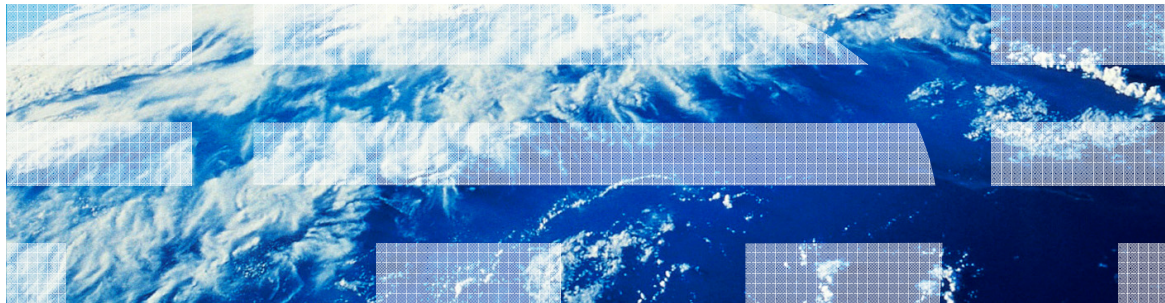


Securing Content

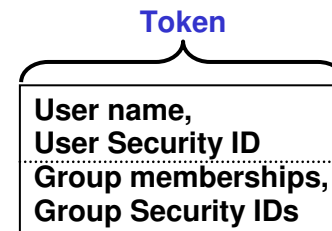


Security in the IBM FileNet P8 domain

- Security
 - Plays a central role in the overall behavior of any IBM FileNet ECM solution.
- Goals of security
 - Control access to information assets in the system.
 - Provide a framework for managing control of the assets.
- IBM FileNet P8 security model
 - Wide range of options for flexible security solutions
- Domain architecture defines a security context
 - Limits access to domain resources.
 - Grants access only to users with sufficient permissions.
 - Specifies permissible actions on objects.

Authentication

- Authentication: Who you are
 - Identifies the user attempting to log on.
 - Requires credentials (a user name and password).
 - Uses an authentication provider (LDAP directory service).
 - Creates a security token (a data structure that typically persists until the user logs out).
 - Uses JAAS and WS-Security standards.
- Example
 - A user attempts to sign in to Content Navigator or WorkplaceXT by entering a user name and password. The system responds with the following message: **Signin Error: Credentials Exception.**



Authorization

- Authorization: What you can do
 - Determines what the user can do (view, delete, modify, and so on).
 - Requires prior authentication.
 - Content Engine authorization is object based.
- Examples
 - A user opens Content Navigator and attempts to browse to an object store called *Finance*. The object store does not appear in the browse selection list.
 - A user is browsing a folder in Content Navigator and attempts to view a document that is in the folder. Content Navigator displays an error message that says that the user does not have access rights to view this document.
 - A user is browsing a folder in Content Navigator and does not see a document that a co-worker had just shown her. The user had seen her co-worker open the document from the co-worker's computer.

Security principals

- Generic name for users and groups
- Identified by a security identifier (SID)
 - Stored internally in the Content Engine.
 - SID does not change after assignment.
- Special logical security principals maintained by the CE:
 - #AUTHENTICATED-USERS (all domain users)
 - #CREATOR-OWNER
- Naming conventions

Name types	Example
Short name	JDoe
Distinguished name	cn=JDoe, cn=users, dc=IBM, dc=com
Principal name	JDoe@IBM.com

Users and groups

- The directory service defines the security principals.
- Users are assigned to groups.
- Use groups as primary security principals whenever possible.
- Example for a Finance department

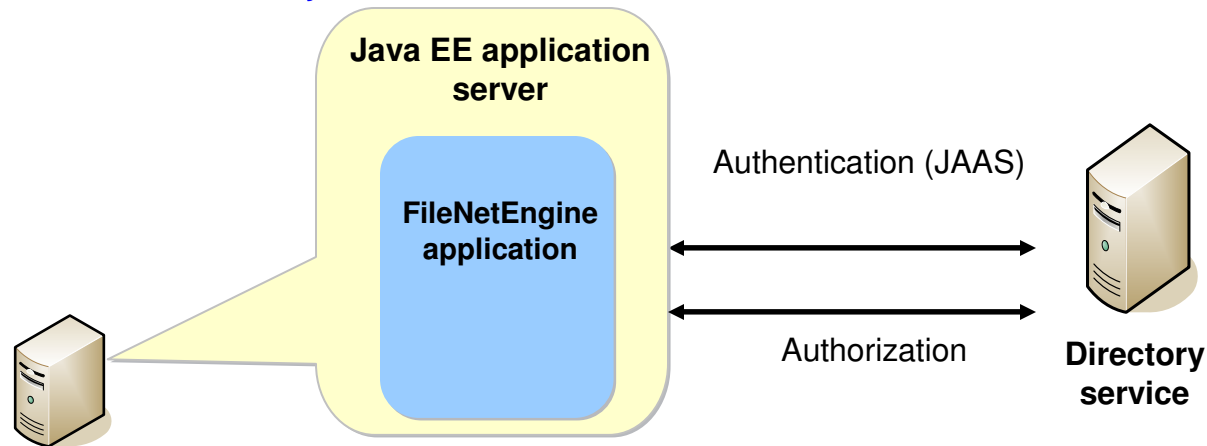
Group name	Users who are members
Finance Admins	adam, allison, steve
Finance Managers	may, mark
Finance Clerks	carol, charles
Finance Reviewers	richard, roberta

Security realms

- Realm
 - A collection of all user accounts and group memberships available to the FileNet P8 domain
 - Created, maintained, and authenticated by the authentication provider
 - Read and used by the FileNet P8 domain
- Multiple realms
 - The IBM FileNet P8 platform supports multiple realms.
 - Security principals can be mapped between realms using the Security Map Wizard.
 - Realms must use the same security server type.

Content Engine web application security

- The application server only allows authenticated users through to the Content Engine.
 - A standard JAAS-based mechanism is used.
- The CE application determines what it will authorize the user to do to the objects it controls.
 - Access is based on the rights of the security principal attempting to access the object.



Security cache

- Content Platform Engine caches directory service information
- Cached security information
 - Can become stale as a result of changes made on the directory server.
- User and group cache Time-To-Live (TTL)
 - The interval between FileNet P8 domain security updates
 - Default: 1 hour
 - Settable through API

Access rights

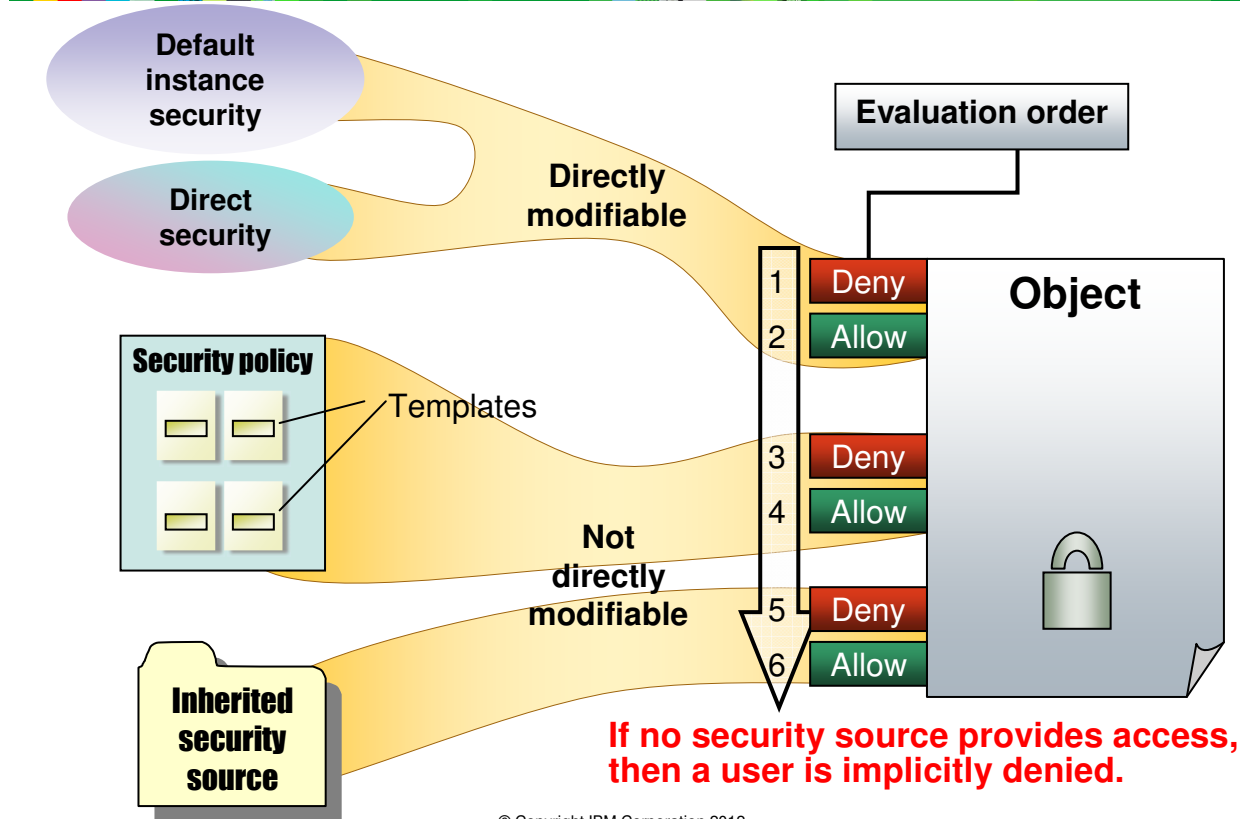
- Content Engine objects are secured by a set of access rights that control all operations on that type of object.
- Partial list of securable Content Engine objects:
 - Object stores
 - Document classes
 - Folders
 - Property templates
 - Documents
 - Event actions and subscriptions
- Access control entry (ACE)
 - A set of access rights for a given object that is associated with a single security principal (grantee)
 - One ACE per security principal
 - Each ACE allows or denies the specified access
- Access control list (ACL)
 - The set of ACEs associated with an independently securable object

Security sources

- Permissions for an object can come from various sources.
 - Sources can be used in combination.
 - Precedence is determined by order of evaluation.
 - Security is determined dynamically when object is accessed.
 - This fact is significant when using inherited security.

