

Best practices for using fine-grained permissions

This document is provided "as-is". Information and views expressed in this document, including URL and other Internet Web site references, may change without notice. You bear the risk of using it.

Some examples depicted herein are provided for illustration only and are fictitious. No real association or connection is intended or should be inferred.

This document does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this document for your internal, reference purposes.

© 2010 Microsoft Corporation. All rights reserved.

Best practices for using fine-grained permissions

**Writer**: Sean Livingston, Product Manager, SharePoint Product Group  
**Published**: September 2010

Summary: This paper describes best practices for fine-grained permissions (FGP) and how to use them within your organization when using SharePoint® Products and Technologies or Microsoft® SharePoint® 2010 Products.

# Contents

[Contents 3](#_Toc271117838)

[Overview of using fine-grained permissions 4](#_Toc271117839)

[SharePoint permission system overview 5](#_Toc271117840)

[Permission levels 5](#_Toc271117841)

[SharePoint groups 5](#_Toc271117842)

[Scope 5](#_Toc271117843)

[Securable object 6](#_Toc271117844)

[Inheritance 6](#_Toc271117845)

[Limited access 7](#_Toc271117846)

[Binary ACL 8](#_Toc271117847)

[Best practices for avoiding common FGP limit issues 9](#_Toc271117848)

[Too many scopes within a list 9](#_Toc271117849)

[Too many members within a scope 10](#_Toc271117850)

[Very deep scope hierarchy 10](#_Toc271117851)

[Recommended solutions for common FGP performance issues 11](#_Toc271117852)

[Solution 1: Remove FGP and use security enforcement only at Web level 11](#_Toc271117853)

[Environmental security cleanup 11](#_Toc271117854)

[Environmental security architecture redesign 12](#_Toc271117855)

[Solution 2: Use fine-grained permissions by hierarchical structure changes 13](#_Toc271117856)

[Environment hierarchy redesign 13](#_Toc271117857)

[Solution 3: Use fine-grained permissions by scope structure changes (2010 only) 14](#_Toc271117858)

[Dynamic security changing code redesign 14](#_Toc271117859)

[Environment architecture example 15](#_Toc271117860)

[Environment overview 15](#_Toc271117861)

[Workflow design 16](#_Toc271117862)

[Fine-grained permission issues 17](#_Toc271117863)

[Resolution of FGP issues 18](#_Toc271117864)

[Summary 20](#_Toc271117865)

# Overview of using fine-grained permissions

This article describes the use of fine-grained permissions (FGP) for SharePoint® 2010 Products (Microsoft® SharePoint® Server 2010 and Microsoft® SharePoint® Foundation 2010) and SharePoint® Products and Technologies (Office SharePoint® Server 2007 and Windows SharePoint® Services version 3.0); performance issues related to FGP; and best practices for configuring solutions that include FGP.

**Note:** We recommend that you use FGP for only those business cases for which it is required. FGP can be expensive in terms of both operational oversight and performance.

You can avoid the use of FGP by doing the following:

* Break permission inheritance as infrequently as possible.
* Use groups based on directory membership to assign permissions.

**Note**: We do not recommend that you use SharePoint groups to assign permissions to sites, because when a SharePoint group is used to assign permissions, a full crawl of the index occurs. Instead, we recommend Domain groups to be used.

* Assign permissions at the highest possible level. As part of this strategy, consider the following techniques:
  + Segregate documents that require fine-grained permissions into document libraries that are defined to support each group of permissions, and keep the document libraries in a segregated site collection or site. For additional information about hierarchical changes, see [Solution 2: Use fine-grained permissions by hierarchical structure changes](#_Solution_2:_Use).

**Note:** In this document, the terms *Web* and *site* equate to the **SPWeb** object, and *site collections* equate to the **SPSite** object.

* + Use different document publish levels to control access. Before a document is published, the advanced permissions and versioning settings can be set for users who can only approve items in the document library.
  + For non-document libraries (lists), use the ReadSecurity and WriteSecurity permission levels. When a list is created, the owners can set the Item-level permissions to either Read access or Create and Edit access.

