# Table of Contents

**TWiki Access Control**

- An Important Control Consideration
- Permissions settings of the webs on this TWiki site
- Authentication vs. Access Control

## Users and Groups

- Managing Users
- Managing Groups
- The Super Admin Group

## Restricting Access

- Controlling access to a Web
- Controlling access to a Topic
- Allowing public access to specific topics in a restricted web
- Empty values in access control variables
- Securing File Attachments
- Controlling who can manage top-level webs
- How TWiki evaluates ALLOW/DENY settings
- Allowing web creation by user mapping manager

## Restricting Access to Specific Users

- Forbid certain users to do certain actions by configuration
- User masquerading

## Dynamic Access Control

- Dynamic access control
  - Example 1 - restriction based on topic name
  - Example 2 - restriction based on form field
  - Dynamic access control in accessing a different web's topic
  - Topic level dynamic access control
  - Dynamic access control and user masquerading
  - Avoiding vulnerability
  - Disabling dynamic access control

## Access Control and INCLUDE

- Access control and INCLUDE
- Customizing "access denied" message
- Custom user/group notations

## Access Control Quick Recipes

- Restrict Access to Whole TWiki Site
- Authenticate and Restrict Selected Webs Only
- Hide Control Settings
- Obfuscating Webs
- Read-only Skin Mode
- Configuring access control for topics of a certain name in all webs
TWiki Access Control

*Restricting read and write access to topics and webs, by Users and groups*

TWiki Access Control allows you restrict access to single topics and entire webs, by individual user and by user Groups. Access control, combined with TWikiUserAuthentication, lets you easily create and manage an extremely flexible, fine-grained privilege system.

💡 **Tip:** TWiki:TWiki.TWikiAccessControlSupplement on TWiki.org has additional documentation on access control.

### An Important Control Consideration

*Your organization will learn that, while fostering an open collaborative environment, soft security (peer review), together with version control (complete audit trail) will take care of any security concern you might have.*

Open, free-form editing is the essence of WikiCulture - what makes TWiki different and often more effective than other collaborative environments. For that reason, it is strongly recommended that decisions to restrict read or write access to a web or a topic are made with great care - the more restrictions, the less wiki in the mix. Experience shows that *unrestricted write access* works very well because:

- **Peer influence** is enough to ensure that only relevant content is posted.
- **Peer editing** - the ability for anyone to rearrange all content on a page - keeps topics focused.
- In TWiki, content is transparently preserved under revision control:
  - Edits can easily be rolled back to a previous revision if needed.
  - Users are encouraged to edit and refactor (condense a long topic), since there's a safety net.

As a collaboration guideline:

- Create broad-based Groups (for more and varied input), and...
- Avoid creating view-only topics (if you can read it, you should be able to contribute to it).

### Permissions settings of the webs on this TWiki site

There are too many webs and it would take too long to show permission settings of them all.

### Authentication vs. Access Control

**Authentication:** Identifies who a user is based on a login procedure. See TWikiUserAuthentication.

**Access control:** Restrict access to content based on users and groups once a user is identified.

### Users and Groups

Access control is based on the familiar concept of Users and Groups. Users are defined by their WikiNames. They can then be organized in unlimited combinations by inclusion in one or more user Groups. For convenience, Groups can also be included in other Groups.
Managing Users

A user can create an account in TWikiRegistration. The following actions are performed:

- WikiName and encrypted password are recorded using the password manager if authentication is enabled.
- A confirmation e-mail is sent to the user.
- A user profile page with the WikiName of the user is created in the Main web.
- The user is added to the TWikiUsers topic.

The default visitor name is TWikiGuest. This is the non-authenticated user.

Managing Groups

The following describes the standard TWiki support for groups. Your local TWiki may have an alternate group mapping manager installed. Check with your TWiki administrator if you are in doubt.

