Migrating from Microsoft Windows NT Server 4.0 to Windows 2000 Server

Microsoft Corporation
September 1997

Summary: The Microsoft® Windows® 2000 operating system evolves the current Windows NT directory into a fully extensible, scalable directory service that can meet the needs of corporate intranets, as well as commercial Internet providers.

This white paper was written to provide the reader with a core understanding of the concepts involved in a migration or new implementation of a Windows 2000 directory.

Contents

Introduction

Active Directory Features

Microsoft Windows 2000 evolves the current Windows NT directory into a fully extensible, scalable directory service that can meet the needs of corporate intranets as well as commercial Internet providers.

The key features of the directory service are:

- Hierarchical and scalable namespace
- Partitioning of directory for scalability
- Multimaster replication
- Dynamically extensible schema
- Online backup and restore
- Open and extensible directory synchronization interfaces
- Lightweight Directory Access Protocol (LDAP) as the core protocol for interoperability

These features are described in some detail in this paper:

Hierarchical and scalable namespace

Windows NT 4.0 is capable of storing up to 40 megabytes (MB) of objects per domain, which allows up to 40,000 users per domain in the registry-based Security Account Manager (SAM).

Because Windows NT 4.0 uses a flat list organization for the namespace, administering a very large domain can be difficult. Administrative tools, such as the User Manager for domains, cannot start and display all objects quickly. Moreover, data represented in a flat list makes it hard to find a particular object.

To allow simpler, more flexible administration, the Windows 2000 directory uses a structured database, based on Microsoft Exchange directory storage as its data store. Using the Extensible Storage Engine (ESE) allows the directory to scale up to 10 million objects in a single data store, overcoming the limitations of the registry-based SAM database in Windows NT 4.0. The Directory Service Agent (DSA) runs on top of the flat database and implements a hierarchical namespace. With this hierarchical namespace, the Active Directory takes the existing domain model forward to a new "tree of trees" model. By splitting the namespace into a hierarchy, you no longer need to view tens of thousands of users in a flat list.

Within a Windows 2000 domain, you can create *organizational units* (OUs), which are containers that hold objects in the Active Directory. OUs contain objects, such as users, groups, printers, and so forth, which can be organized into a logical structure that maps to the way you work and organize your business. Additionally, you can delegate administration based on permissions assigned to the organizational unit.

The Active Directory provides a rich model for access control permissions, which, in many ways, resembles managing permissions for files and directories. You can delegate managing the objects in any organizational unit by assigning ACLs (access control lists). ACLs allow you to delegate management to just those objects and properties you want. For example, you could give the help desk staff permission to reset passwords, but could prevent the same help desk staff from adding or deleting accounts.

Giving an administrator the right to create or modify objects in only one container confines the administrator to only that portion of the tree. This granular management of permissions allows you to provide fine-grained control of administrative responsibilities and boundaries. Using organizational units in the directory hierarchy reduces the number of domains needed to obtain a management hierarchy.

In addition to the trees of organizational units within a domain, you can form a tree of domains. These domains are linked into a hierarchical tree using Kerberos transitive trust. Transitive trust simplifies managing domains by requiring just a single "connection" to the tree. Windows NT-based user accounts are valid anywhere in the tree and provide the single user login that is so important for managing network applications that require authentication and authorization in a distributed environment.

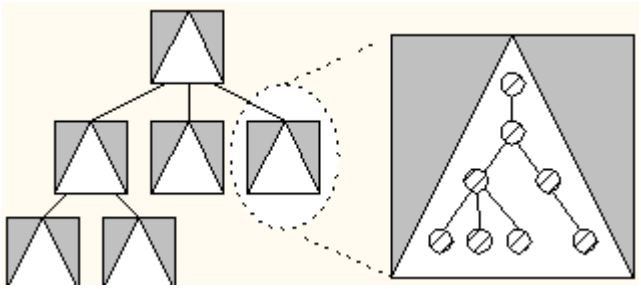


Figure 1. The Active Directory "Tree of Trees"

Multimaster replication

Windows NT 4.0 implements a single master replication model. The primary domain controller (PDC) is the only domain controller that has a read and write copy of the domain database. All other domain controllers are backup domain controllers (BDCs). The PDC replicates all changes in the domain database to the BDCs.

With Windows NT 4.0, the PDC must always be accessible for changes to be made to user accounts, groups, and so forth. If the PDC server is down or if there are network disconnects, the directory is unavailable for changes.

Windows 2000 and the Active Directory implements a multimaster replication model. With multimaster replication, changes can be made on any domain controller in the domain. The domain controller then replicates the changes to its replication partners. This results in 100 percent availability of the directory for changes, even if single domain controllers are unavailable.

Online backup and restore

To achieve a very high availability of domain controllers, the Active Directory allows the backup of domain controllers while they are online.

Dynamically extensible schema

A directory's schema defines the objects and properties that can be created in the directory. In the Active Directory, the schema consists of three tables, one for each of the following:

- Objects
- Attributes or properties
- Syntax objects

For example, in the definition of a user, the Active Directory schema defines a user object, which contains links to the properties that can be set for users—such as a first name property, a last name property, and a password property.

The Active Directory allows you to extend the schema and to create new properties and objects. Developers can use this extensibility feature to create their own data structures in the directory for applications, thereby using the directory as a data store. For example, a human resource application already finds a huge amount of information about an employee in the directory. This information includes the employee's first and last name, phone number, office number, and home address. Using the Active Directory, the application can extend the schema to add necessary attributes, such as the employee's salary.

Furthermore, the extended security model of the Active Directory allows you to define security very granularly. In the human resource application example, human resource employees have access to user objects that are important for their jobs. And while administrators are able to create and delete users, they do not have access to the salary property.

LDAP as the core protocol for interoperability

To ensure that Windows 2000 supports directory synchronization and interoperability across multiple operating systems and directories, the client access protocol for the Active Directory is the Lightweight Directory Access Protocol (LDAP). Microsoft Corporation is implementing this standard protocol for directory access and is also one of the driving companies behind the standardization process for LDAP within the Internet Engineering Task Force (IETF). Many proposals, such as those for directory replication, have been authored or coauthored by Microsoft. This demonstrates the Microsoft commitment to standards-based protocols and interoperability with other directory vendors.

In the Active Directory, LDAP version 2 and version 3 are implemented for client access. Since the standardization process for replication has not been finalized yet, the Active Directory

implements a proprietary replication protocol in the first version. As soon as the standard is available, later versions may use LDAP for directory replication.

Advantages of Migrating to Windows 2000

Many companies today have a large investment in Windows NT. Therefore, the primary goal for any migration is to protect the customer's existing investment by allowing it to migrate seamlessly and gradually from Windows NT 4.0 to the Active Directory.

Support for mixed environments

Windows NT supports a mixed environment of Windows 2000 Active Directory domain controllers and Windows NT 4.0 domain controllers. Customers can migrate at their own pace, based on business needs. Downlevel clients will think they are accessing Windows NT 4.0 domain controllers. Windows NT Workstation and Windows® 95 clients that do not have the Active Directory access software will be able to log on to Active Directory domain controllers by using Windows NT LAN Manager (NTLM) challenge/response authentication.

This 100-percent backward compatibility allows businesses to migrate their domain controllers first and then migrate their clients, or to migrate a mix of servers and clients. There is never a point in the migration process that requires a mass migration to the new operating system version on either servers or clients. It is also never necessary to take a complete domain offline to migrate domain controllers or clients. Individual domain controllers are unavailable only during their OS (operating system) update. This guarantees that companies can migrate to the Active Directory without interrupting their business.

Simplification of domain models

The Active Directory design allows simple migration of both centralized and decentralized Windows NT 4.0 domain models. The typical master or multiple master domain model can be easily migrated to an Active Directory tree or forest.

The combination of the Active Directory and the improved security model allows customers to reduce the number of domains in the enterprise. The primary reason organizations choose a master domain model is to allow local staff to administer local resource domains without granting these users administrative rights to user accounts in the master domain. This is useful for both the central information technology (IT) departments and local users. Central IT staff do not have to travel to remote locations or perform administrative operations over slow WAN links, and local users receive support more quickly from local support staff. Moreover, local support personnel tend to have a better understanding of the daily processes of their local users.

The origin of many multiple master domain enrollments can be found in the limitations of Windows NT 3.1 domain controllers. In the first release, Windows NT could not hold more than 10,000 objects in the database, which was insufficient for larger companies. Therefore, customers had to create additional account or master domains and establish trusts between these master domains. In the Active Directory, the scalability will be sufficient to store all objects in one domain.

The former structure can be reestablished within a domain using an organizational unit hierarchy. Customers can use the migration to the Active Directory as a means of reducing the number of domains and can thus simplify their network administration and structure.

Control of replication traffic

The Active Directory's improved replication engine allows you to differentiate between replication that happens using a local network connection and replication that happens over a slow WAN connection. It allows you to create *sites*, which are a collection of IP subnets with

good connectivity. Within the same site, replication starts after a configurable deferral time. Between sites, replication is scheduled and can use WAN network bandwidth only at selected times.

In general, replication traffic in the Active Directory is reduced when compared with the same number of objects in Windows NT 4.0. Although the Active Directory defines more objects and more properties per object, the granularity of replication is finer, because replication in the Active Directory happens on a single property level. If you change only one property on an object, only the property you changed will be replicated to the replication partners, not the object as a whole.

The next section of this paper discusses directory service concepts and domain models. Subsequent sections provide detailed instructions and scenarios for planning and completing a migration from Windows NT 4.0 to Windows 2000.

Directory Service Concepts

Pre-Windows 2000 Directory Services Domain Models

The following paragraphs describe domain models in a pre-Windows 2000 installation. These concepts are important to your understanding of subsequent migration issues.

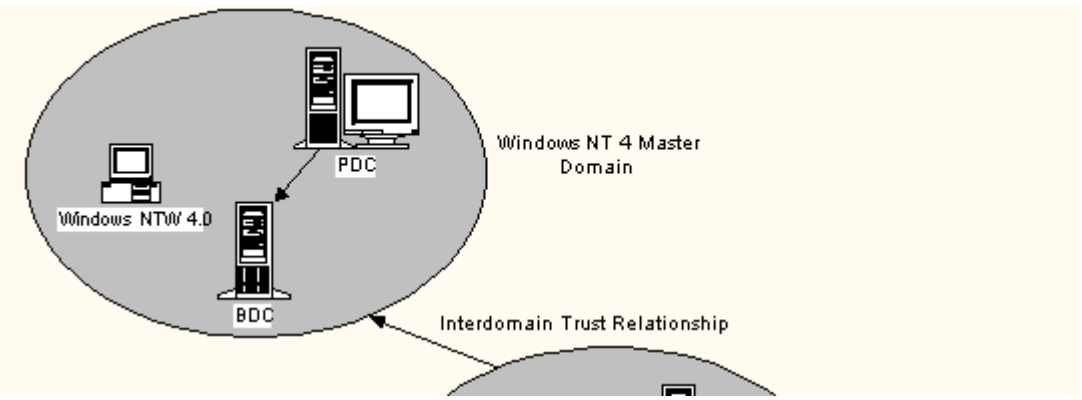
Single domain model

The single domain model is the simplest domain architecture possible in a Windows NT 4.0 topology. In this architecture, there is simply one primary domain controller (PDC) that holds the master copy of the Security Account Manager (SAM) database. In addition, there may be one or more backup domain controllers and several member servers present in the domain.

In this architecture, all user accounts, machine accounts, and resource definitions (such as printer queues and shares) draw security principal definitions from the PDC's SAM database. These accounts are granted rights based on Access Control Entries (ACEs) in ACLs on whatever resource is being shared or restricted. There can, by definition, be only one copy of the SAM database that is modified at any given time, and the SAM database is owned by the PDC.

Master domain model

This model occurs when you partition available network resources into a separate domain space. In doing so, you create a resource domain that houses such resources as printer queue definitions, file shares, and application services (for example, Microsoft Exchange, Microsoft SQL Server™, and so forth). The only difference between a resource domain and a master domain is the placement of Windows NT user and machine account definitions. Security principal definitions are defined in the master domain to simplify administrating the domain model, even though the resource domain is capable of housing them.



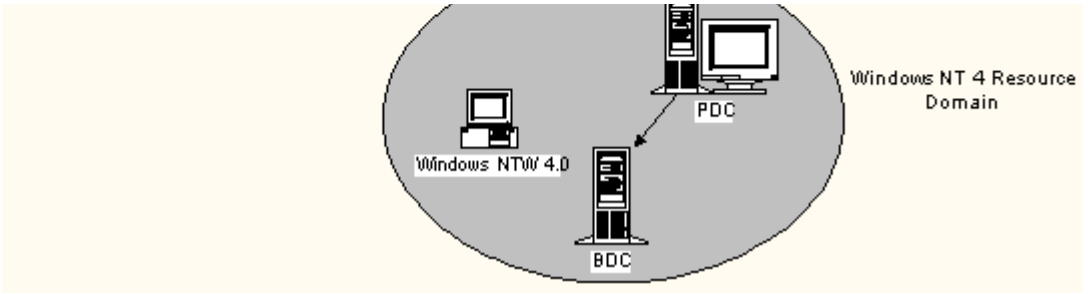


Figure 2. Master domain model with single resource domain

When forming a resource domain, you are required to create a minimum of a single one-way Local Security Authority (LSA) trust relationship to allow the centrally defined security principals access to resources housed in the resource domain. Global groups that have been administratively defined in the master domain can be assigned membership to local groups within the resource domain to further simplify the assignment of rights to resources.

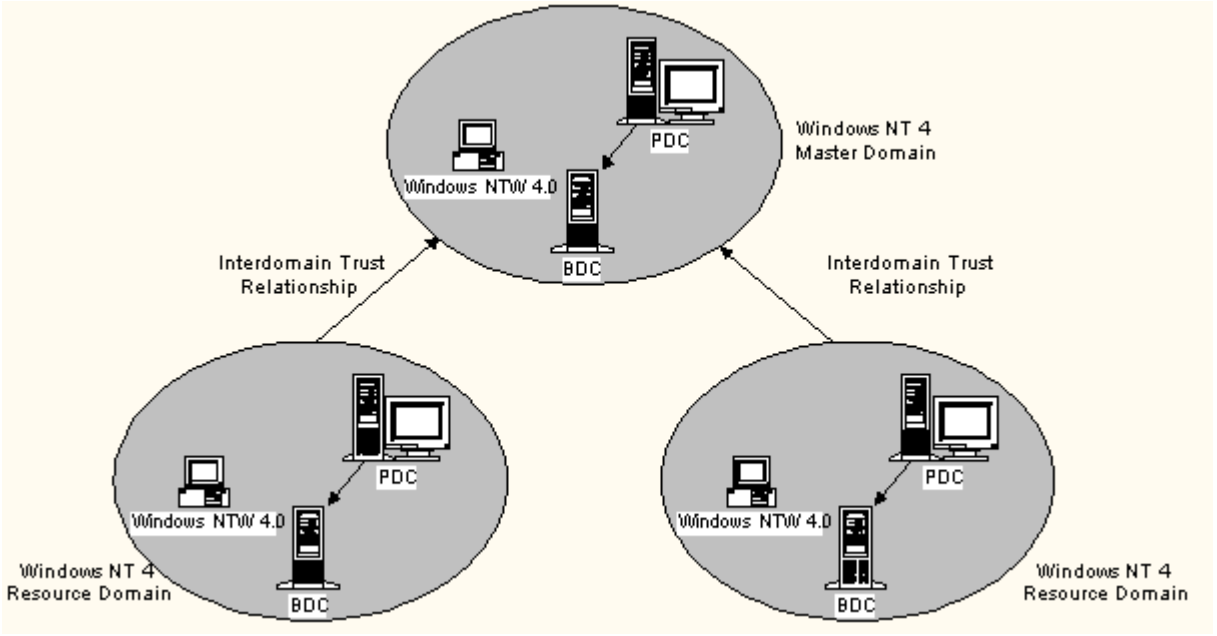


Figure 3. Master domain model with multiple resource domains (local groups)

Multiple master domain model

In the multiple master (or *multimaster*) domain model, the accounts for a given organization either cannot or should not be defined in one master domain. In this architecture, multiple master domains house the security principal definitions for specific pieces of the organization.

