

Microsoft®

Working with Store Permissions in Microsoft Exchange 2000 and 2003



Product Version:
Reviewed by:
Latest Content:
Author:

Exchange Server 2003 and Exchange 2000 Server SP3
Exchange Product Development
www.microsoft.com/exchange/library
Teresa Appelgate

 Microsoft
Windows Server System

Working with Store Permissions in Microsoft Exchange 2000 and 2003

Teresa Appelgate

Updated: September 2003

Applies To: Exchange Server 2003, Exchange 2000 Server SP3

Copyright

Information in this document, including URL and other Internet Web site references, is subject to change without notice. Unless otherwise noted, the example companies, organizations, products, domain names, e-mail addresses, logos, people, places and events depicted herein are fictitious, and no association with any real company, organization, product, domain name, e-mail address, logo, person, place or event is intended or should be inferred. Complying with all applicable copyright laws is the responsibility of the user.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

© 2003 Microsoft Corporation. All rights reserved.

Microsoft, Active Directory, MSDN, Outlook, Windows, Windows NT and Windows Server are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

The names of actual companies and products mentioned herein may be the trademarks of their respective owners.

Acknowledgments

Project Editor: Lindsay Pyfer

Technical Reviewers: Catalin Stafie, Larry Osterman, Keith MacRae, Mike Lee

Artist: Kristie Smith

Production: Sean Pohilla

Table of Contents

Introduction	
Overview	1
What is Updated in This Book?	1
What Will You Learn from This Book?.....	1
How is This Book Structured?	2
Who Should Read This Book?	3
What Terminology is Used in This Book?.....	3
Background	4
Chapter 1	
Accessing Exchange Objects	7
Chapter 2	
Details of the Exchange Access Control Process.....	13
Performing Preliminary Checks	13
Determining the Type of User.....	14
Determining the Type of Client Application	14
Anatomy of Object-Level Access Control	15
Security Descriptors.....	16
ACLs and ACEs	16
ACE Sequences in the ACL	17
Default ACLs for Messages	20
Special Considerations for Coexisting Exchange 2003 and Exchange 5.5 Servers.	21
Chapter 3	
Modifying Permissions.....	23
Controlling Client Access to Mailboxes	23

Controlling Client Access to Public Folders	25
Special Considerations for Messages.....	31
Controlling Administrative Access to Mailboxes and Public Folders.....	32
Mailboxes	32
Public Folders.....	32
Special Controls for Mail-Enabled Public Folders	36
Chapter 4	
Permissions Available in Exchange.....	39
Special Permissions.....	39
Permissions Used in the Windows NT Security Descriptor.....	40
Permissions for Mailbox Folders and Public Folders	40
Permissions for Messages	42
Permissions Used in the Administrative Security Descriptor	47
Chapter 5	
Conversion Between MAPI and Windows Permissions	51
Available MAPI Permissions	51
Converting to MAPI Permissions	52
Converting from MAPI Permissions.....	54
Examples of the Conversion Process.....	55
Chapter 6	
Default Roles Available in Exchange.....	63
Appendix A	
Minimum Permissions Required for Mailbox Stores and Public Folder Stores	69
Appendix B	
Additional Resources.....	71

INTRODUCTION

Overview

This book explains the process by which both Microsoft® Exchange Server 2003 and Microsoft Exchange 2000 Server use permissions to control access to objects in the Exchange store (public folder databases and mailbox databases).

Note

To avoid repeatedly referring to both Exchange Server 2003 and Exchange 2000 Server, the text in this book specifically refers to Exchange 2003. Although the text does not specifically refer to both Exchange 2003 and Exchange 2000, all information applies equally to both versions.

What Is Updated in This Book?

Since the previous version of this book was released, clarifications have been made to the distinction between what Exchange considers a client action (one that provides regular client access) and an administrative one (one that provides administrative client access). This expanded information appears in the following sections:

- "Opening a Folder or Message in a User Mailbox" and "Opening a Public Folder or Public Folder Message" in Chapter 1, "Accessing Exchange Objects."
- "Determining the Type of Client Application" in Chapter 2, "Details of the Exchange Access Control Process."

What Will You Learn from This Book?

This book provides detailed answers to the following questions:

- When I open a folder or a message, what does Exchange check to determine whether I have access? Where does this information come from?
- How do Exchange store permissions differ from permissions in the Microsoft Active Directory® directory service and the Microsoft Windows® 2000 or Microsoft® Windows Server 2003 file system?

- How many sets of permissions does Exchange use, and what is the function of each?
 - Of these sets of permissions, which ones are related to one another and how does Exchange convert information from one form to another?
 - Why does Exchange provide several different user interfaces for viewing permissions, and why is each different?
 - Which interface should I use to set a particular type of permission? Which interfaces should I never use?
 - If I want to change permissions, which settings should I leave alone for Exchange to function smoothly?
-

How Is This Book Structured?

This book is divided into six chapters and two appendixes.

Chapter 1, "Accessing Exchange Objects"

This chapter describes the access checking processes for mailboxes and public folders, placing the checks in the larger context of Exchange functions.

Chapter 2, "Details of the Exchange Access Control Process"

This chapter describes the access checks that are detailed in Chapter 1, "Accessing Exchange Objects." This chapter describes the different sets of permissions that Exchange uses, the differences between Windows access control and Exchange access control, the default access control lists that Exchange constructs, the way Exchange uses inheritance, and special circumstances that arise when an Exchange topology includes servers running Microsoft Exchange Server version 5.5.

Chapter 3, "Modifying Permissions"

This chapter describes how to view and modify the different sets of Exchange permissions. You can use several different interfaces to view permissions, but some of them should never be used to modify permissions.

Chapter 4, "Permissions Available in Exchange"

This chapter describes the permissions that are available, in detail.

Chapter 5, "Conversion Between MAPI and Windows Permissions"

This chapter describes the process that Exchange uses to convert permissions from their Windows-compatible form to the MAPI form used by Exchange 5.5 and Microsoft Outlook®, and back again. This chapter also provides an example of the conversion process.

Chapter 6, "Default Roles Available in Exchange"

This chapter lists the permissions that are available for each of the default "roles" that are used by Exchange and Outlook.

Appendix A, "Minimum Permissions Required for Mailbox Stores and Public Folder Stores"

This section lists the minimum permissions that you must maintain for the Exchange store.

Appendix B, "Additional Resources"

This section contains additional resources to help you maximize your understanding of the Exchange store permission information that is discussed in this book.

Who Should Read This Book?

This book is intended for advanced Exchange administrators. This book provides a high-level explanation of the process by which Exchange controls access to objects in the Exchange store.

What Terminology Is Used in This Book?

To understand this book, make sure you are familiar with the following terms taken from the *Distributed Systems Guide* volume of the *Window 2000 Resource Kit* (<http://www.microsoft.com/windows2000>):

access control entry (ACE)

An entry in an access control list (ACL) containing the security ID (SID) for a user or group and an access mask that specifies which operations by the user or group are allowed, denied, or audited.

access control list (ACL)

A list of security protections that apply to an entire object, a set of the object's properties, or an individual property of an object. There are two types of access control lists: discretionary and system.

access token

A data structure containing security information that identifies a user to the security subsystem on a computer running Windows 2000 or Windows NT. Access tokens contain a user's security ID, the security IDs for groups that the user belongs to, and a list of the user's privileges on the local computer.

discretionary access control list (DACL)

The part of an object's security descriptor that grants or denies specific users and groups permission to access the object. Only the owner of an object can change permissions granted or denied in a DACL; thus access to the object is at the owner's discretion.

permission

A rule associated with an object to regulate which users can gain access to the object and in what manner. Permissions are granted or denied by the object's owner.

security descriptor

A data structure that contains security information associated with a protected object. Security descriptors include information about who owns the object, who may access it and in what way, and what types of access will be audited.

security ID (SID)

A data structure of variable length that uniquely identifies user, group, service, and computer accounts within an enterprise. Every account is issued a SID when the account is first created. Access control mechanisms in Windows 2000 identify security principals by SID rather than by name.

security principal

An account-holder, such as a user, computer, or service. Each security principal within a Windows 2000 domain is identified by a unique security ID (SID). When a security principal logs on to a computer running Windows 2000, the Local Security Authority (LSA) authenticates the security principal's account name and password. If the logon is successful, the system creates an access token. Every process executed on behalf of this security principal will have a copy of its access token.

system access control list (SACL)

The part of an object's security descriptor that specifies which events are to be audited per user or group. Examples of auditing events are file access, logon attempts, and system shutdowns.

For more information about Microsoft Windows 2000 security concepts, see "Access Control" in Chapter 12 of the *Distributed Systems Guide* volume of the *Microsoft Windows 2000 Resource Kit* (<http://www.microsoft.com/windows2000>).

Background

Because of the way Exchange data can be distributed around an Exchange deployment, and as a result of efforts to maintain backward compatibility with previous versions of Exchange, access control in Exchange Server 2003 is not straightforward.

The Effects of Distributed Data

Using the architecture of Exchange Server 2003, you can place multiple mailbox databases and public folder databases on different servers. Exchange 2003 also stores some information about users in Active Directory (instead of using its own directory, the way that Exchange 5.5 did). For example, see the sample deployment of Exchange 2003 in Figure i.1. This deployment includes a domain controller, one mailbox server, and two public folder servers. The figure indicates what data is stored on each of the servers. Controlling access to Exchange user information, mailboxes, folders, and messages often involves controlling access to data on several different servers.

Note

Because Exchange front-end servers do not store data for users, mailboxes, folders, or messages, they are not shown in the diagram.

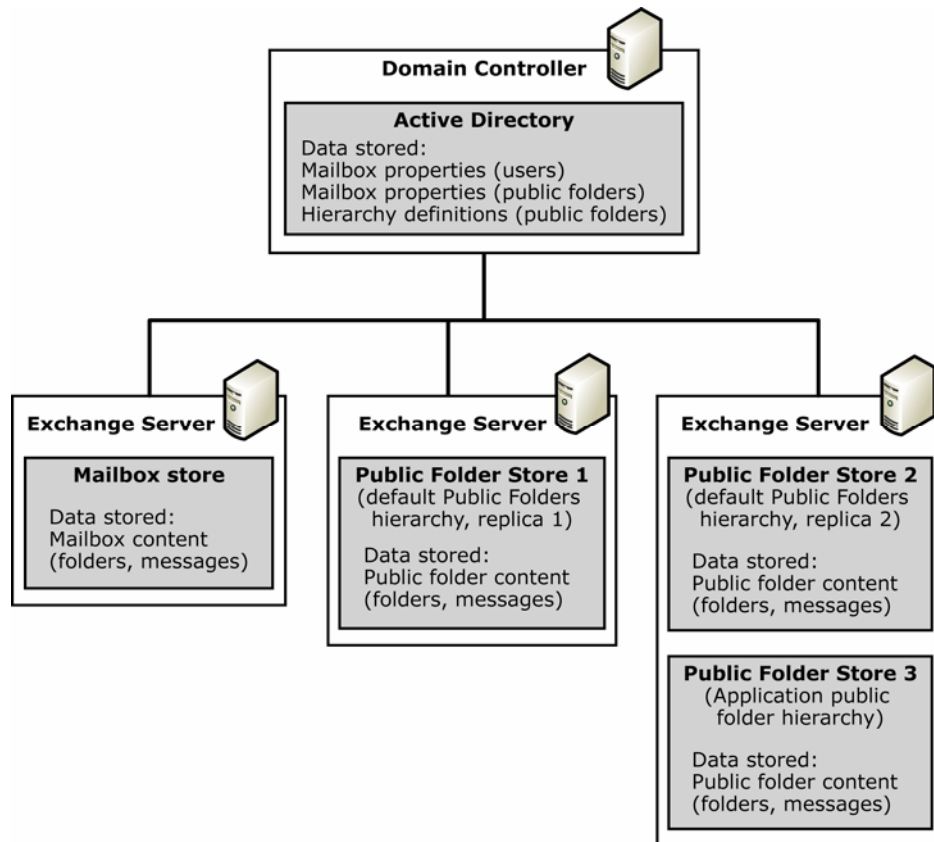


Figure i.1 Example Exchange 2003 deployment showing the distribution of data across several servers

The Effects of Backward Compatibility

Exchange 2003 uses the Windows 2000 security model. All objects (whether in the Windows file system or in Active Directory) have access control lists (ACLs), and in these ACLs, users and groups from Active Directory serve as security principals. Exchange extends this model to the Exchange store: each object in the Exchange store uses ACLs in which the security principals are users and groups from Active Directory. This is a change from Exchange 5.5, in which Exchange used its own Lightweight Directory Access Protocol (LDAP) directory for the security principals used by Exchange objects.

