

realtimepublishers.com<sup>tm</sup>

# *Tips and Tricks* *Guide<sup>tm</sup> To*

# Active Directory Troubleshooting

  
**NETPRO**  
The Directory Experts

*Don Jones*

# Introduction

**By Sean Daily, Series Editor**

Welcome to *The Tips and Tricks Guide to Active Directory Troubleshooting!*

The book you are about to read represents an entirely new modality of book publishing and a major first in the publishing industry. The founding concept behind [Realtimepublishers.com](http://Realtimepublishers.com) is the idea of providing readers with high-quality books about today's most critical IT topics—at no cost to the reader. Although this may sound like a somewhat impossible feat to achieve, it is made possible through the vision and generosity of corporate sponsors such as NetPro, who agree to bear the book's production expenses and host the book on its Web site for the benefit of its Web site visitors.

It should be pointed out that the free nature of these books does not in any way diminish their quality. Without reservation, I can tell you that this book is the equivalent of any similar printed book you might find at your local bookstore (with the notable exception that it won't cost you \$30 to \$80). In addition to the free nature of the books, this publishing model provides other significant benefits. For example, the electronic nature of this eBook makes events such as chapter updates and additions, or the release of a new edition of the book possible to achieve in a far shorter timeframe than is possible with printed books. Because we publish our titles in "real-time"—that is, as chapters are written or revised by the author—you benefit from receiving the information immediately rather than having to wait months or years to receive a complete product.

Finally, I'd like to note that although it is true that the sponsor's Web site is the exclusive online location of the book, this book is by no means a paid advertisement. Realtimepublishers is an independent publishing company and maintains, by written agreement with the sponsor, 100% editorial control over the content of our titles. However, by hosting this information, NetPro has set itself apart from its competitors by providing real value to its customers and transforming its site into a true technical resource library—not just a place to learn about its company and products. It is my opinion that this system of content delivery is not only of immeasurable value to readers, but represents the future of book publishing.

As series editor, it is my *raison d'être* to locate and work only with the industry's leading authors and editors, and publish books that help IT personnel, IT managers, and users to do their everyday jobs. To that end, I encourage and welcome your feedback on this or any other book in the Realtimepublishers.com series. If you would like to submit a comment, question, or suggestion, please do so by sending an email to [feedback@realtimepublishers.com](mailto:feedback@realtimepublishers.com), leaving feedback on our Web site at [www.realtimepublishers.com](http://www.realtimepublishers.com), or calling us at (707) 539-5280.

Thanks for reading, and enjoy!

Sean Daily

Series Editor

**Note to Reader:** This book presents tips and tricks for Active Directory troubleshooting topics. For ease of use and for cross referencing, the questions are numbered.

Introduction.....	i
Q.1: What do the FSMO roles do?.....	1
The FSMO Roles .....	1
FSMO Failover .....	3
Q.2: How do I reset a computer's domain account? .....	3
Resetting the Account .....	3
Rejoining the Domain .....	4
Q.3: How do I troubleshoot the domain controller location process? .....	4
Symptoms .....	4
Verification .....	4
Corrective Action.....	5
Q.4: How do client computers locate a domain controller?.....	6
Starting Up.....	6
Finding the Domain Controllers .....	6
Selecting a Domain Controller.....	7
Q.5: How can I manually sync the time of a client computer with the domain?.....	7
Q.6: How can I tell if my PDC Emulator is working?.....	8
Symptoms .....	8
Verification .....	9
Corrective Action.....	10
Q.7: How does domain time sync work? .....	10
The Components .....	11
Non-Member Computers .....	11
Member Computers .....	12
Domain Controllers.....	12
The Forest Root PDC Emulator .....	14
Adjusting Time .....	14
Q.8: How do I move FSMO roles from one domain controller to another?.....	15
Transferring the Schema Master Role .....	16
Transferring the Domain Naming Master Role .....	17
Transferring the RID Master, Infrastructure Master, or PDC Emulator Roles.....	18

Q.9: I have users who can't log on to the domain for some reason. How do I find the cause of the problem? .....	19
Symptoms .....	19
Verification .....	19
Corrective Action.....	20
Q.10: How does Kerberos work?.....	21
Roles in Kerberos.....	21
The Kerberos Process .....	21
Advanced Kerberos Concepts.....	24
Q.11: How do I configure Kerberos?.....	24
Kerberos Settings .....	26
Q.12: How can I ensure that Kerberos is working? .....	26
Checking for the TGT.....	27
Managing Tickets.....	28
Troubleshooting Steps .....	29
Q.13: How are Relative Identifiers allocated?.....	29
The RID Master .....	29
SID Construction .....	30
RID Management.....	30
Q.14: Why do deleted Active Directory objects sometimes reappear? .....	31
AD Replication .....	31
Offline Domain Controllers .....	31
Authoritative Restores .....	32
Other Tombstone Tricks .....	33
Q.15: Why do objects sometimes appear in the Lost and Found folder in Active Directory Users and Computers? .....	34
A Home for Orphans.....	35
Handling Found Objects .....	36
Other Lost and Found Information .....	36
Q.16: How does the Knowledge Consistency Checker work? .....	36
Automatic Intrasite Replication Topology.....	36
Automatic Connection Objects.....	39
Replication Security.....	39
Manual Connection Objects.....	39

Controlling the KCC .....	40
The KCC and Intersite Replication.....	41
Automatic Updates.....	43
Q.17: How can I force Active Directory to replicate changes? .....	43
Check the Topology .....	44
Forcing Replication.....	45
Q.18: One of my Windows NT Backup Domain Controllers stopped replicating changes from my Active Directory domain controllers. What do I do?.....	45
PDC Emulator Failure.....	45
Domain Account Failure.....	47
Domain Mode or Functional Level.....	47
BDC Logs Events 3210, 7023, or 8032 .....	48
Q.19: How does Active Directory replication work? .....	48
Deciding What to Replicate .....	49
When Replication Occurs .....	52
How Replication Travels .....	53
Q.20: How can I check the replication topology for problems? .....	53
Intersite Replication .....	54
Intrasite Replication .....	56
Q.21: How does DNS work? .....	56
The Process .....	56
DNS Replies.....	58
Record Keeping .....	59
Sharing Zones .....	59
IP Address to Name .....	59
Communications .....	60
Data Storage.....	60
Q.22: How do trusts work under AD? .....	60
Win2K Intra-Forest Trusts.....	60
Foreign Trusts .....	61
Trusting Trees .....	62
Inter-Forest Trusts.....	64
Q.23: I've heard the term <i>split brain DNS</i> ; what does it mean? .....	65
Splitting the Brain .....	65

How it Works .....	66
Split-Brain Pitfalls .....	66
Same-Site Split-Brain .....	66
Q.24: Why won't Active Directory let me create new objects? .....	67
Troubleshooting the Problem.....	68
Preventing the Problem.....	68
RID Threshold Changes.....	69
Q.25: How can I see why some DNS queries are failing? .....	69
Nslookup .....	71
Q.26: How does Active Directory use the Domain Name System? .....	72
Basic Records.....	72
Extra Zones .....	72
SRV Records.....	72
Using DNS .....	74
Q.27: How can I use a non-Microsoft DNS server to support AD? .....	74
Q.28: How do I troubleshoot FRS problems?.....	75
Verify Connectivity and the Service.....	75
Verify Active Directory .....	76
Test FRS.....	76
File and Disk Problems.....	76
FRS Utility .....	76
Log Files .....	77
Q.29: How do Active Directory permissions work?.....	78
The Birth of Permissions .....	78
Genetic Permissions.....	79
Permissions Summary.....	82
Q.30: When promoting a server to be a domain controller, the DNS server couldn't be located. What can I do? .....	82
Basic Checks .....	82
SRV Record Checks .....	83
DDNS.....	83
Checking DNS with Network Monitor .....	84
Q.31: How does the File Replication Service work?.....	86
Replication and SYSVOL.....	87

Key FRS Features .....	87
FRS Replication Process.....	88
FRS Caveats.....	88
FRS Security and Details .....	89
Q.32: Active Directory permissions can be complicated. What's an easy way to troubleshoot permissions issues? .....	89
Effective Permissions.....	90
Finding the Problem.....	90
Q.33: How does Active Directory communicate? .....	91
Basic Communications .....	92
RPC Endpoint Mapping.....	92
RPC over HTTP .....	93
Choosing Your Battles.....	93
Q.34: How do I troubleshoot Active Directory communications issues?.....	94
Know the Sequence.....	94
What to Expect.....	96
No Fixes, Just Symptoms.....	98
Q.35: DNS works sometimes, but sometimes it doesn't. What can we do?.....	98
Low-Hanging Fruit .....	99
Replication Issues .....	99
Protocol Problems .....	99
Q.36: How can we troubleshoot Group Policy application?.....	100
Start at the Bottom .....	100
Centralized Troubleshooting.....	101
Remember the Pecking Order.....	102
Replication Problems Equals Inconsistency .....	102
Q.37: Are WAN links are being over-utilized by Active Directory replication traffic. If yes, why?.....	103
Secret Bridges .....	104
Being Smarter than AD.....	105
Making the Change .....	106

## Copyright Statement

© 2003 Realtimepublishers.com, Inc. All rights reserved. This site contains materials that have been created, developed, or commissioned by, and published with the permission of, Realtimepublishers.com, Inc. (the "Materials") and this site and any such Materials are protected by international copyright and trademark laws.

THE MATERIALS ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. The Materials are subject to change without notice and do not represent a commitment on the part of Realtimepublishers.com, Inc or its web site sponsors. In no event shall Realtimepublishers.com, Inc. or its web site sponsors be held liable for technical or editorial errors or omissions contained in the Materials, including without limitation, for any direct, indirect, incidental, special, exemplary or consequential damages whatsoever resulting from the use of any information contained in the Materials.

The Materials (including but not limited to the text, images, audio, and/or video) may not be copied, reproduced, republished, uploaded, posted, transmitted, or distributed in any way, in whole or in part, except that one copy may be downloaded for your personal, non-commercial use on a single computer. In connection with such use, you may not modify or obscure any copyright or other proprietary notice.

The Materials may contain trademarks, services marks and logos that are the property of third parties. You are not permitted to use these trademarks, services marks or logos without prior written consent of such third parties.

Realtimepublishers.com and the Realtimepublishers logo are registered in the US Patent & Trademark Office. All other product or service names are the property of their respective owners.

If you have any questions about these terms, or if you would like information about licensing materials from Realtimepublishers.com, please contact us via e-mail at [info@realtimepublishers.com](mailto:info@realtimepublishers.com)





## Q.1: What do the FSMO roles do?

**A:** In general, all domain controllers in an Active Directory domain are created equal. That is, they all have the ability to both read from and write to the Active Directory database and are essentially interchangeable. However, certain operations within a domain and forest must be centrally coordinated from a single authoritative source. These operations are handled by only one domain controller within the domain and are divided into five distinct operational categories. These categories are referred to as *Flexible Single Master Operations* (FSMOs).

The term *flexible* refers to the fact that no particular domain controller must handle these operations. Instead, the five FSMO *roles* can be held by any one domain controller; in fact, all five roles can be held by a single domain controller if you desire. When you install the first Active Directory domain in a new forest, the first domain controller you create automatically holds all five roles, and will continue to do so unless you manually move one or more of the roles to another domain controller.

### The FSMO Roles

The five FSMO roles are as follows:

- **Schema master.** This role is held by only one domain controller per forest. This role coordinates all changes to the Active Directory schema, and is required in order to process any schema updates. Only the schema master is permitted to replicate schema changes to other domain controllers in a forest.
- **Domain naming master.** This role is held by only one domain controller per forest. This role handles all changes to the forest-wide domain namespace, and is the only role that can process the addition or removal of a domain to or from the forest.
- **RID master.** This role is held by only one domain controller per domain. This role manages the *relative identifier* (RID) pool for the domain (for more information about RIDs, see the sidebar “Relative Identifiers in a Domain”). This role is also responsible for moving objects from one domain to another within a forest.
- **PDC emulator.** This role is held by only one domain controller per domain. This role is the central authority for time synchronization within a domain, and emulates the functionality of a Windows NT 4.0 Primary Domain Controller (PDC). Any NT Backup Domain Controllers (BDCs) in a domain replicate from the PDC emulator. Pre-Windows 2000 (Win2K) clients without the Microsoft Directory Services Client (DSClient) contact the PDC emulator to change user and computer passwords. The PDC emulator is also responsible for processing account lockouts. Finally, any failed logon attempts are first forwarded to the PDC emulator before returning a bad logon message to the client.



The PDC emulator is the one FSMO role that your domain cannot live without for very long. This role should be placed on a robust server computer, and you should monitor that computer closely to ensure that the PDC emulator is functioning correctly. Because the PDC emulator processes account lockout, it is a key piece of Active Directory's security infrastructure.

- **Infrastructure master.** This role is held by only one domain controller per domain. This role updates object *security identifiers* (SIDs) and distinguished names (DNs) in cross-domain object references.

#### Relative Identifiers in a Domain

All security principals, such as users and computers, in a domain have a unique SID that identifies the principal on access control lists (ACLs) in the domain. SIDs consist of two major portions: the domain SID, which is the same for all SIDs within a domain, and a RID, which is unique for each security principal within a domain. The combination of the domain SID and the RID make the resulting SID completely unique across domains, even though different domains can issue the same RIDs.

The RID master allocates small pools of unique RIDs to each domain controller in a domain. Domain controllers use this pool to assign RIDs when creating new security principals. When a domain controller runs out of available RIDs, the domain controller contacts the RID master to obtain a new pool. Because all RIDs originate from a single source, the RIDs are guaranteed to be unique within the domain.



You might sometimes see references to a sixth FSMO role, the Global Catalog (GC). Although the GC is an extra function that can be assigned to a domain controller, it is not a FSMO. Domains and forests can contain multiple domain controllers acting as a GC server, whereas FSMOs are by definition held by one, and only one, domain controller at a time.




For more information about the FSMO roles, refer to the Microsoft article "Windows 2000 Active Directory FSMO Roles."

The following list provides some best practices for placing FSMOs:

- In a multiple-domain forest, never place the infrastructure master role on a domain controller that is also a GC server. The infrastructure master's job is to update cross-domain references, and it does so by looking for references it does not itself possess. Because a GC contains a reference to every object in the entire forest, the infrastructure master will never be without a reference, and will therefore fail to perform its job properly.
- Because the PDC emulator holds such a crucial, central role in Active Directory, you should place the PDC emulator on a domain controller that has the best possible connectivity to other domain controllers in the domain. The PDC emulator in the forest root domain synchronizes time for all other PDC emulators in the forest, and should have a reliable network connection to the domain controllers holding the role in each domain.
- You should place the schema master on a domain controller that is physically collocated with the administrators responsible for maintaining the forest's schema. This placement will ensure that the administrators have a reliable connection when performing schema updates.


## **FSMO Failover**

Active Directory does not provide automatic failover for the FSMO roles. Some of the roles, such as the schema master, aren't required for Active Directory's day-to-day operations, so automatic failover isn't strictly necessary. However, some roles, such as the PDC emulator, control critical domain operations, and you'll notice pretty quickly if the domain controller containing the role fails. In those cases, you'll have to manually relocate the FSMO role to another domain controller.

 For more information about how to move FSMO roles, see Question 8.


## **Q.2: How do I reset a computer's domain account?**

**A:** Normally, computers' domain accounts are self-maintaining. Computers authenticate to the domain automatically upon startup, and periodically change their domain passwords without your intervention. However, it is possible for a computer's domain account to have problems, which can require you to reset the account.

 Resetting a computer's domain account will break the link between the computer and the domain. The computer will have to be joined to a workgroup (thus removing it from the domain), then re-joined to the domain.

## **Resetting the Account**


You can reset a computer account by using either the Microsoft Management Console (MMC) Active Directory Users and Computers snap-in or a command-line utility. To use Active Directory Users and Computers, open Active Directory Users and Computers, and locate the computer's account. By default, Active Directory places computer accounts in the Computers container. However, your organization might place computer accounts in another organizational unit (OU). Right-click the computer account, and select Reset from the context menu.

 You can't perform this procedure with a domain controller. Generally, there's no need, as the computer can always contact itself to reset its own password. However, if a domain controller's Active Directory account becomes unsynchronized, you'll have to use DCPromo to remove and reinstall Active Directory.

To use a command-line utility, run


```
dsmod computer computername -reset
```

replacing *computername* with the name of the computer you want to reset.

 You must be a Domain Administrator, Enterprise Administrator, or have the appropriate delegated permissions to perform these tasks.

## **Rejoining the Domain**


Once its account is reset, a computer will be unable to authenticate to the domain. Essentially, you've changed the computer's password and have no way to tell it what the new password is. The only solution is for you to remove the computer from the domain, then rejoin it to the domain.

 A side effect of the computer being unable to authenticate to the domain is that no users will be able to log on to the computer by using domain credentials.

To rejoin the domain on a Windows XP Professional computer (the process for Windows 2000—Win2K—is similar), right-click My Computer, and select Properties from the Context menu. On the Computer Name tab, click Change. Select Workgroup, and click OK to close all dialog boxes. You will need to restart the computer. Return to the Computer Name tab after restarting, and click Change again. Select Domain, provide the appropriate domain name, and click OK. You will need to provide the appropriate user credentials to add the computer back into the domain. After completing these steps, the computer should be able to authenticate to the domain. Restart the computer and ensure that the domain logon problem is resolved.

## **Q.3: How do I troubleshoot the domain controller location process?**

**A:** Computers that are unable to locate a domain controller for their domain won't be able to log on and won't be able to process user logons. Troubleshooting the domain controller location process is a key part of solving many logon problems.

 For more information about how computers locate a domain controller, see Question 4.

## **Symptoms**

Symptoms of a client's inability to locate a domain controller include an inability to log on to the domain and an inability to process user logons. You might also see System and Security event log messages indicating that a domain controller for the appropriate domain could not be found.

## **Verification**

To determine the cause of the problem, follow these steps:

1. Verify that the computer has the correct IP configuration for its subnet, including IP address, DNS server, and default gateway. To do so, open a command-line window and run  
`ipconfig /all`  
to display the configured information. If the configuration is wrong, correct it.
2. Use the Ping utility to verify network connectivity to the configured default gateway, DNS server, and at least one domain controller in the local site. If you cannot verify connectivity, troubleshoot and correct the problem.
3. Run

```
netdiag /v
```

to report on any problems with Windows' networking components. Correct any error conditions that are reported by using Netdiag /fix or by manually correcting the problem.



Netdiag is included in the Support Tools on the Windows CD-ROM.

**4. Run**

```
nltest /dsgetdc:domainname
```

replacing domainname with the name of the domain that you are trying to log on to. This command verifies that a domain controller can be located. Nltest is included in Support Tools.

- 5.** Use the Nslookup tool to verify that DNS contains the proper records to locate a domain controller. If either of the following tests fail to return records with the proper host names and IP addresses, restart your domain controllers to force them to register with DNS (also ensure that DNS is configured to accept dynamic updates and can accept service resource (SRV) records):

```
nslookup fully-qualified-server-name
```

where *fully-qualified-server-name* is the complete DNS name of a known domain controller, such as dc1.mydomain.com

```
nslookup guid._msdcs.rootdomain
```

where *rootdomain* is the complete DNS name of the root domain, such as mydomain.com

**6. On your domain controllers, run**

```
dcdiag /v
```

to check for many common domain controller problems. Correct any error conditions that are reported.

### **Corrective Action**


Assuming that your client computer has a proper network configuration and is otherwise able to connect to a domain controller (using ping, for example), the problem is most likely in your DNS resource records, or your domain controller is not functioning properly.

If DNS does not contain the proper records, restart a domain controller. Doing so should re-register the domain controller in DNS; if it fails to do so, then either DNS is at fault or that particular domain controller has failed. Verify that other domain controllers can register with DNS. If they cannot, replace your DNS server. If they can, the original domain controller has failed and might need to be removed from the network.

If DNS contains the proper records, but a domain controller continues to not respond to client requests, restart the domain controller. If doing so fails to correct the problem, you will most likely need to demote the domain controller to a member server, then reinstall Active Directory by re-promoting the server.



Note that very few client-side conditions exist other than basic network misconfiguration that prevents a client from locating a domain controller. Most of the problems are in the DNS server or with a domain controller that fails to respond to a client's initial requests.


 For additional troubleshooting steps that help verify Lightweight Directory Access Protocol (LDAP) connectivity, refer to the Microsoft article "How Domain Controllers Are Located in Windows" for Win2K and "How Domain Controllers Are Located in Windows XP" for Windows XP Professional.

## **Q.4: How do client computers locate a domain controller?**

**A:** One of the first major tasks a domain member computer has to do when it starts is to locate a domain controller. Generally, this task requires the use of a Domain Name System (DNS) server, which contains records for each domain controller in the domain, and the Locator, a remote procedure call to the computer's local Netlogon service.

### **Starting Up**

When the client computer starts, its Netlogon service starts automatically (in the default configuration). This service implements the DsGetDcName application programming interface (API), which is used to locate a domain controller.

 In this context, "client computer" is any computer attempting to contact a domain controller. This definition includes member servers.

The client begins by collecting a number of pieces of information that will be used to locate a domain controller. This information includes the client's local IP address, which is used to determine the client's Active Directory site membership, the desired domain name, and a DNS server address.

### **Finding the Domain Controllers**

Netlogon then queries the configured DNS server. Netlogon retrieves the service resource (SRV) records and host (A) records from DNS that correspond to the domain controllers for the desired domain. The general form for the queried SRV records is *\_service.\_protocol.domainname*, where *service* is the domain service, *protocol* is the TCP/IP protocol, and *domainname* is the desired Active Directory fully qualified domain name (FQDN). For example, because Active Directory is a Lightweight Directory Access Protocol (LDAP)-compliant directory service, clients query for *\_ldap.\_tcp.domainname* (or *\_ldap.\_tcp.dc.\_msdcs.domainname* when locating the nearest domain controller).

Each domain controller in a domain will register its host name with the SRV record, so the client's query results will be a list of domain controller host names. The client also retrieves the associated A records, providing the client with the IP address of every domain controller in the domain. The client then sends an LDAP search query, via the User Datagram Protocol (UDP), to each domain controller. Each domain controller then responds, indicating that it is operational. The Netlogon service caches all of this information so that finding a domain controller in the future won't require a repeat of this initial process. Instead, the service can simply refer to its cache to find another domain controller.


### **Selecting a Domain Controller**

After the client locates a domain controller, the client uses LDAP to access Active Directory on a domain controller, preferably one in the client's own subnet. The domain controller uses the client's IP address to identify the client's Active Directory site. If the domain controller is not in the closest site, then the domain controller returns the name of the client's site, and the client tries to find a domain controller in that site by querying DNS. If the client has already attempted to find a domain controller in that site, then the client will continue using the current, non-optimal domain controller.

Once the client finds a domain controller it likes, it caches that domain controller's information, and the client will continue to use that domain controller for future contacts (unless the domain controller becomes unavailable).

### **Q.5: How can I manually sync the time of a client computer with the domain?**

**A:** You might need to manually synchronize the time of a client computer within a domain. Note that this need should be the exception rather than the rule; Windows 2000 (Win2K) and later computers in a domain should automatically synchronize time with a domain controller. Manually synchronizing the time will not resolve the underlying time sync issue, but might temporarily resolve any other problems that arise from a lack of proper time sync (such as user and computer logon issues). See the sidebar "Manual Sync as a Troubleshooting Step" for more information about manually synchronizing the time.

 For details about how the automatic time sync process works, see Question 7.

#### **Manual Sync as a Troubleshooting Step**

One common reason to manually synchronize a computer's time is as a troubleshooting step. For example, if you notice System event log entries from the W32Time service, which indicate that time synchronization failed, you might attempt to manually sync the time as a troubleshooting step.

Typically, failed time synchronization is the result of the computer being unable to contact a domain controller, and you should troubleshoot that problem directly. Once the W32Time service fails to locate a domain controller, it will reduce its activity to location attempts every 16 hours until restarted. You'll see System event log messages to this effect, with Event ID 64, whenever the service is unable to locate a domain controller for a long period of time.



To manually synchronize time, open a command-line window, and run

```
net stop w32time
```

Run

```
w32time -update
```

Run

```
net start w32time
```

Manually verify the synchronization between the client computer and a domain controller. Also check the System event log to ensure that the W32Time service has not logged additional error messages.

## **Q.6: How can I tell if my PDC Emulator is working?**

**A:** The PDC emulator plays a vital role in the operation of any Active Directory domain. It's responsible for time synchronization, processing account lockouts, and more. If the PDC emulator fails, several key domain functions, including security functions, can stop functioning properly.

### **Symptoms**

If your domain exhibits any of the following symptoms, you need to verify the status of the PDC emulator role:

- Users are unable to log on—This symptom can occur if the domain's time synchronization becomes more than about 5 minutes out of sync. The reason is that the PDC emulator is the central source for time sync; a general lack of domain time synchronization can often be traced to the PDC emulator.

 For more information about troubleshooting domain time sync, refer to [http://www.Microsoft.com/windows2000/techinfo/reskit/samplechapters/dsbi/dsbi\\_add\\_gouy.asp](http://www.Microsoft.com/windows2000/techinfo/reskit/samplechapters/dsbi/dsbi_add_gouy.asp).

- User accounts that should be locked out aren't locked out—The PDC emulator processes account lockouts for the entire domain.
- Pre-Windows 2000 (Win2K) clients are unable to change their passwords—The PDC emulator provides password-change processing for non-Active Directory client computers.
- Windows NT Backup Domain Controllers (BDCs) are not receiving updates to the domain user lists—The PDC emulator is responsible for replicating these updates to down-level domain controllers.



## **Verification**

Some of the symptoms of a PDC emulator failure can be traced to a replication failure, network failure, or other condition unrelated to the PDC emulator. To verify proper operation of the PDC emulator, follow these steps:

- Identify the domain controller that has the PDC emulator role. From the command line of any domain controller, run

```
dsquery server -hasfsmo pdc
```

The command-line utility will report the fully qualified name of the domain controller believed to hold the role. Note that *server* is an actual dsquery parameter and not the name of a particular server on your network.

- Verify network connectivity to the domain controller by using the ping command. Then attempt to connect to the domain controller by using the Active Directory Users and Computer console from another domain controller or client computer. If either of these steps fail, troubleshoot the domain controller for basic network connectivity. Also ensure that all domain controllers in the domain are running the same Windows service pack level.
- Verify that Active Directory replication is working properly. On the domain controller holding the PDC emulator role, run

```
repadmin /showreps servername
```

supplying the server name of the domain controller that holds the PDC emulator role. Any errors indicate a problem with Active Directory replication, which you should resolve.



The Repadmin tool is provided as part of the Support Tools, which are available on the Windows Server product CD-ROM.

- Verify that the PDC emulator role is functioning. On the domain controller holding the PDC emulator role, force a user account to lock out (by logging on with a bad password multiple times, for example). Verify that the account appears locked out in Active Directory Users and Computers on the domain controller. If not, the PDC emulator has failed. If the account locks out, verify that the locked out status replicates to other domain controllers in the domain. If it does not replicate to some domain controllers, troubleshoot for Active Directory replication failure. If it does not replicate to any domain controllers, the PDC emulator role might have failed.




You will need to be familiar with your domain's account lockout policy in order to effect an account lockout. Note that disabling an account is not the same as the account being locked out, and will not be handled the same by the PDC emulator.

### **Corrective Action**


If you determine that the PDC emulator has failed, try these corrective actions:

- If the domain controller believed by Active Directory to hold the PDC emulator role no longer exists, seize the role on another domain controller in the domain.
- If the domain controller containing the PDC emulator role is still functioning, restart it. Re-verify the proper function of the PDC emulator. If it is still not working properly, attempt to transfer the PDC emulator role to another domain controller. If you cannot, remove the domain controller from the network and seize the PDC emulator role on another domain controller.
- If the domain controller that holds the PDC emulator role has failed, seize the PDC emulator role on another domain controller. Remove the original domain controller from the network and do not reconnect it until it has been repaired and demoted to member server status in the domain.

 For steps about transferring or seizing the PDC emulator role, refer to Question 8.

### **Q.7: How does domain time sync work?**


**A:** Active Directory domains rely on synchronized time for a number of key operations. For example, all Kerberos messages passed between domain members and domain controllers have a maximum default lifetime of 5 minutes. This limitation is intended to make the messages useless to attackers who might capture the messages on the network; by the time the attacker is able to decrypt and modify the packet (if the attacker is able to do so at all), the message will be useless.


 Active Directory replication is not especially time-sensitive and only uses timestamps as a tiebreaker in certain circumstances.

If time synchronization fails, the operations that depend upon it can also fail. For example, suppose a client computer has a local time of 10:00am, and its closest domain controller has a local time of 10:06am. Kerberos packets time-stamped by the client will be considered out of date by the domain controller even if those packets arrive immediately. The domain controller will reject the logon attempt, and the client will display a generic logon failure message—making it a difficult problem to detect without careful detective work. Understanding how the time sync process works is important to keeping the process running smoothly and solving any problems that occur.

## **The Components**

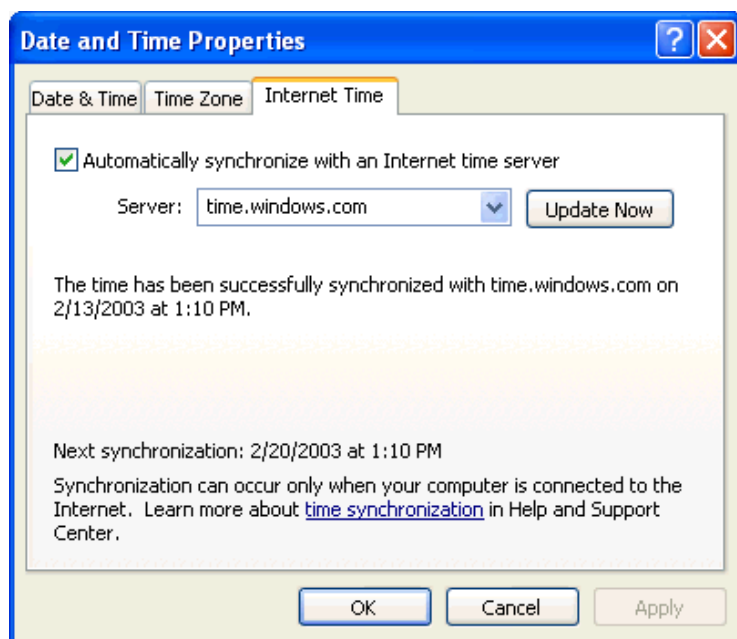
Windows 2000 (Win2K) and later include the Windows Time (W32Time) service, which is configured as a standard Windows background service to start automatically and to log on as the special LocalSystem account. This service is a Microsoft implementation of a Request for Comment (RFC)-compliant Simple Network Time Protocol (SNTP) client and server.

 For more information about SNTP, refer to the Internet Engineering Task Force (IETF) RFC 2030.

 The Windows Time service originally provided in the Windows NT resource kit is not compatible with W32Time. However, Microsoft makes an NT-compatible W32Time service, which allows NT computers to participate in Active Directory domain time sync. Obtain the updated service from Microsoft.

## **Non-Member Computers**


Computers that aren't members of a domain don't automatically synchronize time. However, because the W32Time service is a standard SNTP client, you can configure it to synchronize with any available SNTP server. To do so, simply double-click the clock in the computer's taskbar, and select the Internet Time tab, which Figure 7.1 shows.



**Figure 7.1: Manually configuring time sync.**

As Figure 7.1 shows, Windows XP Professional includes a number of preconfigured Internet time servers, including one made available by Microsoft. Other time sources include the official United States government time server at <http://www.time.gov>.

Obviously, time sync can only occur when the computer is connected to the Internet, so it works best if your computer utilizes an always-on broadband connection. However, Windows will automatically detect a network connection—such as a dial-up connection—and attempt time sync when necessary. By default, Windows attempts to sync about every 8 days.

 Windows doesn't synchronize the system date only the time. Furthermore, Windows won't synchronize the time if the date isn't correctly set.

## **Member Computers**

Within a domain, time synchronization is a good deal more complex because there are so many computers that need to be in sync with one another. At the top of the time sync authority list is the domain controller that holds the PDC emulator role in the forest root domain. That domain controller is considered the most authoritative source for time information in the entire forest, and the time sync process attempts to bring all other domain clocks into sync with that domain controller.

All domain controllers within a domain attempt to synchronize time. If possible, they try to find a reliable time service in their parent domain. If unavailable, they'll try for a reliable time service in their own domain. Generally, the reliable time service is held by the domain controller that holds the PDC emulator role for the domain. This query process of determining a reliable time service is a bit complex, and I'll cover it in more detail next.

All domain controllers holding the PDC emulator role will try to sync time with the PDC emulator of their parent domain. This behavior creates an easy-to-understand hierarchy of time sync leading up to the forest root domain's PDC emulator.

All client computers synchronize time with the domain controller that authenticates them to the domain. The key, then, is in how domain controllers (other than those holding the PDC emulator role) decide which computer to synchronize time with.

## **Domain Controllers**

Domain controllers will nearly always select their parent domain's PDC emulator as their time source. However, if that computer is unavailable or does not respond promptly, they will attempt to instead synchronize with their own domain's PDC emulator. Each domain controller's selection is based upon an initial query; if, for example, the parent domain's PDC emulator doesn't quickly respond to the initial query, the domain controller will be more likely to choose the PDC emulator from its own domain.

The entire time sync process for domain controllers ranks time sources by *stratum*. Stratum one is an external time source, such as the US Naval Observatory (which I'll discuss in the next section). The forest root PDC emulator represents stratum two. All domain controllers accessing time from the forest root PDC emulator are stratum three, and any domain controllers that get their time from *them* (the domain controllers accessing time from the forest root PDC emulator) are in stratum four, and so forth. Each stratum is progressively less accurate due to network latency, local clock inaccuracies, and so forth.

Windows includes the ReliableTimeSource registry entry, which optimizes time synchronization. When set to 1 on a domain controller, the Netlogon service on that domain controller broadcasts an announcement that the domain controller is a reliable time service. Other domain controllers will prefer a reliable time service if one is available. Generally, this registry entry should only be set to 1 when the computer is in stratum two (synchronizing with an external time source).

When a domain controller starts, it will attempt to locate a time source:

- In the same site
- Marked as a reliable time source (stratum two)
- In the parent domain (which by definition will be in a higher stratum)
- That is a PDC emulator

These attributes are ranked from most important to least important, and result in a selection preference something like the following order (from most preferred to least preferred):

- Parent domain controller, same site, marked as reliable
- Local domain controller, same site, marked as reliable
- Parent domain controller, same site, not marked as reliable
- Local PDC emulator, same site, not marked as reliable
- Parent domain controller, not the same site, marked as reliable
- Local domain controller, not the same site, marked as reliable
- Parent domain controller, not the same site, not marked as reliable
- Local PDC emulator, not the same site, not marked as reliable

This list can enable you to determine which time source a domain controller will attempt to select. Keep in mind that if such a source is available but is too busy to respond to a domain controller's initial query, the domain controller will try to find the next preferred source on the list.