Typically, this model is used when there are geographical or organizational boundaries in the corporation or enterprise. In most cases, system administration and security permissions reflect the business organization. This is best summarized with a quick analogy:

Assume that you are an administrator in a very large company. This company has several divisions: distribution, sales, marketing, accounting, production, and administration. You are responsible for accounting resources and users only. It would not be in the best interest of the Company to allow you administrative-level control over any other department's or division's resources. Therefore, your company created a series of Windows NT 4.0 domains that each contain a section of the company's applicable resources. Your administrative rights are limited to the domain that contains accounting resources. You cannot transfer these administrative rights to other domains within the network.

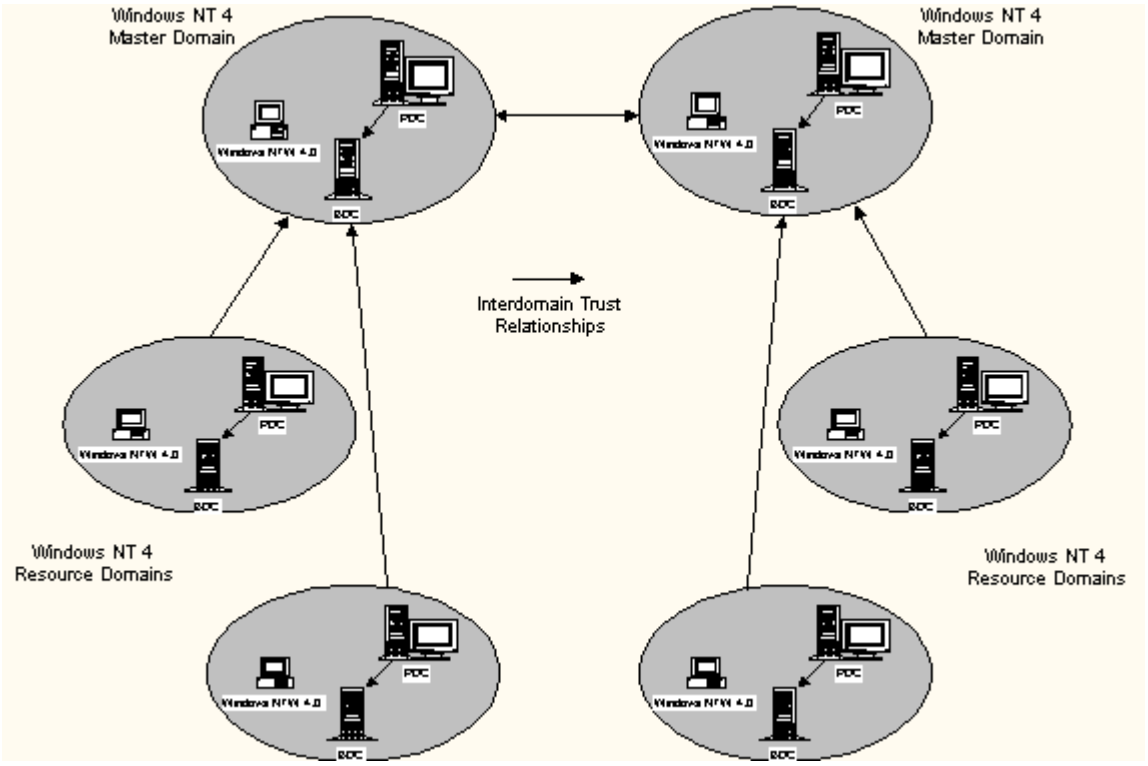
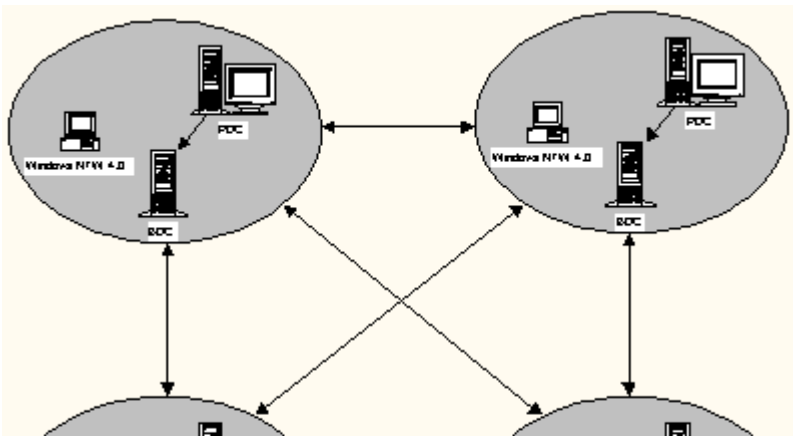


Figure 4. Windows NT 4.0 multimaster domain model

Complete trust model

The Windows NT 4.0 Complete Trust domain model is the most complex to administer and the most costly in terms of network resources. In this architecture, an LSA trust is established between all created Windows NT 4.0 domains such that security principal definitions in any of the established Windows NT 4.0 domains can be granted access to any resource defined in any of the existing domains. Each Windows NT domain has an administrator, and administrative authority can be confined to one of the domains. However, in cases where this domain architecture is used, it is often indicative of one of the following situations:

- The organization is fractured or disjointed and has not taken the time for adequate planning of a simpler topology.
- The organization has complex internal politics that result in multiple groups of administrators whose responsibilities span many areas and may overlap.
- The organization has grown more quickly than expected and many divisions have performed their own Windows NT-based installations.



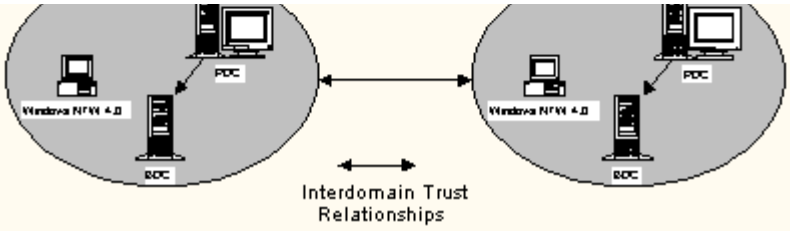


Figure 5. Windows NT 4.0 complete trust model

Windows 2000 Active Directory Domain Model

Domain trees can be viewed two ways. One view is the division of the namespace for the domain tree (this is the physical architecture of the directory). The other view is the trust relationships between domains or trees (this is the logical architecture of the directory).

Active Directory physical architecture

In the Windows 2000 Active Directory, a domain is a partition in the namespace. All domain controllers in the same domain contain the entire directory for the domain and their databases are identical. Replicating objects always happens on the domain level. Domain controllers never replicate domain objects to domain controllers in different domains. This makes a domain both a naming context and a partition in the namespace.

Active Directory logical architecture

An Active Directory tree consists of a hierarchy of domains that have trust relationships to each other. A domain can implement a tree of organizational units within itself, which creates two levels of hierarchies inside the tree—the hierarchy of the domains and the hierarchies of OUs within the domains. The OU hierarchy inside a domain is independent of structure of other domains—each domain can implement its own OU hierarchy.

This two-tier hierarchical structure allows a great deal of flexibility in administrating domain trees. For example, an entire domain tree can be owned and administered by a central IT team. The IT team can create the same OUs in all domains—such as an IT OU where local IT user accounts reside or a technical support OU for support employees. Additional OUs can be formed to meet users' needs in the particular domain.

In the headquarters domain, a human resources and a finance OU can be created. For a regional office domain, an OU for the office sales team can be created. Administrative rights for these particular OUs can be delegated to specific users or groups so that these users can administer their own areas without involving IT. And because these users have administrative rights only on their own OUs, they can never interfere with IT's global rights and responsibilities.

The flexibility in this logical architecture allows organizations to create an environment that mirrors the business's organization. The Active Directory supports a centralized or decentralized business model as well as any combination of the two. For example, you can use the domain structure to provide a centralized framework, and then you can use the OU structure within domains to support decentralized operations.

Characteristics of a Windows 2000 domain

In Windows 2000, a domain is a partition of the namespace where a common security policy applies. A domain's security policy defines how strong the passwords need to be, the password history, the lifetime of Kerberos tickets, account lockouts, and more. When a security principal (user account) is created in a domain, the principal gets a Security ID (SID). A portion of the SID always contains an identifier of the domain, where the SID was originally

issued. This makes it easy to find out which domain contains a user or group and to determine whether to grant access to resources.

A domain is a physical security boundary. Full administrative control is contained within a domain.

Characteristics of an organizational unit

An organizational unit (OU) is a container that can host other objects. If you use the analogy of a file system, an OU is a directory. A directory (or folder) in a file system is a container that holds files or other directories. Similarly, a Windows 2000 OU is a container that holds other objects or other OUs. OUs are used for delegating administrative rights. Objects that are typically stored in OUs are users, groups, printers, or distributed file system (Dfs) shares. The permission to create these objects and change attributes on these objects can be assigned to specific users and groups. This results in an improved granularity of administration.

DNS-style naming

The Active Directory implements a Domain Name System (DNS)-based naming style that is founded on the LDAP proposals. In this naming hierarchy, the single components of the DNS domain names are expanded to DC (domain component) entries. For example, for an Active Directory domain that carries the DNS name *sales.microsoft.com*, the DNS-style LDAP name is *DC=sales,DC=microsoft,DC=com,o=Internet.*

The use of the DNS naming style allows enterprises to leverage an existing DNS namespace and, if available, to use the already registered DNS domains to register the directory service in the Internet.

Directory objects—the schema

The schema in the Active Directory defines what objects and properties can be created in the directory. When the Active Directory is installed on the first domain controller in a forest (a *forest* is a set of trees that share a common schema, configuration, and global catalog, with Kerberos trust among all members of the forest—see the section "Trees and Forests," later in this document), the directory service creates a default schema. This schema includes all objects and properties that are required for the directory service to work and is replicated to all domain controllers that join the forest later.

Any directory service contains comprehensive information about users and objects in an organization. With the Active Directory, the fault tolerance created by the replication model and the extensibility of the database makes the directory a great place to store information that can be used by directory-aware applications. One example is a human resources (HR) application. The directory already includes a great deal of information about users—such as their first and last names, their office numbers, their phone numbers, and perhaps their home addresses. While all of this information is useful to a human resources application, additional information, such as the employee's salary, social security number, tax withholding information, and health insurance information, would need to be added.

The Active Directory allows you to extend the schema to create new properties and classes for all of the information you may want to add. New classes can be derived from existing classes and can inherit all properties from the previous classes. New properties can be created, and these properties can be added to classes. Properties in classes have either "must" attributes or "may" attributes—that is, they are either required or optional. Required properties (those with "must" attributes) are required to obtain a value when a new object is created. These properties can later be changed, but they cannot be deleted. For example, a user object must contain a common name (cn), a **SamAccountName** used for backward compatibility, and a password.

Optional properties (those with "may" attributes) can be added or changed at any time. These properties are not required for the directory service to work, but they hold additional information that is useful for system administration or for other users in the enterprise. Examples include phone numbers, office numbers, and a manager attribute.

For example, suppose that you need to distinguish between employees and contractors who need network access. You derive a new user class, *AcmeUser*, specifically for full-time employees. You also determine that to support your HR application, you need to add salary and social security number properties to the schema. You then add the newly defined salary and social security properties to the AcmeUser class as "may" attributes. The security granularity of Windows 2000 allows you to grant read and write access to these properties to members of the HR department only. The individual user has read access to his or her data. Administrators do not have access to these attributes.

The Active Directory data store

The data store for the Active Directory is the Extensible Storage Engine (ESE). ESE is an improved version of the Jet database that was used in Microsoft Exchange 4.0 and 5.0, and is the same database engine that will also be used for the new versions of Microsoft Exchange.

The improved storage engine allows you to create a database up to 17 terabytes in size. The database can hold up to 10 million objects. Note that ESE reserves storage only for those properties that actually have a value. For a user object, the default schema predefines approximately 50 properties. If you create a user and set only four properties—such as first name, last name, common name, and password—for that user, the database uses space for these four attributes only. If you add more values later, the database dynamically allocates storage for the data.

ESE can store multivalue properties. This feature allows the storage of multiple values for a single property—for example, the database can store multiple phone numbers for a single user without requiring multiple phone number attributes.

Active Directory replication

The Active Directory uses multimaster replication. The Active Directory does not distinguish between primary and backup domain controllers, where changes in the domain database can only be performed on one specific domain controller. Instead, it simply uses domain controllers (DCs), and all domain controllers are peers. Objects can be created or manipulated on any domain controller, and changes are then propagated to the remaining domain controllers. Note that while this approach is conceptually simpler, it does require a means for transferring data among domain controllers and for reconciling contradictory information set among the different domain controllers.

Replication in the Active Directory is not based on time, but on Update Sequence Numbers (USNs). Each domain controller holds a table containing entries for its own USN and the USNs of its replication partners. During replication, the domain controller compares the last known USN of its replication partner (saved in the table) with the current USN that the replication partner provides. If there have been recent changes (that is, if the replication partner provides a higher USN), the data store requests all changes from the replication partner. This is known as *pull replication*. After receiving the data, the directory store sets the USN to the same value as that of the replication partner.

If properties on the same object are changed on different domain controllers, the domain controllers reconcile the data as follows:

- **By version number**. All properties carry a version number, which is used to determine which property should be declared as the correct one. The Active Directory always uses the higher version. This is not always the correct solution; however, the use of an

unequivocal algorithm ensures that reconciliation can be performed locally without negotiating with the replication partner and that it always results in the same data being used on all domain controllers.

- **By timestamp**. If the version numbers on the changed property are the same, the domain controller uses a timestamp to reconcile the data. The timestamp is always created with the property and the version number. The attribute with the latest timestamp is used. Domain controllers assume that time information is accurate. (They do not negotiate the time.) Again, this is not always the correct solution; however, the use of this algorithm ensures that the domain controllers continue serving the clients rather than performing lengthy time negotiations.

- **By buffer size**. If both the version number and the timestamp are the same, the domain controller performs a binary memory copy operation and compares the buffer size. The higher buffer size wins. If the two buffers are equal, the attributes are binarily the same, and one can be discarded.

Note All reconciliation operations are logged, and administrators have the option of recovering and using the rejected values.

Sites and domains

The *site* concept has been used by various BackOffice® applications to minimize replication traffic over slow WAN links. Unfortunately, the site concepts used in the different BackOffice applications do not match. Windows 2000 and the Active Directory introduce a new site concept that is not optimized for the needs of a specific application, but uses the underlying IP network to determine locations where a good network connection is available. Eventually, all BackOffice applications will evolve to this site concept.

An Active Directory site is a combination of one or more IP subnets. The administrator can define these subnets and can add subnets to a site. Sites support two features:

- They optimize replication traffic over slow WAN networks.

- They help clients to find domain controllers that are close to them.

Replication within a site and between sites follows different topologies. Within a site, a domain controller postpones notification of recent changes for a configurable interval. The default value is 10 minutes. Unlike Microsoft Exchange, the Active Directory allows you to manipulate the replication topology within a site. The directory store creates a default replication topology, which consists of a bidirectional ring. You can change this topology and create another architecture (for example, a star). However, the directory service will always make sure that the topology is not broken and that no domain controller is excluded from the replication process. The Knowledge Consistency Checker (KCC)—a process that runs on all domain controllers—checks this. If the replication topology is broken, it is automatically fixed by the KCC.

Clients can use site information to find domain controllers or resources that are close to them. The concept of finding close domain controllers and resources helps to minimize network traffic over slow WAN links.

When a clients starts a logon process, the first information a client receives from a domain controller is its site membership, the domain controller's site membership, and whether the domain controller is the closest one to the client. If the domain controller is not the closest, the client can query for a domain controller in its own site and can then communicate with the closer domain controller only. (The client saves the site information in the registry and can use it later to talk to a close domain controller or to find close resources.)

If a workstation is moved to a different location, the workstation uses its old site information

to locate a domain controller. The domain controller then tells the client that it is no longer the closest domain controller and provides the new site information to the client. The client can use the new site information to query DNS for a closer domain controller.

Global catalog servers

Another new concept in the Active Directory is the *global catalog* (GC). The global catalog holds all objects from all domains in the Windows 2000 directory, and a subset of each object's properties. Internally, the global catalog implements the same hierarchy as the domain tree does. LDAP queries, however, usually return results in a flat record set or list. This allows the global catalog to be used as a repository that functions like a global address book (and is comparable to the Microsoft Exchange Global Address book). Global catalogs can be used for tree-wide searches. If all information can be found on a global catalog, no LDAP referrals to other domain controllers need to be created.