If you must use fine-grained permissions, consider the following recommended practices:

* Ensure that you do not have too many items at the same level of hierarchy in the document libraries, because the time necessary to process items in the views increases.
* Use event handlers to control edit permission. You can have an event handler that registers an event using the **SPEventReceiverType.ItemUpdating** and **SPEventReceiverType.ItemUpdated** methods, and then use code to control whether the update should be allowed. This is extremely powerful, because you can make security decision based on any metadata of a list or item, without affecting the view rendering performance. For additional information about event handlers, see [Resolution of FGP issues](#_Resolution_of_FGP).
* Use **AddToCurrentScopeOnly** method to assign Limited Access membership within a SharePoint group. The key element in this principle is to redesign the architecture so that scope membership does not cause ACL recalculation at the parent document library and Web. For additional information about scope changes, [Solution 3: Use fine-grained permissions by scope structure changes (2010 only)](#_Solution_3:_Use).

# SharePoint permission system overview

This section describes the SharePoint permissions scope system. For more information about planning site security, see:

* [Plan site permissions (SharePoint Server 2010)](http://technet.microsoft.com/en-us/library/cc262778.aspx) (http://technet.microsoft.com/en-us/library/cc262778.aspx)
* [Plan site permissions (SharePoint Foundation 2010)](http://technet.microsoft.com/en-us/library/cc287752.aspx) (http://technet.microsoft.com/en-us/library/cc287752.aspx)
* [Plan site security (Office SharePoint Server)](http://technet.microsoft.com/en-us/library/cc262778(office.12).aspx) (http://technet.microsoft.com/en-us/library/cc262778(office.12).aspx)
* Plan site security (Windows SharePoint Services) <http://technet.microsoft.com/en-us/library/cc287752(office.12).aspx> (http://technet.microsoft.com/en-us/library/cc287752(office.12).aspx)

## Permission levels

A permission level contains a set of individual permissions, for example, View Items or Create Alerts. Permission levels can be predefined or created by the user. The set of permissions can be modified even within the predefined permission levels.

## SharePoint groups

A SharePoint group is a site collection-wide object that can hold other security principals, including Windows user accounts, non-Windows users (for example, forms-based accounts), and Active Directory groups.

## Scope

A scope is the security boundary for a securable object and any of its children that do not have a separate security boundary defined. The scope contains an Access Control List (ACL), but unlike NTFS ACLs, a scope can include SharePoint-specific security principals. The members of an ACL for a scope can include Windows users, non-Windows users (such as forms-based accounts in SharePoint Products and Technologies or claims-based accounts in SharePoint 2010 Products), Active Directory groups, or SharePoint groups.

There is no maximum number of scopes that can be created within a parent scope. However, in SharePoint Products and Technologies, after 1,000 scopes have been created, a code path that requires additional Microsoft® SQL Server roundtrips to analyze the scopes before rendering a view is used. When there 1,000 or fewer scopes, only one roundtrip is required. In SharePoint 2010 Products, the limit of number of scopes returned before switching to a different algorithm is based on a query throttle limit, with a default value of 5,000; however this value even at default can be large enough to significantly detract from performance.

In SharePoint 2010 Products, there is a new method called **SPRoleAssignmentCollection.AddToCurrentScopeOnly**, by which role assignment can occur. For additional inforamation about role assignments, see [SPRoleAssignmentCollection.AddToCurrentScopeOnly](http://msdn.microsoft.com/en-us/library/microsoft.sharepoint.sproleassignmentcollection.addtocurrentscopeonly.aspx) (http://msdn.microsoft.com/en-us/library/microsoft.sharepoint.sproleassignmentcollection.addtocurrentscopeonly.aspx).

