Public Folders

To me, public folders are the cockroaches of Exchange. They persist despite seemingly being on the edge of extinction several times. Microsoft originally launched public folders as the cornerstone of Exchange's collaborative capabilities, something that could take on Lotus Notes' ability to generate end-user applications based on electronic forms. Public folders offer the ability to replicate data so that a copy is close (in network terms) to users, a feature that was tremendously important in the days of expensive and scarce bandwidth. Public folders were the way to share documents and other items across organizations and can be mail-enabled to allow users to send messages to folders, a feature often exploited to allow public folders to serve as the enduring repository for email-based discussion groups. Nothing remotely similar existed in Microsoft's product lineup until SharePoint 2001 appeared, and even though SharePoint has evolved dramatically since then, it still doesn't deliver quite what public folders do. Public folders are antique, undervalued, and creaking, but they act as a valuable applications platform for many companies which is why they are still around. Some companies have hundreds of thousands of folders in their public folder hierarchy, even if they don't quite know what all the folders are for, whether they still hold useful information, who created the folders in the first place, and what to do with them in the future.

Public folders and Exchange 2010

Despite their popularity in some quarters, once Microsoft launches into the development cycle for a new version of Exchange, the question of whether this is the version in which public folders are finally dispatched to the great byte bucket in the sky arises. To its credit, Microsoft listened to customers and started to invest a small amount of engineering resource into public folders in Exchange 2007 and has continued on this path in Exchange 2010. Public folders are only required in Exchange 2010 if you:

1. Have the requirement to support Outlook 2003 clients, which cannot use web-based distribution to access OAB and free/busy data. Outlook 2007 clients are also able to use public folders in this way but Outlook 2010 clients only use web-based distribution to access OAB and free/busy data.

2. Have applications that are based on public folders. Ten years ago, these applications were reasonably popular, but recently they have become a fast-disappearing category because companies have migrated applications to more modern and

functional platforms. SharePoint and InfoPath forms are an option that is often considered for the kind of forms-based applications that were deployed on top of public folders.

If you deploy a new Exchange 2010 organization, you'll be asked by the installation procedure whether any Outlook 2003 clients (or older Entourage clients) exist inside the organization. Outlook Express and other IMAP4 clients are also able to access public folders, but they don't depend on them for OAB and free/busy. If you need to support these clients, Exchange will create a public folder database to be used for the OAB and free/busy. If not, you'll never see a public folder database unless you decide to create one. Clients with their mailboxes on an Exchange 2010 server can only access public folders if replicas are available in a database hosted by an Exchange 2010 server. If this is not the case, clients will be told that they can't access public folders because there is no Exchange 2010 public folder server.

It would be unreasonable to expect a dramatic increase in functionality for public folders after a decade of doubt, but it is nice to see that the code is still being maintained. In summary, public folder management is divided between two consoles and EMS:

- The public folder management console (Figure 1) allows administrative access to the folder hierarchy, including the ability to create new folder replicas on different servers, remove folders, and mail-enable folders. No access is available to the content of public folders through the console, because this feature remains the sole preserve of clients. In other words, if you want to see inside a public folder, you have to use a client like Outlook or OWA that can open a folder and view its contents. Outlook is the only client that is able to report how much storage a public folder occupies. The same is true of folder permissions. You can create a new public folder with the console but then need to revert to Outlook or EMS to set permissions.

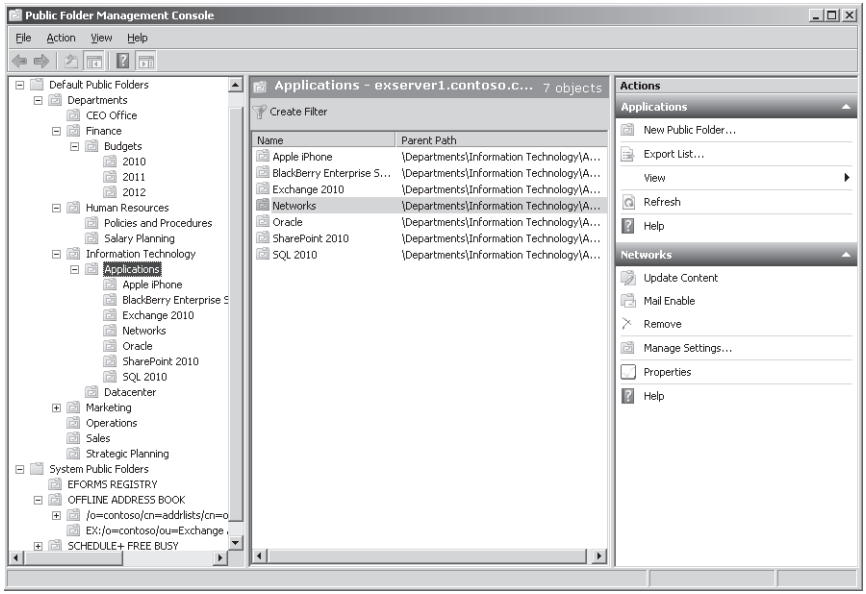

Figure 1 Public folder management console.

- The database management section of EMC allows you to create new public folder databases. Figures 2 and 3 illustrate the two screens of the New Public Folder Database Wizard. After public folder databases are created, you can update their properties such as storage limits, replication schedules, and limits for deletion retention and item age. Inside mixed-mode organizations that support both Exchange 2007 and 2010 servers, you have to perform public folder management using the Exchange 2010 ESM because of the schema changes made in Active Directory and the Store for Exchange 2010. The situation is slightly more complex when Exchange 2003 servers are present but you should still manage public folder servers using the same version of the management tools as the public folder server. Replication between the different versions of servers is sufficient to keep public folder servers synchronized across the organization.

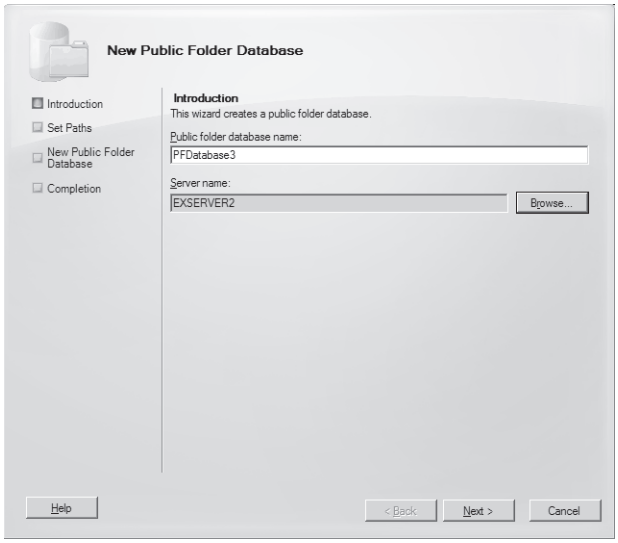

Figure 2 Creating a new public folder database with EMC.

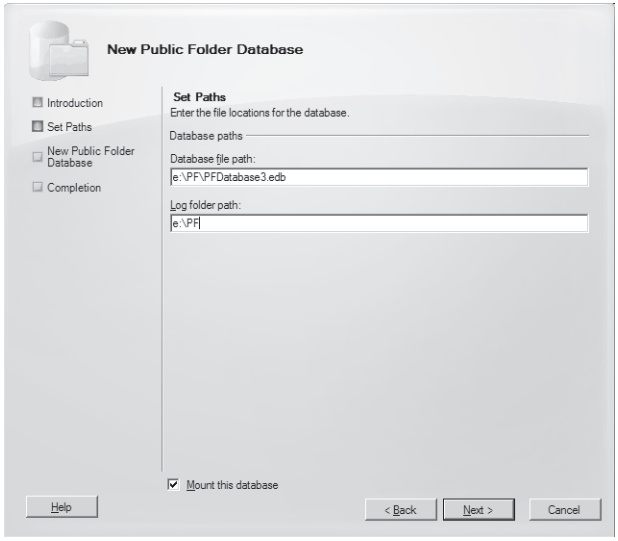

Figure 3 Specifying the locations for the public folder database files.

- Exchange 2010 SP1 includes a new Manage Public Folder Settings Wizard that you can use to set user permissions for folders and their sub-folders (Figure 4). Apart from making it much easier to manipulate public folder settings, the wizard is a good learning aid to help administrators become acquainted with the EMS cmdlets that

can be used to work with public folders, for instance, to learn the proper syntax for the command to grant a user access to a public folder.