Computers will never choose themselves to sync with. They'll move on to the next preferred source on the list, if necessary.


### **The Forest Root PDC Emulator**


The PDC emulator in the forest root domain does not attempt to synchronize time with anyone; it considers itself authoritative by default. For the best time synchronization, however, you should configure this domain controller to synchronize with an authoritative, Internet-based time source. To do so, open a command-line window on the domain controller. Run

```
net time /setsntp:server
```

replacing *server* with the fully qualified name of an authoritative time source.

The US Naval Observatory is considered the United States' official source of time. The observatory maintains a cesium-based atomic clock that is the most accurate timepiece in the world. This clock is connected to several Internet-based time servers that can be used by the public (including [ntp2.usno.navy.mil](http://ntp2.usno.navy.mil) and [tock.usno.navy.mil](http://tock.usno.navy.mil)). Note that the SNTP protocol uses UDP port 123 by default, so your domain controller will need access to the Internet on that port in order to sync time.

 If your network spans multiple time zones, you should always configure your forest root PDC emulator to synchronize with an authoritative outside time source. Doing so will ensure that your entire network receives authoritative time, and that time-dependent network operations will work as smoothly as possible.


 For Microsoft's documentation about configuring a time source, see the Microsoft articles "How to Configure an Authoritative Time Server in Windows 2000" and "How to Configure an Authoritative Time Server in Windows XP."

### **Adjusting Time**

When computers sync time, they don't necessarily make an instant, radical adjustment to their local clocks. Doing so could disrupt other processes, so the time sync process takes a more gradual approach.


First, the W32Time service exchanges network packets with its selected time source to determine network latency. Doing so provides an internal adjustment based on how much time is required for time sync packets to physically cross the network, and is measured in nanoseconds.

Next, the service examines the target time provided by its time source. If the target time is ahead of the current local time, the service immediately adjusts the local time to the target time. A slow local clock can be a major problem for Kerberos and other operations, so any other potential disruptions by the sudden shift in time are considered secondary concerns.

 The actual formula used to calculate target time is specified in RFC 1769, and is  $\text{LocalClockOffset} = ((\text{ReceiveTimestamp} - \text{OriginateTimestamp}) + (\text{TransmitTimestamp} - \text{DestinationTimestamp})) / 2$ , which accounts for network latency.


If the target time is behind the current local time, the local time is slowed until it equals the target time. Effectively, local time will begin to pass more slowly until the target time catches up. However, if the local time and target time are more than 3 minutes out of sync, the local clock is immediately adjusted to match the target time.

Time sync normally occurs about every 45 minutes by default. Once time sync has successfully completed at least three times, the period is extended to 8 hours, and remains at that setting so long as each attempt to sync is successful. If any attempt fails, time sync reverts back to 45 minutes.


 Sync periods are more frequent than 8 days for computers that use the Kerberos protocol because time is so important to this protocol.

### **Q.8: How do I move FSMO roles from one domain controller to another?**

**A:** On occasion, you might need to transfer one or more Flexible Single Master Operation (FSMO) roles from one Active Directory domain controller to another. Perhaps the domain controller holding the FSMO has failed, or you simply need to relocate the FSMO role to optimize Active Directory performance.

 For information about how the FSMO roles work and what they do, see Question 1.

There are two means of transferring FSMO roles: *seizing* and what I'll refer to as a *peaceable* transfer. In a peaceable transfer, which Microsoft documentation simply refers to as a *transfer*, the domain controller already holding the FSMO role is online and functioning perfectly, and you're usually transferring the role to optimize domain performance. This situation is ideal, and I prefer to accomplish the transfer through the various Active Directory consoles. However, if the domain controller holding the FSMO has failed, you might need to seize the role from a different domain controller, which I prefer to accomplish with command-line tools.

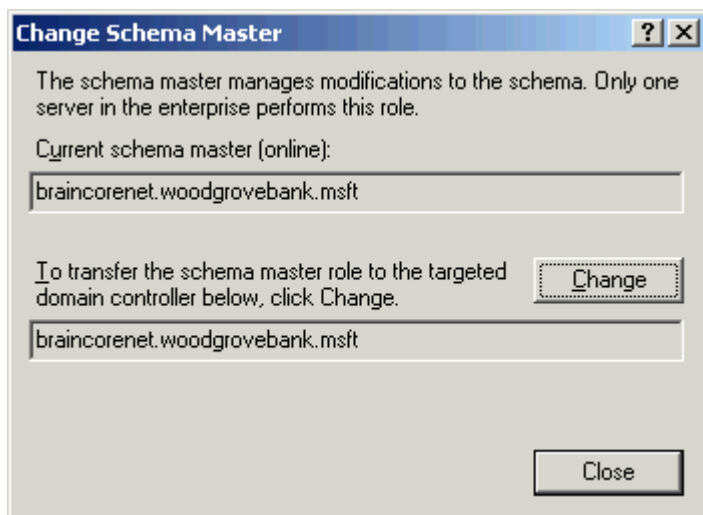
 Seizing a FSMO role can result in problems if the original domain controller is ever returned to service. I'll discuss these concerns in more detail for each FSMO role.

### **Transferring the Schema Master Role**

You accomplish a peaceable transfer of the schema master by using the Schema console. By default, Windows does not provide a preconfigured Schema console or even make the console snap-in available for your use. Follow these steps to gain access to the Schema console:

1. Open a command prompt window.
2. Change to the \Windows\System32 folder.
3. Run  
`regsvr32 schmmgmt.dll`  
to register the Schema snap-in.
4. Close the command prompt window.
5. Open a blank Microsoft Management Console (MMC) window.
6. From the File menu, choose Add/Remove Snap-In.
7. Click Add.
8. Locate Active Directory Schema in the list, and double-click it.
9. Click Close, then click OK.

To transfer the schema master, right-click Active Directory Schema, and select Operations Master from the pop-up menu. You'll see the Change Schema Master dialog box, which Figure 8.1 shows.




**Figure 8.1: Changing the schema master role.**

Click Change, and select the domain controller that you want to hold the schema master role.



To seize the schema master role:


1. Open a command prompt window.
2. Run Ntdsutil.
3. At the Ntdsutil command prompt, enter  
`roles`
4. Enter  
`connections`
5. Enter  
`connect to server servername`  
providing the fully qualified name of the domain controller that you want to seize the schema master role.
6. Enter  
`qui`
7. Enter  
`seize schema master`

 After you seize the schema master, do not return the original schema master domain controller to the network. Doing so will result in an authority conflict. If you are able to repair the original domain controller, first demote it to a member server while it is disconnected from the network, reconnect it, then reinstall Active Directory using DCPromo.

### ***Transferring the Domain Naming Master Role***

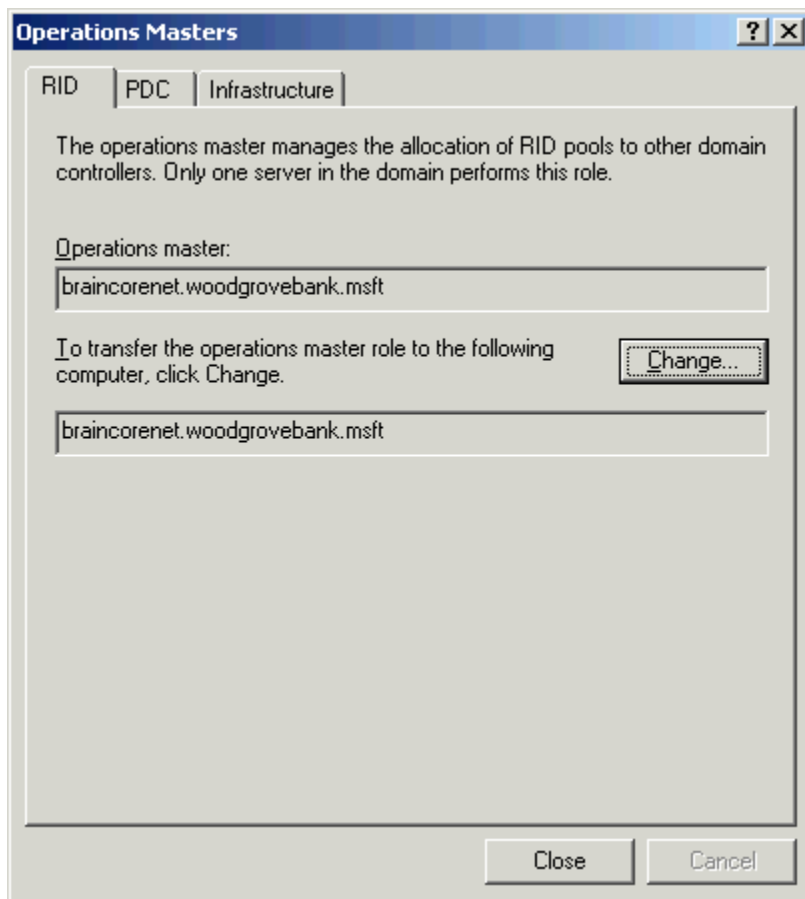
The process of transferring the domain naming master role involves basically the same steps as transferring the schema master does. However, rather than using the schema console, you'll use the Active Directory Domains and Trusts console on the domain controller that currently holds the role. Simply right-click Active Directory Domains and Trusts in the console, and select Operations Masters from the pop-up menu.

Seizing this role requires the use of the Ntdsutil command-line utility. Run the utility as described for seizing the schema master role, but enter `seize domain naming master` at the appropriate prompt.

 If you are forced to seize the domain naming master role, do not return the original domain controller to the network until it has been demoted to member server status.

### **Transferring the RID Master, Infrastructure Master, or PDC Emulator Roles**


Transferring the RID master, infrastructure master, or PDC emulator roles involves basically the same steps as transferring the schema master role. However, rather than using the schema console, you'll use the Active Directory Users and Computers console, which is configured by default on every domain controller. Simply right-click the appropriate domain in the console, and select Operations Masters from the pop-up menu. You'll see a dialog box similar to the one that Figure 8.2 shows, which provides a separate tab for transferring each of the three domain-specific roles.



**Figure 8.2: Transferring the domain-specific FSMO roles.**

Seizing these roles also requires the use of the Ntdsutil command-line utility. Run the utility as described for seizing the schema master role, but enter the appropriate seize command at the appropriate prompt:

- To seize the PDC emulator role, use `seize PDC`.
- To seize the infrastructure master role, use `seize infrastructure master`.
- To seize the RID master role, use `seize rid master`.


 If you are forced to seize any of these roles, do not return the original domain controller that held the role to the network until that domain controller has been demoted to member server status. Although this step isn't strictly necessary with the PDC emulator role (the original holder of the role will generally "let go" when it comes back online), I prefer safe over sorry and always demote the original holder before returning it to the network.

### **Q.9: I have users who can't log on to the domain for some reason. How do I find the cause of the problem?**

**A:** A number of different problems can lead to users (or computers) being unable to authenticate to the domain. To easily resolve these problems, you need an ordered troubleshooting methodology that simply moves from one possibility to another until you arrive at the actual root cause of the problem.

#### **Symptoms**

The symptoms of this problem are obvious: Users can't log on to the domain. However, keep in mind that an underlying problem might be that the user's client computer might not be logging onto the domain either. Client computers that are domain members maintain their own security account in the domain, and must authenticate to the domain before they are able to log on a user account. Normally, client computers attempt to authenticate to the domain as soon as Windows starts. Log on locally and check the System and Security event logs for error messages that indicate that the computer was unable to authenticate.


 I'll assume that you've already attempted to reset the user's domain password, checked the Caps Lock key, and the other common culprits of logon problems. For the sake of this discussion, I'm assuming that the user's domain password isn't the problem.

#### **Verification**

First, ensure that the computer is authenticating to the domain. You can use the Nltest utility from the Support Tools package on the Windows CD-ROM to verify authentication. If authentication isn't working, use the same verification steps presented here, but focus on the computer's domain account rather than the user's account.

#### **Can the Domain Controller Be Found?**

First, verify that the computer is able to locate a domain controller. Check the System event log of the client computer for messages indicating that a domain controller couldn't be found. If error message are present, troubleshoot the domain controller location process.

 For more information about how to do so, see Question 3 and Question 4.


You can also force the domain controller location process to run by using the Nltest support tool and running

```
Nltest /dsgetdc:domain
```


on a client computer (replacing *domain* with the appropriate domain name). If Nltest fails to locate a domain controller, you've found your logon problem.

### Is Time Synchronized?

By default, Windows' Kerberos authentication protocol allows for a 5-minute difference between the clocks of client computers and domain controllers. Kerberos messages are time-stamped at their source and cannot be more than 5 minutes old when they arrive at their destination. If your computers are not time-synchronized correctly, packets might be considered old by the destination even if they arrive almost immediately.

 You can configure your domain security policies to allow a longer period of time difference. However, doing so makes your network more vulnerable, as it gives attackers additional time to decrypt and re-use Kerberos messages captured from the network. Rather than extending the Kerberos time tolerance, you should correct the time synchronization problem.

Checking time synchronization is easy: Simply check the clocks on the client computer and a domain controller. If they are more than a minute or two off, troubleshoot time synchronization problems.

 For information about how to troubleshoot time sync problems, see Question 7, and check out Question 5 for steps on manually correcting a time sync problem.

### Is the Computer Account Valid in the Domain?

Use Active Directory Users and Computers to verify that the computer's domain account isn't locked out or disabled. Also, a computer account that has been cloned from one domain to another can cause problems until the original source account is deleted. If event log messages on the client computer continue to indicate logon problems, you might need to reset the computer's domain account.

 For step-by-step instructions, see Question 2.


### Corrective Action

Corrective action for logon problems will be determined by the root cause of the problem:

- If the client computer is unable to locate a domain controller, troubleshoot and resolve that problem.
- If the client computer's time is out of sync with the domain, troubleshoot and resolve that problem or manually synchronize the time. Note that a manual synchronization will not permanently resolve the problem.
- If the client computer's domain account isn't working, resolve that problem. Doing so might require you to re-enable the account or even reset it.

## **Q.10: How does Kerberos work?**

**A:** Kerberos is an industry-standard authentication protocol and part of the TCP/IP suite of internetworking protocols. Originally developed at MIT, Kerberos is now up to version 5, which is primarily defined in Internet Request for Comments (RFC) 1510.

 You can find RFC 1510 at <http://www.ietf.org/rfc/rfc1510.txt>. Microsoft has proposed several extensions to Kerberos that are used in Windows 2000 (Win2K) and later; review RFCs 3244 and 1964 at the same site for more information about Kerberos specifics and Microsoft extensions.

Kerberos provides a number of advantages over Microsoft's older authentication protocols:

- The burden of authentication is placed on the client. Older protocols placed the burden on the server, creating a less distributed and less scalable architecture.
- Authentication between clients and servers is mutual, meaning both are assured of the other's identity. Older protocols focused primarily on the server being assured of the client's identity.
- Kerberos uses strong encryption technologies and timestamps, making it very difficult to compromise. Older protocols relied primarily on well-known encryption hashes and simple obfuscation for protection, which resulted in the success of cracking utilities such as L0phtCrack.

### ***Roles in Kerberos***

Kerberos is named for the dog in Greek mythology that guarded the gates to the underworld. Like the three-headed mythical dog, the Kerberos protocol identifies three primary roles for authentication participants:

- The Key Distribution Center (KDC) is responsible for issuing authentication tickets to clients.
- Clients are computers that attempt to access resources located on a server.
- Servers are computers that contain resources being accessed by clients.

Any discussion of Kerberos in the Microsoft world can become confusing because computers often hold all three roles. For example, all domain controllers in an Active Directory (AD) domain are KDCs. They might also be file servers, making them Kerberos servers. You might also log on to a domain controller's console and attempt to access files on another server, in which case the domain controller becomes a Kerberos client. So in any discussion of Kerberos, it's important to consider what role each participant is playing in a single particular authentication transaction.

### ***The Kerberos Process***

Kerberos defines two basic processes, which are functionally very similar to one another. The first process occurs whenever a security principal, such as a user or computer, logs on to an AD domain. The second process occurs whenever a logged-on principal attempts to access a resource located on another computer. For information about how encryption relates to Kerberos, see the sidebar "Encryption and Kerberos.")



Remember that both users and computers have accounts in an AD domain, and that both are considered individual security principals. For example, the logon process occurs twice on every Windows XP Professional computer: Once when the computer starts, and again when a user logs on to the computer's console.

### Encryption and Kerberos

Understanding how digital encryption works is a key to understanding Kerberos. By default, Kerberos uses *shared secret* encryption, which means that both the sender and recipient know a secret phrase or password that is used as an encryption key. Generally, this shared secret is the recipient's password, something that both the recipient—whether a computer or a user—and the KDC both know.

In Win2K and later, Microsoft extends the standard Kerberos protocol to support Public Key Infrastructure (PKI) and digital certificates. In this scenario, the security principals and the KDC don't share a secret. Instead, they use each others' public keys—obtained from a certificate authority (CA) server—to encrypt data, and their own private keys—stored in the computer or on a smart card—to decrypt data.

### Logging on and Obtaining a TGT

When a security principal authenticates to a domain, the principal has to prove its identity to the domain controller. This is done by creating a special packet of data called an *authenticator*, then encrypting the packet and sending it to the KDC. A clear-text (unencrypted) version of the authenticator is also sent. How the authenticator is encrypted depends on the encryption scenario in use.

Usually, the principal's hashed password is used as the encryption key. For example, if the principal is a user, the user would type his or her user name and password into the logon dialog box on the client computer. The client computer hashes the password and uses the result to encrypt the authenticator. The domain controller receives the authenticator and looks up the user account in AD to retrieve its copy of the hashed password. If the hashed password from AD can be used to decrypt the authenticator, then the user must have typed the correct password. If the user hadn't, the password stored in AD wouldn't be able to decrypt the authenticator.



A *hash* is the result of a one-way encryption algorithm, meaning it is impossible to start with the encrypted result and ever arrive at the original unencrypted password. Domain controllers only store hashed passwords for principals; when passwords are changed, they are immediately re-hashed and stored.

In a PKI scenario, the user's client computer would use a smart card containing the user's private key to encrypt the authenticator. Smart cards generally require the user to provide a PIN or some other identifying factor (such as a thumbprint) to unlock the card. Once the domain controller received the authenticator, it would obtain the user's public certificate from AD or a CA, and try to use the public key to decrypt the authenticator. If the decryption works, the user's identity would be verified.



Technically, anyone could intercept the authenticator and use the user's public key to decrypt it. Because the authenticator doesn't contain any sensitive data, that's fine. An intruder could not modify and re-encrypt the authenticator without the user's private key, which is safely stored in the smart card.



The authenticator includes a timestamp and a unique identifying number. Domain controllers will reject any authenticators that are too old (5 minutes, by default) or that the domain controller has already processed. This behavior reduces the likelihood that an intruder could capture an authenticator off of the network and replay it against the domain controller at a later time (presumably after modifying it in some way).

Once the domain controller is convinced of the principal's identity, it creates a *ticket-granting* ticket. The TGT contains a unique, temporary encryption key that the domain controller makes up on the spot. By using this *session key* for future communications, the user's regular encryption key won't be overexposed on the network. One copy of the TGT is encrypted with the domain controller's private encryption key, and can be decrypted only by that domain controller. That copy of the TGT, along with an unencrypted copy, are encrypted using the principal's shared secret (or, in a PKI scenario, with the principal's public key), then sent back to the principal.

When the principal receives the TGT, the principal decrypts it—using either its shared secret or private key—and store it in memory. So now the principal has two important pieces of information:

- An unencrypted TGT containing the session key to be used for future KDC communications.
- An encrypted TGT that can only be decrypted by the KDC.

The TGT is also marked with an expiration time (8 hours, by default) and a maximum lifetime (7 days, by default). When the TGT expires, the principal can ask the KDC to renew the TGT for up to the maximum lifetime. Expiration times and maximum lifetimes ensure that session keys aren't used for such a long time that an intruder might be able to compromise them. Once the two copies of the TGT are in RAM, the logon process is complete.



TGTs are stored in a special non-swappable area of volatile RAM. In other words, the TGTs can't be written to disk under any circumstances and will be lost if the client computer loses power. User TGTs are also destroyed when the user logs off.

## Accessing Server Resources

By now, the principal has a TGT from the KDC. That TGT is a key part of further authentication operations, such as attempting to access resources on a server. Note that Kerberos doesn't directly provide *authorization*, which determines *which* resources a principal may access. Kerberos simply provides *authentication*, which tells the server who the principal is. The server can then use other technologies—such as NTFS file access control lists (ACLs)—to determine what the principal is allowed to do.

When a client needs to access resources on a server, it contacts the KDC and requests a *ticket* for that server. The ticket request is accompanied by the KDC-encrypted copy of the TGT from the client's memory, and the request itself is encrypted with the session key from the unencrypted copy of the TGT. The result:

- The KDC can decrypt its half of the TGT to retrieve the session key, which ensures that the session key wasn't tampered with.



- The KDC uses the now-decrypted session key to decrypt the ticket request. This re-validates the client's identity; only this client had a copy of the session key, and it matches the one contained in the TGT.
- The KDC didn't have to persist any information about the client because the client is presenting all the necessary information—in the request and TGT—to revalidate the client's identity to the KDC.

Satisfied with the client's identity, the KDC generates a ticket. This ticket contains a new, unique session encryption key. One copy of the ticket is encrypted with the session key shared between the client and KDC. Another copy is encrypted with a secret shared between the KDC and the server to which the client is requesting access. Both copies are sent back to the client.

The client can decrypt one half of the ticket to obtain a session key, which it uses to encrypt an access request packet. The client sends this encrypted packet to the server, along with the other half of the server ticket—the half that only the server can decrypt.

When the server receives the request, it decrypts its half of the server ticket, revealing a session key. The server then uses the session key to decrypt the actual request. The result:

- The server is assured that a KDC validated the client's identity because only the KDC would have generated a valid server ticket that the server could decrypt.
- The client is assured of the server's identity because only the proper server would have been able to decrypt the server ticket.


The server can now process the access request and provide the client with the resources requested.

### **Advanced Kerberos Concepts**

Kerberos has an additional advantage over Microsoft's older authentication protocols: delegation and proxying. These techniques essentially allow a background service or application to access resources using a user's security credentials, ensuring that the service or application can only access resources that the end users would normally be allowed to access on their own. However, both delegation and proxying require services and applications that have been specially programmed to deal with Kerberos' advanced features, making a discussion of how these features work beyond the scope of tip.


## **Q.11: How do I configure Kerberos?**

**A:** Kerberos is configured entirely within Active Directory (AD) and is a part of the Default Domain Policy Group Policy object in every domain. As an administrator, you can modify several Kerberos' key configuration settings.

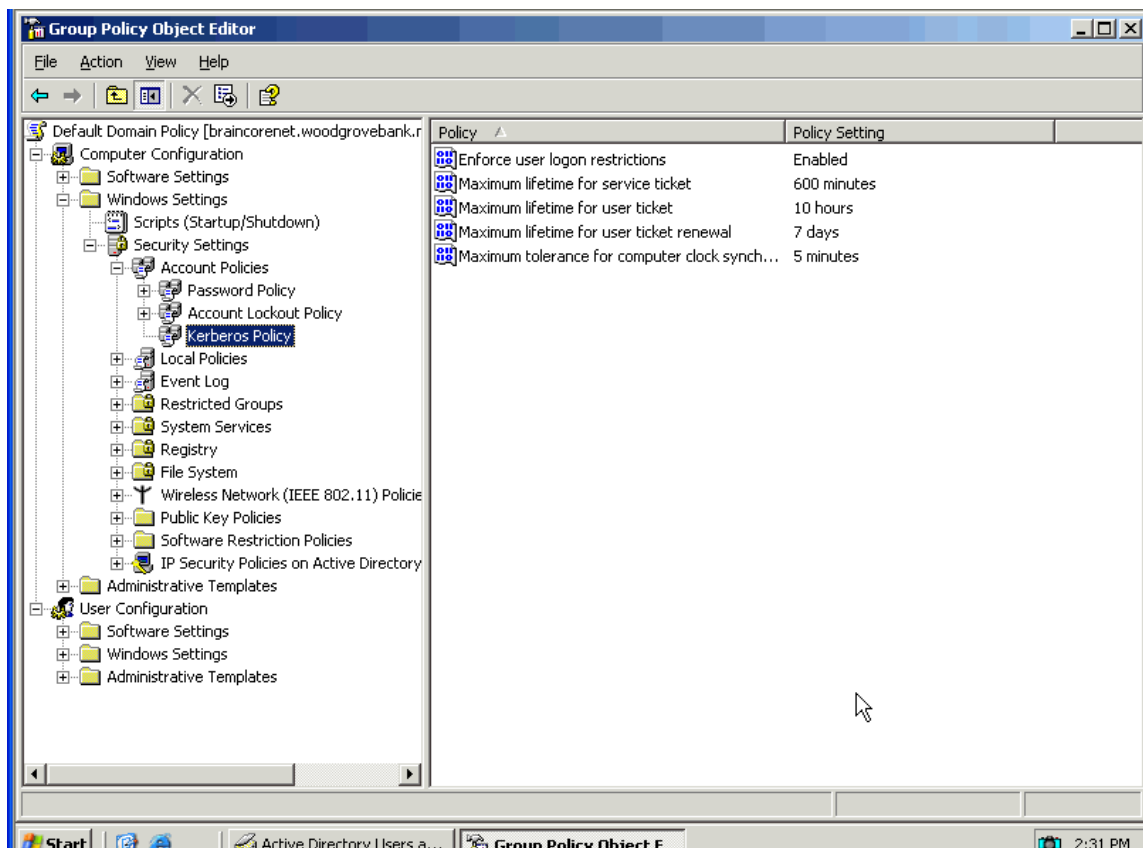
 In general, you shouldn't need to modify the default Kerberos configuration settings. Microsoft came up with the default settings based on a typical work environment, and it's rare for them to be unsuitable or wrong. Be sure that you understand exactly what each setting does and how it impacts your domain before making any changes.



To modify the Kerberos policies for a domain, follow these steps:

 The screen shots that follow and the default setting values are from Windows Server 2003—the steps are identical in Windows 2000 (Win2K) domains.

1. Open Active Directory Users and Computers.
2. Right-click the domain, and select Properties.
3. Select the Group Policy tab.
4. Select Default Domain Policy from the list, and click Edit.
5. Locate the Kerberos settings, shown in Figure 11.1.



**Figure 11.1: Kerberos policies in the Default Domain Policy Group Policy object.**


6. Modify the settings as appropriate.
7. Close the Group Policy Object Editor.
8. Close the Properties dialog box.

Your changes will be replicated throughout the domain.

## **Kerberos Settings**


Windows contains only five Kerberos settings:

- **Enforce user logon restrictions**—Enabled by default, this setting determines whether the Key Distribution Center (KDC) validates every request for a session ticket against the user's rights. Enabling this policy provides better security because changes to user rights will be effective for the user's next ticket request. However, the validation step does require a bit of extra processing on the part of the KDC.
- **Maximum lifetime for service ticket**—Service tickets are those requested by Windows background services to act on behalf of a user. The default 600-minute lifetime is generally sufficient and prevents a service from being able to impersonate a user for an unreasonably long period of time. Shortening this value places increased load on domain controllers, as services will have to contact domain controllers more frequently to obtain new tickets.
- **Maximum lifetime for user ticket**—User tickets are those requested by users to access resources. The default setting, 10 hours, covers a typical workday and allows a user to utilize tickets for the entire day. Lengthening this value might overexpose the ticket and leave it more vulnerable to compromise; shortening this value might place an undue burden on domain controllers.
- **Maximum lifetime for user ticket renewal**—When a ticket's maximum lifetime expires, a client computer can renew the ticket. Renewals don't require a new session key, thereby saving a bit of processing on the KDC. The default setting for this value is 7 days, ensuring that ticket session keys don't last longer than a week.
- **Maximum tolerance for computer clock synchronization**—This value defaults to 5 minutes, meaning a client computer's clock can as many as 5 minutes ahead or behind the KDC's own clock. The default value should be fine for most environments. Lengthening this value could give intruders extra time to attempt to compromise a ticket request captured from the network; if your environment has network latencies or time sync problems that seem to necessitate lengthening this setting, you are better off correcting those time or latency problems than compromising Kerberos security.

 For more information about how time affects Kerberos, see Question 10.


## **Q.12: How can I ensure that Kerberos is working?**

**A:** Troubleshooting Kerberos can be difficult; Windows pretty much does everything behind the scenes, and doesn't give an administrator a terrific amount of control over what Kerberos is doing. There are, however, some steps you can take to see what Kerberos is up to.

 You'll rely heavily on utilities included with the Windows resource kits, such as *The Microsoft Windows 2000 Server Resource Kit*. If you don't have these kits, you'll need to obtain a copy in order to follow along. Some of the utilities can be downloaded for free from Microsoft's Web site.

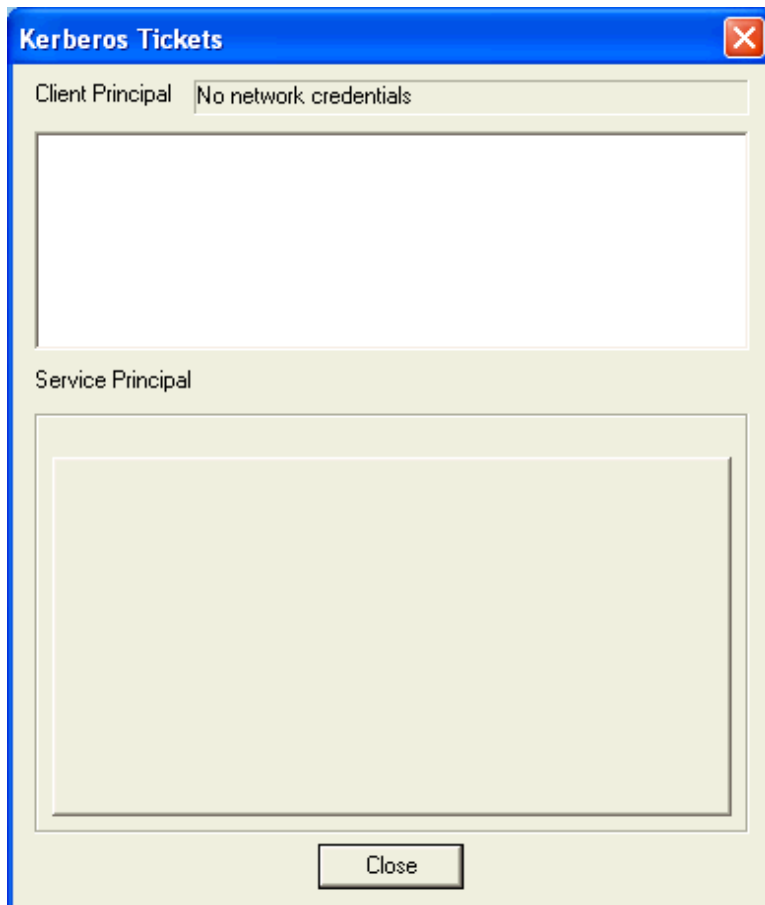
### **Checking for the TGT**

After logging on to a workstation, first ensure that the computer obtained a ticket-granting ticket (TGT) from a Key Distribution Center (KDC—domain controller). I prefer to use KerbTray to check this.

 For more information about how your computer obtains a TGT, refer to Question 10.

 You can download KerbTray for free from <http://www.Microsoft.com/windows2000/techinfo/reskit/tools/existing/kerbtray-o.asp>.


After downloading and running KerbTray, you'll notice an additional icon in the system tray of the taskbar. Simply double-click the icon to see your computer's existing tickets, including the TGT. If the window appears empty, as Figure 12.1 shows, your computer isn't authenticating to an Active Directory (AD) domain (and might not, in fact, even be a member of a domain).



**Figure 12.1:** KerbTray won't display credentials if your computer isn't logged on to an AD domain.

## **Managing Tickets**

Microsoft also provides a command-line utility, Klist.exe, in the resource kit (and as a download from <http://www.Microsoft.com/windows2000/techinfo/reskit/tools/existing/klist-o.asp>). You can use Klist not only to view tickets and your TGT (which KerbTray displays), but also to delete tickets.

 If you delete tickets, especially the TGT, you might not be able to authenticate to network resources until you log off and back on again.

Use Klist as follows:

- **klist tgt**—Displays information about the TGT in your system, including the domain that issued the TGT and how long it's good for. A system without a TGT can't access domain resources.
- **klist tickets**—Displays all cached tickets, including which server each ticket is for and how long each ticket is good. If you've been unsuccessfully attempting to access a particular resource and don't have a ticket for it, then you've found the problem.

When you check a TGT, you're looking for a few specific pieces of information in Klist's output:

- **DomainName**—The domain name should match the domain name that you logged on to.
- **KeyExpirationTime**—This time should be some date and time in the future.
- **StartTime**—This time should be some date and time in the past, even if it's the recent past.
- **EndTime**—Again a date and time in the future.
- **TimeSkew**—This date and time should be in the future, and should be greater than the **EndTime**.

If any of these values look wrong, then the KDC service on the domain controller that authenticated you might be having problems. Check the domain controller's event logs for error messages that provide additional clues.

Similarly, when checking a ticket, you're looking for:

- **Server**—This server should be the fully-qualified name of the resource you're trying to access.
- **EndTime**—Should be in the future by at least an hour or so.

If the **EndTime** value in particular is off, try deleting the ticket and letting Windows acquire a new one from the KDC.

## Troubleshooting Steps

Having trouble accessing resources in the domain? Try these steps to narrow the problem:

1. Run `Klist tgt` to check your TGT. If your TGT is expired or not present, log off and back on again to repeat. If that doesn't fix it, your computer either isn't in the domain or its domain credentials need to be reset. See *How-To: How to Reset a Computer's Domain Account* for corrective steps.
2. Run `Klist tickets` to see if you have a ticket for the resource you're trying to access. If you don't, try to access another resource. If that works, then the resource giving you problems might not be in the domain. If you can't access any resources, but have a TGT, then the KDC service on the domain controller that issued your TGT might have failed. Check the service on the domain controller, then log off and back on again.
3. If you have a valid TGT and a valid ticket, use `Klist purge` to delete the ticket associated with the resource you're having problems with. Then try to access the resource again. Windows should obtain a new ticket from the KDC; if it doesn't, refer back to step 2. If you do get a new ticket (verify with `Klist tickets`) but still can't access the resource, you need to check the resource server for problems, such as a failure in the Netlogon service or an out-of-sync domain computer account.

