



# Sitecore CMS 6

# Security Reference

*A Conceptual Overview for CMS Administrators, Architects, and Developers*

## Table of Contents

Chapter 1	Introduction.....	3
Chapter 2	The Sitecore Security Model.....	4
2.1	Overview.....	5
2.1.1	The .NET Security Model.....	5
2.2	Users.....	6
2.2.1	Predefined Users.....	6
2.2.2	The User Template.....	6
2.3	Roles.....	7
2.3.1	Predefined Roles.....	7
	The Everyone Role.....	7
	Sitecore\Author.....	7
	Sitecore\Designer.....	7
	Sitecore\Developer.....	7
	Sitecore Client Account Managing.....	7
	Sitecore Client Authoring.....	8
	Sitecore Client Configuring.....	8
	Sitecore Client Designing.....	8
	Sitecore Client Developing.....	8
	Sitecore Client Maintaining.....	8
	Sitecore Client Publishing.....	8
	Sitecore Client Securing.....	8
	Sitecore Client Translating.....	8
	Sitecore Client Users.....	8
	Sitecore Limited Content Editor.....	8
	Sitecore Limited Page Editor.....	8
	Sitecore Local Administrators.....	9
	Sitecore Minimal Page Editor.....	9
2.3.2	The Role Template.....	9
2.4	Domains.....	10
2.4.1	Default domains.....	10
2.5	Locally Managed Domains.....	11
2.5.1	Globally Visible Roles.....	11
2.5.2	Managed Domains.....	11
2.5.3	The Local Administrator.....	11
2.6	Security Providers.....	12
2.6.1	Sitecore's Security Providers.....	12
2.7	Access Rights.....	13
2.7.1	Access Rights.....	13
2.7.2	Access Right Settings.....	14
2.7.3	Security Inheritance.....	14
2.7.4	Effective Access Rights.....	14
2.7.5	Field Security.....	14
2.8	Security Presets.....	15
2.8.1	Predefined Security Presets.....	15
2.8.2	Security Preset Template.....	15
2.9	Workflow Security.....	16

# Chapter 1

## Introduction

This document describes the concepts that CMS administrators need to understand when designing, implementing, and maintaining the security infrastructure associated with a Sitecore Web site.

This document contains the following chapters:

**Chapter 1 — Introduction**

A brief description of this document and its intended audience.

**Chapter 2 — The Sitecore Security Model**

A complete description of the Sitecore security model.

## Chapter 2

# The Sitecore Security Model

This chapter contains a detailed description of the components and concepts that make up the Sitecore security model.

This chapter contains the following sections:

- Overview
- Users
- Roles
- Domains
- Locally Managed Domains
- Security Providers
- Access Rights
- Security Presets
- Workflow Security

## 2.1 Overview

The Sitecore security model leverages the following concepts:

- User — a uniquely identified named individuals.
- Role — a named element used to assign access rights to a group of users.
- Account — a generic name used when referring to either users or roles.
- Domain — a collection of accounts.
- Security Provider — a repository that stores a domain.
- Access Right — the permission to act upon an element of content or UI construct in a certain way.
- Item Security — a collection of access rights and associated accounts that influence how users may act upon a specific item.
- Field Security — a collection of access rights and associated accounts that influence how users may act upon a specific field.
- Security Inheritance — the ability to define access rights that apply to a specific item on one or more of the item's ancestors in the content tree.

### 2.1.1 The .NET Security Model

Sitecore leverages the .NET security model. This offers several immediate advantages, including:

- The .NET security model provides an abstraction from the details of a specific security provider.
- Using a pure .NET solution provides performance advantages.
- Sitecore work with any available .NET standard plug and play security provider.
- The ability to leverage multiple distinct security providers for a single solution.

Sitecore has, however, implemented some extensions to the .NET security model. For instance, Sitecore supports the concept of “roles in roles”, that is, the ability of a role to be a member of another role, which is not supported by the standard .NET security model. In such cases, Sitecore manages the extended features internally, so that the feature works even when using a 3<sup>rd</sup>-party .NET security provider.

