

# Deployit Cookbook - Security Setup

Version 3.7.3

## Table of Contents

Table of Contents	2
Setting up Security Roles	3
Overview	3
Roles	3
Directories	4
Assigning Roles	4
Refining Security Levels	5
Restricted access in the UI	6

## Setting up Security Roles

Deployit provides fine-grained security settings based on roles and permissions and allows them to be configured through the CLI and in the GUI.

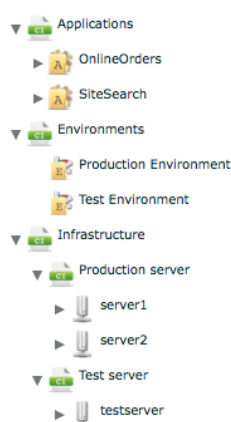
In this example, we'll be setting up security roles using the GUI. The example environment has two applications, **OnlineOrders** and **SiteSearch** that are both deployed to a test server before going to production. There are two different teams developing and deploying the applications. One team can't see the other's team application. Moreover, **developers** can only deploy to the test environment. **Deployers** can deploy both to the test environment and to production.

We'll set up different security roles for the different users in the organization.

This example assumes that there are already users and groups present in the system, for example by way of LDAP integration. For more detailed information on Deployit Security (including LDAP integration), please refer to the [Deployit Security Manual](#)

## Overview

Here's an overview of the environment for this example.



We have two applications: **OnlineOrders** and **SiteSearch**. They're maintained by two different teams and deployed independently.

Both applications are deployed to the same environments: The Test Environment, containing one server called **testserver** and the Production Environment, that contains two physical servers, **server1** and **server2**.

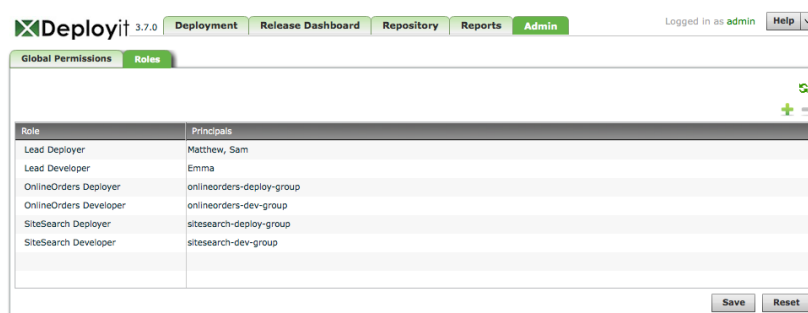
## Roles

At the current state, no roles have been implemented.

This is the list of roles we want to have in the system.

- **OnlineOrders developer** - May update the OnlineOrders application.
- **SiteSearch developer** - May update the SiteSearch application.
- **Lead developer** - May update and install any application on the Test Environment
- **OnlineOrders Deployer** - May update OnlineOrders application in Test and Production
- **SiteSearch Deployer** - May update SiteSearch application in Test and Production
- **Lead Deployer** - May install any application on any environment

To add roles, log in as the user with admin rights and got the Admin tab. Under the Admin tab, click on the Roles tab. This is the screen where you add, modify and delete roles. To add a role, click on the green plus on the right-hand side of the screen.



Enter the name of the role in the first column. The role is used within Deployit only. In the second column, 'Principals', you can specify a list of users and groups. If you have configured Deployit to point to your LDAP directory, you can add users and groups from LDAP here. To enter more than one principal, use a comma-separated list.

Hit Save to persist the changes. Note that the UI will display the roles in alphabetical order.

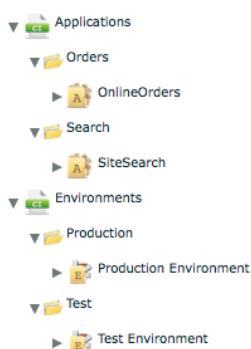
## Directories

The following is important enough to put in italics:

*In Deployit, you can only set security permissions on directories.*

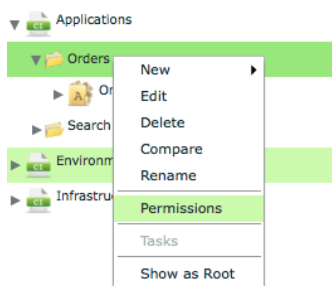
That means it's not possible to set permissions on individual CI nodes, like Packages or Environments. In practice you will need to impose a folder structure on your repository in order to set fine-grained permissions. The advantage here is that security will be explicit and localized in one place. Note that the root nodes in the repository ('Applications', 'Environments', etc.) are a special kind of folder and also allow setting of permissions.

Now let's add some directories. Go to the Repository browser, right-click on Applications and choose New > core > core.Directory. Name the new directory "Order". Now move the OnlineOrder package into the newly created Order directory by way of drag-and-drop. Following this procedure we create the following structure:



## Assigning Roles

To assign roles, right-click on a Directory and choose Permissions. Let's start by assigning roles to the Applications/Orders directory.



In the resulting screen, we add roles for Lead Developer, Lead Deployer, OnlineOrders Developer and OnlineOrders Deployer.

We assign the roles as follows.

Roles	import initial	import remove	import upgrade	read	repo edit
Lead Deployer	✓	✓	✓	✓	✓
Lead Developer	✓	✓			✓
OnlineOrders Deployer	□	□	✓	✓	□
OnlineOrders Developer	□	□	✓	✓	□

This means that both the Lead Developer and the Lead Deployer are allowed to add and remove applications in the Orders directory, but regular developers and deployers assigned to the project may only upgrade an existing deployment, say from OnlineOrders 1.0 to 2.0.