Security source	How source is applied to objects
Default	Permissions are initially copied from the Default Instance Security ACL of its class to an object ACL.
Direct	Permissions are applied directly on individual objects. If a Default ACE is edited, its source becomes Direct.
Template	Permissions are applied by a security policy, and are not directly editable.
Inherited	Permissions are applied by a security parent, such as a folder or another object, and are not directly editable.
Security markings	Restrictions are applied by marking sets.

Security sources and order of evaluation



What is direct security?

- Default security is set by the class definition.
 - Default instance security settings are applied to new instances.
- After Default security on an object is modified, it becomes Direct security.
 - In some cases, Default security becomes Direct after object is added.
 - Example: When a document is added using Add Document wizard in Content Navigator
- Object security can be modified:
 - For exceptional cases
 - For objects that do not follow predictable rules
 - By users with sufficient access
- Typically, security is automatically assigned to an object from other sources.
 - Tip: Design security solutions to minimize the need to modify direct security.

Configure security on content

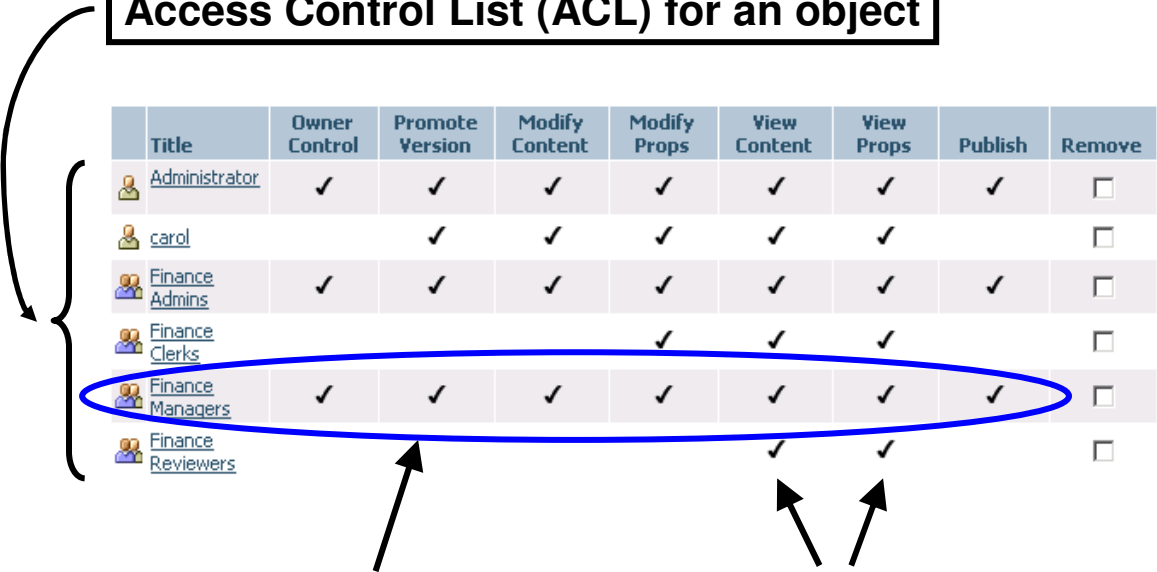
Primary interfaces for modifying security

- Content Navigator and Workplace XT
 - The set of property pages for each object includes a security page.
- Enterprise Manager and ACCE
 - Property sheet for each object has a Security tab
 - Allows finer control of object security than Content Navigator or Workplace XT
- Use these tools to do the following:
 - View and modify ACLs
 - View and modify ACEs
- Security can also be read and set programmatically.

Configure security on content

Access control from Workplace XT

Access Control List (ACL) for an object



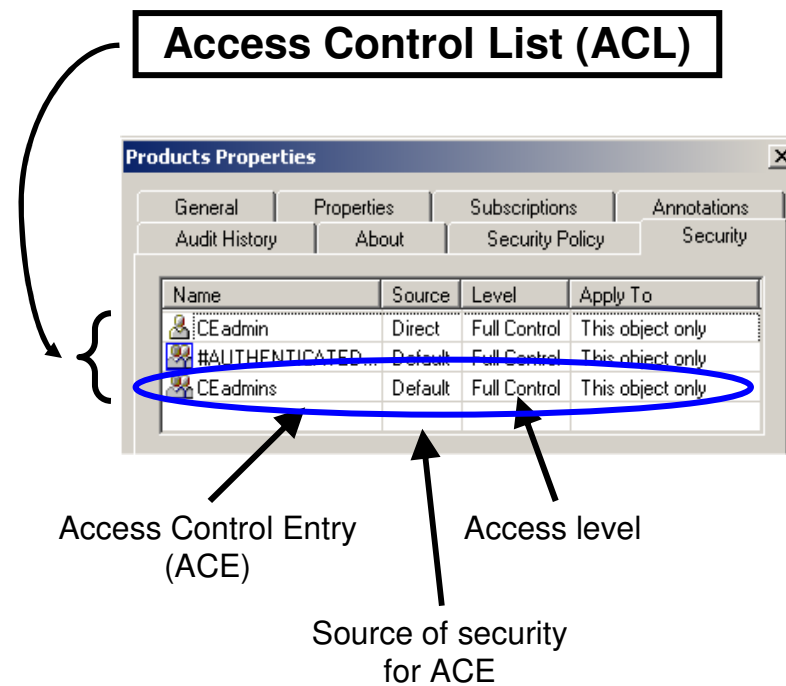
	Title	Owner Control	Promote Version	Modify Content	Modify Props	View Content	View Props	Publish	Remove
	Administrator	✓	✓	✓	✓	✓	✓	✓	<input type="checkbox"/>
	carol		✓	✓	✓	✓	✓		<input type="checkbox"/>
	Finance Admins	✓	✓	✓	✓	✓	✓	✓	<input type="checkbox"/>
	Finance Clerks				✓	✓	✓		<input type="checkbox"/>
	Finance Managers	✓	✓	✓	✓	✓	✓	✓	<input type="checkbox"/>
	Finance Reviewers					✓	✓		<input type="checkbox"/>

Access Control Entry
(ACE)