## Securable object

A securable object is an object that can have an ACL assigned to it. In SharePoint Products and Technologies, the **ISecurableObject** interface can be used, and in SharePoint 2010 Products, the **SPSecurableObject** class should be used. For additional information about securable objects, see [ISecurableObject interface](http://msdn.microsoft.com/en-us/library/microsoft.sharepoint.isecurableobject.aspx) (http://msdn.microsoft.com/en-us/library/microsoft.sharepoint.isecurableobject.aspx or [SPSecurableObject class](http://msdn.microsoft.com/en-us/library/microsoft.sharepoint.spsecurableobject.aspx) (http://msdn.microsoft.com/en-us/library/microsoft.sharepoint.spsecurableobject.aspx).

## Inheritance

If a securable object does not have a unique scope, the object inherits the scope of its parent. When an object inherits from its parent, no scope is created for the object. Instead, whenever a security check is made, it verifies only against the parent object. In the simplest environment, this scope is at the root Web of the site collection that contains the item. When an item or container is changed to have unique membership, its inheritance is broken, which means that a new scope is created for that item and, by default, for any of its children that inherit its permission scopes.

The following diagram shows an object hierarchy for a document library, in which all objects but one inherit their scope from their parents. Each numbered gold hexagon represents a permissions scope. All child objects within a container inherit from that parent scope unless they have their own unique permissions scope.



## Limited access

When a security principal is added to the scope of an item with unique permissions, the security principal is immediately added with the Limited Access permission level to each unique permission scope in the hierarchy above the item until a parent Web with unique permissions is located.

The reason for adding the user to the scopes with Limited Access is to allow enough access to the object hierarchically above the uniquely permissioned item so that the Object Model (OM), master pages, and navigation can render when the user attempts to navigate to the item. Without the Limited Access permissions at the parent scopes, the user would not be able to successfully navigate to or open the item that has unique permissions.

The following diagram shows how the hierarchical depth of scopes can affect the amount of work required to add Limited Access users to parent scopes. The larger the number of unique scopes above the item, up to and including the uniquely permissioned Web, the larger the number of additions that must occur. The diagram shows a simplified representation of a physical structure that has unique scopes defined at every level from the Web down to individual items. As in the previous diagram, each differently numbered gold hexagon represents a unique permission scope, and all child objects within that container inherit from that scope unless they have their own unique permissions scope. The chain of Limited Access promotion is shown using red arrows.



The diagram also includes the set of unique scopes along with the Limited Access membership additions that must occur on each parent scope, represented by separate boxes within the scope. No additional programming is required to add unique scopes whenever a security principal is added to an object scope with unique permissions that is below a Web with unique permissions.

When a security principal with the Limited Access permission level is added to a parent scope, no check is made to see whether the security principal is already in the parent scope. A security principal that already has access to the parent scope is added again with Limited Access permissions, regardless of its existing permissions on the parent scope.

When a security principal is removed from the Limited Access permission level at a parent scope, each instance of that security principal within every child scope is removed from the Limited Access permission level, regardless of whether the security principal has Limited Access or a wider set of permissions at the child scopes.

## Binary ACL

A binary ACL performs rapid comparisons of a user token to determine whether the user should have access to the object covered by the scope. Whenever the membership of a scope changes, a binary ACL is calculated, including when a new limited access member is added. The binary ACL takes more time to calculate as the membership gets larger, and access to the objects will be blocked until the ACL can be recalculated.

Although there is no explicit size limitation on a binary ACL other than the maximum size of an image column in SQL Server, some services cannot accept an ACL that is larger than 64KB. In this case, the number of security principals in the binary ACL may be able to grow very large, but should be limited due to performance and interoperability considerations. For information about limitations in image column sizes in SQL Server, see [ntext, text, and image (Transact-SQL)](http://msdn.microsoft.com/en-us/library/ms187993.aspx) (http://msdn.microsoft.com/en-us/library/ms187993.aspx).

# Best practices for avoiding common FGP limit issues

When working fine-grained permissions, it is easy to unintentionally encounter limits that prevent permissions from resolving.

