Ansible Jenkins DevOps Roles Documentation

Release 0.0.1

Michael Jansen

Jul 27, 2017

Contents

This is not a turn-key solution to install jenkins. Its a set of roles to tailor yourself a running jenkins that is immediately useful without any manual touch. But you need to write your playbook yourself.

The following rules were applied when writing those roles.

Contents:

CHAPTER 1

Coding Standards

Global Variables

install_prefix Installation prefix to use for software. The default value is */opt*

It should be used for all *optional* software not installed using the default distribution package manager. It should not be used directly but used to define a role default installation prefix to allow fine grained control of the installation directory. Eg:

```
jenkins_prefix: "{{install_prefix}}/jenkins-ci"
```

Remember to use software specifx names for the prefix var if applicable (eg. java_home, catalina_home).

cache_directory Base directory of the local artefact cache. See *artefact cache*

Roles

Role Variables

All role arguments should be documented in *role/vars/main.yml*. But we have to keep in mind that variables set there have very high precedence. So the distinction between role argument and role default has to be done very carefully.

Role Arguments Role arguments are hard to override. Starting with version 2.0 ansible variable precedence is more clearly defined and tells us that role variables are only second to

- **role and include vars**
- block vars (only for tasks in block)
- task vars (only for the task)
- extra vars

Role Defaults Role defaults lose out to all other kind of variables. Because of this they should be named carefully to minimize the change for an accidently override. They should be prefix by role name (including any directory parts) joined by undescore.

```
# java/lang/oracle-sdk/defaults.yml
java_lang_oracle_sdk_default_version_8: 8.0.65
```

Become User

The *become user (aka sudo or su)* feature is forbidden in roles. Every role has to work without sudo rights for the installation user.

The rationale for this is that the role designer has no idea if all of his users have the possibility to use sudo on their machines. This information is only available to the playbook/play designer. A role should clearly document its requirements and thats it.

A role to install apache tomcat should NOT

- Create a tomcat user on the system
- Create a directory /srv/tomcat on the system as root

Instead it should document that

- it takes a parameter catalina_base and it needs the rights to create that directory/write to it. The playbook designer then has to make sure thats true. Either by creating a tomcat user and that directory in the playbook or by having the system administrators do it.

Important: A role that installs tomcat AND creates all desired users/groups/directories utilizing become_user is unusable in environments where sudo is not allowed (companies, hosted servers). The role therefore is NOT reusable in those environments and that equals wasted effort maintaining two roles.

A role should only do ONE thing. Playbooks/plays are responsible to chain roles together.

Artefact Cache

To make offline usage possible all artefacts acquired from external resources should be cached locally. A global variable is available to use for the location:

```
jenkins_cache: "{{cache_directory}}/jenkins-ci
```

CHAPTER 2

Apache Tomcat

Roles to download install and configure apache tomcat.

The setup implemented is the one described in tomcat advanced configuration. One installation of the tomcat release can power multiple tomcat instances.

Role apache/tomcat/defaults

A convenience role that provides some common defaults for all tomcat related roles.

You should not need to call it directly unless you implement a tomcat role.

Role `apache` **`tomcat-defaults`**

Become No

Defaults

- **`apache_tomcat_download_server`** – Mirror to download from (default: "http://mirror.synyx.de/apache/tomcat")
- **`apache_tomcat_prefix`** – Default installation prefix (default: *{install_prefix}*`/apache-tomcat`).
- **`apache_tomcat_server_port`** – Tomcat server port (default: 8005).
- **`apache_tomcat_http_connector_port`** – The nopn-ssl/tls http/1.1 connector port (default: 8080).
- **`apache_tomcat_ajp_connector_port`** – The ajp connector port (default: 8009).

Role apache/tomcat-8/configuration-copy

Copy a custom tomcat configuration into *CATALINA_BASE*.

Role apache/tomcat-8 **configuration-copy**

Become No

Dependency apache/tomcat-defaults

Parameters

- **catalina_home** – The apache tomcat install to use.
- **configuration** – Directory with the custom tomcat configuration.

This copies the content of +configuration+ into the tomcat instances *conf* directory.

Role apache/tomcat-8/configuration-default

Copy the tomcat default configuration into *CATALINA_BASE*.

Role apache/tomcat-8 **configuration-default**

Become No

Dependency apache/tomcat-defaults

Parameters

- **catalina_base** – Base directory of the new tomcat instance.
- **catalina_home** – The apache tomcat install to use.