Groups are defined by group topics located in the Main web. To create a new group, visit TWikiGroups and enter the name of the new group ending in Group into the "new group" form field. This will create a new group topic with two important settings:

- Set GROUP = < list of Users and/or Groups >
- Set ALLOWTOPICCHANGE = < list of Users and/or Groups >

The GROUP setting is a comma-separated list of users and/or other groups. Example:

- Set GROUP = SomeUser, OtherUser, SomeGroup

The ALLOWTOPICCHANGE setting defines who is allowed to change the group topic; it is a comma delimited list of users and groups. You typically want to restrict that to the members of the group itself, so it should contain the name of the topic. This prevents users not in the group from editing the topic to give themselves or others access. For example, for the MarketingGroup topic write:

- Set ALLOWTOPICCHANGE = MarketingGroup

⚠️ Note: TWiki has strict formatting rules. Make sure you have a real bullet. (In raw edit it is three or six spaces, an asterisk, and an extra space in front of any access control rule.)

The Super Admin Group

A number of TWiki functions (for example, renaming webs) are only available to administrators. Administrators are simply users who belong to the SuperAdminGroup. This is a standard user group, the name of which is defined by {SuperAdminGroup} setting in configure. The default name of this group is the TWikiAdminGroup. The system administrator may have chosen a different name for this group if your local TWiki uses an alternate group mapping manager but for simplicity we will use the default name TWikiAdminGroup in the rest of this topic.

You can create new administrators simply by adding them to the TWikiAdminGroup topic. For example,

- Set GROUP = RobertCailliau, TimBernersLee

A member of the Super Admin Group has unrestricted access throughout the TWiki, so only trusted staff should be added to this group.
Restricting Access

You can define who is allowed to read or write to a web or a topic. Note that some plugins may not respect access permissions.

- Restricting VIEW blocks viewing and searching of content. When you restrict VIEW to a topic or web, this also restricts INCLUDE and Formatted SEARCH from showing the content of the topics.
- Restricting CHANGE blocks creating new topics, changing topics or attaching files.
- Restricting RENAME prevents renaming of topics within a web.

Note that there is an important distinction between CHANGE access and RENAME access. A user can CHANGE a topic, but thanks to version control their changes cannot be lost (the history of the topic before the change is recorded). However if a topic or web is renamed, that history may be lost. Typically a site will only give RENAME access to administrators and content owners.

Controlling access to a Web

You can define restrictions on who is allowed to view a TWiki web. You can restrict access to certain webs to selected Users and Groups, by:

- **authenticating all webs and restricting selected webs**: Topic access in all webs is authenticated, and selected webs have restricted access.
- **authenticating and restricting selected webs only**: Provide unrestricted viewing access to open webs, with authentication and restriction only on selected webs.

You can define these settings in the WebPreferences topic, preferably towards the end of the topic:

- Set DENYWEBVIEW = < comma-delimited list of Users and Groups >
- Set ALLOWWEBVIEW = < comma-delimited list of Users and Groups >
- Set DENYWEBCHANGE = < comma-delimited list of Users and Groups >
- Set ALLOWWEBCHANGE = < comma-delimited list of Users and Groups >
- Set DENYWEBRENAME = < comma-delimited list of Users and Groups >
- Set ALLOWWEBRENAME = < comma-delimited list of Users and Groups >

For example, set this to restrict a web to be viewable only by the MarketingGroup:

- Set ALLOWWEBVIEW = Main.MarketingGroup

If your site allows hierarchical webs, then access to sub-webs is determined from the access controls of the parent web, plus the access controls in the sub-web. So, if the parent web has ALLOWWEBVIEW set, this will also apply to the subweb. Also note that you will need to ensure that the parent web’s FINALPREFERENCES does not include the access control settings listed above. Otherwise you will not be able override the parent web’s access control settings in sub-webs.

Creation and renaming of sub-webs is controlled by the WEBCHANGE setting on the parent web (or ROOTCHANGE for root webs). Renaming is additionally restricted by the setting of WEBRENAME in the web itself.

**Note**: If you restrict access to the Main, make sure to add the TWikiRegistrationAgent so that users can register. Example:

- Set ALLOWWEBCHANGE = TWikiAdminGroup, TWikiRegistrationAgent

**Note**: For Web level access rights Setting any of these settings to an empty value has the same effect as not setting them at all. Please note that the documentation of TWiki 4.0 and earlier versions of TWiki 4.1 did not
reflect the actual implementation, e.g. an empty ALLOWWEBVIEW does not prevent anyone from viewing the web, and an an empty DENYWEBVIEW does not allow all to view the web.