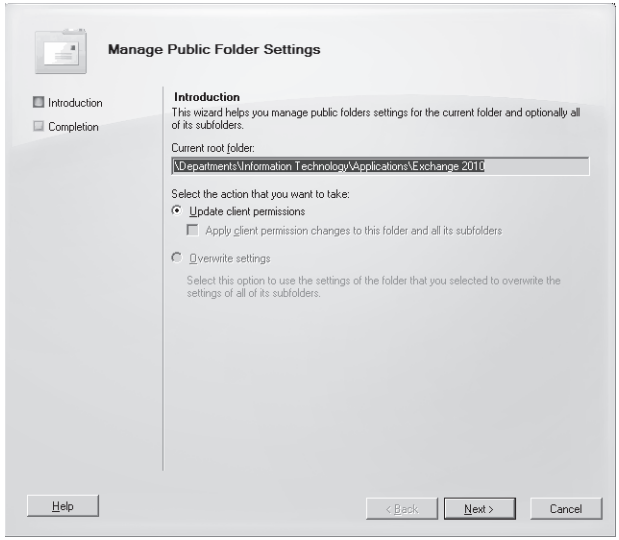

Figure 4 Manage Public Folder Settings Wizard.

- EMS offers a set of cmdlets for public folder management that underpin the features offered by EMC and the public folder management console.

An account must hold the public folder management role before it can perform any public folder management task.

Public folders continue to be organized into two sub-trees within the folder hierarchy:

- Default public folders (the IPM_Subtree) are those available for population by client applications.
- System public folders (the non_IPM_Subtree) are not revealed to client applications and are used by Exchange to store system data such as the OAB and free and busy data as well as the organizational forms used by Outlook 2003.

Exchange does not include the content of public folders in its content indexes. You have to use SharePoint if you need to index public folders for search purposes.

If you are used to working with public folders through EMS in Exchange 2007, you need to check any code (especially in scripts) that you have developed for this purpose because Microsoft has made a number of changes in the way that the cmdlets work.

Creating a new public folder database

Exchange 2010 creates a default public folder database if you tell the installation procedure that you need to support Outlook 2003 clients. Because DAGs don't support public folder databases, you need to create a second database on another server if you want to replicate public folders to ensure some level of high availability. You can create a new public folder database with the New-PublicFolderDatabase cmdlet. This example creates a new public folder database on a specified server and then mounts the new database. Note that you have to tell Exchange where to place the database and transaction logs; this is a difference from Exchange 2007, in which you always create a public folder database in a nominated storage group so Exchange can determine the location of the files from the default location of the storage group:

```
New-PublicFolderDatabase 'PubFolders1' –Server 'ExServer1' -EdbFilePath
'D:\Exchange\PF\PFDatabase.edb'
–LogFolderPath 'D:\Exchange\PF\'
–CircularLoggingEnabled $True
Mount-Database 'PubFolders1'
```

You can use the Get-PublicFolderDatabase cmdlet to validate that the new database exists:

```
Get-PublicFolderDatabase –Identity 'PubFolders1' | Format-List
```

The same cmdlet is used without a parameter to see all the public folder databases that exist within the organization. By default, Exchange does not return the names of public folder databases that reside on Exchange 2003 or 2007 servers, so if you want to see this information, you have to specify the *–IncludePreExchange2010* parameter as shown below. The *–Status* parameter instructs Exchange to return information about the mount and last backup status.

```
Get-PublicFolderDatabase –IncludePreExchange2010 –Status
```

Retrieving information about all of the public folder databases in an organization can take some time to complete because of the need to access information from the different servers that host the databases. You can restrict retrieval to a specific server by passing the server name in the *–Server* parameter.

```
Get-PublicFolderDatabase –Server 'ExServer1' –Status
```

Adding new public folders

Now that we have a public folder database, we can populate it with folders using the New-PublicFolder cmdlet. To create a new top-level folder, we use "\" (the root) as the location. For example:

```
New-PublicFolder –Name 'Technical Information' –Path '\' -Server 'ExServer1'
```

We can then populate the sub-folders under the top-level folder that we have just created. This example creates a new public folder called Exchange 2010 under the Technical Information top-level public folder and hosts the initial replica on the server called ExServer1.

```
New-PublicFolder –Name 'Exchange 2010' –Path '\Technical Information' –Server
'ExServer1'
```

If you run this command on a mailbox server that has a public folder database, you can omit the server name and Exchange will create the folder in the local database. The root folder (named Technical Information in this case) must exist on the target server. To create the folder at a deeper location in the hierarchy, you pass the complete path to the folder. For example:

```
New-PublicFolder –Name 'Administration' –Path '\Technical Information\Exchange 2010'
–Server 'ExServer1'
```

Retrieving information about public folders

After you create the public folder, you can check its properties with the Get-PublicFolder cmdlet. As you can see, the complete path has to be stated to allow Exchange to locate the folder:

```
Get-PublicFolder –Identity '\Technical Information\Exchange 2010\Administration'
```

A complete list of public folders under the IPM_Subtree is generated by passing the root folder as the path and including the *–Recurse* parameter to force Exchange to traverse the hierarchy and report all folders that it finds. Recurse instructs Exchange to retrieve information about all child folders and their children. Use the *–GetChildren* parameter instead of *–Recurse* if you want to fetch details of just the child folders under the path. The two parameters are mutually exclusive. In either case, because you always work with the copy of the public folder hierarchy on the server to which you are connected, this operation can take some time to complete if the public folder hierarchy is large and might not contain any newly created folders if their details are still being replicated. As shown here, you can pipe the list to capture it in a text file.

```
Get-PublicFolder –Identity '\' –Recurse –ResultSize Unlimited >
C:\Temp\PublicFolders.txt
```

Scanning a large list of folders to find a particular one can be frustrating. You can filter what Exchange returns as follows:

```
Get-PublicFolder –Identity '\' –Recurse | Where {$_.Name –eq 'Exchange 2010'} |
Format-List
```

```
HasSubFolders                  : True
HiddenFromAddressListsEnabled  : False
IssueWarningQuota              : unlimited
LocalReplicaAgeLimit           :
MailEnabled                    : False
Name                           : Exchange 2010
ParentPath                     : \Technical Information
PerUserReadStateEnabled        : True
ProhibitPostQuota              :
Replicas                       : {PFDatabase1}
ReplicationSchedule            : {}
RetainDeletedItemsFor          :
UseDatabaseAgeDefaults         : True
UseDatabaseQuotaDefaults       : True
UseDatabaseReplicationSchedule : True
UseDatabaseRetentionDefaults   : True
HasModerator                   : False
Identity                       : \Technical Information\Exchange 2010
OriginatingServer              : ExServer1.contoso.com
```

We can see that the folder has some sub-folders and we also know the path to the folder, so we can discover the set of sub-folders by executing another Get-PublicFolder cmdlet using the folder path and the *–Recurse* parameter.

```
Get-PublicFolder –Identity '\Technical Information\Exchange 2010' –Recurse |
Select Name, Identity | Format-Table –AutoSize
```

The same technique can be used to retrieve information about system public folders such as those used to hold OAB and free/busy information.

```
Get-PublicFolder –Identity '\Non_IPM_Subtree' –Recurse | Select Name, Identity |
Format-Table –AutoSize
```

Two cmdlets are available to retrieve information about public folder contents. The intention of these cmdlets is that they can be used for basic reporting purposes so that you understand what's going on within public folders. They are not intended to provide comprehensive reports, although there is no doubt that considerable ingenuity can be exerted to create good-looking and interesting reports based on the data generated by these cmdlets.

- Get-PublicFolderItemStatistics: Provides an insight into the content that exists in a public folder by letting administrators view the item titles, size, and so on.
- Get-PublicFolderStatistics: Provides summary information about anything from a single folder to all of the public folders known in the hierarchy, including system folders. The summary is provided in alphabetical order and includes the folder's name,

the number of items in the folder, the size of the items in the folder, and the date and time of the user access to the folder.