It is a good idea to have at least one global catalog in each site. By doing so, clients will always have a local repository for search operations.

Contiguous and disjointed namespaces

In an LDAP directory, the namespace can be organized in either a contiguous or a disjointed namespace.

In a contiguous namespace, the name of a child domain always contains the name of the parent domain as a part of its name. For example, if an Active Directory domain with the LDAP name *DC=Sales,DC=Microsoft,DC=com* is a child of *DC=Microsoft,DC=com*, a contiguous namespace was used. The name of the parent domain can always be constructed by removing the first part of the child domain name.

In a disjointed namespace, the names of the parent and child domains are not directly related to each other. An example is the Active Directory domain *DC=MSN,DC=com*, which should be a child domain of *DC=Microsoft,DC=com*. In this case, the child domain does not carry the name of the parent domain as part of its own name.

The use of a contiguous or a disjointed namespace affects LDAP search operations. In a contiguous namespace, a domain controller always creates referrals to the child domains. Using a disjointed namespace, however, terminates the search operation—referrals are never created.

The use of both contiguous and disjointed namespaces within the same tree led to much confusion about how search operations really work in the tree. For that reason, the tree model was refined for the Active Directory and introduces the concept of trees and forests, which is explained next.

Trees and forests

In the Active Directory, a tree is defined by:

- A hierarchy of domains
- A contiguous namespace
- Transitive Kerberos trust relationships between the domains
- A common schema
- A common global catalog

A forest is defined by:

- One or more sets of trees
- Disjointed namespaces between these trees
- Transitive Kerberos trust relationships between the trees
- A common schema
- A common global catalog

Figure 6 shows three subtrees that implement one forest. The DNS name of the root domain of the left subtree is Microsoft.com. The LDAP name of the Active Directory domain could be *DC=Microsoft,DC=Com,o=Internet*. The LDAP name of the root domain in the second subtree could be *DC=MSN,DC=Com,o=Internet*. The namespaces are disjointed.

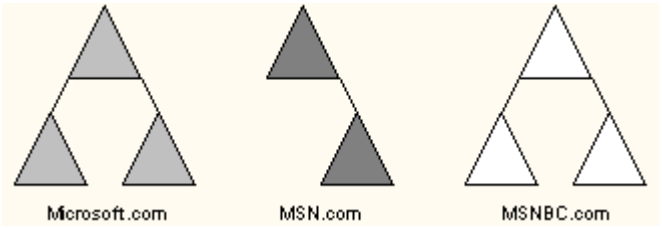


Figure 6. Example forest and tree structure

Subdomains in all trees implement a contiguous namespace. The LDAP name for a sales domain within Microsoft would be *DC=sales,DC=Microsoft,DC=com,o=Internet*.

The concept of contiguous trees combined in a forest makes it much easier to understand how search operations work within the forest or within the subtrees. Security principals are still valid within the forest. It is possible to form virtual teams by creating groups that have members from domains living in different trees within the forest. Trees can join a forest at any time. This helps to implement both grassroots deployments and enterprise merges.

Tree metadata

The tree metadata contains all information needed for a tree or a forest to work. The metadata is built of two containers: the *configuration* container and the *schema* container. Each container implements its own naming structure and, by doing this, its own replication topology.

The configuration container is the information that really glues the trees in a forest together. The configuration information includes the available domains in the forest, the sites, and all domain controllers. Whenever a domain controller joins a domain, the configuration information must be updated and replicated.

Tree and forest management

One of the Active Directory's primary design goals is easy administration and flexibility of the resulting domain trees and forests. For example, if a newly formed company designs trees and forests to suit its initial size and company structure, it is unlikely that this first implementation will be the best long-term solution. As the company grows, its needs and organization will change. The IT department may require a refined naming convention later, or the hierarchy of the trees and forests may cease to match the organizational structure. For that reason, the Active Directory implements a set of operations that allows renaming and restructuring of objects in the directory. These operations include:

- Easy addition of domains
- Easy deletion of domains
- Renaming of domains
- Rearrangement of the domain trees and forests
- Merging of trees in a forest

Of the operations listed, the easiest is the addition of a domain. An Active Directory domain can join a domain tree when its first domain controller is installed. The domain controller must be pointed to an existing Active Directory domain as its parent domain. This establishes the transitive Kerberos trusts and allows the domain to join the tree.

A domain deletion is not really a deletion; it merely removes a domain from a tree. The Active Directory allows removal of domains from a tree at any time; however, the domain cannot have any child domains that are supposed to remain in the tree. Removing parent domains breaks the trust relationships between the parent of the domain to be removed and the child domain of the domain to be removed. Note that if the child domain is to be removed together with the parent, a recursive deletion is possible.

Any object in the Active Directory can have several names—a common name, a relative name, and so forth. The only object identifier that can never be changed is the object's Globally Unique Identifier (GUID). The GUID is a very large number that is created by the domain controllers. The algorithm used for GUID creation ensures that a GUID can never be created twice. By using this unique identifier as the only identifier that can never be changed, the Active Directory allows all other names to be changed. For this reason, it is easy to rename any object or domain in the Active Directory.

Using GUIDs also allows objects, such as domains, to be moved in the directory tree or forest. During partial replication, the global catalog servers receive a subset of the properties. The GUID is included in this subset. If an object is moved, the global catalogs can use the GUID to locate the object and can construct the distinguished name of the object using the object's new relative ID (RID) and the LDAP path to the domain where the object now resides.

If the Windows 2000 Dynamic DNS server is used, when a domain is renamed or repositioned in the tree of forest, the administrative tools automatically remove the old DNS entries for the domain and the domain controllers from DNS. Then the tools use the new domain names to publish the new entries in DNS. If a UNIX DNS server is used, the administrative tools create files that include both the records that must be removed and the records that must be added. In addition, for Windows 2000 workstations, the TCP/IP configuration is changed automatically—administrators do not have to touch each machine to enter the new domain name in the TCP/IP configuration.

Sample Migration—Single Domain Update

Domain Controllers in Windows NT 4.0 and Windows 2000

At installation, Windows NT Server 4.0 requires you to decide whether you are installing a primary domain controller in a new domain, a backup domain controller in an existing domain, or a member/standalone server. The member/standalone server never receives a copy of the domain database and cannot act as a logon server. Much of this initial assignment can never be changed during the server's lifetime. A backup domain controller (BDC) can be promoted to a primary domain controller (PDC), and a PDC can be demoted to BDC.

However, a member or standalone server can never become a domain controller and a PDC or

BDC can never become a member or standalone server (although you can disable the net logon service causing the server not to validate logins). In addition, Windows NT 4.0 extensively uses machine security identifiers and Windows NT 4.0-style trusted relationships when constructing the domain namespace. Because of this heavy dependency on machine and domain security identifiers—which are defined at the time of machine installation and cannot be changed—PDCs and BDCs cannot be moved from one domain to another without reinstalling the operating system. Moreover, domain controllers and domains cannot be arbitrarily renamed without reinstallation.

In Windows 2000, installing the directory service is separate from the core operating system installation. In Windows 2000, installing and executing the Directory Service Agent (DSA) determines whether a server is a domain controller. This service is similar to existing services in Windows NT 4.0—it is installable and can be removed.

However, since the directory runs as a part of the trusted domain on the server and is thus part of lsass.exe, the directory service does not appear as a service in the Control Panel. It cannot be configured to start manually, and it cannot be stopped and restarted like other services. Installing and executing the DSA service on preexisting Windows 2000 servers changes their role in the domain namespace without requiring you to reinstall the core operating system. This means that domain controllers can be added from existing servers as needed and moved from domain to domain.

Roles in a Windows NT 4.0 Domain

Figure 7 shows a small Windows NT 4.0 domain that consists of a primary domain controller, two backup domain controllers, and two Windows NT Workstation products.

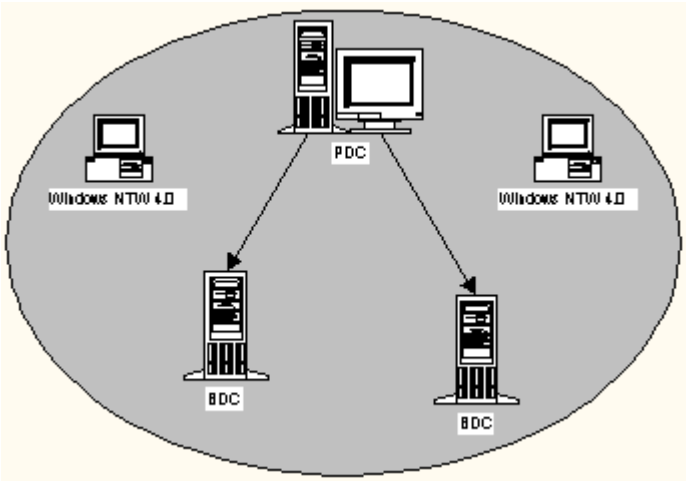


Figure 7. Small Windows NT 4.0 domain

The primary domain controller is the only domain controller that has a readable and writeable copy of the domain database. If a network administrator wants to make changes to the domain, such as adding new users or machines, the PDC must be online. However, all domain controllers can serve as logon servers. Because the PDC is the only domain controller that has the original of the domain database, the PDC must make sure that all other logon servers have a copy of the most recent version of the database. To accomplish this, the PDC uses a specific replication protocol—the single master replication protocol—to communicate changes. This protocol allows only one master to make changes and then replicates those changes to all backup domain controllers.

Primary Domain Controller Update

When a Windows NT 4.0 domain is migrated to the Active Directory, the first domain controller to be updated is always the primary domain controller. This approach offers two advantages:

the domain can immediately join the tree (even if it is still a mixed domain) and the administrators can immediately start to use the new administration tools and create Active Directory objects.

After its upgrade to Windows 2000, the PDC is able to represent itself as a Windows 2000 domain controller (DC) to other Windows 2000 servers and as a Windows NT 4.0 PDC to Windows NT 4.0 servers. In this respect, Windows 2000 is fully backward-compatible and can play any role in a domain. The PDC can still be used to create new security principals and to replicate these changes to the Windows NT 4.0 BDCs, the Windows NT 4.0 BDCs still recognize the Windows 2000 PDC as the domain master, and workstations can use the PDC as a logon server.

While the Windows 2000 PDC now uses the directory store to save objects, this does not affect replication and logon operations, because the directory store exposes the data as a flat Windows NT 4.0-style store to downlevel clients. If the Windows 2000 PDC is offline or otherwise unavailable and no other Windows 2000 BDCs exist in the domain, then a Windows NT 4.0 BDC can be promoted to PDC. If, however, a Windows 2000 PDC is operational, no Windows NT 4.0 BDC can be promoted to PDC.

Once the Windows 2000 DC is running, there are two ways to proceed. The first is to upgrade all other Windows NT Server products to Windows 2000. The second way is to install the Active Directory on the PDC and leave the other servers as Windows NT 4.0 servers. Either approach is acceptable, but perhaps the best strategy is a compromise. Multiple Windows 2000 DCs can provide fault tolerance for Active Directory information. If one machine fails, another Windows 2000 DC with the Active Directory installed has a copy of the information. And maintaining a Windows NT 4.0 BDC guarantees a fallback if problems with replication between Windows 2000 and Windows NT 4.0 machines occur.

Joining a Tree

After the installation of the core operating system on the PDC, you can install the Active Directory Service on this machine. Note that the setup routine requires you to decide whether you want to join an existing tree or create a new tree. If you choose to join an existing tree, you must provide a reference to the desired parent domain.

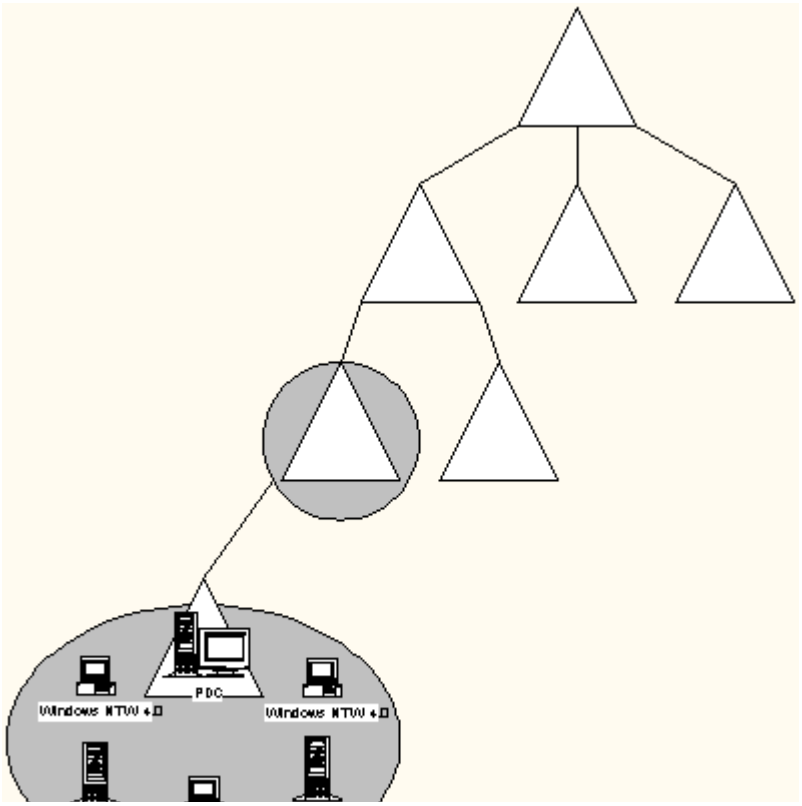


Figure 8. Joining a tree

The Active Directory setup then installs all necessary components on the domain controller, such as the database and the Kerberos authentication software. The existing Security Account Manager (SAM) objects are copied from the registry to the new database store. These objects are the security principals (user accounts, local and global groups, and machine accounts). Once Kerberos is installed, the setup routine starts the authentication service and the ticket granting service, then establishes a two-way transitive Kerberos trust relationship to the parent domain. Eventually, the domain controller from the parent domain replicates all schema and configuration information to the new child domain. The new domain is added to the domain and site structure, and all domain controllers receive the notification that a new domain joined the tree.

From this moment on, the migrated domain is a fully functional member of the Active Directory domain tree. Clients that already use the Active Directory client software, such as Windows 2000 workstations, Windows 98-based (previously code-named *Memphis*) workstations, and Windows 95-based machines that have the new client access software installed, can benefit from the transitive trusted relationships that exist within an Active Domain tree and allow users to access resources from all over the domain tree. These clients can find objects in the directory by querying domain controllers and global catalog servers. (Note that instead of NetBIOS, DNS is used as locator service.) Downlevel clients, however, still view the domain as if it were a Windows NT 3.*x* or Windows NT 4.0 domain.

Administration tools, such as those used for assigning file ACLs or share permissions, function as they did before and can only use the Windows NT 4.0-style nontransitive, one-way trust relationship that the domain established before migration started.

Administration Tools in Windows 2000

As soon as the PDC is migrated to Windows 2000 and the Active Directory is installed on the domain controller, the administrator can start using the new tools that come with Windows 2000 to create new objects that only exist in the Active Directory, such as organizational units.

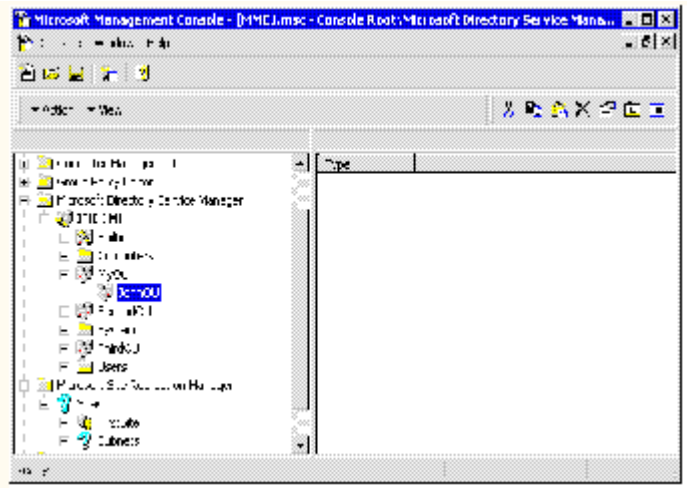


Figure 9. Active Directory Administrative Tools (based on the Microsoft Management Console)

The Active Directory allows you to create a tree of organizational units (OUs) within a domain. Organizational units are containers that can hold objects such as users, groups, or resources,

and allow you to organize domains that hold many objects—whereas your only previous option was to create resource domains.