## Q.13: How are Relative Identifiers allocated?

**A:** Relative Identifiers (RIDs) are used to uniquely identify each object within a domain. In any Active Directory (AD) domain, each domain controller has the ability to create new objects—users, computers, groups, and so forth. Each of these new objects needs a unique ID number to avoid conflict with other new objects being created at any given time by other domain controllers in the domain.



The unique ID numbers given to each domain object are actually a combination of a domain ID and a RID; RIDs can be duplicated across domains because the combination of domain and RID will always be unique. The uniqueness of the domain ID is ensured by the forest-wide domain naming master.

### The RID Master

In order to ensure that domain controllers don't duplicate ID numbers, AD includes a special Flexible Single Master Operations (FSMO) role in each domain, called the *RID master*. The RID master's job is to allocate each domain controller with a unique range of RIDs. Because all RIDs stem from this single source and the RID master doesn't issue overlapping pools to different domain controllers, each domain controller has a unique range of "spare" ID numbers to use when creating new objects.

As part of its role in ensuring uniqueness for each AD object, the RID master is also responsible for removing the entries for domain objects that are moved to another domain. However, you should note that the RID from the removed object is never reused in the domain.



For more information about the RID master FSMO role, see Question 1.



## **SID Construction**

The unique number assigned to each domain object is called a Security Identifier (SID). A typical SID looks like this:

S-1-5-21-917267712-1342860078-1792151419-500

- S designates this identifier as a SID
- 1 indicates the revision level of the SID construction scheme
- 5 represents the identifier authority
- Everything else is the SID itself; the combination of domain ID and RID.

## **RID Management**

You can't directly affect the allocation of RIDs except through a few documented workarounds to specific operating system (OS) problems. You can view certain RID attributes directly in AD.



It is possible for a domain controller to use up its allocated RID pool more quickly than it can request a new one. For example, if you're migrating thousands of users to a domain controller that has poor connectivity to the RID master, the domain controller might run out of RIDs. For more information about this problem, see the Microsoft article "RID Pool Allocation and Sizing Changes in Windows 2000 SP4."

AD contains several attributes that contain information about RIDs; these attributes, in fact, are the sources that DisplayRID queries for its output. The major attributes are:

- FsmoRoleOwner—Contains the fully qualified domain name of the current holder of the RID master role.
- RidAvailablePool—Defines the number of security principals that the domain can contain (a fixed value currently just over 1 billion), and the number of RIDs that have been allocated already.
- RidAllocationPool—Defines the current pool for a domain controller, and its next pool.
- RidNextRid—The next RID that will be used on the domain controller.
- RidPreviousAllocationPool—The current pool of RIDs used to create new SIDs; this value includes the value of RidNextRid.
- RidUsedPool and NextRid—Unused attributes that are still defined in AD.




The values of these attributes will differ from domain controller to domain controller.

## Q.14: Why do deleted Active Directory objects sometimes reappear?

**A:** It's one of the strangest things that can happen in Active Directory (AD): You delete an object, such as a user, group, or organizational unit (OU), then a few minutes—or even days—later the object mysteriously reappears. Generally, deleting the object a second time makes it “stick,” but reappearing objects—especially users and groups—can have serious security implications. After all, if a user account reappears, it can be used to access domain resources. So where do deleted objects come back from, and how can you prevent it from happening in the future?

### AD Replication

The key to this puzzle lies within AD replication. Remember that every time an object is changed, its Update Sequence Number (USN) is incremented. When the object is replicated, it comes from the domain controller that has the highest USN for the object, and the object overwrites any older copies on other domain controllers.

 Technically, object *attributes* have USNs; for more information about how AD replication operates, see Question 7.

When you delete an object from AD, it doesn't go away immediately. Instead, the object is *tombstoned*, meaning AD places a special marker on it indicating that the object is no longer active. The tombstone itself has a USN and is replicated to all domain controllers in the domain. 60 days after the tombstone, all domain controllers independently remove the object from their copy of the AD database.

There are a few situations in which the tombstone can be removed, and even situations in which a tombstoned and deleted object can mysteriously reappear. You'll need to review these possible situations to determine which occurred in your environment, and take steps to prevent it from happening again.

### Offline Domain Controllers

Some companies decide to keep a spare domain controller off the network as a last-resort backup, in case all of their active domain controllers somehow become corrupted or fail in some fashion. Some companies also ship new domain controllers to new branch offices, and unforeseen circumstances keep the domain controller offline for longer than intended. Unfortunately, keeping an offline domain controller can cause deleted objects to return from the dead. Here's what happens:

- A domain controller is taken offline. Obviously, it is no longer replicating with the other domain controllers.
- An AD object is deleted by an administrator. This causes the object to be tombstoned for 60 days.
- If the offline domain controller is returned to the network, the results will depend on exactly when it is returned:



- If the domain controller is returned within the tombstone lifetime, then the domain controller will replicate the tombstone and the object will not reappear.
- If the domain controller is returned after the tombstone lifetime, then the formerly offline domain controller will have a copy of an object that doesn't exist on other domain controllers. Thus, most of the domain's domain controllers don't have the object, which means they don't have a USN for it. Any USN is better than none, so the formerly offline domain controller will replicate the object within AD and all other domain controllers will be happy to accept it. The object reappears.



The same scenario can occur if you restore an AD backup that's older than 60 days. For that reason, Windows' tools will try to prevent you from restoring a backup that is more than 60 days old. There are documented workarounds that can enable you to bypass this protective feature, but you shouldn't try.

How can you prevent this from happening? Take some basic precautions. First, make a backup of AD every single day. It only takes a few minutes and ensures that you have a recent backup to work from at all times. Second, never keep domain controllers offline for more than a few days. If you have to have a domain controller offline for a couple of weeks, ship a more recent AD backup that can be used to update the domain controller's database.



In Windows Server 2003, you can write the AD database backup to removable media, such as a CD-ROM. This media can be used to initialize a new domain controller's AD database. It's an ideal way to deploy domain controllers to remote offices: Ship a non-domain controller Windows Server 2003 computer to the office, use DCPromo to install AD, and initialize AD from removable media. Doing so will prevent a heavy WAN load for initial replication while ensuring a recent copy of AD is placed onto the new domain controller.

### **Authoritative Restores**

Here's another scenario: Imagine you've got an OU named Sales, which contains a user named Maria and a user named Pete. You make a backup of the AD database on Monday morning. On Tuesday morning, Pete leaves the company, so you delete his AD account. Tuesday afternoon, someone accidentally deletes Maria's user account. To correct the problem, you decide to perform an authoritative restore of the Sales OU. Maria's account comes back, but so does Pete's.

This mistake seems like a fairly obvious gaffe: You could have performed an authoritative restore of just Maria's user account. However, it's a common situation. When you perform an authoritative restore, AD increments the USNs of all restored objects by several thousand, making them the highest USNs in the domain. This forces AD to accept the restored objects—which in this case have no tombstone—as authoritative, and they are replicated throughout the domain.




Authoritative restores from backups older than 7 days can cause trust and computer account problems. The passwords for trusts and computer accounts are automatically changed about every 7 days.




Had you performed a non-authoritative restore, neither object would have reappeared. That's because the USN on the backup tape was older than the current USN held by the domain's domain controllers. When you restored the objects, they would briefly reappear on the domain controller that you restored them to, but that domain controller would quickly re-replicate the higher-USN version of the objects from other domain controllers, and the objects would again be tombstoned.

The moral is to restore as few objects as necessary when performing an authoritative restore. When restoring an entire OU, you might not realize that other objects contained within the OU will be restored, so you'll also need to perform a follow-up inspection after your restore to see what's back. For example, suppose someone had deleted the Sales OU on Tuesday afternoon, rather than deleting Maria's user account. Obviously, you'll need to perform an authoritative restore of the entire OU at this point, and Pete's account is going to come back whether you like it or not. A quick follow-up inspection of the OU after the restore will let you re-delete Pete's account and correct the problem.


 Windows Server 2003 provides a means for restoring accidentally deleted objects, effectively removing their tombstone. At least one domain controller in the domain must be running Windows Server 2003 to enable this functionality. A new user right, *Reanimate Tombstone*, provides the necessary permissions to the Domain Administrators group by default. Unfortunately, there's currently no user interface for using this feature; you'll need a custom utility to take advantage of it for now.

### **Other Tombstone Tricks**

Windows 2000 (Win2K) Service Pack 3 (and some post-SP2 hotfixes) introduced a new feature known as Strict Replication Consistency. This feature can cause additional problems for out-of-date domain controllers, causing them to log event ID 1084 in the System event log.

 The Microsoft article "Lingering Objects Prevent Active Directory Replication from Occurring" provides more information about this problem and describes how Strict Replication Consistency works.

You can also modify the tombstone lifetime in AD. The default 60-day value is designed to accommodate domain controllers that might be offline for an extended period of time and to ensure that backups are useful for a good, long while. However, you can reduce the value to as little as 2 days or even increase it. To do so, use the ADSI Edit snap-in, which Figure 14.1 shows, to edit the `tombstoneLifetime` attribute.

 There's no default console that contains the ADSI Edit snap-in; you'll need to install the support tools from the Windows CD-ROM and run `Adsiedit.msc`.

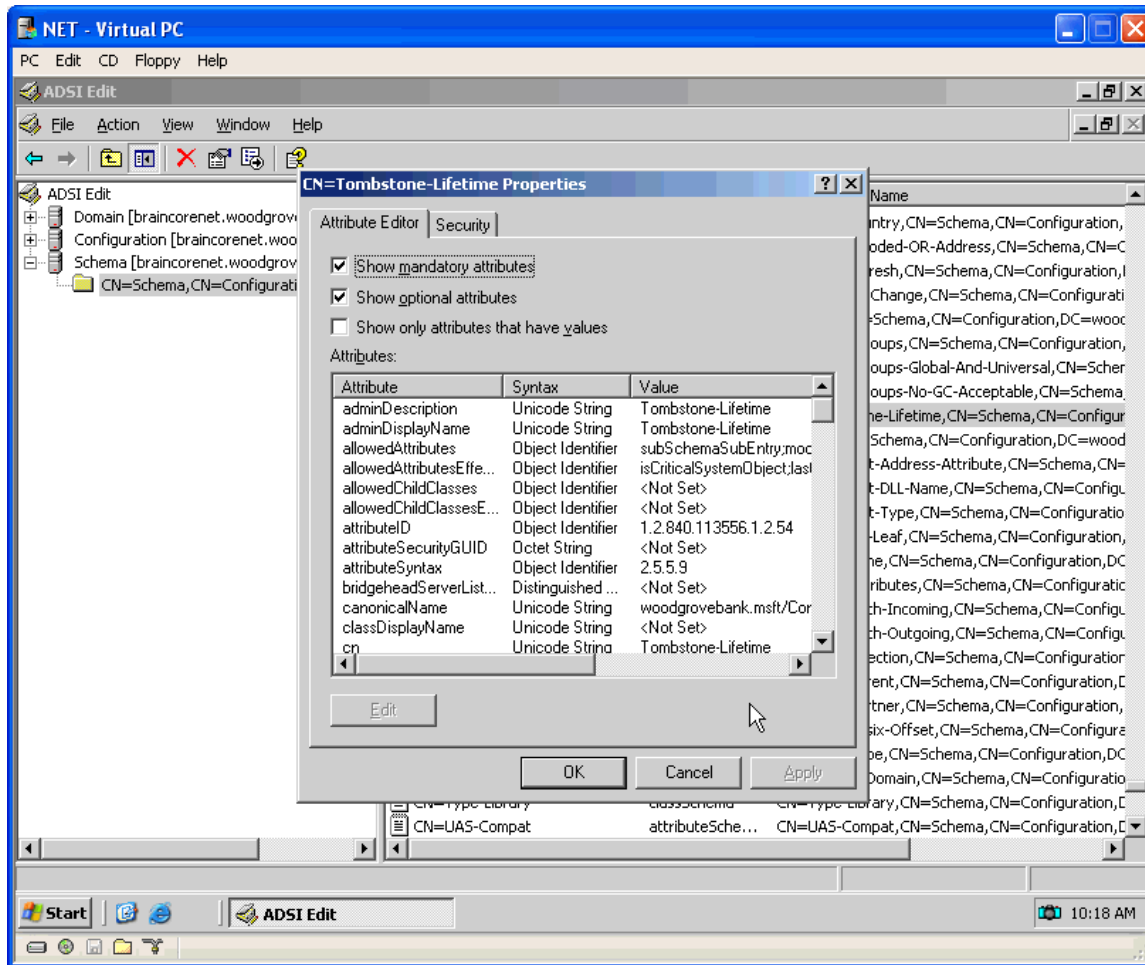


Figure 14.1: Using the ADSI Edit console to modify the tombstoneLifetime attribute.

Few administrators prefer to shorten the tombstone lifetime; many, however, lengthen it to accommodate longer domain controller absences or to extend the useful life of AD backups. In most environments, 60 days is more than sufficient. 30 days is a reasonable minimum if you specifically want deleted objects removed more quickly and aren't concerned about backup lifetime. Anything longer than 90 days is probably excessive, and will keep deleted objects on your domain controllers a lot longer than they need to be.

## Q.15: Why do objects sometimes appear in the Lost and Found folder in Active Directory Users and Computers?

**A:** Many administrators don't even notice the Lost and Found folder. It's easy to miss, especially because Active Directory Users and Computers only displays it when you configure the snap-in to display advanced options. Open up your copy of Active Directory Users and Computers, and make sure that Advanced is selected in the View menu. Ideally, your Lost and Found folder will look like the one in Figure 15.1—empty. That means that all is well.

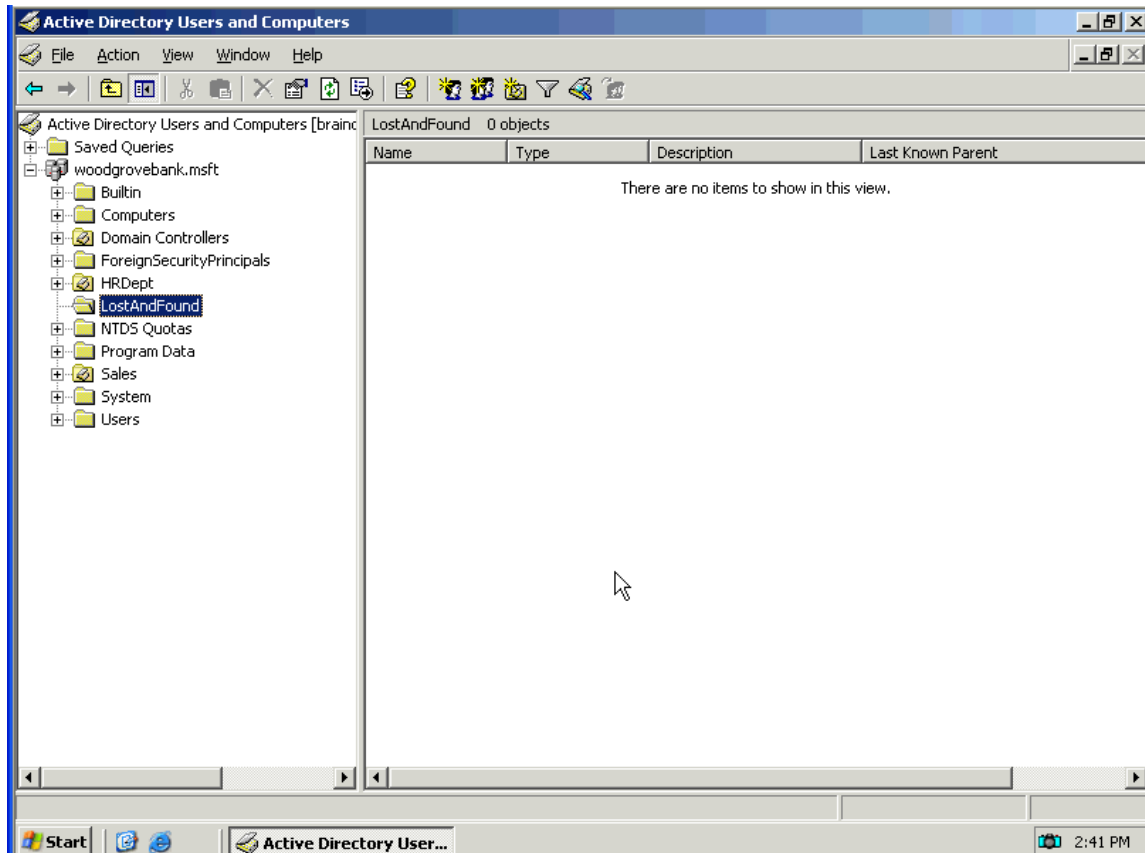


Figure 15.1: An empty Lost and Found folder.

But if Lost and Found does contain objects, how did they get there? And what should you do about them?

### A Home for Orphans

Essentially, the Lost and Found folder is a home for orphaned Active Directory (AD) objects. Objects usually become orphans through AD replication *convergence*. Every AD domain controller contains a complete read/write copy of the domain database. That means that it is possible for two administrators to make conflicting changes to AD at the same time. Eventually, AD replication will converge, resolving the discrepancy. The time it takes to do so is referred to as *replication latency*.

Most of the time, AD doesn't actually need to deal with what appears to be a conflicting change. For example, suppose one administrator changes a user's password, while another changes the user's name. AD replicates each attribute individually, so there's no conflict, even though two administrators made changes to the same user.

However, some types of conflicts can't be easily handled. For example, suppose that one administrator moved a user into the Sales organizational unit (OU), at the same time another administrator deleted the Sales OU on another domain controller. When convergence occurs, where does the user account wind up? In Lost and Found.

## Handling Found Objects

When objects appear in Lost and Found, many administrators' first instinct is to delete them, thinking they're some kind of bogus object reference. Don't! They're real AD objects, and you simply need to move them into a regular container or OU.


## Other Lost and Found Information

In some environments, administrators use copies of the Ntdsutil utility that they obtained from prerelease copies of Windows 2000 (Win2K). In the article "How to Troubleshoot an 'Internal Error' Error Message During the Replication Phase of DCPromo," Microsoft documents a problem with prerelease copies of this utility in which using it to perform an authoritative restore would incorrectly increment the internal version number of the Lost and Found container, causing an internal Windows error. Obviously, stay away from prerelease utilities!

Lost and Found has one legitimate use that doesn't indicate a problem or a replication issue—moving objects between domains. When you use Microsoft's MoveTree utility to move objects between domains, the utility first moves objects into the Lost and Found folder, they're then copied to the destination domain and removed from Lost and Found. If MoveTree fails to work correctly, you might find objects still lingering in Lost and Found. Further, if you try to use MoveTree to move objects such as computer accounts, which it can't handle, they'll wind up in a subfolder under Lost and Found.


## Q.16: How does the Knowledge Consistency Checker work?

**A:** Active Directory (AD) supports two distinct types of replication: *intrasite*, which covers all domain controllers within a site, and *intersite*, which covers replication between different sites. Intrasite replication is managed pretty much automatically by the AD Knowledge Consistency Checker (KCC). The KCC is responsible for ensuring that each domain controller within the site participates in replication in a timely fashion. The KCC also plays a role in generating the replication topology between sites. Although the KCC generally works flawlessly, it's a good idea to understand exactly how it works so that you can troubleshoot it when problems arise.

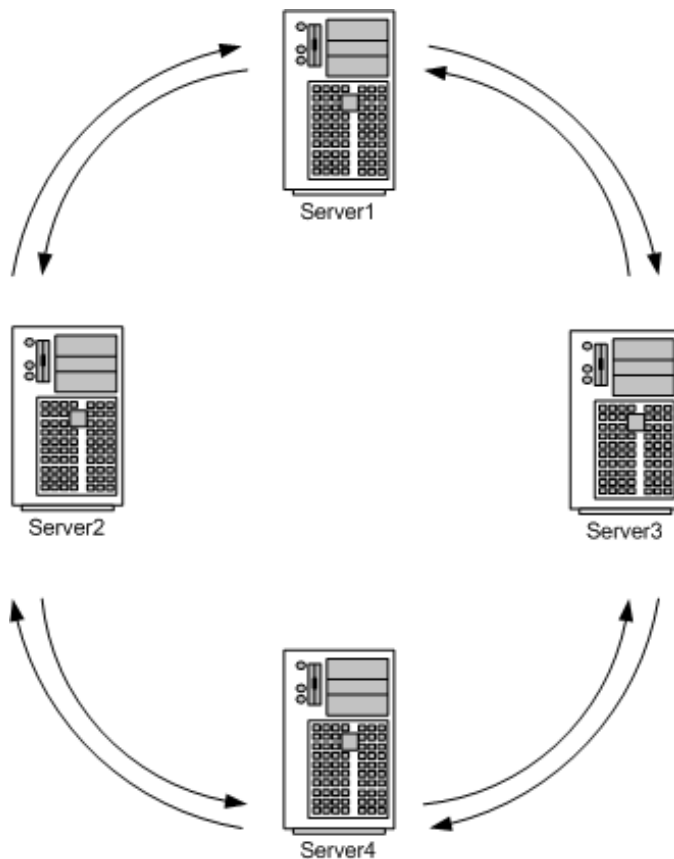
 The KCC is physically implemented as a component of a service that runs on all domain controllers.

## Automatic Intrasite Replication Topology

One of the KCC's most important tasks is to generate the intrasite *replication topology*, a sort of logical map that decides which domain controllers will replicate with each other. AD does not use a *fully enmeshed* replication topology in which each domain controller in a site replicates with every other domain controller in the site; such a topology would generate an unacceptably high level of replication traffic, especially in sites with a large number of domain controllers. Instead, the KCC tries to generate a topology that minimizes latency and network utilization while providing fault tolerance.

 Latency refers to the amount of time it takes a change to replicate from the domain controller on which the change was made to all other domain controllers. A high degree of latency can create inconsistent results for users and can even be the source of potential security problems, so reducing latency is a big priority for the KCC.

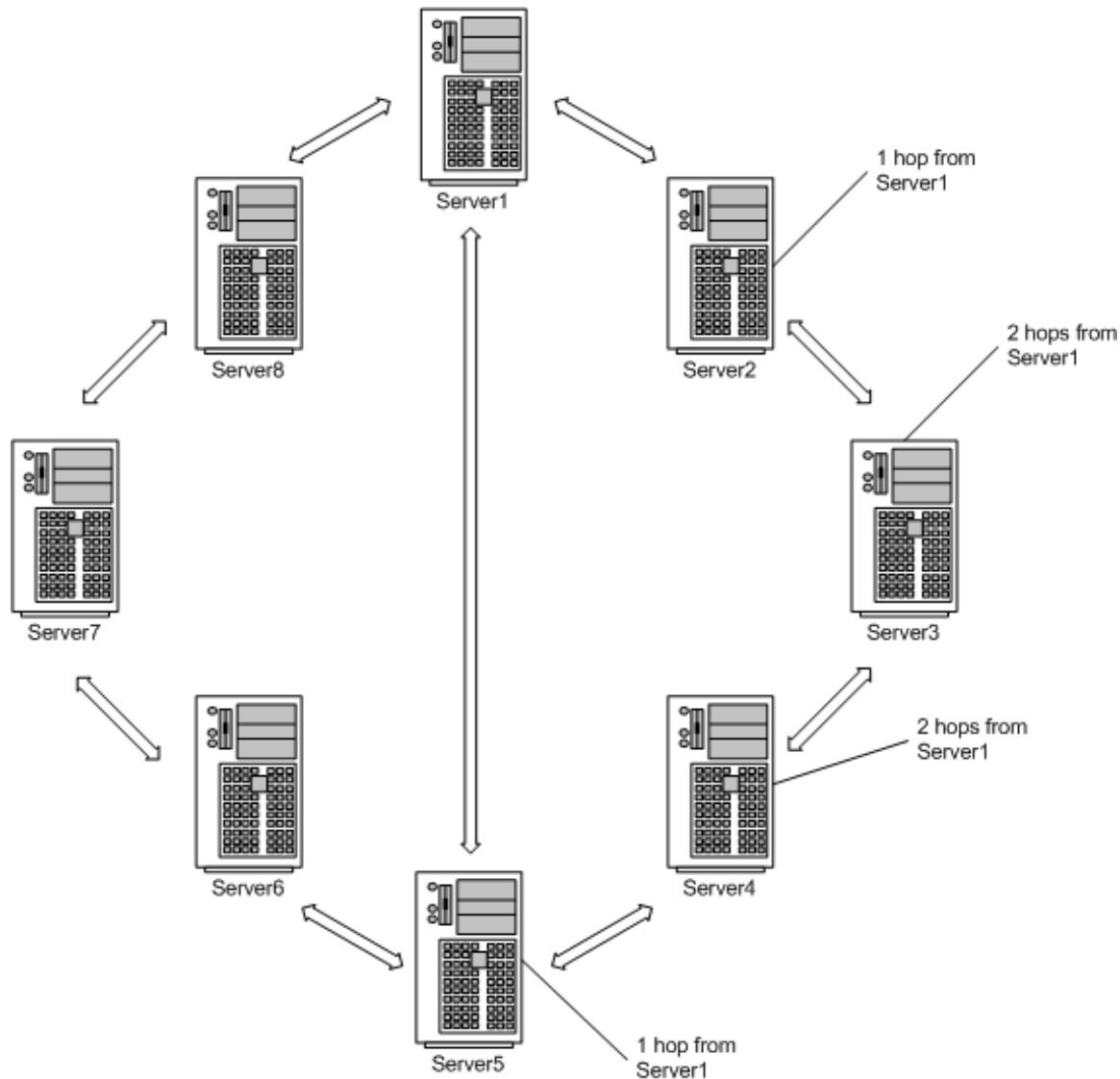
By default, the KCC tries to ensure that each domain controller in the site has at least two replication partners. That way, if one becomes unavailable, replication can still continue. Additionally, the KCC tries to ensure that no single domain controller is more than three partners away from any other domain controller. This configuration reduces latency because any change to AD would require no more than three replication cycles. Generally, the KCC creates a *bidirectional ring* topology, as Figure 16.1 shows. In this pattern, each domain controller replicates with the domain controllers to its left and right (two partners apiece).



**Figure 16.1: Bidirectional ring topology.**

In a site with many domain controllers, a ring topology can quickly violate the no-more-than-three-hops rule, so the KCC will generate shortcuts across the ring to reduce the number of hops between domain controllers. Figure 16.2 shows a ring topology with a larger number of domain controllers in which shortcuts are used to reduce hops. Note that some domain controllers are therefore required to carry more than the minimum two connections. However, to reduce domain controller overhead, no domain controller is required to have more than three replication partners.


Server5, which would have been four hops away from Server1 in a bidirectional ring without shortcuts, is reduced to a single hop by the shortcut. Note that the two-way arrows indicate two individual connection objects apiece; I've simplified them into a single arrow for this drawing.



**Figure 16.2: Bidirectional ring topology with shortcuts.**

As I've mentioned, this topology is automatically generated by the KCC. Actually, the KCC builds several different, independent replication topologies: One for each AD partition. This setup results in a schema replication topology, a configuration topology, a domain topology, and an application topology. Although the various topologies will often be similar, they might not be identical. Note, however, that the KCC never creates duplicate connection objects.

For example, suppose the KCC generates an identical topology for both the schema and configuration partitions. Only one entire site of connection objects will be created for both topologies. If the KCC generates a different topology for the domain partition, then it might create different connection objects. The KCC will never, for example, create two objects from Server1 to Server2; if different topologies require such a connection, they will all share a single connection.

 For more information about how replication works, see Question 19.

## **Automatic Connection Objects**

You can't directly affect the KCC's operation. When it creates its replication topology, the result is a set of replication objects. The security on these objects sets the KCC itself as the owner, although members of the Domain Administrators group have permission to modify those objects. As an administrator, you *can* create your own intrasite replication objects. The KCC won't have the ability to modify any objects you create. (For information about AD connection objects, see the sidebar "AD Connection Objects.")

### **AD Connection Objects**

Keep in mind that each connection object represents a one-way, inbound replication path from the domain controller on which the change occurred to the local domain controller. AD replication is *pull-based*, meaning domain controllers request changes from other domain controllers. This concept is important for security: domain controllers do *not* accept *pushed* changes, meaning there's no way for an intruder to send fake replication data around your network and mess up your domain.

When the KCC wants two domain controllers to replicate with each other, it must create two connection objects with each object representing one direction of replication traffic.

You can use the Active Directory Sites and Services console to see the connection objects that exist for each domain controller. Open the console, and expand Sites. Then select the appropriate site (such as Default-First-Site), and expand Servers. Locate the server you're interested in, expand it, and select NTDS Settings. You'll see the connection objects that control replication to other servers within the site. You'll also see the intrasite replication schedule, which is about every 5 minutes by default. Connection objects created by the KCC will have a name of <automatically generated>; connection objects you create will have a descriptive name that you assign.

## **Replication Security**

How does the KCC have permission to perform its job? The AD partitions each grant special permissions to both the Enterprise Domain Controllers group and the built-in Administrators group:

- Replicating Directory Changes—This permission lets replication actually occur.
- Replication Synchronize—This permission lets a synchronization request be issued.
- Manage Replication Topology—This permission allows for the creation and modification of connection objects, which describe the replication topology.

## **Manual Connection Objects**

When the KCC is automatically generating connections, why should you bother? There are a number of reasons. Perhaps the most common reason is to reduce latency. You might, for example, want to have no more than two hops between any two domain controllers to decrease replication latency. You can accomplish this setup by manually creating connection objects. To do so

1. Navigate to the appropriate server in Active Directory Sites and Services, and select the NTDS Settings item. Remember that you're creating an object that represents an inbound connection, so select the server that will be the incoming replication partner.



2. Right-click NTDS Settings, and select New Active Directory Connection from the context menu.
3. Select a domain controller from the list. It should be the domain controller that will be the replication partner. Note that you cannot create a connection from one server to itself.
4. Provide a name for the connection object, and you're done!

Keep in mind a couple of things regarding manually created connections:

- The KCC will never delete your connections. If a domain controller becomes unavailable, breaking a manual connection, then the KCC might automatically generate new automatic connections to compensate.
- If a KCC-generated topology requires a connection between, say, Server1 and Server2, and a manually created connection already exists, the KCC will not generate a new connection. The topology will instead use the manually created connection.

Be very careful when creating manual connections. In fact, unless you plan to regularly monitor your connections, don't create any at all. Manual connection objects can easily create additional domain controller overhead by requiring domain controllers to maintain connections that aren't as efficient. Also, the more connections you create, the fewer the KCC will create (remember that it tries to minimize the number of connections maintained by each domain controller), which can result in an inefficient replication topology.

### ***Controlling the KCC***

You can use the Repadmin.exe command-line tool to manipulate some of the KCC's functionality and to check its performance. For example, executing

```
repadmin /kcc servername
```

will force the KCC on the designated server to run and immediately begin recalculating the replication topology. As Figure 16.3 shows, you can omit the server name to run the KCC on the local domain controller.



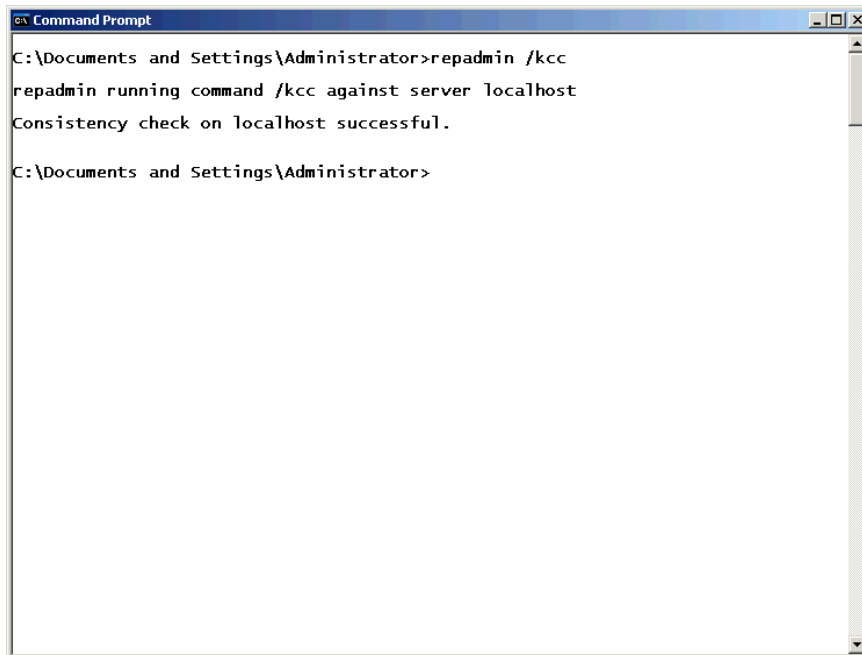


Figure 16.3: Forcing the KCC to run.

Or you can run

```
repadmin /showreps servername
```

to show the replication topology as viewed from the designated server. The output of this command displays both inbound and outbound *neighbors*, or replication partners.




Inbound neighbors are ones from which the reporting domain controller will request and pull changes. Outbound neighbors are those that will receive change notifications from the reporting domain controller.

Manually drawing the replication topology based on this information can be complex. The Windows Support Tools includes Replmon, a graphical tool that can draw a graphical picture of the replication topology to help you more easily spot topology issues.

### The KCC and Intersite Replication

The KCC also generates the topology for intersite replication. However, it cannot do so completely on its own. Instead, it must rely on information that you supply about the intersite networking infrastructure (generally WAN links). You supply this information by accurately configuring AD sites to represent your LANs, and by creating site links that represent the WAN links between LANs.

The KCC on a single domain in each site is designated as the *intersite topology generator* and is responsible for managing that site's connections to other sites. The generator uses a fancy algorithm—called a *least-cost spanning tree* algorithm—to calculate the most efficient way to replicate information across sites.

 By default, the intersite topology generator is the first domain controller in the site. However, this can change if that domain controller becomes unavailable for more than an hour; you can check Active Directory Sites and Services to see which server holds the role. You cannot, however, designate a server to hold this role. If the existing generator becomes unavailable, the domain controller with the next-highest AD GUID will become the new generator.

The KCC has to consider several factors when generating the intersite topology. These factors include the availability of site links, which tells the KCC which sites it can reach and which sites can be used to reach other sites. It must also consider the network protocols available on site links, which will usually include IP and/or SMTP. The KCC must also consider the cost of site links. In the event that multiple possible paths are available to a site, the KCC will choose the least expensive option. You configure the cost of site links when you create them.

For intersite replication, the KCC chooses a single domain controller as the *bridgehead* for each domain in the site. The bridgehead is responsible for all replication traffic to a particular site. If the bridgehead fails, the KCC will choose another from the same domain.

 Remember that only the KCC on the intersite topology generator deals with intersite replication topologies.