The ACLs on objects in the Exchange store use Windows 2000 permissions to control access (with a few additional permissions that are specific to Exchange). This is another change from Exchange 5.5, in which the ACLs used Messaging Application Programming Interface (MAPI) permissions. However, Exchange 2003 substitutes MAPI permissions for Windows 2000 permissions in the following circumstances:

- When communicating with MAPI-based client applications, such as Outlook.
- When replicating data to Exchange 5.5 servers in a deployment that contains coexisting servers that run Exchange 5.5 and servers that run Exchange 2003.

Note

Both of these circumstances apply to mailboxes and to public folders in the default Public Folders hierarchy (and all of the folders and messages that it contains). Folders and messages in application public folder hierarchies cannot be accessed by MAPI-based clients and are not replicated to Exchange 5.5 servers. Therefore, Exchange always uses Windows 2000 permissions with these folders and messages.

Exchange handles all conversions between Windows 2000 permissions and MAPI permissions automatically. However, as an administrator, be aware that when you use Exchange System Manager to set permissions, you may need to work with either Windows 2000 permissions or MAPI permissions, depending on the type of object that you are securing.

This book includes more information about how this access control process works, including information about the modifications that Exchange makes to the Windows 2000 security model to apply that model to objects in the Exchange store. This book also describes when and how Exchange converts Windows 2000 permissions to MAPI permissions (and MAPI permissions back to Windows 2000 permissions).

Accessing Exchange Objects

This chapter uses two examples to illustrate the access control process that Microsoft® Exchange uses. In the first example, a user attempts to open a folder or a message in his or her mailbox. In the second example, a user attempts to open a folder or a message in a public folder.

Opening a Folder or a Message in a User Mailbox

When a user attempts to gain access to, or perform an operation on, a folder or message in a mailbox, Exchange uses the process that is illustrated in Figure 1.1 to determine whether or not the user is authorized to perform the operation.

Note

Windows 2000 performs the actual checks, but Exchange controls the process, sometimes imposing special rules and modifications. For a description of these rules and modifications, see Chapter 2, "Details of the Exchange Access Control Process."

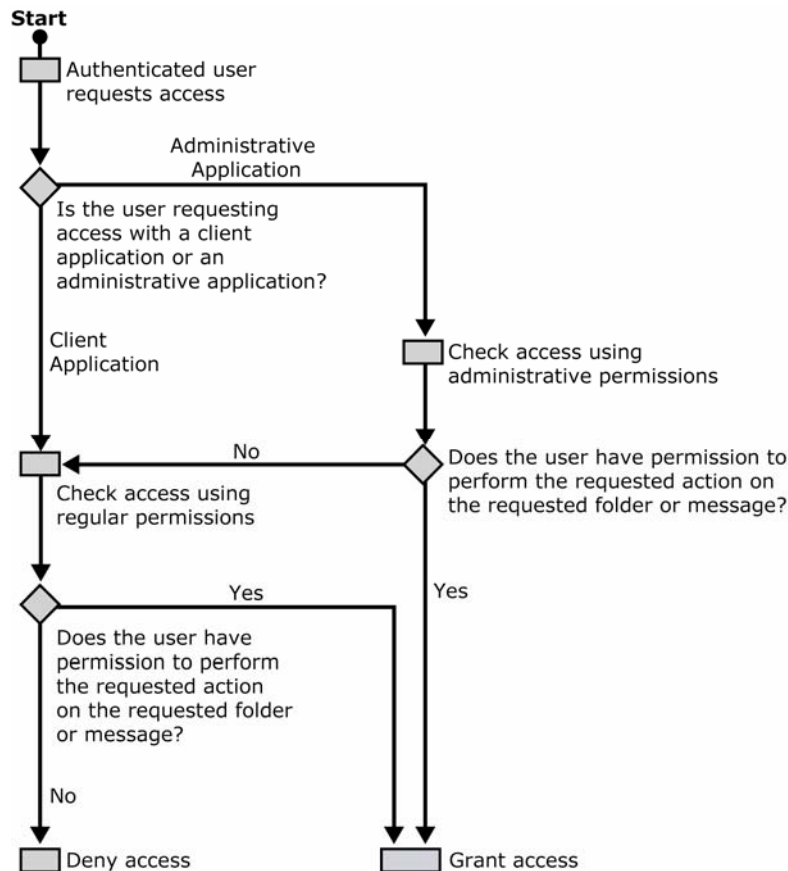


Figure 1.1 Access checks that Exchange performs when a user attempts to gain access to a folder or a message in a mailbox

As shown in the preceding figure, the authorization process takes place in two main steps: a set of preliminary checks, and a folder-level or message-level check.

Preliminary checks for user mailboxes Exchange performs three preliminary checks:

- Is the user requesting an attribute of a folder or a message? (Access control for attributes is a topic beyond the scope of this book.)
- What type of user is requesting access? Certain types of users, such as the mailbox owner, have full access permissions to all of the items in the mailbox.
- Is the user also performing an administrative action? Administrative actions (such as changing the storage limits for a particular mailbox) are controlled by a different set of permissions than those that control client actions (such as creating a new message). Exchange uses this check to determine which set of permissions to use when it checks folder or message permissions.

For information about how Exchange identifies administrative actions, see "Determining the Type of Client Application" in Chapter 2.

Folder or message check for user mailboxes Exchange handles this check in one of two ways, depending on the outcome of the first check:

- If the user is a designated special user (for example, the user is the mailbox owner), Exchange skips this check and grants full access permissions.
- If the user is not a designated special user, Exchange determines whether the user has the appropriate permissions to perform the requested action. If the user is using an administrative application but does not have the appropriate permissions to perform the requested action, Exchange performs another check using the regular permissions.

Because the second check in the authorization process occurs at the folder or message level, it is possible for a user to log on to a mailbox that the user does not own; however, the user will not be able to access items in the mailbox unless access permissions have been specified for this user for a particular folder or message.

Note

Although this example describes access checks for a user mailbox, the process is the same for a system mailbox (or any other type of mailbox that you create).

Opening a Public Folder or a Public Folder Message

When a user attempts to gain access to, or perform an operation on, a public folder or a message in a public folder, Exchange uses the process that is illustrated in Figure 1.2 to determine whether or not the user is authorized to perform the operation.

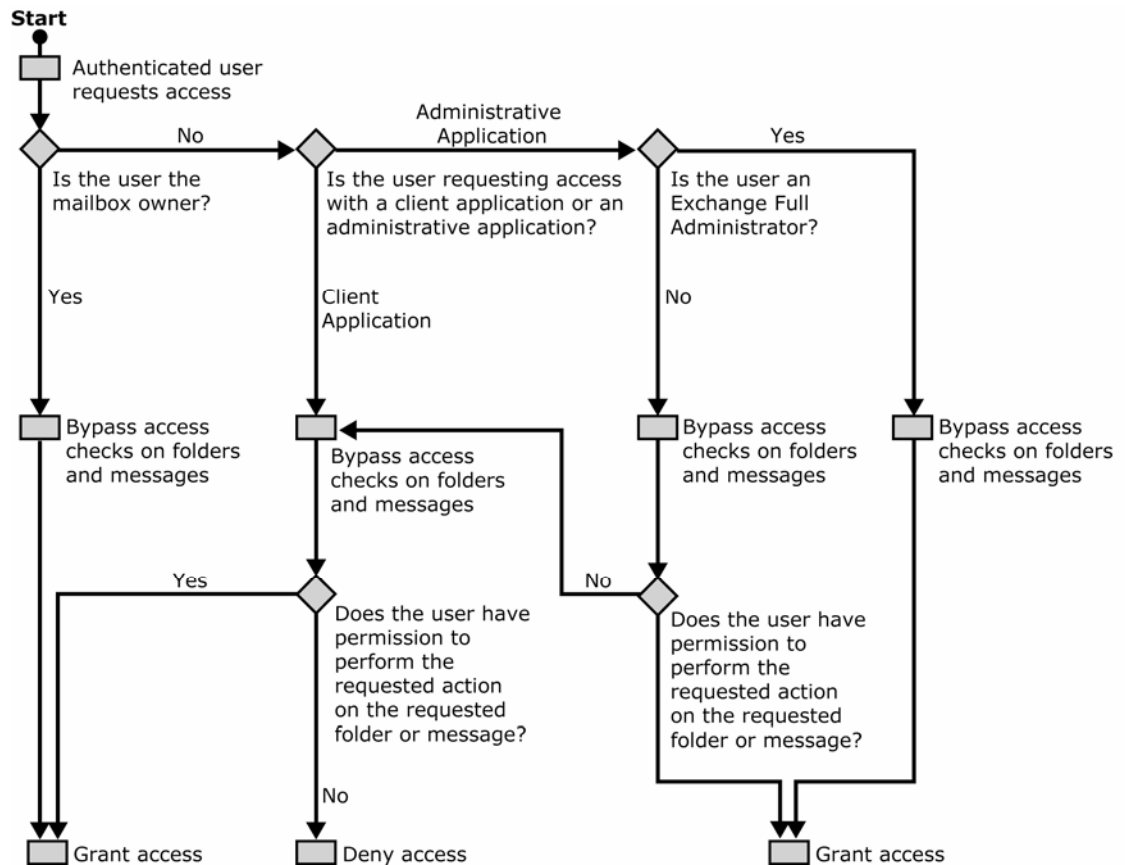


Figure 1.2 Access checks that Exchange performs when a user attempts to gain access to a public folder or a message in a public folder

As shown in the preceding diagram, the authorization process takes place in two main steps: a set of preliminary checks, and a folder-level or message-level check.

Preliminary checks for public folders or messages Exchange performs the following preliminary checks:

- Is the user requesting an attribute of a folder or a message? (Access control for attributes is a topic beyond the scope of this book.)
- Is the user also performing an administrative action?

Exchange uses this check to determine which set of permissions to use when it checks folder or message permissions. Administrative actions are controlled by a different set of permissions than the permissions that control client actions. Exchange identifies an action as "administrative" based upon the type of application that is requesting the action. If Microsoft Outlook® requests the action, Exchange treats the action as a client action. If Exchange System Manager requests the action, Exchange treats the action as an administrative action.

Even if the action is the same (such as changing permissions) and the user might think of the action as "administrative," Exchange treats the action as a client action when it is requested by Outlook and as an administrative action when it is requested by Exchange System Manager. For more information about how Exchange identifies administrative actions, see "Determining the Type of Client Application" in Chapter 2.

Folder or message check for public folders or messages Exchange handles this check in one of three ways, depending on the outcome of the preliminary checks:

- If the user is a designated special user (for example, the user is a folder owner) and the requested action is not an administrative action, Exchange skips this check and grants full access permissions.
- If the user is a designated special user and the requested action is an administrative action, Exchange determines whether the user has the appropriate administrative permissions to perform the requested action.
- If the user is not a designated special user, Exchange determines whether the user has the appropriate non-administrative permissions to perform the requested action.

Note

System folders (such as the free and busy public folder) have additional restrictions that are not addressed here.

Details of the Exchange Access Control Process

As discussed in Chapter 1, "Accessing Exchange Objects," Microsoft® Exchange uses a two-step process to control access to items in the Exchange store:

- **Preliminary checks** These checks streamline the access control process. Certain types of users always have full access to items in the Exchange store (with the exception of administrative tasks in public folders), so further checks are unnecessary, and the user is granted access to the requested item. In addition, if subsequent access checks are required (for example, a qualified user is requesting administrative access to a public folder), the preliminary checks determine which set of permissions to use for subsequent access checks.
- **Object-level access control** Each object in the Exchange store (mailbox, folder, or message) has a set of security descriptors. Each security descriptor is an attribute that contains information about which users have access to the object, and what type of access each user has. Additional controls are used to control access to special objects such as public folder trees (as opposed to individual public folders) and the extra properties of mail-enabled public folders.