Access levels

Configure security on content

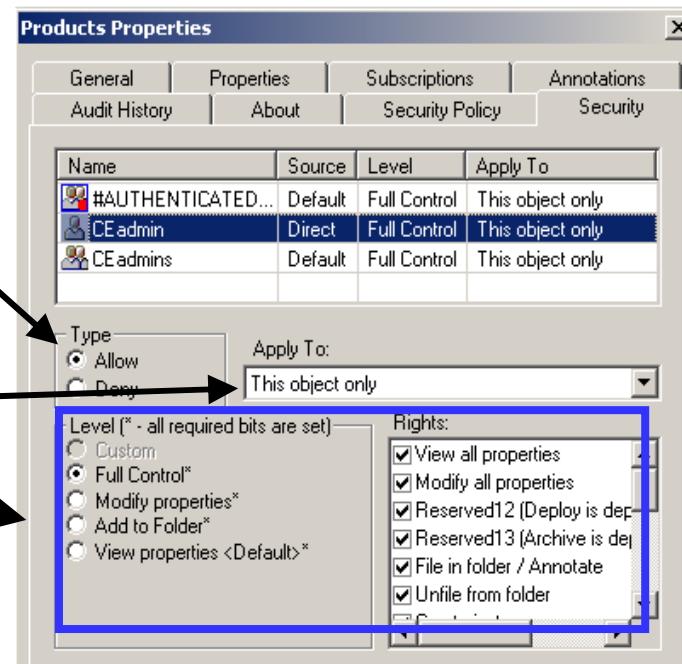
Access control from Enterprise Manager



Configure security on content

ACE features

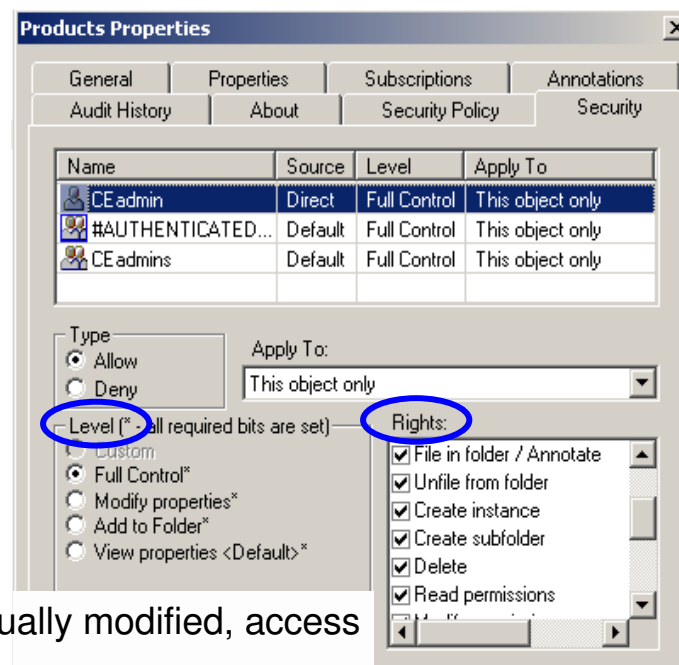
- Access type
 - Allow or Deny
 - Deny access is evaluated before Allow access for each ACE
- Inheritable depth
 - Applies only when objects use security inheritance
- Permissions



Configure security on content

Access rights versus access levels

- Access rights
 - Granular permissions tied to specific operations
 - Examples: Delete, File in folder
- Access levels
 - Common, preset groupings of access rights
- Workplace XT
 - View and edit access levels only
- Enterprise Manager
 - View and edit access levels and access rights
 - If access rights are individually modified, access level changes to Custom.



Configure security on content

Example access levels and rights

- Access level correlation with access rights (for documents)

Access level	Access rights
View content	View all properties View content Read permissions
Modify properties	View all properties Modify all properties View content Link a document or Annotate Create instance Change state Read permissions Unlink document

Configure security on content data structures

Required accounts

- An IBM FileNet P8 system requires some users and groups in order to install, configure, and administer it.
- These accounts are usually created during installation.
- FileNet P8 documentation has a complete list.

Account examples

Users required for installation

- Content Platform Engine system user
- Application server administrator
- Application Engine administrator
- Others

Users required for administration

- GCD administrators
- Object store administrators
- Content Platform Engine operating system user
- Others

Configure security on content data structures

Overview of initial security configuration

- Important principle:
 - Plan security before you create a new object store.
 - To change security after objects exist can be complex and can cause unexpected consequences if not done correctly.
- Configure the following during object store creation:
 - Object store administrators
 - Object store users
- Configure the following before use in production:
 - Root folder security
 - Default instance security
 - Property modification access
 - Who has ownership of new objects

Configure security on content data structures

Configure object store administrators and users

- Specify object store administrators
 - These users are able to retrieve all objects in the object store, even if explicitly denied access.
 - Does not mean they can see the content or change properties.
- Examples of initial object store administrator groups:
 - FileNet P8 domain administrators
 - Object store administrators
- Specify initial user groups to prevent an object store from being used by all domain users.
- Examples of initial object store users:
 - #AUTHENTICATED-USERS (all domain users)
 - Finance users (only members of the Finance department)
- If a security principal is removed from an object store ACL, that person or group is denied access to the entire object store.

Configure security on content data structures

Example security scenario

- GCD administrator
 - Creates object store, setting initial security groups.
 - Creates file storage areas if needed.
- Object store administrator
 - Creates classes and other supporting entities as needed.
 - Defines default instance security for the classes.
 - Configures permissions for Root Folder.
 - Creates root-level subfolders to be used by department managers.
- Business users
 - Managers create subfolder structures.
 - Clerks add documents.
 - Reviewers view documents.

Configure security on content data structures

Default instance security

- An ACL that is configurable at the class level
 - Used as the source for default security when objects are instantiated.
- Most common technique for applying security
 - Determines the initial proposed security of an object.
 - Works automatically.
 - Used to enforce consistency in assigning initial security.
- Changes to default instance security have no effect on existing objects.
- Direct object security can be modified during or after instantiation.
- By default, the creator of an object has Owner access.
 - Remove #CREATOR-OWNER from the default instance security to override this behavior.

Configure security on content data structures

Root folder security

- The security on the Root Folder of an object store determines who can add folders to the top level.
 - Access to the Root Folder is usually restricted.
- Root Folder security is accessible from Content Navigator or WorkplaceXT.
 - Users must have proper permission to view or edit the security.
- Documents in Root Folder are not listed in Content Navigator or Workplace XT when browsing.

Configure security on content data structures

Property modification security

- Custom properties can be independently secured.
 - Optional extra layer of security
 - Properties can be set to have Modification Access Required (MAR).
 - Set on a property template (affects all classes that use it) or on a particular class only.
- Example:
 - Clerks can add invoices but must be prevented from changing the value of the *Approved* property later.
- How to configure
 - All property templates have a Modification Access tab.
 - Select the access rights a user must have in order to modify the property value.
 - Example: If you select Delete, only users who can delete the object can modify that property value.

Configure security on content data structures

Changing ownership

- All objects have an owner.
 - Ownership confers some special privileges on the object.
- The user who creates the object is the owner by default.
 - Change this default behavior by changing the default instance security for the class, or by setting the default owner property.
- To take ownership of an object
 - User must have *Modify owner* access right.
- To change ownership of an object store
 - User must have *Set Owner of any object* access right.
- To change user and group access on object store after creation
 - Use the Security Script Wizard.
- To change security on multiple objects after creation
 - Use the Query Builder and bulk operations.

Configure security policies

Review: document version states

- Documents go through version states when they are checked out and checked in again.
- Each of these states might require different security.

Versioning actions		Versions created		
1	Add as minor version	Version 0.1 Status: In Process Current version		
2	Check out	Version 0.1 Status: In Process Current version	Version <u>0.2</u> Status: <u>Reservation</u>	
3	Check in as minor	Version 0.1 Status: <u>Superseded</u>	Version 0.2 Status: <u>In Process</u> <u>Current version</u>	
4	Check out	Version 0.1 Status: Superseded	Version 0.2 Status: In Process Current version	Version <u>0.3</u> Status: <u>Reservation</u>
5	Check in as major	Version 0.1 Status: Superseded	Version 0.2 Status: <u>Superseded</u>	Version 1.0 Status: <u>Released</u> <u>Current version</u>

Configure security policies

Two kinds of security templates

- Version security templates
 - A mechanism for controlling security on documents
 - Controls multiple documents without directly editing each ACL.
 - Designed for changing security during versioning
 - Used on document type objects only, because only documents are versionable.
 - Modifies ACL when the document version state changes.
- Application security templates
 - Applied by using the API
 - Control changes unrelated to versioning
 - Examples: A property value changes, an external condition is met, a lifecycle state changes
- This course uses only version security templates.

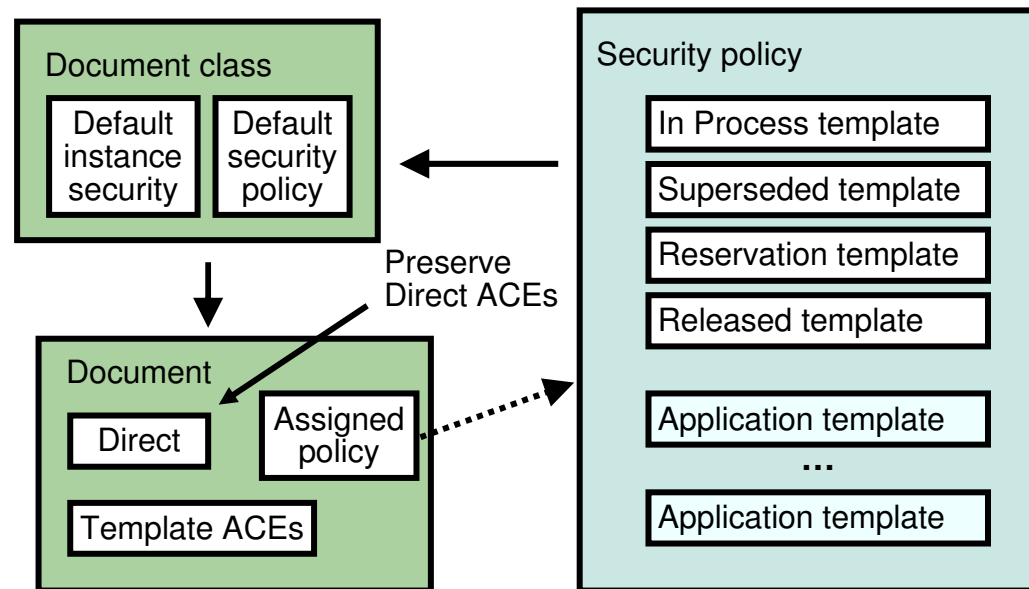
Security policies



- A security policy is a collection of one or more security templates.
 - The appropriate security template is applied to an object as the associated condition in the security policy is met.
 - Example: Apply a template as a document gets promoted from a minor version to a major version.
 - Permissions are added to the ACL of the object by the template.
 - An ACE is added for each grantee specified in the security template.
 - The ACE displays Template as the security source.
 - Optionally, existing direct ACEs can be replaced by new permissions.
 - In that case, all existing direct ACEs are deleted.