## Too many scopes within a list

There is a built-in limit of 50,000 scopes per list or document library. After 50,000 scopes are reached addition of new scopes within a given list or document library is prohibited.

In SharePoint 2010 Products, the built-in scope limit can be modified by using a Windows PowerShell script.

**To modify the built-in scope limit to less than 50,000 scopes**

1. Verify that you meet the following minimum requirements: See [Add-SPShellAdmin](http://technet.microsoft.com/en-us/library/ff607596.aspx) (http://technet.microsoft.com/en-us/library/ff607596.aspx).
2. On the **Start** menu, click **All Programs**.
3. Click **Microsoft SharePoint 2010 Products**.
4. Click **SharePoint 2010 Management Shell**.
5. At the Windows PowerShell command prompt, type the following syntax:

$webapp = Get-SPWebApplication <http://serverName>

$webapp.MaxUniquePermScopesPerList

$webapp.MaxUniquePermScopesPerList = <Number of scope limit>

Often, however, the effective limit is much smaller than 50,000 if many scopes exist at the same hierarchical level. This is because display checks for items below that hierarchical level must be checked against all scopes above them. This limitation can cause the effective number of scopes allowed in a particular query to be reduced to 1,000 to 2,000.

**Best practices**:

* Only set unique scopes on parent objects such as folders.
* Do not create a system with many uniquely permissioned objects below an object that has many scopes.

If your business requires that you more than 50,000 uniquely permissioned items in a list or document library, then you must move some items to adifferent list or document library.

## Too many members within a scope

As described earlier, a binary ACL is calculated whenever the membership of that scope changes, including when a new limited access member is added. As the scope membership number increases, the amount of time it takes to recalculate the binary ACL increases.

However, the problem can be made worse as the additions of users at a child objects unique scope will cause its parent scopes to be updated with the new Limited Access members, even if this ultimately results in no change to the parent scope membership. When this occurs, the binary ACL for the parent scope(s) must also be recalculated, at the expense of more processing time even if it ultimately results in the same ACL.

**Best practice**:

Rely on group membership instead of indivudal user membership in the scopes. For example, if a single group can be used in place of 1,000 users, the scope will be 999 membership entries smaller for the scope and any of its parent scopes which will be updated with Limited Access rights for that single group instead of all 1,000 individual users with Limited Access rights. This additionally helps increase the speed of Limited Access rights push and ACL recalculation at the parent scope objects.

**Important:** Using a SharePoint group will cause a full crawl of the index. If possible, use a domain group.

## Very deep scope hierarchy

As indicated earlier, hierarchical depth of scopes can affect the amount of work required to add Limited Access users to parent scopes. The larger the number of unique scopes above an item, up to and including the uniquely permissioned Web, the larger the number of additions that must occur. If a scope hierarchy is very deep, a scope membership change can take a very long time to occur, as each membership change in the deepest scope item will have to iteratively update parent scopes with a membership addition for the explicitly added user or group with Limited Access rights. Additionally this will increase the number of binary ACLs that need to be recalculated, with an according performance impact.

**Best practice:**

Reduce the numbers of uniquely permissioned parent objects, thereby reducing the numer of scopes that need to be updated with Limited Access members whenever any child objects scope changes.

# Recommended solutions for common FGP performance issues

The following solutions can help mitigate performance issues that are specifically related to the extensive use of fine-grained permissions. Each of the following covers changes to the environment security, object hierarchy or custom code that is contributing tho the FGP related performace issue. Each solution will start with the following example environment where a single Web contains multiple document libraries each with a great many number of uniquely permissioned child objects.



## Solution 1: Remove FGP and use security enforcement only at Web level

To re-architect the environment so it no longer requires fine-grained permissions, an environment cleanup process can be implemented, and then the number of scoped items can be adjusted to improve the scalability of the environment over the longer term. The following recommendations describe the environment cleanup and architectural security changes required to accomplish this solution.