Performing Preliminary Checks

Exchange uses a set of preliminary checks to determine what kind of check (if any) to use for the object-level access control check.

Note

This section describes two preliminary checks. Exchange performs another preliminary check to determine whether the user is requesting access to an attribute of a folder or message. However, a full discussion of the mechanisms that Exchange uses to implement access control on individual attributes is beyond the scope of this book.

Determining the Type of User

When a user attempts to access an object in the Exchange store, before reviewing the access control settings at the folder or message level, Exchange first checks to determine whether the user is one of the following user types:

- **Mailbox owner (for mailboxes only)** The mailbox owner has full access to the mailbox and all folders and messages in that mailbox. No further checks are performed.
- **Exchange Full Administrator** When using an administrative application, a Full Administrator has full access to all objects in the Exchange store, and no further checks are performed (with the exception of requests to perform administrative tasks on public folders—see the following section, "Determining the Type of Client Application").
- **Exchange View-Only Administrator** When using an administrative application, a View-Only Administrator has read access to all objects in the Exchange store, and no further checks are performed before granting the user this level of access. For more information about administrative applications, see the following section, "Determining the Type of Client Application."

Determining the Type of Client Application

When a user requests access to a public folder or a message in a public folder, Exchange checks whether the user matches one of the user types that is described in the preceding section. Exchange also checks what type of application the user is running. Exchange performs these checks to determine whether subsequent access checks should use the security information that controls regular client access or the security information that controls administrative client access. Exchange stores the two types of security information separately (as described in "Anatomy of Object-Level Access Control" later in this chapter).

Note

If the user who is requesting access to a public folder or a message in a public folder is running an administrative application, but does not match one of the user types described in the preceding section (for example, an end user who is not the owner of the requested object), Exchange will ignore the user's request for administrative access, and will treat the user as if he or she is using a regular client application.

As discussed in Chapter 1, "Accessing Exchange Objects," Exchange treats an action that is performed in a client application, such as Microsoft Outlook®, as a client action (one that requires regular client access) even if the user might consider the action to be "administrative." For example, when a user is changing the permissions on a public folder, if the user uses Outlook to change the permissions, Exchange treats the action as a client action (one that requires regular client access). However, if the user uses Exchange System Manager (an administrative application) to change permissions on the public folder, Exchange treats the action as administrative (one that requires administrative client access). This distinction between a client action and an administrative one provides protection against an accidental lockout. No

Outlook user (even the folder owner) can modify the permissions that control access by Exchange System Manager and thereby prevent an administrator from using Exchange System Manager to access the public folder.

- If you are developing custom applications, Exchange needs to be able to identify them as either administrative applications or end user applications. For more information, see the topics "Using the Administrative Virtual Root" in "Exchange Store URLs" and Exchange Management Security" in "CDO for Exchange Management" in the *Microsoft Exchange Software Development Kit (SDK)* (for either Exchange 2000 Server or Exchange Server 2003). You can download the *Exchange SDK* or view it online from the Exchange developer center at <http://msdn.microsoft.com/exchange>.
- For additional information about how to establish the ownership of an object, see the topics "Access Control" and "Security" in the *Exchange SDK* (<http://msdn.microsoft.com/exchange>).

Anatomy of Object-Level Access Control

To understand the way Exchange handles access control, it is necessary to examine the basics of access control as implemented in the Microsoft® Windows 2000 security model (as extended and modified by Exchange).

If you are developing custom applications, Exchange needs to be able to identify them as either administrative applications or end user applications. For more information, see the following topics in the *Exchange SDK* (<http://msdn.microsoft.com/exchange>):

- "http://schemas.microsoft.com/mapi/ Namespace" in "Properties by Namespace"
- "http://schemas.microsoft.com/exchange/security/ Namespace" in "Properties by Namespace"
- "Exchange 5.5 Access Rights and the Exchange Store" in "Exchange Store XML Security Descriptor Format"

For information about working with security descriptors on folders and messages, see "Application Security Module Reference" in "Reference" in the *Exchange SDK* (<http://msdn.microsoft.com/exchange>).

Security Descriptors

Each object (mailbox, folder, or message) in the Exchange store has two security descriptors:

- **Windows NT security descriptor** Every object in a Windows 2000 system has a Microsoft Windows NT® security descriptor. This security descriptor, described in the *Exchange SDK* as the **ntsecuritydescriptor** field (also as the **ptagNTSD** property), applies to most operations that users perform on the object (such as reading or editing messages).
- **Admin security descriptor** Every Exchange Server 2003 object has an Admin security descriptor. This security descriptor, described in the *Exchange SDK* as the **admindescriptor** field (also as the **ptagAdminNTSD** property), applies to administrative actions (such as creating or deleting public folders).

Both types of security descriptors contain the same information that is relevant to Exchange:

- Control information (information about the security descriptor, such as whether it was created explicitly or by a "defaulting" mechanism)
- Owner of the object
- Primary group to which the object owner belongs
- Discretionary Access Control List (DACL)
- System Access Control List (SACL)

Note

Security descriptors can contain additional information, but Exchange 2003 does not use that information.

For a full description of the elements of a security descriptor in Windows 2000, see Chapter 12, "Access Control," in the *Distributed Systems Guide* volume of the *Microsoft Windows 2000 Resource Kit* (<http://www.microsoft.com/windows2000>).

ACLs and ACEs

In the Exchange store, as in Windows 2000, both of the access control lists (ACLs) that are used in security descriptors—discretionary access control lists (DACLs) and system access control lists (SACLs)—consist primarily of lists of access control entries (ACEs). ACEs in the SACL pertain to auditing and alert functions, whereas ACEs in the DACL control which users can perform tasks on the secured object, and what tasks each user can perform. The ACEs in the DACL are of primary interest in this book.

Each ACE contains the following information (this list is simplified for the purposes of this book):

- ACE type (either Grant Access or Deny Access)
- Security ID (SID) of the user or group that is granted or denied access
- Specific permissions to be granted or denied (in the form of a hexadecimal access mask)
- Flags, which specify further information about how to process the ACE. The possible flag values are the following:
 - **OBJECT_INHERIT_ACE** The ACE should be inherited to objects within this container.
 - **CONTAINER_INHERIT_ACE** The ACE should be inherited to containers that are created below this container.
 - **NO_PROPAGATE_INHERIT_ACE** The ACE will only be propagated to the immediate children of this container, but not to containers that are created below the immediate children.
 - **INHERIT_ONLY_ACE** The ACE is not effective on this object, but is instead effective only on objects that are created below this object.
 - **INHERITED_ACE** The ACE was inherited from a parent object.

The flags **OBJECT_INHERIT_ACE**, **CONTAINER_INHERIT_ACE**, and **INHERIT_ONLY_ACE** are of particular importance to the way that Exchange manages security. Exchange uses these flags to identify whether an ACE applies to folders or to messages, as follows:

- Folder ACEs carry the **CONTAINER_INHERIT_ACE** flag.
- Message ACEs carry the **OBJECT_INHERIT_ACE** and **INHERIT_ONLY_ACE** flags.

ACE Sequences in the ACL

Some of the Windows NT security descriptors that are used on objects in the Exchange store use a different sequence of ACEs in the DACL than the sequence that is used by standard Windows NT security descriptors ("standard" Windows NT security descriptors are used by objects in the Active Directory® directory service and the Windows 2000 file system).

For Application public folder hierarchies, Exchange supports the standard Windows 2000 rules for ordering ACEs.

However, MAPI-based clients such as Outlook use a MAPI-based ACL format instead of the Windows 2000 format. To support these clients, Exchange uses a special set of rules for the security descriptors of mailboxes and for folders in the default Public Folders hierarchy (also known as the MAPI hierarchy). As a result of the special format, Windows evaluates whether a

user has access in a manner that mimics the access control behavior of Exchange 5.5 (which depended on MAPI-based permissions). This format is known as the Exchange Canonical ACL format (also called the Exchange Canonical security descriptor format).

The following rules control the sequence of ACEs in an Exchange Canonical ACL:

- Each ACL contains both ACEs that apply to folders and ACEs that apply to messages within those folders. Folder and message ACEs can be intermixed in the ACL.
- For each Grant ACE, there must be a corresponding Deny ACE for the same security principal (unless no rights are granted or denied). The Grant ACE must precede the Deny ACE.
- All of the ACEs for users must precede all of the ACEs for groups (corresponding to Exchange 5.5 distribution lists). ACEs for the groups **Everyone** and **Anonymous** must occur last in the sequence.
- All Grant ACEs for distribution groups must precede the corresponding Deny ACEs for these groups.

Table 2.1 shows an example of the sequence of ACEs in an Exchange Canonical ACL.

Table 2.1 Example of the sequence of information in an Exchange Canonical ACL

ACE type	Security principal
Grant	User <A>
Deny	User <A>
Grant	User
Deny	User
Grant	Distribution group <C>
Grant	Distribution group <D>
Deny	Distribution group <C>
Deny	Distribution group <D>
Grant	Everyone

With the ACL in this format, Exchange can translate the security descriptor information into the MAPI permissions that are expected by MAPI clients such as Outlook.

When a user sets permissions on a mailbox in Outlook, or when you set permissions on a mailbox or a MAPI public folder in Exchange System Manager, the user interface lists MAPI permissions and roles that are consistent with those used by Exchange 5.5. Figure 2.1 shows the dialog box you see in Exchange System Manager. If you are using Outlook, this information appears on the **Permissions** tab of the folder's **Properties** dialog box.

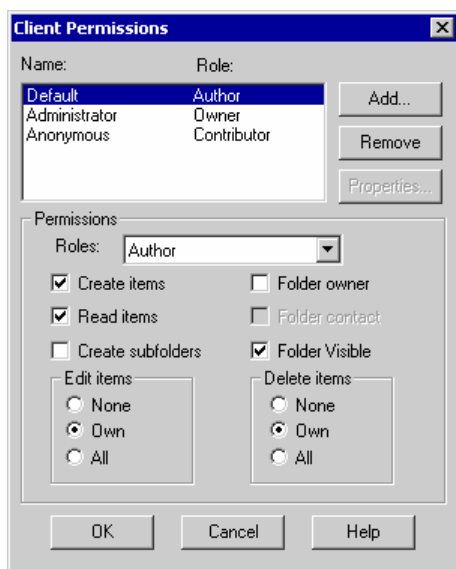


Figure 2.1 Client Permissions dialog box for a public folder in the default Public Folders hierarchy

Because application public folders are intended for access by HTTP or Network News Transfer Protocol (NNTP) clients (for example, Outlook Web Access), they do not use this conversion to MAPI permissions. As shown in Figure 2.2, the permissions that are listed are normal Windows 2000 permissions. For more information about application public folders (also called general-purpose public folders) see the *Exchange Server 2003 Administration Guide* (<http://www.microsoft.com/exchange/library>).

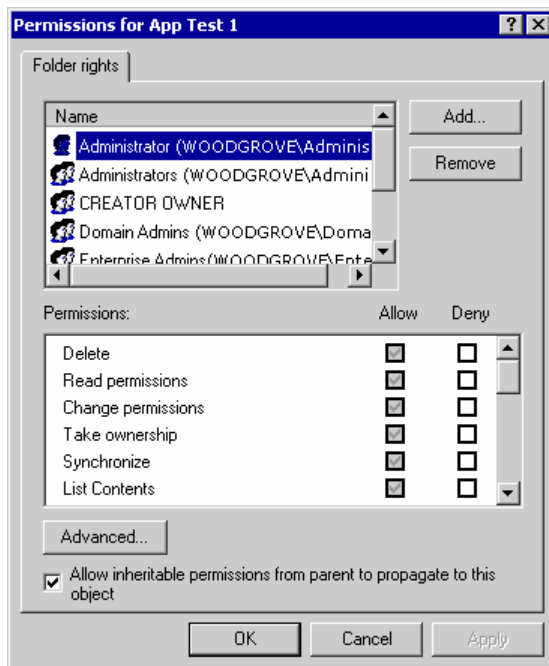


Figure 2.2 Client Permissions dialog box for a public folder in an application public folder hierarchy

Warning

Use only the **Permissions** dialog box provided by Outlook and the **Client Permissions** dialog box provided by Exchange System Manager to modify MAPI permissions. When you use the Outlook or Exchange System Manager user interface to edit MAPI permissions, Exchange reverses the translation to store the modified security descriptor and maintains the Exchange Canonical ACL format. However, if you edit permissions using the Windows file system user interface (for example, by using the M:\ drive supplied by the Installable File System in Exchange 2000 Server), the security descriptor will be stored using the normal Windows ACL format. This means that the ACEs will be in a different sequence, and Exchange will no longer be able to translate the security descriptor into MAPI permissions. For more information about this issue, see the Microsoft Knowledge Base Article 270905, "XADM: Unable to Set Client Permissions on Public Folders Through Exchange System Manager" (<http://support.microsoft.com/?kbid=270905>).