The Get-PublicFolderItemStatistics cmdlet is new to Exchange 2010 and lists the basic details of the items in a folder. You cannot see the actual content without accessing the folder with a client. For example:

```
Get-PublicFolderItemStatistics –Identity '\Departments\Finance' |
Select Subject, CreationTime, MessageSize | Format-Table –AutoSize
```

```
Subject                              CreationTime             MessageSize
-------                              ------------             -----------
Finance Planning Sessions - FY10     11/30/2009 6:07:44 AM    9.479 KB (9,707 bytes)
Restricting expenses                 11/30/2009 6:06:45 AM    47.9 KB (49,046 bytes)
Planning for change                  11/30/2009 6:02:20 AM    1.367 MB (1,433,576 bytes)
News about finance changes           11/30/2009 5:56:07 AM    1.312 KB (1,343 bytes)
```

Moving to Get-PublicFolderStatistics, if you don't pass a folder identifier, you get a summary list of all folders in the hierarchy. If you don't specify the name of a server that hosts a public folder database, Exchange connects to the first public folder database available in the site. The *LastUserAccessTime* property is of interest in order to identify public folders that have not been accessed recently by a user because these folders become candidates for deletion if you decide to clean up the public folder structure. It's unfortunate that many public folders are created, used for a short period, and then rapidly become a repository that is accessed less and less frequently before becoming unwanted orphans that take up valuable database space. If you want to identify public folders that should be deleted, you could sort the output from the Get-PublicFolderStatistics cmdlet by the *LastUserAccessTime* property so that the folders that have not be accessed recently are at the top of the output. You could then contact the owner of each folder to ask whether the folder should be deleted or kept. The command to generate the report about public folder statistics sorted by last access time is:

```
Get-PublicFolderStatistics –Server ExServer1 | Sort-Object LastUserAccessTime |
Format-Table Name, ItemCount, LastAccessTime –AutoSize
```

```
Name                                 ItemCount        LastUserAccessTime
----                                 ---------        --------------
Add-ons                                     34        12/30/2009 3:43:31 AM
Administration                             112        12/30/2009 3:46:29 AM
Annual Budgets                             250        12/30/2009 3:58:56 AM
Audits                                      17        12/30/2009 4:01:08 AM
Capital Expenditure                         52        12/30/2009 4:00:56 AM
```

Here's what you might see when you drill down on a specific folder. This output reveals that there is more information available about a folder than shown in the summary view.

```
Get-PublicFolderStatistics -Identity '\Departments\Finance' | Format-List
```

```
ContactCount             : 1
CreationTime             : 11/30/2009 3:57:54 AM
DeletedItemCount         : 12
FolderPath               : Departments\Finance
ItemCount                : 533
LastAccessTime           : 12/30/2009 3:57:54 AM
LastModificationTime     : 12/30/2009 6:58:53 AM
LastUserModificationTime : 12/30/2009 5:22:30 AM
LastUserAccessTime       : 12/30/2009 5:23:30 AM
Name                     : Finance
OwnerCount               : 1
TotalAssociatedItemSize  : 0 B (0 bytes)
TotalDeletedItemSize     : 0 B (0 bytes)
TotalItemSize            : 1.426 MB (1,494,997 bytes)
DatabaseName             : PFDatabase1
```

We can combine the knowledge we've gained from the two examples to do something more useful, such as outputting some data about all folders into a CSV file that we can later manipulate with Excel to understand different aspects of our public folder deployment. For example, we might ask questions such as what the largest folder is, what folders have not been accessed in several months, and what folders are empty. Here's how to generate the list:

```
Get-PublicFolderStatistics -Server ExServer1 | Select Name, ItemCount, TotalItemSize,
LastUserAccessTime, LastUserModificationTime | Export-CSV
'C:\Temp\Public-Folders.CSV'
```

Updating public folder properties

Set-PublicFolder is the basic cmdlet used to update the properties of a public folder. Table 1 lists the properties that are most commonly altered.

Table 1 Common public folder property

Property	Meaning
Age Limit	Sets the age limit for all replicas of a public folder. Folders and their replicas are automatically removed when they reach the age limit. Default: Not defined, which means that folders are persistent until they are removed by an administrator.

IssueWarningQuota	Sets the limit at which Exchange issues a warning to public folder owners that the folder is almost full. Default: Unlimited (database default is used).
MaxItemSize	Sets the limit of the size of an item that a user can post to a folder. Default: Not set (database default is used).
PerUserReadStateEnabled	Tells Exchange to maintain a read state for each individual user. The read state tracks whether an item has been read by a user. Default: True.
ProhibitPostQuota	Sets the limit at which users are no longer allowed to post items to a folder. Default: Not set (database default is used).
RetainDeletedItemsFor	Sets the retention period for items deleted from a public folder. Default: Not set (database default is used—typically 14 days).

To see the default limits for a public folder database, select it through EMC, view Properties, and then click the Limits tab (Figure 5).

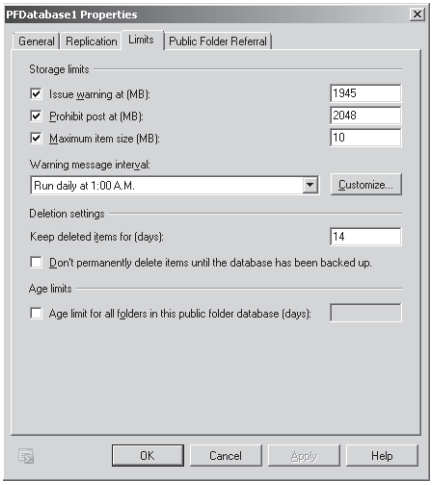

Figure 5 Public folder database limits.

A public folder that is used as a repository for electronic forms is a good instance of when you might need to override the default database storage limits to set a maximum item size that allows the forms to be posted without allowing users to post other items. This example sets a maximum item size of 25 KB and sets appropriate storage limits for a folder.

```
Set-PublicFolder –Identity '\Departments\Finance\Forms' –AgeLimit '365.00:00:00'
–UseDatabaseQuotaDefaults $False –MaxItemSize 25KB –ProhibitPostQuota 500MB
–IssueWarningQuota 465MB –RetainDeletedItemsFor '90.00:00:00'
```

Creating new replicas

The default state for a public folder is that a single replica exists in the database in which an administrator originally creates the folder unless replicas exist for the parent folder, in which case Exchange creates replicas for the new child folder in the databases where replicas exist for the parent.

Having a single replica of a folder might be the most appropriate state for the folder if all users are able to make reliable connections to the server that hosts the database. However, in large or distributed environments it is common to create several replicas so that users can always make a local connection. We will discuss how to control the process by which Exchange decides to which replica a client should connect in the "Removing public folders" section. For now, let's review how to create the replica folders.

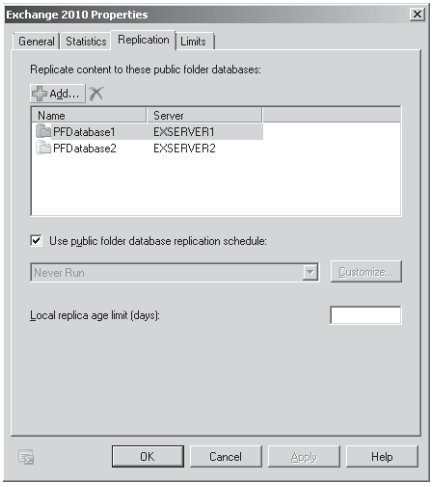

Figure 6 Configuring public folder replicas.

Open the public folder management console and locate the folder that you want to replicate. Click Properties and then select the "Replication" tab (Figure 6). You will see the list of databases that currently have replicas and will be able to add new databases. The equivalent EMS command is:

```
Set-PublicFolder -Identity '\Departments\Finance' -Replicas
'PFDatabase1','PFDatabase2'
```

Replication is performed through special messages sent between the servers (using SMTP) that host the public folder databases that include the replicas. Each database has a replication schedule that typically replicates new items or other content updates for public folders every 15 minutes. Content is broken up into relatively small (by today's standards) 300 KB

messages to ensure fast and efficient distribution. You can force immediate replication by selecting the Update Content option from the action pane. This might be required if a user wants the immediate distribution of some important information throughout the organization. If you don't already have the public folder management console fired up and ready to go, it's probably quicker to invoke the Update-PublicFolder cmdlet from EMS. In this instance, we force replication of the contents of the Press Releases folder using the folder from the database on server ExServer1:

```
Update-PublicFolder –Identity '\Departments\Marketing\Press Releases'
–Server ExServer1
```

By comparison to folder content, changes to the folder hierarchy are replicated as soon as they occur, so the addition or deletion of a folder should replicate throughout the organization very quickly.

Mail-enabling a public folder

A mail-enabled public folder is able to accept new items via email. If anonymous access is permitted, anyone will be able to create a message, address it to the folder, and send it to Exchange for delivery. A public folder is mail-enabled with the Enable-MailPublicFolder cmdlet. Alternatively, you can select the folder from the Public Folder console and click on the "Mail Enable" option in the action pane. Mail-enabled public folders are listed in the console with a different icon from folders that are not mail-enabled. The top two folders shown in Figure 7 are mail-enabled; the bottom folder is not.

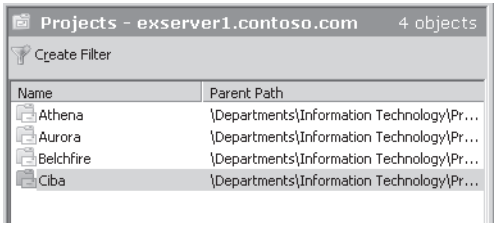

Figure 7 Mail-enabled public folders.