This copies the conf/ directory from the tomcat archive into the instance/

Role apache/tomcat-8/install

Download and install tomcat 8.

The *prefix* directory needs to exist and writable.

Note: This does not create a running tomcat instance. It downloads and copies the archive to the host. To create a instance see `role-apache/tomcat-8-instance`.

Role apache **tomcat-8-install**

Become No

Dependency apache/tomcat-defaults

Parameters

- **version** – Tomcat version to install (eg. 8.0.29)
- **checksum** – Checksum for the tomcat archive (eg. 4b7ba7a5af0a5c395c0740fc011b59d1)
- **prefix** – Install prefix (default: *{apache_tomcat_prefix}*).

Role apache/tomcat-8/instance

Create a tomcat 8 instance.

Role `apache` **`tomcat-8-instance`**

Become No

Dependency apache/tomcat-defaults

Defaults

- **`apache_tomcat_8_default_version`** – Default tomcat version to install (default: unset)
- **`apache_tomcat_8_instance_catalina_home`** – Default *CATALINA_HOME* to use for instance setup (default: *{{apache_tomcat_prefix}}/apache-tomcat-{{tomcat_version|default(apache_tomcat_8_default_version)}}*)
- **`apache_tomcat_8_instance_manager_apps`** – Default tomcat webapps to activate for the instance (default: all - see below)
- **`apache_tomcat_8_java_opts`** – Default options for the jvm

Parameters

- **`manager_apps`** – Default tomcat webapp to activate for the instance (default: *"{{apache_tomcat_8_instance_manager_apps}}"*
- **`catalina_base`** – Base directory of the new tomcat instance.
- **`catalina_home`** – The apache tomcat install to use.
- **`java_home`** – Java installation to use for running tomcat.
- **`java_opts`** – Options for the jvm.
- **`server_port`** – Server port (default: *{{apache_tomcat_server_port}}*).
- **`http_connector_port`** – Http connector port (default: *{{apache_tomcat_http_connector_port}}*).
- **`ajp_connector_port`** – Ajp connector port (default: *{{apache_tomcat_ajp_connector_port}}*).
- **`cache_directory`** – Where to cache downloaded artifacts for future reuse on play host.

Configuration

This creates a tomcat instance with an empty *conf/* directory. The next step is to add some configuration. As there is no common tomcat configuration that fits all use cases its your responsibility to fill in the configuration according to your needs.

The role *`role-apache/tomcat-8/configuration-default`* copies the default tomcat configuration which is **NOT SUITED FOR PRODUCTION USE**. Its only provided for convenience in development setups.

The role *`role-apache/tomcat-8/configuration-copy`* can be used to copy a complete directory into ‘conf/.

Default Applications

Tomcats comes with the following web applications

- “host-manager”
- “manager”
- “ROOT”
- “examples”
- “docs”

CHAPTER 3

Jenkins

Roles to deploy and configure *jenkins*.

Role jenkins/bootstrap-job

Install a bootstrap job into jenkins

A bootstrap job will be created that checks out from a VCS (version control system) and then executes all scripts that match the file glob *bootstrap/*.job* as a job-dsl script. This naturally assumes the job-dsl plugin is installed and ready.

Currently only git is supported but adding support for other vcs is a simple matter.

Role `jenkins` **`bootstrap-job`**

Become No

Parameters

- **`jenkins_home`** – Jenkins instance home.
- **`git_repo`** – Git repository with job-dsl scripts to bootstrap the jenkins instance.
- **`git_branch`** – Branch to checkout from git_repository (default: master)

Role jenkins/configure

Base configuration for jenkins.

- Authentication
- Authorization
- Users
- Admin email and url.

Role `jenkins` **configuration**

Become No

Defaults

- **`jenkins_default_authentication_strategy`** – Default authentication strategy (*hudson_private*)
- **`jenkins_default_authorization_strategy`** – Default authorization strategy (*project_matrix*)

Parameters

- **`jenkins_home`** – Jenkins instance home.
- **`authentication_strategy`** – Authentication strategy to configure (default: *jenkins_default_authentication_strategy*)
- **`authorization_strategy`** – Authorization strategy to configure (default: *jenkins_default_authorization_strategy*)
- **`users`** – A list of users to create. Read the comments below.
- **`admin_email`** – Admin email address.
- **`url`** – Jenkins url.
- **`system_message`** – System message. (default: "Provisioned with ansible, all changes will be lost")
- **`number_of_executors`** – Number of executors (default: 2)
- **`node_mode`** – Node usage method. Valid values are "NORMAL" and "EXCLUSIVE" (default: "NORMAL")
- **`node_labels`** – Node labels as string. (default: "")

Authentication

Set *authentication_strategy* to one of the following values.