Default ACLs for Messages

As stated previously, each folder security descriptor contains ACEs that apply only to messages. Known collectively as the default message ACL, these ACEs provide security for messages in the folder that do not have their own security descriptor information.

When a user opens such a message, the message inherits message ACEs dynamically from the folder security descriptor. If the folder security descriptor changes, all new messages that are

opened within the folder will immediately inherit the security descriptor change. Messages that are already open will retain their original security descriptors until the messages are saved.

Exchange records the security descriptor on an attachment (or embedded message) in the database, but does not use it. Instead, the Exchange store uses the security descriptor of the outermost message for all attached messages. The store retains the security on attachments to allow a client to send a message to another user for debugging purposes. In addition, if a user copies the attached message to a top-level folder, its security descriptor will become effective immediately.

Special Considerations for Coexisting Exchange 2003 and Exchange 5.5 Servers

If your deployment includes both Exchange 2003 and Exchange 5.5 servers, you have an additional level of complexity to deal with when managing permissions, especially public folder permissions. For a more detailed explanation of how Exchange passes access control information between Exchange 2003 and Exchange 5.5 servers, see the technical article *Public Folder Permissions in a Mixed Mode Microsoft Exchange Organization* (<http://go.microsoft.com/fwlink/?LinkId=10228>). The important points that relate to managing public folder permissions are the following:

- Before any data can be replicated between Exchange 2003 and Exchange 5.5 servers, any users or groups with mailboxes on the Exchange 5.5 servers must have accounts in Active Directory.
 - If the user or group has only an Active Directory account (not a Windows NT 4.0 account), then the Active Directory account is an "enabled" account.
 - If the user or group has a Windows NT 4.0 account, then the Active Directory account is a "disabled" account. This account, created using the Active Directory Migration Tool, is a placeholder that associates an Active Directory SID with the existing Windows NT 4.0 account.

Important

If you plan to maintain user accounts in Windows NT 4.0 for a period of time and then fully migrate those accounts to Active Directory, you will need to create disabled accounts that have a SID history. The Active Directory Migration Tool can migrate the Windows NT 4.0 SID into the **sidHistory** attribute of the new disabled account in Active Directory. If you enable the accounts at a later date, Exchange can use the SID history information to determine where newly enabled accounts have replaced Windows NT 4.0 accounts in ACEs. For more information about this process, see the Knowledge Base Article 316047, "XADM: Addressing Problems That Are Created When You Enable ADC-Generated Accounts" (<http://support.microsoft.com/?kbid=316047>).

Exchange 5.5 uses MAPI-based permissions, identifies users and groups by their distinguished names in the Exchange Directory, and uses a property called **ptagACLData** to store access control information. When Exchange 2003 replicates access control information to an Exchange 5.5 server, it does the following:

- a. Converts the Active Directory security identifiers (SIDs) of users and groups to Exchange Directory distinguished names.
- b. Converts the Windows 2000 permissions to MAPI permissions.
- c. Stores the converted access control information in **ptagACLData**.
- d. Replicates **ptagNTSD**, **ptagAdminNTSD**, and **ptagACLData** to the Exchange 5.5 server.

When an Exchange 2003 server receives data replicated by an Exchange 5.5 server, it does the following:

- a. Discards the incoming values of **ptagNTSD** and **ptagAdminNTSD**. This step protects against any changes that may have been made to these properties while they were under the control of Exchange 5.5.
 - b. Extracts the user and group distinguished names from **ptagACLData** and converts them to Active Directory SIDs.
 - c. Extracts the permissions from **ptagACLData** and converts them to Windows 2000 permissions.
 - d. Stores the converted access control information in **ptagNTSD**. The original value of **ptagAdminNTSD** remains unaffected.
 - e. Discards the value of **ptagACLData**, unless a problem occurred during the conversion in Step b or Step c. If a conversion problem occurs, Exchange 2003 keeps the **ptagACLData** value.
- Exchange 5.5 applies permissions to folders. You cannot assign permissions to individual messages (item-level permissions) explicitly, as you can with Exchange 2003. If you are replicating folders and their contents from Exchange 5.5 to Exchange 2003, do not attempt to set explicit permissions on messages. Exchange 2003 manages permissions so that the messages are secure, but if you attempt to change the message permissions in this situation, the changes will be lost in the next replication cycle.

Modifying Permissions

Two primary applications provide access to Microsoft® Exchange security descriptor information:

- Microsoft Outlook®, through which users can set permissions on mailbox folders and can create new public folders
- Exchange System Manager, through which administrators can set permissions on public folders and public folder trees

Note

Exchange System Manager also provides access to the security descriptors of mailbox and public folder databases. However, Exchange uses standard Windows 2000 security descriptors for the databases, without any special format requirements.

Important

You can also use the ADSI Edit MMC console to view security descriptors on Exchange-related objects in the Active Directory® directory service. However, if you use ADSI Edit to change the permissions that are stored in these security descriptors, ADSI Edit saves them using the Windows 2000 ACL format instead of the Exchange Canonical ACL format. To ensure that permissions remain in the format that Exchange expects, use Outlook or Exchange System Manager whenever possible to edit permissions.

For information about working with permissions programmatically, see the "Security" topic in the "Key Tasks" section of the *Microsoft Exchange Software Development Kit (SDK)*. You can download the *Exchange SDK* or view it online from the Exchange developer center at <http://msdn.microsoft.com/exchange>.

Controlling Client Access to Mailboxes

The security descriptor for the mailbox object resides both in the Exchange store and in the user object in Active Directory. In the user object, the mailbox security descriptor is stored using a special attribute. This attribute keeps the mailbox security descriptor separate from the security descriptor of the user object, which is a normal Windows security descriptor.

When you create a new mailbox-enabled user, the mailbox inherits a security descriptor from the mailbox database. This security descriptor is stored in Active Directory. The Exchange store does not actually create the mailbox until the first time the user opens the mailbox, at which time Exchange creates the security descriptor in the store. The default folders in the new mailbox inherit security descriptors from the mailbox.

Users can modify the permissions on folders or messages in their mailboxes using Outlook.

To view the permissions for a mailbox folder

1. In Outlook, right-click the folder that you want to change, and then click **Properties**.
2. In the **Properties** dialog box, click the **Permissions** tab (see Figure 3.1).

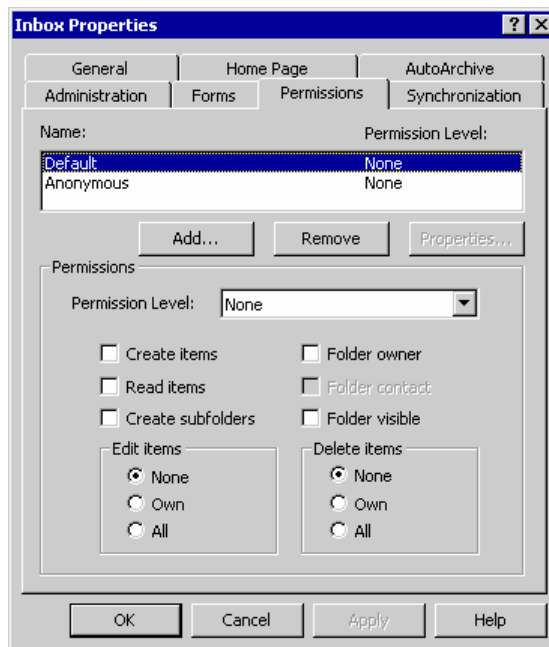


Figure 3.1 Setting permissions on a mailbox folder using Microsoft Outlook

Note

The **Permissions** tab uses MAPI permissions and roles.

Controlling Client Access to Public Folders

As with mailboxes, security descriptor information for public folders resides both in the Exchange store and in Active Directory. However, the way Exchange manages security descriptors for public folders is much more complex.

Table 3.1 lists the different locations in which Exchange places public folder information.

Table 3.1 Locations of security information for public folders

Data	Location	Security information
Public folder objects (one object for each folder, including the top folder in the hierarchy)	Exchange store (public folder databases)	Security descriptors (ntsecuritydescriptor and Admindescriptor for each folder object)
Folder hierarchy objects (one object for each hierarchy, used for hierarchy-wide attributes such as hierarchy type)	Active Directory (Configuration container)	Security descriptors (ntsecuritydescriptor and Admindescriptor for each hierarchy object) Additional security attributes (These attributes, although not used as security descriptors by Active Directory, hold information that is needed to create default security descriptors for new top-level folder objects in the store.)
Public folder proxy objects (one object for each mail-enabled public folder, used to hold mail attributes for public folders)	Active Directory (Domain container)	Security descriptors (ntsecuritydescriptor and Admindescriptor for each proxy object)

When you create a new top-level public folder (in an existing hierarchy), Exchange creates the security descriptor for the folder by combining the security descriptor of the folder hierarchy object with stored permissions information.

As an Exchange administrator, you are already familiar with the view that Exchange System Manager provides of objects in the Exchange store (see Figure 3.2). This is the interface that you normally use when you are modifying permissions on objects in the store.

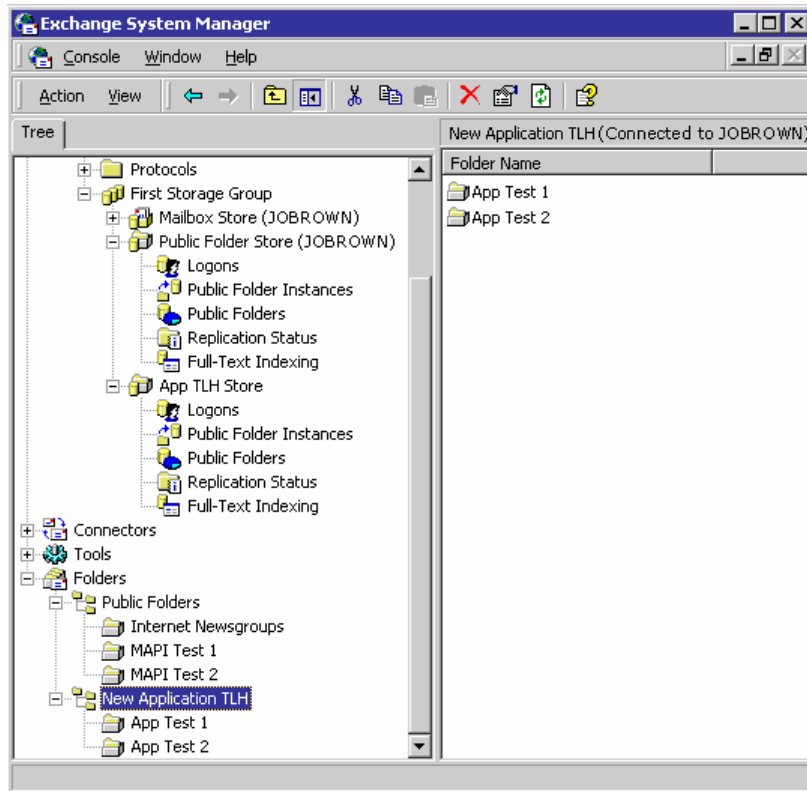


Figure 3.2 The default Public Folders tree and application public folder trees in Exchange System Manager

To view permissions that control client access to a public folder

1. In Exchange System Manager, right-click the folder for which you want to view permissions, and then click **Properties**.

- In the **Properties** dialog box, click the **Permissions** tab, and then click **Client permissions** (see Figure 3.3).