## 2.2 Users

Users are the named accounts individuals use to log in to Sitecore clients. Sitecore provides a number of default users as examples, as described in section 2.2.1, Predefined Users.

### Important

Sitecore provides a default “Admin” user which has unrestricted access. By default, this user has the password “b” (lower case, single letter). You should change the password associated with this account and store the password in a secure location.

### 2.2.1 Predefined Users

The predefined default users are:

- **Built-in\anonymous** — a virtual user which is assigned to any unauthenticated visitor viewing a Web site that does not have an explicitly assigned domain.
- **built-in\owner** — a virtual role that refers to the user specified in an item’s creator / owner field.
- **extranet\anonymous** — a user which is assigned to any unauthenticated visitor viewing the default Web site.
- **sitecore\anonymous** — a user which is assigned to a visitor when accessing the Sitecore login page.
- **sitecore\Admin** — a predefined “Administrator” user which has unrestricted access to all features and content.

### 2.2.2 The User Template

Each user has a definition item based on the User template, which has the following fields:

- **Administrator** — if this checkbox is selected, the user has unrestricted access to all features and content.
- **CanBoost** — reserved for future use.
- **ClientLanguage** — the language used in the user interface of the Sitecore client.
- **ContentLanguage** — the language used in the user interface for Sitecore content.
- **DefaultItem** — defines the item that is displayed by default in Content Editor.
- **Email** — the user’s e-mail address.
- **Fullname** — the user’s full name.
- **Password** — the user’s password (the value is masked).
- **Portrait** — an icon or image which represents the user in the Sitecore menu of the Desktop.
- **RegionallsoCode** — the regional ISO code used by the user. This affects how Sitecore formats numbers, currency, dates and times.
- **Roles** — the roles assigned to this user.
- **Start Url** — the URL of the Sitecore client that the user must use, if applicable.
- **Wallpaper** — the Sitecore client’s wallpaper.

## 2.3 Roles

Roles provide a mechanism for assigning access rights to a group of users. Roles allow the grouping of users into structured units, such as managers, sales staff, anonymous users, and so on. Assigning access rights to roles, rather than directly to specific users, makes maintenance of security much easier. Roles allow security administrators to assign or revoke many access rights to an individual user by simply assigning or removing membership in one or more roles.

Users can belong to any number of roles, giving them different access rights to different areas of a site. When users belong to multiple roles, Sitecore merges all access rights assigned directly to the user and all roles, giving no precedence to access rights defined either on the user or roles. In cases where users have been assigned both deny and allow access rights, then deny takes precedence.

Roles can also be assigned other roles, referred to as “roles in roles”. When you assign a role to a user, the user will automatically also be assigned membership in the role’s role memberships.

### 2.3.1 Predefined Roles

The following sections describe the default predefined roles. Sitecore defines two types of roles: content roles and Sitecore Client roles. Content roles provide access to various areas of the content tree and are often members of one or more Sitecore Client roles. Sitecore Client roles provide access to functionality in the Sitecore user interfaces.

#### The Everyone Role

The *Everyone* role is a virtual role. Sitecore provides a global *Everyone* role, plus one for each domain. The role is used when assigning and resolving security for all users.

#### Sitecore\Author

*Sitecore\Author* provides basic authoring access to appropriate content items in the content tree. This role is a member of the *Sitecore Client Authoring* and *Sitecore Client Users* roles.

#### Sitecore\Designer

*Sitecore\Designer* provides read and write access to the areas of the content tree required when changing layout details for individual items and groups of items via template standard values, as well as items required when configuring the Page Editor Design Pane. This role is a member of the *Sitecore Client Designing* and *Sitecore Client Users* roles.

This role provides access to the Page Editor Design Pane features and the designer options of the **Presentation** tab of the Content Editor to allow various aspects of the page design to be edited.