You can immediately start to create these special objects in the domain, even if the domain is still mixed and contains downlevel domain controllers. Note that the OU structure is only visible for machines that have the Active Directory client software installed. For all other machines, OUs do not appear. If a downlevel client queries a domain controller for security principals, the domain controller exposes all objects as a flat store so that these clients find objects at the domain level, no matter in which OU the objects actually reside.

Backup Domain Controller Updates

In the next stage of the migration process, additional domain controllers can be converted to Windows 2000 and the Active Directory can be installed on these machines, or new Windows 2000 domain controllers can be added to the domain.

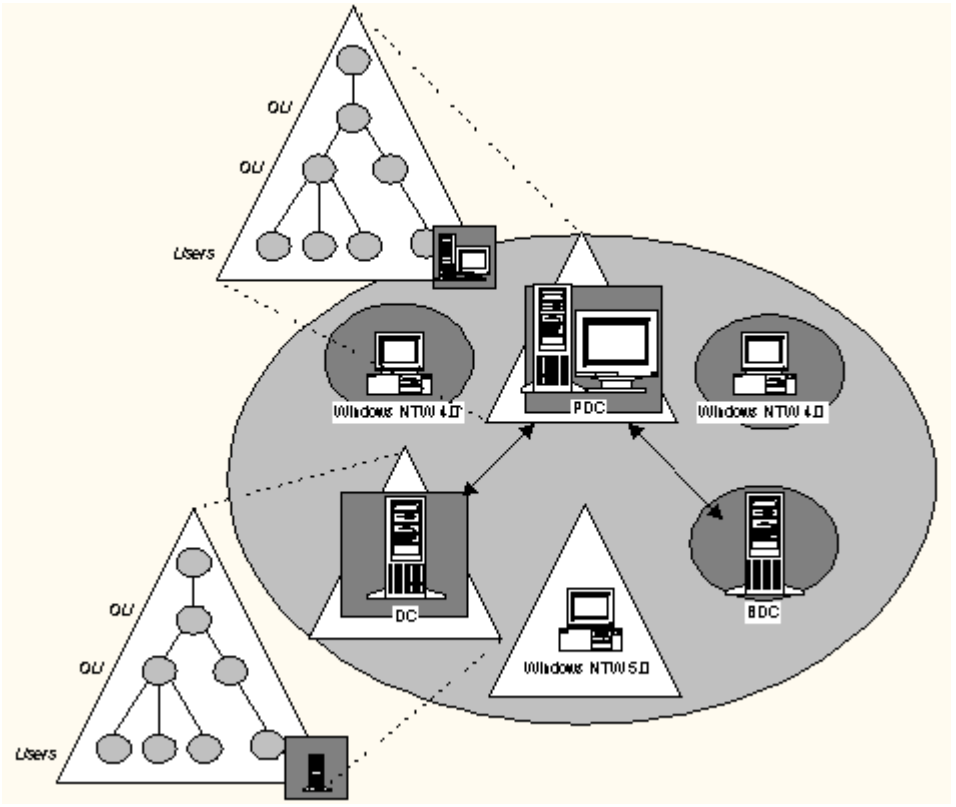


Figure 10. Updating additional domain controllers to Windows 2000

Since downlevel domain controllers rely on using consecutively incremented relative IDs (RIDs) in security IDs (SIDs), the Windows 2000 PDC is the only machine that can create security principals. However, nonsecurity principals, such as organizational units, can be created on each Active Directory domain controller.

The PDC now uses two replication protocols, depending on who the replication partner is. When talking to Active Directory domain controllers, the PDC uses the new replication protocol; when talking to downlevel domain controllers, the PDC uses NTLM replication protocol.

Finalized States—Migration Complete

Once all domain controllers are converted to Windows 2000 and have the Active Directory installed, the domain can be switched from a mixed domain to a pure Active Directory domain. Several things change during this switch:

- The domain now uses the Active Directory multimaster replication protocol, even for security principals.
- The former PDC turns off the Windows NT 4.0 downlevel replication support. (From this point on, it is no longer possible to add Windows NT 4.0 domain controllers to the domain.)
- Downlevel clients now benefit from transitive trusts.

The network administrator must initiate the switch; it cannot be performed by the directory service automatically. Only the administrator can decide that he or she never wants to add a Windows NT 4.0 domain controller to the domain again.

The Active Directory and Windows 2000 servers use SIDs to identify security principals (just as previous versions of Windows NT did). However, while there is no difference in using SIDs on the operating system level, the security principals are created differently. In former versions of Windows NT, only the PDC was able to create security principals, and all BDCs expected consecutively incremented RIDs. The Active Directory, however, uses a multimaster replication protocol that allows each domain controller to create objects—such as security principals—and replicate new objects or changes in existing objects to all other domain controllers. Using a multimaster protocol prohibits the use of consecutive numbered RIDs because all domain controllers participate in an RID assignment, each domain controller working with its own allocation of RID numbers.

When the Windows 2000 PDC is installed, the PDC owns all RIDs in the domain. As subsequent domain controllers are migrated or new Windows 2000 DCs join the domain, they each become owners of the RID pool by requesting it from the current owner. (Each DC can query the directory store to determine the current owner.)

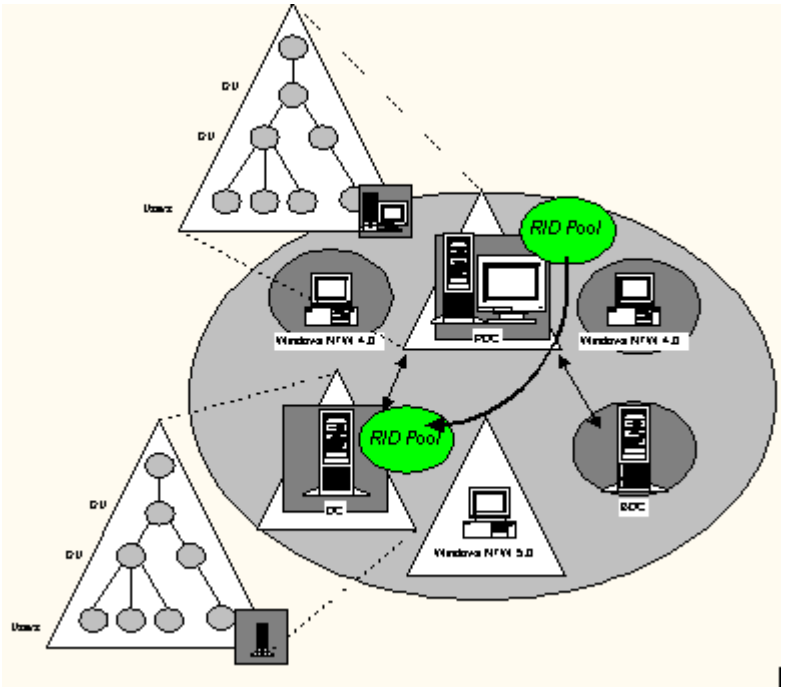


Figure 11. Transfer of RID Pool

The RID pool is allocated to the requesting domain controller, which becomes the new role owner. Once a DC has acquired role ownership, it is then free to carve off a block of RIDs (currently set to 100K) from the domain-wide RID pool. When a domain controller creates a new security principal, it uses one RID from its pool. The server then creates the principal and replicates the new object to its replication partners. Whenever a DC reaches a threshold of

RID pool exhaustion, it acquires the domain-wide RID pool again and removes another amount of RIDs for its personal use. Using an RID pool and different blocks of RIDs being generated by different DCs results in nonconsecutive RID use.

Active Directory domain controllers are aware of using nonconsecutive RIDs and do not have a problem when nonconsecutive RIDs are used. However, only Active Directory domain controllers can use this protocol for creating and replicating security principals. Downlevel domain controllers are confused when replication partners offer them security principals with nonconsecutive RIDs. As long as downlevel DCs are part of the domain, the old-style, master-slave replication protocol must be used. As a result, in a mixed or downlevel environment, only the PDC can create security principals. Even if all domain controllers are migrated to Windows 2000, the domain uses the master-slave protocol until the administrator declares that the domain is fully migrated. From that point on, the new multimaster protocol is used, and the domain will not allow a Windows NT 4.0 domain controller to join the domain.

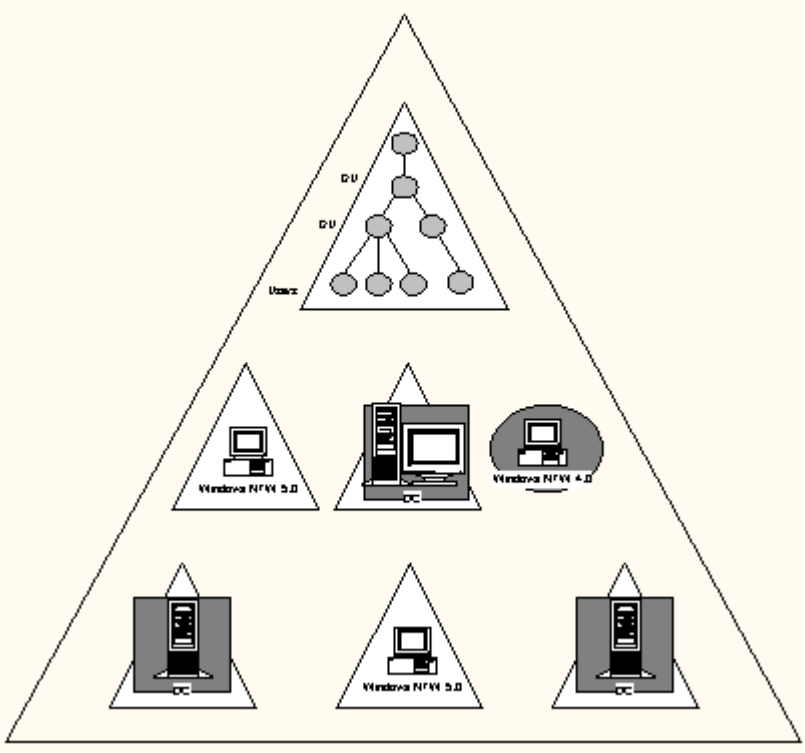


Figure 12. Fully migrated Windows 2000 domain

Clients are not aware of any changes in the domain. Downlevel clients still see the domain as a Windows NT 4.0 domain and can use all tools they used before the migration started. However, downlevel clients do benefit from the domain tree structure. Because the Kerberos trust relationships are transitive, once the domain is switched to a fully migrated domain where all DCs are guaranteed to be Kerberos-aware, downlevel clients can access resources from all over the tree. Although these clients are not Kerberos-enabled, the passthrough authentication provided by the DCs allows the clients to be authenticated in any domain in the tree. Thus, even users of downlevel clients can access resources from all over the tree.

Ensuring a Safe Migration

Network administrators are usually very concerned when faced with an operating system upgrade. In particular, administrators may be concerned about upgrading the primary domain controller first. For that reason, the following examples demonstrate methods for performing a migration without involving the production environment in the critical phase.

Method 1: Install bdc first and save the machine

This example requires installing a new Windows NT 4.0 backup domain controller before starting the migration process. Alternatively, if there are more domain controllers available to perform the client logons, it is also possible to use an existing BDC. This machine will be a reserve machine that you can use if you experience problems during the migration process. This BDC must have a copy of a pure Windows NT 4.0 domain database that was never touched by a Windows 2000 machine. Before starting the migration process, remove the new BDC from the production environment or simply turn it off.

Begin migrating your network as explained previously in this document. If, during the migration, you experience problems with the Windows 2000 PDC or Windows 2000 backup domain controllers, remove the problematic Windows 2000 machine(s) from the production environment. At this point, the domain will not have a PDC, which means that if there are still operational Windows NT 4.0 BDCs, clients can still log on, but the domain database cannot be changed.

Reinstate the "saved" Windows NT 4.0 BDC into the production network and promote it to be the new PDC in the domain. The new PDC will replicate its untouched Windows NT 4.0 database to all BDCs, and thus return the domain to its previous well-defined state.

The drawback to this solution is that all changes made between the time the BDC is removed and brought back will be lost. However, when the domain is in a well-known and stable state, you can periodically turn the BDC on and off again to replicate changes from the PDC. By using two BDCs, you assure that there is one domain database that is never touched by Windows 2000 and one database that is updated on a regular base.

Method 2: Remove pdc from the network first

A second method is to remove the PDC from the production environment before starting the migration. If your goal is to form a domain tree using multiple domains, you could bring all PDCs into a lab or a special environment that is not part of the production network. With this method, you can migrate the PDCs and form the tree without affecting your network.

After you have migrated the PDCs (as explained earlier in this document), you can begin testing the system by gradually adding workstations and/or backup domain controllers to the domain tree. If all tests deliver good results, the PDCs can be returned to the production environment and turned on. The tree will continue to function, and the remaining machines in the domains can be updated to Windows 2000 or the new client access software can be installed on downlevel clients.

If a domain tree already exists, you can bring one domain controller from the existing tree to the established lab environment and have all other PDCs from the domains join the tree. The Windows 2000 domain controller provides schema and configuration data from the existing tree and can later replicate the existence of new domains in the tree to all other domain controllers in the tree.

Advantages of this method are that the production environment remains uninfluenced until the PDCs and the domain trees have been proven stable—it is possible to form the entire tree without affecting the production environment. A disadvantage is that it is not possible to make changes to the domain databases until the PDCs are once again online in the production environment.

Migrated Objects

During the installation of the Active Directory on the PDC, the setup program starts to fill the domain controller's new database with objects. These objects come from two different sources: the first group of objects includes those required by the Active Directory—such as site and domain objects; the other group of objects includes the security principals that were created on the Windows NT 4.0 PDC. Setup copies these objects to the new database. The

Active Directory then uses the new database as the sole data store—the Security Account Manager (SAM) database, which is housed in the registry, is not used after the migration.

Users, groups, and machines

All security principals are migrated from Windows NT 4.0 to the Active Directory, which has separate containers for storing and organizing data. This differs from the Windows NT 4.0 domain organization. In Windows NT 4.0, all objects are organized in a flat list. Users are created on a domain level only—there is no means to structure containers to hold user objects.

Windows NT 4.0 distinguishes between *local* and *global* domain groups. Those domains that have a trusted relationship established with another domain can use global domain groups from this other domain. Local groups can only be accessed from within the domain. Note that local groups can contain users from the same domain, users from trusted domains, and global groups from trusted domains. However, global groups can only contain users from the same domain.

The Active Directory has only one kind of group, and groups are always accessible from all over the domain tree. Groups can contain any users from the domain tree and any other groups. This allows you to nest groups in groups, and thus provides much more flexibility in administrating Active Directory trees. Note that as long as your network is a mixed environment (that is, it has not been fully migrated to the Windows 2000 Active Directory), nested groups are not available. However, in a mixed environment, the Active Directory domain is fully compatible with Windows NT 4.0 domains so that global groups can still be members of local groups.

The Active Directory setup program creates special containers in the new data store to hold the converted security principals. These containers are not organizational units; they are addressed as cn (common name) objects. For all user accounts, the name of the created container is **cn=users**; computer accounts go to the **cn=computers** container.

Windows NT 4.0 groups are migrated to different containers, depending on the nature of the group. The Windows NT 4.0 built-in local groups, such as Administrators or Server Operators, are copied to a container called **cn=builtin**. Windows NT 4.0 global groups appear in the **cn=users** container. All other groups, no matter whether they are local groups or global groups created later, are migrated to the **cn=users** container.

Security Policies

In Windows NT 3.*x* and 4.0, domain policy is stored in the SAM database on the account domain object. Domain policy consists of a security descriptor granting access for operations (such as creating accounts and enumerating accounts), properties describing required password properties (such as minimum length, password history length), and lockout properties. Local policy is stored in the LSA policy database and consists of privilege accounts and auditing information. Local policy is replicated between domain controllers, so it is not possible to have different audit settings or privileges between two domain controllers. Domain policy applies to the whole domain and is not shared between domains.

In Windows 2000, security policy has a much more powerful implementation. Windows 2000 allows you to set security policies that apply to the entire directory service. Individual domain controllers can have different policies than other domain controllers within the same domain. You can set policy in one place and affect all servers or workstations in a domain. And the finer granularity of access control and improved delegation removes the need for an all-powerful administrator.