You can also manually designate bridgehead servers by manually creating connection objects between two domain controllers in the same domain but in different sites. The KCC will detect the existing connection object when it generates the intersite topology and use the manually created connection rather than creating a new connection. Manually designating bridgeheads ensures that you can select a server that can support the additional traffic that intersite replication entails. You can also ensure that, if desired, a single domain controller acts as the bridgehead for its domain to all other sites, providing a consolidated bridgehead (and putting all of your replication eggs in one basket, so to speak). If your designated bridgehead fails, the KCC will automatically select a new bridgehead for each site connection.

Finally, as Figure 16.4 shows, you can designate particular domain controllers as *preferred* bridgeheads. Simply right-click the server in Active Directory Sites and Services, and indicate which transport protocols the domain controller is preferred for. The KCC will always choose a preferred server as the bridgehead if one is available.

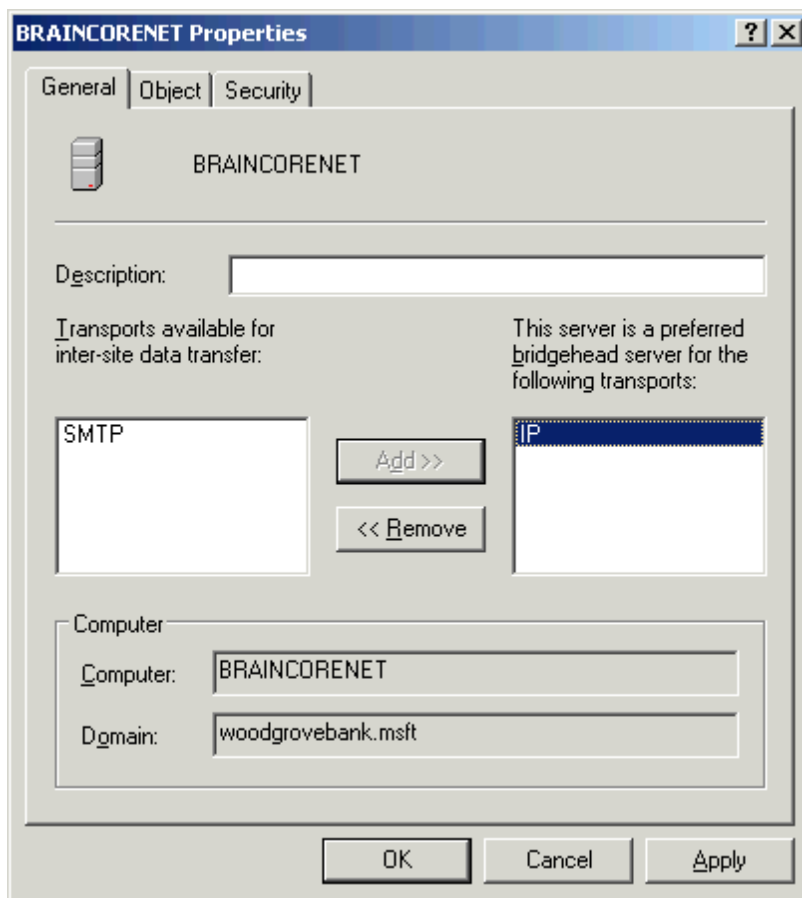



Figure 16.4: Making a domain controller the preferred bridgehead for IP connections.

## Automatic Updates

The KCC is configured to automatically recheck the network topology every so often. Doing so allows the KCC to reconfigure the topology to respond to changing network conditions, such as new domain controllers, failed domain controllers, failed WAN links, new site link configurations, and so forth.

## Q.17: How can I force Active Directory to replicate changes?

**A:** Generally, Active Directory (AD) replication works completely automatically and, due to the way its replication topology works, provides replication with very low latency. However, there might be times when—either as a troubleshooting step or as a workaround to a problem—you need to force AD perform replication.

 Having to force replication is a sign of a problem in most instances. You should attempt to fix the problem so that manual replication isn't necessary.

 For more information about how AD replication works, see Question 19.

## **Check the Topology**

Before forcing replication, try to perform a quick fix by checking the replication topology. Windows' Support Tools includes Repadmin.exe on domain controllers to help with this. Simply execute

```
repadmin /showreps servername
```

to show the replication partners for a designated server (you can omit *servername* to run the tool against the local computer, if it's a domain controller). If you suspect that one server isn't replicating properly, check its partners. Then verify that each of those partners is functioning and considers the suspect domain controller to be a partner as well.


If the topology seems to be the problem, a quick fix might be to force the Knowledge Consistency Checker (KCC) to regenerate the topology. It could be that you've caught a recent topology problem and that the KCC simply hasn't run yet. Run

```
repadmin /kcc servername
```


to force the KCC on the designated domain controller to regenerate its topology. Follow up with

```
repadmin /showreps servername
```


to see the newly selected replication partners.

 For more information about how the KCC generates the replication topology, see Question 16.

For intersite replication issues, determine which domain controller in each affected site (and domain) is acting as the bridgehead server. These will be the only domain controllers in the site with a connection object to a domain controller in another site. If a designated bridgehead domain controller is unavailable or disconnected, the intersite replication will fail. You can check these connections using either repadmin or the Active Directory Sites and Services console.

 One potential cause of intersite replication issues is that the intersite topology generator has failed within the last hour, and problems have occurred with the replication topology (such as a designated bridgehead domain controller also failing). AD will correct this problem automatically within about an hour, because it will choose a new topology generator and recalculate the intersite topology.


Why spend all this time worrying about the replication topology? Simple—forcing replication doesn't recalculate the topology. It simply forces AD to replicate using the existing topology; if that topology is flawed, then forcing replication won't solve any problems.

 For more information about troubleshooting topology issues, see Question 20.

### **Forcing Replication**

The easiest way to force replication is through the Active Directory Sites and Services console. To do so, open the console, and locate the domain controller that you want to replicate. This domain controller will request changes from its replication partners. Locate the connection over which you want to force replication, right-click the connection, and select Replicate Now.

If the domain controller that you want to replicate doesn't have any valid connection objects, you have a replication topology problem. You can provide a quick fix by manually creating a connection object to a known-working domain controller in the same site (if possible) and domain, and forcing replication over that connection.


 For additional methods of forcing replication, refer to the Microsoft article "Initiating Replication Between Active Directory Direct Replication Partners."

### **Q.18: One of my Windows NT Backup Domain Controllers stopped replicating changes from my Active Directory domain controllers. What do I do?**

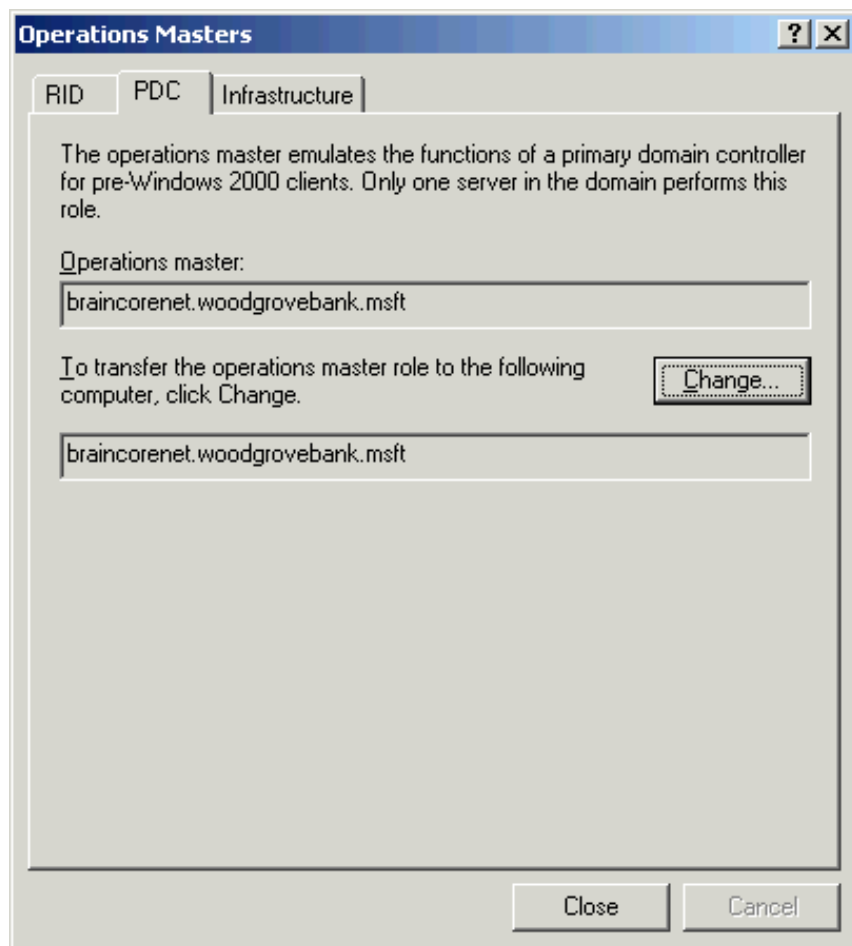
**A:** For better or for worse, many Active Directory (AD) domains still contain down-level Windows NT Backup Domain Controllers (BDCs). Perhaps your BDCs are needed to support an older application or you've got a mission-critical application running on one. Whatever the case, it's usually important that the BDC continue to replicate changes from your AD domain, just as it used to do from your NT Primary Domain Controller (PDC). Sometimes, however, replication stops working. Fortunately, there are a fairly limited number of causes.

#### **PDC Emulator Failure**

Easily the most common cause for BDC replication issues is a failure of some kind in the domain's PDC emulator. The PDC emulator is a Flexible Single Master Operation (FSMO) role that is assigned to one domain controller in the domain.


 For more information about how the PDC emulator works, see Question 1.


To troubleshoot this problem, first determine which domain controller holds the PDC emulator role. To do so, open Active Directory Users and Computers, right-click the domain, and select Operations Masters from the context menu. Select the PDC tab, which Figure 18.1 shows, and note the name of the server listed.




**Figure 18.1: Identifying the domain's PDC emulator.**

If the designated server is not available on the network or is not responding to ping commands from your NT BDC, you need to resolve that problem. You can either try restarting the PDC emulator to fix it or you might need to seize the PDC emulator role at another domain controller.

 For instructions about transferring and seizing FSMO roles, see Question 8.


 Make sure that there are no IPSec policies in effect that would prevent an NT computer from communicating with the PDC emulator. For example, applying certain IPSec policies to a Windows 2000 (Win2K) or later computer will prevent communications with down-level clients, which includes NT BDCs.


 Other issues can occur when Win2K or later domain controllers don't replicate properly with the PDC emulator; see Microsoft article "Event 1586 Message: The Checkpoint with the PDC Was Unsuccessful" for more information.

### **Domain Account Failure**

A less common cause for BDC replication failure is when the BDC's domain computer account becomes locked out or out of synch. BDCs maintain a domain account in much the same way as users and other computers do. The BDC's accounts must be available and the BDC must know the account password in order for the BDC to participate in the domain.

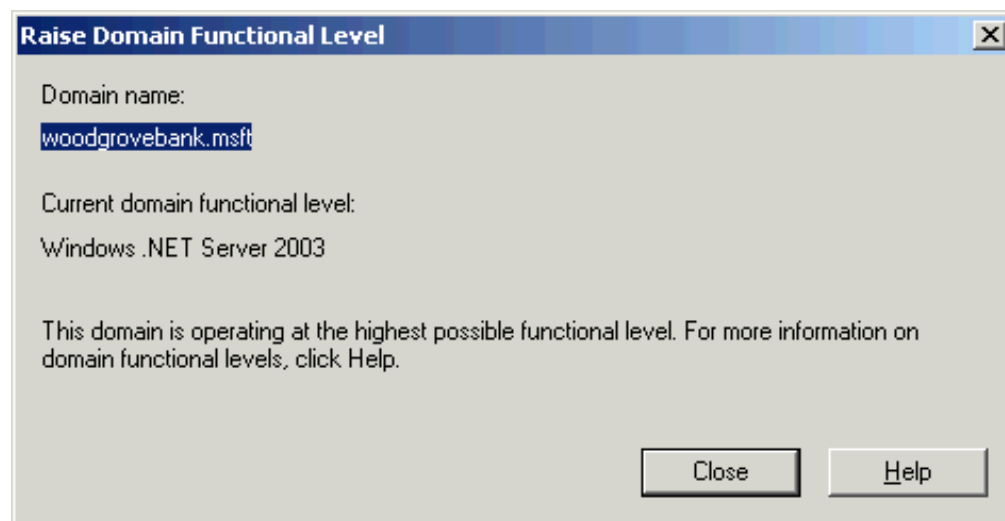
Use Active Directory Users and Computers to ensure that the BDC's account is available and not locked out. Try logging on to the domain from the BDC's console; the BDC must first log on to the domain in order to process any domain user logons. If you can successfully log on to the domain from the BDC console, the BDC's domain account is probably fine. If you cannot or if the BDC's event logs contain domain logon errors, you might need to reset the BDC's domain account, which is a particularly tricky task with a BDC.

 For more information about how to reset the BDC's domain account, see Question 2.

 BDC domain accounts might appear as users rather than computers; see the Microsoft article "HOW TO: Create a Computer Object in the Active Directory for a Windows NT 4.0 BDC" for more information.

### **Domain Mode or Functional Level**

Hopefully, your Win2K domain is running in mixed mode or your Windows Server 2003 domain is running in its Win2K mixed mode functional level. In Windows Server 2003, check the functional level by opening Active Directory Users and Computers, right-clicking the domain, and selecting Raise Domain Functional Level from the context menu. The resulting window will show you the current level (see Figure 18.2). Any level other than Win2K mixed mode will result in NT BDCs being unable to replicate. There is no solution for this problem other than decommissioning your NT BDCs or restoring every Win2K (or Windows Server 2003) domain controller from a backup—effectively performing forest-wide disaster recovery.



**Figure 18.2:** The domain's functional level.



Generally, Windows Server 2003 and Win2K will warn you if you attempt to change the domain mode or functional level and NT BDCs still exist in the domain. Windows Server 2003, in fact, will attempt to stop you outright. However, if you do change the mode or functional level, it's a one-way operation that can't be undone; any remaining NT BDCs will be useless.

### **BDC Logs Events 3210, 7023, or 8032**

If you inadvertently configure a Win2K domain to restrict anonymous connections, NT BDCs might be unable to locate a domain controller, replicate the domain database, start the Net Logon service, and so forth. Although restricting anonymous access to the domain is a valuable security technique (and is included in Microsoft's Hisecdc.inf security template), it can cause problems for NT BDCs. You can correct the problem with a registry edit or decommission the NT BDC.



For more information about this problem and its solution, see the Microsoft article "The Net Logon Service of a Windows NT 4.0 BDC Does Not Function In a Windows 2000 Domain". The article "How to Use the RestrictAnonymous Registry Value in Windows 2000" describes more about the RestrictAnonymous registry setting, which is included in the Hisecdc.inf security template.

## **Q.19: How does Active Directory replication work?**

**A:** Active Directory (AD) is a multi-master directory, meaning each directory services server—referred to as a *domain controller*—contains a fully readable and writable copy of the directory services database. Because all domain controllers can accept changes to the database, some method is needed to replicate those changes to other domain controllers, ensuring a consistent database across all domain controllers. This scheme is referred to as AD replication.

AD replication can be broken down into four basic operational components:

- *Who*, which is a list of servers that participate in replication and the servers with which they replicate. Referred to as a *replication topology*, this list is generated by a special AD component called the Knowledge Consistency Checker (KCC).




For information about how the KCC works, see Question 16.

- *What*, which is the information that is being replicated. AD uses attribute-based replication and versioning to determine which information has changed and requires replication.
- *When*, which is a schedule that determines when replication will occur. Separate schedules exist for replication within an AD site and for each link connecting different sites.
- *How*, which defines how the replicated data is physically transported across the network.



### ***Deciding What to Replicate***

AD maintains multiple attributes for each object. For example, a user object has attributes such as password, account lockout status, user name, and so forth. Each attribute is versioned independently, letting AD replicate only attribute changes. For example, when a user changes his or her password, only that particular attribute receives a new version number and is replicated to other domain controllers (rather than replicating the entire object).

 Windows Server 2003 uses *linked value replication* to improve replication efficiency. Security groups are an example of a multivalued attribute with linked values, in which the group's multiple attributes are references to the group's members. In Windows 2000 (Win2K), changes made to a member of a group require the entire group to be replicated. In Windows Server 2003, only the group member that has changed is replicated. This feature is only available when the forest functional level is at Windows Server 2003.

### **Version Control**

Obviously, some form of version control is necessary to determine which domain controller has the most recent version of each attribute. AD uses *update sequence numbers* (USNs) as a form of version number. Each USN is a 64-bit number; whenever a domain controller changes an attribute, it increments the domain controller's USN. Each domain controller maintains only a single USN.

Each domain controller also maintains a list of USNs that have been received from other domain controllers during replication. When a domain controller informs its replication partners that changes are available for replication, its partners respond with the USN previously received from that domain controller. The sending domain controller can then send only the attributes that have changed since that USN, ensuring that data already replicated isn't replicated again. This technique also allows for domain controllers to remain offline for a period of time; when they replicate the next time, they'll receive every change that's been made since the last time they replicated. Figure 19.1 illustrates this process.

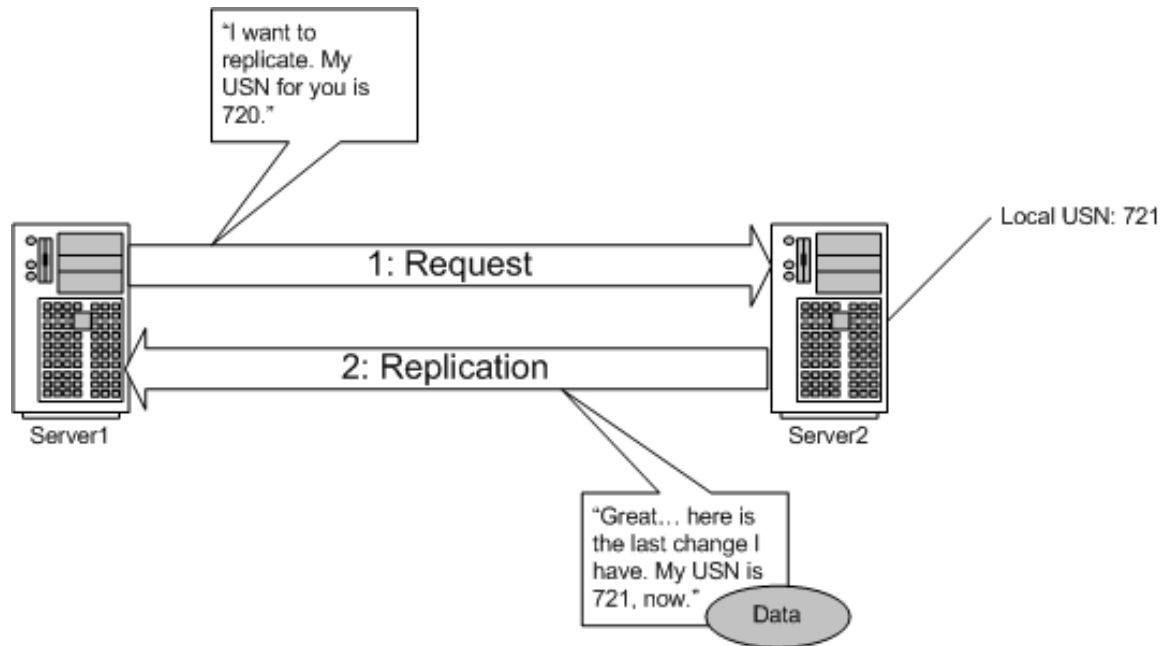


Figure 19.1: USNs and replication.

Dealing with conflicts is also important because it's possible for a single attribute to be changed on two domain controllers at once (for example, two administrators might attempt to change the same user's password at the same time). A replication *collision* occurs when a single attribute is changed on one domain controller, while a previous change to that attribute is still in the process of replicating. To help resolve collisions, AD maintains a *property version number* for each attribute in the directory. USNs are server-specific, whereas property version numbers are attached to each attribute in the directory and are replicated along with that attribute.

Changing an attribute increments its property version number. *Replicating* an attribute does *not* change its property version number; only "new" changes have that effect. Whenever a domain controller receives a replicated attribute with the *same version number as the local copy*, the domain controller knows a collision has occurred. In other words, another source has changed the attribute but that source hadn't first received a replicated change from a second source. The result is two changes running around the network with the same property version number. When this occurs, the domain controller receiving the change keeps the one with the latest timestamp. This situation is the only instance in which AD replication relies on timestamps and is the reason that Win2K and later includes a time synchronization service. Figure 19.2 shows how a replication collision is handled.

✎ It's possible for an attribute to be replicated with a *lower* property version number. This situation can happen when the domain controller making the attribute change has missed two or more replications of that attribute from other sources. AD discards any replicated data with a lower property version number than the local copy.

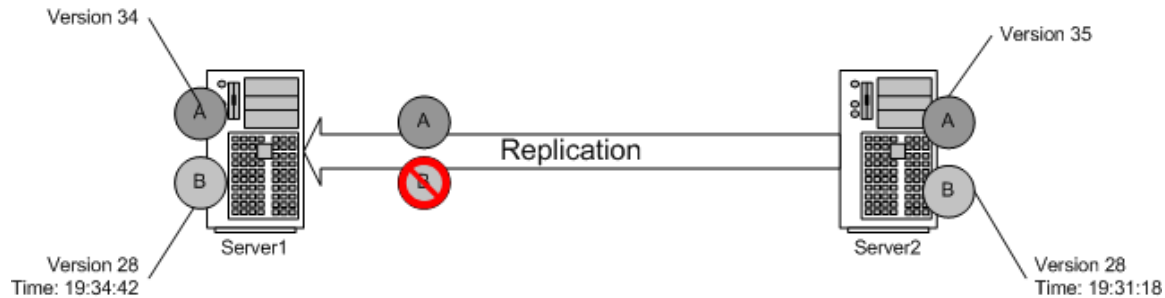



Figure 19.2: Replication collisions.

 For more information about how AD handles replication conflicts, see the Microsoft article “How Conflicts Are Resolved in Active Directory Replication.”

## Propagation Dampening

The AD replication topology allows loops—particularly within a site in which a ring-based topology is the norm. In theory, these loops could result in change being replicated infinitely around the loop. However, AD replication has a built-in *propagation dampening* scheme, which detects and stops looping replication data.

The dampening system uses *up-to-date vectors*. The vector is a list of servers within the site and a USN for each server. The vector at each server thus indicates the highest USN of new changes received from each server within the site. When a new replication cycle starts, the requesting domain controller sends its up-to-date vector to its sending replication partner. That partner, as already described, filters the changes sent to the receiving domain controller so that it only receives changes made after that USN was used. If the requesting domain controller sends a USN that is greater than or equal to the sending domain controller’s own USN, then no changes need to be sent, and the replication cycle stops. Figure 19.3 shows the vectors in action.

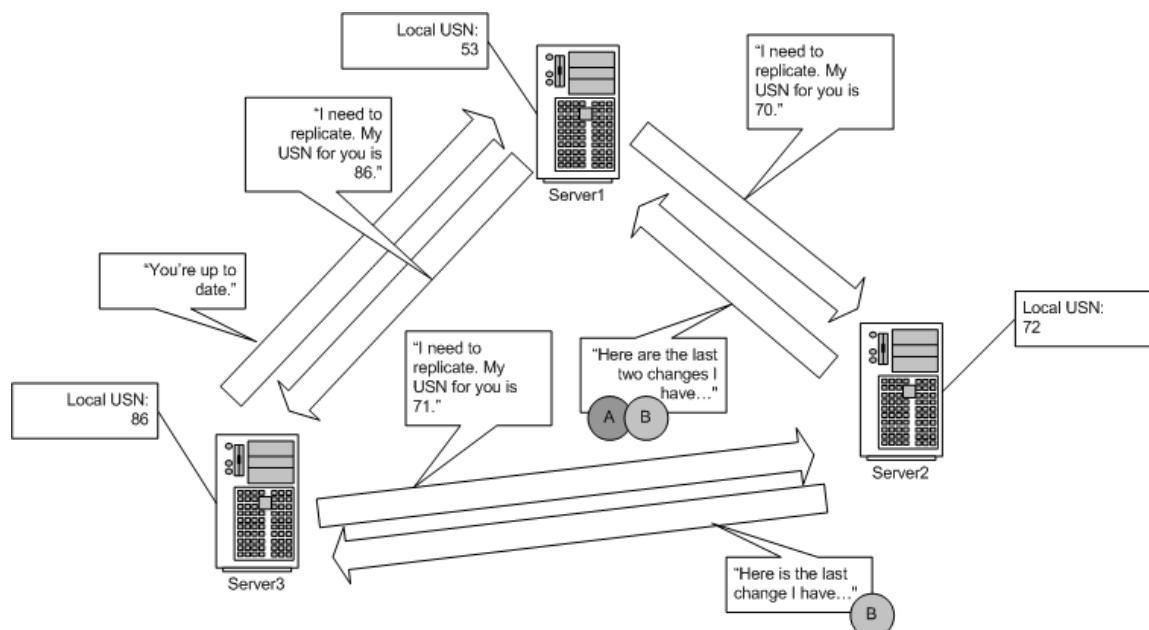



Figure 19.3: Propagation dampening in AD replication.

## When Replication Occurs

For intrasite replication, each domain controller informs its partners when it has changes available. Those partners then send a request for those changes, providing the domain controller with the USN numbers from their last replication update.

Domain controllers don't always inform their partners of changes as soon as those changes occur. For most changes, domain controllers wait about 5 minutes before sending a notification. This time period allows an administrator to make several changes and have them replicated in one batch, rather than the domain controller sending out independent change notifications for each minor change. However, certain types of security-sensitive changes—such as the lockout status of a user account—are considered high-priority and are always replicated immediately.

 For more information about high-priority replication triggers, see the Microsoft article "Urgent Replication Triggers in Windows 2000."

 You can modify the default intrasite replication interval. For details, refer to the Microsoft article "How to Modify the Default Intra-Site Domain Controller Replication Interval."

Intersite replication can occur less frequently, according to a schedule that you specify. This flexibility allows you to configure replication to best utilize your available intersite WAN bandwidth. Intersite replication traffic is compressed somewhat, helping to reduce WAN bandwidth utilization. Figure 19.4 shows how you can alter the schedule for a site link, for example, to prevent replication during business hours.

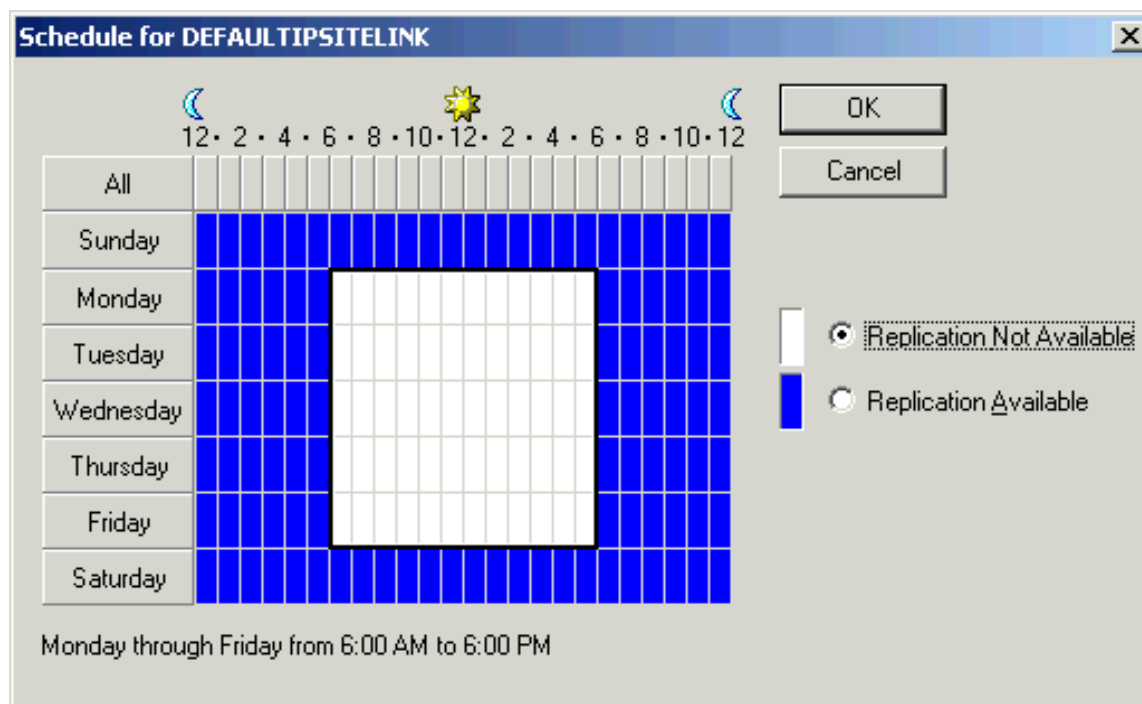


Figure 19.4: Restricting replication on a site link.


## **How Replication Travels**

Replication can use either IP or Simple Mail Transport Protocol (SMTP) as a transport. IP is the standard intrasite transport and the default intersite transport. Replication actually uses IP to carry remote procedure call (RPC) traffic, uses Kerberos to mutually authenticate both replication partners, and uses Microsoft's CryptoAPI to encrypt the replicated data for maximum security.


The SMTP transport packages replicated data in packets, which allows for disconnected sites that have only mail-based connectivity. In effect, domain controllers email each other replication data. SMTP replication can only be used between sites, and can only be used between domain controllers in different domains (which still replicate certain forest-wide information to one another). SMTP requires the use of an enterprise certificate authority (CA), which allows the domain controllers participating in replication to authenticate one another through a trusted root CA.

## **Q.20: How can I check the replication topology for problems?**

**A:** Active Directory (AD) replication is dependent entirely upon an accurate *replication topology*, a map of which domain controllers will exchange replicated changes with one another. The topology is generated by the AD Knowledge Consistency Checker (KCC), which generates a topology both for intrasite and intersite replication.

 For details about how the KCC works, see Question 16.

Generally, the first symptom of topology problems is when one or more domain controllers, or an entire site, fail to replicate AD changes. You might, for example, notice that user password resets aren't being replicated properly or that new users and groups aren't consistently available throughout the domain. Often your first response to this problem is to use the Active Directory Sites and Services console to force replication; but doing so is useless if the underlying replication topology isn't correct.

 For instructions about forcing replication, see Question 17.

Troubleshooting topology issues requires a methodical approach. If possible, start by determining whether you're dealing with an intersite or intrasite topology problem, as you'll need to troubleshoot them separately. To help make that determination, connect to a specific domain controller by using Active Directory Users and Computers. Make a change to AD, such as creating a new user group or organizational unit (OU). Check to see whether the change appears on another domain controller within the same site and within another site. Keep in mind that intrasite replication can take as long as 5 minutes or so to complete under normal conditions. Intersite replication is dependent upon the replication schedule you configured for the site links.

## **Intersite Replication**

When intersite replication seems to be a problem, check the obvious first:

- Make sure your site links are configured with the correct replication schedule. If you have multiple site links between two sites, check the schedule on each. It's possible that one link has failed, and that AD was switched to an alternative link that uses a different schedule.
- Check the network connectivity between the two sites to make sure that your network isn't creating the problem.

Next, figure out which domain controllers are acting as the bridgehead servers in each site. Keep in mind that each domain in each site will have at least one designated bridgehead server. Sites with connections to multiple other sites might have multiple bridgehead servers per domain, with each bridgehead handling connections to another site. You can find the bridgehead servers by looking at the connection objects in Active Directory Sites and Services, and noting which domain controller is configured with connection objects to a domain controller in another site.

If you can't find a connection object on any domain controller in one site to a domain controller in another site and both sites contain domain controllers in the same domain, then you have a topology problem. Troubleshoot the intersite topology generator (ITSG).

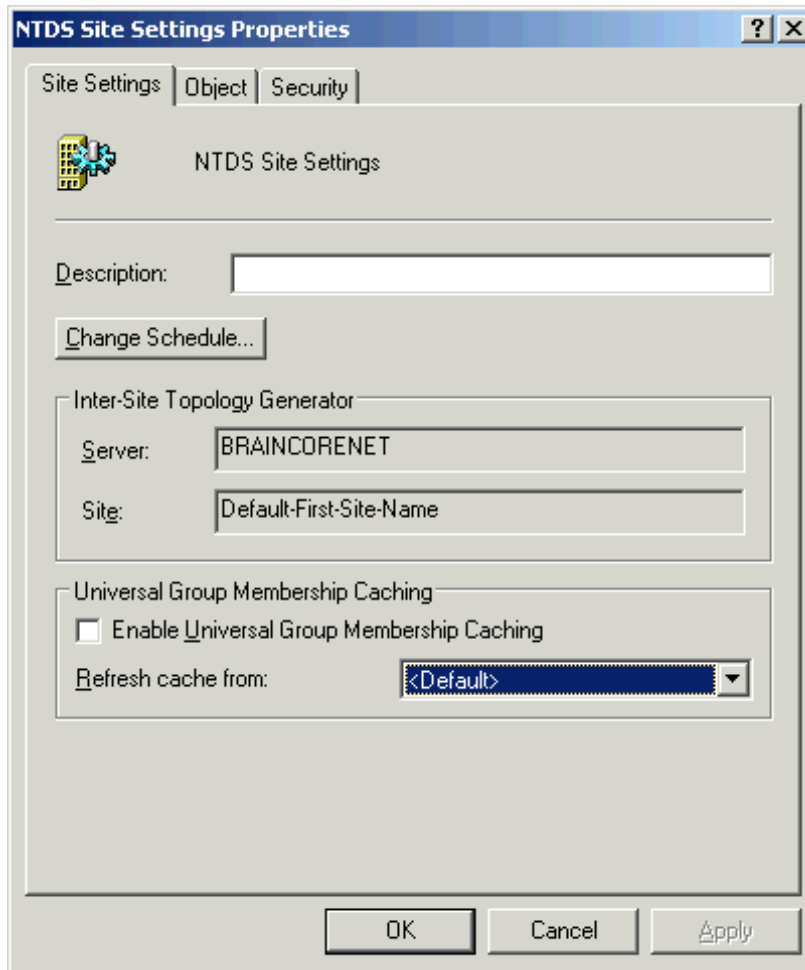
On each bridgehead server, ensure that you can ping the bridgehead servers at the other site(s). If you can't, a network issue is causing the problem and must be resolved.

If network connectivity between the bridgehead servers is OK, it's possible that the bridgehead servers aren't able to handle the replication traffic. Although this situation is rare, you can test the theory by manually creating connection objects to different domain controllers, thus creating new bridgehead connections. Delete the automatically configured bridgehead connection objects. If this action resolves the problem, the former bridgehead domain controllers are at fault and should be examined for functionality and capacity.