Security policy architecture

- You can assign security policies to a document.
 - Generally you configure the default security policy for the document class.



Configure security policies

When to use security policies

- When different version states require different security
 - In Process
 - Superseded
 - Reservation
 - Released
- For any other change that requires security adjustment
 - Requires using the API (custom application or script).
 - Application determines when to apply the change.
 - Avoids having to program direct security on an object.

Security policy guidelines (1)

- Combine security policies with default instance security to allow additional permissions.
 - Security policies are not effective for denying permissions that are already allowed (because of evaluation order rules).
 - Template security is evaluated after default and direct security.
 - Policies are configured to preserve direct ACEs by default.
- A single policy can be used for multiple classes.
- Templates are unique to each policy (not shared).
- Security policies provide primary security settings if direct ACEs are not preserved.
 - In that case, you must modify security through the policy and not the object.

Security policy guidelines (2)

- Assign a default policy to document class before instantiating documents.
- Change a security policy for an individual document only under exceptional circumstances.
 - It is more difficult to manage individual document assignments.
- Be aware of subtle version template behavior.
 - If a security policy does not have a template for a particular version state, permissions are retained or copied from the previous state of the document.
 - If a security template for a version state exists but has no permissions, any existing permissions on that object that were previously applied by a security policy are removed.

Configure security policies

Security policy configuration

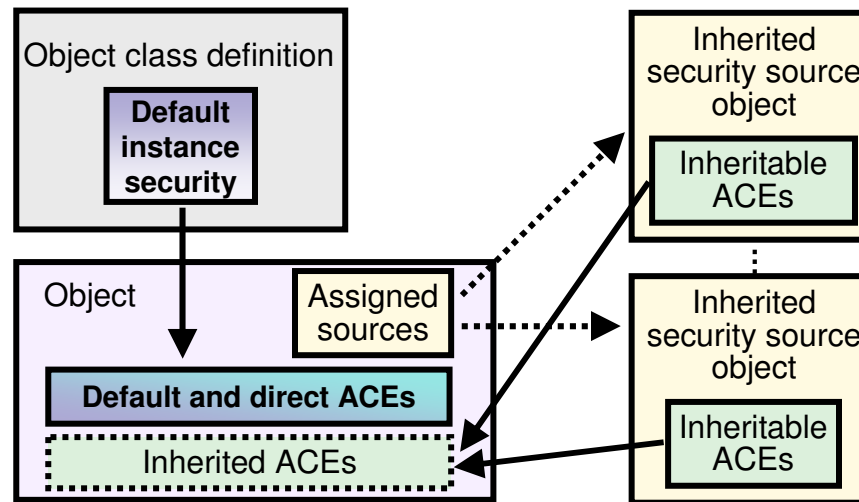
- Use Enterprise Manager, Workplace XT, or ACCE.
 - Security policy wizard interface varies among them.
- Add one template to the security policy for each version state you wish to affect.
 - Optionally, add application templates if needed.
- Apply the security policy to a document class.
 - The policy is assigned to an object when it is instantiated.

Definition of terms

- Security inheritance
 - The ability to pass permissions from a source object to a child object
- Inheritable depth
 - A property that determines whether access rights are not to be inherited, inherited only by objects that are immediate children, or by all children
- Inherited security source
 - A general term used throughout this lesson that refers to any object from which another object inherits security
- Security Folder
 - A folder that is used to provide the security for child documents to inherit.
- Security proxy
 - An object that is used to provide the security for other objects to inherit.

Inherited security source architecture

- Inherited security sources can be assigned to an object.
 - Inheritable ACEs are applied to the child object.
- Inherited security is computed only when needed for access.
 - When permissions on a security source change, the ACL of the child object is not changed.



Inherited security sources

- Additional access can be granted to objects that already have direct, default, or template security.
 - Inherited permissions are added to the permissions set by the other types.
 - Inherited permissions are a lower precedence than other sources (default, direct, or template) and so can be overridden by them. This is an issue only when using deny permissions, since using only allow permissions causes them all to be combined, no matter what their source.

Configure security inheritance

Characteristics of inherited permissions

- Inherited permissions are not directly modifiable.
 - You must modify the permissions on the security source object.
 - Inherited permissions are displayed as disabled in security interfaces.
- Deleting inherited security sources
 - If you delete an inherited security source, the inherited permissions are removed from the child objects.

Configure security inheritance

Changing inheritable permissions

- Changing inheritable permissions
 - Changes to the inheritable permissions on a security source apply to all versions of a document that inherits those permissions.
 - This behavior can be modified in custom applications using the API.
- Default instance security ACLs
 - For folder, document, and custom object classes, several access rights are listed as *Inherit Only*.
 - These rights do not control access to the parent object, but are passed on to the children.
 - Example: You can set the *Major versioning* access right on a custom object so that it can be inherited by documents.

Configure security inheritance

Configuring security inheritance in an object store

- Two methods are available for setting up security inheritance in an object store:
 - The *Security Folder* method uses folders to set security on objects.
 - Inheriting objects have exactly one folder as the inherited security source.
 - Set the Security Folder property on the inheriting object.
 - The *Security proxy* method can use any class of object as an inherited security source.
 - Inheriting objects can have multiple security sources of this type.
 - Set the Security Proxy Type property on the inheriting object.
- An inheriting object can inherit whatever its security source inherited.

Configure security inheritance

Method 1: Use a folder as a Security Folder

- Use any folder in the same object store as the security source.
- Objects can have only one Security Folder at a time.
- The desired permissions on the folder must be configured as inheritable.
 - Inheritable depth for the ACEs must be set to include immediate children or all children.
- Security source assignment is done on each inheriting object.
 - Requires copying the object reference of the folder that you designate as the inherited security source.
- Deleting the folder removes the Security Folder relationship from the object.
- Moving the child object has no effect on the Security Folder relationship.

Configure security inheritance

Method 2: Use an object as a security proxy

- Objects can have multiple inherited security sources.
 - Each contributes equally to object security.
- Use any object in the same object store for the security source.
- The desired permissions on the security proxy object must be configured as inheritable.
 - Inheritable depth for the ACEs must be set to include immediate children or all children.
- Security source assignment is done on each inheriting object or on the class definition.
 - Requires using an object reference (GUID) of the object that you designate as the inherited security source
 - A custom application can be used to set the property.
- When the inherited security source object is deleted, the inherited security is removed from the object.

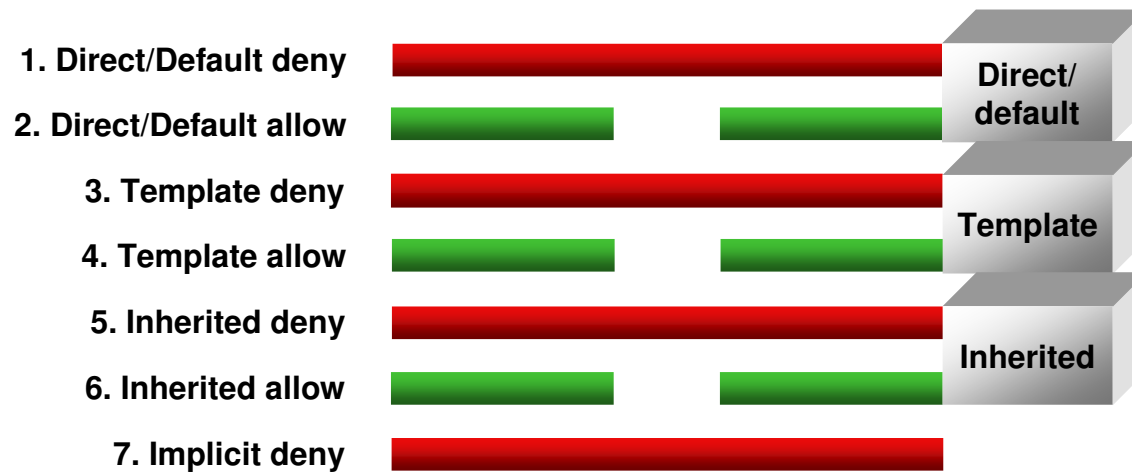
Security concepts – review



- Multiple security sources can provide security for an IBM FileNet P8 object.
- IBM FileNet P8 evaluates security settings based on the Order of Evaluation to determine precedence.
- Deny is evaluated first and takes precedence over Allow at the same security source level.
- If no Access Control Entry (ACE) exists at the Direct/Default Deny security source level, the next level is evaluated.
- Evaluation continues until an ACE is identified.
- If no ACE exists at any security source level, the security principal is implicitly denied.
- Security templates can be configured to combine template ACEs with Direct ACEs or replace them.