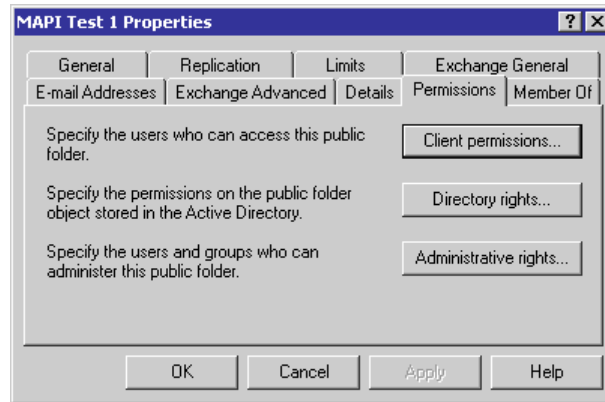


Figure 3.3 First Permissions tab displayed for a mail-enabled public folder

Note

The **Directory rights** and **Administrative rights** options are addressed later in this chapter.

After clicking **Client permissions**, you will be presented with one of two different dialog boxes, depending on the type of public folder hierarchy with which you are working. If you are working with a folder in the default Public Folders hierarchy, you will see a dialog box that contains MAPI permissions and roles, as shown in Figure 3.4.

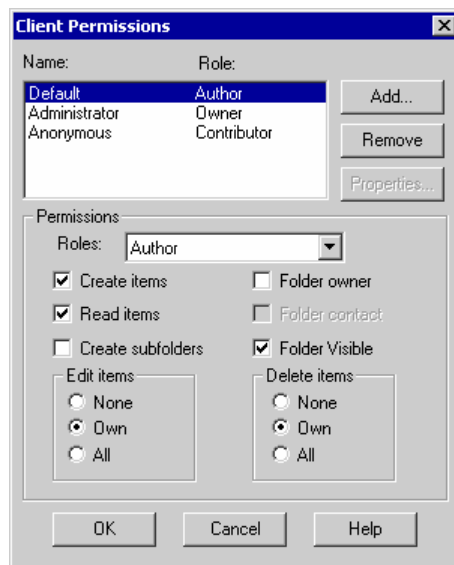


Figure 3.4 Client Permissions dialog box for a public folder in the default Public Folders hierarchy

If you are working with a folder in an application public folder hierarchy, you will see a dialog box that contains Windows 2000 permissions, users, and groups, as shown in Figure 3.5.

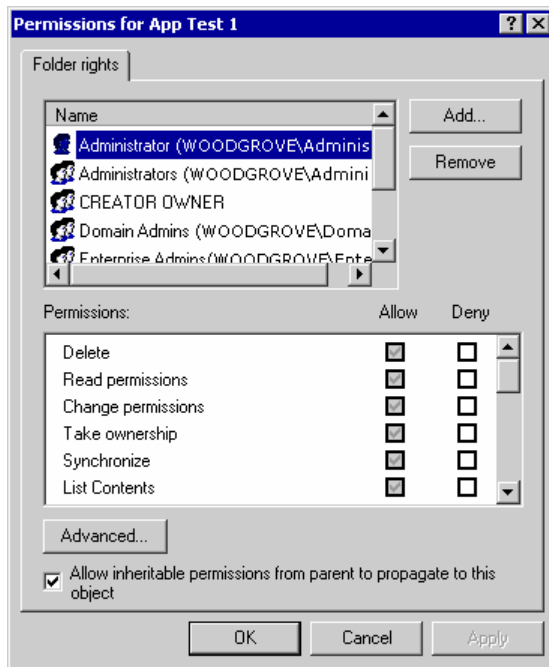


Figure 3.5 Client Permissions dialog box for a public folder in an application public folder hierarchy.

You can also use Exchange System Manager to view the Windows 2000 version of the permissions on a folder in the default Public Folders hierarchy.

Warning

Although you can view the Windows 2000 version of the default Public Folders hierarchy permissions, do not attempt to edit the permissions in this view. The Windows user interface that is used to display the permissions will format the ACL in such a way that Exchange will no longer be able to convert the permissions to their MAPI form. If this happens, you will no longer be able to use Outlook or the regular Exchange System Manager dialog boxes to edit the permissions.

To view the Windows 2000 version of MAPI permissions

1. In Exchange System Manager, right-click the folder whose permissions you want to view, and then click **Properties**.

2. In the **Properties** dialog box, click the **Permissions** tab, and then press and hold the CTRL key and click **Client permissions**.

The resulting dialog box is shown in Figure 3.6. Note that all of the permissions check boxes are empty.



Figure 3.6 Windows 2000 Permissions dialog box for a folder in the default Public Folders hierarchy

3. To see the actual permissions information, click **Advanced**. The resulting dialog box is shown in Figure 3.7.

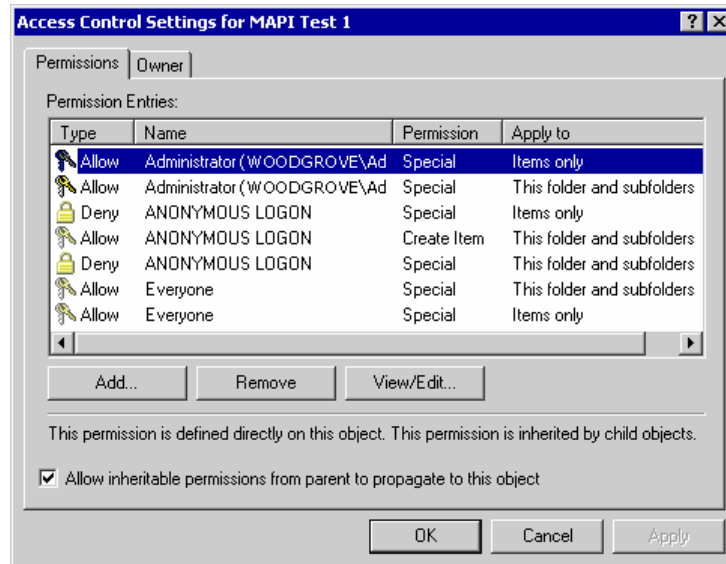


Figure 3.7 Advanced version of the Windows 2000 Permissions dialog box

4. To view detailed permissions information, click a permissions entry and then click **View/Edit**. Remember that it is highly inadvisable to edit the permissions in this manner.

Figure 3.8 shows an example of the detailed Windows 2000 permissions information that you can view.

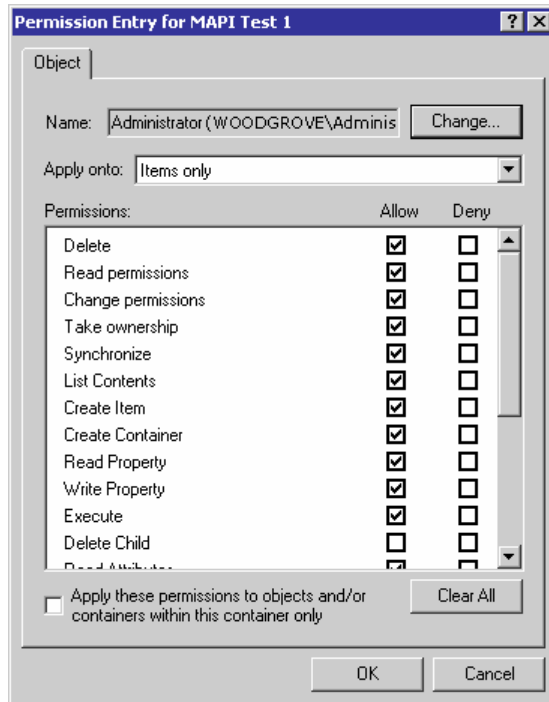


Figure 3.8 Detailed view of Windows 2000 permissions

Special Considerations for Messages

When a user opens a message, the message inherits message ACEs dynamically from the folder security descriptor. If the folder security descriptor changes, all new messages that are opened within the folder will inherit the security descriptor change immediately. Messages that are already open will retain their original security descriptors until the messages are saved.

Exchange stores the security descriptor on an attachment (or embedded message) in the database, but does not use it. Instead, the Exchange store uses the security descriptor of the outermost message for all attached messages. The store retains the security on attachments to allow a client to send a message to another user for debugging purposes. In addition, if a user copies the attached message to a top-level folder, its security descriptor will immediately become effective.

Controlling Administrative Access to Mailboxes and Public Folders

In general, the administrative security descriptor follows the Windows 2000 rules for format and inheritance.

Note

Database-level administrative tasks, such as moving users or connecting and disconnecting mailbox stores or public folder stores, are controlled by the individual database security descriptor, which is a normal Windows 2000 security descriptor.

Mailboxes

Exchange System Manager does not provide administrative access to user mailboxes. Use Active Directory Users and Computers to perform administrative tasks on the Exchange-related attributes of user objects.

Public Folders

Administrative permissions for public folders are stored both in the Exchange store (in the **admindescriptor** property of each object) and in Active Directory. These permissions are Active Directory permissions; no MAPI conversion is involved.

Whether you are modifying administrative permissions on a public folder in the default hierarchy or on a public folder in an application hierarchy, you will use a common user interface (UI)—the Windows UI—and a common set of permissions, as shown in Figure 3.9 and Figure 3.10.

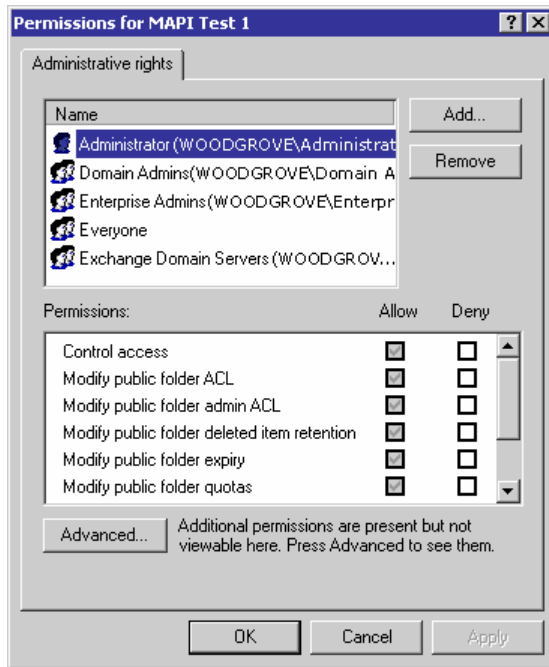


Figure 3.9 Administrative rights dialog box for a public folder in the default Public Folders hierarchy

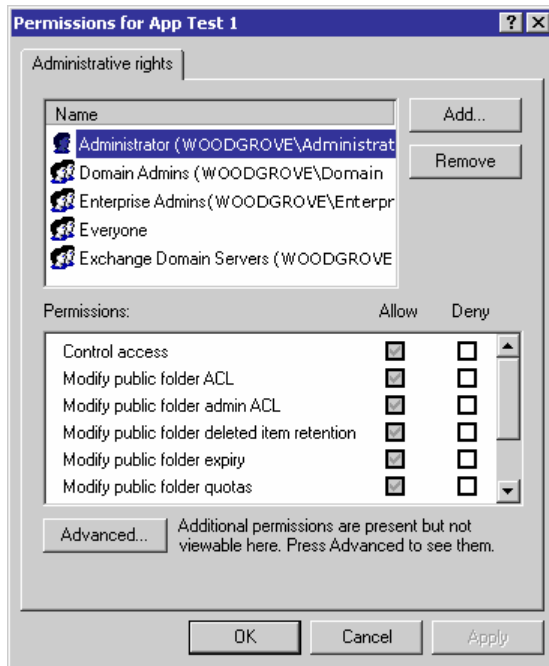


Figure 3.10 Administrative rights dialog box for a public folder in an application public folder hierarchy

To store properties that govern all folders in a hierarchy, each public folder hierarchy has a "folder hierarchy" object in Active Directory. This object stores properties such as the distinguished name of the Exchange store in which the hierarchy resides, and the hierarchy type (MAPI or application) for the hierarchy.

Important

The security descriptor of the folder hierarchy object includes permissions such as **Create public folder** that may override permissions that are set on individual public folders in the hierarchy.