If no steps thus far have solved the problem or indicated a possible cause, you might have a serious replication problem. Check the System and Security event logs on your domain controllers for any events that might provide clues to the source of the problem.

## **ITSG**

Each site has a designated ITSG, which generates the topology for that site. You can discover which domain controller is the ITSG by using Active Directory Sites and Services. To do so, open the console, select the site in question, and in the details pane, right-click NTDS Settings. As Figure 20.1 shows, you'll see the name of the server acting as ITSG.



**Figure 20.1:** Server **BRAINCORENET** is the ITSG for this site.

After you've located the ITSG, ensure that it's functioning properly (services are all started and you can log on) and connected to the network. Next, force it to regenerate the intersite replication topology by running

```
repadmin /kcc
```

from the server's console. If the new topology doesn't solve your problem, consider taking the domain controller offline or restarting it. AD will choose a new ITSG within about an hour, and the new domain controller might have better luck generating the correct topology.



### **Intrasite Replication**

The intrasite replication is generated by the KCC service running on each domain controller in the site. Intrasite replication generally occurs automatically and constantly throughout each domain in the site.

If a particular domain controller (or domain controllers) within the site don't seem to be replicating properly, check its replication partners. You can do so by running

```
repadmin /showreps
```

at each domain controller's console, or running

```
repadmin /showreps servername
```

from a single console, providing the appropriate *servername*.

Document the incoming and outgoing partners for each involved domain controller, and ensure that the domain controller has proper network connectivity (such as ping) to its replication partners. If network connectivity between any domain controller and one or more of its replication partners isn't available, troubleshoot and resolve that problem.

If network connectivity appears to be OK, try to force each involved domain controller to generate a new replication topology by running

```
repadmin /kcc
```

on each domain controller. This process normally occurs automatically every so often, but it's possible that a new topology problem hasn't yet been corrected by the automatic process.

If a new topology doesn't correct the problem, try restarting each domain controller involved. If that doesn't fix the problem, you have a more serious AD replication issue that does not involve the replication topology; check the System and Security event logs for messages that provide clues as to the source of the problem.

## **Q.21: How does DNS work?**

**A:** The Domain Name System (or Service, depending on who you listen to—DNS) is one of the most important components of modern networks, including the global Internet. Without it, you couldn't type [www.Microsoft.com](http://www.Microsoft.com) into your Web browser; you'd have to type a difficult-to-remember IP address instead. DNS saves the day by translating human-friendly names into computer-friendly IP addresses. It actually does much more than that—providing computers with critical information about network services such as the locations of mail servers, domain controllers, and more.

### **The Process**

In any DNS transaction, at least two computers are involved: the DNS server and a DNS client. In most networks, there is often also a second DNS server known as the *forwarder* as well as additional *authoritative* DNS servers. Consider an example: You're on your company's internal network, which is equipped with a DNS server. That server is set up to forward to your ISP's DNS server, and you're trying to surf to [www.Microsoft.com](http://www.Microsoft.com). Here's what happens:



1. Your client sends a DNS request packet, usually via User Datagram Protocol (UDP), to the local DNS server.
2. That DNS server has no clue what the IP address for `www.Microsoft.com` is, so it sends the entire request to your ISP's DNS server—a process known as forwarding.
3. The ISP's DNS server also has no clue. However, it is configured with root hints, meaning it knows the IP address of a DNS server that is authoritative for the “.com” top-level domain (TLD).
4. The ISP's DNS server contacts the .com TLD server and asks for a reference for the Microsoft.com domain. This action is called a recursive query. Your own DNS server could have performed this type of query, but it's more common for businesses to forward requests to their ISP's DNS server.
5. The TLD server, being authoritative, has records for every .com domain in existence (actually, there are a bunch of load-balanced DNS servers handling that TLD, but let's pretend it is one big server for now). So the TLD server sends back the IP address of a DNS server that's authoritative for the Microsoft.com domain. Now the DNS server knows how to contact Microsoft.com, so it just needs to figure out the “www” part.
6. The ISP's DNS server then contacts the Microsoft.com DNS server. In all likelihood, that server is owned by Microsoft and contained in one of Microsoft's data centers (it's also probably part of another load balanced set of DNS servers because Microsoft.com gets so much traffic). The ISP's DNS server asks for the address of `www.Microsoft.com`.
7. As an authoritative server for the Microsoft.com domain, the Microsoft.com DNS server has a record for the host named `www`. So it sends that host's IP address back to the ISP's DNS server.



If the Microsoft.com server didn't have a record for a host named `www`, it would send back a negative reply. As an authoritative server, it *by definition* has records for all hosts in the domain; in other words, if that DNS server doesn't know about the host, the host doesn't exist.

8. The ISP's DNS server sends the IP address for `www.Microsoft.com` to your local DNS server.
9. The local DNS server sends the IP address to your client.
10. Your client now has the IP address, and can contact `www.Microsoft.com` directly.

Now, there's a bit of extra subtlety in there. Each DNS server along the way will try to cache previous lookups. With a popular site such as `www.Microsoft.com`, it's likely that your local DNS server, or certainly your ISP's DNS server, will have already been through this process once. DNS servers will use their cached results from earlier attempts whenever possible to avoid having to go through this entire process every single time a request comes through.

## DNS Replies

DNS replies can often contain multiple responses. For example, if your client queries my.yahoo.com, you'll get back a response with at least two IP addresses. Your client is free to try either one, although it'll usually try the first one. A DNS technique called *round robin* can be implemented on the server; when more than one record exists for a single host, the server will rotate the order of the IP addresses in outgoing responses. This behavior helps direct clients to different servers in a Web farm, for example, more evenly spreading the workload. Figure 21.1 shows a Network Monitor capture of a DNS response in which you can clearly see two IP addresses for my.yahoo.com: 216.115.105.17 and 216.115.105.16.

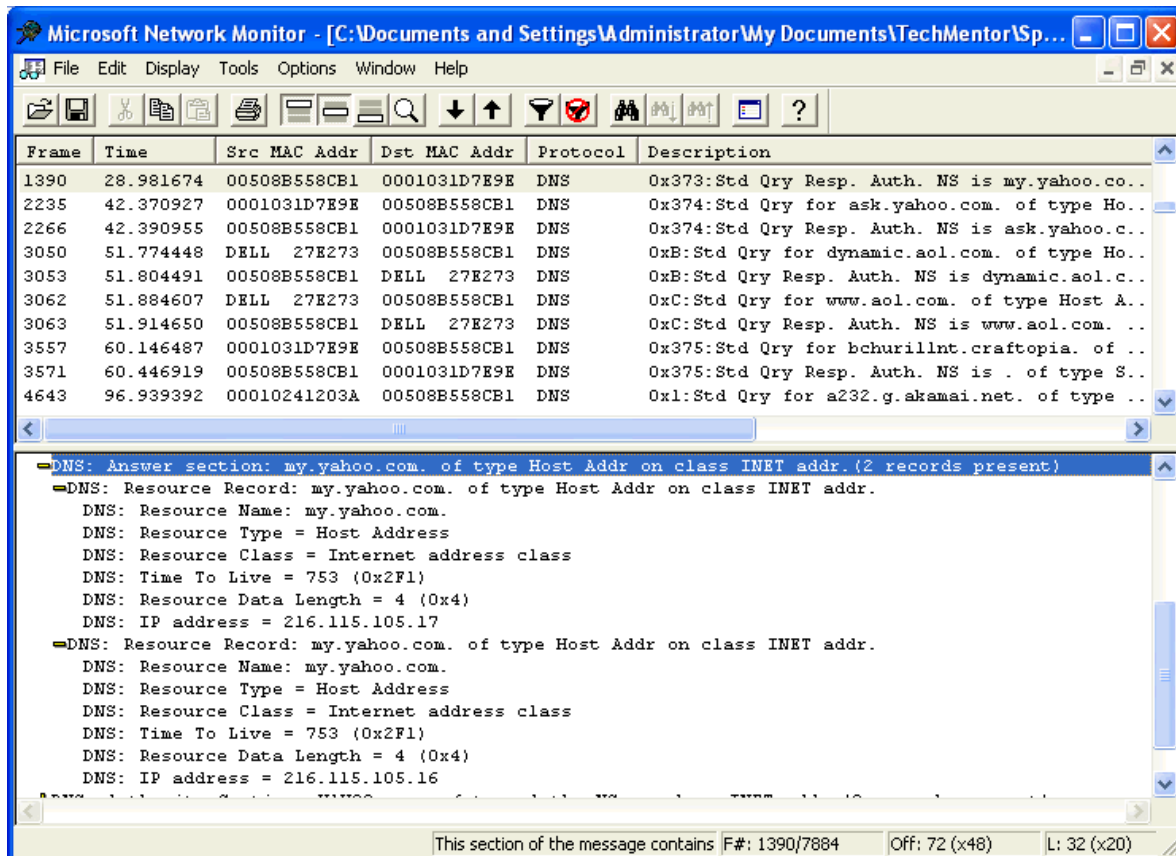


Figure 21.1: Multiple replies for my.yahoo.com.

## **Record Keeping**

DNS keeps track of information in *zones*. Essentially, a zone is a flat-file database for a particular domain, such as `www.Microsoft.com`. The zone can contain different record types, all of which can be queried by clients:

- **A**, which is a Host Address record—This resolves a single host name, such as `www`, to an IP address.
- **CNAME**, or Alias—This resolves a name such as `www` to an actual host name, such as `www1`. Think of it as a nickname for a computer—“`www`,” for example, is easier to remember and more standardized than a computer name like “`w4laswin`,” which is what a Web server’s real name might be.
- **MX**, or Mail Exchanger—This provides the name of the mail server for a domain. Multiple MX records can be provided for fault tolerance or load balancing and a priority assigned to each. Clients, such as sending mail servers, will attempt to contact the server in the MX record with the lowest-numbered priority.
- **AAAA**—This maps an IPv6 IP address to a host name.
- **SRV**, or Service—This provides the IP address of one or more servers providing a particular service. AD uses SRV records to allow clients to locate domain controllers, among other things.
- **SOA**, or Start of Authority—This special record indicates that the DNS server hosting the zone is authoritative for the zone and is the primary source of name resolution for hosts within that domain.

## **Sharing Zones**

It’s common for companies to have more than one DNS server for their domains. In fact, the rules for DNS say that each domain must be hosted by at least two DNS servers for fault tolerance. However, having multiple DNS servers can be a configuration nightmare, because you would have to update records in multiple locations.

Rather than creating this maintenance nightmare, DNS servers can be configured to host *secondary zones*, which are essentially read-only copies of a *primary zone*. Configuration changes can be made in the primary zone, then transferred, or replicated, to the secondary zones through a process known as a *zone transfer*.

## **IP Address to Name**

Sometimes computers need to resolve an IP address to a computer name. For example, when two SMTP email servers communicate, the receiving server might want to verify that the sending server is, in fact, really coming from the domain it says it is. The server can use DNS to perform a *reverse lookup*, attempting to translate the sending server’s IP address into a host name and domain name.

Reverse lookups are provided through special, independent *reverse lookup zones*, which contain *PTR records*. PTR, or pointer records, resolve an IP address into a computer name. Like regular *forward lookup zones*, you can create primary and secondary reverse lookup zones for load balancing and redundancy.

## Communications

Most DNS communications—queries and replies, at least—are conducted over connectionless UDP transmissions on UDP port 53. UDP does not provide packet confirmation, so if a request gets dropped in transit, the client will never know. Clients therefore wait for a fixed period of time—called a *timeout* period—before assuming the packet was lost and retransmitting it. Most clients will try three times, then assume that their DNS server has failed.

Zone transfers, which are usually large and involve hundreds or thousands of packets in sequence, are conducted over TCP port 53. TCP requires a bit more processing time to set up, but provides packet confirmation and allows for single-packet retransmissions if necessary.

The DNS specification makes it clear that UDP communications are intended to be single packet, and DNS requests are therefore limited to 256 bytes over TCP. This limitation is normally sufficient for any query. However, some replies with multiple hosts can exceed 256 characters. Although the DNS specification is unclear how these should be handled, in practice, the server will set up a TCP communications channel to send the response. In these cases, TCP port 53 is used. Clients listen for replies on both UDP and TCP port 53, ensuring clear communications.

## Data Storage

Most DNS servers, including nearly all UNIX-based servers, store their DNS records in *zone files*, which are essentially text files. In fact, many UNIX DNS servers are simply background daemons (services), with no UI whatsoever. Changing DNS records requires you to edit the text file, then restart (or *hup* in UNIX parlance) the service, which rereads the updated file.

Microsoft DNS running on a Windows 2000 (Win2K) domain controller can convert a standard, file-based zone to an AD -integrated zone. In this type of zone, records are stored in the AD database and replicated to all domain controllers. Any domain controller running the Microsoft DNS service is therefore a DNS server. Technically, because all domain controllers have full read/write access to the AD database, all DNS servers in an AD -integrated zone are said to host a *primary zone* because they can all make changes to the zone. Changes made on one server are replicated through AD to the others.

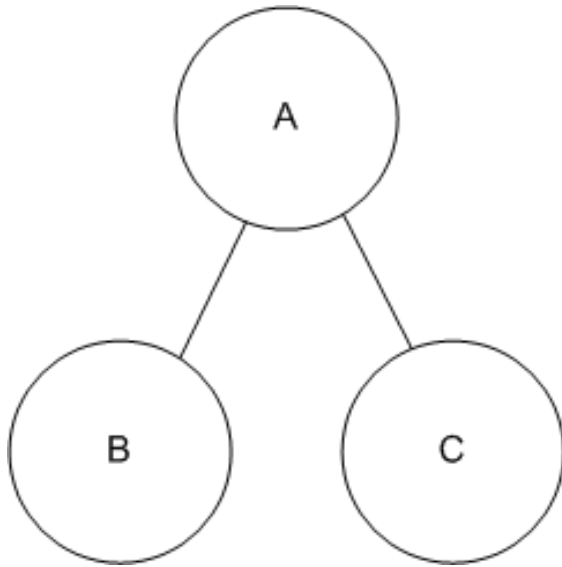
 For more information about AD replication, see Question 19.

## Q.22: How do trusts work under AD?

**A:** Trusts didn't go away when Windows 2000 (Win2K) came onto the scene—they just changed a lot. Actually, Active Directory (AD) continues to support Windows NT-style trusts, and for good reason.

### Win2K Intra-Forest Trusts

First, a quick review of the primary type of trust—the one that exists between parent and child domains in the same forest. As Figure 22.1 shows, every child domain has a two-way, automatic, nonconfigurable trust with its parent. This trust is transitive, meaning that, in this example, domains B and C trust one another through their trust with their parent, domain A.

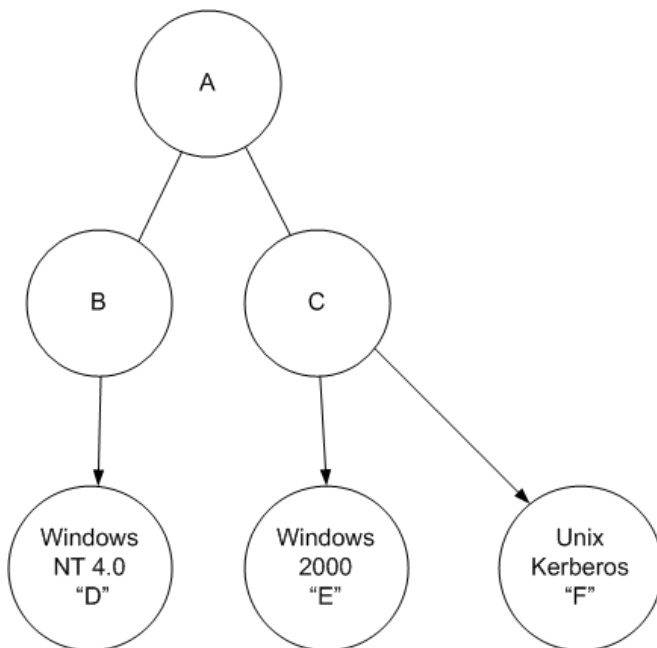


**Figure 22.1: Native Win2K AD trusts.**

It's important to remember that trusts don't necessarily confer any special privileges to any users; it simply makes it possible for user accounts in one domain to be granted permissions to resources in another domain. The exception is the Enterprise Admins group, which has special, full administrative control over every domain in the forest.

### **Foreign Trusts**

An AD domain, not a forest, can establish a manual, one-way, non-transitive trust with another AD domain, a Kerberos domain, or an NT 4.0 domain. Figure 22.2 shows an example.




**Figure 22.2: Foreign AD trusts.**

In this example, Domain B trusts Domain D, meaning Domain B can assign permissions on resources to user accounts contained within Domain D. The relationship is not reciprocal; if Domain D wants to utilize accounts from Domain B, a separate, independent D-trusts-B trust will have to be created.

In addition, these trusts are not transitive. Although Domain C trusts Domain E, it does not follow that Domain B also trusts Domain E even though Domain B trusts Domain C. If Domain B wants to utilize user accounts that exist in Domain E, a separate trust will have to be established in which Domain B trusts Domain E.

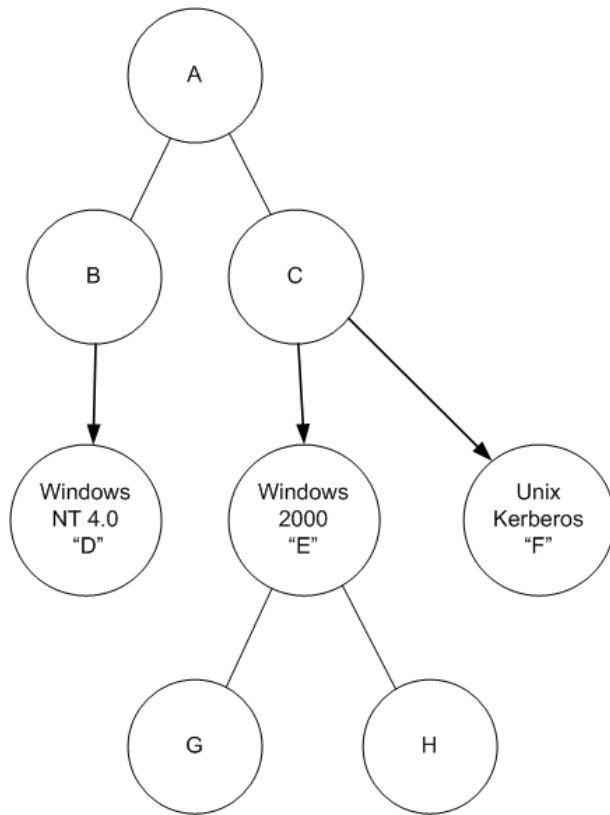
The trust relationship, especially the ability to obtain security IDs (SIDs) for user accounts in the trusted domain, lies with the domain controller holding the PDC emulator role in AD domains, with the PDC in NT domains, and with an individual Kerberos server for Kerberos realms. The inter-domain communications are critical and are encrypted. The encryption key is a shared secret, or password, which you create when you establish the trust. Actually, as soon as the trust is established, the two domains negotiate a new password, which they then change on a periodic basis. One problem in trust relationships is when the two domains lose sync of the trust password. In that case, you'll have to delete and re-create the trust. Doing so will not generally affect the permissions assigned to users from the trusted domain, provided you repair the trust quickly.

 You can use the Netdom command-line tool from the Support Tools to verify, delete, and create trusts in a Win2K domain.

Note that the trust relationship between Domain C and Domain E in this example does *not* necessarily make Domain E a member of the forest containing Domain A, Domain B, and Domain C. New domain trees can be added to an existing forest, but you make this decision when you install the domain controller. Domain C and Domain E, in this example, do not necessarily share a common AD schema because they are separate forests.

### ***Trusting Trees***

It's possible for a domain in one tree to trust a domain in another tree, as Figure 22.3 illustrates.



**Figure 22.3: Two trees with inter-domain trusts.**

In this example:

- Domain A trusts Domain B, and Domain B trusts Domain A
- Domain A trusts Domain C, and Domain C trusts Domain A
- Domain B trusts Domain C, and Domain C trusts Domain B
- Domain C trusts Domain E
- Domain E trusts Domain G, and Domain G trusts Domain E
- Domain E trusts Domain H, and Domain H trusts Domain E.
- Domain G trusts Domain H, and Domain H trusts Domain G.

In other words, the normal two-way transitive trusts operate within each tree, and one domain from each tree trusts a foreign domain. The manual trust between Domain C and Domain E is non-transitive and one-way. Again, this configuration has not formed a forest of six domains: there are two forests, with three domains, and one domain tree apiece, and the two forests don't trust one another. Password synchronization occurs between the PDC emulators in Domain C and Domain E.



## Inter-Forest Trusts

New in Windows Server 2003, inter-forest trusts allow two forests to trust one another. These trusts are one-way or two-way, and they take advantage of the transitive nature of intra-tree trusts. For example, Figure 22.4 shows two forests, each with three domains in a single domain tree. The root domains in each forest trust one another.

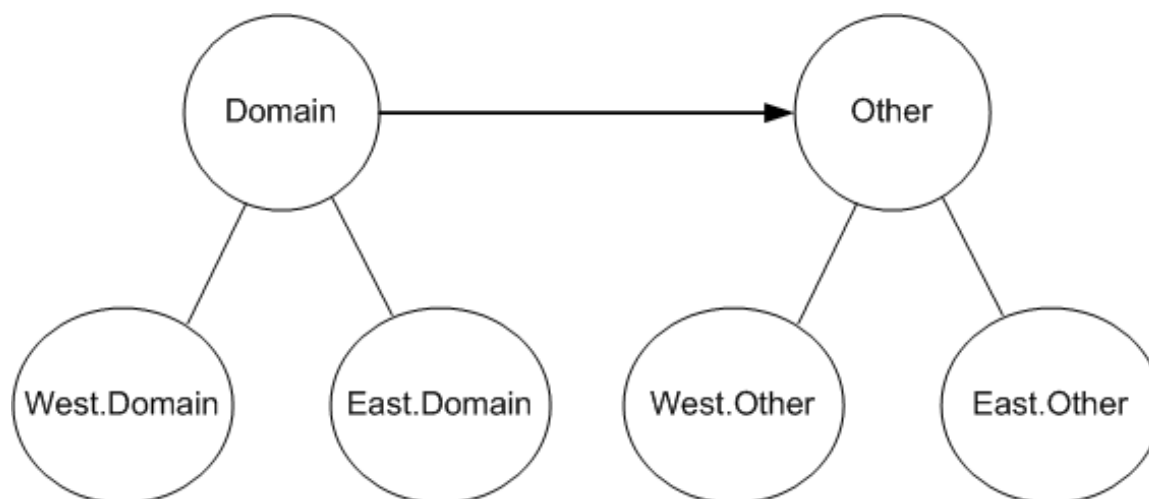



Figure 22.4: Inter-forest trust.

This feature is only available in AD domains in which all domain controllers are running Windows Server 2003 and the forest functional level has been raised to its highest level, Windows Server 2003 Native.

In this example, Domain, West.Domain, and East.Domain all trust Other, West.Other, and East.Other. The trust, then, is transitive. You can also configure a two-way trust, which in this example, would result in all six domains trusting one another. Forest trusts can use *SID filtering*, which prevents specific SIDs in one domain from being used in another domain.

 New in Windows Server 2003: Forest trusts were created to acknowledge the power of the Enterprise Admins group. Prior to forest trusts, Microsoft often stated that domains were the primary security boundary. In fact, as a result of the forest-wide rights of Enterprise Admins, the forest was the only true security boundary. Many enterprises created separate forests to resolve political issues about who controlled which resources; forest trusts give these organizations a way to provide easier cross-forest access to their users, while restricting control of the Enterprise Admins group in each forest. SID filtering can be used to ensure that foreign Enterprise Admins have no control in a forest, and they have no rights in a foreign forest by default.

Forest trusts are considered to be an external trust, and as with any other external trust, there's no implicit synchronization of data. For example, the Domain and Other domain trees do not have to share a common AD schema, as they would if both trees belonged to the same forest. There's also no synchronization of address lists or directory objects of any kind, and Global Catalog (GC) servers in each forest maintain independent catalogs. If synchronization of some kind is desired, check out Microsoft Metadirectory Services (MMS), which can be used to synchronize objects across multiple directory services or domains.



## Q.23: I've heard the term *split brain DNS*; what does it mean?

**A:** Most companies these days have a public Internet domain name, such as [www.Microsoft.com](http://www.Microsoft.com). When companies begin implementing Active Directory (AD), which requires a DNS name for the domain name, they have to make a decision: use a private domain name (such as [Microsoft.pri](http://Microsoft.pri)) for the AD domain, or keep using the public name ([Microsoft.com](http://Microsoft.com)) for both AD and public use.

The benefit of using a private domain name is that your domain is safer from harm. You have two domains, a public one and a private one. The downside is that users might think it's weird. For example, in AD, users can log on using their fully qualified user name; [don@braincore.net](mailto:don@braincore.net), for example. That's easy for users because it looks like their email address. However, if you select a private domain name, users will receive email at [don@braincore.net](mailto:don@braincore.net), but they'll log on with [don@braincore.pri](mailto:don@braincore.pri), which is confusing.

So-called *split brain DNS* offers an elegant solution that combines the benefits of a private domain name with the benefits of a public domain name—with almost none of the downsides of either.

### **Splitting the Brain**

Figure 23.1 shows how split DNS can be configured so that both your internal and public domains use the same domain name, [company.com](http://company.com).

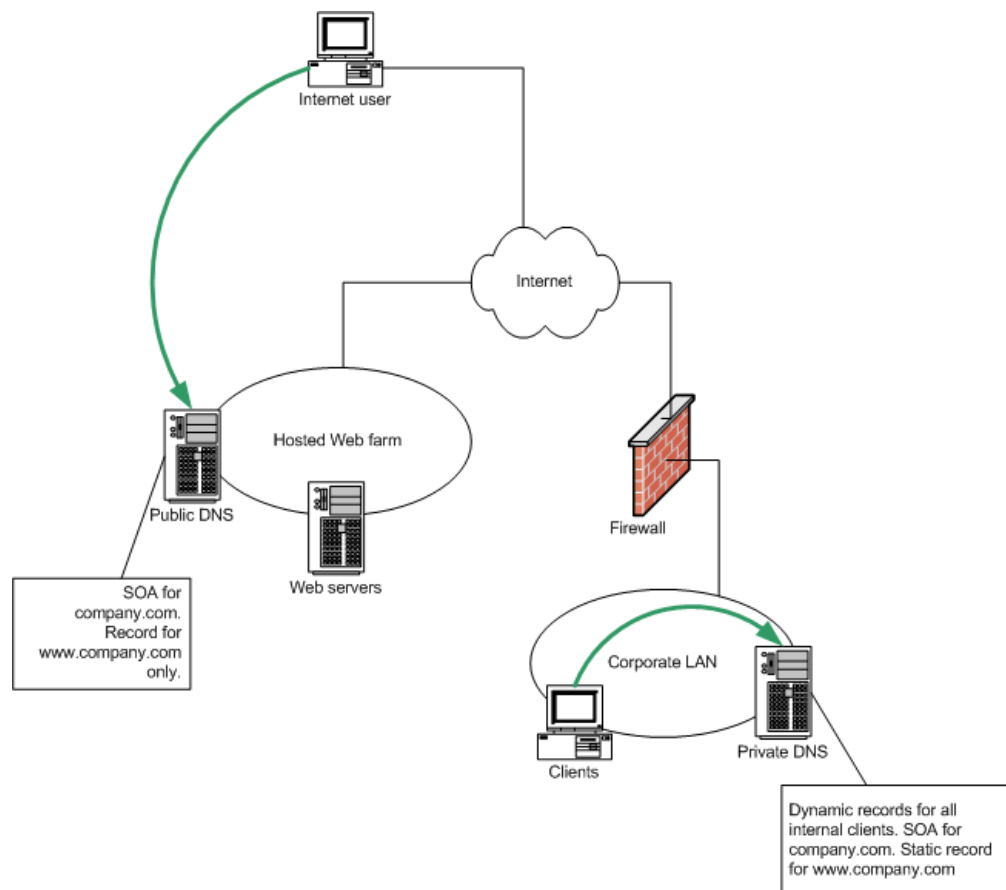


Figure 23.1: An example split DNS configuration.

In this example, you have two DNS servers: one on your internal network, and one that's accessible to the public (you'd likely have at least two in each location, but that's not relevant to the basic configuration). Both DNS servers believe they're authoritative for your domain, meaning that they are the final source for all records related to the domain.

 For more information about authoritative servers, see Question 21.

### **How it Works**

Your public domain record lists your public DNS server(s), meaning that the top-level domain (TLD) domain name servers will also list your public DNS server(s). When Internet users attempt to look up any host in your domain, such as `www.company.com`, they'll use your public DNS server. That server is configured with static records, including any Web servers, mail servers, and other publicly accessible servers.

Internally, your client computers are configured to use your internal DNS server. It might have dynamic DNS configured and believe it is authoritative for `company.com`. It might also be configured to forward requests to your ISP's DNS server so that your client computers can access Internet Web servers. Hence, the trick—because your Web server isn't connected to your internal DNS server, it won't be able to dynamically register its name. Thus, your internal DNS server won't contain a record for, say, `www.company.com`.

No problem, right? Isn't that what forwarding solves? Wrong. Because your internal DNS server is authoritative for `company.com`, it will never forward requests for anything involving `company.com`. "If I don't have it, nobody else will" is your DNS server's motto in this case. So you will need to create static records for all publicly accessible servers in the `company.com` domain. Doing so will permit internal users to access these external assets.

### **Split-Brain Pitfalls**

Split-brain DNS only has a couple of minor drawbacks. Primarily, you'll need to manually keep both DNS servers up to date with your publicly accessible resources. Technically, the public DNS server could use dynamic DNS, but the security implications of doing so aren't worth the convenience.

You can't perform zone transfers between the servers because they're both hosting a primary zone for `company.com`, and they can only transfer to another *secondary* zone of the same name. You wouldn't necessarily configure them as forwarders to one another, either; remember, authoritative servers never forward requests for their own domains.

### **Same-Site Split-Brain**

In the previous example, the external resources were hosted on a different network. So what if you host your own Web servers? The picture looks remarkably similar, as Figure 23.2 shows.

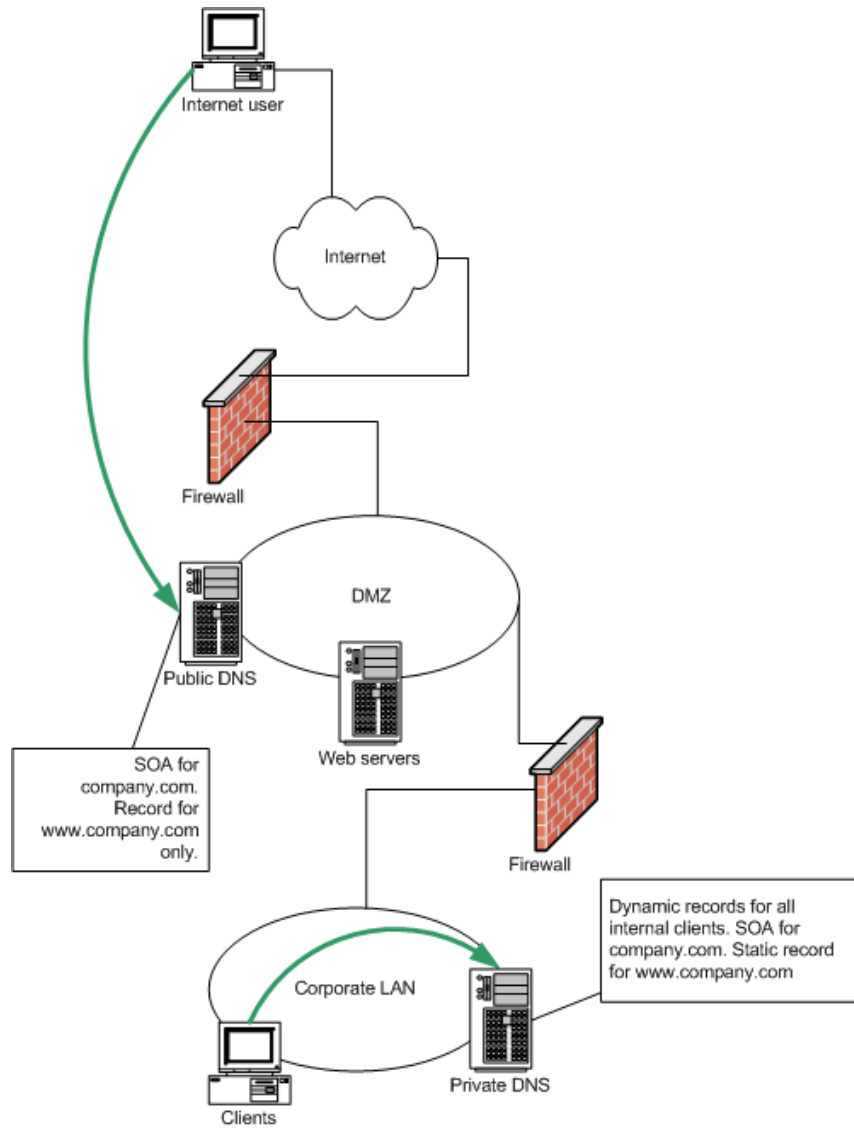



Figure 23.2: Split DNS in a single location.

In this example, a DMZ or perimeter network contains the publicly accessible resources, including the Web server and the public DNS server. The internal LAN contains the internal resources. Close physical proximity or network connectivity doesn't matter for split DNS; the two DNS servers never talk to one another, and each is unaware that the other even exists.

## Q.24: Why won't Active Directory let me create new objects?

**A:** Active Directory (AD) objects must all have an identifier that is unique within the domain. This identifier, called a security ID (SID), is a combination of a domain-wide identification number and a unique, per-object *relative identifier* (RID). Because every domain controller in a domain can create new objects, the possibility exists for duplicate RIDs, which would be a problem. To prevent that, each domain controller is only permitted to issue RIDs from a pool, and that pool is assigned by the RID master, a special Flexible Single Master Operations (FSMOs) role held by one domain controller in each domain.

 For more information about the RID master role, see Question 1.


Normally, when a domain controller runs out of RIDs, it contacts the RID master for a new pool. The RID master ensures that each domain controller receives a unique pool, preserving the uniqueness of the object SIDs in the domain. However, sometimes a domain controller uses up RIDs faster than it can get them. This situation can happen more frequently in a domain that is receiving migrated user accounts, each of which needs a RID. Automated migration tools can create new user accounts more quickly than a domain controller can get more RIDs.