#### Note

The *sitecore\Designer* role is not a member of the *Author* and *Authoring* roles, and therefore users with this role do not necessarily have access rights to allow changing content.

#### Sitecore\Developer

*Sitecore\Developer* provides access to developer specific content and functionality. This role is a member of the *sitecore\Author*, *sitecore\Designer*, *sitecore\Sitecore Client Developing*, *sitecore\Sitecore Client Maintaining*, and *sitecore\Sitecore Client Configuring* roles.

This is an appropriate role to assign to users who will make modifications to the Web site information architecture and presentation components.

#### Sitecore Client Account Managing

*Sitecore Client Account Managing* provides access to applications used to maintain users, roles, and domains.

## Sitecore Client Authoring

*Sitecore Client Authoring* provides access to basic item editing features and applications. Most client users should include membership in this role.

## Sitecore Client Configuring

*Sitecore Client Configuring* provides access to the Content Editor features that allow a user to change the configuration details associated with items, such as the icon associated with the item and whether the item is protected or hidden.

## Sitecore Client Designing

*Sitecore Client Designing* provides access to the Page Editor Design Pane features which allow a user to set layout details associated with items.

## Sitecore Client Developing

*Sitecore Client Developing* provides access to the developer specific functionality in the user interface, including access to the Developer Center and other developer applications and features in the Content Editor.

## Sitecore Client Maintaining

*Sitecore Client Maintaining* provides access to the Template Manager and the features related to the maintenance of the templates.

## Sitecore Client Publishing

*Sitecore Client Publishing* provides access to Sitecore's publishing features in the Content Editor, the Sitecore menu, and other applications.

## Sitecore Client Securing

*Sitecore Client Securing* provides access to features and applications used to assign access rights.

## Sitecore Client Translating

*Sitecore Client Translating* provides access to Sitecore's translation features in the Content Editor.

## Sitecore Client Users

*Sitecore Client Users* provides access to the Sitecore user interfaces. All users should be assigned this role. All the *Sitecore Client* roles are members of this role by default.

## Sitecore Limited Content Editor

*Sitecore Limited Content Editor* limits the amount of functionality provided in the Content Editor by the *Sitecore Client Authoring* role to display a simplified ribbon interfaced. This role is appropriate for users with limited Sitecore and/or computer skills.

## Sitecore Limited Page Editor

*Sitecore Limited Page Editor* limits the amount of functionality provided in the Page Editor by the *Sitecore Client Authoring* role to display a simplified ribbon interface. This role is appropriate for users with limited Sitecore and/or computer skills.

## Sitecore Local Administrators

*Sitecore Local Administrators* is a member of *Sitecore Client Users*, *Sitecore Client Account Managing*, and *Sitecore Client Securing*. This role provides a shortcut for adding these roles to a user.

## Sitecore Minimal Page Editor

*Sitecore Minimal Page Editor* limits the amount of functionality provided in the Page Editor by the *Sitecore Client Authoring* role to remove the ribbon interface and only display a minimal set of buttons to allow content modification. This role is appropriate for users with very little Sitecore and/or computer skills.

### 2.3.2 The Role Template

Each role has a definition item based on the Role template, which has the following fields:

- **Roles** — The list of roles associated with this role.

## 2.4 Domains

A domain is a group of accounts. Domains are usually used to collect accounts that have some logical relationship, for example, by default, all the accounts who have access to use the Sitecore clients are stored in the Sitecore domain, whereas all the accounts who have access to the published Web site are stored in the Extranet domain.

### 2.4.1 Default domains

Sitecore comes with 3 preinstalled domains, these include:

- **Built-in** — A virtual domain that is assigned to Web sites that are not explicitly associated with a domain in the `web.config` file.
- **Extranet** — By default, the Web site is configured to use this as its security domain, which contains users who can authenticate themselves when accessing the Web site.
- **Sitecore** — Internal security domain. Handles security for the Sitecore clients. Stores information about content editors, administrators, developers and other members who build and maintain the site.