Exchange System Manager provides access to some of this information by displaying a tree object for each public folder hierarchy. You can also use ADSI Edit to view the folder hierarchy objects in Active Directory, as shown in Figure 3.11.

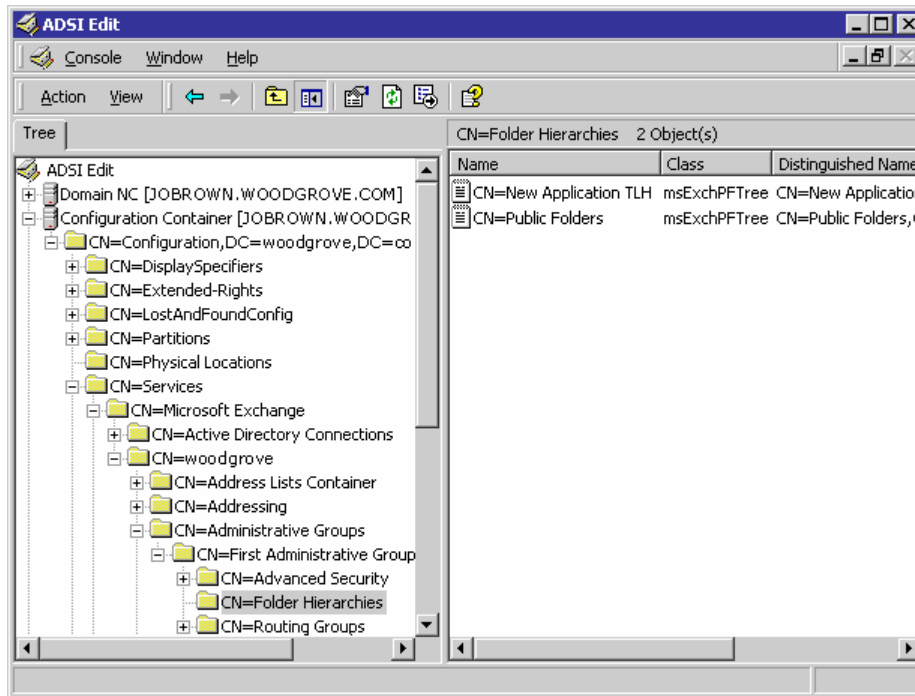


Figure 3.11 Public folder tree objects in Active Directory

When you create a new top-level folder within a hierarchy, the Exchange store copies the administrative security descriptor from the folder hierarchy object in Active Directory and uses that as the parent administrative security descriptor. For all other folders, the store copies the administrative security descriptor from the parent folder and uses that as the parent administrative security descriptor.

Note

When Exchange checks the administrative security descriptor, it first updates the inherited ACEs in the security descriptor. Exchange retrieves the current administrative security descriptor of the hierarchy object, and uses that information to replace the inherited information in the administrative security descriptor that is being checked. ACEs that have been set explicitly for the folder in question are not affected.

Note

If your deployment includes both Exchange 2003 and Exchange 5.5 servers, and the folder in question is homed on an Exchange 5.5 server, that folder will not have an administrative security descriptor. In this case, if the folder is not marked as secured to a site, or if site of the folder is the same as the current site, the store uses the regular security descriptor on the folder hierarchy object in Active Directory for the administrative security descriptor.

Special Controls for Mail-Enabled Public Folders

Mail-enabled public folders use special Active Directory objects to store their mailbox attributes, such as automatically-generated proxy addresses. Each mail-enabled folder has one such object, and the object's name is the same as the folder name. Figure 3.12 shows the location of these objects in Active Directory, as viewed with ADSI Edit.

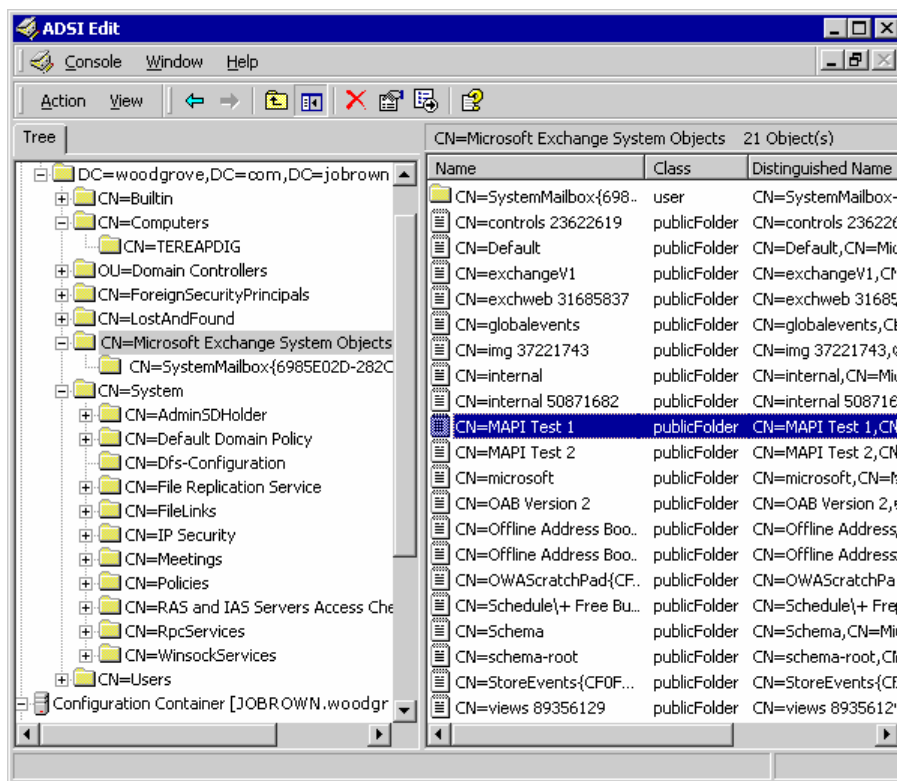


Figure 3.12 Public folder proxy objects in Active Directory

As with public folder tree objects, the permissions on public folder proxy objects are the same whether the public folder is a member of a MAPI tree or an application tree. The user interface is the standard Windows UI.

For more information about mail-enabled public folders, see the Exchange Server 2003 Help or the *Exchange Server 2003 Administration Guide* (<http://www.microsoft.com/exchange/library>).

Important

In the following procedure, it is recommended that you leave the settings at their default values. The main purpose of the security descriptor of the proxy object is to give Exchange services access to the object while limiting other access to the object.

To view permissions that control access to mail-enabled folder attributes

1. In Exchange System Manager, right-click the folder that you want to view, and then click **Properties**.
2. In the **Properties** dialog box, click the **Permissions** tab, and then click **Directory rights** (see Figure 3.13).



Figure 3.13 First **Permissions** tab displayed for a mail-enabled public folder

Permissions Available in Exchange

The following sections list the different permissions that are available in Microsoft® Exchange. For more detailed information about most of these permissions (such as their hexadecimal, access-mask values), see the Exchange Server 2003 documentation at <http://www.microsoft.com/exchange/library>, or the Microsoft Windows® 2000 or Active Directory® directory service documentation in the *Microsoft Platform SDK*, available on the MSDN® Web site at <http://www.msdn.microsoft.com>.

Special Permissions

Exchange uses special permissions internally to govern access to mailboxes, in conjunction with the preliminary checks that it performs. Exchange does not use these permissions for security descriptors. Table 4.1 lists these mailbox-specific permissions.

Table 4.1 Mailbox-specific permissions

Exchange permissions	Member permissions or Windows equivalents
fsdpermUserDeleteMailbox	DELETE
fsdpermUserMailboxOwner	
fsdpermUserSendAs	
fsdpermUserPrimaryUser	(Used in conjunction with the Master Account SID in Active Directory)
sdpermUserGenericRead	STANDARD_RIGHTS_READ
sdpermUserGenericExecute	STANDARD_RIGHTS_EXECUTE

Exchange permissions	Member permissions or Windows equivalents
sdpermUserGenericWrite	STANDARD_RIGHTS_WRITE fsdpermUserDeleteMailbox
sdpermUserGenericAll	STANDARD_RIGHTS_ALL FsdpermUserMailboxOwner FsdpermUserSendAs fsdpermUserPrimaryUser

Permissions Used in the Windows NT Security Descriptor

The following tables list the permissions that are used in the Microsoft Windows NT® security descriptor. For more information about these permissions, see the "Security" topic in the "Key Tasks" section of the *Microsoft Exchange Software Development Kit (SDK)* (<http://msdn.microsoft.com/exchange>).

Permissions for Mailbox Folders and Public Folders

The permissions used in the Windows NT security descriptor of a folder are the same whether the folder resides in a mailbox or in a public folder hierarchy. Table 4.2 and Table 4.3 list these permissions (the permissions listed in Table 4.3 are available on objects throughout Windows 2000).

In Table 4.2, an asterisk (*) denotes an Exchange-specific permission.

Table 4.2 Folder permissions

Display name	Permission	Description
List contents	fsdtrightListContents	Same as FILE_LIST_DIRECTORY. Trustee can list file contents.
Create item	fsdtrightCreateItem	Same as FILE_ADD_FILE. Trustee can add a file to a folder.

Display name	Permission	Description
Create container	fsdrightCreateContainer	Same as FILE_ADD_SUBDIRECTORY. Trustee can add a subfolder.
Read property	fsdrightReadProperty	Same as FILE_READ_EA.
Write property	fsdrightWriteProperty	Same as FILE_WRITE_EA.
Read attributes	fsdrightReadAttributes	Same as FILE_READ_ATTRIBUTES. Reserved for future use.
Write attributes	fsdrightWriteAttributes	Same as FILE_WRITE_ATTRIBUTES. Reserved for future use.
Write own property	fsdrightWriteOwnProperty*	The trustee can modify his or her own items.
Delete own item	fsdrightDeleteOwnItem	The trustee can delete his or her own items.
View item	fsdrightViewItem*	The trustee can view items.
Owner	fsdrightOwner*	The trustee is the owner of the folder. This right corresponds to the frightsOwner right in previous versions of Exchange and is provided for backward compatibility.
Contact	fsdrightContact*	Not used for security. Identifies the user as the contact for the folder. This right corresponds to the frightsContact right in previous versions of Exchange and is provided for backward compatibility.
-	fsdrightReserved1	Same as FILE_DELETE_CHILD. Currently unused.

Note

The permissions fsdrightReadProperty and fsdrightReadAttributes are related and are always granted or denied together. Similarly, fsdrightWriteProperty and fsdrightWriteAttributes are always granted or denied together.

Table 4.3 Standard Windows permissions used in folder security descriptors

Display name	Permission	Member permission or Windows equivalent
Delete	fsdrightDelete	DELETE
Read permissions	fsdrightReadControl	READ_CONTROL
Change permissions	fsdrightWriteSD	WRITE_DAC
Take ownership	fsdrightWriteOwner	WRITE_OWNER
Synchronize	fsdrightSynchronize	SYNCHRONIZE
-	sdrightsFolderOwner	fsdrightWriteProperty fsdrightOwner fsdrightWriteSD fsdrightDelete fsdrightWriteOwner fsdrightWriteAttributes

Permissions for Messages

Table 4.4 through Table 4.7 list the message permissions.

Table 4.4 Message permissions

Display name	Permission	Member permission or Windows equivalent
Read body	fsdrightReadBody	Only on messages, same as FILE_READ_DATA.
Write body	fsdrightWriteBody	Only on messages, same as FILE_WRITE_DATA.
Append message	fsdrightAppendMsg	Only on messages, same as FILE_WRITE_DATA. Enforced by IFS.
Read property	fsdrightReadProperty	Same as FILE_READ_EA.

Display name	Permission	Member permission or Windows equivalent
Write property	fsdrightWriteProperty	Same as FILE_WRITE_EA.
Execute	fsdrightExecute	Same as FILE_EXECUTE/FILE_TRAVERSE. Enforced by IFS.
Read attributes	fsdrightReadAttributes	Same as FILE_READ_ATTRIBUTES. Currently unused.
Write attributes	fsdrightWriteAttributes	Same as FILE_WRITE_ATTRIBUTES. Currently unused.
Write own property	fsdrightWriteOwnProperty	Only on messages.
Delete own item	fsdrightDeleteOwnItem	Only on messages.
View item	fsdrightViewItem	

Note

The permissions fsdrightReadProperty, fsdrightReadAttributes, and fsdrightReadBody are related and are always granted or denied together. Similarly, fsdrightWriteProperty, fsdrightWriteAttributes, and fsdrightWriteBody are always granted or denied together.