In Windows 2000, policy is stored in two types of policy objects in the directory service: *local* policy objects and *domain* policy objects. Domain policy objects store the following information:

- Password policy
- Lockout policy
- Security descriptor for domain operations
- Kerberos logon policy (ticket renewal options)
- Encrypting File System policy

A domain can either create its own domain policy object or store a replica and a link to a domain policy object elsewhere in the tree. If it replicates from another domain, the global catalog server must handle replication between domains. The link to a domain policy object is stored in the domain object.

Local policy objects store the following information:

- Auditing policy
- Privilege policy (what accounts receive which privileges and system access)
- Local roles (what accounts receive the built-in domain SIDs in their token)

Privilege policy is also stored in link tables in the directory service.

Since Windows 2000 and the Active Directory use a very different model for security policies than the model used in previous versions, the old information from the SAM database can no longer be used. If the migrated domain joins a tree, the Local Security Account manager copies the domain security policy object from the parent domain and enforces this. If the domain creates a new tree, a default object with default security is created.

Planning the DNS Architecture

This chapter highlights the basic considerations for migrating to Windows 2000 Domain Name Service (DNS) and assumes a basic knowledge of DNS. A complete planning guide for designing and maintaining a domain name server environment for Windows 2000 will be published in the near future.

Windows 2000 uses DNS (Domain Name System) as the name resolution method to access the LDAP-based Windows NT Active Directory. Every Windows NT domain is represented by a DNS domain name. A typical Windows 2000 domain name is *develop.microsoft.com*. Note that prior to Windows 2000, domains were represented by NetBIOS names and the old NetBIOS names remain after migration for access by older clients. However, in Windows 2000, DNS is the required name resolution protocol.

Because each Windows NT domain has a corresponding DNS domain representation, you can use DNS to contact the directory servers. In addition, any client or server can have a DNS name.

Windows NT requires a DNS server, which supports the service locator record type defined in RFC 2052 (SRV records). SRV records are a generalization of the MX record concept in which several different servers can advertise a similar service. In the Windows NT case, the SRV record points to a domain controller.

The format of an SRV record is:

Service.Proto.Name TTL Class SRV Priority Weight Port Target

Where:

- *Service* is the symbolic name of the desired service as defined in Assigned Numbers or locally.
- *Proto* is the protocol. Presently, TCP and UDP are the most useful values for this field, although any name defined by Assigned Numbers or locally can be used.
- *Name* is the domain name for this record.
- *TTL* is the standard DNS time to live.
- *Class* is the standard DNS class IN.
- *Priority* is the priority of this target host (this is similar to an MX record). A client must attempt to contact the target host with the lowest-numbered priority it can reach; target hosts with the same priority are tried in pseudo-random order.
- *Weight* is a load-balancing mechanism used when selecting a target host among those that have equal priority—a host should be chosen first by its weight. A weight of 0 is used when no load balancing is needed.
- *Port* is the port on the target host of this service.
- *Target* is the domain name of the target host (this is the same as an MX record). There is at least one A record for this name.

A typical SRV record for domain *develop.microsoft.com* would be *ldap.tcp.develop.microsoft.com 0 IN SRV 0 0 389 ntssmith.develop.microsoft.com*.

In addition, special DNS entries are used to identify subsets of domain controllers that have special roles. For example, a typical SRV record for a domain controller that is writeable would be: *ldap.tcp.writeable.ms-dcs.develop.microsoft.com 0 IN SRV 0 0 389 ntssmith.develop.microsoft.com*.

A DC that is located in the site *redmond* would be: *ldap.tcp.redmond.sites.ms-dcs.develop.microsoft.com 0 IN SRV 0 0 389 ntssmith.develop.microsoft.com*.

Name Migration Planning

Windows NT 3.*x* and 4.0 environments are configured to use NetBIOS names for both machine and domain names. When you migrate to Windows 2000, you need to have DNS names. In many cases, the best way to accomplish this is to create NetBIOS names that are compatible with standard DNS names.

The standard DNS characters are the letters A–Z, numbers 0–9, and the dash (-). DNS names are case-insensitive. If you keep your NetBIOS names constrained to the above character set, you can use the same name in a DNS or NetBIOS context. If you have NetBIOS names that do not meet the DNS standard, you should change them if possible.

If you have extensively used Unicode and you have a Microsoft DNS server and client environment, you will have the option of using full Unicode names in DNS. Windows 2000 DNS supports Unicode Character Support based on UTF-8 encoding. The UTF-8 encoding allows complete use of non-ASCII character sets.

Dynamic DNS

Windows 2000 contains an implementation of Dynamic DNS that follows RFC 2136. Dynamic DNS allows clients and servers to register domain names and IP address mappings. This frees administrators from the time-consuming process of manually updating DNS entries. Note that you can use Windows NT without Dynamic DNS, but the network administrator's workload will increase significantly because of the work involved in manually updating DNS information.

The NETLOGON service on the Windows NT domain controller uses Dynamic DNS to maintain all the DNS entries used to access the Active Directory. Clients servers register DNS address records when they boot.

DHCP integration

The Dynamic Host Configuration Protocol (DHCP) server that shipped in Windows 2000 is able to optionally register pointer (PTR) records that map IP addresses to DNS names. For non-Dynamic DNS clients, the DHCP server can register both a host name and PTR record.

The process used to integrate DHCP and Dynamic DNS is defined in the draft RFC proposal, "Draft-IETF-DHC-DHCP-DNS-09.txt," located at http://www.ietf.cnri.reston.va.us/internet-drafts/.

Active Directory replication integration

Standard RFC 2136 Dynamic DNS implementations use a single master model for DNS updates. This means that the DNS primary server for the *zone* (a DNS zone is a partition of the DNS namespace) must be available at all times for clients to register their entries.

The Active Directory contains an ideal data store for DNS information and the Active Directory supports multimaster replication. Therefore, Dynamic DNS in Windows 2000 can optionally store DNS zone data in the Active Directory. This integration allows any domain controller in a domain to act as a primary DNS server for a zone, and the multimaster replication in the Active Directory allows each Dynamic DNS server to handle client requests.

DNS Design

The approach you take to set up the Windows 2000 DNS architecture will vary, depending on your current DNS environment. There are several key things you should understand before planning your DNS namespace:

- Every Windows 2000 domain uses a DNS domain name for its DC location information.
- The host names of the computers in a domain are not required to be related to the Directory DNS name.
- DNS zones can be integrated into the Active Directory multimaster replication.
- You should use standard RFC host names if DNS interoperability is important.
- The Windows Internet Name Service (WINS) will coexist with DNS until all computers are migrated to Windows 2000.

New DNS deployments

In a new DNS deployment, the easiest approach is to place a host in a Dynamic DNS zone that corresponds to each Windows NT domain. The Active Directory DNS integration should be enabled so that all DNS data is safely stored and replicated in the Active Directory.

For example:

Windows NT Domain:develop.microsoft.com

DNS Zone:develop.microsoft.com

Sample client:ssmith-pc.develop.microsoft.com

In this example, a new DNS zone is created to match the Windows NT domain. This DNS zone contains all of the entries needed by the domain controller in this domain. Once the DNS zone is stored in the directory, any domain controller can act as a fully functional DNS server with read/write access. Note that while not every domain controller needs to be a DNS server, each location that has a domain controller should have a DNS server.

You may find it useful to place the clients in their own domains in the zone (based on site). For the above example, some sample names could be:

ssmith-pc.nj.develop.microsoft.com, and

ssmith-pc.redmond.develop.microsoft.com

Since each subdomain is in the same DNS zone, a single DNS database is used.

Preexisting DNS deployment

If you have a large existing DNS structure supporting a very heterogeneous environment, such as a university campus, you do not have to do a major redesign to upgrade to Windows 2000 and Dynamic DNS. Complex existing environments where client domains do not correspond to Windows NT domains can operate using standard DNS zone transfer mechanisms. You can create new DNS zones in the enterprise to contain the Dynamic DNS data for the new Windows 2000 domains. By keeping these domains dynamic, all DC updates will be automated.

It is important to note that there does not need to be a relationship between the domain name of a client or server and the DNS domain name of the directory service. Examples of unrelated DNS namespaces are:

Windows NT Domaindevelop.microsoft.com

Clientsssmith-pc.nj.microsoft.com

In Windows 2000, trees have a contiguous namespace. This name space can be in one zone or in individual zones on a per domain basis. In a large, highly decentralized company, more zones are effective. In a smaller, centralized company, fewer zones work well. This is because smaller zones reduce replication traffic. Establishing zones based on the physical structure of the company generally keeps the local DNS information close to the region that requires the DNS name most often. However, a large centralized company with a corresponding large zone structure requires DNS zone transfer over a much larger area, which increases traffic. These are the same considerations you must examine when deciding on the scope of domain replication traffic.

WINS

The Windows Internet Name Service (WINS) will continue to be included in Windows 2000, but is not required in a pure Windows 2000 environment. WINS will continue to be used for downlevel clients, such as Windows for Workgroups, Windows 95, and servers that use NetBIOS. Over time, using WINS will decrease as clients and servers are migrated to a pure DNS environment.

The DNS/WINS integration feature provided in Windows NT 4.0 can be used to map the WINS

NetBIOS namespace into DNS. During transition from Windows NT 4.0 to Windows 2000, all names can appear in DNS.

Planning The Windows 2000 Directory Tree

This section discusses site and tree planning for the Active Directory. It explains how sites can be used to optimize replication traffic over wide area network (WAN) links and how tree and forest planning affects replication. The section also discusses how a geographical or organizational approach to forest planning influences network traffic, especially over slow links.

Site Planning

As explained previously, the domain structure defines the naming context and thereby determines what information is copied between domain controllers. Domain controllers in the same domain store have exactly the same data in their databases (unless changes were made very recently). Domain controllers from different domains share no data except the tree metadata (from the configuration container) that is copied to all domain controllers in the tree. Global catalog servers are special domain controllers—they receive a copy of each object in the domain tree, but only with selected properties (partial replication).

While the amount of data (full or partial replica) exchanged between Directory Service Agent (DSAs) is defined by their domain membership, the scheduling of the replication is determined by *site* membership. The site structure controls the replication process for domain controllers that are replication partners, because the site structure actually controls the scheduling of the replication process. The site object contains a scheduling attribute for intersite replication. Using this attribute, the administrator can define when replication can occur.

A site is defined as a collection of computers with a local affinity. By definition, a site is a collection of IP subnets. A domain controller belongs to exactly one site, even if the server has multiple homes. Subnets that have fast, cheap, and reliable network connections (generally those that are close to each other) should be combined in one site.

To be considered fast, a network connection should be at least 512 kilobits per second. However, the available bandwidth for directory replication has to be taken into consideration. You may have a one megabit network connection between locations, but if certain applications already consume 80 percent of the bandwidth and the remaining 20 percent must be shared between file-sharing applications and network administrative applications—such as DNS—the bandwidth cannot be classified as a high-speed connection. Microsoft Exchange specifies that a fast network connection should have an available bandwidth of 128 kilobits per second or higher.

If your network consists of only one local area network (LAN) or metropolitan area network (MAN), the site structure design is fairly straightforward. These connections are always fast, and you can define one default behavior for intrasite replication for all DSAs.

This situation is more complicated in a multilocation environment. For example, if you have an organization with headquarters in Seattle and a branch office in Los Angeles, you may have a variety of connections and connection speeds—the two locations may be connected by a slow dial-up connection, a 64 kilobit-per-second frame-relay connection, high-speed T-1 connections, or a virtual network using PPTP over the Internet. In planning for such an environment, you must first decide whether the branch office needs a domain controller at all. Branch workstations could use a WAN connection to log on to a domain controller at headquarters. This approach would work well if a very fast and reliable connection to headquarters was available. You should consider, however, that a slow logon process is annoying for users; therefore, eliminating the branch domain controller is acceptable only for very small branches with two to five workstations.

Since the amount of data replicated between DSAs in the branch office and headquarters is determined by domain membership and not by site membership, the only site consideration is the availability of data. You could choose to defer replication within a site for 10 minutes and to replicate to the branch site once. To do this, you need to have different sites for headquarters and the branch office.

The existence of mission-critical applications that use the directory as a data store can demand faster replication to all locations. In this specific case, if replication to off-site locations must happen as fast as replication on site, it may appear that you need only one site. However, you have more flexibility if you create separate sites, because sites can have more than one connection. For example, a site can use a high-speed T-1 line as its default connection and a dial-up connection as backup.

To summarize, separate locations always benefit from having separate sites. If high availability of directory data is mandatory, the replication schedule of intersite replication should be frequent. Multiple connection objects between sites provide fault tolerance.

Domain Tree Modeling

When determining your domain tree structure, you need to consider three main concepts:

- The existence of other namespaces
- The necessity of holding the network traffic for as short a time as possible
- The organizational structure

The following subsections discuss the various considerations and provide examples and different approaches for structuring a domain tree.

Network traffic

The amount of data replicated between different sites depends on whether the domain controllers are members of the same domain, whether global catalog servers are available, and whether there is a global catalog server in the target site.

For example, let's say that your organization has three locations: a headquarters in North America, a big branch office in Europe, and a small branch office in Asia. Network connections exist between North America and Europe and between Europe and Asia. The connection between North America and Europe is a fast one-megabit link and the connection from Europe to Asia is a slower 64-kilobit link.

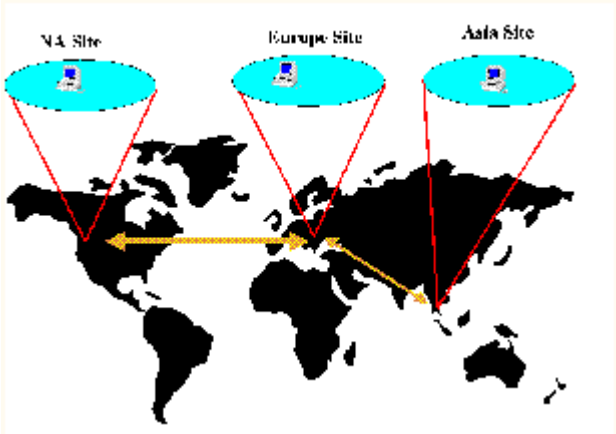


Figure 13. Sample organization with three sites

In scenario 1, the network is constructed as one big domain. All domain controllers receive a full replica of the data store. To guarantee fast logon, all locations have domain controllers.

You create a user object in the North American site, and the object is replicated to all sites. This means that the entire object is copied first from North America to Europe and then from Europe to Asia. Next, you change a property in the object, and only the changed property is transferred to the other sites.

In scenario 2, all sites create their own domains, and domain controllers are located at each site. Local administrators at the sites create users at the sites. In this case, the network traffic is actually zero with regards to replicating newly created or changed objects. The only data that is replicated is the tree metadata. (The tree metadata is replicated to all DSAs, regardless of their domain membership.)

If network traffic or cost of replication is your biggest concern and IT personnel are available in all sites, this second approach seems to be the best solution. Kerberos trust allows users to log on to all servers in the tree, regardless of the formation and depth of the tree.

Nevertheless, this solution is suitable only if users from one domain never query the directory for information stored in the domain controllers of other domains. For example, if a user from the Asian domain queries the directory for a user of the North American domain, the domain controller in the Asian domain returns a referral to a domain controller that should have more information about the object. If the client follows the referral and queries a domain controller in a different domain, the query and the result are sent over the network.

Even if the result contains only some properties of the object, the network advantage of not replicating data across sites would disappear after two or three queries for the object. Note that most enterprises have one or more client applications that must look up users from other domains; examples include LDAP-based messaging clients and human resource applications built on top of the directory.

In scenario 3, domain controllers for all domains are installed at all sites. For example, at the North American site, there are DCs from the North American domain as well as DCs from the European and Asian domains. With this configuration, client queries are always local operations, because a local DC can handle all referrals from a DC from one domain to a DC of a different domain in the same site.

Moreover, this approach results in a great flexibility of administration. Work can be done from a central location, from the branches, or from a combination of the two. Regardless of where the administrative work is done, a new created or changed object is copied only once over the network. Obviously, this configuration increases the network traffic caused by creating or changing objects, because all objects have to be copied to all sites. However, in the long run, the same amount of data is copied over the wire.