When you mail-enable a public folder, Exchange also reveals it to users in the GAL, albeit with minimal useful detail (Figure 8). It might not be desirable to have public folders shown in the GAL; you might prefer to have users post to the folder by addressing messages to the folder's SMTP address. These commands mail-enable a public folder and hide it from the GAL:

```
Enable-MailPublicFolder –Identity '\Departments\Finance\Forms'
Set-PublicFolder –Identity '\Departments\Finance\Forms'
–HiddenFromAddressListEnabled $True
```

You can also hide a public folder from Exchange address lists by selecting it in the Public Folder management console, view its properties, navigate to the Exchange General property page, and select the checkbox to hide the folder. Hiding mail-enabled public folders only conceals them from users. If you hide a mail-enabled public folder from the Exchange address lists, administrators will continue to see these objects listed in the picker dialogs used to select recipients for various tasks such as adding recipients to distribution groups.

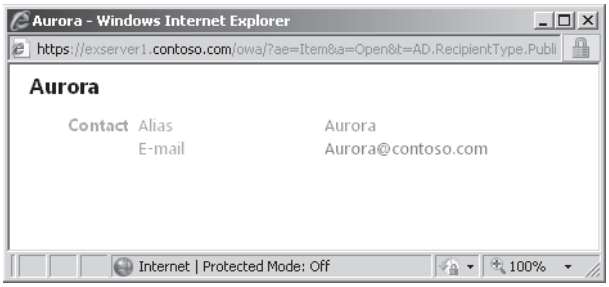

Figure 8 How a mail-enabled public folder appears in the GAL.

Behind the scenes, when you mail-enable a public folder, Exchange creates a new object for the folder in the Microsoft Exchange System Objects OU (Figure 9). The new object is used to hold properties of the public folder such as the proxy email addresses, mail tips, and so on. The properties of the new object can then be viewed with the Get-MailPublicFolder cmdlet and manipulated with the Set-MailPublicFolder cmdlet.

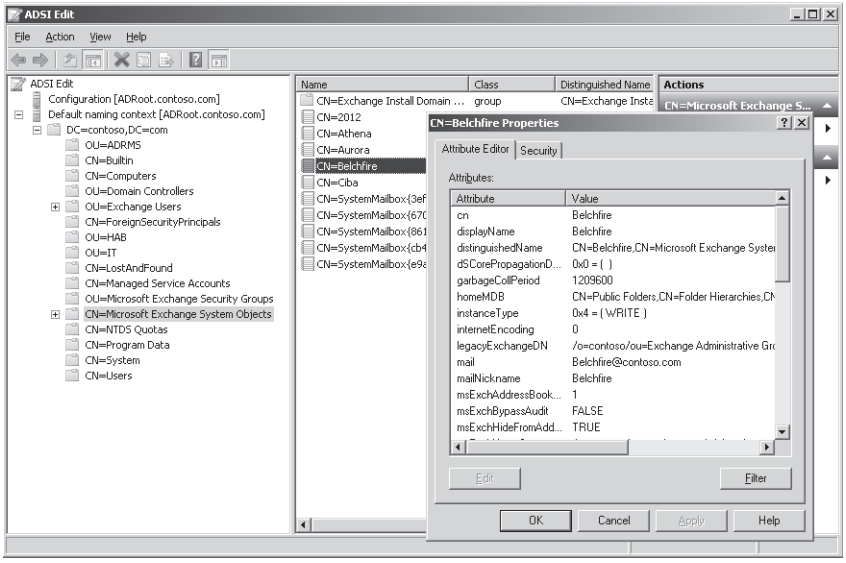

Figure 9 Mail-enabled public folders in Active Directory.

The mail-related properties that we discussed when we reviewed how to manage mailboxes, groups, and contacts are also available for public folders. For example, here's how to set up moderation for a public folder so that only users from a specific group plus the moderator can post items. Any other attempt has to go through moderation.

```
Set-MailPublicFolder –Identity '\Departments\Finance' –ModeratedBy 'Redmond, Tony'
–MailTip 'Only members of the Finance Department can post to this folder'
–ByPassModerationFromSendersOrMembers 'FinanceUsers' , 'Redmond, Tony'
–ModerationEnabled $True
```

You can disable a mail-enabled public folder with the Disable-MailPublicFolder cmdlet. This will also remove the Active Directory object and the GAL.

```
Disable-MailPublicFolder –Identity '\Departments\Finance\Forms'-Confirm:$False
```

Bad email addresses for email-enabled public folders

No system folder needs to be mail-enabled. However, you might see errors flagged when you apply an email policy to these objects. The likely cause is that these folders were enabled for email in a previous version of Exchange and their names or aliases include spaces or special characters that cannot be used in SMTP addresses, and so are found to be invalid when Exchange attempts to use them to create email addresses according to the email policy. If you see an error like this, you can disable email for the folders and remove their email addresses with this code:

```
Get-PublicFolder '\non_ipm_subtree' –Recurse –ResultSize Unlimited |
Disable-MailPublicFolder
–ErroractionSilentlyContinue
```

Afterward, you can fix the names of the folders to remove any offending characters from their names or aliases and then re-enable them for mail.

Removing public folders

The Remove-PublicFolder cmdlet deletes a public folder from the public folder hierarchy and removes all content and replicas that exist in public folder databases in the organization. The simplest deletion is to remove just one folder. For example:

```
Remove-PublicFolder –Identity '\Departments\Finance\Tools' –Confirm:$False
```

This command works if the folder has no children. If a child folder exists you'll be told that you can't delete a parent folder until you first remove any child folders. You can do this with much the same command by adding the *–Recurse* parameter to force Exchange to locate and remove any child folders before it removes the parent.

```
Remove-PublicFolder –Identity '\Departments\Finance\Tools' –Recurse –Confirm:$False
```

In these examples I've suppressed the confirmation prompt because I am pretty confident about the folder that I want to delete. A delete of a public folder cannot be reversed because there's no equivalent of a deleted items folder for public folders or the items that they contain—unless you use ExFolders to recover the deleted folder. Even then, ExFolders might not be able to recover a deleted public folder, so it's a good idea to be sure that you really want to remove a folder before you proceed.

It's a good idea to clean up old and obsolete public folders on a regular basis. You can scan for folders that don't hold a lot of content or haven't been accessed recently. This command fetches the set of folders in the hierarchy as known in the public folder database on a server and generates statistics for each folder before sorting the set by the last modification time and then outputting details into a report.

```
Get-PublicFolder –Identity '\' -Server ExServer1 -Recurse
-ResultSize Unlimited | Get-PublicFolderStatistics | Select Name, ItemCount,
TotalItemSize, LastModificationTime| Sort-Object LastModificationTime
–Descending | Format-Table –AutoSize> C:\Temp\PFReport.txt
```

Controlling public folder referrals

Any public folder can be replicated to databases on multiple servers. The public folder hierarchy is always replicated to every public folder database so that each server can present a complete view of every public folder that is available in the organization to clients that request this data. The folder hierarchy lists every folder, its permissions, and details of the servers that currently host databases that contain replicas of each folder. Replicas of every folder do not necessarily exist in every public folder database. When public folders exist in an organization, every mailbox database is associated with a public folder database and clients always attempt to connect to folder replicas in the public folder database associated with their mailbox database. If clients need access to a folder replica that is not present in the database to which they are currently connected, Exchange creates a referral to the most appropriate public folder database that contains a replica. You can view the public folder database that is used by a mailbox database by looking at its properties through the Client Settings tab. As you can see in Figure 10, the IT Department database currently uses PFDatabase1, so any connections from mailboxes hosted by this database will go to PFDatabase1. If this isn't the most appropriate public folder database to use, we can select another one—perhaps a public folder database that is hosted by the same server as the mailbox database.

When a mailbox database is created, Exchange checks the available public folder databases to decide which to associate with the new mailbox database. If a public folder database is available on the same server, it is selected. If not, the first public folder (listed alphabetically) in the organization is used. Over time, this leads to a certain imbalance in the work-

load going to the first public folder database. To check the current allocation of mailbox databases to the available public folder databases:

```
Get-MailboxDatabase | Select Name, PublicFolderDatabase
```

You can then move work by updating a mailbox database as follows:

```
Set-MailboxDatabase -Identity 'IT Department'-PublicFolderDatabase 'PFDatabase2'
```

The default mechanism used to determine the best referral is based on Active Directory site link costs. First, Exchange looks for servers in the local site to see if there are public folder databases available that contain the required folder. If no local servers are available, Exchange builds a list of available servers that host public folder databases in other sites and orders them by cost. Exchange then attempts to contact each server to see whether it has an available replica and will connect to the first replica folder that it finds.

