Sitecore CMS 6.1 or later

Security Reference

A Conceptual Overview for CMS Administrators, Architects, and Developers

Table of Contents

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 documents focuses on the constructs and definition facilities Sitecore provides to give you the features and flexibility required to design, and implement the security structure required to meet business needs.

This document discusses the constituent parts of Sitecore that structure the way security is controlled in the form of domains, users, roles and access rights. It explains the purpose and structure of the parts of the system that configure how security is assigned and controlled. This includes topics such as how to set up domains and roles within the domains. It also explains how users and roles are structured and how security can be assigned to individual items and fields. It defines the purpose of security presets, used for controlling a whole range of security all at one time. This document also explains the preinstalled components of the security model and the default constituent parts for each area of the security infrastructure.

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 detailed description of the components and concepts that make up 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. In this chapter you will find details describing the function and purpose of all the components and concepts that make up the Sitecore security model.

This chapter contains the following sections:

- Features
- Big Picture
- Domains
- Locally Managed Domains
- Users
- Roles
- Access Rights
- Security Providers
- Security Presets
- Item and Field Security
- Workflow Security

2.1 Features

Domains are the main way of grouping users and roles as a unit with common rules and procedures. Within a domain we have a set of defined roles, which allows the grouping of users into structured units, such as when you want all the users of a particular department to have the same access rights. Within these roles are the Users, the named accounts individuals use to log in to Sitecore browser-based clients.

Once within the client the Users or Roles (commonly called Accounts) can be assigned whole sets of Access Rights, which are the different possible actions that can be assigned to an account on a per item basis.

You will be shown how the various access rights assigned to Roles and Users can be assigned with a single click through the use of Security Presets.

You will then be introduced to the concept of Security Providers and how Sitecore uses 3 of the Microsoft security providers to provide a structured interface between the Sitecore client and the security data source.

We will show you how access rights can be assigned to individual items and even individual fields through the use of Item and Field Security and finally how security can affect workflows through the use of Workflow Security.

2.2 Big Picture

The Sitecore Security Model permits administrators to allow or deny access to almost every aspect of a Web site, including both the content and the functional features.

2.2.1 The .NET Model

Using the .NET security engine in Sitecore offers several immediate advantages.

The advantages of using the .NET security engine are:

- Sitecore CMS uses the standard ASP.NET way of handling security, thereby bringing Sitecore into line with Windows technology.
- Sitecore can use a variety of plug and play features provided directly by Microsoft.
- Abstraction from the real data source.
- Easy option to replace / extend the default configuration with custom providers.
- The performance speed of a pure ASP.NET solution.
- The possibility of using several providers simultaneously and thus keeping the accounts in identifiable storage areas.

2.2.2 What is an Account?

Sitecore collectively refers to users and roles as “accounts”. Thus, an “account” is either a user and/or a role.

In general terms, throughout the system, security is set for an account rather than for a user or a role. That is to say when security is set the system does not specifically differentiate between users and roles, it treats them all as accounts.

2.3 Domains

A Sitecore domain is a group of accounts that are administered as a unit with common rules and procedures.

A domain is used to collect accounts that have some logical relationship, for example, all the accounts who have access to use the Sitecore clients may be stored in the Sitecore domain, whereas all the accounts who have access to the published Web site may be stored in the Extranet domain.

2.3.1 Default domains

Sitecore comes with 3 preinstalled domains, these are:

- **Built-in** - A virtual domain that only exists in memory. Sitecore CMS is setup with Extranet as the default domain and in that case most accounts will be **extranet\anonymous** when they visit the Web site. However, if the Web site does not specify a default domain, then accounts will be registered as **built-in\anonymous**.
- **Extranet** – Web site security domain. Defines who can access the information published on the Intranet, Extranet and/or Internet Web site. Generally referred to by Sitecore documents as extranet security regardless of what application is being built.
- **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.

2.4 Locally Managed Domains

A locally managed domain is a domain where the users of that domain only see that specific 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. Also, users defined in a locally managed domain will only see accounts that have been defined within their own domain. 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”

2.4.1 Globally Visible Roles

This is a list of roles that the users in every domain can see.

2.4.2 Managed Domains

This is a list of the domains associated with a user, which allows users to see multiple domains.

For users of a locally managed domain, this gives them the added permissions to see the other domains they have been associated with.

Uses of normal domains (not locally managed ones) 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.

2.4.3 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. The local administrator is defined by being member of the Sitecore Local Administrators role and being a member of a locally managed domain (or having one or more managed domains assigned.)

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.

2.5 Users

Users are the named accounts individuals use to log in to Sitecore clients. Sitecore 6 provides a number of default users, and we recommended that you do not change these users.

Important
The only exception to this is the “Admin” user. Changes to the password of the “Admin” user are essential to protect the security of your system. This should be the first security change that is made to your system.

If you wish to have a user with similar authority as one of the default users, then it is advised to create a new user rather than edit one of the supplied users as this may have implications within other areas of the security model.