Cost for hardware is probably higher, because all sites must have three domain controllers from the different domains (Windows 2000 DCs can only host one partition). Client queries are faster than those in scenario 2, but are sometimes slower than in the one-domain solution, because even if referrals can be handled locally, it still takes some time to contact another domain controller.

Scenario 4 uses one domain per site and adds global catalog servers to all sites. Global catalog servers are special domain controllers that hold a complete replica of their own domain databases and a partial replica of all objects in the domain tree. Global catalog servers implement the same namespace structure as that used by the directory tree of the forest. However, search operations that use the global catalog usually return the results in flat lists or record sets (this is just how LDAP queries work). The global catalogs contain only those object properties that are included in partial replication. For the default schema, this is predefined. If the administrator adds objects and/or properties to the schema, these objects

are always included in partial replication, but the administrator can define on a per-object basis which properties will be replicated.

At each site, there is now a server that knows all objects in the tree. If a new user object is created in the North American site, a partial replica of the object is transferred to all sites. If clients search for specific data, they can query the entire directory using the global catalog server. Clients can use DNS to find a global catalog server, because GCs publish a host record in DNS using the well known name gc.ms-dcs.<*domain-name*>. Global catalog servers never return referrals to other DSAs, but they return the fully qualified name of an object. If a client application requires properties of an object that is not included in the global catalog's database, the client can use the fully qualified name of the object to learn the object's domain name and contact a domain controller in that domain.

The global catalog solution is particularly effective if network traffic is a big concern, but all objects need to be accessible for all clients in all domains.

Organizational structure

Many companies prefer to use the Active Directory to reflect their organizational structure. In this model, users who work in the same department are placed together in one domain. They may need common resources, such as DFS shares published in the directory; they may rely on specific portions of applications that use the Active Directory as a data store; and they may use workgroup applications such as scheduling or document routing. To simplify the locating of colleagues and to make use of common security, it makes sense to group these employees and resources in domains.

Figure 14 shows a possible organizational structure.

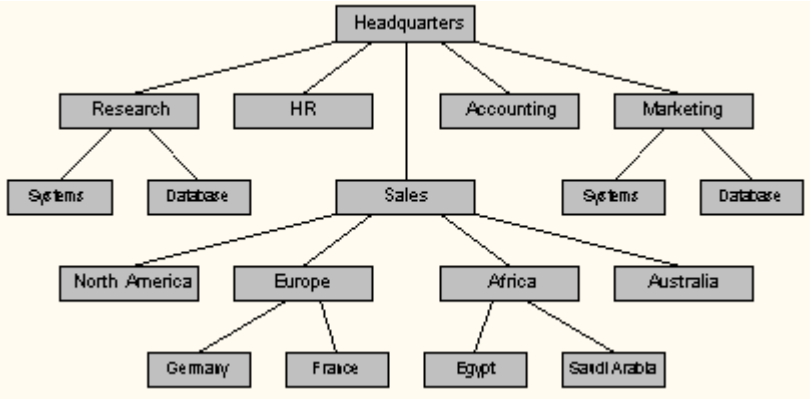


Figure 14. Domain structure that reflects the business organization

The enterprise is a very centralized organization. At the top of the organizational chart, is the Headquarters department. On the next lower level are the Research, HR, Sales, Accounting, and Marketing departments. Some of the departments have several groups. All departments and groups are located in one city in North America except for the Sales department. The Sales department has some personnel in North America, but most of the employees work in different cities in different continents. France serves as the European headquarters, Egypt as the African.

The site structure is easy to develop. There is one big site in North America and smaller sites in Germany, France, Egypt, Saudi Arabia, and Australia. The bigger sites—North America, France, and Egypt—need global catalog servers. No global catalog servers are planned for the smaller sites (Germany, Saudi Arabia, and Australia), but if the networks traffic caused by client queries becomes problematic, global catalog servers can be added there as well.

The domain tree requires more thought. For Headquarters, Research, HR, Accounting, and

Marketing, network traffic that results from replication is not a major concern, because these departments are all located in the same site. The first approach could be to combine all of these departments in one domain. Sales would be placed in a separate domain, because the sales force is widely dispersed. This simplifies administration, because you have just one North American domain where most objects are created. (The Sales domain is discussed separately, below.) Replication requires no special consideration, because the entire database is copied to all domain controllers. The database, however, must be able to hold all objects. Because the Active Directory can store more than 10-million objects in the data store, this shouldn't be a problem.

To achieve a clearer representation of the directory's organization, the network administrator could decide to divide the big domain into several organizational units. The domain name would be HQ. OUs on the next lower level would be Research, HR, Accounting, Marketing, and so forth, This results in a finer granularity of administration and control. Because the Active Directory allows delegating appropriate rights to special users, you could create OU-specific administrators.

The disadvantages of the one-domain model include the amount of data stored on the domain controllers and the fairly open security model. Although the database of the domain controllers can handle the large amount of data, the size of the database affects logon and query operations. Because all data is stored on all domain controllers, data exists *n* times (where *n* is the number of domain controllers). If a location (in this case, the North American Headquarters) requires 10 or more domain controllers, each object must be stored 10 times. This is inefficient, and it not only slows down client operations, but also makes backups of the database more complicated.

Although the network connections between replication partners are very fast, every new user has to be copied nine times over the network, because every domain controller has at least one replication partner. The one-domain model may be suitable until the organization reaches a certain size, but beyond that threshold this approach causes too many network and storage problems.

Usually more thought is required to develop a workable security model, and security considerations can greatly influence domain structure. By default, every security principal that is created in a domain has the same domain rights. Although organizational units can contain security principals, a principal's membership in an OU does not affect the principal's rights in the domain. For example, many resources are available for the built-in group "Domain Users." This ensures that newly created users can automatically access default resources in a domain, such as public group shares or printers.

However, the administrator only rarely intends to give all users in the organization the same rights to all resources. A user in the Marketing department, for example, should not have rights to resources in the HR department by default. To resolve this situation without changing the domain structure requires complicated administration. The administrator must set up groups for every department, add new users to these groups, and avoid using the built-in groups (or use the Active Directory Service Interfaces to create Visual Basic® programming system scripts or applications to create new users and add them to the appropriate groups automatically). This approach is complicated, error-prone, and fails to use the basic security boundaries inherent in the Active Directory domain structure. The use of default rights on a domain level is a much more powerful alternative.

If you omit the Sales domain (which spans several sites), the structure shown in Figure 15 could be a workable solution for a domain tree.

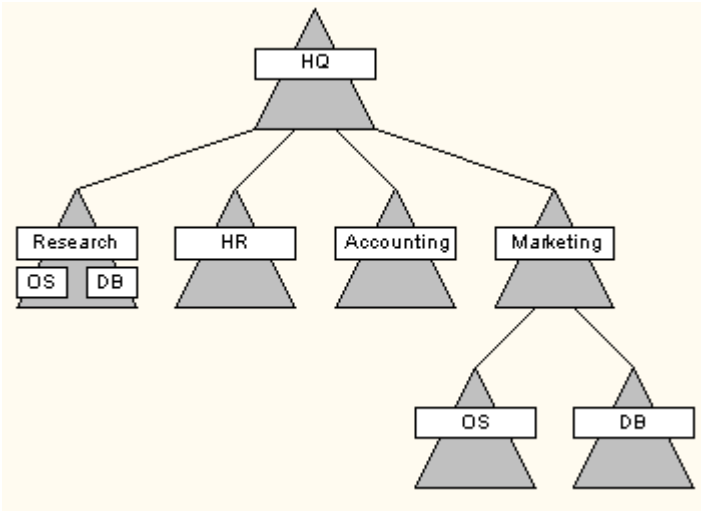


Figure 15. Domain structure that reflects the business organization and imposes security boundaries

HQ is the top-level domain. Research, HR, Accounting, and Marketing are children. For the OS and DB groups that exist in both the Research and the Marketing department, two different solutions were chosen. In the Research domain, OS and DB are implemented as OUs in the domain. In the Marketing department, OS and DB are independent domains. This is an arbitrary solution, which assumes that the Marketing Department is larger and more dispersed than the Research department. Security policy can flow down the tree from parent to child domains. It is possible to configure the tree so that the security policy that is set in the HQ domain applies to all child domains.

The Sales department can be incorporated easily into this approach. Sales would be in a new child domain under HQ. Child domains for the Sales domain would be created for North America, Europe, Africa, and Australia. Because France and Egypt host the headquarters operations, it would be appropriate to implement these groups as organizational units in the European and African domains. Germany and Saudi Arabia can be implemented as domains or OUs, depending on the number of employees (and their security needs) and the available network infrastructure. In every case, global catalog servers should be available at all sites.

Although the site topology is usually simple to define, the tree and forest structure is influenced by whether you choose a geographical or an organizational approach for your network. If you use the organizational approach, the tree and forest plan is also driven by the organizational structure. It is only important to sort out when you should create domains as opposed to organizational units—and this choice is largely driven by security considerations.

If you choose the geographical approach, you may want to implement a root of a tree that is owned by your IT department—this allows the central IT group to administer and enforce security policies for the entire domain tree. On the next level of tree, you can place continents. In the continent domains, you can create domains for larger countries and OUs for smaller countries. Alternatively, you can implement a forest that consists of independent trees for each continent. This gives the continents more freedom of administration, because there is no root that belongs to a central IT group; therefore no central group can enforce security policies for the entire domain tree.

Migrating from Windows NT 4.0 Domain Models

Windows NT 4.0 supports many domain topologies that can be constructed from two basic types of domains (*master* and *resource* domains) and several relationships that can exist between the domain types. The difference between the domain types centers on the types of objects that each type holds. Master domains contain definitions of user accounts and some

machine accounts, while resource domains contain definitions of resources that are to be made available to users in one more master domain.

Windows 2000 allows a more flexible domain structure. Therefore, it is very important for organizations to decide what type of structure they want to implement before they begin the migration process. This section describes some of the issues you may want to consider when choosing your domain model and migrating from Windows NT 4.0 to the Active Directory. It then describes how to migrate the various Windows NT 4.0 domain models to the Active Directory.

Choosing Your Domain Structure

Before migrating to Windows 2000, examine your organization's business structure and operations. Many companies have a centralized structure. Typically, these companies have strong IT departments that define the network structure and implement that structure down to the smallest detail. Other organizations, especially large enterprises, are very decentralized. These companies have multiple businesses, each of which is very focused, and they need decentralized approaches to managing their business relationships and their networks. In such enterprise operations, there may be no single place in the corporation that can control or even begin to manage all resources.

Centralized or distributed?

With Windows NT 4.0, network administrators tended to centralize the structure even if this approach did not reflect that of the organization. A centralized structure simplified administration and was easier to design. With the Windows 2000 Active Directory, however, flexibility of organization is a primary goal. Both centralized and decentralized structures are fully supported. Customers can choose the correct model for their organization without being pushed to a less appropriate model just because it is more convenient to implement and administer.

Since the Active Directory uses organizational units to implement a hierarchical namespace within a domain, one consideration that has become important is the number of domains that a company wants to implement in the directory tree or forest. Many customers today have more domains than they want to manage. Windows 2000 organizational units have features—such as delegation of administration rights—that allow these customers to reduce the number of domains without sacrificing administrative flexibility. This is especially important for centralized companies.

Decentralized companies may prefer to transform their Windows NT 4.0 resource domains into full-featured domains and move user accounts from the former master domains to the domains where they really belong. The transitive Kerberos trust relationship now allows user accounts to be valid within the whole forest, eliminating the need for complicated trust management. This makes it possible to create a very distributed and scalable management architecture.

To summarize, if you have a centralized organizational structure, you may find that fewer domains and a greater use of OUs is desirable. Decentralized organizations will most likely choose to have more domains in the forest and allow local control of the domains.

Network traffic and international issues

In a widely distributed network, you should also consider replication traffic and international issues. If the underlying network includes some very slow WAN links that are already saturated, you may choose not to replicate the entire directory over the WAN links, but to replicate only the subset needed for the global catalog servers. This results in a geographical approach in the domain planning process, with domains in each country or location.

In many cases, it may be easier for international companies to administer users and resources in their national language. This again results in a geographical approach with separate domains

for regions where different languages are spoken.

The following sections discuss the Windows NT 4.0 domain models and suggest planning solutions for their migration to the Active Directory.

Single Domain Model

This is the simplest domain architecture possible in a Windows NT 4.0 topology. In this architecture, you simply have one primary domain controller that is holding the master copy of the Security Account Manager (SAM) database. In addition, there may be one or more backup domain controllers. In this architecture, all user accounts, machine accounts, and resource definitions (such as printer queues and shares) use security principal definitions from the SAM database on the primary domain controller. These accounts are granted rights based on Access Control Entries (ACEs) in Access Control Lists (ACLs) on whatever resource is being shared or restricted. By definition, there can be only one copy of the SAM database that is modified at any given time, and the database is owned by the PDC.

When migrated, a single domain model usually results in a single domain in the Active Directory. However, you can use many of the new features of the Active Directory—such as an OU hierarchy and delegation of administrative rights—to better arrange and administer this single domain. By organizing users and resources into a hierarchical OU namespace, you have a much clearer representation of the actual business structure reflected in the domain. And with the new delegation features, users or user groups can perform very granular administrative work—such as resetting passwords or maintaining special rights in special containers.

Master Domain Model

A master domain model's migration typically happens in a top-down manner, where the master domain is the first domain to be migrated and the resource domains are migrated later. Depending on whether the company is centralized or decentralized, you may find that your entire organization now easily fits into one domain. In a centralized company, you can use organizational structure to mirror the old master domain structure.

Resources from second-tier domains can be moved slowly from the former resource domains to the new domain. This approach is effective if an organization wants to migrate from several domains to a single domain and wants to dissolve the resource domains into OUs in the master domain. This approach results in both more centralized control for the administrators and finer granularity of administrative delegation.

Another approach is to use the flexibility of domain trees by retaining the resource domains and moving the user accounts to the domains where they really belong. This results in a more decentralized topology and allows people and groups to work independently.

Many companies buy and sell business units and want to treat them as separate companies. These business units may have complete independence in their budget and hiring decisions, but there should be an overall framework for running the network. For these companies, a domain tree with multiple domains is a good solution. Migrations that account for this business model are described next.

Multiple Master Domain Model

A multimaster domain model may be necessary for several reasons. For example, the business may need to mimic its actual business structure in the domain model. In such a business, different business units manage different account domains and different resource domains. Another reason to use this model is that an organization may need to split up the network, because of domain replication issues in different parts of the world.

For this type of organization, two migration approaches are appropriate. The first approach is to build a single domain tree. In this case, the business units have to agree on one root domain and who will own that root domain (preferably the IT department). The individual business units can be implemented as different domains on the next lower level of the tree, and they can again implement their resource domains on a deeper level using their own security policy.

The root domain is a good place to offer global resources that need to be available for everyone, such as corporate Web servers and HR data. This solution allows using a central location to manage the root of the tree and to administer central resources, but also allows the business units to retain their independence by implementing their own structure and security policy in their sub trees.

An even more decentralized approach is to build one domain tree for each master domain and its resource domains and to join these subtrees into one forest. The forest concept allows these subtrees to retain a common schema, a common global catalog, and the transitive Kerberos trust relationship so that users can still access resources from all over the domain tree. The costly management of the trust relationships disappears, because each domain requires only one trust to their parent domain.

Using the forest approach, business units can implement their own subtrees using their own structure and security policy. The only thing they have in common is the connection to the other trees on the forest level. No administrative work is necessary for this. All administration happens on the tree level. This very flexible model is particularly appropriate for companies that buy and sell business units and want to treat them independently.

Complete Trust Model

Some organizations are so decentralized that they cannot agree on an owner (such as a central IT department) that can manage common resources or establish a common root in a tree. These companies used the complete trust model in the past, and they had to assume the corresponding burden of maintaining all of the trust relationships between the domains.

With Windows 2000 and the Active Directory, it is still possible to retain this independence. The organization can again choose to either introduce some centralization and build one tree or to fully retain the decentralized structure and to build a forest that consists of several trees (which are the former domains). In both cases, this results in a very flat structure. However, the management of the trust relationships will be much easier and more scalable in Windows 2000.

Another solution for this model is to migrate to separate forests. Windows NT 4.0-style trust relationships can be established between selected domains to allow users from other domains some access to some domains in another forest. However, this model would seem to work best to allow communication between businesses, rather than being the structure for a single corporation—no matter how large and dispersed that corporation is.