## 2.5 Locally Managed Domains

A locally managed domain is a domain where the users of that domain only see that specific domain and accounts defined in that domain, and not the other domains within the system. Locally managed domains are usually maintained by a local administrator, who also will not be able to see the other domains within the system. This simplifies the process of supporting multiple sites from within a single installation as each locally managed domain will be administered and used by accounts that cannot see the other domains.

The concept of locally managed domains is part of the Sitecore “delegation model”. The delegation model includes additional features, such as *Globally Visible Roles* and *Managed Domains*.

### 2.5.1 Globally Visible Roles

This is a list of roles that the users in every domain can see. For example, the Sitecore Client roles are all globally visible roles by default.

### 2.5.2 Managed Domains

Users may have an associated list of managed domains, which is intended to allow users to see multiple locally managed domains.

Users of domains which are not locally managed can by default work with all domains. If you add one or more managed domains to this type of user he will then be restricted to only being able to see these managed domains, whether or not they are locally managed domains or just normal domains.

### 2.5.3 The Local Administrator

A local administrator is a user that is limited to working with accounts in either a locally managed domain or in one or more managed domains. Local administrators are typically members of the Sitecore Local Administrators role, which provides access to appropriate security applications.

#### Note

Sitecore local administrators can log in to Sitecore and manage the security applications (including assigning security) within that domain. A local administrator cannot create domains or assign managed domains to users.

## 2.6 Security Providers

A security provider is a security oriented software module that provides a structured interface between a service and a data source. In a similar way that device drivers abstract physical hardware devices, security providers abstract physical storage media.

ASP.NET can be configured to store security information virtually anywhere. All that's required is a custom provider to retrieve the information.

### 2.6.1 Sitecore's Security Providers

Sitecore CMS 6 uses three of the Microsoft security providers.

- **Membership Provider** — provides the interface between ASP.NET's membership service and Sitecore's membership data sources. The job of the membership provider is to connect to Sitecore security data sources containing information about registered users, and to provide methods for creating and deleting users, verifying login credentials, changing passwords and storing and retrieving membership information.
- **Role Provider** — provides the interface between ASP.NET's role management service and Sitecore's role data source. The job of a role provider is to interface with data sources containing role data mapping users to roles, and to provide methods for creating roles, deleting roles, adding users to roles, and so on. Given a user name, the role manager relies on the role provider to determine which roles the user belongs to.
- **Profile Provider** — provides the interface between ASP.NET's profile service and Sitecore's profile data sources. The job of a profile provider is to write profile property values supplied by ASP.NET to Sitecore's profile data sources, and to read the property values back from the data source when requested by ASP.NET. Profile providers also implement methods that allow consumers to manage profile data sources—for example, to delete profiles that haven't been accessed since a specified date.

## 2.7 Access Rights

Access rights are a set of possible actions that can be assigned to an account on a per item basis. Access rights may be explicitly allowed, explicitly denied, or inherited from a parent item. Security access rights may be granted or denied to individual users or roles.

### Note

Sitecore takes the least permissive approach to security. Unless specifically allowed for a user or any of its roles, Sitecore denies that access right to that user. In cases where an access right is both explicitly allowed and explicitly denied in separate roles assigned to an individual user, Sitecore will deny the access right.

### 2.7.1 Access Rights

Sitecore supports the following access rights:

- **Read** — controls whether an account can see an item in the content tree and/or on the published Web site, including all of its properties and field values.
- **Write** — controls whether an account can update field values. The write access right requires the read access right and field read and field write access rights for individual fields (field read and field write are allowed by default).
- **Create** — controls whether an account can create child items. The create access right requires the read access right.
- **Rename** — controls whether an account can change the name of an item. The rename access right requires the read access right.
- **Delete** — controls whether an account can delete an item. The delete access right requires the read access right.

### Important

The Delete command removes both the selected item and all child items, even if the account has been denied Delete rights for one or more of the subitems.