Note

The permissions listed in Tables 4.5 and 4.6 do not appear in the user interface. However, they can be used in custom applications.

Table 4.5 Standard Windows permissions used in message security descriptors

Display name	Permission	Member permission or Windows equivalent
Delete	fsdrightDelete	DELETE
Read permissions	fsdrightReadControl	READ_CONTROL
Change permissions	fsdrightWriteSD	WRITE_DAC
Take ownership	fsdrightWriteOwner	WRITE_OWNER

Display name	Permission	Member permission or Windows equivalent
Synchronize	fsdrightSynchronize	SYNCHRONIZE

Note

In Table 4.6, access rights that belong to the sdrightsIgnored permission are ignored in the determination of an Exchange Canonical ACL. Because the Exchange store ignores these rights, their presence or absence doesn't make an ACL canonical.

Table 4.6 Groups of message permissions

Permission	Member permission or Windows equivalent
sdrightsNone	
sdrightsBestAccess	MAXIMUM_ALLOWED
sdrightsReadOnly	GENERIC_READ
sdrightsReadWrite	GENERIC_READ GENERIC_WRITE
sdrightsGenericRead	fsdrightReadControl fsdrightReadBody fsdrightReadAttributes fsdrightReadProperty fsdrightViewItem fsdrightSynchronize

Permission	Member permission or Windows equivalent
sdrightsGenericWrite	fsdrightReadControl fsdrightWriteBody fsdrightWriteAttributes fsdrightWriteProperty fsdrightAppendMsg fsdrightCreateItem fsdrightDelete fsdrightCreateContainer fsdrightOwner fsdrightSynchronize fsdrightWriteSD fsdrightWriteOwner
sdrightsGenericExecute	fsdrightReadControl fsdrightReadAttributes fsdrightExecute fsdrightViewItem fsdrightSynchronize

Permission	Member permission or Windows equivalent
sdrightsGenericAll	fsdrightDelete fsdrightReadProperty fsdrightWriteProperty fsdrightCreateItem fsdrightCreateContainer fsdrightReadControl fsdrightWriteSD fsdrightWriteOwner fsdrightReadControl fsdrightViewItem fsdrightOwner fsdrightWriteOwnProperty fsdrightDeleteOwnItem fsdrightSynchronize fsdrightExecute fsdrightReserved1 fsdrightReadAttributes fsdrightWriteAttributes fsdrightReadBody fsdrightWriteBody fsdrightSynchronize fsdrightContact
sdrightsIgnored	fsdrightExecute fsdrightAppendMsg fsdrightContact fsdrightReserved1

Table 4.7 Backward-compatible message permissions

Permission	Member permissions
msgrightsGenericRead	sdrightsGenericRead sdrightsItems
msgrightsGenericWrite	sdrightsGenericWrite sdrightsItems
msgrightsGenericExecute	sdrightsGenericExecute sdrightsItems
msgrightsGenericAll	sdrightsGenericAll sdrightsItems
fldrightsGenericRead	sdrightsGenericRead sdrightsFolders
fldrightsGenericWrite	sdrightsGenericWrite sdrightsFolders
fldrightsGenericExecute	sdrightsGenericExecute sdrightsFolders
fldrightsGenericAll	sdrightsGenericAll sdrightsFolders

Exchange provides an additional control value, `EXCHANGE_RM_SET_EXPLICIT_SD`, which can be set in the control field of a security descriptor. An administrator can then set the security descriptor of the object explicitly. For more information about setting `EXCHANGE_RM_SET_EXPLICIT_SD`, see the Windows XP API documentation.

Permissions Used in the Administrative Security Descriptor

In addition to standard Windows 2000 permissions, Exchange defines a set of extended permissions that pertain specifically to Exchange functions. Table 4.8 lists these permissions.

Note

The permissions Create public folder, Create top-level public folder, and Create named properties in the information store can apply to both administrative and non-administrative users.

Table 4.8 Extended permissions defined and used by Exchange

Display name	Common name (cn)
Add PF to admin group	ms-Exch-Add-PF-To-Admin-Group
Exchange administrator	ms-Exch-Admin-Role-Administrator
Exchange full administrator	ms-Exch-Admin-Role-Full-Administrator
Exchange public folder read-only administrator	ms-Exch-Admin-Role-Read-Only-Administrator
Exchange public folder service	ms-Exch-Admin-Role-Service
Create public folder	ms-Exch-Create-Public-Folder
Create top level public folder	ms-Exch-Create-Top-Level-Public-Folder
Mail-enable public folder	ms-Exch-Mail-Enabled-Public-Folder
Modify public folder ACL	ms-Exch-Modify-PF-ACL
Modify public folder admin ACL	ms-Exch-Modify-PF-Admin-ACL
Modify public folder deleted item retention	ms-Exch-Modify-Public-Folder-Deleted-Item-Retention
Modify public folder expiry	ms-Exch-Modify-Public-Folder-Expiry
Modify public folder quotas	ms-Exch-Modify-Public-Folder-Quotas
Modify public folder replica list	ms-Exch-Modify-Public-Folder-Replica-List
Open mail send queue	ms-Exch-Open-Send-Queue
Read metabase properties	ms-Exch-Read-Metabase-Properties
Remove PF from admin group	ms-Exch-Remove-PF-From-Admin-Group
Administer information store	ms-Exch-Store-Admin

Display name	Common name (cn)
Create named properties in the information store	ms-Exch-Store-Create-Named-Properties
View information store status	ms-Exch-Store-Visible
Send-As	Send-As
Receive-As	Receive-As

Conversion Between MAPI and Windows Permissions

As described in "ACE Sequences in the ACL" in Chapter 2, Exchange routinely converts Windows 2000 permissions to and from MAPI permissions. This section lists the MAPI permissions that Exchange 2003 uses and demonstrates how Exchange converts permissions from one form to the other.

Available MAPI Permissions

Tables 5.1 and 5.2 list the MAPI permissions that Exchange 2003 uses. Only the permissions in Table 5.1 are available in the user interface. The permissions in Table 5.2 are available programmatically.

Table 5.1 MAPI permissions available in the user interface

Display name	Permission
Read items	frightsReadAny
Create items	frightsCreate
Edit items: Own	frightsEditOwned
Delete items: Own	frightsDeleteOwned
Edit items: All	frightsEditAny
Delete items: All	frightsDeleteAny
Create subfolders	frightsCreateSubfolder

Display name	Permission
Folder owner	frightsOwner
Folder contact	frightsContact (not part of rightsAll)
Folder visible	frightsFolderVisible

Table 5.2 Groups of MAPI permissions available programmatically

Permission	Member permissions
rightsNone	None
rightsReadOnly	frightsReadAny
rightsReadWrite	frightsReadAny frightsEditAny
rightsAll	All

Converting to MAPI Permissions

Exchange uses the following process to convert Windows 2000 permissions to MAPI permissions.

1. Convert the message ACEs, as described in Table 5.3.

Table 5.3 Message ACEs

Exchange 2003 permissions	MAPI permissions
fsdrightReadProperty	frightsReadAny
fsdrightWriteOwnProperty	frightsEditOwned
fsdrightWriteProperty	frightsEditAny
fsdrightDeleteOwnItem	frightsDeleteOwned

Exchange 2003 permissions	MAPI permissions
fsdrightDelete	frightsDeleteAny

- Convert the folder ACEs, as described in Table 5.4.

Table 5.4 Folder ACEs

Exchange 2003 permissions	MAPI permissions
fsdrightCreateContainer	frightsCreateSubfolder
fsdrightContact	frightsContact
If all of the folder owner rights are present: fsdrightWriteProperty fsdrightOwner fsdrightWriteSD fsdrightDelete fsdrightWriteOwner fsdrightWriteAttributes fsdrightViewItem	frightsOwner
fsdrightCreateItem	frightsCreate

- For either folders or messages, if fsdrightViewItem is present, map it to frightsVisible.
- Ensure that the converted permissions are internally consistent by making the following additional changes:
 - If frightsDeleteAny is granted, grant frightsDeleteOwned.
 - If frightsEditAny is granted, grant frightsEditOwned.
 - If frightsReadAny or frightsOwner is granted, grant frightsVisible.

Converting from MAPI Permissions

Exchange uses the following process to convert MAPI permissions to Windows 2000 permissions.

Important

Exchange overwrites the existing security descriptor ACEs with the freshly converted ACEs. The same conversion takes place whether you are using the MAPI user interface to modify permissions, or if the ACEs have been replicated from a coexisting Exchange 5.5 server. If the ACEs have been replicated from Exchange 5.5, the conversion starts with the preliminary step of converting the Exchange 5.5 distinguished name of the user or group to a Windows 2000 security identifier.

1. Expand the MAPI ACE into two ACEs.
 - An object-inherit/inherit-only (OI/IO) ACE (for messages; ignored on folders)
 - A container-inherit (CI) ACE (for folders; ignored on messages)
2. Convert the permissions in the ACEs, as described in Table 5.5.

Table 5.5 Permissions in the ACEs

MAPI permission	Exchange 2003 permission
frightsEditAny	All of the legal generic write bits on messages (except delete)
frightsDeleteAny	fsdrightDelete
frightsEditOwned	fsdrightWriteOwnProperty
frightsDeleteOwned	fsdrightDeleteOwnItem
frightsReadAny	All of the legal generic read bits on messages All of the legal generic execute bits on messages, including fsdrightViewItem on messages On folder ACEs, this affects fsdrightViewItem
frightsCreateSubfolder	fsdrightCreateContainer

MAPI permission	Exchange 2003 permission
frightsOwner	fsdrightWriteProperty fsdrightOwner fsdrightWriteSD fsdrightDelete fsdrightWriteOwner fsdrightWriteAttributes fsdrightViewItem
frightsContact	fsdrightContact
frightsVisible	All of the legal generic read bits on folders All of the legal generic execute bits on folders fsdrightViewItem

3. Verify that the required Grant/Deny pairs of ACEs are present and in the correct sequence.

Examples of the Conversion Process

These examples illustrate the process of converting Windows 2000 permissions to MAPI permissions, and back again. As a starting point, consider a security descriptor on a folder in the Exchange store. The security descriptor contains two ACEs that specify permissions that are granted to the FolderUsers group:

- A folder ACE (an ACE that carries the CONTAINER_INHERIT_ACE flag)
- A message ACE (an ACE that carries the OBJECT_INHERIT_ACE and INHERIT_ONLY_ACE flags)

To simplify the discussion, these ACEs grant all applicable permissions to the FolderUsers group. Otherwise, two additional ACEs would be required to deny any permissions that were not granted.

Table 5.6 lists the permissions that are granted by the two ACEs.

Table 5.6 Permissions specified by the FolderUsers group's two Grant ACEs

Permissions in the folder ACE	Permissions in the message ACE
fsdrightSynchronize	fsdrightSynchronize
fsdrightDelete	fsdrightDelete
fsdrightReadControl	fsdrightReadControl
fsdrightWriteSD	fsdrightWriteSD
fsdrightWriteOwner	fsdrightWriteOwner
fsdrightOwner	fsdrightWriteAttributes
fsdrightWriteAttributes	fsdrightWriteOwnProperty
fsdrightViewItem	fsdrightDeleteOwnItem
fsdrightWriteProperty	fsdrightViewItem
fsdrightExecute	fsdrightWriteProperty
fsdrightReserved1	fsdrightExecute
fsdrightReadAttributes	fsdrightReadAttributes
fsdrightListContents	fsdrightReadBody
fsdrightCreateItem	fsdrightWriteBody
fsdrightCreateContainer	fsdrightAppendMsg
fsdrightReadProperty	fsdrightReadProperty

This example concentrates on the folder ACE. For the moment, it is not necessary to be concerned with the message ACE.

Table 5.7 shows how the Windows 2000 permissions map to MAPI permissions. Note that in many cases a single MAPI permission maps to a group of Windows 2000 permissions, and some of the Windows 2000 permissions are dropped.