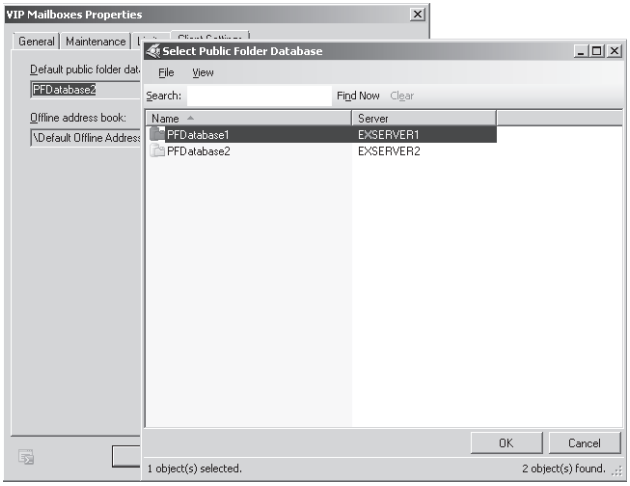

Figure 10 Selecting a public folder database for a mailbox database.

It is possible to ignore the Active Directory site link costs and use a custom referral list instead. To do this, go to the Organization Configuration node of EMC and then Mailbox, and then select the public folder database with which you want to work. Now click Properties To and then select the Public Folder Referral tab. As expected, the default option is to use Active Directory site costs. However, as shown in Figure 11, you can create a custom list of mailbox servers (that host public folder databases) and assign each a cost from 1 to 100 to have Exchange use this list when it determines client referrals. Costs are prioritized from lower to higher, so you should assign a cost of 1 (one) to the server to which you want to direct client referrals whenever possible and any other value up to 100 for the other servers. The higher the cost value, the less likely it is that Exchange will use it for client referrals.

You can create a custom referral list with EMS too. In the example shown below, we create a list of three mailbox servers that host public folder databases and assign each server a different cost to provide Exchange with an order to use to check the servers for public folder replicas. Finally, we tell Exchange to use the custom server list rather than using the default Active Directory site cost.

```
Set-PublicFolderDatabase -Identity 'ExServer1\PubFolders1'
-CustomReferralServerList 'ExServer2:1','ExServer3:10','ExServer5:25'
-UseCustomReferralServerList $True
```

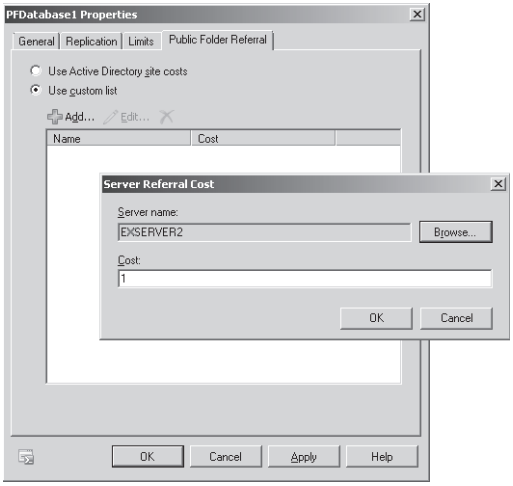

Figure 11 Defining a public folder referral server.

Permissions

Public folder permissions are divided into client permissions, which allow clients to access the public folder and work with its contents, and administrative permissions, which manage the users who have administrative rights over the folder. Public folder permissions can only be manipulated through EMS or Outlook.

Client permissions

When you create a new public folder, a default set of client permissions is applied. You can see these permissions with the Get-PublicFolderClientPermission cmdlet:

```
Get-PublicFolderClientPermission –Identity '\Departments\Finance\Annual Budgets'
```

```
Identity     : \Departments\finance\annual budgets
User         : Default
AccessRights : {Author}
Identity     : \Departments\finance\annual budgets
User         : contoso.com/Exchange Users/Exchange Administrator
AccessRights : {Owner}
Identity     : \Departments\finance\annual budgets
User         : Anonymous
AccessRights : {CreateItems}
```

The first permission means that Exchange has assigned all users the Author permission. This means that all users can post new items to the public folder. Users have read access too, so they can copy items from the folder, including from the folder into a folder in their mailbox. They can also move items that they create from one public folder to another, or back into their mailbox. Authors are able to edit items that they create, but only with clients that support this capability. OWA doesn't, but Outlook does.

The second permission tells us that the user called "Exchange Administrator" is the owner of the folder. The user that creates a folder automatically assumes this right and has total control over the folder, including the ability to delete it if so desired.

The third permission allows anonymous users to create items in the folder. You have to set this permission on a public folder to allow non-authenticated users to email items to the folder. If you don't expect this to happen (and haven't mail-enabled the folder), you can remove the permission with the Remove-PublicFolderClientPermission cmdlet. For example:

```
Remove-PublicFolderClientPermission –Identity '\Departments\Finance\Annual Budgets'
–User Anonymous
–AccessRights 'CreateItems'-Server ExServer1
```

Client permissions are broken down into roles and individual client access rights. A role spans one or more client access rights and is therefore a convenient way to assign multiple client access rights in one operation. In the examples that we have discussed to this point, CreateItems is a client access right that explicitly allows a client to create items in a public folder. On the other hand, Owner and Author are roles that both include the CreateItems client access right. Table 2 lists all of the available client access rights and the roles to which they are assigned.

Table 2 **Public folder client access**

Role	Create-Items	ReadItems	CreateSub-Folders	Folder-Owner	Folder-Contact	Folder-Visible	EditOwn-Items	EditAll-Items	Delete-Own-Items	Delete-AllItems
Owner	X	X	X	X	X	X	X	X	X	X
PublishingEditor	X	X	X			X	X	X	X	X
Editor	X	X				X	X	X	X	X
PublishingAuthor	X	X	X			X	X		X	X
Author	X	X				X	X		X	
Non-EditingAuthor	X	X				X				
Reviewer		X				X				
Contributor	X					X				
None						X				

Public folder client permissions are updated with the Add-PublicFolderClientPermission cmdlet. For example:

```
Add-PublicFolderClientPermission –Identity '\Departments\Finance\Annual Budgets'
–User Redmond
–AccessRights 'Owner' –Server ExServer1
```

We reverse the process to remove client permissions on a public folder by using the Remove-PublicFolderClientPermission cmdlet.

```
Remove-PublicFolderClientPermission –Identity '\Departments\Finance\Annual Budgets'
–User Redmond
–AccessRights 'Owner' –Confirm:$False
```

While you can cheerfully amend client permissions on a public folder, matters are often not straightforward because public folders operate in a hierarchy where child folders inherit the permissions of their parents. You might amend the permission on one folder without making the same change on another and expose information to unwanted access. The code to amend client permissions on a folder and be sure that all of the permissions are respected by child folders is complicated. For this reason, Microsoft provides two scripts with Exchange to allow you to replace a user with another user for a folder and to replace a user's permission with another set of permissions on a folder. Both scripts perform recursive updates to ensure that the same changes are applied to child folders. The scripts are:

- ReplaceUserWithUserOnPFRecursive.Ps1 (replace complete user with another)
- ReplaceUserPermissionOnPFRecursive.Ps1 (replace permissions for a user)

The Public Folder Settings Wizard provided in Exchange 2010 SP1 (Figure 12) makes the process even easier. You can opt to set permissions on a single public folder or on a folder

and all its children. In addition, you can assign permissions to multiple users at one time and select a different set of permissions to assign to each user.

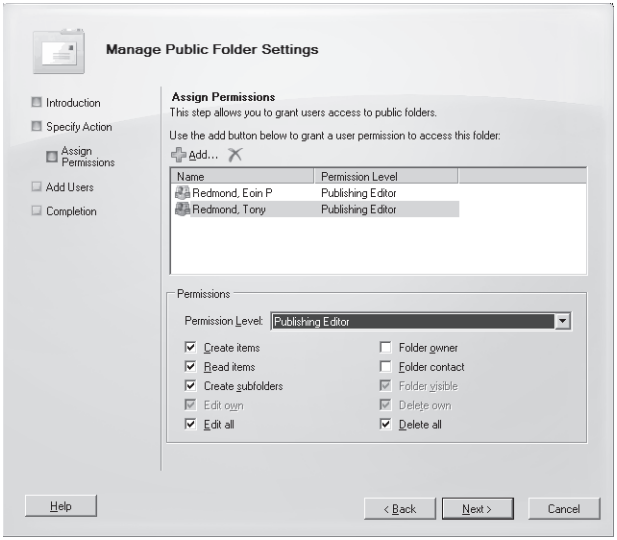

Figure 12 Assigning permissions to a public folder with the SP1 wizard.