Controlling access to a Topic

- You can define these settings in any topic, preferable towards the end of the topic:
  - Set DENYTOPICVIEW = < comma-delimited list of Users and Groups >
  - Set ALLOWTOPICVIEW = < comma-delimited list of Users and Groups >
  - Set DENYTOPICCHANGE = < comma-delimited list of Users and Groups >
  - Set ALLOWTOPICCHANGE = < comma-delimited list of Users and Groups >
  - Set DENYTOPICRENAME = < comma-delimited list of Users and Groups >
  - Set ALLOWTOPICRENAME = < comma-delimited list of Users and Groups >

For example, set this to restrict a topic to be viewable only by the MarketingExecGroup:

- Set ALLOWTOPICVIEW = Main.MarketingExecGroup

See "How TWiki evaluates ALLOW/DENY settings" below for more on how ALLOW and DENY interacts.

⚠️ If the same setting is defined multiple times the last one overrides the previous. They are not OR'ed together.

Allowing public access to specific topics in a restricted web

You may want to completely open up access to a specific topic within a restricted web - allowing access by anybody. There is a special group for that - Main.AllUsersGroup. The following setting allows view access to the topic by anybody even if they are not authenticated.

- Set ALLOWTOPICVIEW = Main.AllUsersGroup

Alternatively, you can grant access only to authenticated users by Main.AllAuthUsersGroup. If an unauthenticated user accesses a topic having the following setting, they are asked to authenticate themself.

- Set ALLOWTOPICVIEW = Main.AllAuthUsersGroup

Remember when opening up access to specific topics within a restricted web that other topics in the web - for example, the WebLeftBar - may also be accessed when viewing the topics. The message you get when you are denied access should tell you what topic you were not permitted to access.

As mentioned in the following section, meaning of an empty value set to DENYTOPICVIEW, DENYTOPICCHANGE, and DENYTOPICRENAME has been changed in TWiki 6.0. To keep those TWiki topics having empty DENYTOPICOPERATION accessible by everybody, those need to be replaced with

- Set ALLOWTOPICOPERATION = Main.AllUsersGroup

For that, tools/eliminate_emptydenytopic is provided. After upgrading from pre 6.0 to post 6.0, you need to run it.

Empty values in access control variables

Setting an empty value to an access control variable is the same as not setting at all:

- Set ALLOWTOPICVIEW =
Since TWiki 4.0 and prior to TWiki 6.0 setting DENYTOPICVIEW, DENYTOPICCHANGE, or DENYTOPICRENAME to an empty value meant "do not deny anyone regardless of the corresponding ALLOWTOPICX", which is no longer the case. Back then, setting an empty value to DENYTOPICX was the only way to open up a topic to everybody in a restricted web. Now that we have AllUsersGroup and AllAuthUsersGroup, there is no need for that behaviour, which caused a lot of confusion and debate.

Securing File Attachments

By default, TWiki does not secure file attachments. Without making the following changes to the twiki.conf file, it is possible for anyone who has access to the server to gain access to an attachment if they know the attachment's fully qualified path, even though access to the topic associated with the attachment is secured. This is because attachments are referred to directly by Apache, and are not by default delivered via TWiki scripts. This means that the above instructions for controlling to topics do not apply to attachments unless you make the changes as described below.

An effective way to secure attachments is to apply the same access control settings to attachments as those applied to topics. This security enhancement can be accomplished by instructing the webserver to redirect accesses to attachments via the TWiki viewfile script, which honors the TWiki access controls settings to topics. See the notes below for implications.

The preferred method to secure attachments is by editing the twiki.conf file to include:

```plaintext
  ScriptAlias /do /filesystem/path/to/twiki/bin
  Alias /pub/TWiki /filesystem/path/to/twiki/pub/TWiki
  Alias /pub/Sandbox /filesystem/path/to/twiki/pub/Sandbox
  ScriptAlias /pub /filesystem/path/to/twiki/bin/viewfile
```

**Notes:**

- It is recommended to use TWiki:TWiki/ApacheConfigGenerator to generate the Apache config file for your TWiki.
- You will need to restart your Apache server after this change.
- Images embedded in topics will load slower since attached images will also be delivered by the viewfile script. The TWiki web and Sandbox web are excluded for performance reasons.
- The viewfile script sets the mime type based upon file name suffix. Unknown types are served as text/plain which can result in corrupt files.