You'll see evidence of this in the Directory Services event log on the affected domain controller. The event ID to look for is 16645, which reads:

The maximum account identifier allocated to this domain controller has been assigned. The domain controller has failed to obtain a new identifier pool. A possible reason for this is that the domain controller has been unable to contact the master domain controller. Account creation on this domain controller will fail until a new pool has been allocated. There may be network or connectivity problems in the domain, or the master domain controller may be offline or missing from the domain. Verify that the master domain controller is running and connected to the domain.

### ***Troubleshooting the Problem***


The event itself gives good advice: Make sure that the RID master is online, and if it isn't consider transferring the role to another domain controller.

 For instructions on transferring the RID master role, see Question 8.


You can verify that the RID pool is the problem by connecting directly to another domain controller and attempting to create a new user or group. If that domain controller is able to create the object, the problem is confined to the domain controller on which the event appeared. Once the RID master is back online, the affected domain controller will eventually request a new pool (there's no way to force the process), and start working again.


### ***Preventing the Problem***

If you know you're going to be using up RIDs quickly, consider expanding the size of the RID pool allocated to each domain controller. Doing so requires a registry hack on the domain controller that's playing RID master: Locate the key `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\NTDS\RID Values`. Modify the RID Block Size value from its default of 0 to a higher number. The default value of 0 is treated internally as 500, which is the default RID pool size.

 Prior to Service Pack 4 (SP4), values beyond 500 are treated as 500, meaning there's effectively no way to increase the pool size. SP4 corrects this problem. However, 500 is still the minimum size you can configure; setting the value to 400 will still be treated as 500.

Be sure to remember your change; if you ever decommission that domain controller or move the RID master role to another one, you will have to modify the new RID master's registry. You can safely modify this value on all domain controllers, if you like, to make this possibility less of a concern.

 For more details, refer to the Microsoft article "RID Pool Allocation and Sizing Changes in Windows 2000 SP4."

 Worried about using up all the RIDs? Don't. It's possible, but there are about 2<sup>30</sup> RIDs available in a domain, which represents millions and millions of objects.


### ***RID Threshold Changes***

Domain controllers are designed to request a new RID pool when they've exhausted 80 percent of their current pool. At a fast rate of consumption, especially when the domain's RID master is across a WAN link (as might be the case when migrating a field office), the remaining 20 percent of the RID pool is used up before the domain controller can snag a new pool.

Under Win2K Server SP4 and later, this threshold was changed to 50 percent. This decreases the chances of RID pool exhaustion becoming a problem, unless you're creating new objects at a truly breakneck pace.

## **Q.25: How can I see why some DNS queries are failing?**

**A:** DNS is a reasonably simple system, although it does have some specific quirks that can make troubleshooting difficult.

 For more information about how DNS works, see Question 21.

One way to troubleshoot DNS problems is to use Network Monitor or a similar packet-capture tool to capture the client's DNS queries. Also look for the DNS server's replies. If your network has a firewall, attempt to capture the requests and replies on *both* sides of the firewall. Figure 25.1 shows a sample DNS communication in Network Monitor; the top pane shows the flow of responses and queries, and the bottom pane shows the detail of one response.

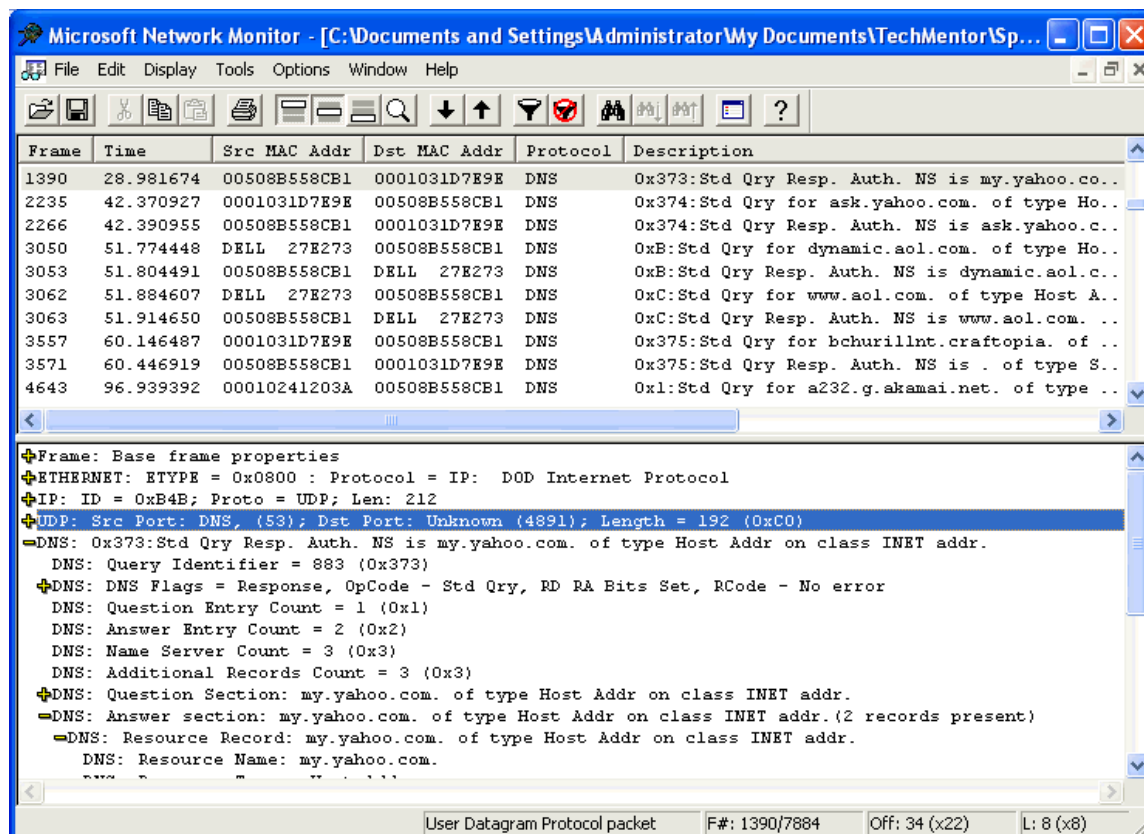


Figure 25.1: Sample DNS query response.

Fortunately, there's not much that can go wrong with DNS, and examining these packets will allow you to quickly narrow the source of the problem. Check the following:

- Is the packet making it through the firewall? Check captures from both sides for both the query and response. Queries will usually, although not always, be sent on UDP port 53; replies may come back on either UDP or TCP port 53. The DNS specification talks mostly about UDP for queries and responses, and so some firewalls might only allow UDP responses to come in. In the sample packet that Figure 25.1 shows, the **IP** line shows that this was sent via **Proto=UDP**, not TCP.
- Is any response coming back? If not, there's obviously a DNS configuration problem. If negative (empty) responses are coming back, it's possible that one or more servers in a chain of forwarders isn't working. Try to capture packets from your internal DNS server to wherever it forwards packets, and check for responses.
- Is the request properly formed? It's rare for it not to be, but compare suspect requests to those from computers that are working. Request packets are usually quite short.
- Is the response properly formed, and does it contain accurate data? You can use the response that Figure 25.1 shows as a sample. The response should contain the original query (**DNS Question Section**) and a reply (**DNS Answer section**). The reply should contain at least one entry with a valid IP address; you can use ping or another utility to verify the IP address returned.

## Nslookup

Nslookup is a command-line tool that is included with every Windows NT-based Windows operating system (OS) from NT 4.0 on up, and can be used to verify the workings of DNS. As Figure 25.2 shows, you can simply type any host or domain name to receive a list of records through DNS. This query shows a *non-authoritative answer*, meaning that the answer was provided from a DNS server other than Microsoft.com's own authoritative one. The server delivering the answer was SERVER1.mshome.net, which looks suspiciously like a gateway. The gateway probably forwarded the request to an ISP, which returned the reply. The actual record came from www.Microsoft.akadns.net, which is a DNS hosting service that probably helps Microsoft load-balance its DNS requests. This illustration shows how most replies on most corporate networks are received.

```

C:\WINDOWS\System32\cmd.exe - nslookup
> www.microsoft.com
Server: SERVER1.mshome.net
Address: 192.168.0.1

Non-authoritative answer:
Name: www.microsoft.akadns.net
Addresses: 207.46.249.27, 207.46.249.190, 207.46.249.222, 207.46.134.190
Aliases: www.microsoft.com
>

```

Figure 25.2: Nslookup results.

Notice that the reply returned five unique IP addresses. It's likely that the DNS server is using round-robin to reorder these addresses on each query, helping to balance incoming traffic to this busy Web site. A second query, which Figure 25.3 shows, confirms this—notice that the first IP address in the second query was the second address in the first query.

```

C:\WINDOWS\System32\cmd.exe - nslookup
> www.microsoft.com
Server: SERVER1.mshome.net
Address: 192.168.0.1

Non-authoritative answer:
Name: www.microsoft.akadns.net
Addresses: 207.46.249.27, 207.46.249.190, 207.46.249.222, 207.46.134.190
Aliases: www.microsoft.com

> www.microsoft.com
Server: SERVER1.mshome.net
Address: 192.168.0.1

Non-authoritative answer:
Name: www.microsoft.akadns.net
Addresses: 207.46.249.190, 207.46.249.222, 207.46.134.190, 207.46.134.222
Aliases: www.microsoft.com
>

```


Figure 25.3: Seeing round robin in action.

Nslookup can be a valuable tool for seeing the details of how DNS is working. You can type ? at any Nslookup prompt (>) for a list of available commands.



## Q.26: How does Active Directory use the Domain Name System?

**A:** Active Directory (AD) uses the Domain Name System (DNS) not only to resolve host names to IP addresses, but also to advertise services such as domain controllers so that client computers can find them.

 For more information about how DNS works, see Question 21.

Normally, AD registers the proper DNS records dynamically using dynamic DNS (DDNS). However, in some situations, you might need to manually verify that the proper records have been created or create them manually on your own. (For information about what AD requires of DNS, see the sidebar “AD DNS Requirements.”)

### AD DNS Requirements


Keep in mind that AD’s only hard-and-fast requirement for DNS is that DNS support SRV records, as specified in Request for Comments 2782. Microsoft’s own DNS server provides this support, and any DNS server compatible with newer versions (8.1.2 or later) of BIND should work.

Contrary to popular opinion, DDNS is *not* a requirement for AD. It does, however, make things a *lot* easier, as AD creates a large number of DNS records that you’ll otherwise have to create and maintain by hand.

### Basic Records

All DNS servers supporting AD must include a zone named the same as the AD domain, such as braincore.net or west.mycompany.com. Only a forward lookup zone is required; reverse lookup zones are optional from AD’s perspective.

Authoritative servers must contain name server (NS) and start of authority (SOA) records indicating that the zone is authoritative. Additionally, each domain controller must have an A (host) record (and optionally an AAAA—IPv6 host) record registered, allowing basic translation of host names to IP addresses.

 Only Windows Server 2003 and later has a production IPv6 stack and an IPv6 DNS client. Older operating systems (OSs) are not capable of registered IPv6 names (Windows XP has a preliminary IPv6 stack not recommended for production use); of Microsoft’s DNS servers, only the one included with Windows Server 2003 can currently handle IPv6 AAAA records.

### Extra Zones

The primary forward lookup zone must contain a sub-zone named `_msdcs`, which must contain an NS record for each DNS server in the domain. The DNS server must contain another forward lookup zone named `_msdcs.domainname`. This zone must contain its own SOA record and an NS record for each DNS server in the domain.


### SRV Records

SRV records, first defined in RFC 2782 and RFC 2052, provide a locator service for specific network features. They allow, for example, client computers to locate domain controllers and Global Catalog (GC) servers.



In a Windows Server 2003 domain, the domain's primary forward lookup zone must contain two subfolders: *DomainDnsZones* and *ForestDnsZones*. These are similar in construction—the forest folder obviously contains records from across the forest, while the domain folder contains only records for the domain.

Each of these folders has a *\_sites* and a *\_tcp* subfolder. The *\_tcp* subfolder must contain an *\_ldap* SRV record for each domain controller in the domain, and the data of the SRV records must contain the name of each domain controller.

 Because the SRV records point to computer names, the computers must also have A or AAAA records in the forward lookup zone. These will allow clients to resolve the names to IP addresses.

The forest and domain folders also contain one subfolder for each site in the domain (or forest). These folders contain a subfolder for each network transport, which is generally just *\_tcp*. The *\_tcp* folder contains an additional *\_ldap* record and is used by clients to find domain controllers within their own site.

Additional SRV records, *\_kerberos* records, are contained in the *\_tcp.dc.\_msdcs.domainname* zone, the *\_tcp.sitename.\_sites.dc.\_msdcs.domainname* zone, and the *\_tcp.domainname* zone. These records are used to locate domain controllers providing authentication (which is to say, all domain controllers).

There are *\_gc* records, which provide the location of GC servers, created in the *\_tcp.sitename.\_sites.domainname* zone and the *\_tcp.domainname* zone.

Finally, there are *\_kpasswd* records created in the *\_tcp.domainname* and *\_udp.domainname* zones, which tell clients where to locate a domain controller that can process password changes (again, in AD, this is all domain controllers).

These SRV records not only provide the server name of domain controllers, but also provide the correct TCP or UDP port for appropriate services. For example, *\_kpasswd* records generally point to port 464, which is used by clients to change passwords.


 The UNIX world also uses *\_kerberos-master* and *\_kerberos-adm* SRV records; these are not implemented or used by AD.

Figure 26.1 shows the folder hierarchy that you're looking for.

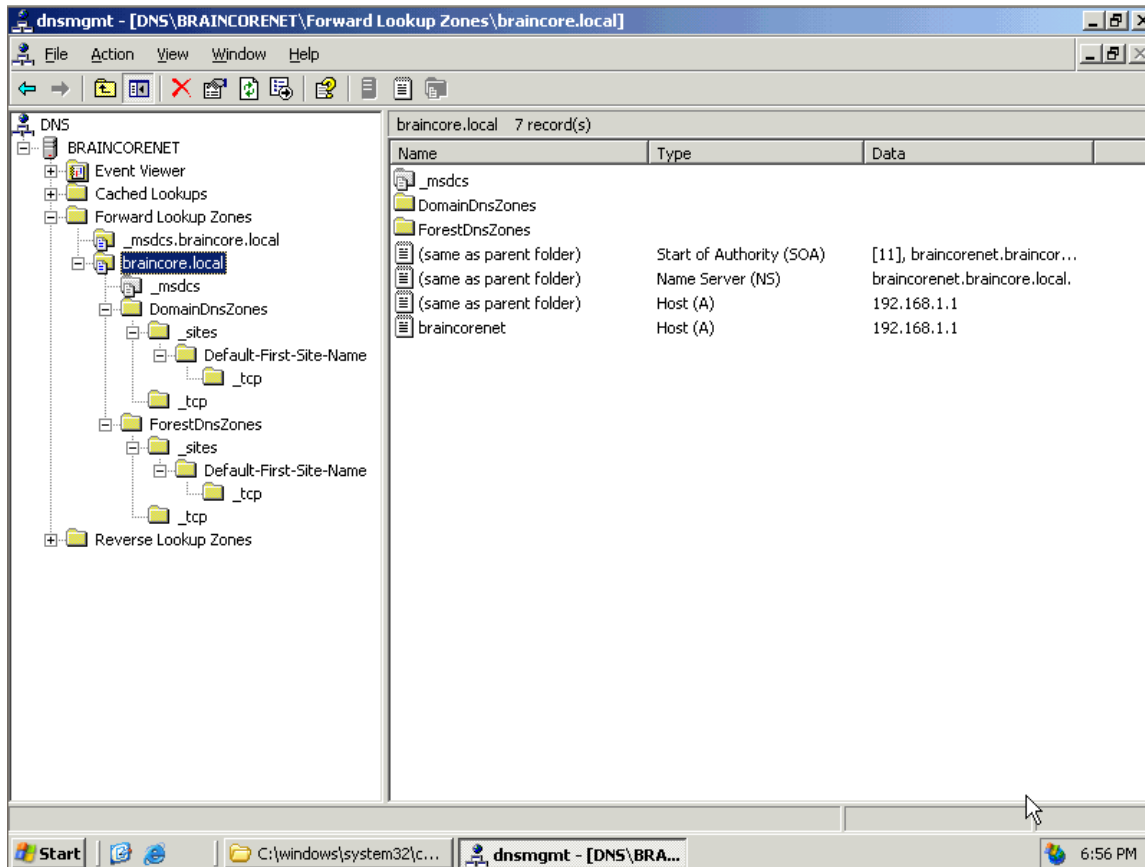


Figure 26.1: Folder hierarchy in an AD DNS zone

## Using DNS

Clients use DNS heavily during their startup and logon processes. Clients first query for a `_gc` record in their current site, if one is available, or from the forest if one isn't available within the site. Clients also query for a site-local `_kerberos` record to process their logons, and will fall back to any other `_kerberos` record in the same domain if a site-local domain controller isn't available. Most subsequent queries to AD—such as Contact object lookups—are processed through references to `_ldap` servers.


Again, bear in mind that these SRV records list *host names* and port numbers, not IP addresses. They're similar in that regard to CNAME records; clients must make a second query to DNS to resolve the host name to an IP address (or use a previously resolved, locally cached IP address).


## Q.27: How can I use a non-Microsoft DNS server to support AD?

**A:** A fairly decent percentage of Active Directory (AD) implementations aren't running on Microsoft Domain Name System (DNS) servers. In many cases, AD was brought into the environment long after DNS was up and running, typically on UNIX-based servers. In such cases, the company wasn't about to ditch its existing DNS infrastructure just to accommodate AD.

Fortunately, you don't have to scrap your existing DNS infrastructure. AD works quite well with non-Microsoft DNS servers, provided the servers support, at a minimum, the storage of SRV resource locator records. SRV support was first added to the industry-standard Berkeley Internet Name Distribution (BIND) in version 8.1.2, and most every new DNS server available supports SRV records.

AD will be considerably easier to work with if your DNS server also supports dynamic DNS (DDNS). Most newer builds do, but for security reasons, most DNS administrators disable this feature. Microsoft's DNS server supports secured DDNS through the use of Kerberos authentication, but Microsoft DNS server is about the only one to do so. If your DNS server doesn't, or can't, support DDNS, you'll need to manually create the necessary SRV records for each domain controller in the AD domain.

 For more information about which records to create, see Question 26.

 Be sure to thoroughly test your DNS server. Many builds equivalent to BIND 8.1.2 are buggy, and while they theoretically support SRV records, they often do so incorrectly or crash when the records are added or queried.

## **Q.28: How do I troubleshoot FRS problems?**

**A:** The File Replication Service (FRS) is a fairly simple service. Fortunately, there are several items you can check to narrow your search for the cause of a problem.

### ***Verify Connectivity and the Service***

FRS uses fully qualified domain names (FQDNs), not IP addresses, to contact replication partners. Start troubleshooting problems by confirming that you can use the ping utility to contact each replication partner from the console of each domain controller. If you can't, the domain controller might have a DNS configuration problem, DNS might be set up wrong, DNS might be down, or basic network connectivity might be affected. Troubleshoot these problems, and FRS will likely resume working.

You'll also need to verify that FRS is started. It should be configured to start automatically on all domain controllers; if such isn't the case, modify the service configuration appropriately and try starting the service. If it won't start, check the FRS event log for error messages.

Finally, verify that the remote procedure call (RPC) protocol can connect between domain controllers. This problem is rare unless domain controllers are separated by firewalls or port-filtering routers. The FRS event log on domain controllers will log event ID 13508 if the RPC service is unable to connect to its replication partners or if it's unable to create a secure RPC connection. Perform a network packet capture to see where the RPC traffic is going astray, particularly if there are port-filtering devices between the affected domain controllers.

## **Verify Active Directory**

Use the Microsoft Management Console (MMC) AD Sites & Services console to verify the replication schedule on the connection objects between any affected domain controllers. Ensure that replication is enabled; if Active Directory (AD) replication isn't working, FRS won't work either.

Connection objects should be automatically created between domain controllers in each site by the Knowledge Consistency Checker (KCC). If necessary, you can create manual connection objects, but in general, you want to get the KCC working properly—it provides the best way of keeping your replication topology working correctly.

 For more information about the KCC, see Question 16.


## **Test FRS**

The simplest way to test FRS is to create a file and see whether the file replicates. Place the file under one domain controller's SYSVOL hierarchy, and check to see whether the file shows up on other domain controllers. You should also be able to delete the file and see it disappear on other domain controllers. Don't delete or mess with any of the files already under SYSVOL—you can break AD in some fairly subtle and difficult-to-repair ways by manually removing Group Policy Objects (GPOs), scripts, and other files.

## **File and Disk Problems**

Ensure that both the source and destination computers have sufficient free space on their system volumes for *two* copies of each file being replicated. Remember that FRS creates a temporary copy of the file when sending and receiving; space must exist to make that possible.

Two events in the FRS event log can alert you to other disk space problems. Event ID 13511 indicates that the FRS database is out of space; either free up space on the system volume or move the FRS database. Event ID 13522 indicates that the staging folder used for temporary files is full. If the system volume should have enough space, then it's possible that the domain controller has not been able to replicate files for some time and it has built up a large number of files in the staging area while attempting to replicate them. Delete the connection objects from the affected domain controller, then stop and restart FRS. Doing so should delete the temp files and allow you to troubleshoot the basic connectivity issue.

 FRS cannot replicate encrypted files. Windows will never apply Encrypting File System (EFS) to any SYSVOL files on its own, but it's possible that another administrator may have done so. Doing so will break FRS until the files are decrypted.

## **FRS Utility**

Windows includes a diagnostic tool, Ntfrsutl.exe, which can help troubleshoot FRS problems. The syntax for this tool is:

- Ntfrsutl memory and Ntfrsutl threads—Lists the memory or threads being utilized by FRS. Optionally, include a remote computer name (Ntfrsutl memory server1) to get statistics for another computer.



- `Ntfrsutl ds`—Shows the FRS view of the directory service. Optionally, include a remote computer name to get the view from another computer's FRS.
- `Ntfrsutl poll`—Lists the current polling interval for FRS. Optionally, include a remote computer name (`Ntfrs poll server1`) to see another computer's polling interval.
- `Ntfrsutl poll /quickly`—Repeatedly polls partners in rapid succession until all partners are in sync. You can include a remote computer name to force this process on another domain controller.
- `Ntfrsutl poll /now`—Forces an immediate one-time poll. You can include a remote computer name to force it to poll.

## Log Files

In addition to its event log, FRS can produce a detailed debugging log to help pinpoint troubles. To change the log configuration, stop FRS, and open a registry editor. Browse to `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\NtFrs\Parameters`, and modify the following values:

- **Debug Log Severity**—Set this value to zero to record only the most severe events to the log; set this value to 5 to record almost every event that FRS can generate.
- **Debug Log Files**—Sets the maximum number of log files. FRS will create multiple files, deleting the oldest when the number specified in this value is reached. Set this value to a high number when troubleshooting to get the most data to work with. Five is a good value for normal operations.
- **Debug Maximum Log Messages**—Sets the maximum number of entries per log file, with 10,000 entries requiring about 1MB of disk space. When the log fills, a new one is created, up to the maximum number specified in Debug Log Files.

Restart FRS. Log files are created in `%systemroot%\Debug` and are named `Ntfrs_xxxx`, where `xxxx` is the log file number, up to the maximum you specified.

What should you look for in the log files? Primarily the keywords *error*, *warn*, or *fail*—all of which indicate a problem. You'll also see `SHARING_VIOLATION` when a file or process has locked a file that FRS is trying to replicate; repeated sharing violations indicate a problem that you can often resolve by restarting the affected domain controller.

To look for problems related to a specific file, open the log on the source computer. Look for the file name and the corresponding `:: COG` number in the file; note the GUID from that log entry and look for it in the destination domain controller's log files.

☞ Most log entries will contain a thread identifier. Because each file (or group of files) is replicated on a separate thread, if you find a failure or warning, look for all further log entries using the same thread ID. Doing so will allow you to follow that particular thread to see what it's doing and why it might be having problems.

## Q.29: How do Active Directory permissions work?

**A:** Active Directory (AD) permissions can be pretty complicated. Troubleshooting permissions issues requires you to know exactly how permissions work, where they come from, and how they're treated by AD.

### The Birth of Permissions

Permissions start in the AD schema, which defines the various classes and attributes that can be used to make AD objects, such as users and organizational units (OUs). The schema also defines the default permissions for each attribute and class.

For example, you probably know that users in AD can, by default, modify their personal information—data such as their phone number, address, and so forth. Figure 29.1 shows the AD Schema console with the User class open. As the figure shows, the special SELF user is granted Write Personal Information permissions. This default permission is copied to all new user objects, enabling the user (SELF) to modify personal information.

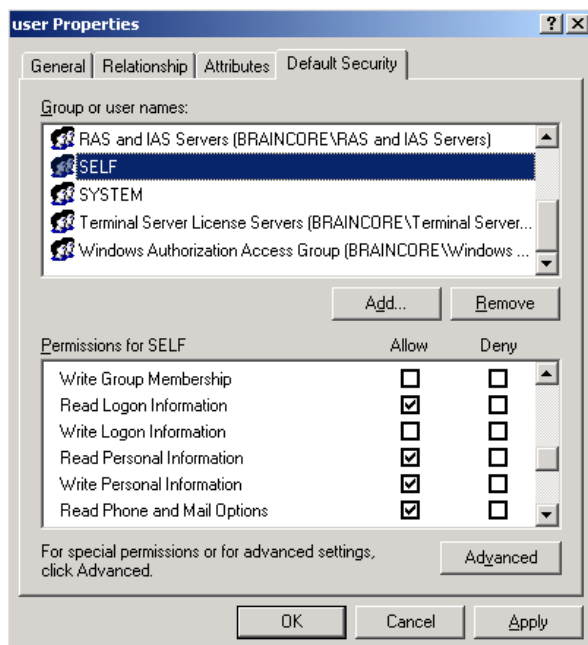


Figure 29.1: Default permissions of a user object.

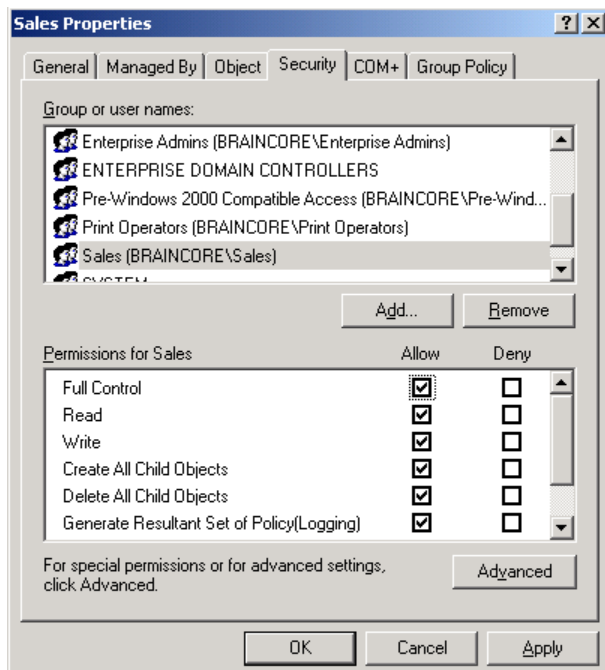
You can modify this default security by modifying the schema. Keep in mind that your change will affect the entire forest, you must be a member of the Enterprise Admins group to make this change, the schema must be in write mode for the change to be made, and that your change will not affect any existing objects—only new ones that are created after the change.

One popular default security change for user objects is to remove the Authenticated Users group's permissions to read personal information. Doing so makes AD less useful as an enterprise-wide phone book, but helps protect users' personally identifiable information—a step required by law in many European countries.

## Genetic Permissions

When you're born, you inherit a number of attributes from your parents, such as eye color. When AD objects are created, they inherit certain security attributes from their parent container. For example, an OU created under another OU will inherit certain properties from its parent OU. Similarly, user accounts created in an OU will inherit certain security settings from the OU.

AD inheritance is not very intuitive, though. For example, suppose you have a domain that contains an OU named Sales. Sales contains a user named RonB and a security group named Sales. The Sales OU also contains a child OU named InsideSales, which contains a user named SeanD. Suppose you modify the permissions on the Sales OU so that the Sales user group has Full Control permissions, as illustrated in Figure 29.2.



**Figure 29.2: Setting permissions on an OU.**

Now, quickly check the permissions on the InsideSales OU. You'll notice, as Figure 29.3 shows, that the permissions change wasn't inherited. The reason is that, by default, AD does not mark permissions for inheritance. Only some of the default permissions originally applied from the schema will be inherited by child objects by default.



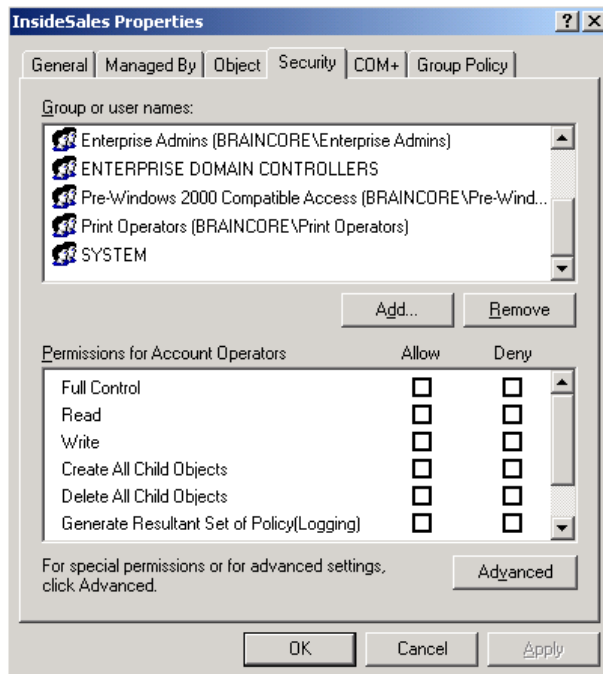


Figure 29.3: The new permissions aren't inherited by child objects by default.

Going back to the Sales OU, you can click Advanced to view the windows that Figure 29.4 shows. As the figure illustrates, I've re-added the Sales user group and given it full control over the OU. This time, I have the option to select the *Allow inheritable permissions from the parent to propagate to this object and all child objects. Include these with entries explicitly defined here* check box.

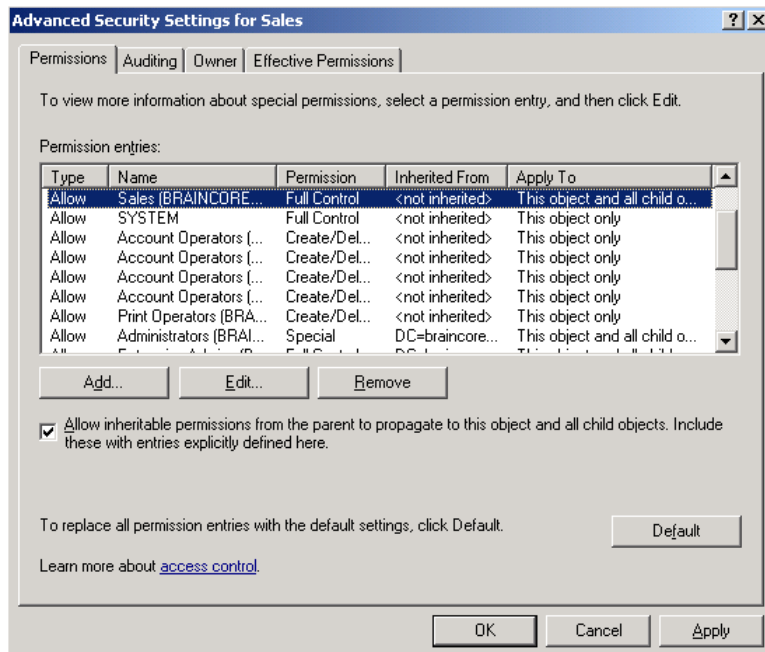
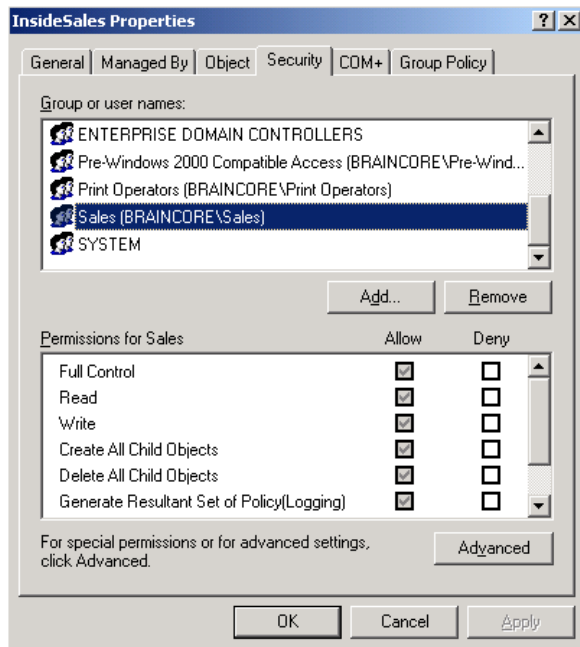


Figure 29.4: Adding an advanced permission that specifies inheritance.

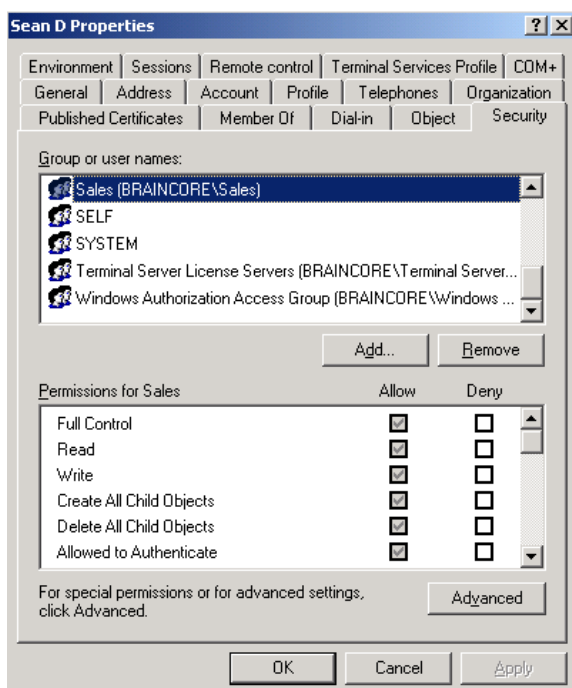


After saving the changes, reopen the InsideSales OU. Now the Sales user group has permissions (see Figure 29.5). In fact, as shown, their permissions are unavailable, which is AD's visual indicator that the permission is inherited, and you can't modify it on this child object.



**Figure 29.5:** The new permission now inherits to the child OU.

As one final check, open the permissions for the SeanD user account, which Figure 29.6 shows. Remember, that SeanD is the user account contained within the InsideSales OU. Sure enough, the Sales user group's permissions are inherited here, too.