- **Administer** — controls whether an account can configure access rights on an item. The administer access right requires the read and write access rights.
- **Field Read** — controls whether an account can read a specific field on an item.
- **Field Write** — controls whether an account can update a specific field on an item.
- **Language Read** — controls whether a user can read a specific language version of items.
- **Language Write** — controls whether a user can update a specific language version of items.
- **Site Enter** — controls whether a user can access a specific site.
- **Workflow State Delete** — controls whether a user can delete items which are currently associated with a specific workflow state.
- **Workflow State Write** — controls whether a user can update items which are currently associated with a specific workflow state.
- **Workflow Command Execute** — controls whether a user is shown specific workflow commands.
- \* — controls all the access rights at once. You can use it to allow or deny all the rights assigned to a specific item at once.
- **Inheritance** — controls whether security rights can be passed from the parent items to the child items. The security model supports the possibility to choose inheritance on a per

account basis (applies to all access rights). The inheritance settings you choose apply to the selected account only.

## 2.7.2 Access Right Settings

Each access right may be set to one of the following:

- **Allow** — explicitly permit the associated access right for the selected account.
- **Deny** — revokes the associated access right for the selected account. Deny access overrides “allow” access when a user is both allowed and denied an access right through membership in multiple roles.
- **Inherit** — the default setting; neither permits nor revokes an access right. The status of the access right for a given user is determined based on a number of factors, including the complete collection of explicit access rights set on the user and assigned roles for the item in question and items higher up in the content tree.

## 2.7.3 Security Inheritance

By default, items inherit security for any given account. The right to inherit security can be allowed or denied for any given account. If security inheritance is enabled, the effective access rights granted to a user is also based on settings specified for items higher in the content tree.

## 2.7.4 Effective Access Rights

The term “effective access rights” refers to the resolved set of access rights for a specific user at a given time, based on the access rights set explicitly set for the user and all assigned roles, security inheritance, workflow state security, locking, protection, and other factors that dynamically affect access rights.

## 2.7.5 Field Security

Field security allows information architects to restrict access rights on specific fields to specific accounts. By default, every field in an item reflects the security applied to the item — if a user can write to the item, they can write to all fields of that item. Field security further restricts the access that certain users with read or write access to the item have to some fields in that item. Field security is set on data template field definition items.

## 2.8 Security Presets

Security presets provide a mechanism that security administrators can use to set a group of commonly used access right settings with a single click. Security presets are stored in the content tree under the `Sitecore/System/Settings/Security/Presets` folder.

### 2.8.1 Predefined Security Presets

The Sitecore client comes with two predefined security presets:

- **Remove Inherit** — this preset sets the *Inherit* right to *Denied* for the `**` (*Everyone*) role. This causes the item to not inherit any security rights from parent items.
- **Require Login** — this preset sets *Read* access to *Denied* for the *extranet/Anonymous* user, therefore requiring extranet login to access the item and its children.

### 2.8.2 Security Preset Template

The security preset definition comes from items created with the “Security Preset” template, which contains the following fields:

- **Security Preset** — the access rights to assign when a user clicks the preset in the Sitecore user interface.
- **Overwrite** — whether the preset completely overwrites previous assignments on the item.

## 2.9 Workflow Security

Security settings on workflow component definition items influence how Sitecore presents workflow related user interface features to the user.

The following access rights can be assigned to workflow state definition items:

- **Workflow State Delete** — controls whether or not a user can delete items which are currently associated with a specific workflow state
- **Workflow State Write** — controls whether or not a user can update items which are currently associated with a specific workflow state.

The following access right can be assigned to workflow command definition items:

- **Workflow Command Execute** — controls whether or not a user is shown specific workflow commands.

The effective access rights on an item can influence the behavior of the Workbox application. A user must have write access to an item in order to see the item in the Workbox. Note that a user may not have write access to an item if the item is currently checked out (locked) by another user.