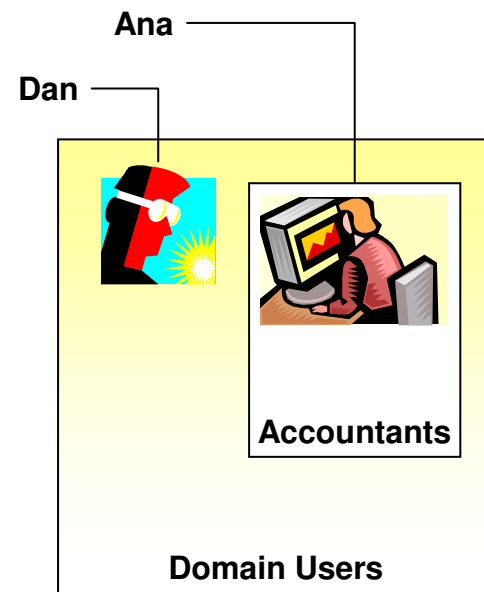
Order of evaluation

- IBM FileNet P8 checks the ACL in the following order.

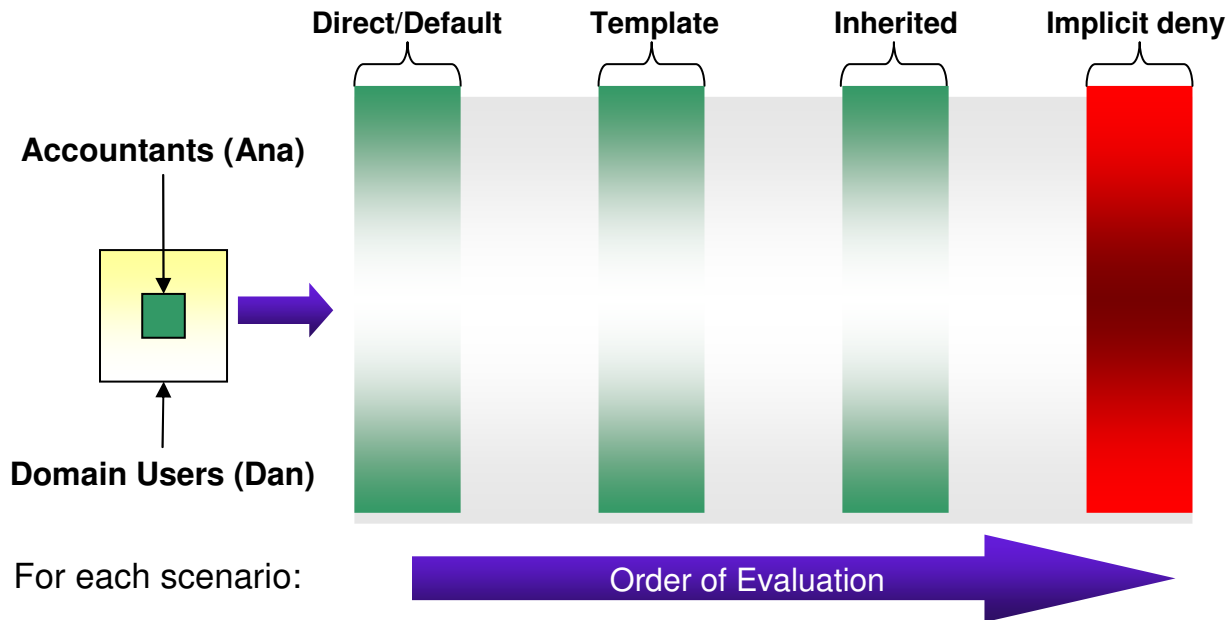


Analyzing security settings (1)

- The following scenarios test your knowledge of
 - Security principal relationships
 - Security sources
 - Evaluation order
- All scenarios assume that Ana and Dan receive rights through their group security principal membership instead of rights assigned to an individual user.
- Dan is a Domain User. Ana is a member of the Accountants security principal, which is a member of the Domain Users principal.
- All Accountants are Domain Users, but not all Domain Users are Accountants.
- All scenarios assume built-in functionality, with no custom code.



Analyzing security settings (2)



For each scenario:

1. Read the description.
2. Use the security source and evaluation order concepts to determine which security principals have rights to the object.

Security evaluation order review

Scenario 1:



Direct/Default

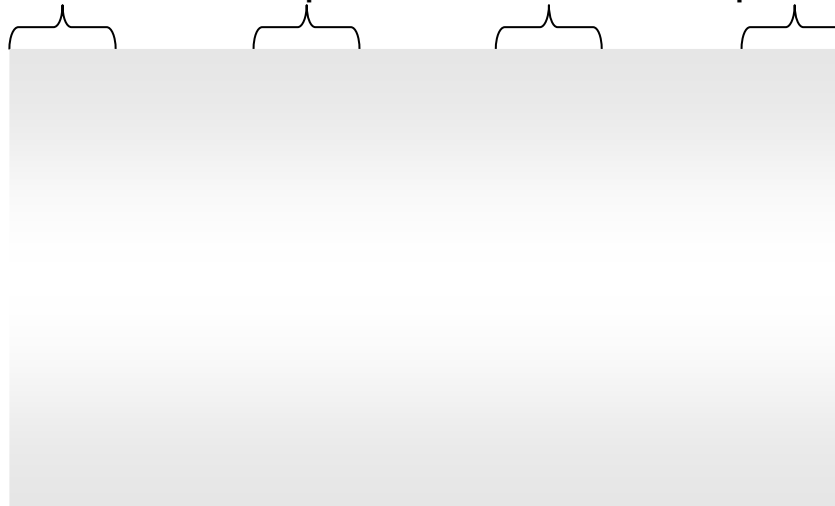
Template

Inherited

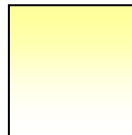
Implicit deny

Accountants and
Domain Users are
not mentioned in an
ACL for the object.

What are the results?

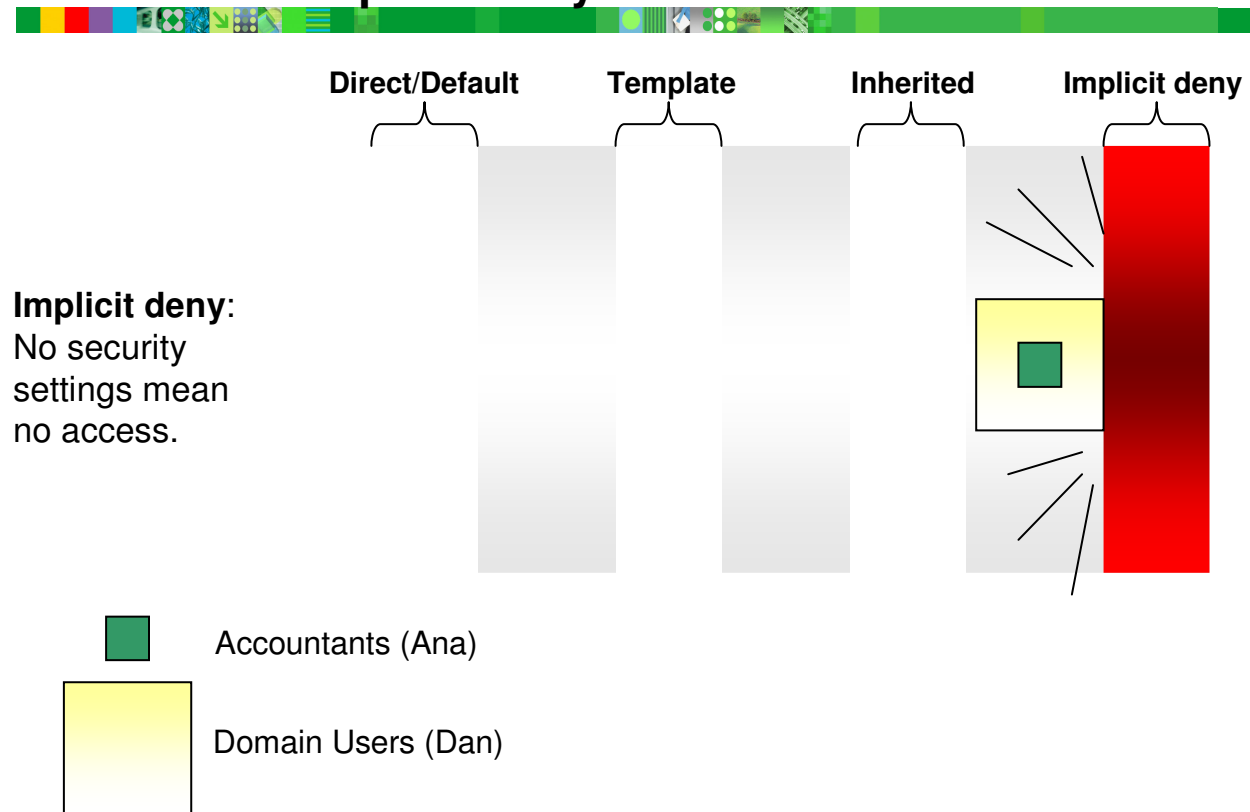


Accountants (Ana)



Domain Users (Dan)