Controlling who can manage top-level webs

Top level webs are a special case, because they don't have a parent web with a WebPreferences. So there has to be a special control just for the root level.

- You can define these settings in the Main.TWikiPreferences topic, preferable towards the end of the topic:
  - Set DENYROOTCHANGE = < comma-delimited list of Users and Groups >
  - Set ALLOWROOTCHANGE = < comma-delimited list of Users and Groups >

Note that you do not require ROOTCHANGE access to rename an existing top-level web. You just need WEBCHANGE in the web itself.

Empty values in access control variables
How TWiki evaluates ALLOW/DENY settings

When deciding whether to grant access, TWiki evaluates the following rules in order (read from the top of the list; if the logic arrives at **PERMITTED** or **DENIED** that applies immediately and no more rules are applied). You need to read the rules bearing in mind that VIEW, CHANGE and RENAME access may be granted/denied separately.

1. If the user is an administrator  
   ♦ access is **PERMITTED**.
2. If DENYTOPIC is set to a list of wikinames  
   ♦ people in the list will be **DENIED**.
3. If DENYTOPIC is set to **empty** (i.e. Set DENYTOPIC = )  
   ♦ the access control setting is **ignored**.  
   ▲ **Attention:** The spec changed in TWiki-6.0; access was permitted in earlier TWiki releases.
4. If ALLOWTOPIC is set  
   1. people in the list are **PERMITTED**  
   2. everyone else is **DENIED**  
5. If DENYWEB is set to a list of wikinames  
   ♦ people in the list are **DENIED** access  
6. If ALLOWWEB is set to a list of wikinames  
   ♦ people in the list will be **PERMITTED**  
   ♦ everyone else will be **DENIED**  
7. If you got this far, access is **PERMITTED**

Allowing web creation by user mapping manager

There are cases where DENYROOTCHANGE, ALLOWROOTCHANGE, DENYWEBCHANGE, and ALLOWWEBCHANGE are not capable enough to implement web creation permission you want. To cope with such cases, when a new web is created, the `canCreateWeb($cUID, $web)` method of the user mapping manager is called if the method exists. If it returns true, TWiki goes ahead and create the web without checking access control variables. Please read AllowWebCreateByUserMappingManager for more details.

Forbid certain users to do certain actions by configuration

You may have an unruly registered users (e.g. a crawler program) who don't follow the rules while you don't have control over such users. And the web application container in which TWiki is installed may be managed by somebody else and you don't have tight and quick control.

To cope with such situations, certain users can be forbidden certain scripts by setting `{ForbidUserAction}`. A good example is worth more than a lengthy explanation, so here it is:

```
$TWiki::cfg{ForbidUserAction} = '  
   AggresiveCrawler: edit, oops, search;  
   ReadOnlyUser: !view, viewfile;  
   TotallyForbidden: !nothing;  
';
```

In this example:

- AggresiveCrawler is forbidden edit, oops, and search scripts.
- ReadOnlyUser is permitted view and viewfile but forbidden the others.
  ♦ If a script list is preceded by !, only the listed scripts are permitted for the user. ! at the beginning of the list negates the list.
TotallyForbidden is forbidden all actions.

Here's the logic. There is no script named nothing, which means all scripts don't match "nothing", hence all scripts are forbidden.

And here are some rules:

- Spaces, tabs, new lines are ignored
- It consists of semicolon separated list of per-user specifications
- A specification consists of a user name, colon, and a comma separated list of scripts
- A user name needs to be in the canonical form. In most cases the canonical user name is the same as the wiki name. But if you are using non-standard user mapping, the canonical user name of a user is different from the wiki name.

User masquerading

There are cases where it's handy to access TWiki on behalf of somebody else retaining a trace of your real identity rather than completely becoming a different user. We call it user masquerading. TWiki provides a framework to implement that. Please read UserMasquerading for more information.

This is an advanced feature and not many TWiki sites are using, but there is a part in the following section mentioning it, it's mentioned here.

Dynamic access control

There are pitfalls and you need to harden your web to avoid unexpected access. Before using this feature, please read this entire section through carefully.

You may want to restrict access dynamically -- based on topic name, a form field value, or some combination of factors. To cope with such situations, the dynamic access control mechanism is provided. If you set `DYNAMIC_ACCESS_CONTROL` on at WebPreferences of the web, TWiki variables in access control variables mentioned above are expanded.