We can do a similar setup for the Search directory, substituting the OnlineOrders Developer and Deployer roles with the corresponding SiteSearch roles.

Note that if a role is not added to this screen, it effectively has no permissions. To remove a role, simply untick all its permissions and hit Save.

Now we'll set permissions to the Test environment. This environment is can be accessed by

both development and ops teams. Simply add all roles and grant all permissions to the Lead Developers and Deployers. Regular developers / deployers have more restricted access.

Roles	deploy initial	deploy undeploy	deploy upgrade	read	repo edit	task move step	task skip step
Lead Deployer	✓	✓	✓	✓	✓	✓	✓
Lead Developer	✓	✓	✓	✓	✓	✓	✓
OnlineOrders Deployer	□	□	✓	✓	□	□	□
OnlineOrders Developer	□	□	✓	✓	□	□	□
SiteSearch Deployer	□	□	✓	✓	□	□	□
SiteSearch Developer	□	□	✓	✓	□	□	□

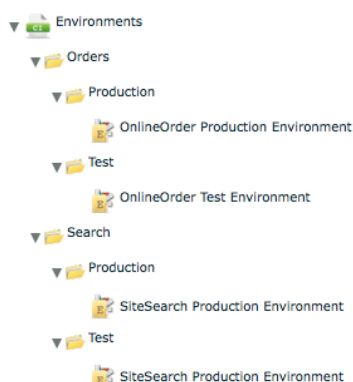
For production, we only want deployers to have access. Only the lead deployer may alter the environment from Deployit.

Roles	deploy initial	deploy undeploy	deploy upgrade	read	repo edit	task move step	task skip step
Lead Deployer	✓	✓	✓	✓	✓	✓	□
OnlineOrders Deployer	□	□	✓	✓	□	□	□
SiteSearch Deployer	□	□	✓	✓	□	□	□

## Refining Security Levels

In our current setup, the Test and Production environments are shared for the OnlineOrders and SiteSearch applications. That means that the OnlineOrders deployers can see the SiteSearch application and vice versa. To avoid this, we need to split the environments into application-specific environments. So the Production environment will be split into an 'Online Order Production environment' and 'SiteSearch Production environment'. Note that they will still refer to the same physical servers. But, by splitting them up, you can effectively hide them from different sets of users in Deployit. To do so, we will also need another Directory layer, that splits the applications under the Environment node.

Here's the new setup:



The permissions for Environment/Orders/Production look like:

Roles	deploy initial	deploy undeploy	deploy upgrade	read	repo edit	task move step	task skip step
Lead Deployer	✓	✓	✓	✓	✓	✓	✓
OnlineOrders Deployer	□	□	✓	✓	□	□	□

There are several important issues to note here.

**NOTE 1** - Security setting on a lower level overwrite *all* permissions from a higher level. So the settings on the Orders/Production directory is all there is for the OrderSearch Production Environment. There is no inheritance from higher levels, combining settings from various directories.

**NOTE 2** - If there are no permissions set on a directory, the permission settings from the parent are taken (recursively). So if you have a deep hierarchy of nested directories, but you don't set any permissions on them, Deployit will take the permissions set on the root node, Environments for example.

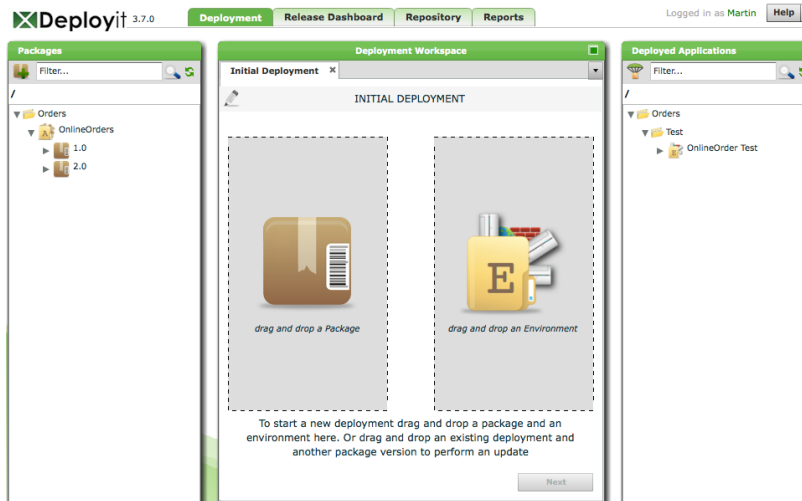
**NOTE 3** - All directories higher up in a hierarchy need to provide read permission for the roles defined in the lowest directory. Otherwise the permissions itself can't be read! This analogous to file permissions on unix directories. On unix, you also need read permissions on (all) parent directories, in order to list its contents. For example, here's the permissions table of Environment/Orders

Roles	deploy initial	deploy undeploy	deploy upgrade	read	repo edit	task move step	task skip step
Lead Deployer	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Lead Developer	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
OnlineOrders Deployer	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
OnlineOrders Developer	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

## Restricted access in the UI

After setting up the security roles, users will have restricted access to the UI. For example, this is what an OnlineOrder developer will see when logging in:

Note that the SiteSearch application and environments are missing, as well as the Production environment for OnlineOrders. The Admin tab is also inaccessible.



This concludes our Security Setup example. For more information, please refer to the [Deployit Security Manual](#)