Like mailboxes and other mail-enabled objects, you can assign permission for a user to send email on behalf of a mail-enabled public folder. This requires you to grant the Send As Active Directory right to a user. Once the right is granted, the user can select the public folder from the GAL and address a message from it as shown in Figure 13. Replies sent in response to the message are delivered to the public folder. The Add-AdPermission cmdlet is used to grant the Send-As right, as shown in this example:

```
Add-AdPermission –ExtendedRights 'Send-As' –Identity 'CN=Annual Budgets,CN=Microsoft
Exchange System Objects,dc=contoso,dc=com' –User 'Redmond, Tony'
```

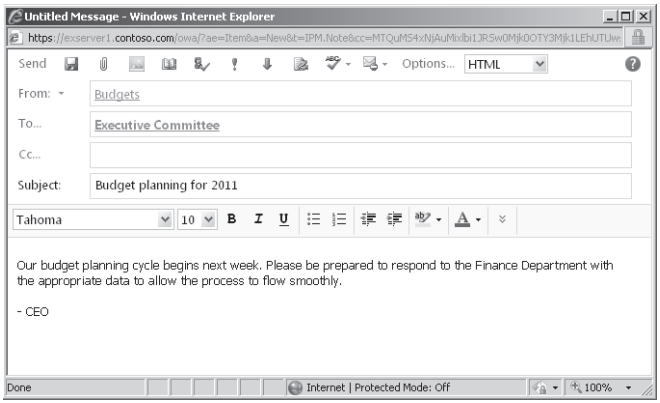

Figure 13 Sending a message on behalf of a public folder.

Administrative permissions

Public folder administrative permissions are assigned through RBAC roles or specifically to the owner of a folder. We can use the Get-PublicFolderAdministrativePermission cmdlet to view the administrative permissions on a folder. If we do this for the folder for which we've been playing with client permissions, we see:

```
Get-PublicFolderAdministrativePermission –Identity '\Departments\Finance
\Annual Budgets' |Select User, AccessRights | Format-Table -AutoSize
```

```
User                              AccessRights
----                              ------------
CONTOSO\Organization Management    {ViewInformationStore}
CONTOSO\Public Folder Management   {ViewInformationStore}
CONTOSO\Organization Management    {AdministerInformationStore}
CONTOSO\Public Folder Management   {AdministerInformationStore}
CONTOSO\Organization Management    {ModifyPublicFolderACL}
CONTOSO\Public Folder Management   {ModifyPublicFolderACL}
CONTOSO\Organization Management    {ModifyPublicFolderQuotas}
CONTOSO\Public Folder Management   {ModifyPublicFolderQuotas}
CONTOSO\Organization Management    {ModifyPublicFolderAdminACL}
CONTOSO\Public Folder Management   {ModifyPublicFolderAdminACL}
CONTOSO\Organization Management    {ModifyPublicFolderExpiry}
CONTOSO\Public Folder Management   {ModifyPublicFolderExpiry}
CONTOSO\Organization Management    {ModifyPublicFolderReplicaList}
CONTOSO\Public Folder Management   {ModifyPublicFolderReplicaList}
CONTOSO\Organization Management    {ModifyPublicFolderDeletedItemRetention}
CONTOSO\Public Folder Management   {ModifyPublicFolderDeletedItemRetention}
CONTOSO\Exchange Servers           {AllExtendedRights}
CONTOSO\E14Admin                   {AllExtendedRights}
CONTOSO\Organization Management    {AllExtendedRights}
CONTOSO\Exchange Trusted Subsystem {AllExtendedRights}
CONTOSO\Enterprise Admins          {AllExtendedRights}
CONTOSO\Domain Admins              {AllExtendedRights}
```

This data means:

- Members of the Organization Management and Public Folder Management RBAC role groups have the rights to manage the properties of the public folder and perform tasks such as modifying the client permissions list, the quota for the folder, the administrative permissions, the expiry settings for the folder and its content, adding or removing replicas from the folder replica list, and changing the deleted items retention setting for the folder. To discover the users who have these rights through their membership in a role group, type:

  ```
  Get-RoleGroupMember –Identity 'Public Folder Management'
  ```

- Specific permissions are assigned to individual accounts or to groups. These rights allow the accounts or groups to work with the folder at the specified level. AllExtendedRights is specified here, meaning that all administrative permissions are available, but you can allow or deny individual permissions as required. The folder owner is automatically included (E14Admin in this example), as are the members of the Enterprise Admins and Domain Admins group. You can also see that the Exchange Trusted Subsystem is included to allow Exchange to perform background maintenance on public folders. To check the owners assigned to a public folder, type:

  ```
  Get-PublicFolderAdministrativePermission –Identity '\Departments\Finance
  \Annual Budgets' –Owner
  ```

The easiest way to assign public folder administrative permissions is to add users to the Public Folder Management role group. For example, to add user Smith to the Public Folder Management role group:

```
Add-RoleGroupMember –Identity 'Public Folder Management' –Member Smith
```

Likewise, the easiest way to remove administrative permissions is to remove users from the Public Folder Management group.

```
Remove-RoleGroupMember –Identity 'Public Folder Management' –Member Smith
```

Membership in the public folder management role group allows a user to manage the entire public folder hierarchy. You only need to resort to assigning specific administrative permissions if you want to allow someone to manage a sub-section of the hierarchy or anything from one folder to a complete set of folders and their children. In this scenario, the Add-PublicFolderAdministrativePermission and Remove-PublicFolderAdministrativePermission cmdlets provide the way to control permissions. You have to add or remove permissions; there is no way other way to update or change existing permissions.

For example, if we wanted to allow a user to have administrative permissions over the Finance folder, we'd use a command like this:

```
Add-PublicFolderAdministrativePermission –Identity '\Departments\Finance'
–User 'Redmond' –AccessRights 'AllExtendedRights'
```

The permission is removed with:

```
Remove-PublicFolderAdministrativePermission –Identity '\Departments\Finance'
–User 'Ruth' –AccessRights 'AllExtendedRights' –Confirm:$False
```

OWA and public folders

MAPI clients like Outlook have been able to access public folders since the earliest version of Exchange and remain the most functional client in that you can set permissions for a folder, view the storage used for a folder, and so on. OWA has had a somewhat erratic history with public folders with missing features or even support for public folders completely non-existent, as happened in the RTM version of Exchange 2007. Public folders are supported by OWA 2010 and present much the same attractive interface as mailbox folders do (Figure 14). It's important to understand that before OWA 2010 can connect to a public folder, a replica of that folder must exist on an Exchange 2010 mailbox server. OWA 2010 is not able to access public folders on Exchange 2003 or 2007 servers.

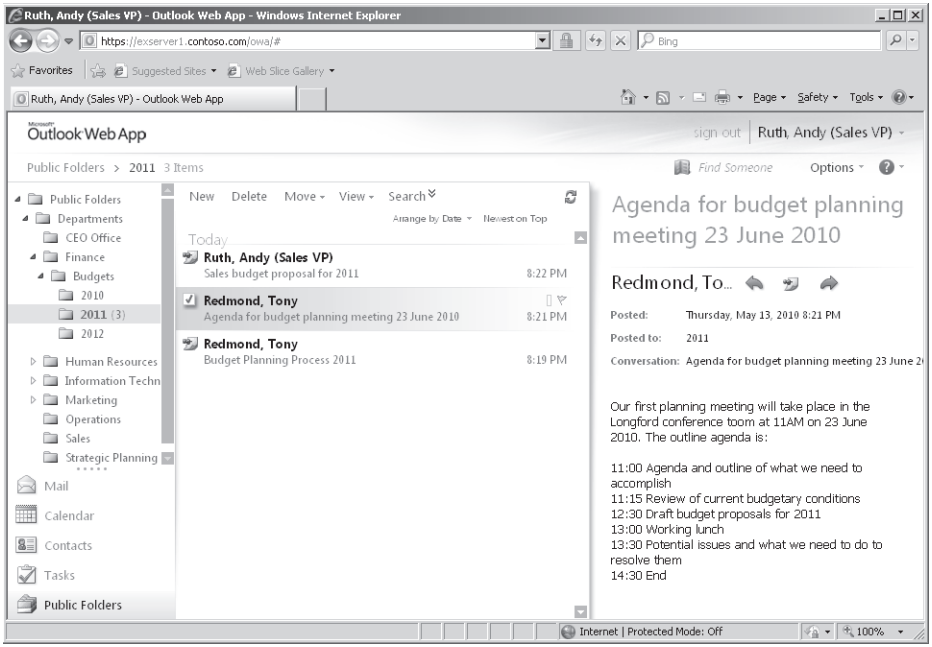

Figure 14 OWA access to public folders.