Example 1 - restriction based on topic name

Let's assume you need to restrict changes only to the CroniesGroup members except with topics whose name ends with Public, which need be changed by anybody. That is achieve by the following settings on WebPrefences.

* Set `DYNAMIC_ACCESS_CONTROL = on`
* Set `ALLOWWEBCHANGE = %IF{"%CALCULATE{$SUBSTRING(%TOPIC%, -6, 6)}%' = 'Public'" then="%WIKINAME%" else="CroniesGroup"}%`

Example 2 - restriction based on form field

Let's assume:

- a web storing requests on topics whose name starts with ReqEnt
- Each request topic has a form field "Requestor", which has the wiki name of the requestor
- Users can view only requests they created
- The members of the SupportGroup mail group can view all requests

That is achieve by the following settings on WebPreferences.

* Set `DYNAMIC_ACCESS_CONTROL = on`
* Set `ALLOWWEBVIEW = %IF{"%CALCULATE{$SUBSTRING(%TOPIC%, 1, 6)}%' = 'ReqEnt' and '%FORMFIELD{Requestor}%' != '%WIKINAME%" then="SupportGroup" else="%WIKINAME%"}%`
Specifically the following access control variables are subject to TWiki variable expansion in their values.

- DENYTOPIC* (e.g. DENYTOPICVIEW, DENYTOPICCHANGE)
- ALLOWTOPIC*
- DENYWEB*
- ALLOWWEB*

DENYROOT* and ALLOWROOT* are not subject to variable expansion. Because there has been no good use cases presented.

**Dynamic access control in accessing a different web's topic**

Let's assume WebA has the following lines on WebPreferences.

```plaintext
* Set DYNAMIC_ACCESS_CONTROL = on
* Set MEMBERS = JaneSmith, JoeSchmoe
* Set ALLOWWEBVIEW = %MEMBERS%
```

This is not a good way to use dynamic access control but it does restrict access only to those listed in MEMBERS. However, access control doesn't work as expected when WebA.TopicB is accessed from WebC.TopicD by `%INCLUDE{WebA.TopicB}%` or other variables. This is because `%MEMBERS%` is defined in WebA and may have a different value in other webs.

You may think the following lines cheat the access control on WebA but actually not.

```plaintext
* Set MEMBERS = %WIKINAME%
%INCLUDE{WebA.TopicB}%
```

This is because when a topic (e.g. WebC.TopicD) is accessed from browser and the topic refers to another topic in a different web (e.g. WebA.TopicB) and the different web employs dynamic access control, access to another topic is defined being on the safer side.

**Topic level dynamic access control**

On a topic, it's possible to use a variable defined on the topic for topic level access restriction. E.g.

```plaintext
* Set MEMBERS = JaneSmith, JoeSchmoe
* Set ALLOWTOPICVIEW = %MEMBERS%
```

[This is not a good way to use dynamic access control]

**Dynamic access control and user masquerading**

Your user mapping handler may be providing the UserMasquerading feature. In that case, you expect dynamic access control to just work when user masquerading is in effect. Otherwise, you cannot test if your dynamic access control configuration is working as expected on your own.

Dynamic access control does work as expected even if user masquerading is in effect. For that, the following things are happening under the hood.

Let's think about Example 2 mentioned above. When you masquerading as SomebodyElse, you need to be able to see SomebodyElse's requests only. In the access control setting, a form field value is compared with `%WIKINAME%`. While user masquerading is in effect, your wiki name is YourNameOnBehalfOfSomebodyElse. It cannot match the form field value.
To make dynamic access control work under these circumstances, variable expansion for dynamic access
control is skewed as follows. Specifically, the following variables are expanded to the value of SomebodyElse's
rather than YourNameOnBehalfOfSomebodyElse's.

- WIKINAME
- USERNAME
- WIKIUSERNAME

Avoiding vulnerability

By default, user level preferences are read before web level preferences. This means a user can set a
preferences variable at the user level and finalise it. To prevent this sort of attack, you need to harden your
web or site by disabling user preferences by e.g. having the following line on lib/LocalSite.cfg

```
TWiki::cfg{DemoteUserPreferences}= 1;
```

and having the following line on your WebPreferences and then finalise DENYUSERPREFERENCES.

```
* Set DENYUSERPREFERENCES = all
```

Please read TWikiVariables#ControllingUserLevelPrefsOverride for details.