### Environmental security cleanup

When a user is removed from the Web-level scope, the internal OM must remove the user from every scope below the Web level. However, removing individual users in order to clean up existing permissions is a time-consuming process. Instead, first remove each of the individual item-level unique scopes so that the item is set to inherit permissions from its parent object. This will take comparatively less time than attempting to remove users first, because it has to act on only a single scope for the item.

**Important:** If the current Web is not at the root of the site collection, and if it is then set to inherit its permissions from its parent Web, all the unique scopes under it will be removed, and all the Limited Access memberships will be overwritten at once using in a single SQL Server roundtrip.



After all item-level scopes have been removed, individual scope memberships at the Web-level scope can be replaced with one or more group memberships to allow access.



### Environmental security architecture redesign

After the existing fine-grained permissions and scopes are removed, the long-term architecture plan should be to maintain a unique scope only at the Web level. The following diagram shows how this could be structured so that only the Web-level scope remains. The core requirement in the architecture is to not have too many items at the same level of hierarchy in the document libraries, because the time necessary to process items in the views increases. As a best practice, the maximum count of items or folders at any level in the hierarchy should be roughly 2,000 items.



If additional changes are needed to the architecture, consider moving document libraries to different Webs or site collections. The number of document libraries could also be changed to more closely support business needs and scaling recommendations that are based on the taxonomy or audience of the stored content.

## Solution 2: Use fine-grained permissions by hierarchical structure changes

To re-architect the environment so it still uses requires fine-grained permissions, but without causing excessive updates to or or sizing of a single Web scope, consider moving differently secured document libraries to different Webs.

### Environment hierarchy redesign

In the following diagram, the physical architecture has been modified so that each document library is in a uniquely permissioned Web. Additionally, when item-level FGP must be preserved, as a best practice the cumulative number of security principals who will be granted access should be limited to approximately 2,000, although this is not a fixed limit. As such, the effective membership of each Web, including all Limited Access members users, should be no more than approximately 2,000 users in order to keep each Web-level scope from growing too large.



The number of uniquely scoped children is not a significant issue, and can scale to large numbers, but the number of principles that will be added as limited access up the chain of scopes to the first uniquely permissioned Web will be a limiting factor.

Lastly, althoiugh not specifically an FGP issue, the folder structure should ensure that no single hierarchical level of the document library ever exceeds roughly 2,000 items. This limit can help ensure good performance of views requested by users.

## Solution 3: Use fine-grained permissions by scope structure changes (2010 only)

To re-architect the environment so it still uses requires fine-grained permissions, but without causing excessive updates to or or sizing of a single Web scope, consider using a different process of securing items. This is mainly applicable if the cause of the excessive number of unique scopes was through an automated process such as an event handler or workflow that dynamically modified object permissions. The recommendation in this case is to make a code change to whatever process was creating the unique security scopes.

### Dynamic security changing code redesign

In the following diagram, the scope architecture has been modified so that scope membership does not cause ACL recalculation at the parent document library and Web. As mentioned earlier, the effective membership of the Web, including all Limited Access members, should be no more than approximately 2,000 in order to keep the Web-level scope from growing too large. In this case, however, by implementing a new SharePoint group to hold all members who should have Limited Access rights, the scope will not grow too large. When users are added to individual scopes under the Web level, using the new SharePoint 2010 Products **SPRoleAssignmentCollection.AddToCurrentScopeOnly** method, they can also then be added, by additional code, to the new group that has already been established as having Limited Access rights at the Web and document library level.



As mentioned earlier, when item-level FGP must be preserved, as a best practice the cumulative number of security principals who will be granted access should be limited to approximately 2,000, although this is not a fixed limit. As such, when this number increases, the amount of time it takes to recalculate the binary ACL increases. If the membership of a scope is changed, the binary ACL must be recalculated. However, the additions of users at a child item unique scope will cause parent scopes to be updated with the new Limited Access members, even if this ultimately results in no change to the parent scope membership. When this occurs, the binary ACL for the parent scope(s) must also be recalculated.