The biggest limitation is that public folders are presented in a separate window from the rest of OWA. This means that while you can post items directly into public folders, you cannot drag and drop items from mailbox folders into public folders. The workaround is to go to the mailbox folder, select the item that you want to move, right-click, and then select the Move To Folder or Copy To Folder option followed by the target public folder (Figure 15). And, of course, you can always email items to public folders, provided that they are mail-enabled.

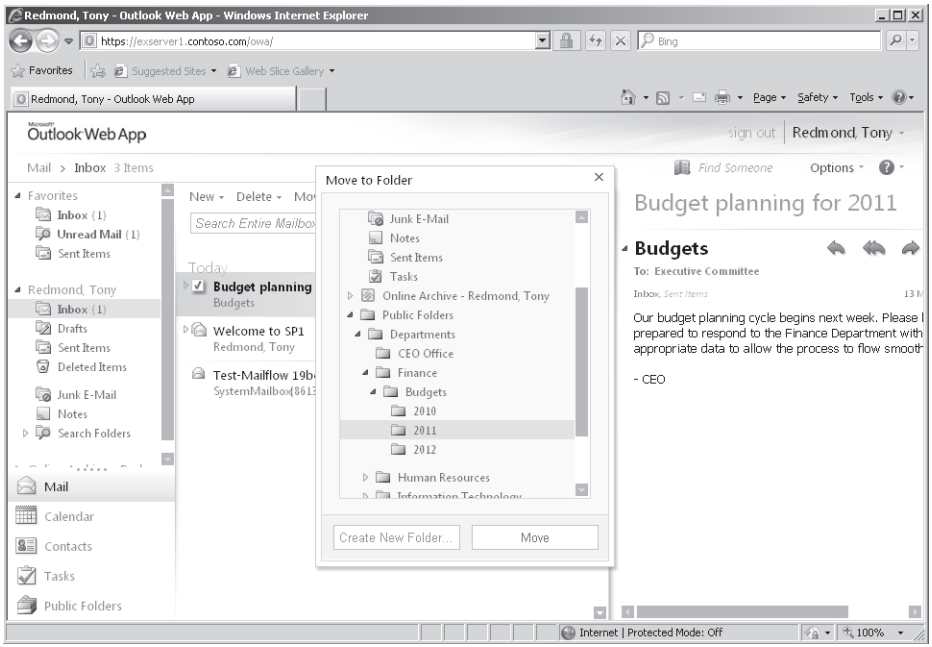

Figure 15 Using OWA to move an item into a public folder.

Apart from not being able to drag and drop items or files into a public folder, the other minor irritant that users notice most often is the delay that can exist before new content is available across the different replicas. Remember, the default replication schedule updates replicas every 15 minutes, so it is possible users will have to wait up to 15 minutes before they see an item. The solution is to decrease the replication interval so that items are copied more quickly, if available bandwidth allows. I've also noticed that new folders don't always show up quickly in the hierarchy despite immediate replication and that I have to refresh the folders to get them to show up. This isn't a big problem because not many new public folders are created today.

Public folder connections and the CAS

Incoming connections from Outlook clients to mailboxes are served by the RPC Client Access layer running on a CAS server and not by the server that hosts the mailbox. However, if an Outlook client needs to connect to a public folder, that client continues to connect directly to the nearest mailbox server that hosts a replica of the public folder. The logic in excluding public folders from the new database replication mechanism is not that the data held in public folders is any less important than mailboxes. Instead, it's because public folders have had their own multi-replica replication model since the first version of Exchange. This replication works well and there's no need to replace it at this point.

However, if you want to provide high availability for public folders, you have to configure multiple replicas and ensure that more than one replica exists in each site. Microsoft's view is probably fair because they avoided a great deal of engineering effort that might potentially have introduced some instability in Exchange 2010 by leaving public folder replication alone, but it does create an interesting split between two separate replication mechanisms in the product.

Fault-tolerant public folders

Public folders have their own form of replication that is item-based rather than using asynchronous log replication as in a DAG. Some commentators have asked why Microsoft did not change public folder replication to use log shipping. The answer varies depending on whether it is delivered by an engineering or marketing spokesperson or whether you care to read between the lines, but you can interpret Microsoft's position as follows:

- Public folders have had multi-master replication since Exchange 4.0; replication today essentially works in the same way that it worked in 1996, so, given that this mechanism has stood the test of time, why would you change it?
- Changing the replication mechanism would introduce unnecessary complications, costs, and risks into the engineering schedule for an area that is non-core for Exchange.
- The existing replication mechanism works smoothly between Exchange servers of different versions. Changing the mechanism would require some method of reconciling the differences between the "old" and the "new" versions, including the issue of dealing with multiple changes applied to a single item from different public folder servers.

The replication mechanism for public folders is unlikely to change anytime soon. Bearing this point in mind, you can incorporate advice about deploying public folders in a fault-tolerant manner that has been used since 1996 as follows:

- Whenever possible, deploy separate servers for mailbox and public folders.
- Deploy at least two public folder servers per site and replicate important folders to all servers to ensure that clients can access a local replica even when one server is unavailable.
- Analyze public folder usage on a regular basis and do not allow folders to proliferate without good reason; eliminate unused folders as soon as you can to restrict replication traffic and remove network load. Many other platforms (such as SharePoint) provide superior features for requirements such as shared document management, so always consider the alternatives before you create a new public folder.

The ExFolders utility

Over the years, Microsoft has released a number of tools to help administrators manage public folders. The tools were originally developed and used by Microsoft Support to help them debug problems with public folder permissions, replication, and so on. PFAdmin for Exchange 2003 was followed by PFDAVAdmin for Exchange 2007. However, the deprecation of WebDAV means that the PFDAVAdmin tool does not work with Exchange 2010 and has been replaced with the ExFolders tool for Exchange 2010. ExFolders is essentially a port of PFDAVAdmin, updated and improved to deal with the Exchange 2010 database structures.

ExFolders doesn't come with a very sophisticated installation procedure. You download it from the Microsoft website and place the executable in the Exchange binaries directory and launch the program. You then connect to either a mailbox or a public folder database and a domain controller. The domain controller provides access to the Exchange configuration data. After that, it's a matter of navigating to the point in the folder or mailbox hierarchy in which you're interested. ExFolders does a good job of revealing information in a more user-friendly and graphical manner than EMS. Figure 16 shows how to list permissions on a public folder. It's the same data as you see with the Get-PublicFolderClientPermission cmdlet, but it's a lot easier to view and understand.

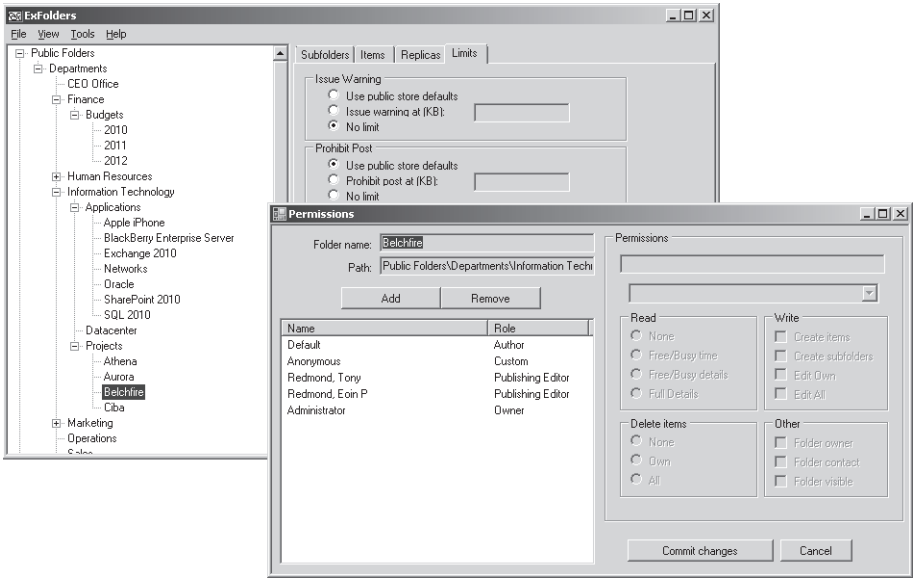

Figure 16 Viewing public folder permissions with ExFolders.

Unlike Exchange's Public Folder management console, ExFolders is able to list some information about items that are in a public folder (Figure 17). Even better, ExFolders is able to

view deleted folders and restore them. Sometimes the restore isn't possible, but when it is, it's a real life-line to rescue the embarassment caused by a mistaken folder deletion.