Again by default, predefined variables such as `%IF{...}%` can be overridden by preferences variables. If user
preferences are disabled, ordinary users cannot attack using user preferences, but topic level preferences may
cause unexpected consequences. As such, all predefined variables need to be made un-overridable by having
the following line on WebPreferences and then finalise OVERRIDABLEPREDEFINEDVARIABLES.

```
* Set OVERRIDABLEPREDEFINEDVARIABLES =
```

Please read TWikiVariables#PredefinedVariables for details.

Disabling dynamic access control

You may not be comfortable with dynamic access control because it may slow things down. Or you may not
want to be bothered by questions raised by users about it. If so, you can disable it by setting
DYNAMIC_ACCESS_CONTROL 'off' and then finalizing at the local site level. (cf.
TWikiVariables#Setting_Preferences_Variables)

Access control and INCLUDE

ALLOWTOPIVIEW and ALLOWTOPICHANGE only applies to the topic in which the settings are
defined. If a topic A includes another topic B, topic A does not inherit the access rights of the included topic B.

Examples: Topic A includes topic B

- If the included topic B has ALLOWTOPICHANGE set to block editing for a user, it does not
prevent editing the including topic A.
- If the included topic B has ALLOWTOPIVIEW set to block view for a user, the user can still view
topic A but he cannot see the included topic B. He will see a message No permission to view B
Customizing "access denied" message

When access is denied, a page as follows is displayed:

You may want to customize the passage annotated in the red rectangle. For example, with a web restricting access, you may want to show the link to an access request form.

You can achieve that by setting \texttt{TOPIC\_ACCESS\_CONTACT} variable on WebPreferences. e.g.

* Set \texttt{TOPIC\_ACCESS\_CONTACT} = If you need to access this site, please apply \[Main\\_Access\_Form\] here

Please note that setting it on a topic other than WebPreferences does not take effect. This is a limitation of the current implementation.

Custom user/group notations

You can have custom user/group notations such as \texttt{USER:userid} and \texttt{LDAPGROUP:group-name} and use them for access control. For example:

* Set \texttt{ALLOWWEBCHANGE} = USER:buzz, LDAPGROUP:foo-bar

In a large organization, TWiki may need to depend on user and group data provided by its infrastructure. Custom user/group notations are handy in such situations though it's not trivial to implement. Please read here for details.

Access Control quick recipes
Restrict Access to Whole TWiki Site

In a firewalled TWiki, e.g. an intranet wiki or extranet wiki, you want to allow only invited people to access your TWiki. There are three options:

1. **Install TWiki Behind Firewall:**

   The firewall takes care of giving access to TWiki to authorized people only. This is a typical setup for a company wiki. As for TWiki configuration, no special setup is needed.

2. **Extranet TWiki Using Template Login:**

   All TWiki content (pages and attachments) need to be access controlled. The Template Login allows users to login and logout. Only logged in users can access TWiki content.

   **Configuration:** Follow the default setup, then change these configure settings:

   - Secure attachments as documented. The TWiki:TWiki.ApacheConfigGenerator is useful to get the setting right.
   - Require authentication for all TWiki scripts except backuprestore, configure, login, logon and resetpasswd with the following configure setting:
     
     ```
     $TWiki::cfg{AuthScripts} = 'attach, changes, edit, manage, oops, preview, rdiff, rdiffauth, register, rename, rest, save, search, twiki_cgi, upload, statistics, view, viewauth, viewfile';
     ```
   - When you install additional plugins make sure to add scripts they might introduce also to `twiki/bin` also to the `AuthScripts` configure setting.

   **Attention:** Some scripts of additional plugins might not be aware of TWiki's template login. Test all new scripts with a non-authenticated user!

3. **Extranet TWiki Using Apache Login:**

   All TWiki content (pages and attachments) need to be access controlled. The Apache Login does not offer a logout; typically the browser needs to be restarted to logout. Only logged in users can access TWiki content.