To summarize, all domain models can be easily migrated to the Active Directory. Depending on the organizational structure, the flexibility of the Active Directory allows you to reflect a centralized or a decentralized business structure even more than you could with Windows NT 4.0. On the other hand, a migration can be an opportunity to change the structure of your domain topology to better meet your current business needs.

Migrating from Other Namespaces

Microsoft Systems Management Server

Microsoft Systems Management Server (SMS) Site definition is very different from the site

definition in Windows 2000. Sites, as defined by SMS, are fundamentally not bounded by bandwidth—although this may be a limiting factor for the overall architecture. In SMS, the site concept is generally used in the context of creating a container for administerable *objects*. In the SMS context, these objects are either of the *System* or *Package* type. Additionally, there are groups of systems that can have packages installed, based on some defining characteristic of the group or the machine. In Windows 2000, however, a site is defined as a subnet or area of high and persistent bandwidth.

In future revisions of all Microsoft BackOffice applications, a site concept centered around the Windows 2000 will be implemented. This will simplify the implementation architectures for all of these applications and will eliminate some confusion.

Microsoft Exchange Server

> **Note** All information that references Microsoft Exchange Server migration and coexistence with Windows 2000 is highly volatile and subject to change.

If you are migrating a Microsoft Exchange Server implementation to Windows 2000, you need to examine several points before you attempt the upgrade. Generally, this type of migration will occur from one of two types of topologies:

- An implementation based on Exchange Server versions 4.0 to 5.5.
- An implementation based on the release of Exchange Server that will integrate the Active Directory (this release is code-named "Platinum").

In either topology, the Exchange implementation typically creates several namespaces that can be duplicated in a Windows 2000 implementation. For example, on installation, Exchange extends the Windows NT 4.0 directory with additional properties that are accessible via an LDAP client. It also creates an SMTP (RFC 822) namespace for most recipients. Both of these namespaces are also represented in Windows 2000.

To resolve the issues with duplicated and nonsynchronized namespaces present in Exchange Server and Windows 2000, the "Platinum" release includes a replication agent that will keep the Exchange and Windows 2000 directory information synchronized in the interim period before all Exchange Servers in the organization are upgraded to "Platinum." Note that a server must have Windows 2000 installed before the upgrade process to "Platinum" can begin.

Exchange 4.0-5.5 coexistence with Windows 2000

Exchange Server versions 4.0 through 5.5 use the Exchange directory data store to store directory information for the organization's site configurations and user definitions. Exchange architectures rely on the existence of a viable Windows NT 4.0 domain topology to provide messaging services to potential Exchange users inside an organization. However, many of the attributes that are necessary to establish the messaging and organizational/workflow services that Exchange is designed to provide cannot be stored in the Windows NT 4.0 directory. Therefore, an Exchange implementation adds the capacity to map arbitrary attributes to any security principal defined in a Windows NT 4.0 domain.

When an organization decides to install a Windows 2000 server, they need to move attributes that they have defined for Exchange 4.0 through 5.5 users to the Windows 2000 directory. This can be accomplished with the ADSI scripting capabilities of Exchange 5.5 or with a replication agent that performs the one-time replication that is necessary to make sure that the Windows 2000 directory is appropriately populated. Microsoft will provide a replication agent that will be installed automatically after the first server is installed. This agent may be made available before the Exchange "Platinum" version is available.

Even if the properties are migrated, the Exchange implementation will not use the Windows 2000 directory until the "Platinum" timeframe. Therefore an organization will need to keep the

Exchange directory populated until they make the upgrade decision to move to "Platinum." Windows NT 4.0 backward compatibility will guarantee that Exchange, as well as other BackOffice applications, will not cease to function correctly.

"Platinum" to Windows 2000 upgrade

Upgrading your directory from "Platinum" to Windows 2000 means that the Exchange configuration information that is currently housed in the Exchange directory, as well as the definitions of users and their attributes, needs to be migrated. To accomplish this, you must first create an Active Directory Auxiliary class and add it to any defined Windows 2000 user or third-party application or service. Quotas, maximum transferable message size, and so forth, are contained as attributes in this auxiliary class.

Configuration information for Exchange sites and servers is stored in the Active Directory once the migration is complete and the following containers are added to the Windows 2000 Directory for Exchange information:

- "\Configuration\Sites\<Site>\Exchange" (for site-based information)
- "\Configuration\Services\Exchange" (for global Exchange configuration information and settings)

In the "Platinum" release, Exchange supports the transition from the existing Exchange directory to the Windows 2000 Active Directory. To provide for seamless interoperation of the two products for the amount of time necessary to provide a smooth migration, there will need to be some mechanism that keeps the two directories synchronized. "Platinum" will require that you create a Windows 2000 directory tree before you install the first "Platinum" server. When the first "Platinum" server is installed, the setup program will create a replication agent that will provide full replication between the two infrastructures.

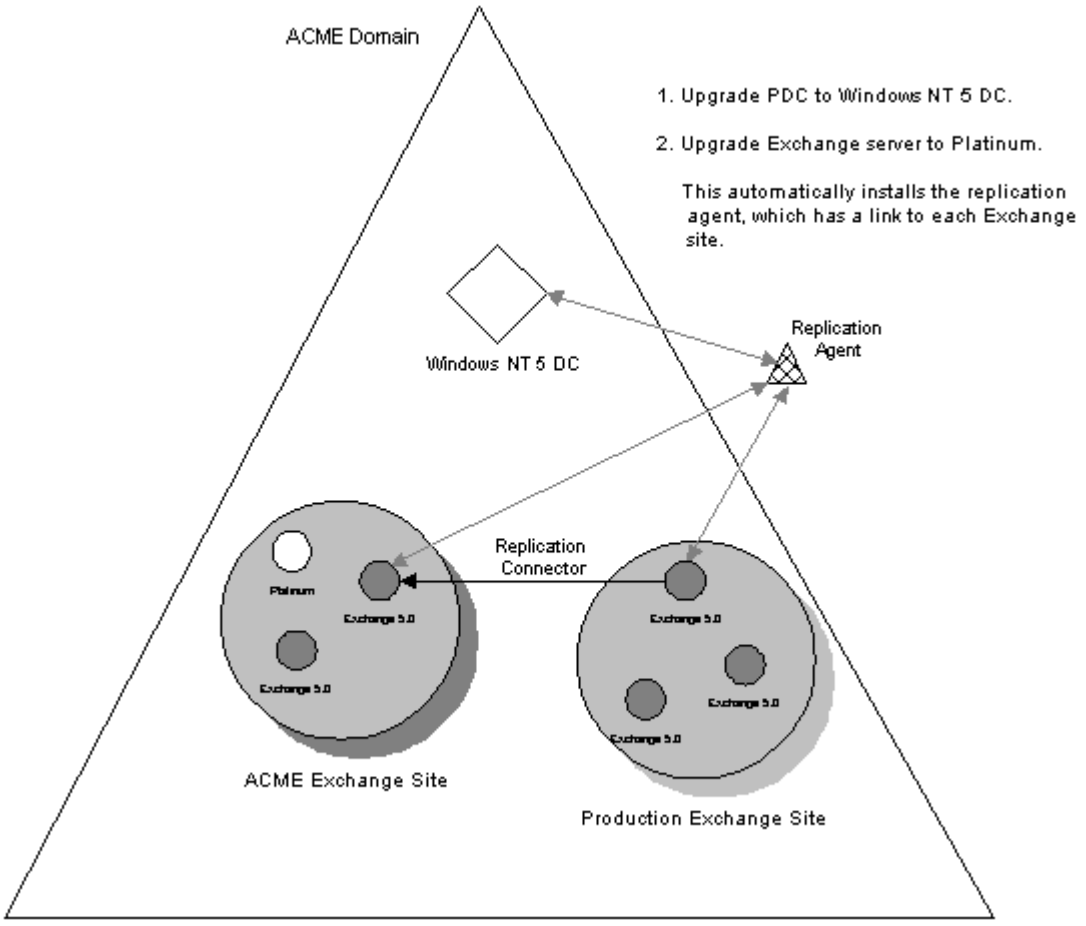


Figure 16. Multisite Windows 2000 and Exchange upgrade of PDC and initial Exchange Server

The initial replication will include the migration of the existing properties and objects from the Exchange directory to the Windows 2000 Active Directory. The result will be a completely populated Windows 2000 copy of the existing Exchange directory. The replication agent will continue to function until the Exchange infrastructure has migrated completely to "Platinum," at which time the Exchange infrastructure will use only the Windows 2000 Active Directory. Messaging namespaces will be saved as properties for individual users in the Active Directory.

The next step in the migration is to upgrade all other nonbridgehead servers in each of the Exchange sites of the organization.

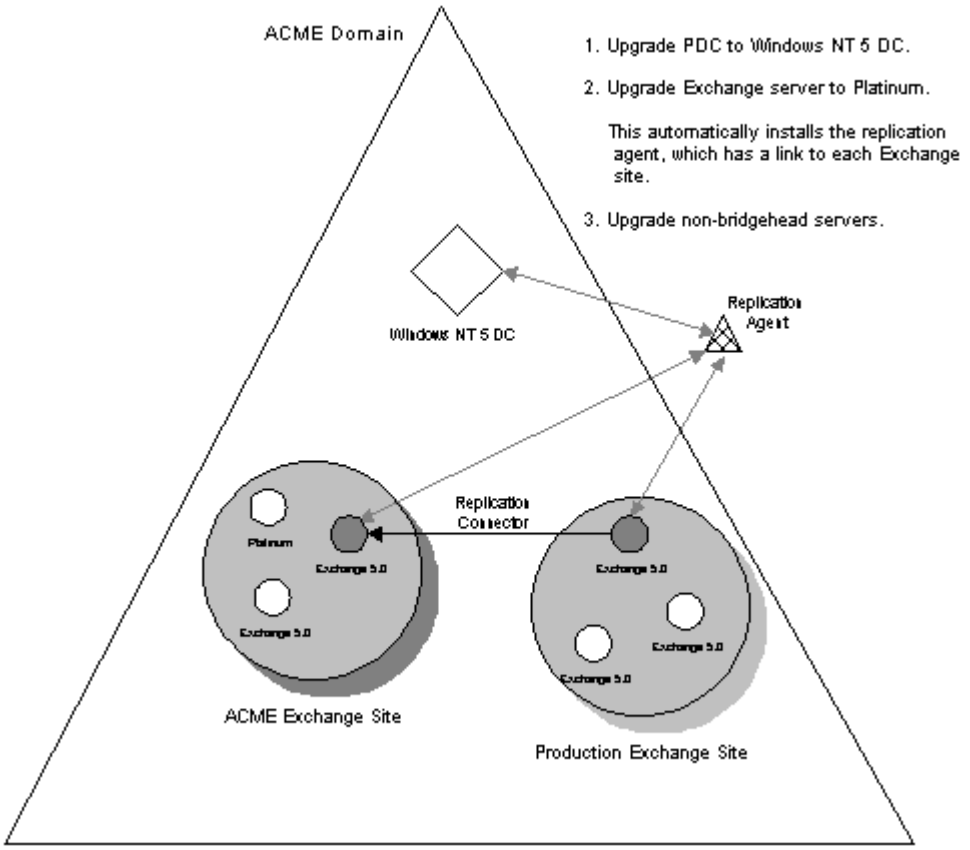
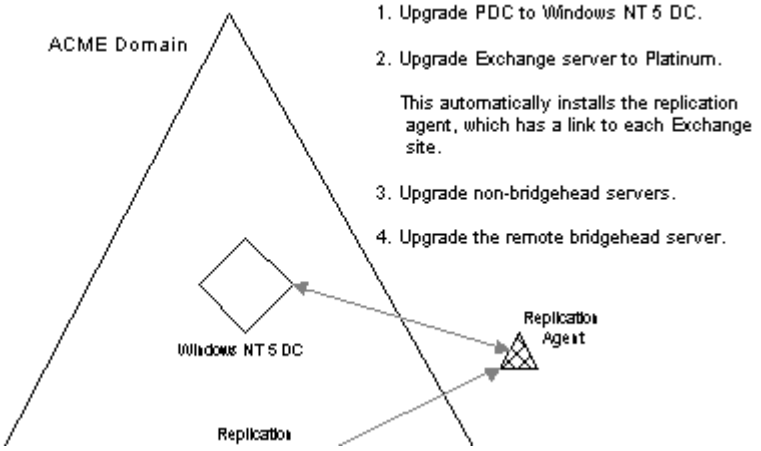


Figure 17. Multisite Windows 2000 and Exchange upgrade of nonbridgehead servers

After this step is complete, the remote bridgehead server is upgraded to "Platinum."



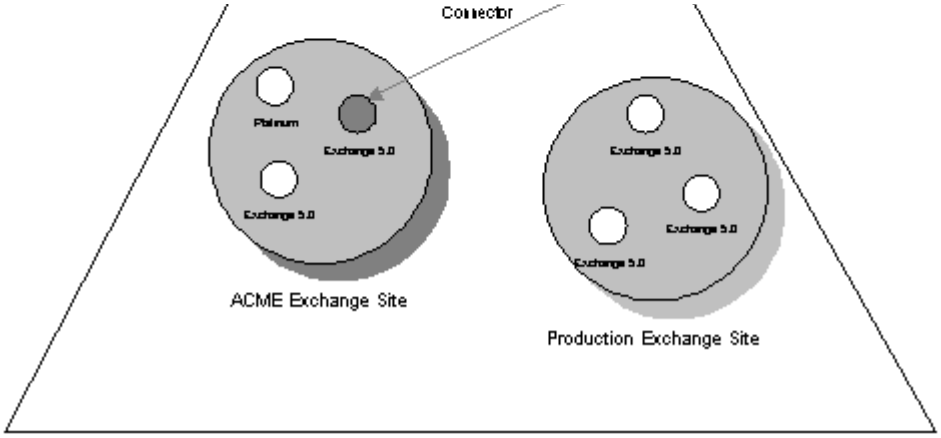


Figure 18. Multisite Windows 2000 and Exchange upgrade of remote bridgehead server

After the final server is upgraded, the replication agent is automatically uninstalled. At this point, your entire organization is using the Windows 2000 Active Directory.

Dissolving Second-Tier Domains into the Master Domain

Today, many companies use the master domain model to administer users and groups in first-tier (or master) domains and resources in second-tier (or resource) domains. Because the majority of reasons for entertaining this model go away when domains are migrated to the Active Directory and a domain tree is established, these companies may choose to dissolve existing second-tier domains into organizational units in the former first-tier domains.

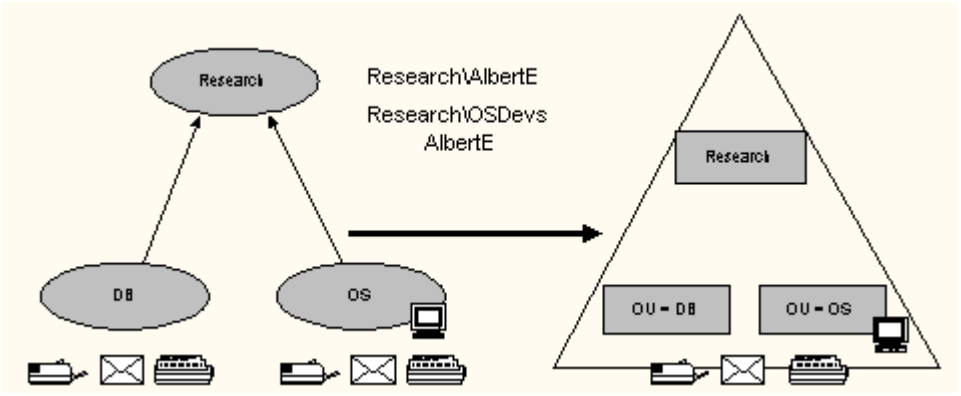


Figure 19. Dissolving second-tier domains

While this is not an automatic process that can be performed through the administration user interface, it is still a relatively easy task. The only issue that requires some planning occurs if your organization uses local domain groups from the second-tier domains to organize access rights for the resources in these domains. This section provides guidelines for performing this migration and dissolving the second-tier domains without losing access right information.

Identifying How Access Rights Are Organized

In a Windows NT 4.0 master domain model, there are three methods for using groups to organize access rights on resources in second-tier domains. The first method is to use global groups in the master domain; the second is to use local domain groups in the second-tier domains; and the third is to use local groups that reside on member servers in the second-tier domains.

