

Roles and Permissions



- [Managing Roles](#)
 - [Assigning Roles](#)
 - [Content Level Permissions](#)
 - [Advanced Permission Logic](#)
-

Within BookStack the abilities of a user are controlled by the roles assigned to them and the permissions provided to those roles. A user can be assigned multiple roles, in which case the permissions will stack and the user will receive any ability if any of the roles is provided that specific ability.

Managing Roles

Roles can be created and edited by an admin user by navigating to "Settings > Roles" within the BookStack interface. A set of default roles are provided in a fresh BookStack instance for common use cases. When altering a role you'll be able to change the system permissions which provides certain system-wide abilities. In addition, you'll see a range of "Asset Permissions" which are the default abilities provided to members of that role for content within the system. These may be overridden by content-level permissions. Many of the asset permissions have an "Own" option. This provides the members of that role the ability only when they are the owner of an item in the system.

Assigning Roles

Roles can be assigned to a user, by an admin, via editing a user's profile in the "Settings > Users" area of the BookStack interface. Here multiple roles can be assigned to a user, in which case the permissions will stack and the user will receive any ability if any of the roles is provided that specific ability.

Keep in mind that certain mechanisms within BookStack do have the ability to alter the roles of a user. These include:

- The "Default user role after registration" option in the registration settings.
- LDAP group syncing.

- SAML2 group syncing.

Content Level Permissions

Permissions can be overridden at a Shelf, Book, Chapter or Page level where required. This can be done using the "Permissions" action when viewing such content, by any user that has a role permission to "Manage all/own permissions".

The screenshot shows the 'Book Permissions' interface. At the top, there is a breadcrumb trail: 'Books > BookStack User Guides > Book Permissions'. The main heading is 'Book Permissions'. Below it, a note states: 'Set permissions here to override the default permissions provided by user roles. Permissions set on books will automatically cascade to child chapters and pages, unless they have their own permissions defined.' To the right, there is an 'Owner' dropdown menu showing 'Admin'. The main content area contains three permission rows: 'Public' (The role given to public visitors if allowed), 'Editor' (User can edit Books, Chapters & Pages), and 'Everyone Else' (Set permissions for all roles not specifically overridden). Each row has checkboxes for 'View', 'Create', 'Update', and 'Delete', along with a red 'X' icon. The 'Public' row has 'View' checked. The 'Editor' row has 'View', 'Create', and 'Update' checked. The 'Everyone Else' row has 'Inherit defaults' checked. Below the rows, there is an 'Override permissions for role' dropdown menu with 'Select' as the current selection. At the bottom right, there are 'CANCEL' and 'SAVE PERMISSIONS' buttons.

Role	View	Create	Update	Delete
Public The role given to public visitors if allowed	☒	☐	☐	☐
Editor User can edit Books, Chapters & Pages	☒	☒	☒	☐
Everyone Else Set permissions for all roles not specifically overridden.	☐	☐	☐	☐

Within the permissions view you can use the "Override permissions for role" dropdown to select a role to set permissions for. When saved, these custom permissions will override the default permissions provided by roles for the configured content item.

An "Everyone Else" option is available to allow permission override/control for any roles that are not specifically overridden. By default, "Inherit defaults" will be selected for this item, which means that normal role permissions are used unless overridden. Deselecting this checkbox will use the permissions configured, in the same row, instead.

Content level permissions within BookStack have the following behavior:

- Custom permissions applied to books or chapters will auto-cascade to all child content within unless that content has its custom permissions.
- Custom permissions applied to shelves **will not** auto-cascade, due to the many-to-many relation to books, but they can be copied to all child books in a single action where

required.

- The default admin role always retains all permissions for ultimate control, and this role will not be configurable in permission views.

When custom content level permissions are active and affecting the currently viewed item, you'll see an indicator within the details sidebar section. If you have permission to edit the active content level permissions, this acts as a link that will take you to the relevant permissions view, even if applied to a parent chapter or book.

Details

🕒 Revision #10

★ Created 3 months ago by Dan Brown

✎ Updated 2 months ago by Dan Brown

🔒 Book Permissions Active

Advanced Permission Logic

With all the different permission options in BookStack, the scenarios and logic can become quite complex as these combine & stack. To help make sense of permission control interplay, below are some general high-level rules that BookStack follows. Within the below, "content permissions" refers to those set directly on shelves, books, chapters, and pages.

There are three main levels of permission, which have different levels of specificity.

From least specific to most specific:

1. Role permissions (Edited via the "Roles" interface).
2. Content "Everyone Else" permissions (Edited via the "Permissions" view for an item).
3. Role-specific content permissions (Also edited via the "Permissions" view for an item).

With those levels in mind:

- Most specific permission application (as above) take priority and can deny less specific permissions.
- Parent role-specific content permissions, that may be inherited ("Everyone Else" Inheriting checkbox active), are considered to essentially be applied on the item they are inherited to unless a lower level has its own permission rule for that specific role.
- Where both grant and deny exist at the same specificity, we side towards the grant.