Jenkins' own user database hudson_private

Disable Security no_authentication

LDAP not yet implemented

Unix user/group database not yet implemented

Authorization

Set *authorization_strategy* to one of the following values.

Anyone can do anything no_authorization

Logged-in user can do anything full_control_once_logged_in

Matrix-based security global_matrix

Project-base Matrix Authorization Strategy project_matrix

Permissions

Jenkins permissions have string presentation. They consist of <group>.<permission>.

These are the group names for some permissions. The given category is from jenkins *Configure Global Sercurity* Page. To give a permission just append the permission to the group (eg. hudson.model.Hudson.Administer). In doubt configure the permission manually apply and check the `jenkins/config.xml` file in `JENKINS_HOME`

Overall hudson.model.Hudson

Credentials com.cloudbees.plugins.credentials.CredentialsProvider

Slave hudson.model.Computer

Job hudson.model.Item

Run hudson.model.Run

View hudson.model.View

SCM hudson.scm.SCM

Users

If set *users* is expected to be a list of hashes to define the users to create.

Only the authentication strategy *hudson_private* support creating users in jenkins.

Only for authorization strategy *project_matrix* and *global_matrix* permissions are configurable. Authentication strategy does not matter for permissions. Unless its *no_authentication*.

The password will never be changed if the user already exists.

Example configuration:

```
users: [
  {
  id: 'admin',
  password: 'admin',
  fullname: "Technical Administration Account",
  email: 'admin@example.com',
  permissions: [ "hudson.model.Hudson.Administer" ]
  },
  {
  id: 'mjansen',
  password: 'mjansen',
  email: 'mjansen@example.com',
  fullname: "Michael Jansen",
  permissions: [ "hudson.model.Hudson.Administer" ]
  },
  {
  id: 'test1',
  password: 'mjansen',
  email: 'mjansen@example.com',
  fullname: "Michael Jansen",
  permissions: [
    "hudson.model.Computer.Configure",
    "hudson.model.Item.Discover",
    "hudson.model.View.Delete",
    "hudson.model.Run.Update",
    "com.cloudbees.plugins.credentials.CredentialsProvider.Update" ]
  }
]
```

Role jenkins/configure-email-notification

Configure the email notification part of the jenkins configuration.

Role `jenkins` **`configure-email-notification`**

Become No

Parameters

- **`jenkins_home`** – Jenkins instance home.
- **`smtp_host`** – SMTP host to configure. If undefined mailer is reset to null.
- **`smtp_port`** – SMTP port (default: 587).
- **`smtp_replyto`** – Sender email address.
- **`smtp_usessl`** – Use SSL for smtp connection? (default: false).
- **`smtp_user`** – Username for smtp authentication. This is optional.
- **`smtp_password`** – Password for smtp authentication.
- **`smtp_charset`** – Character set for emails (default: utf-8).

If *smtp_host* is null then all other attributes are optional.

If *smtp_host* is given then *smtp_replyto* is required too.

If *smtp_username* is given then *smtp_password* is required too.

Parameter *smtp_usessl* is about SSL not TLS!

Role jenkins/configure-proxy

Configure the proxy settings in jenkins.

Role `jenkins` **`configure-proxy`**

Become No

Parameters

- **`jenkins_home`** – Jenkins instance home.
- **`proxy_host`** – Proxy Hostname
- **`proxy_username`** – Username on proxy
- **`proxy_password`** – Password on proxy
- **`proxy_port`** – Proxy port
- **`proxy_noproxy`** – Use direct connections for these. One host per line.

If *proxy_username* is defined *proxy_password* is required too.

If *proxy_hostname* is defined *proxy_port* is required too.

If *proxy_hostname* is undefined jenkins will be configured for direct connection.

no_proxy is a list of newline separated hostnames.

Role jenkins/deploy

Deploy the jenkins webapp into a application server. Currently only apache tomcat is supported.

The application server is *not* started after dropping jenkins. Neither is it stopped before doing that.

It will download and drop jenkins into the tomcat at *catalina_base*.

Role `jenkins` **`core`**

Become No

Defaults

- **`jenkins_download_mirror`** – Mirror to download from (default: http://mirrors.jenkins-ci.org/war)
- **`jenkins_deploy_default_context_path`** – Apache tomcat context path (default: *"jenkins"*).