**Figure 29.6:** The new permission also propagates to child users.

There's one other way commonly used to modify AD permissions: The Delegation of Control (DoC) Wizard. The DoC Wizard understands AD inheritance, and allows you to specify whether your delegated permissions should inherit to child objects. It automatically configures the permissions appropriately.

### **Permissions Summary**

AD permissions, then, are a combination of:

- The default permissions defined for an object class in the AD schema
- Any permission inherited from the object's parent container or containers
- Any permissions applied directly to the object itself

Permissions are never in conflict; they are additive. Thus, if a class doesn't grant permissions to a particular group through the schema, but that class is used to create an object in an OU that *does* grant permissions to a particular group, the new object will have inherited permissions for that group. When permissions seem to conflict, it's the "last permission in" that wins.

For example, suppose you've added a new user to an OU. The default permissions for the user class specify that the special SELF user doesn't have write permissions to personal information. However, the OU permissions grant write permissions to SELF, and that permission is marked to propagate to child objects. The new user, then, will grant the permission to SELF based upon the inherited permission.


The only exception to this general rule is a Deny permission. If the AD schema is set up to specifically deny a permission on the user class, for example, that permission will always be denied on all new user objects regardless of the inherited permissions. Inherited permissions cannot override a Deny permission.

### **Q.30: When promoting a server to be a domain controller, the DNS server couldn't be located. What can I do?**

**A:** When you run DCPromo, it performs several checks to make sure that a suitable Domain Name System (DNS) server is available on the network to handle the new domain controller that is about to be created. If you're installing the first domain controller in a new forest, and any of the checks fail, DCPromo will offer to install and configure DNS for you on the domain controller as a part of the promotion process. However, if you're promoting a domain controller into an existing domain, DNS must be up and functional first.

### **Basic Checks**

DCPromo starts by querying your configured DNS server (as listed in the computer's TCP/IP configuration) to find a zone that matches the domain name you've typed into the DCPromo wizard. So, for example, if you attempted to promote a server into the braincore.net domain, DCPromo looks for a DNS zone named braincore.net. DCPromo needs to find an *authoritative zone*, or one containing an SOA record.

 For more information about SOA and other records and how DNS works in general, see Question 21.

If these checks fail but you know you have an appropriate DNS zone, check the following:

- Ensure that the server is configured with the proper DNS IP address
- Ensure that the server can contact its DNS server and resolve other DNS names
- Ensure that the DNS server supports SRV records and is configured to support dynamic DNS (DDNS). AD doesn't *require* DDNS, but DCPromo looks for it to determine whether it will be able to create the necessary DNS records.

### **SRV Record Checks**

DCPromo needs to be able to find certain SRV records in DNS. Doing so allows DCPromo to locate a domain controller in the domain to which the new domain controller will belong. In the case of a new domain in an existing forest, DCPromo needs to be able to locate a domain controller in the forest. If the machine on which DCPromo is running isn't configured to look at a DNS server with the proper records, DCPromo either won't be able to proceed or will insist that you install DNS first.

Here's what to check:

- If you're installing a new domain controller in an existing domain, DCPromo queries the DNS record `_ldap._tcp.dc._msdcs.domainname`, where *domainname* is the domain to which the new domain controller will belong.
- If you're installing the first domain controller in a new child domain, DCPromo queries the DNS record `_ldap._tcp.dc._msdcs.parentdomain`, where *parentdomain* is the domain of the new domain's parent.
- If you're installing the first domain controller in a new root domain in an existing forest, DCPromo queries the DNS record `_ldap._tcp.dc._msdcs.forestroot`, where *forestroot* is the name of the forest root domain.

Finally, make a note of each host name referenced in these SRV records and ensure that a corresponding correct A record exists in the DNS zone. Then make one more check to ensure that the machine running DCPromo can ping each of the hosts listed in an SRV record by using their DNS name. If all of these checks succeed, DCPromo should have no problems with that portion of its checks.

### **DDNS**

Ensure that the DNS zones for the domain are configured to support DDNS updates. Although AD can technically function in a non-dynamic zone, assuming that you manually create the large number of required SRV records, DCPromo prefers a DDNS-enabled zone in order to run correctly. On Windows DNS, the dynamic updates option is easy to enable through the properties of the zone. For Windows Server 2003, DDNS is enabled by default for secure updates, as Figure 30.1 shows.

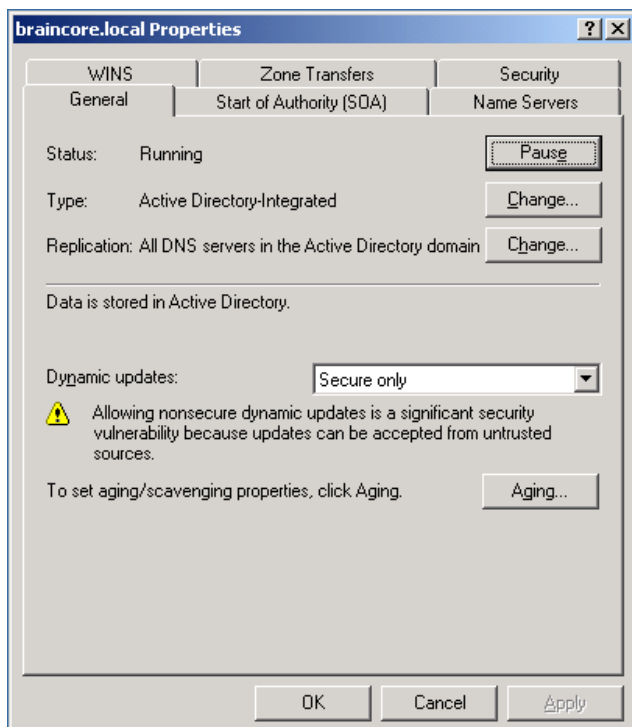


Figure 30.1: Dynamic updates enabled in a DNS zone.

## Checking DNS with Network Monitor

One way to see what DCPromo is doing is to use Network Monitor. You can install the version of Network Monitor that comes with Windows, and use it to capture all traffic sent to and from the machine running DCPromo. Run the capture while DCPromo is running; after DCPromo displays an error message, stop the capture. While viewing the capture, use Network Monitor's filter to display only DNS records, as Figure 30.2 shows.

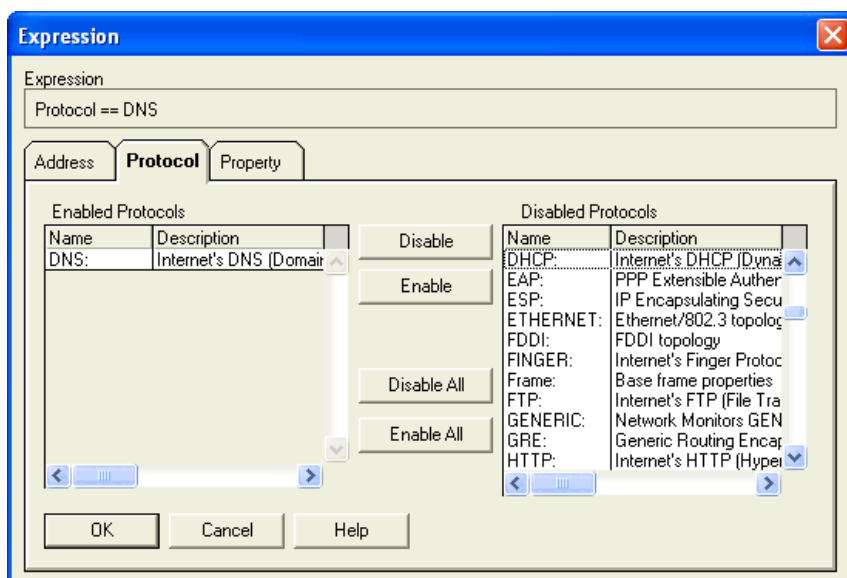


Figure 30.2: Filtering Network Monitor to show only DNS traffic.

Next, as Figure 30.3 shows, look for a Std Qry entry coming from the machine running DCPromo (Figure 30.3 shows a typical DNS query, which is what DCPromo will produce). Open the packet and look at the Question Section, which is highlighted in Figure 30.3. Doing so will show you which record DCPromo is trying to locate in DNS.

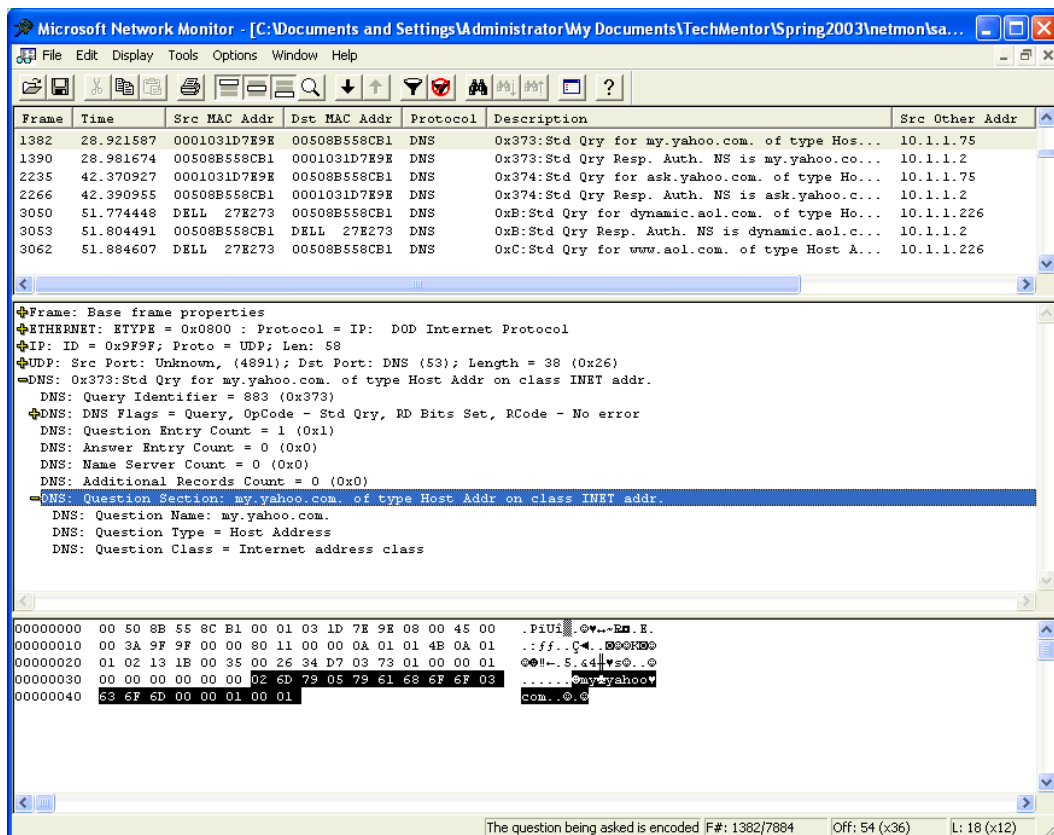


Figure 30.3: A DNS query packet in Network Monitor.

Finally, look for the corresponding Std Qry Resp packet, and open it. As Figure 30.4 illustrates, look for the Answer section (highlighted in Figure 30.4, but will vary depending upon the type of record DCPromo queried). In this example, the answer response is for a host (A) record. The answer section will show you the IP address or addresses returned by DNS; DCPromo will generally use the first address listed. Notice that in Figure 30.4, there are actually two Resource Records returned; clients will use the first of these.

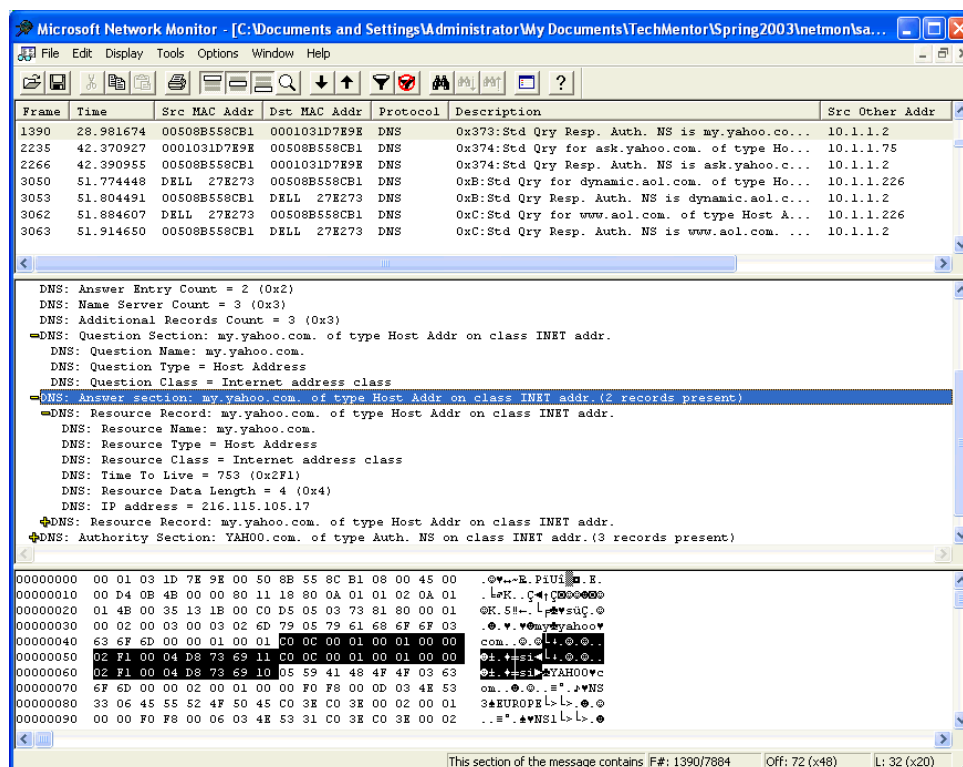


Figure 30.4: A DNS response packet in Network Monitor.

I've often used Network Monitor to troubleshoot DNS problems. Usually, the DNS client was querying a record I wasn't expecting, getting back a response, and was unable to contact that server. This happens more frequently on a large WAN, where the client is perhaps querying for a forest root domain controller and getting a response for a domain controller located across several WAN links.

## Q.31: How does the File Replication Service work?


**A:** Windows' File Replication Service (FRS) is present on all Windows 2000 (Win2K) and Windows Server 2003 servers. On member servers, the service is configured to start manually; on domain controllers, it starts automatically. FRS doesn't have a console for management and is largely self-maintaining. It is used to replicate both the contents of the SYSVOL share between domain controllers and the contents of Distributed File System (DFS) replicas. For our purposes, it's the SYSVOL replication that's important, because SYSVOL is in many ways the "other half" of Active Directory (AD) replication.

## Replication and SYSVOL

AD replication is responsible for replicating changes made within the directory, such as user and group changes. However, the directory also relies on data stored in SYSVOL for proper operation.

 For more information about AD replication, see Question 19.

SYSVOL includes the actual SYSVOL file share, the NETLOGON file share, all Windows 9x and Windows NT System Policies, and all Win2K and later Group Policy Objects (GPOs).


 GPOs are *configured* and *assigned* within AD, but the policy files reside in SYSVOL and must be replicated to all domain controllers in order for Group Policy to function correctly.

SYSVOL also contains all user and computer logon and logoff (and startup and shutdown) scripts. By default, SYSVOL is stored in C:\Windows\Sysvol, exists on all domain controllers, and should be identical on each domain controller.

## Key FRS Features

FRS supports *multimaster replication*, meaning that the data in SYSVOL can be changed on any domain controller and will be properly replicated to all other domain controllers in much the same way that data within AD can be changed on any domain controller and replicated to others. FRS is site-aware, meaning it replicates constantly between domain controllers within a site—it follows the AD replication schedule for intrasite replication. FRS is also multithreaded, which is an important improvement over the older LMRepl service in NT. FRS is capable of copying multiple files between multiple servers at the same time, improving performance and efficiency.

FRS does not, however, provide *deterministic* replication. In other words, you can never be sure when FRS will complete replication. FRS begins replicating a file whenever the file is closed so that files will always *begin* replication in the same order that they were modified. However, file sizes, network traffic, and a number of other factors can impact file copying, so files might *finish* copying out of order. Fortunately, SYSVOL doesn't generally need deterministic replication; it's enough that the files will *eventually* be replicated between all domain controllers.

 One way to quickly determine how well FRS is working is to create a small text file in SYSVOL on a domain controller. Then you can check each domain controller's copy of SYSVOL to see which domain controllers are replicating SYSVOL properly. However, don't rely on date and time stamps on the file, as FRS replicates the timestamps along with the file's access control list (ACL) and other attributes.



## **FRS Replication Process**

FRS registers with the operating system (OS) to receive notifications of changed files and folders under the main SYSVOL folder and its subfolders. When a file in that hierarchy is changed, FRS is notified and begins its replication process.

Within FRS, each domain controller is said to hold a *replica* of SYSVOL, meaning each domain controller is supposed to have an identical copy of the content. All domain controllers within a domain belong to the same *replica set*, which is simply a unit of organization within FRS. Keep in mind that FRS also supports DFS, which may define different replicas for file server content and different replica sets.

When FRS begins replicating, it contacts its replication partners. FRS uses the same AD connection objects as AD replication, so a single domain controller's replication partners will also be its FRS partners. FRS notifies its partners of a changed file, and the partners decide for themselves whether they want to accept the change.

Each time a file is changed, an *event time* is associated with it, marking the time of the change or when the file was last replicated. Partners will always accept a change if the event time on the replication source is more than 30 minutes later than the last event time for the same file on the destination. However, if the event time is more than 30 minutes *earlier* on the source than the destination, the change is ignored because the destination has a later time (and presumably a more recent copy).

If the event times on the source and destination are *within* 30 minutes of each other, additional rules apply. First, FRS checks the file version number; if the source has a greater version than the destination, the destination accepts the changed file from the source. Otherwise, the destination rejects the change. FRS uses its own version numbers, sequencing them in a fashion similar to the Update Sequence Number (USN) used in AD replication. Whenever a file changes and is closed, FRS increments its version number by one.

If the version comparison results in the source and destination having equal version numbers, the event time is checked again. If the source's time is later than the destination by any amount, the destination accepts the change; otherwise, it rejects it.

The purpose of this three-step process—event time with 30-minute window, version number, and event time without the window—is designed to determine as quickly as possible which replication partner has the most recent version of a file. Keep in mind that most SYSVOL content doesn't change very often; unless you update a bunch of GPOs at once, you're not going to change more than one file at a time, and so the 30-minute window is a very reasonable, fast first check.

## **FRS Caveats**

Unlike AD replication, which can replicate individual changed attributes, FRS can replicate only entire files. So even if you make the smallest change to a GPO, for example, FRS must re-replicate the entire file containing the GPO. This shortcoming is primarily a limitation of Windows' file storage, which makes it difficult to determine the "delta" between two binary files. Fortunately, most SYSVOL files are small, making replication of entire files an easy task. FRS runs into more difficulty when used as part of DFS, which can contain very large files.



## FRS Security and Details

The FRS service contacts its replication partners by means of an authenticated remote procedure call (RPC), which uses Kerberos. NTFS permissions on replicated files are preserved within the domain.

FRS only works with Win2K and alter; older OSs don't provide the Kerberos security nor do they provide the NTFS change journal which is used to notify FRS of changes to individual files. The change journal persists across system restarts and crashes, ensuring that FRS will pick up a changed file even if the system crashes or is restarted immediately after the file is changed.

FRS does not attempt to replicate files until they have been closed for at least 3 seconds. This setup ensures that files undergoing rapid, successive updates have time to complete and are closed before replication occurs.

When FRS prepares to replicate a file, it first copies it to a staging area on the source. Doing so allows the original file to immediately be opened for changes while FRS continues to replicate the most recently closed version. Replicated files are copied using standard Windows server message blocks (SMBs), just as if you were manually copying the files over the network. Receiving partners store the incoming file in a staging folder until the file copy is complete, then move the incoming file to the proper location in the local SYSVOL hierarchy. This progression ensures that the file copy completes successfully before the file is placed into service.

## Q.32: Active Directory permissions can be complicated. What's an easy way to troubleshoot permissions issues?

**A:** Active Directory (AD) permissions can be complicated. For example, consider the organizational unit (OU) structure that Figure 32.1 shows.

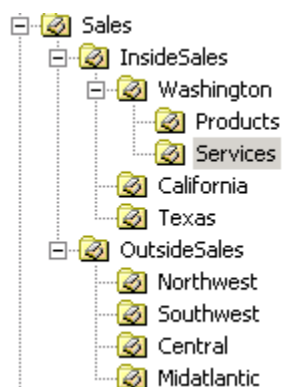


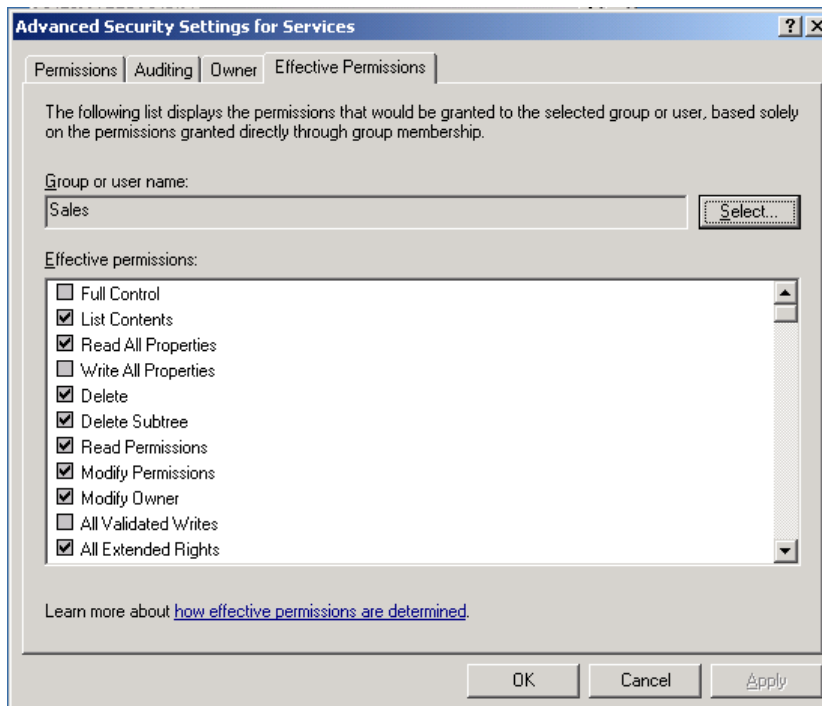
Figure 32.1: An example OU structure.

The Services OU is a child of the Washington OU, which is a child of the InsideSales OU, which is a child of the Sales OU—there are plenty of places at which security permissions could be applied and marked for inheritance. If you're trying to figure out who has control over child objects in the Services OU, the task can be daunting. This type of scenario is where most AD permissions issues occur.

For details about how AD permissions work, see Question 29.

## **Effective Permissions**

What you need is a way to see the final, effective set of permissions that any given user or group has to a particular object. That's your starting point for troubleshooting the problem. Fortunately, AD (in Windows Server 2003) provides an Effective Permissions tool. Start by selecting Advanced Features from the View menu in the AD console (in this case, Active Directory Users and Computers) to enable these features. Then open the properties for the object you need to investigate. On the Security tab, click Advanced, then select the Effective Permissions tab. Click Select to choose a user or group. As Figure 32.2 shows, AD will then display the effective permissions for that user or group. In this example, I'm trying to find out why the Sales user group can't modify certain properties of user objects in the Services OU. I gave the Sales group full control permissions on the Sales OU and marked those permissions to propagate to all child objects. It appears from this display, however, that something is removing some of those permissions somewhere along the hierarchy.



**Figure 32.2: Displaying effective permissions.**

## **Finding the Problem**

At this point, the grunt work sets in. I know the permissions on the Sales OU are correct, so I need to start at one end and work up or down the OU hierarchy, looking for a change. Turns out that, as Figure 32.3 shows, the necessary permissions were denied on the Services OU itself. The Deny permission overrides all others, removing the inherited permissions from the Services OU's parent.

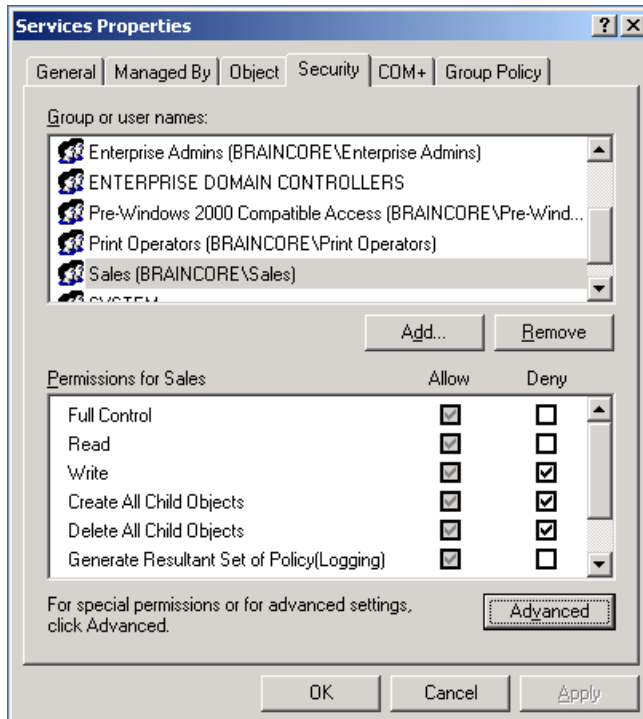


Figure 32.3: The Deny setting overrides the inherited permissions.

Anytime you're not getting the permissions you expected from AD, I recommend starting at the bottom with the effective permissions, then working your way up until you find the permissions change that's causing your problems.

### Q.33: How does Active Directory communicate?

**A:** Active Directory (AD) relies on several communications services to communicate with client computers and between domain controllers. The variety of communications protocols used reflects the complex nature both of AD and of the industry-standard protocols that AD implements, such as Kerberos and the Lightweight Directory Access Protocol (LDAP). Understanding how AD communicates can be critical when you're working with domain controllers or clients that are separated from domain controllers by firewalls or other port-filtering devices (such as routers).

## **Basic Communications**

AD needs only a few basic services to be available for normal operations:

- User Datagram Protocol (UDP) port 88 is used for Kerberos authentication. Transmission Control Protocol (TCP) port 88 can also be used, although it's less common.
- TCP and UDP ports 135 are needed for remote procedure call (RPC) endpoint mapping. RPCs are used for a number of domain controller-to-domain controller and client-to-domain controller operations. Unfortunately, not all communications take place over port 135, as I'll discuss later.
- TCP port 139 and UDP port 138 are needed for file replication between domain controllers. This port combination is the standard NetBIOS session service port set.
- UDP port 389 handles LDAP queries and is used for normal domain controller operations.
- TCP and UDP ports 445 are used for file replication and are the standard Windows file sharing ports.
- TCP and UDP ports 464 are the Kerberos password change protocol ports.
- TCP port 593 is used by the RPC over HTTP transport. Although you don't technically need this port for normal operations, I'll discuss later how this feature can make working with domain controllers through firewalls a bit easier.
- TCP port 636 is for LDAP over Secure Sockets Layer (SSL), which is the default LDAP methodology for Windows Server 2003 and later.
- TCP port 3268 and 3269 handle Global Catalog (GC) queries. Port 3269 handles secure queries. Any domain controller that needs access to a GC or that is acting as a GC server will use these ports.
- TCP and UDP ports 53 are used to communicate with Domain Name System (DNS), which is a vital part of AD communications.

Generally, opening these ports between clients and domain controllers, or between domain controllers, will enable AD to function normally. One exception is RPC traffic.

## **RPC Endpoint Mapping**

Most RPC communications first start on TCP port 135. However, that's merely the RPC endpoint mapper service. Its function is to select a new destination port for further communications in that RPC session. Exchange Server is a major user of RPC endpoint mapping, and it's very difficult to get Exchange traffic through a firewall as a result. The range of potential endpoint addresses used by RPC communications is huge, essentially requiring the entire firewall to be opened to allow all the possibilities. The ports selected by the endpoint mapper can range from TCP 1024 to TCP 65535.

Fortunately, you can force AD to always map endpoints to specific ports. The Internet Assigned Numbers Authority (IANA) has set aside ports 49152 to 65535 for private port assignments, so choose ports from this range and force AD to always use them. You'll then be able to open a much smaller range of ports in your firewalls.

To force port selection, modify the registry key

HKEY\_LOCAL\_MACHINE\SYSTEM\CurrentControlSet\Services\NTDS\Parameters. You'll need to create or modify a DWORD value named TCP/IP Port, and set it to whichever port you're going to use. However, there are some downsides to this modification:

- You'll need to modify every domain controller on your network; otherwise, they won't be able to communicate properly. You're effectively disabling endpoint mapping, so all domain controllers will have to be manually told which port to use.
- Your domain controllers will have to work a bit harder to handle the same number of connections. The servers communicate less efficiently when forced to use a single port for all communications because they can't rely on the port number to identify individual "conversations."

### ***RPC over HTTP***


Windows Server 2003 offers an exciting new communications protocol: RPC packets embedded within easily transported HTTP packets. This protocol is called RPC over HTTP, and it's handled by an RPC proxy DLL that's installed as an optional IIS 6.0 component.

Unfortunately, the computer initiating a conversation must choose to use RPC over HTTP, and Windows isn't currently designed to do so for domain communications. The only practical use for RPC over HTTP at the moment is Outlook 2003 communications with Exchange Server 2003; RPC over HTTP is invaluable there because it allows an RPC-heavy client such as Outlook to communicate through easy-to-manage HTTP ports. Hopefully, in the future, RPC over HTTP will become a more widespread means of communication.

### ***Choosing Your Battles***

If you're in a situation in which you have to have AD communications passing through a firewall, try to choose the path of least resistance. For example, domain controller-to-domain controller communications are amongst the most difficult as a result of the wide range of protocols in use and the need for constant RPC connectivity. However, client-to-domain controller communications are significantly less complicated, so placing a domain member in a perimeter network, for example, will be easier to deal with than placing a domain controller there.


If you absolutely must have a firewall between domain controllers, you'll need to restrict the ports they use. The File Replication Service (FRS) will need to be restricted, as will general communications. I explained earlier how to force an RPC port for general communications; Microsoft can help you with other types of traffic.

 See the Microsoft articles "Restricting Active Directory Replication Traffic to a Specific Port" for information about restricting AD replication traffic and "How to Configure a Firewall for Domains and Trusts" for information about configuring firewalls to support domain and trust communications. In addition, see "How to Restrict FRS Replication Traffic to a Specific Static Port" for information about restricting FRS traffic.

## **Q.34: How do I troubleshoot Active Directory communications issues?**

**A:** Active Directory (AD) is a complex product and requires a complex chain of communications for proper operation. There are really a few basic, common sequences of communication:

- Logon traffic—This category includes computers logging on to the domain in addition to user logons.
- Ticket request traffic—This type of communication occurs whenever clients need to access a new domain resource.
- Replication traffic—This type of communication occurs periodically between domain controllers and involves both intra-site and inter-site replication.

 For more information about AD communications, see Question 10, Question 19, and Question 33.

Additional traffic can occur when clients attempt to look up cross-domain references by contacting a Global Catalog (GC) server. Clients might contact GCs for other reasons, such as during the logon process, to resolve Exchange 200x Address Book lookups, and more.

Troubleshooting this traffic can be difficult. Kerberos traffic, for example, is always encrypted using temporary encryption keys established between the Key Distribution Center (KDC—domain controller) and clients. Even simple Lightweight Directory Access Protocol (LDAP) queries to GC servers or domain controllers are generally encrypted by Secure Sockets Layer (SSL) encryption, particularly in Windows Server 2003, in which LDAP over SSL is the default.

### **Know the Sequence**

The only way to effectively troubleshoot AD communications is to use Network Monitor. Although other utilities are very good at troubleshooting AD's operations, they don't get into low-level communications. You'll need to know what to expect, then you can see what traffic is actually being transmitted.

#### **Network Monitor Versions**

Keep in mind that Microsoft makes two versions of Network Monitor. The one that comes with Windows is restricted and can only capture traffic coming to and going from the local computer. If this version is the only one you have access to, you'll need to run it on the domain controller that you're troubleshooting.

The other version comes with Systems Management Server (SMS) and can capture any traffic on the local segment, even if the computer running Network Monitor isn't involved in the communication.

The one caveat you'll need to be aware of with Network Monitor is that it doesn't have built-in parsers for most of the traffic AD uses, including Kerberos. That's OK; you don't need a parser to troubleshoot traffic, you just need to know what you're doing. Network Monitor might not, for example, recognize Kerberos traffic when it sees it (because the tool lacks a parser), but I'll show you what to look for so you'll recognize the traffic.

Figure 34.1 shows a typical Network Monitor capture. In the top pane are all the capture frames of traffic; the bottom pane shows a breakdown of the frame selected in the top pane. Notice that the protocol in this case is listed as TCP. Network Monitor will show the Transmission Control Protocol (TCP) or User Datagram Protocol (UDP) for most AD traffic. To determine the actual traffic type, you'll need to look at the packet's details.

