Create deployment apps

A **deployment app** consists of any arbitrary content that you want to download to a set of **deployment clients**. The content can include:

- A Splunk Enterprise app (such as those on Splunkbase)
- A set of Splunk Enterprise configurations
- Other content, such as scripts, images, and supporting files

You add a deployment app by creating a directory for it on the deployment server. Once you create the directory, you can use the forwarder management interface to map the app to deployment clients.

You can add to or change the content of an app at any time -- when you initially create the directory, or later, when you're ready to deploy or redeploy the app to deployment clients.

Create the app directories

You create separate directories for each deployment app in a special location on the deployment server. The default location is `$SPLUNK_HOME/etc/deployment-apps`, but this is configurable through the `repositoryLocation` attribute in serverclass.conf. Underneath this location, each app must have its own subdirectory. The name of the subdirectory serves as the app name in the forwarder management interface.

Note: After an app is downloaded, it resides under `$SPLUNK_HOME/etc/apps` on the deployment clients.

You can add apps at any time. After creating any new app directories, you must run the CLI `reload deploy-server` command to make the deployment server aware of them:

```
splunk reload deploy-server
```

By creating an app directory, you have effectively created the app itself, even if the directory does not yet contain any content. The app now appears in the forwarder management interface and you can use it to define server classes, as described in "About server classes".

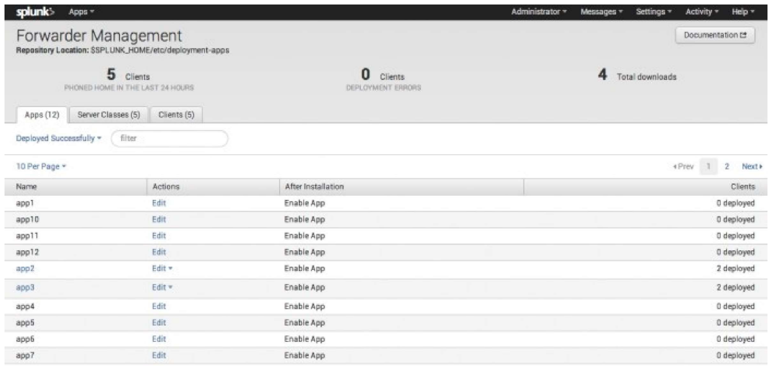
Important: When specifying app names (that is, when creating the app directories), you should be aware of the rules of configuration file precedence, as described in the topic "Configuration file precedence" in the Admin manual. In particular, note that app directory precedence is determined by ASCII sort order. For example, if you set an attribute/value pair `whatever=1` in the configuration file `x.conf` in an app directory named "A", the setting in app A overrides the setting `whatever=0` in `x.conf` in an app named "B", and so on.

Populate the apps

You can put content into an app directory at any time prior to deploying the app to its clients. You can later update and redeploy the app. To update the app, just add to or overwrite the files in the directory. For information on updating and deploying apps to clients, see "Deploy apps to clients".

View apps with forwarder management

Once you create an app directory, the deployment server adds it to its list of apps under the **Apps** tab of the forwarder management interface. For example:



Forwarder Management

Repository Location: $SPLUNK_HOME/etc/deployment-apps

5 Clients PHONED HOME IN THE LAST 24 HOURS

0 Clients DEPLOYMENT ERRORS

4 Total downloads

Name	Actions	After Installation	Clients
app1	Edit	Enable App	0 deployed
app10	Edit	Enable App	0 deployed
app11	Edit	Enable App	0 deployed
app12	Edit	Enable App	0 deployed
app2	Edit	Enable App	2 deployed
app3	Edit	Enable App	2 deployed
app4	Edit	Enable App	0 deployed
app5	Edit	Enable App	0 deployed
app6	Edit	Enable App	0 deployed
app7	Edit	Enable App	0 deployed

App management issues

Before deciding whether to use the deployment server to manage an app, there are some issues to consider.

The decision to manage an app with deployment server is irreversible

Once you start using the deployment server to manage an app, you cannot later stop using the deployment server to manage the app. It is important to understand the implications of this.

If you remove an app from the deployment server's `repositoryLocation` (defined in `serverclass.conf`), the deployment client will delete its copy of the app. There is no way to tell the deployment client instead to start managing the app on its own.

For example, say you are using the deployment server to manage updates to "appA". To do this, you have created a directory called "appA" on the deployment server and you have placed the app's contents there. From now on, whenever the deployment clients poll the server to check for updates, they compare their checksum for appA with the server's checksum for appA. If the checksums differ, the clients download the latest version of the app from the server. However, if appA has been deleted from the server's app repository, the client behavior is to delete their own instances of the app.

Therefore, by deleting an app from the deployment server, you are **not** telling the clients to stop using the deployment server to manage the app and to start managing it on their own. Instead, you're actually telling them to delete the app. Once the deployment server manages an app, it always manages that app.

The only way to allow an instance to continue managing its own copy of such an app is to disable the instance's deployment client functionality. If an instance is no longer a client of a deployment server, the deployment server will no longer manage its apps.

Because of this behavior, you should be "extremely" cautious before deciding to use the deployment server to manage the Splunk Enterprise search app. Managing the search app through the deployment server prevents users from saving any unique searches on their search heads. And because there's no way to tell the deployment server to stop managing an app, you're stuck with that decision.

Apps with lookup tables

In some cases, your indexers or search heads might be running apps that save information in lookup tables. Be careful about using the deployment server to manage such apps. When the deployment server distributes an updated app configuration, it overwrites the existing app. At that point, you'll lose those lookup tables.

You can work around this issue with the `excludeFromUpdate` setting in `serverclass.conf`. See Protect content during app updates.