2.5.1 Predefined Users

The predefined default users are:

- **Built-in\anonymous** — a virtual user which is assigned to a user viewing the free to access parts of the Web site through the **built-in** domain.
- **built-in\owner** — a virtual role that refers to the user referred to in an item’s current creator / owner field.
- **extranet\anonymous** — a user who is viewing the free to access parts of a Web site.
- **sitecore\anonymous** — a user who only has access to the login screen of the Web site. This is used for users who will go on to access the application framework section of the Web site.
- **sitecore\Admin** — a predefined “Administrator” CMS user.

2.5.2 The User Template

The user’s definition comes from the items created with the User template, which has the following fields:

- **Administrator** — if this field is selected, the user can access everything regardless of their security settings.
- **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, stored as a “password” field type.
- **Portrait** — an icon or image which represents the user in the Sitecore menu of the Desktop.
- **RegionalIsoCode** — the regional ISO code used by the user. This affects how numbers, currency, dates and times are formatted.
- **Roles** — the roles assigned to this user.
- **Start Url** — the URL used for the home page of the user when they login.

- **Wallpaper** — the Sitecore client's wallpaper.

2.6 Roles

Roles are defined in the system to manage Web site authorization. Roles allow the grouping of users into structured units, such as managers, sales staff, anonymous users, and so on. This makes organization of security access much easier with the ability to assign security access rights through a role to multiple users, rather than having to assign the same access rights to a list of users. Roles give you the flexibility to change permissions and add and remove users without having to make changes to the whole site.

Users can belong to more than one role, giving them different access rights to different areas of a site.

Users who have been assigned multiple roles gain their access rights from all the assigned roles.

Roles can also be assigned other roles. We call this feature 'roles in roles'. For more information about this topic, see section 2.6.3 *Roles in Roles*.

2.6.1 Predefined Roles

Sitecore comes with a series of predefined roles.

The Client Users Role

The *Sitecore Client Users* role is required to grant minimal access to Sitecore shell. With this role you should be able to login to Sitecore Desktop, but you will not have access to any applications.

All of the other Sitecore client roles are members of the *Sitecore Client Users* role. Users in any Sitecore client role are automatically members of the *Sitecore Client Users* role.

The Everyone Role

The *Everyone* role is not a physical role, it is a virtual role. It mirrors the Windows 'Everyone' construct. It does not exist as a part of the role database but is purely used when assigning and resolving security. The *Everyone* role can be used to assign access rights to every user or every user in a specific domain. The *Everyone* role is available as both a global role and a local role in every domain.

Sitecore\Author

This is what we call a "content role". It provides access to content in the content tree (hence the name "content role". This role also has two of the *Sitecore Client* roles assigned to it, so that you can assign just this role to a user and the *Sitecore Client Authoring* and *Sitecore Client Users* roles will be automatically assigned to the user.

This role provides access to basic item editing features such as the Media Library and the Content Editor with a reduced set of tabs on the ribbon.

Sitecore\Designer

This role 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 also has two of the *Sitecore Client* roles assigned to it, so that you can assign just this role to a user and the *Sitecore Client Designing* and *Sitecore Client Users* will be automatically assigned to the user. It should also be noted that the *sitecore\Designer* role is not a member of the *Author* and *Authoring* 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. Again, it should be noted that this roles has no access to manipulating items in any way.

Sitecore\Developer

This role is also a “content role” in that users assigned this role have access to manipulation of items within the content tree. This role also provides the access that both the *sitecore\Author* role and the *sitecore\Designer* roles have. Also added to this role are the *sitecore\Sitecore Client Developing*, the *sitecore\Sitecore Client Maintaining* and *sitecore\Sitecore Client Configuring* roles to allow access to all the areas of the system that a Sitecore developer would normally require.

This role provides access to content manipulation facilities in the Content Editor, plus all the design and authoring roles normally used by client authors and client designers. It also provides access to more functionality on the ribbon of the Content Editor to allow full development features for users assigned to this role. This role also has access to the “Development Tools” menu on the Sitecore menu, giving access to further development tools such as the “Package Designer” and other tools.

Sitecore Client Authoring

This is a “user interface” role which allows users to access to basic item editing features. Most client users should have access to this role to allow access to basic authoring features.

Sitecore Client Designing

This role provides access to Page Editor Design Pane features which allow a user to set layout details associated with items in the Sitecore client.

Sitecore Client Securing

This role allows a user to assign access rights using the Content Editor and other appropriate applications.

Sitecore Client Account Managing

This role allows a user to maintain users, roles, and domains through the use of the Access Manager, the Domain Manager, the Role Manager and the User Manager.

Sitecore Minimal Page Editor

This role limits the amount of functionality provided by the *Sitecore Client Authoring* role (which is still required for users given this role). This role restricts the amount of functionality provided in the Page Editor to the absolute minimum and users who have been assigned this role do not have access to the Page Editor ribbon.

Sitecore Limited Page Editor

This role limits the amount of functionality provided by the *Sitecore Client Authoring* role (which is still required for users given this role), but allows more functional access than the *Sitecore Minimal Page Editor* role. This role restricts the amount of functionality that is available in the Page Editor. However, unlike the *Minimal Page Editor* role users assigned this role will see a simple version of the standard Page Editor ribbon.