Scenario 1: Implicit deny



Scenario 2



Direct/Default

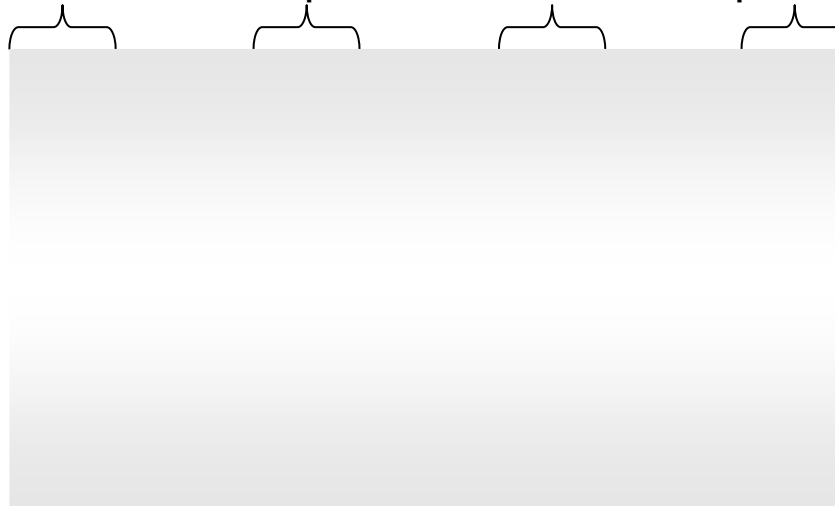
Template

Inherited

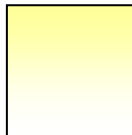
Implicit deny

An Accountants allow
View content ACE
exists at the Inherited
security source level.
Domain Users are not
members of an ACE
associated with the
document.

What are the results?

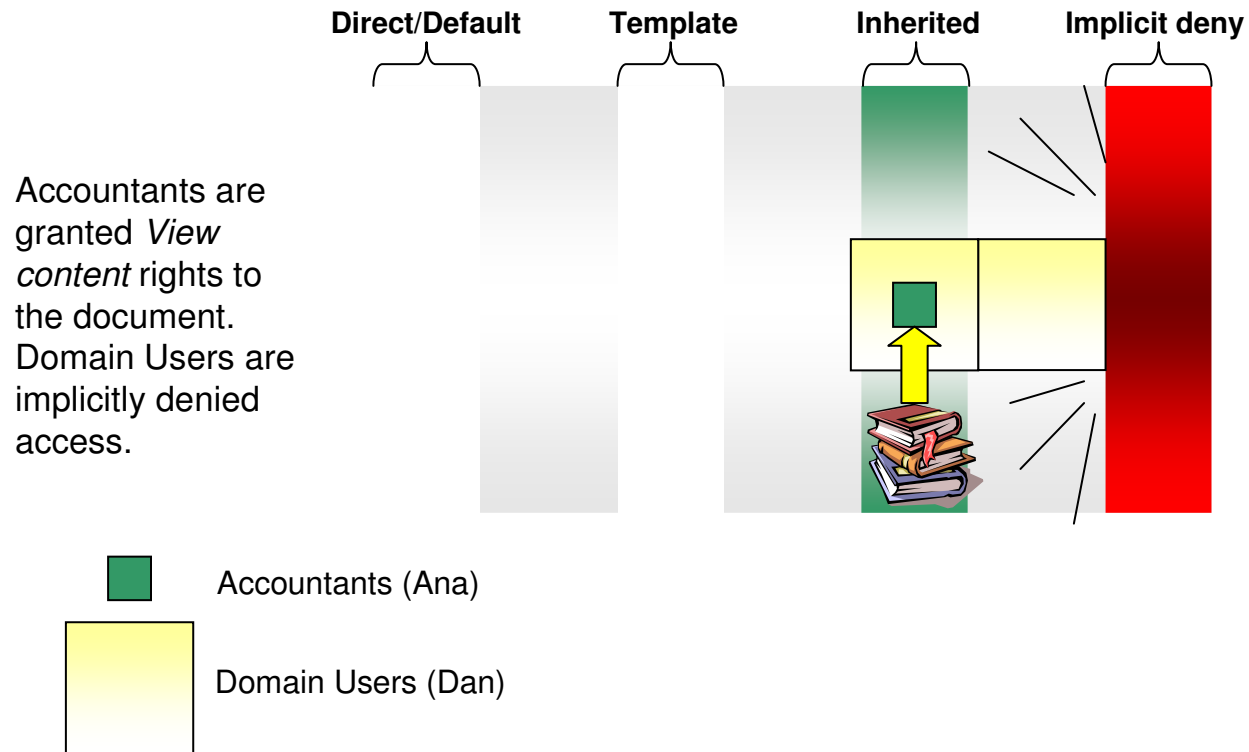


Accountants (Ana)



Domain Users (Dan)

Scenario 2: Groups and inherited allowance



Scenario 3



Direct/Default

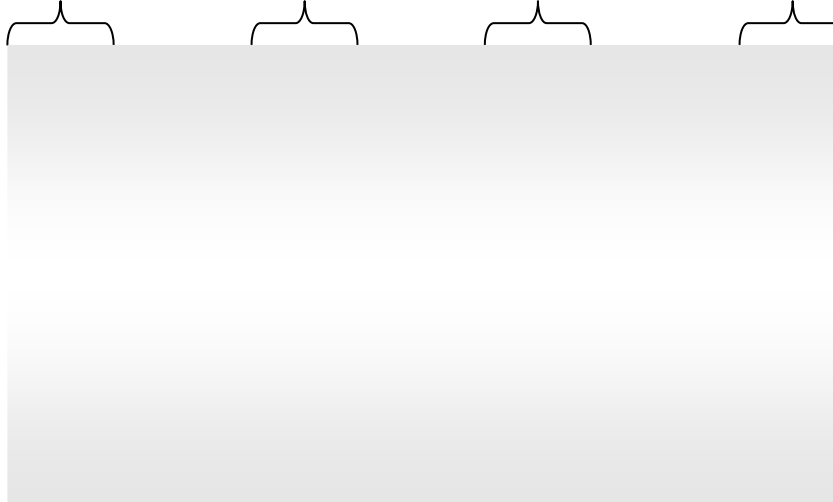
Template

Inherited

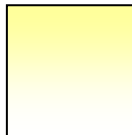
Implicit deny

An Accountants allow
Modify properties
ACE exists at the
Inherited level.
Domain Users are
denied *View content*
rights at the Inherited
level.

What are the results?

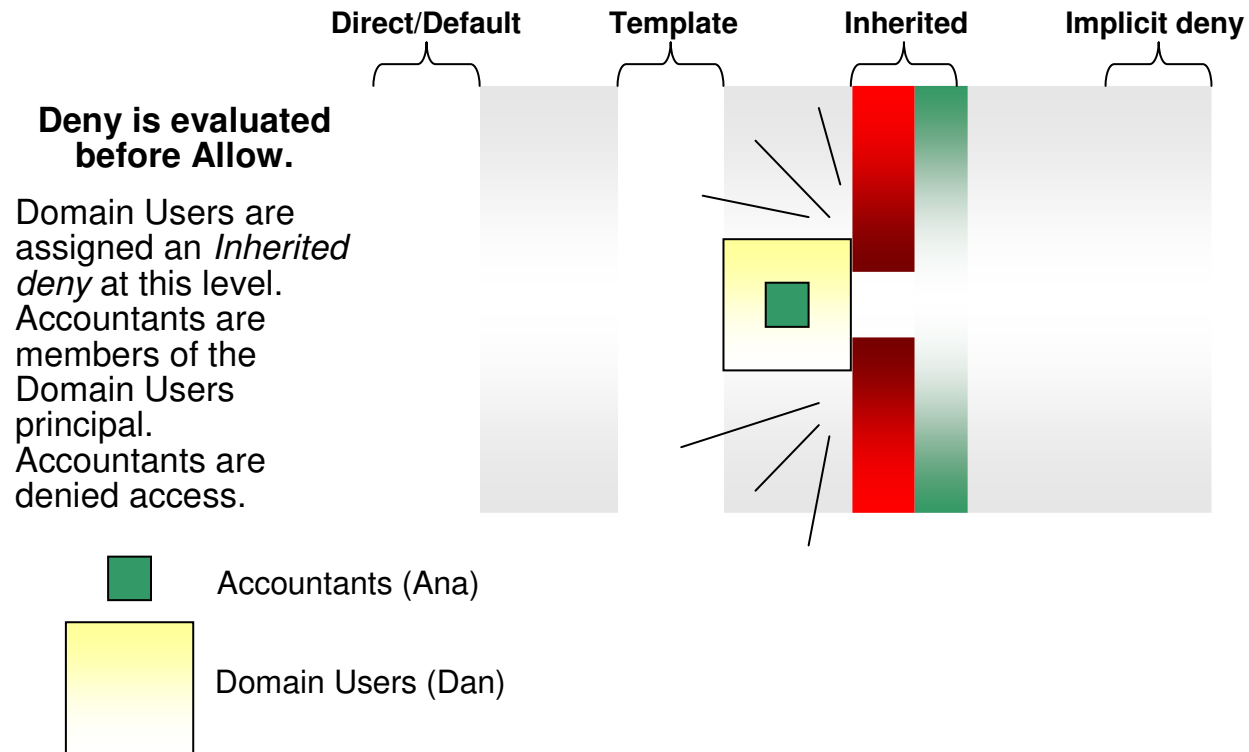


Accountants (Ana)



