is JCasC. JCasC allows developers to configure the overwhelming majority of their needs from Jenkins programmatically. If a product changes teams migrating the instance level configuration is a relatively small task. If a patch breaks a Jenkins instance recovering into a clean instance is a relatively small task. If someone wants to set up an identical instance with minor changes it's a relatively small task. Essentially committing to configuring Jenkins programmatically makes distributing, modifying, copying, and restoring configuration a relatively low effort activity compared to the high effort option of managing configuration through Jenkins' UI.

Configuration Example

Every Jenkins instance has a corresponding configuration repository that is merged with the core configuration repository at instance deployment time. The instance specific repository grants development teams the ability to override inherited settings (i.e. adding new plugins) or to add new settings or functionality (i.e. describing jobs using the Job DSL).

jenkins:

clusterZone: "apps.devk8s.ercot.com"

master:

ingress:

hostName: "jenkins.<TEAM>.apps.devk8s.ercot.com"

tls:

- hosts:

- "jenkins.<TEAM>.apps.devk8s.ercot.com"

secretName: "jcasc-tls-certificate"

The YAML above is the standard instance configuration. It inherits all settings from the master configuration and only provides clusterZone, hostName, and tls configuration itself.

jenkins:

clusterZone: "apps.devk8s.ercot.com"

master:

ingress:

hostName: "jenkins.<TEAM>.apps.devk8s.ercot.com"

tls:

- hosts:

- "jenkins.<TEAM>.apps.devk8s.ercot.com"

secretName: "jcasc-tls-certificate"

master:

installPlugins:

- foobar:latest

The YAML above overrides the value for jenkins.master.installPlugins. Instead of using the value supplied in the core jcasc-configuration the overridden value specified in the example YAML file will be used. It is important to understand that overrides are in fact overrides and not merges. Choosing to manipulate a value, particularly values that are arrays, amounts to choosing to own that value in its entirety. Please reference the defaults documentation for information on the defaults included in the core configuration.

More Information

Please consult the JCasC [documentation](https://www.jenkins.io/projects/jcasc/)and the Jenkins Helm Chart [documentation](https://hub.helm.sh/charts/bitnami/jenkins/5.0.26)for details on what can be overridden.