Parameters

- **`version`** – Jenkins version to install (eg. 1.643)
- **`checksum`** – Checksum for the jenkins archive (eg. 4b7ba7a5af0a5c395c0740fc011b59d1)
- **`catalina_base`** – Tomcat instance to install into.
- **`jenkins_home`** – Jenkins instance home.
- **`context_path`** – Apache tomcat context path (default: *"{{jenkins_deploy_default_context_path}}"*)
- **`cache_directory`** – Where to cache downloaded artifacts for future reuse on play host.

The *context_path* becomes part of your jenkins url. Change to ROOT to deploy it at the top-level.

Role jenkins/plugins

Download and install the plugins specified in *plugins*.

An example

```
- jenkins_plugins:

  # CVS Plug-in
  - name: cvs
    version: "2.12"
    enabled: true
    bundled: true
    pinned: yes

  # Javadoc Plugin
  - name: javadoc
    version: "1.3"
    enabled: true
    bundled: true
    pinned: yes
```

If your plugins get overwritten after installation on a restart you need to look up the definition of pinned plugins.

To clone a jenkins instance executing the following script will give you the plugins in correct format

```
for (plugin in Jenkins.instance.pluginManager.plugins)
{
  pinned = plugin.pinned
  if (plugin.hasUpdate())
  {
    if (plugin.bundled)
    {
      pinned = false
    }
    version = plugin.updateInfo.version
  }
  else
  {
    version = plugin.version
  }

  println("""\
    # ${plugin.displayName}
    - name: ${plugin.shortName}
      version: "${version}"
      enabled: ${plugin.active}
      bundled: ${plugin.bundled}
      pinned: ${pinned}
    """.stripIndent())

}
```

Role jenkins **plugins**

Become No

Defaults

- **jenkins_plugins_download_url** – The url to download from.

Parameters

- **jenkins_home** – Jenkins instance home.
- **plugins** – List of plugins to install. See description (default: []).

CHAPTER 4

Programming Languages

Roles to install programming languages

Role lang/groovy-sdk

Download and install apache groovy.

Role `lang` **`groovy-sdk`**

Become No

Defaults

- **`lang_groovy_sdk_server`** – The download server to use (default: *"http://dl.bintray.com"*).
- **`lang_groovy_default_version`** – The default version to install. (default: *"2.4.5"*).
- **`lang_groovy_install_prefix`** – The installation_prefix (default: *"{{install_prefix}}/groovy-sdk"*).

Parameters

- **`version`** – The groovy version number to install (default: *"{{lang_groovy_default_version}}"*).
- **`prefix`** – Installation prefix (default: *{{lang_groov_install_prefix}}*).
- **`download_url`** – Url for the artefact (default: *"{{lang_groovy_sdk_download_server}}/groovy/maven/{{artifact}*
- **`artifact`** – Name of the artifact to download (default: *"apache-groovy-sdk-{{version}}.zip"*).
- **`cache_directory`** – Where to cache downloaded artifacts for future reuse on play host.

The *prefix* directory needs to exist and writable by the remote user.

The *cache* directory needs to exist and writable by the local user.

Note: The resulting GROOVY_HOME is *{prefix}*/groovy-sdk-*{version}*

Role lang/java/oracle-sdk

Download and install oracle java sdk.

Role lang/java **oracle-sdk**

Become No

Defaults

- **lang_java_oracle_sdk_server** – The download server.
- **lang_java_oracle_sdk_version** – The default java version to install (default: none).

Parameters

- **version** – The java version number to install (eg. 1.8.0_65)(default: *{{lang_java_oracle_sdk_version}}*.
- **build** – The build number of the version (eg. 17).
- **platform** – The platform to download.
- **format** – The format to download.
- **prefix** – Install prefix (default: *{install_prefix}*/java).
- **server** – Server to download from (default: *{lang.java.oracle_sdk.server}*)
- **cache_directory** – Where to cache downloaded artifacts for future reuse on play host.

Values for *platform* are “linux-x64” and “windows-i568”. All valid values can be seen at the download page. They are part of the archive filename.

Values for *format* are “tar.gz” and ”.Z”. Downloading and installing *rpms* or *.exe* is not yet implemented.

The *prefix* directory needs to exist and writable.

Note: The resulting JAVA_HOME is *{prefix}*/jdk*{version}*

CHAPTER 5

Indices and tables

- genindex
- search

Index