Domain Users (Dan)

Scenario 3: Groups and inherited denial (1)



Security evaluation order review

Scenario 4



Direct/Default

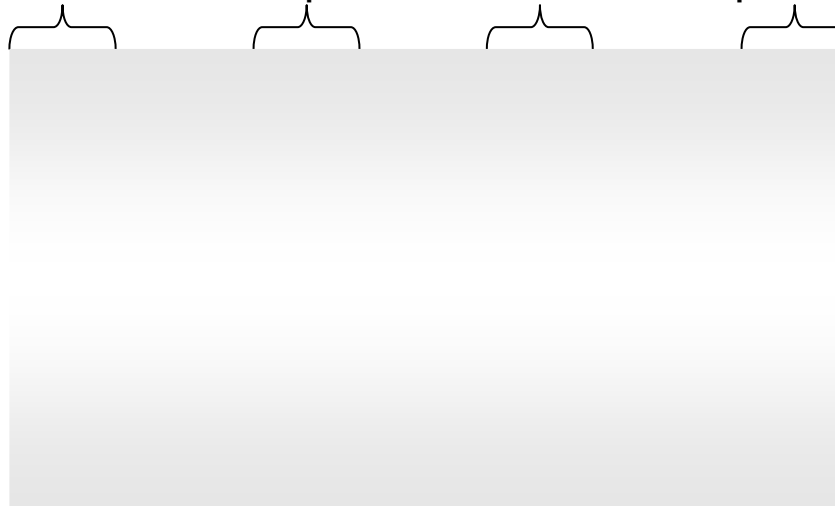
Template

Inherited

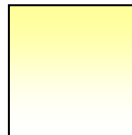
Implicit deny

Accountants are denied *View content* rights at the Inherited level. Domain Users inherit allow *View content* rights at this level. Accountants are members of Domain Users.

What are the results?

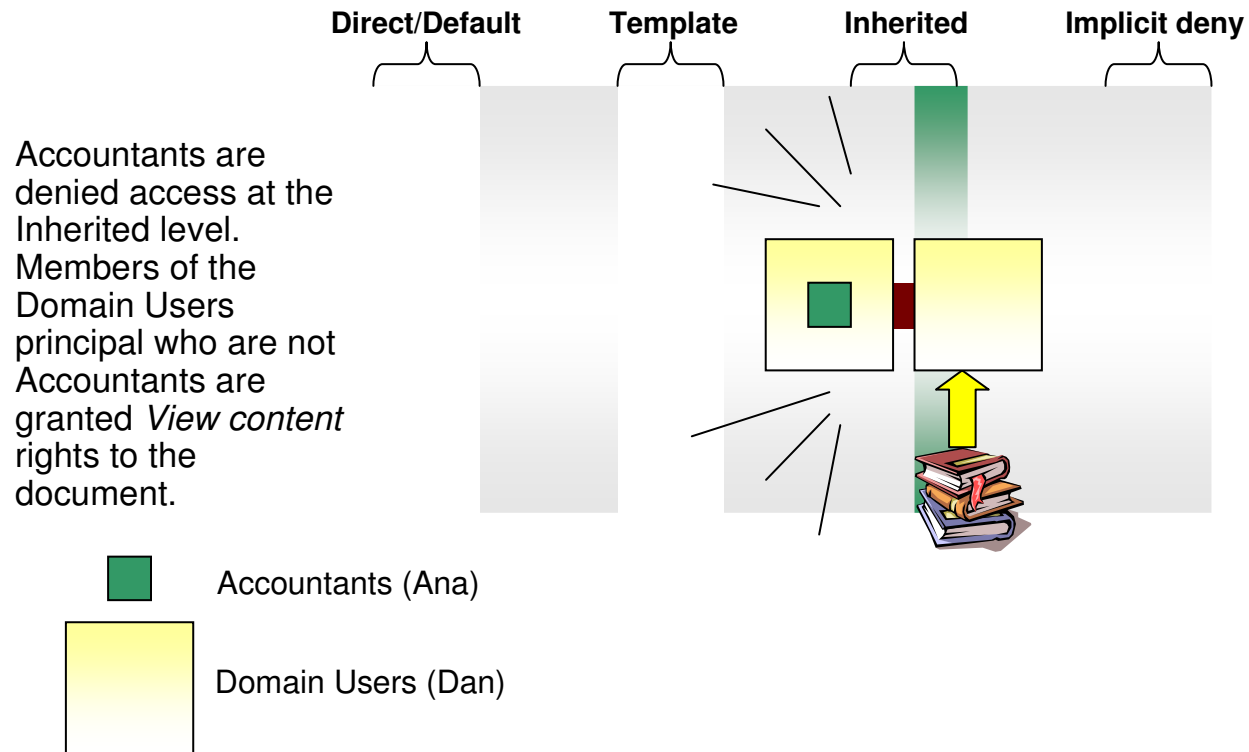


Accountants (Ana)



Domain Users (Dan)

Scenario 4: Groups and inherited denial (2)



Security evaluation order review

Scenario 5



Direct/Default

Template

Inherited

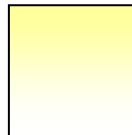
Implicit deny

Accountants are granted full control of a document at the Direct/Default level. Domain Users are denied *View content* at the Inherited level. Accountants are members of the Domain Users principal.

What are the results?

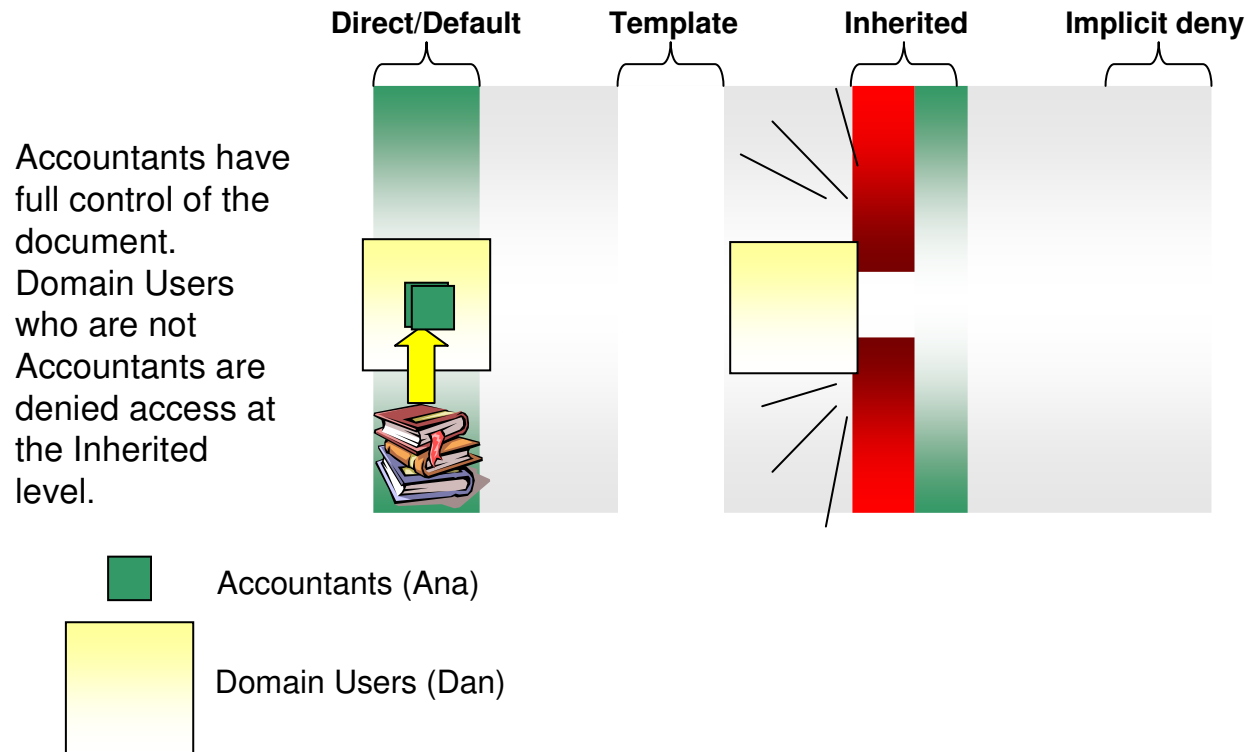


Accountants (Ana)



Domain Users (Dan)

Scenario 5: Groups and Default/Direct allowance (1)



Scenario 6



Direct/Default

Template

Inherited

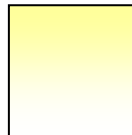
Implicit deny

Accountants are granted *View content* rights to a document at the Direct/Default level and *Modify properties* rights at the Inherited level. Domain users are not associated with an ACE for the document.

What are the results?