   **Configuration:** Enable user authentication with ApacheLogin and lock down access to the whole `twiki/bin` and `twiki/pub` directories to all but valid users. In the Apache config file for TWiki (`twiki.conf` or `.htaccess`), replace the `<FilesMatch ".*"> require valid-user </FilesMatch>` section with this:

   ```
   <FilesMatch ".*"> 
     require valid-user 
   </FilesMatch>
   ```

   **Notes:**

   - In all three options, content can be restricted selectively with ALLOWWEBVIEW and other access control settings documented above. See also the next quick recipe.
   - In the extranet setup, someone with access to the site needs to register new users. If you still want public users to be able to register automatically follow TWiki:TWiki.RegisterOnViewRestrictedSite.

**Authenticate and Restrict Selected Webs Only**

Use the following setup to provide unrestricted viewing access to open webs, with authentication only on selected webs. Requires TWikiUserAuthentication to be enabled.
1. **Restrict** view access to selected Users and Groups. Set one or both of these variables in its WebPreferences topic:

- Set DENYWEBVIEW = < list of Users and Groups >
- Set ALLOWWEBVIEW = < list of Users and Groups >

*Note:* DENYWEBVIEW is evaluated before ALLOWWEBVIEW. Access is denied if the authenticated person is in the DENYWEBVIEW list, or not in the ALLOWWEBVIEW list. Access is granted if DENYWEBVIEW and ALLOWWEBVIEW are not defined.

### Hide Control Settings

*Tip:* To hide access control settings from normal browser viewing, you can put them into the *topic preference settings* by clicking the link *Edit topic preference settings under More topic actions* menu. Preferences set in this manner are not visible in the topic text, but take effect nevertheless. Access control settings added as topic preference settings are stored in the topic meta data and they override settings defined in the topic text.

Alternatively, place them in HTML comment markers, but this exposes the access setting during ordinary editing.

```html
<!--
* Set DENYTOPICCHANGE = Main.SomeGroup
-->
```

### Obfuscating Webs

Another way of hiding webs is to keep them hidden by not publishing the URL and by preventing the *all webs* search option from accessing obfuscated webs. Do so by enabling the NOSEARCHALL variable in WebPreferences:

- Set NOSEARCHALL = on

This setup can be useful to hide a new web until content its ready for deployment, or to hide view access restricted webs.

*Note:* Obfuscating a web without view access control is very insecure, as anyone who knows the URL can access the web.

### Read-only Skin Mode

It is possible to turn the PatternSkin and TopMenuSkin into read-only mode by removing the edit and attach controls (links and buttons). This is mainly useful if you have TWiki application pages or dashboards where you do not want regular users to change content. The read-only skin mode is not a replacement for access control; you can use it in addition to access control. Details at PatternSkinCustomization#ReadOnlySkinMode.

### Configuring access control for topics of a certain name in all webs

You may need to restrict access to topics of a certain name in all webs. For example, there might be an add-on referring to a certain topic of all webs. And the add-on does things only administrators are supposed to do. In that case, change to the topic needs to be restricted only to administrators and must not be overridable.
Let's say there is AutomationAddOn which refers to WebAutomation of all webs. And WebAutomation needs to be modifiable only by administrators. That can be achieved by the following configuration.

```
$TWiki::cfg{Access}{Topic}{WebAutomation} = {
   DENYCHANGE => 'Main.AllUsersGroup',
};
```

In addition to ALLOWCHANGE, you can specify DENYCHANGE, ALLOWVIEW, DENYVIEW, ALLOWRENAME, and DENYRENAME as follows.

```
$TWiki::cfg{Access}{Topic}{SpecialTopic} = {
   DENYVIEW  => 'JoeSchmoe',
   ALLOWVIEW => 'FooGroup',
};
```

$TWiki::cfg{Access}{Topic}{TOPICNAME} has precedence over DENYTOPIC* and ALLOWTOPIC*. For example, if the configuration for WebAutomation is there as above, there is no way to allow non-administrators to change the WebAutomation topic of any web.

As a way to configure access control, this may look crude. The reason why configured this way is that this can be part of plugin/add-on/contrib's configuration. For example, Config.spec of AutomationAddOn would have the following lines, with which proper access control to WebAutomation topics is implemented without the administrator knowing it.

```
$TWiki::cfg{Access}{Topic}{WebAutomation} = {
   DENYCHANGE => 'Main.AllUsersGroup',
};
```


This topic: TWiki > TWikiAccessControl
Topic revision: r43 - 2015-03-30 - TWikiContributor