Table 5.7 Converting permissions

Windows 2000 permissions	MAPI permissions
fsdrightOwner fsdrightWriteAttributes fsdrightWriteSD fsdrightWriteOwner (fsdrightViewItem, fsdrightWriteProperty, and fsdrightDelete must also be present for this conversion to succeed)	frightsOwner
fsdrightViewItem	frightsVisible
fsdrightWriteProperty	frightsEditAny frightsEditOwned
fsdrightDelete	frightsDeleteAny frightsDeleteOwned
fsdrightCreateItem	frightsCreate
fsdrightCreateContainer	frightsCreateSubfolder
fsdrightReadProperty	frightsReadAny
fsdrightSynchronize fsdrightReadControl fsdrightExecute fsdrightReserved1 fsdrightReadAttributes fsdrightListContents	Ignored

The mapping process shown above results in a folder ACE for the group FolderUsers that grants MAPI permissions.

To reverse the conversion, Exchange starts by duplicating the ACE to again produce a folder ACE and a message ACE as shown in Table 5.8.

Table 5.8 Duplicate ACEs

Folder ACE	Message ACE
frightsOwner	frightsOwner
frightsVisible	frightsVisible
frightsEditAny	frightsEditAny
frightsEditOwned	frightsEditOwned
frightsDeleteAny	frightsDeleteAny
frightsDeleteOwned	frightsDeleteOwned
frightsCreate	frightsCreate
frightsCreateSubfolder	frightsCreateSubfolder
frightsReadAny	frightsReadAny

Next, Exchange converts the permissions, as shown in Tables 5.9 and 5.10.

Table 5.9 Converting the folder ACE

MAPI permissions	Windows 2000 permissions
frightsOwner	fsdrightOwner fsdrightWriteProperty fsdrightWriteSD fsdrightDelete fsdrightWriteOwner fsdrightWriteAttributes fsdrightViewItem

MAPI permissions	Windows 2000 permissions
frightsVisible	fsdrightViewItem fsdrightReadControl fsdrightReadAttributes fsdrightExecute fsdrightReadProperty fsdrightSynchronize
frightsEditAny	fsdrightReadControl fsdrightWriteBody (ignored on folders) fsdrightWriteAttributes fsdrightWriteProperty fsdrightAppendMsg (ignored on folders) fsdrightCreateItem fsdrightDelete fsdrightCreateContainer fsdrightOwner fsdrightSynchronize fsdrightWriteSD fsdrightWriteOwner
frightsEditOwned	fsdrightWriteOwnProperty (ignored on folders)
frightsDeleteAny	fsdrightDelete
frightsDeleteOwned	fsdrightDeleteOwnItem (ignored on folders)
frightsCreate	fsdrightCreateItem
frightsCreateSubfolder	fsdrightCreateContainer

MAPI permissions	Windows 2000 permissions
frightsReadAny	fsdrightReadControl fsdrightReadBody (ignored on folders) fsdrightListContents fsdrightReadAttributes fsdrightReadProperty fsdrightViewItem fsdrightSynchronize fsdrightExecute

Table 5.10 Converting the message ACE

MAPI permissions	Windows 2000 permissions
frightsOwner	fsdrightOwner (ignored on messages) fsdrightWriteProperty fsdrightWriteSD fsdrightDelete fsdrightWriteOwner fsdrightWriteAttributes fsdrightViewItem
frightsVisible	fsdrightViewItem fsdrightReadControl fsdrightReadAttributes fsdrightExecute fsdrightReadProperty fsdrightSynchronize

MAPI permissions	Windows 2000 permissions
frightsEditAny	fsdrightReadControl fsdrightWriteBody fsdrightWriteAttributes fsdrightWriteProperty fsdrightAppendMsg fsdrightCreateItem (ignored on messages) fsdrightDelete fsdrightCreateContainer (ignored on messages) fsdrightOwner (ignored on messages) fsdrightsynchronize fsdrightWriteSD fsdrightWriteOwner
frightsEditOwned	fsdrightWriteOwnProperty
frightsDeleteAny	fsdrightDelete
frightsDeleteOwned	fsdrightDeleteOwnItem
frightsCreate	fsdrightCreateItem (ignored on messages)
frightsCreateSubfolder	fsdrightCreateContainer (ignored on messages)
frightsReadAny	fsdrightReadControl fsdrightReadBody fsdrightListContents (ignored on messages) fsdrightReadAttributes fsdrightReadProperty fsdrightViewItem fsdrightSynchronize fsdrightExecute

The preceding conversions produce the two Grant ACEs, as shown in Table 5.11.

Table 5.11 Permissions specified by the FolderUsers group's two Grant ACEs

Permissions in the folder ACE	Permissions in the message ACE
fsdrightOwner	fsdrightWriteProperty
fsdrightWriteProperty	fsdrightWriteSD
fsdrightWriteSD	fsdrightDelete
fsdrightDelete	fsdrightWriteOwner
fsdrightWriteOwner	fsdrightWriteAttributes
fsdrightWriteAttributes	fsdrightViewItem
fsdrightViewItem	fsdrightReadControl
fsdrightReadControl	fsdrightReadAttributes
fsdrightReadAttributes	fsdrightExecute
fsdrightExecute	fsdrightReadProperty
fsdrightReadProperty	fsdrightSynchronize
fsdrightSynchronize	fsdrightWriteBody
fsdrightCreateItem	fsdrightAppendMsg
fsdrightCreateContainer	fsdrightWriteOwnProperty
fsdrightListContents	fsdrightDeleteOwnItem
fsdrightReserved1	fsdrightReadBody

The folder ACE and the message ACE effectively grant all of the available permissions to FolderUsers, so that the two corresponding Deny ACEs are not needed. Basically, these are the same two ACEs that were listed in Table 5.6, before any conversions took place.

Default Roles Available in Exchange

Microsoft® Exchange provides a set of default levels of access that you can use when you are setting permissions on objects in the Exchange store. These levels of access are called "roles." However, they are not related to the roles that you can configure programmatically in the Exchange store.

Table 6.1 lists the default roles that are available when you are working with application public folder hierarchies and the permissions that each role has. These permissions are listed in "Permissions Used in the Windows NT Security Descriptor" in Chapter 4.

Table 6.1 Default roles used for folders in application public folder hierarchies

Role	Permissions
Owner	Create Read Modify Delete all items and files Create subfolders Change Permissions (All rights in the folder)
Publishing Editor	Create Read Modify Delete all items and files Create subfolders

Role	Permissions
Editor	Create Read Modify Delete all items and files
Publishing Author	Create Read items and files Modify Delete own items and files
Author	Create Read items and files Modify Delete own items and files
Nonediting Author	Create Read items and files Delete own items and files
Reviewer	Read items and files
Contributor	Create items and files (No read/list permissions)
None	(No permissions)

Table 6.2 lists the default roles that are available when you are working with mailboxes, folders in the default Public Folders hierarchy, and their contents. The permissions listed for each role are MAPI permissions, not Windows 2000 permissions.

Table 6.2 Roles used for mailbox folders and for folders in the default Public Folders hierarchy

Role	Permissions
Owner	frightsOwner frightsCreateSubfolder frightsReadAny frightsCreate frightsEditAny frightsDeleteAny frightsEditOwned frightsDeleteOwned frightsContact frightsVisible
Publishing Editor	frightsReadAny frightsCreate frightsEditAny frightsDeleteAny frightsEditOwned frightsDeleteOwned frightsCreateSubfolder frightsVisible
Editor	frightsReadAny frightsCreate frightsEditAny frightsDeleteAny frightsEditOwned frightsDeleteOwned frightsVisible

Role	Permissions
Publishing Author	frightsReadAny frightsCreate frightsEditOwned frightsDeleteOwned frightsCreateSubfolder frightsVisible
Non-Publishing Author (Nonediting Author?)	frightsReadAny frightsCreate frightsDeleteOwned frightsVisible
Reviewer	frightsReadAny frightsVisible
Contributor	frightsCreate frightsVisible
None	RightsNone frightsVisible
Custom?	0x00000000
Complete?	rightsAll frightsContact

Appendixes



Minimum Permissions Required for Mailbox Stores and Public Folder Stores

If you modify the default permissions on Microsoft® Exchange Server 2003 mailbox stores and public folder stores, make sure you maintain the following minimum permissions:

- **Administrators group** Full Control
- **Authenticated Users group** Read and Execute, List Folder Contents, and Read
- **Creator Owner** None
- **Server Operators group** Modify, Read and Execute, List Folder Contents, Read, and Write
- **System account** Full Control

You may experience difficulties in mounting the mailbox stores or public folder stores if you do not maintain these permissions for these groups and accounts. The following error messages and events indicate this type of problem:

- An internal processing error has occurred. Try restarting the Exchange System Manager or the Microsoft Exchange Information Store service, or both.
- MAPI or an unspecified service provider. ID no: 00000476-0000-00000000.
- Information Store (2520) An attempt to determine the minimum I/O block size for the volume "[drive:]" containing "[drive:]Exchsrvr\Mdbdata\" failed with system error 5 (0x00000005): "Access is denied." The operation will fail with error -1032 (0xffffbf8).
- Error 0xffffbf8 starting Storage Group [*dn of storage group*] on the Microsoft Exchange Information Store.
- The MAPI call 'OpenMsgStore' failed with the following error: The Microsoft Exchange Server computer is not available. Either there are network problems or the Microsoft Exchange Server computer is down for maintenance. The MAPI provider failed. Microsoft Exchange Server Information Store ID no: 8004011d-0526-00000000.

You may also encounter problems when mounting public folder stores if you have turned off the **Allow inheritable permissions from parent to propagate to this object** option for the public folder hierarchy. The following error messages indicate this type of problem:

- The store could not be mounted because the Active Directory information was not replicated yet.
- The Microsoft Exchange Information Store service could not find the specified object. ID no: c1041722

To resolve this issue, add the required permissions again.

To restore permissions inheritance

1. Right-click the public folder MAPI tree, and then click **Properties**.
2. In the **Properties** dialog box, click the **Security** tab, and then select the **Allow inheritable permissions from parent to propagate to this object** check box.

After you complete this procedure, wait for the Active Directory® directory service to replicate the change to all of the domain controllers. After the change has been replicated, you can remount the store.

A P P E N D I X B

Additional Resources

The following resources provide valuable information about Exchange Server 2003 store permissions:

- Chapter 12, "Access Control," in the *Distributed Systems Guide* volume of the *Microsoft Windows 2000 Server Resource Kit* (<http://www.microsoft.com/windows2000>)
- Topics in the *Microsoft Exchange Software Development Kit (SDK)* (for either Exchange 2000 Server or Exchange Server 2003) (<http://msdn.microsoft.com/exchange>):
 - "Security" in "Key Tasks"
 - "Using the Administrative Virtual Root" in "Exchange Store URLs"
 - "Exchange Management Security" in "CDO for Exchange Management"
 - "http://schemas.microsoft.com/mapi/ Namespace" in "Properties by Namespace"
 - "http://schemas.microsoft.com/exchange/security/ Namespace" in "Properties by Namespace"
 - "Exchange 5.5 Access Rights and the Exchange Store" in "Exchange Store XML Security Descriptor Format"
 - "Application Security Module Reference" in "Reference"
- Exchange Server 2003 documentation at <http://www.microsoft.com/exchange/library>
- "Windows 2000" and "Active Directory" sections in the *Microsoft Platform SDK* at <http://www.msdn.microsoft.com>.
- Microsoft Knowledge Base articles:
 - 270905, "XADM: Unable to Set Client Permissions on Public Folders Through Exchange System Manager" (<http://support.microsoft.com/?kbid=270905>)
 - 316047, "XADM: Addressing Problems That Are Created When You Enable ADC-Generated Accounts" (<http://support.microsoft.com/?kbid=316047>)

- *Microsoft Exchange 2000 Server Resource Kit*
- Technical paper *Public Folder Permissions in a Mixed Mode Microsoft Exchange Organization*
(<http://go.microsoft.com/fwlink/?LinkId=10228>)

For more information, go to: <http://www.microsoft.com/exchange>.

Did this book help you? Please give us your feedback. On a scale of 1 (poor) to 5 (excellent), how would you rate this book?