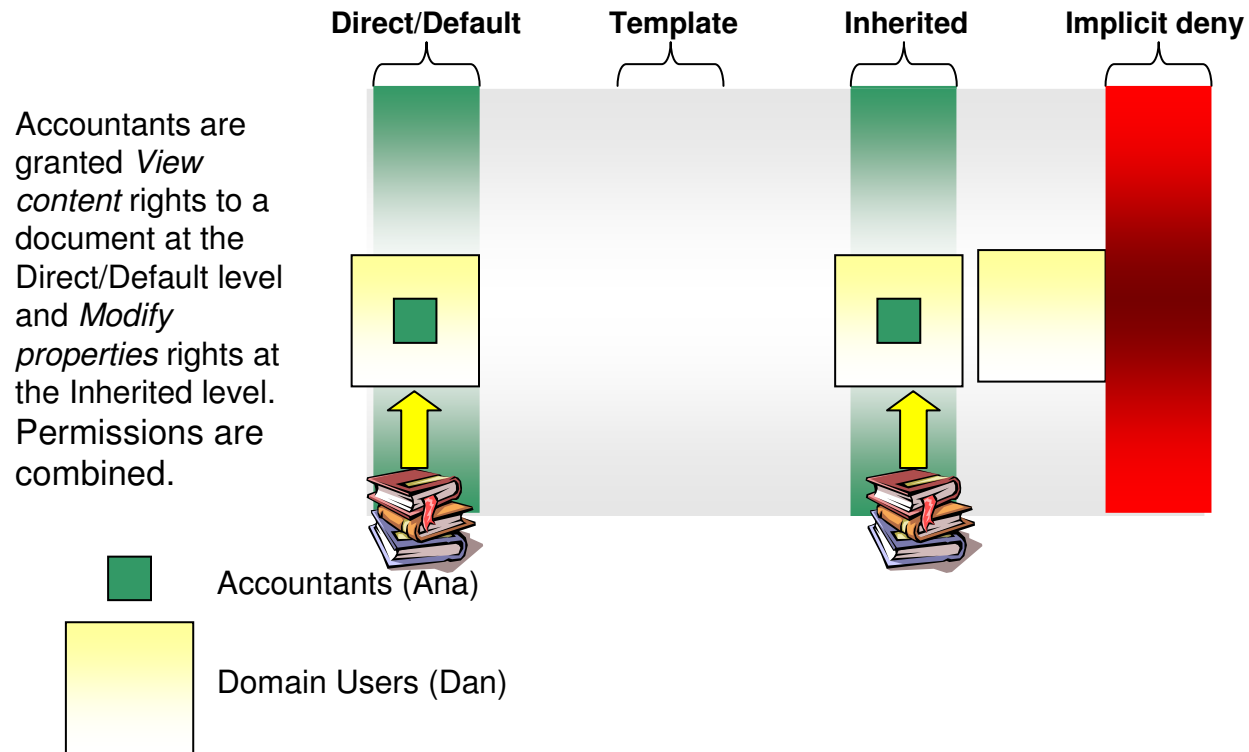


Accountants (Ana)



Domain Users (Dan)

Scenario 6: Groups and Default/Direct allowance (2)



Scenario 7



Direct/Default

Template

Inherited

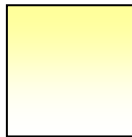
Implicit deny

Domain Users are denied *View content* rights at the Direct/Default level and allowed *View content* rights at the Inherited level.

What are the results?

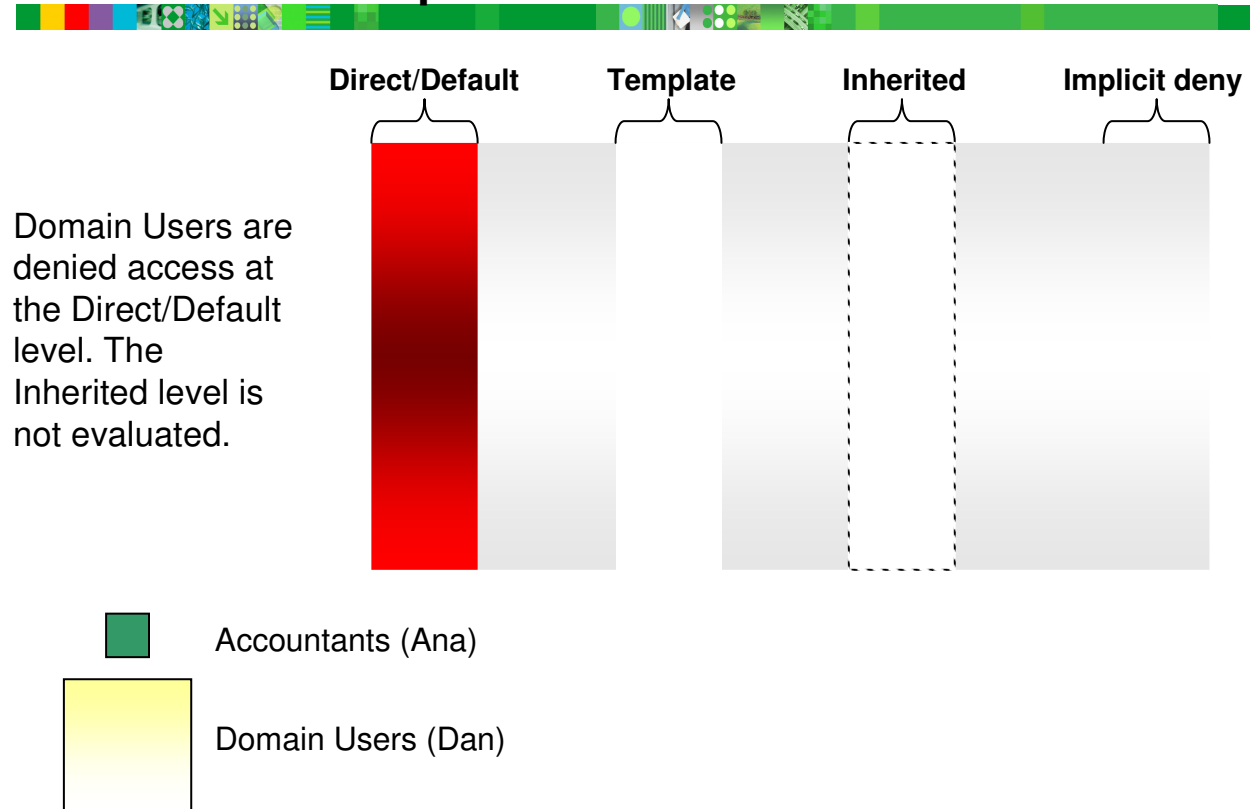


Accountants (Ana)



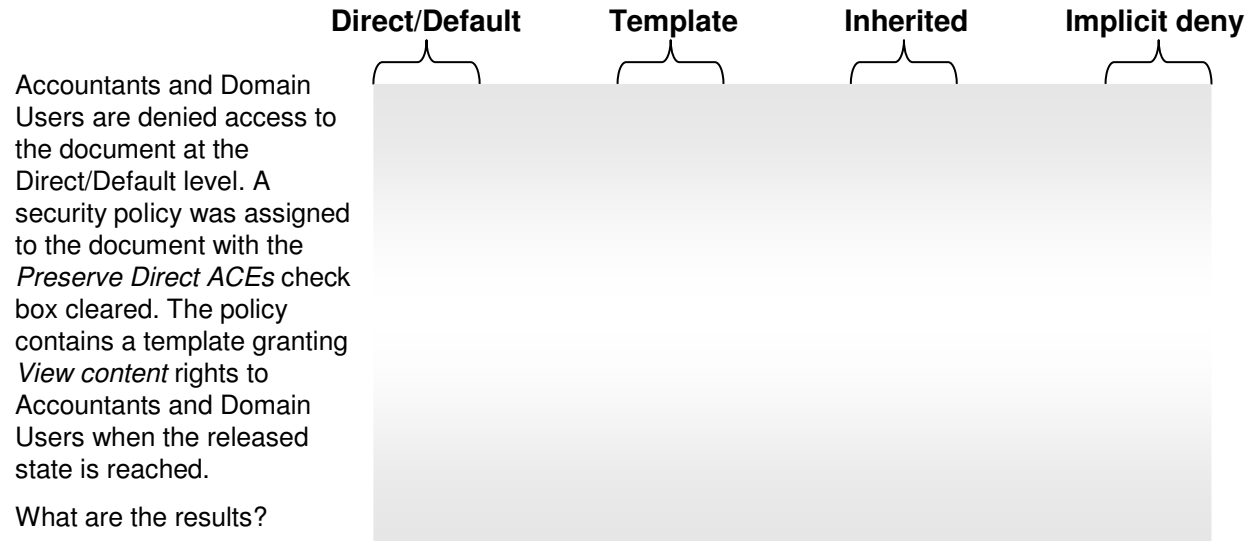
Domain Users (Dan)

Scenario 7: Groups and Default/Direct denial

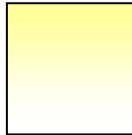


Security evaluation order review

Scenario 8



Accountants (Ana)



Domain Users (Dan)

Scenario 8: Groups and template security