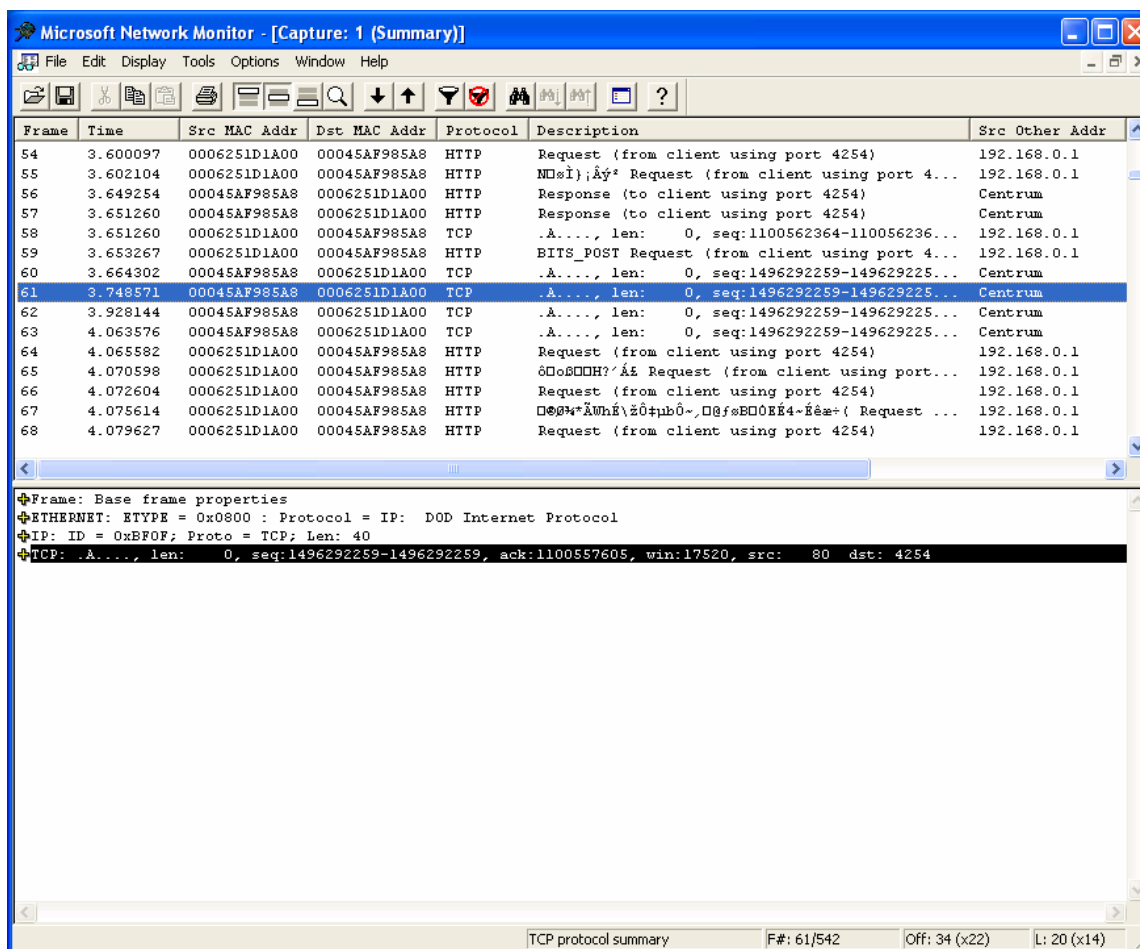


Figure 34.1: Typical Network Monitor capture.

In this example, the packet has a source (src) port of 80 and a destination (dst) port of 4254. I can see that the source address was a server and the destination was a client, so I can presume that this is a reply. That means the server's source port is the same port that the client originally used to contact the server: port 80. Port 80 is HTTP traffic, although Network Monitor obviously didn't recognize this frame as a traditional HTTP packet. The destination port was randomly chosen by the client when the client initiated the communication session.



Most AD communications will appear as TCP, and you'll need to look at the source port used by the server or the destination port used by the client to determine the traffic. Expect to see the following TCP and/or UDP ports:

- 88—Kerberos
- 135—Remote procedure call (RPC) endpoint mapper
- 53—Domain Name System (DNS)
- 137—NetBIOS name server
- 139—NetBIOS session service
- 389—LDAP query
- 445—Server Message Blocks (SMBs)
- 636—Secure LDAP (LDAP over SSL)
- 3268—GC LDAP
- 3269—Secure GC (LDAP over SSL)

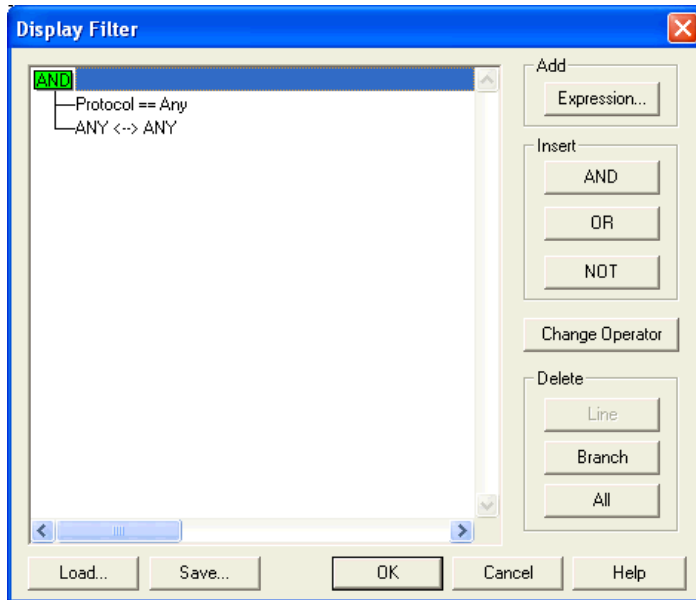
### **What to Expect**

The exact sequence of packets will differ from environment to environment, especially because AD traffic is usually interleaved with many other, unrelated forms of communication. If you're experiencing any form of problem, check the following items (these don't require you to memorize complex sequences of communications):

- If traffic is passing through a firewall or router, check the traffic on both sides of the device to make sure it's all getting through.
- Make sure DNS queries are working. Network Monitor will parse these and display them as DNS queries; ensure that queries are receiving appropriate responses and that clients are querying the correct servers.
- You can't actually read most exchanges between domain controllers or clients, but you can ensure that there's an actual conversation. In other words, make sure you're seeing the client make a request, then see the domain controller send something back.
- Watch for patterns. For example, if some operation seems to be taking too long or repeatedly times out, you'll likely see an identical sequence of packets repeat over and over. This repetition generally means something internal to the traffic has gone wrong, such as a user password being wrong or an incorrect configuration parameter.

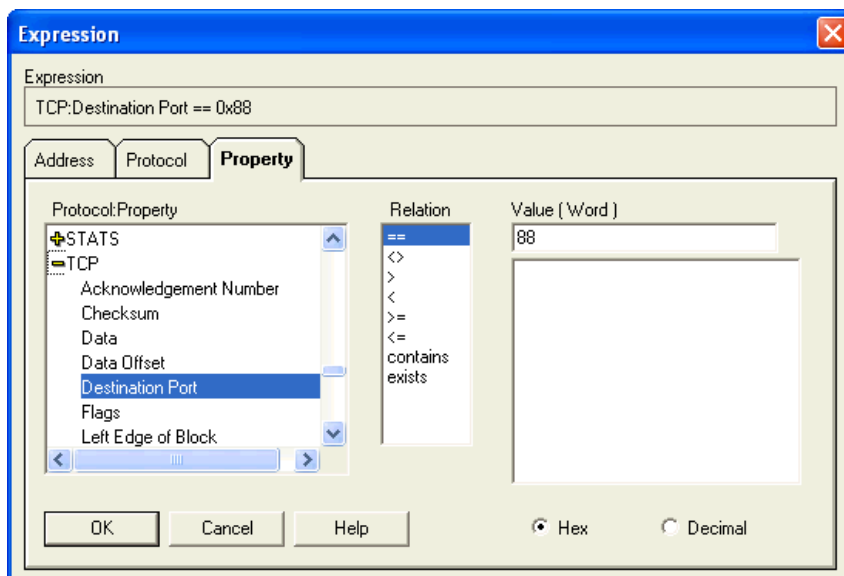
One way that Network Monitor can help draw out the AD traffic from everything else on your network is with filters. When viewing a capture, click Filter. You'll see a dialog box similar to the one that Figure 34.2 shows.





**Figure 34.2: Viewing filters in Network Monitor.**

Click Expression to add a filter expression. As Figure 34.3 shows, create an expression that filters for packets having a specific destination port. In this example, the filter will eliminate all traffic except Kerberos traffic, which uses UDP and TCP 88.



**Figure 34.3: Filtering for TCP port 88.**

Be sure to create filters that allow for a source or destination port so that you'll capture both the client and server sides of the exchange. You can use the Or button on the filter dialog box to ensure that any packets meeting any one of your criteria will be included in the display. Figure 34.4 shows an example of the configured filters.

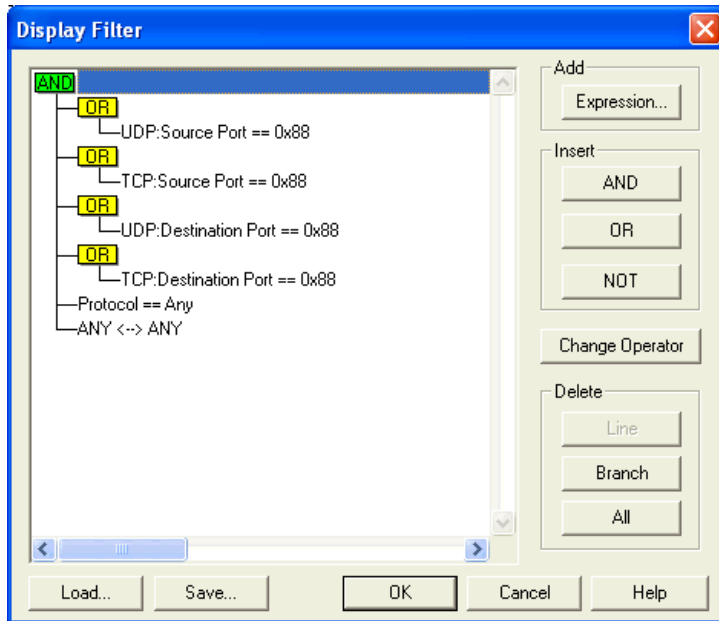


Figure 34.4: Configured display filters.

Filtering for specific types of traffic will help you focus on different areas to be sure they're working properly without having to manually wade through all of the packets that Network Monitor may have captured. You can load and save filters for later use, making it easier to quickly look at Kerberos traffic, DNS traffic, LDAP queries, and so forth.

### No Fixes, Just Symptoms

Network Monitor won't allow you to directly fix any problems. However, it can point the way to what's broken. For example, if capturing packets from each side of a firewall reveals that Kerberos (port 88) packets aren't making it through, you'll know that you need to reconfigure the firewall. If DNS queries contain incorrect IP addresses, you'll need to fix the DNS zone database. If traffic seems to be repeating itself, there's something wrong or misconfigured within the traffic itself (such as a username or password contained within the traffic), and you'll need to closely examine the configuration of the computers involved.

### Q.35: DNS works sometimes, but sometimes it doesn't. What can we do?

**A:** Troubleshooting any kind of intermittent network problem can be a nightmare. Fortunately, Domain Name System (DNS) is a fairly simple protocol, and there are only so many things that can go wrong.

### **Low-Hanging Fruit**

Start by eliminating obvious problem areas, such as unavailable DNS servers, WAN links that are down, unplugged cables, failed network cards, and so on. These types of problems will almost always manifest in other ways, because DNS won't usually be the only thing affected. Because your client will more often than not be configured with multiple DNS server addresses, use Network Monitor to analyze the network traffic being sent by your clients. That way, you'll know exactly which DNS server they're trying to talk with, and you can focus your troubleshooting efforts there first.

Another common problem root is multihomed servers. Unless specifically instructed to do otherwise, these servers will register all of their IP addresses with DNS. Some of those addresses, however, may be associated with network interfaces that not all clients can access. The result is that some clients will have access to the server and others won't. You may also have clients that switch between having access and not, particularly if DNS round robin is enabled on their DNS server. Round robin may be alternating between an accessible IP address and an inaccessible one, creating intermittent problems for clients.

### **Replication Issues**


Replication issues can cause intermittent problems in Active Directory (AD)-integrated DNS zones. Ensure that AD replication is working properly to start with. Clients that are querying different DNS servers may be receiving different responses if the two servers haven't yet converged.

 For more information about how AD replication works, see Question 19.

If replication latency is a problem for your DNS zones, consider upgrading to Windows Server 2003. In Windows Server 2003, the DNS zone is stored in an AD partition, and you can control which domain controllers contain a copy of the partition. By limiting the partition to just those domain controllers that are acting as DNS servers, you'll force a new replication topology to be generated for that partition. The result will be fewer servers replicating the information. Thus, replication will be able to occur more quickly, causing the different copies of the partition to converge more quickly and reducing problems caused by replication latency.

### **Protocol Problems**

Another problem can occur if your network is assuming that DNS uses User Datagram Protocol (UDP) port 53 and blocks access to Transmission Control Protocol (TCP) port 53. The DNS specification requires DNS to use the connectionless UDP transport protocol, but only for small queries. Larger queries—or, more accurately, larger query *responses*—that won't fit into a single UDP packet may be broken into multiple TCP packets instead. This switch to TCP can cause bewildering problems on your network because some DNS queries will work fine and others will simply time out.

 DNS *queries* will nearly always go out via UDP (the notable exception being the Simple Mail Transfer Protocol—SMTP—service in IIS, which seems to always use TCP); *replies* will come in on UDP or TCP depending upon the number of hosts and IP addresses contained within the replies.

If you're not sure whether this circumstance relates to your problems, try using Network Monitor to capture DNS traffic on both sides of your firewall. If you're not seeing identical traffic on both sides of the firewall, the firewall is obviously blocking some DNS traffic, most likely large replies. To play it safe, I recommend opening your network to incoming DNS traffic on both UDP and TCP ports 53.

### **Q.36: How can we troubleshoot Group Policy application?**

**A:** Group Policy application seems straightforward enough: Group Policy Objects (GPOs) are linked to organizational units (OUs); users and computers are in OUs. All the GPOs from a user's OU hierarchy filter down to the user. Easy enough.

Things get more complicated, though, when you remember that GPOs can be linked to a domain and to sites—meaning you'll have to open a whole new console to see what's going on. You also have to consider local security policies, which exist solely on the client computer and are applied before any domain-based policies arrive. Throw in options such as Block Policy Inheritance, No Override, and loopback processing, and it's no wonder why there's such a robust market for third-party GPO tools. However, with some patience and a methodology, you can do quite a bit of quality troubleshooting on your own.

#### ***Start at the Bottom***

Too many administrators try to start at the top, working their way down the hierarchy of GPOs and figuring out which ones apply. That method is time-consuming, error-prone, and just plain boring. It's a lot easier to start at the bottom—the client—and work your way *up* the tree.

Windows XP's Gpresult tool, for example, is a great troubleshooting tool. Run from the command line, it will tell you which groups the current user is a member of (which can affect GPO application), and give you a list of every GPO that is currently affecting the user. You'll also see the last time that GPOs were applied to the computer. What Gpresult is displaying is called resultant set of policy (RSOP). It sorts through all the blocked inheritance, no overrides, and conflicting policies to sort out exactly which policies are being applied.

By default, Gpresult doesn't show you which individual *policies* are applied or what they are set to; because GPOs successively overwrite one another as they are applied, you can still be left with a troubleshooting task to figure out which of the GPOs listed is responsible for the settings you're seeing. Fortunately, Gpresult has a “superverbose” mode, enabled by running

```
Gpresult /z
```

This mode not only displays which GPOs have been applied, but lists every single policy that's enabled in each GPO, allowing you to see which GPO modified which setting, and which GPO finally won out in the end. Figure 36.1 shows a portion of Gpresult's superverbose output. In this example, the GPO being applied is Local Group Policy, and you can see exactly which registry keys each setting is modifying.

```

rs
State: Enabled

GPO: Local Group Policy
Setting: Software\Policies\Microsoft\Windows\Safer\CodeIdentifie
rs\262144\Paths\{8868b733-4b3a-48f8-9136-aa6d05d4fc83}
State: Enabled

GPO: Local Group Policy
Setting: Software\Policies\Microsoft\Windows\Safer\CodeIdentifie
rs\262144\Paths\{7272edfb-af9f-4ddf-b65b-e4282f2deefc}
State: Enabled

GPO: Local Group Policy
Setting: Software\Policies\Microsoft\Windows\Safer\CodeIdentifie
rs\262144\Paths\{191cd7fa-f240-4a17-8986-94d480a6c8ca}
State: Enabled

GPO: Local Group Policy
Setting: Software\Policies\Microsoft\Windows\Safer\CodeIdentifie
rs\262144\Paths\{7272edfb-af9f-4ddf-b65b-e4282f2deefc}
State: Enabled

GPO: Local Group Policy
Setting: Software\Policies\Microsoft\Windows\Safer\CodeIdentifie
rs\262144\Paths\{191cd7fa-f240-4a17-8986-94d480a6c8ca}
State: Enabled

GPO: Local Group Policy


```

Figure 36.1: Gpresult's superserverbose mode.

Superserverbose mode also breaks down the user and computer policies, allowing you to see every setting that is affecting the current users or their machines.

## Centralized Troubleshooting

For Windows Server 2003, Microsoft introduced a sort of server-side Gpresult. It's called RSOP, and it's built into the Active Directory Users and Computers console. With this tool, you can actually view applied policies in a graphical user interface, which can be a bit easier to work with than the text-only Gpresult output.

 The Windows Server 2003 version of the Active Directory Users and Computers console will work fine against Windows Server 2000 domains running Service Pack 3 (SP3) and later; however, you might need to purchase a copy of Windows Server 2003 to be able to legally use the new the new Active Directory Users and Computers console.

To launch the new tool, open the Active Directory Users and Computers console and select a user or computer. Right-click the object, and select Resultant Set of Policy (Planning) from the All Tasks menu. The RSOP wizard will step through a number of screens that allow you to specify the user or computer account location, determine whether you want to simulate the effect of a slow network link or loopback processing, choose a specific site, or modify either the user's or computer's security group memberships. The final result, which Figure 36.2 shows, is a console that looks a lot like the Group Policy Object Editor, displaying each of the policies that have been applied to the user. For each policy, you'll see which GPO was responsible for the final application of the policy, making it easy to see where you need to go to make changes.

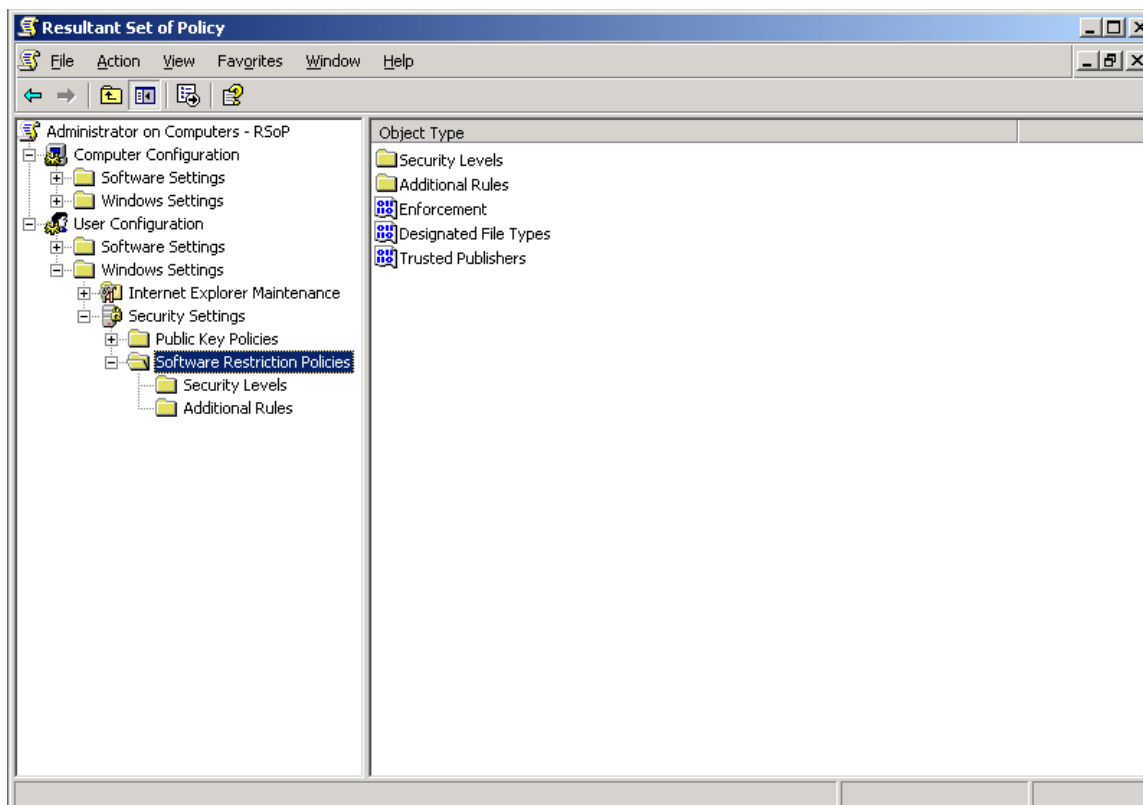


Figure 36.2: Output from the new Active Directory Users and Computers RSOP tool.


### Remember the Pecking Order

Remembering the pecking order of GPOs can be helpful when troubleshooting. In general, the *least specific* policies apply first; those which are *more specific* to a user or computer apply last and have the opportunity to overwrite policies applied earlier. That's an important concept: The *last* policy applied remains effective, even if an *earlier* policy had a contradictory setting.

Local policies apply first; they have the advantage of living right on the computer, so they don't have to wait for the user to log on. Next comes the Active Directory (AD)-based policies, and of those, site policies apply first. Domain policies, which are more specific, apply next. Finally, OU policies apply starting at the topmost OU and working down. OUs can block policy inheritance, meaning higher-up policies will not apply from that point on down. However, higher-up policies can also be marked as No Override, meaning they'll break through Block Policy Inheritance and be inherited anyway.

### Replication Problems Equals Inconsistency

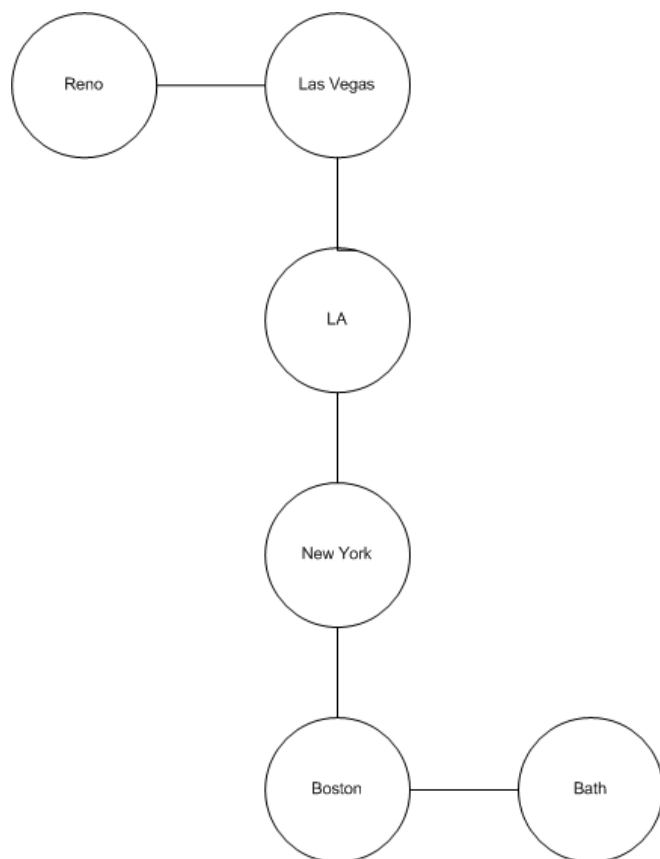
If users are experiencing inconsistent GPO application, the problem is most likely a failure in Active Directory's (AD's) GPO replication process. Although AD *defines* GPO links in the AD database, the GPOs themselves are contained in normal files, which are replicated from domain controller to domain controller by the File Replication Service (FRS). A failure in the FRS can result in inconsistent GPOs on domain controllers, which results in users having inconsistent GPO application.

 For more information about the FRS, see Question 31.

Perhaps the simplest way to verify that the GPOs have replicated consistently is to check the files themselves, located in each domain controller's SYSVOL share. Provided each domain controller has the same files, with the same date and time stamps (which are replicated by FRS, not recreated on each domain controller), then everything should be consistently applied to all users.

### **Q.37: Are WAN links are being over-utilized by Active Directory replication traffic. If yes, why?**

**A:** Most companies do the right thing when it comes to Active Directory (AD) site design. For example, suppose you have several sites connected by T1 lines, as Figure 37.1 shows. The T1 lines represent your WAN's physical connectivity, and all the AD design guidelines tell you that your site links should reflect that physical connectivity.



**Figure 37.1: Sample physical WAN design.**

If you set up one site link per T1 link, you'll wind up with a design somewhat like the one in Figure 37.2. This configuration is straightforward and reflects the way that most companies build their sites and networks. If you have additional backup links between sites, you might even configure separate site links for those, configuring a higher link cost to reflect the link's nature as a backup.



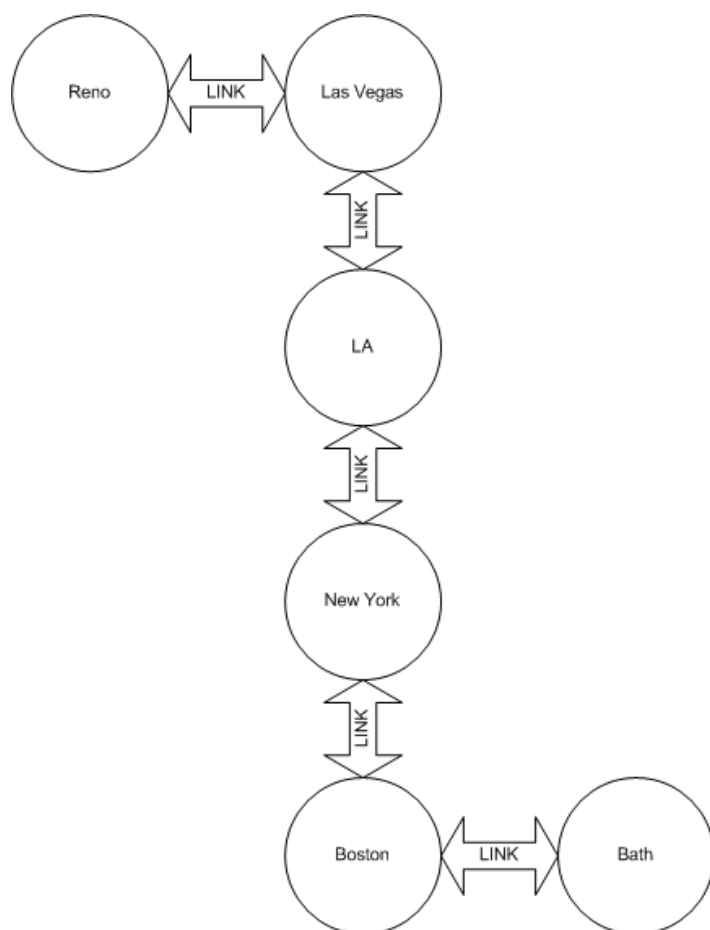
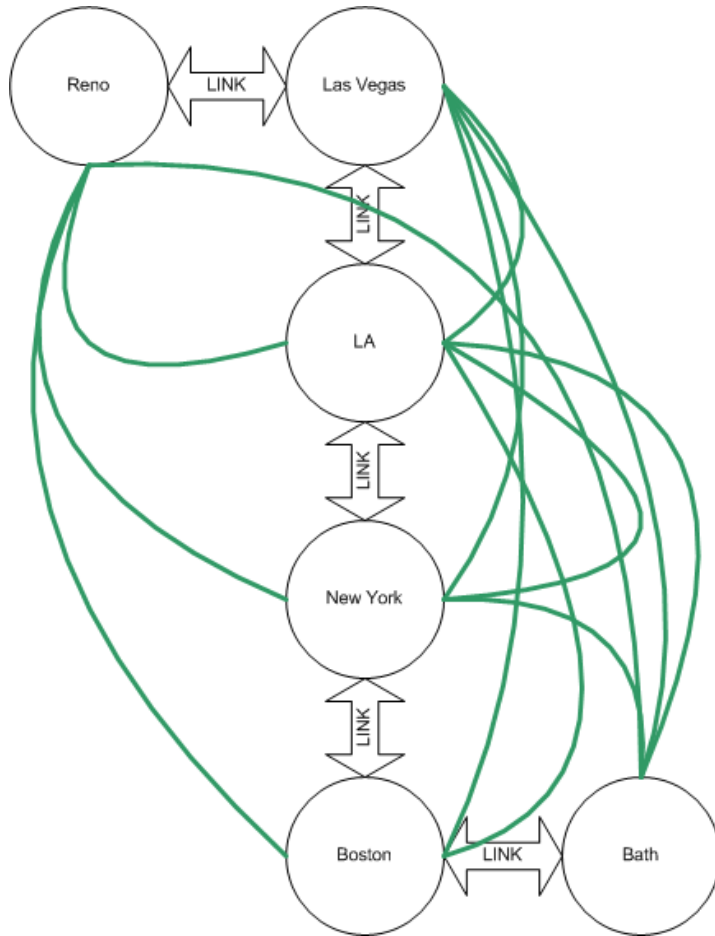


Figure 37.2: Site link topology.

### Secret Bridges


What you might not realize is that AD, by default, creates *site link bridges* for every site link. This setup isn't necessarily a bad idea. For example, consider what happens if an administrator locks out a user account in the Bath office. Obeying only the site links, AD will have to replicate that change from a bridgehead in the Bath office to a bridgehead in the Boston office, then to New York, LA, Las Vegas, and finally Reno. Depending upon your replication schedules, it could be quite some time before a Reno domain controller actually locked out the user's account, even though account lockout is a high-priority change for replication. In the meantime, a user could be logging on to a Reno or Las Vegas domain controller, relying on the fact that those domain controllers haven't yet heard about the lockout.

AD's automatic site link bridges create a topology similar to the one in Figure 37.3, in which each site is *logically* linked to the others.



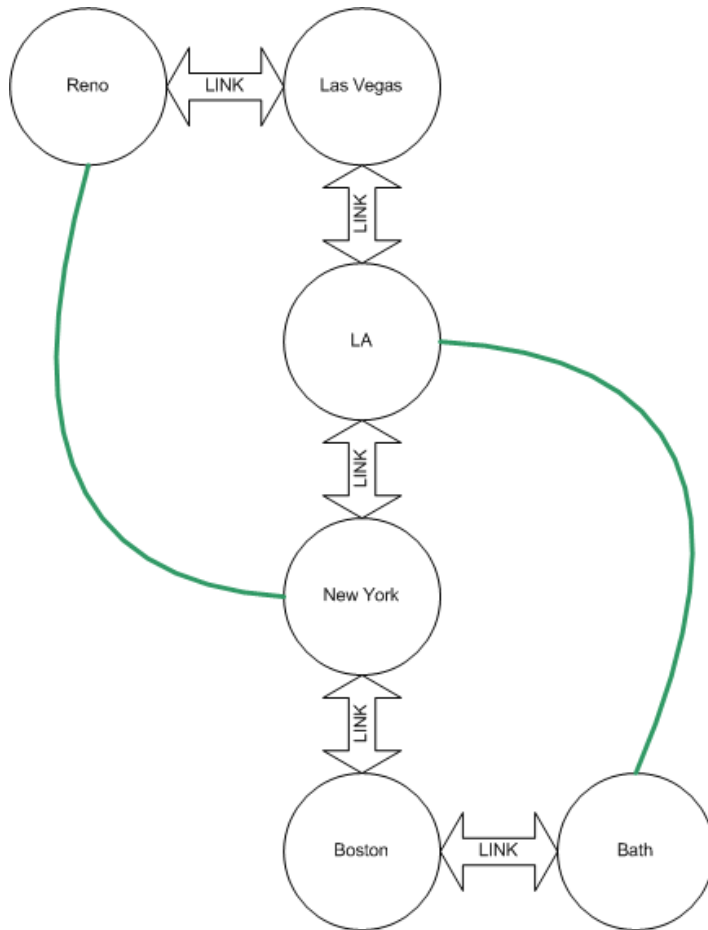
**Figure 37.3:** Site link bridges are shown in green.

When a change is made at the Bath office, its bridgehead domain controllers replicate *directly* to bridgehead domain controllers in each of the other offices. Effectively, AD is ignoring the physical layout of your network a bit in order to speed replication. The cost is that your WAN links are going to carry more traffic than you might expect. For example, the link connecting Las Vegas and LA will carry Bath's change *twice*—once as Bath replicates with Las Vegas, and once as Bath replicates with Reno. The link between Bath and Boston will carry the same replication traffic *five* times—once for each of the other offices.

 For more information about how site link bridges are created and how the replication topology is generated, see Question 19.

### Being Smarter than AD

You don't have to let AD create site link bridges automatically. In fact, you can disable the behavior entirely. However, doing so puts you right back to a high degree of replication latency, which may not be any more desirable than wasting WAN bandwidth. Fortunately, there's a happy middle ground. Consider the topology that Figure 37.4 shows, in which two site link bridges have been manually created.

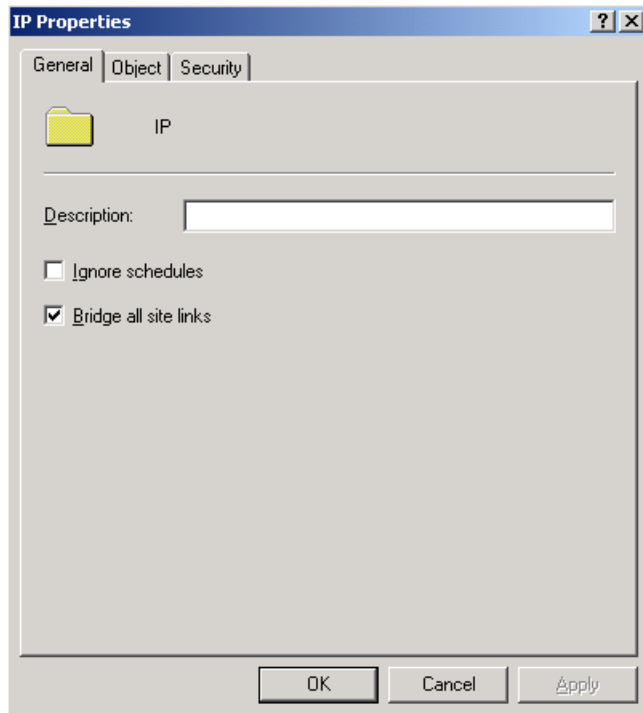


**Figure 37.4: Manually created site link bridges.**

In this case, a change made at Bath would replicate to Boston and LA, because LA is “virtually” connected to Bath. LA would then replicate to New York and Las Vegas. Reno would receive the information last, either from New York or Las Vegas. You might reconfigure this setup a bit to have the Reno site link bridge connecting to LA rather than New York; doing so would place more burdens on LA-based domain controllers, but would disseminate the information in fewer steps. The site link bridges effectively shortcut the physical topology, wasting a small amount of WAN bandwidth but providing minimal replication latency.

### ***Making the Change***

You can make this configuration change in the AD Sites & Services console. You’ll need to make the change for each intersite replication transport in use, although most companies will just have IP. Right-click the transport’s folder, and select Properties from the context menu. As Figure 37.5 shows, the default behavior is to bridge all site links; you can disable this behavior by clearing the check box.



**Figure 37.5: Default settings of the IP intersite transport.**

If you disable this behavior, you should definitely review your manual site link bridges and create new ones as necessary to provide the desired amount of replication latency. I recommend testing your new topology by making changes in your furthest-out office (Bath, in our example) and measuring the amount of time it takes the change to replicate to the opposite corner of your network (Reno, in our example). Adding site link bridges that bridge from the edges of your WAN to the middle of your WAN is the most effective strategy, as it provides the most efficient shortcut for replication traffic.