Figure 20 shows a master domain called "Research" and two resource domains called "DB" and

"OS." All user accounts reside in the Research domain. The one-way trust relationships from the resource domains to the master domain allow access to resources in the second-tier domains for users and global groups from the master domain.

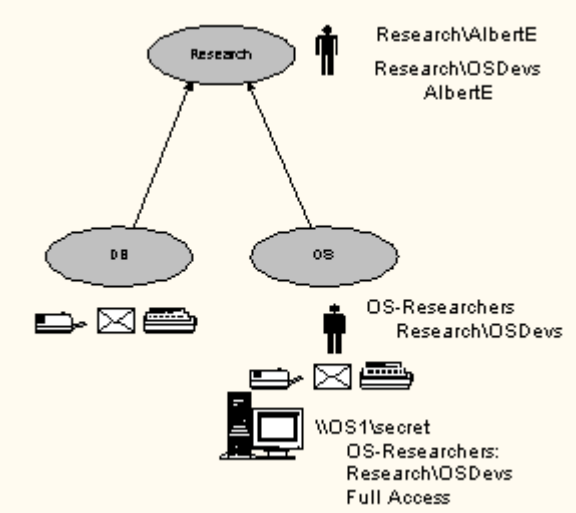


Figure 20. Organizing access rights in second-tier domains

The user *AlbertE* has an account in the Research domain. He is working with a Windows NT Workstation version 4.0 that belongs to the OS domain to which he logged on using his user account from the master domain.

The server OS1, which belongs to the OS domain, shares a directory called "secret." When the system administrator grants a user access to this resource, the administrator creates a new Access Control Entry (ACE) in the Access Control List for the resource.

There are three ways for AlbertE to obtain access to the "secret" share on the OS1 server (these methods mirror the organization of access rights for the server):

- Using the trust relationship, he can use the user account AlbertE from the master domain Research.
- Using the trust relationship, he can use the global group Research\OSDevs from the master domain Research, where user account AlbertE is also a member.
- He can use a local group. There are two different kinds of local groups—local domain groups that can be used by domain controllers and local groups on member servers that can only be used by this particular server. Either group can contain user accounts and global groups from the own domain and from trusted domains. Depending on the nature of the server OS1, either a local domain group for a domain controller (such as Research\OSDevs) or a local group on a member server (such as OS1\OSDevs) can be used.

Security identifiers and second-tier domains

When an ACE is added to an ACL, the security principal that is the subject of the ACE is represented by a Security Identification Descriptor (SID). The SID is a unique identifier. Among other information, the SID always contains an identification of the domain or machine where the SID was created.

If a user wants to use a resource, the user must be authenticated and the user's access rights must be evaluated. Depending on the credentials offered to the authentication package, the server either uses local account information stored in the Security Account database or passes the information to the NETLOGON service to perform a domain logon. The

domain controller uses the credentials to determine in which domain the user account resides. If the user has an account in the same domain, the domain controller performs the logon and copies the user's SID and the SIDs of all global groups to which the user belongs, to the server. If the user account does not reside in the server's domain, but instead resides in a trusted domain, the domain controller contacts a domain controller in the trusted domain.

This process is called *pass-through authentication*. If either the server or a domain controller can successfully evaluate the user, the LSA on the server builds an access token for the user. Among other information, the access token contains the user's SID and the SIDs of global groups to which the user belongs. The server then uses the token to try to access the resource on behalf of the client, and the operating system either grants or denies the access request.

In our example, the access token created by the server OS1 contains the SID of the user *Research\AlbertE* and the SID of the global domain group *Research\OSDevs* that reside in the master domain. Depending on whether OS1 is a domain controller or a member server, OS1 adds either the local group *OSResearchers* (which is a local domain group if OS1 is a domain controller) or the local group *OS1\OSResearchers* (which is a local group on a member server if OS1 is a member server).

When a second-tier domain is dissolved into an organizational unit in the former master domain, the local domain's SIDs for user and group accounts are no longer valid. This can cause problems if servers that functioned as domain controllers in the second-tier domains are moved to the former master domain, and then use local domain groups to organize access rights. However, if the server that is moved to the master domain was a member server in the second-tier domain, and local groups on the member server were used for access rights, these local groups move together with the server, and the SIDs remain valid.

To solve the disappearing SID problem that occurs when local domain groups are used, the Active Directory allows you to copy security principals from one domain to another domain. After the copy operation, the security principal has two SIDs: one SID from the old domain and one SID from the new domain. When the logon service on a server machine builds an access token for a user, both SIDs representing a user or group account are added to the token. This ensures that the link between SIDs used in an ACE and the corresponding security principal will never be lost.

In summary, the Active Directory provides the tools that are necessary to dissolve a second-tier domain into an organizational unit without losing access right information. However, it is important that you perform the migration steps in the correct order. The following paragraphs provide a step by step guideline for performing the migration and dissolving resource domains in a safe manner.

Step 1: Migrate the Master Domain

In step 1, you migrate the master domain to Windows 2000 and install the Active Directory on a minimum of the PDC. As mentioned earlier, the master domain can join an existing Active Directory tree at this stage.

Although the master domain now establishes a two-way transitive Kerberos trust with the parent domain in the tree, the trust relationships to the resource domains remain untouched. Even if all domain controllers are running Windows 2000 and the Active Directory, trust relationships to downlevel domains are still entertained as Windows NT 4.0-style trusts.

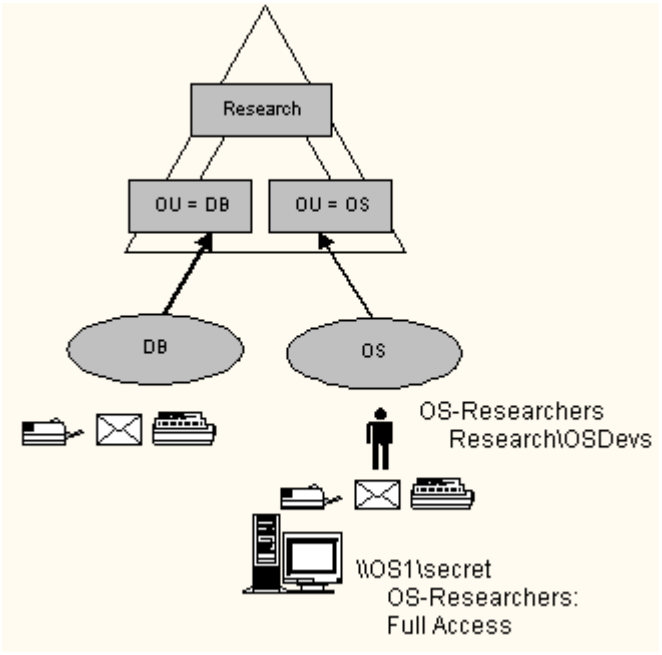


Figure 21. Step 1, migrating the Master Domain

Step 2: Create Organizational Units

Because the goal is to dissolve the second-tier domains into organizational units in the master domain, you must create the organization before the resources can be moved. You can now start to use the OUs to create a hierarchical namespace. You can create users and groups in OUs or you can move them from the cn=users container to OUs.

The new organizational units can only be accessed from client or server machines that already have the Active Directory client access software installed. Downlevel clients aren't aware of the existence of the organizational units, and resource domains do not notice any changes in the master domain. If a downlevel client or server uses the standard Windows NT LAN Manager APIs to query a domain controller for security principals, the domain controllers automatically expand the security principals from all containers in flat list that looks exactly like the old Windows NT 4.0-style domain user database.

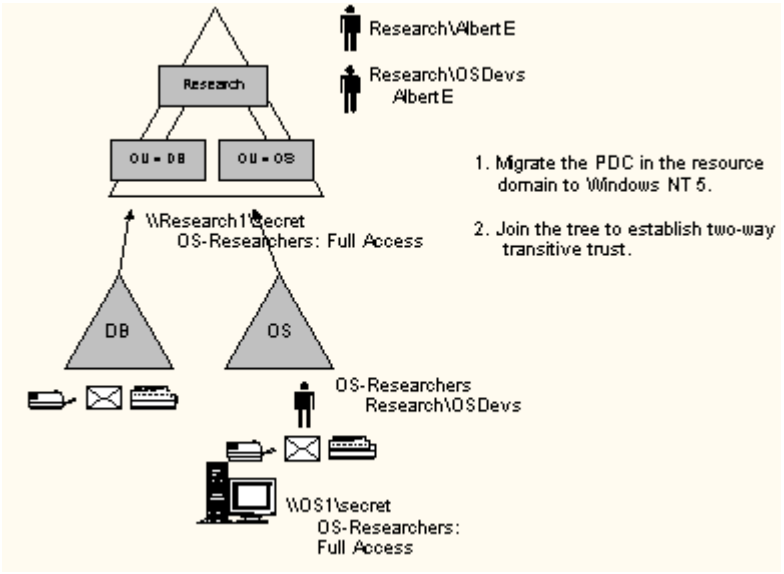


Figure 22. Step 2, Creating the OU hierarchy in the master domain

Step 3: Migrate PDCs in the Resource Domains

In step 3, you need to upgrade the PDCs in the second-tier domains to Windows 2000 and the Active Directory.

Note that this step is mandatory if you intend to move either servers or security principals from the second-tier domains to the master domain later. Only the Active Directory supports this operation. Windows NT 4.0 did not allow the moving of objects from one domain to another, because the SID includes information about the issuing domain, and applications used this information to identify domain controllers that can resolve SIDs. The Active Directory provides a SID tracking mechanism that allows the correct domain controller to be found, even if the SID has changed.

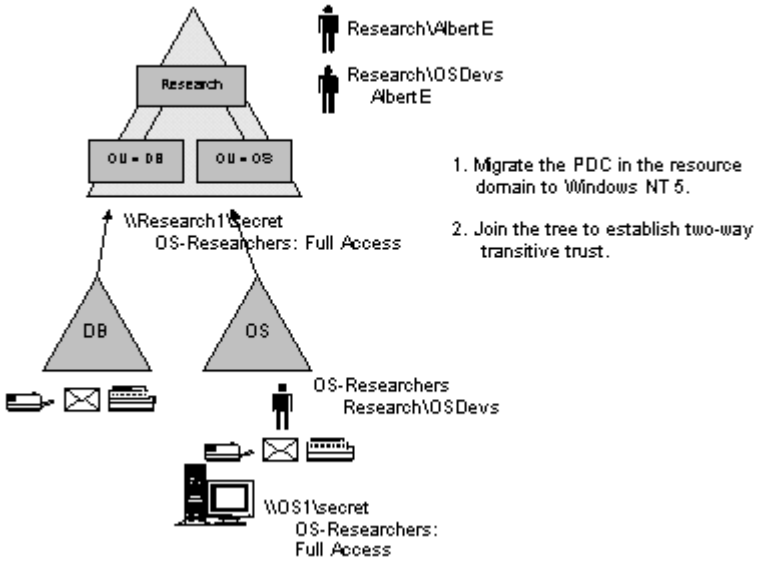


Figure 23. Step 3. Upgrading PDCs to Windows 2000

Step 4: Move Servers to the Master Domain

At this point, all tools are in place to move resources from the resource domains to organizational units in the master domain. The administration user interface provides a drag-and-drop tool that makes it very easy to perform this step.

Client applications that use UNC names to connect to servers will continue to function correctly. UNC names are not aware of the domain membership of a server. Clients use NetBIOS browsing mechanisms to find the server. These mechanisms have no domain dependencies.

As administrator of the master domain, you can now use both NetBIOS and Active Directory publishing to advertise the existence of a resource. This allows clients that work with the new access software for the Active Directory to use the extended query mechanism of the directory to find resources and to connect to them. Note that as long as downlevel clients exist in the network, the NetBIOS advertisement should be used in addition to publishing the resources in the Active Directory. As soon as all clients have the new directory access software, NetBIOS can be turned off.

Removing the PDCs from second-tier domains is premature at this point. The domain still exists and is responsible for the SIDs it issued. If, for example, a former BDC from the resource domain was moved to the master domain and a local domain group was used for access permissions, then the former BDC must contact a domain controller (the PDC) in the resource

domain to perform the user's logon operation. The use of two-way transitive Kerberos trust between Active Directory domains allows pass-through authentication from the master domain to the resource domain.

Step 5: Check Access Control Lists

During step 5, you need the information about the organization of access rights on shared resources. As described above, either global groups from the master domain, local groups on member servers, or local domain groups can be used to grant access in Windows NT 4.0 domains.

- If global groups from the master domain were used, these global groups are still valid. You do not need to make changes to the ACLs.
- If local groups on member servers were used, the member servers take the groups with them to the new domain. Again, you do not need to make changes to the ACLs.
- If local domain groups in the resource domains were used, the domain controllers that are responsible for these groups will eventually disappear. You need to move the local domain groups to the master domain. Afterwards, these groups will appear as normal domain groups. You do not need to make any additional changes to the ACLs.

Step 6: Move Workstations to the Master Domain

Besides the server machines, you need to move the workstations from the resource domains to the master domain. For Windows 95 and Windows 98-based workstations, this is not costly. It is sufficient to change the workgroup name in the network configuration section. Windows NT Workstation products must be removed from the old domain and added to the new domain. The master domain will create new machine accounts.

You can complete this step for all workstations when you install the software upgrade required for the Active Directory. Windows NT 4.0 workstations must be upgraded to Windows 2000. If you use an automated setup script for the update, the domain change can be performed automatically. For Windows 95 or Windows 98-based machines, you must install a service pack. If you use a software distribution tool such as SMS to install the service pack, you can use the same script to move the workstations.

Step 7: Turn off PDCs in Second-Tier Domains

When you reach this step, you have completed all prerequisites for eliminating the second-tier domains. You can either turn off all domain controllers in the second-tier domains or move the domain controllers to the master domain. Either way, the resource domains disappear, and all resources are available in the organizational units in the remaining domain.

Conclusion

The Active Directory introduces many new concepts, such as scalable trees that use transitive Kerberos trusts, LDAP as the standard protocol, DNS as the locator service, and organizational units to create hierarchical namespaces within a domain. This makes the Active Directory a compelling product to create scalable directories in large enterprises and to connect intranets with Internet.

You do not need to go to any great lengths to prepare a Windows NT network for the migration. The best preparation is to have a good Windows NT 4.0 domain model and a DNS structure that serves the needs of the network. The DNS namespace can be used as a starting point in planning your Active Directory domain structure.

The flexibility of the Windows NT domain structure allows you to use the migration process as a means of reorganizing a network or to simply migrate your existing structure to the Active Directory. Whatever migration method you choose, the process can be accomplished gradually. Mixed domains consisting of both Windows NT 4.0 and Active Directory domain controllers are fully supported. One domain controller can be updated after another—there is never a point that requires you to take a domain offline. Both Windows NT 4.0 and Active Directory domain controllers can serve Active Directory-aware clients and downlevel clients. Even after all domain controllers are updated, downlevel clients can still be authenticated and can still use resources in the network.

The Active Directory distinguishes between *trees and forests*, which typically are used to manage administration and security in an organization, and *sites*, which generally reflect geographical boundaries and help you to optimize replication over slow connections. In most cases, enterprises can choose to arrange their trees and forests using a geographical or an organizational approach. In a geographical model, the IT department could own the root, continents could form the first level of the tree, and countries could own either the next level or could be placed in organizational units within the continent domains.

If you select this model, you can best prepare for the transition by building a corresponding DNS structure first. This model also helps to minimize the replication traffic over WAN links. No matter which model you choose, all existing Windows NT-based structures can be migrated easily to the new forest. The improved flexibility of security principals and domain controllers allows you to move users, user groups, domain controllers, servers, and all other objects easily from domain to domain. Active Directory domains can easily be renamed, and the trees and forests can be easily restructured by moving domains to new locations. This allows greater flexibility during migration and allows you to restructure your model as business needs change.

Migration to the Active Directory is a process that can be implemented gradually, allowing you to reorganize and incorporate needed changes and new features into your directory structure. Or migration can be performed quickly using your existing structure. All workstations and servers remain functional during the migration and you can improve your tree and forest structure at a later date if necessary. Migrating to the Active Directory will move networks from internal solutions to an Internet-ready structure for the enterprise.

For More Information

For the latest information on Windows NT Server, check out our World Wide Web site at www.microsoft.com/ntserver/, the Windows NT Server Forum on the Microsoft Network (GO WORD: MSNTS).