Sitecore Limited Content Editor

This role limits the amount of Content Editor functionality provided by the *Sitecore Client Authoring* role (which is still required for users given this role).

With they are assigned this role, a content author only has access to the **Home**, **Review** and **Publish** tabs in the Content Editor ribbon and has no access to the copying, moving and sorting facilities on the item “right-click” menu.

2.6.2 The Role Template

The role definition comes from items created from the “Roles” template, which has the following fields:

- **Roles** – The list of defined Roles.

2.6.3 Roles in Roles

A role can be a member of other roles. Users who are members of a role automatically inherit the membership of the additional roles that the role is a member of. Roles in Roles is useful in administration for grouping commonly used roles together, rather than adding several roles to users, as in the *Sitecore Author* and *Designer* roles.

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.

Important
To simplify administration, an item can inherit access rights from its parent. When an item does not specify access rights for an account or any of its roles, the security inheritance flag in the item controls whether access rights defined in the parent item apply to the item. Sitecore evaluates inherited access rights recursively until it finds rights defined for the account or any of its roles, the inheritance flag disabled, or the root of the data repository.

Important
Security inheritance is enabled by default. If security inheritance is not explicitly disabled for an item, it is allowed.

Important
Sitecore uses the least permissive approach to determine access rights. If an item does not specify an access right for an account or any of its roles, Sitecore denies that right by default. The default value for access rights is denied.

Important
User access rights override role access rights. If an item explicitly allows or denies an access right to a user, access rights for the user's roles do not apply. Explicitly granting an access right to a user overrides denying that access right to one of the user's roles.

Important
If two roles have conflicting access rights, deny always overrides allow. Denial of an access right to one of a user's roles overrides allowance of that access right to another of their roles. If an item does not specify an access right for the user, but allows that access right to one of the user's roles and denies it to another, Sitecore denies that access right to that user.

2.7.1 Access Rights

Sitecore comes with a series of predefined access rights. These are:

- **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 also deletes 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 has one of three possible settings. They are:

- **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 in that a user is denied the access right if the user or any of their roles is denied, or if the access right is never specifically allowed for the user or any of their roles.
- **Inherit** — 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 Constructs

There are also three security constructs within the security model. These are:

- **Security Inheritance** — If security inheritance is disabled, the effective access rights granted to a user is based on settings specified for the user and roles assigned to the user. 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.

- **Effective Rights** — the set of access rights assigned to the user after considering settings for the user and all assigned roles, security inheritance, workflow state security, locking, protection, and other factors that dynamically affect access rights.
- **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.

2.8 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 information virtually anywhere. All that's required is a custom provider to retrieve the information.

2.8.1 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 interface with Sitecore Security data sources containing data regarding the 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 what role or 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.

2.9 Security Presets

Security presets are a way to set a group of commonly used security access rights with a single click. A security preset can be created and stored in `sitecore/System/Settings/Security/Presets`.

You can then "stack" the accounts in one preset, each with unique settings. You can apply the preset from the Content Editor, **Security** tab, **Presets** group.

2.9.1 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. Therefore the item does *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.

2.9.2 Security Preset Template

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

- **Users** — the list of users who will be assigned the preset.
- **Domain** — the domain to which the preset/user combination will be applied to.
- **Security Preset** — the security preset.
- **Overwrite** — whether the preset completely overwrites previous assignments on the item.

2.10 Item and Field Security

In the Sitecore security model each item can be assigned individual. These rights are, if allowed, inherited recursively. Field security allows information architects to restrict the access rights on specific fields to specific users and roles.

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 in that item. However, field security further restricts the access that certain users with read or write access to the item have to some of the fields in that item.

You can also assign security settings at field level. For instance, a developer can configure one role to have read and write access to a particular field, while another role might only have read access, and a third role neither read nor write access.

The security settings for an individual field are set in the Field Security control, in the Security section for a given field.

2.10.1 Standard Template field Security

All items inherit their security settings, by default, from the Sitecore Standard template.

You can change the security settings for particular fields in the Standard template, thereby granting or denying roles or users read or write access to the fields in the Standard template.

Note
The *Everyone* role is denied access to most of the fields in the Standard template. This means that only administrator users can see these fields. If you want to make protected fields visible to non-administrator users, you must first set the read access right to inherit for the *Everyone* role. Set the read access right to allow for roles and/or users that you want to be able to view the given field. Allow write access for roles and users that should be able to modify the fields.

Important
Sitecore strongly recommends that you do *not* modify the Standard template as this will complicate the upgrade process and may cause other problems as the Standard template is inherited throughout the system.

2.11 Workflow Security

Security settings on workflow states influence the actions a user can take on items in workflows and also control which workflow commands a user can see in the workbox.

2.11.1 Workflow Item and Command Access Rights

The following access rights can be assigned to workflow 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 commands:

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

2.11.2 Resolved Workflow Access Rights

Resolved workflow access rights can be set for individual items. The resolved item access rights are:

- **Read** — a user must have read access to an item to see the item in the Content Editor, the published Web site, or anywhere else.
- **Write** — 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.