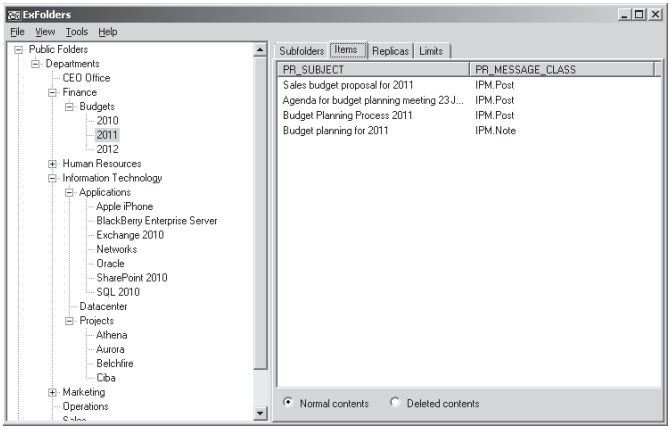

Figure 17 Listing items in a public folder with ExFolders.

In addition to public folder databases, ExFolders can open and explore mailbox databases. Many interesting items can be discovered in mailboxes, such as the structure of the mailbox and the many hidden items that are stored in the mailbox root to be used for Exchange internal processing. Figure 18 is a good example. You can see the two move reports that Exchange holds for a mailbox. You can also see the dumpster folder structure (under Recoverable Items), the folders used for Reminders and Views, and the client-visible folder structure under Top Of Information Store.

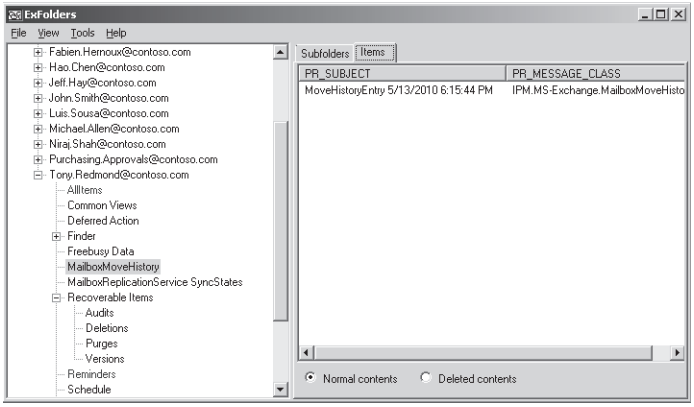

Figure 18 Mailbox data viewed by ExFolders.

Even if you're not interested in public folders, ExFolders is a very valuable tool for an Exchange administrator. It doesn't come with formal support and it doesn't always work so well, but these small deficiencies are unimportant in the context of the value that the program delivers.

Scripts provided for public folders

The Exchange development group includes a set of scripts in the installation kit that they have developed to help with public folder management. When you install a mailbox server, the scripts are placed into the \Scripts directory under the Exchange \V14 root. The scripts demonstrate that some public folder management tasks are complex, especially those that deal with deep folder hierarchies. All of the work done by the scripts is accomplished by the cmdlets that we have discussed in this section and, while they might not solve all the problems that you encounter, the scripts can be customized to extend their functionality or to address some important but esoteric detail of your deployment.

Table 3 offers a brief description of the scripts provided with Exchange and their use. See the Exchange help file for further information on the input parameters or simply browse through the script code to see how it works. The Exchange scripts directory should be in the path that Windows PowerShell uses to locate scripts, so you should be able to type in the name of the script to execute it. If not, position yourself in the directory where the scripts are located and prefix the name of the script that you want to execute with '.\' or provide the full path to the script. For example:

```
C:> .\AggregatePFData.ps1
```

Or

```
C:> 'Program Files\Microsoft\Exchange Server\V14\Scripts\AggregatePFData.ps1'
```

Table 3 Public folder administration

Script	Use
AddReplicaToPFRecursive.ps1	Adds a new server to the replica list for a folder and all its child folders. A public folder database must be present on the target server.
AggregatePFData.ps1	Aggregates public folder data for all replicas within an organization to provide a view of the amount of data held in the folders, the folder owners, and the last access time. The output for this script is best directed to a text or CSV file for later analysis. The version provided with Exchange 2010 RTM extracts data from just one public folder server while the version in Exchange 2010 SP1 extracts information from all folder replicas.

RemoveReplicaFromPFRecursive.ps1	Reverses the work of AddReplicaToPFRecursive.ps1 by removing the public folder database on a server from the replica list of a folder and all its child folders.
MoveAllReplicas.ps1	Replaces the public folder database hosted by a server with another in the replication list for all public folders. This script is typically used when you want to decommission a server that hosts a public folder database and you first need to move all the replicas from that server.
ReplaceReplicaOnPFRecursive.ps1	Replaces the public folder database on a server with another in the replica list for a folder and all its child folders.
AddUsersToPFRecursive.ps1	Add a user and specified permissions for that user to a folder and all its child folders.
ReplaceUserWithUserOnPFRecursive.ps1	Removes one user and replaces that user with another in the permissions list for a folder and all its child folders.
ReplaceUserPermissionOnPFRecursive.ps1	Replaces the permissions for a specified user with new permissions for a folder and all its child folders.
RemoveUserFromPFRecursive.ps1	Removes a user from the permissions list for a folder and all its child folders.

Migrating public folder content

No one has yet developed a magical silver bullet to migrate the data and applications that companies have accumulated in public folders since 1996. Every deployment uses public folders differently and every company needs to develop its own plan.

The first challenge is to understand the inventory of public folders that exist and then identify the folders that are actually in use. You can use the AggregatePFData script, or indeed the Get-PublicFolderStatistics cmdlet, to create a report about the current set of public folders. Such a report is a good first step to understand the folders that exist and the ones that are actually in use. However, you'll still be missing some data, such as why the public folder was created, who created the folder and who manages it now, whether the folder is associated with an application and what functionality is represented by the application, or anything about the business value of the data held in the folder. These questions have to be answered through patient and detailed investigation.

As part of the inventory process, it is helpful to categorize public folders according to their use, because this might determine whether it is worthwhile to move the data and what suitable platforms exist as a migration target. Common uses for public folders include:

- Repositories for forms-based applications, sometimes workflow in nature. These applications are much more common in companies who deployed the first few versions of Exchange (pre-Exchange 2000) because of the heavy emphasis that Microsoft then placed on electronic forms linked to public folders.

- Project document repositories. These folders store an eclectic set of items in varying formats from mail messages to Word documents to Excel worksheets. Users post items to the folders on an on-demand ad hoc basis by including the folder in project distribution groups, creating new post items in the folder, or dragging and dropping from Outlook or Windows Explorer.

- Archives for discussion groups. Exchange has never offered list server functionality, but people have created their own basic implementation by using distribution groups as the list and a public folder (automatically copied on every post by including the SMTP address of the folder in the list) as the list archive.

One option is to ignore the data in public folders on the basis that it is old and of no relevance to current company operations. This might be true, but it might also be true that some data is required to comply with regulatory or legal requirements, so you need to know if any data falls into this category. The only cost for this option is some storage to maintain the public folder databases.

Another option is "degrade in place." This means that you leave the data and applications in place until users stop accessing them. Perhaps you run a report every month to identify folders that have not been accessed in the last 180 days and then export the contents of these folders to a PST and write the PST to DVD before you remove the folder and all its replicas.

The third option is to seek another platform for the public folder and move all content and applications there to allow you to decommission public folders and remove them completely from Exchange. While Microsoft doesn't provide any public folder migration utilities, you can use utilities such as Quest Software's Public Folder Migrator for SharePoint. As the name implies, this utility moves content from public folders to SharePoint team sites or portals. Over the last few years, SharePoint has become the most popular platform considered for public folder migration, partly because tools have slowly become available to enable the move, and partly because the Microsoft Exchange and SharePoint engineering groups have given hints that they prefer this option. See *http://blogs.msdn.com/sharepoint/archive/2008/04/01/updated-exchange-public-folder-vs-sharepoint-guidance.aspx* for some evidence on this point. It's also true that migrating to another Microsoft solution often aligns well with the IT strategy of some companies.

If you have developed code for use with public folders, you are going to have to redevelop the application for use on a new platform. While it's relatively easy to move data to a new platform, it's much more difficult to handle the accompanying logic embedded in code that was probably written several years ago. No tools to address this need exist today.

The bottom line is that public folders are well past their sell-by date as an application platform. There is no good reason to continue to invest in public folders and you need a plan to move off them before the curtain comes crashing down. Chose a new platform and invest your time into tailoring it for your purpose. It will be time that generates more rewards than trying to extract the last possible gasps out of public folders.