As in the previous solution, the number of uniquely scoped children is not a significant issue, and can scale to large numbers, but the number of principles that will be added as limited access up the chain of scopes to the first uniquely permissioned Web will be a limiting factor.

# Environment architecture example

This section describes an example environment that was experiencing significant issues related to a confluence of fine-grained permissions related issues, and covers the combination of solutions used to fix the issue.

## Environment overview

A knowledge management system based on SharePoint Server 2007 contained two site collections each with a single Web, Contoso-Draft and Contoso-Production. Contoso-Draft was where initial drafts were published and where workflows interacted with the documents. Contoso-Production was the final destination of each approved document, and was the repository for all approved content.

Documents could be assigned to one of multiple content types that convey the intended purpose of the document (such as project plans or troubleshooting guides). Additionally the documents were classified within technology domains (of which there couild be a hundred or more of increasing specificity), and for various disciplines (such as project management or operations). The draft publishing site collection contained one document library per discipline, each with a hierarchy of increasingly specifc folders for each technology domain, and users were expected to first select into a discipline library and specific technology domain folder when creating a new document.

The following diagram shows a simplified representation of the original physical structure of the Web, where each uniquely numbered gold hexagon represents a unique permissions scope, and all child objects within that container inherit from that same scope unless they have their own unique permissions scope.



Each combination of content type, technology domain, and discipline could have a non-overlapping reviewer assigned who was an expert in the technology domain or discipline. The document library was expected to hold a large number of items while they were undergoing workflow operations which dynamically changed the assigned reviewer and security of the item. Once the document was final reviewed, it was then copied to a matching Contoso-Production based location where it remained unmodified as a published version, and available to all company employees.

For information about content type and workflow planning, see:

* [Content type and workflow planning (SharePoint Server 2010)](http://technet.microsoft.com/en-us/library/cc262735.aspx) (http://technet.microsoft.com/en-us/library/cc262735.aspx)
* [Content types planning (SharePoint Foundation 2010)](http://technet.microsoft.com/en-us/library/ff607870.aspx) (http://technet.microsoft.com/en-us/library/ff607870.aspx)
* [Plan content types (Office SharePoint Server)](http://technet.microsoft.com/en-us/library/cc262735(office.12).aspx) (http://technet.microsoft.com/en-us/library/cc262735(office.12).aspx)
* [Plan content types (Windows SharePoint Services)](http://technet.microsoft.com/en-us/library/cc287765(office.12).aspx) (http://technet.microsoft.com/en-us/library/cc287765(office.12).aspx)

## Workflow design

When the workflow process began, the author of a document was blocked from accessing it so others could review it without the author making changes at the same time. For each succeeding step of the workflow, the users who previously had access to the document were denied access and the reviewer(s) for the next stage of the workflow were given access.

The workflow process used both a coded workflow and a custom event handler, which worked together. When an item was changed in a document library, it was initially acted upon by the custom event handler which would change permissions and start a new workflow instance. Both the workflow and the event handler changed the permissions for the specific file being updated, so that each item was given a unique permissions scope. This permission change meant that only a single user or small subset of users—that is, the reviewers for that step, had access to the item at a time. The final step in the workflow, once the document was fully approved, was to copy it to the equivalent Contoso-Production location as a new published version of the document, with permissions inheriting from the parent Web.

## Fine-grained permission issues

The environment and workflow design tested well during development, but is now experiencing signficant issues in performance, with users experiencing delays from one to dozens of minutes before tasks can be accomplished. The testing used only hundreds of test accounts, but once the design was made available and and then announced as a mandatory knowledge capture tool for the entire company, usage quickly grew to greater than 15,000 users cumulatively working on over 30,000 documents. The performance issues reported prevented a large portion of the company from being able to use the new knowledge management system which was expected to support upwards of 60,000 users.

When the permission changes happened through the workflow, a permission scope was created for each individual item. Following the requirements of the Limited Access permission level as described previously, each unique security principal was added with Limited Access to the various unique permission scopes in the hierarchy above the item until a uniquely permissioned Web was located. Therefore, the more unique scopes that were above the uniquely permissioned item but below the uniquely permissioned Web, the more scopes that the security principal was added to with Limited Access.



A key thing to be aware of here is that the problem is not due to the sheer number of unique scopes that have been created within the site collections root Web, but that the effective number of unique security principals within that Web-level scope has grown to over 15,000 unique users. Each user added to any unique permissions scope below the Web was also added to the Web’s own scope, which caused a binary ACL recalculation for each addition.

Due to the large size of the Web-level scope combined with the frequency of binary ACL recalculation can cause blocking in several SQL Server stored procedures. Each time an item with broken inheritance has its membership scope changed, it causes each member of the scope to be added as a having Limited Access user membership at the Web-level scope. Additionally, each each time the membership of the Web scope was updated with existing or new members, including for Limited Access, it caused a recalculation of the Web-level scope binary ACL. Due to the Web-level scope containing over 15,000 security principals, it took a long time to recalculate. While it was recalculating, no access was available to that object, and end users experienced intermittent login difficulties.

## Resolution of FGP issues

The previously mentioned soltions were considered as part of the process of mitigating the experienced FGP related performance issues, with both a short term and long term plan enacted. The short term decision was to refactor the workflow to no longer set per item FGP, and the environment structure was left hierarchically the same. The individual FGP scopes were then removed, initially by attempting to remove each user from the Web scope or item level scopes, but as the performance was unsatisfactory, a removal process for each item scope was enacted by having the item inherit permissions from its parent. Additionally, some content rebalancing was used to prevent too many items from displaying at a specific level of hierarchy.

The event handler was modified to enforce a form or read access for those not currently assigned as reviewer by preventing modifications to documents or workflows. This approach did not limit who could view items, because there is no way other than the use of scopes to securely restrict viewing, but it could be used to prevent modifications to documents or workflows, such as for example, mistakenly allowing the author to modify the document while it was in a review cycle. Once individual item security scopes had been removed, and the updated workflow and event handler were installed users were able to use the environment, minus the individual item level security enforcement with no further performance issues.

The following diagram shows a simplified representation of the physical structure of the Web after security scope removal, where each uniquely numbered gold hexagon represents a unique permissions scope, and all child objects within that container inherit from that same scope unless they have their own unique permissions scope.



A planned longer-term solution in SharePoint Products and Technologies would be to separate content into different Webs, so that FGP could continue to be used, but with overall impact limited to a much smaller set of changes.

The following diagram shows a simplified representation of the physical structure of the content after separating to different Webs, where each uniquely numbered gold hexagon represents a unique permissions scope, and all child objects within that container inherit from that same scope unless they have their own unique permissions scope.



The following diagram shows the logical scope design and highlights the limits of how many unique security principals could be added to each Web's scope if FGP was reena bled after moving to different Webs. Note that although a large number of uniquely permissioned items still would remain, the key issue of excessive numbers of security principals in a scope is solved.



Lastly some consideration was made for when an eventual switch to SharePoint 2010 Products would bring new capabilities to the workflow design, specifically the ability to dynamically assign FGP by using the **SPRoleAssignmentCollection.AddToCurrentScopeOnly** method to assign membership only to each items individual scope, and then granting a SharePoint group, containing the membership, Limited Access at the parent Web. This process would enable FGP to be implemented via the workflow and/or event handler without impacting performance.

# Summary

This paper describes best practices on how your organization can use fine-grained permissions and what potential performance issues can occur. It additionally covers strategies and processes to mitigate issues if an environment is currently experiencing issues due to improper use or scale of fine-grained permissions. Lastly, it covers an example environment that was experiencing issues from improper fine-grained permissions use, and the process used to fix the issues found.