Chapter 1

Understanding Active Directory

In This Chapter

- Defining Active Directory
- Examining the origins of Active Directory: X.500
- Understanding Active Directory terms
- Investigating the benefits of Active Directory: What's in it for you?

Since the release of Active Directory in Windows 2000 Server, Active Directory has become a very integral part of many information technology (IT) environments. As such, Active Directory has become a very popular topic with the people that have to design and support it. Because of all the terms and technology surrounding Active Directory, you might already be a bit intimidated by the prospect of working with it yourself.

But Active Directory doesn't need to be difficult! In this chapter, you find out in clear and simple language what Active Directory is, what it does, and what benefits it brings to your organization and to your job.

What Is Active Directory?

If you visit the Microsoft Web site seeking a definition of Active Directory (AD), you find words such as *hierarchical, distributed, extensible,* and *integrated*. Then you stumble across terms such as *trees, forests,* and *leaf objects* in combination with the usual abbreviations and standards: TCP/IP, DNS, X.500, LDAP. The whole thing quickly becomes pretty overwhelming. (Appendix B has a glossary that defines these abbreviations for you!)

I prefer to define things in simpler terms, as the following sections demonstrate — drum roll, please . . .

Active Directory is an umbrella

What? Am I saying that if it's raining you had better have AD with you? No, I would still recommend a real umbrella in a rainstorm. I'm saying that in Windows Server 2008, the scope of what Active Directory is has greatly expanded. Active Directory has become an umbrella for a number of technologies beyond what AD was in Windows 2000 Server and Windows Server 2003. (See Figure 1-1.)

You discover new uses for Active Directory in the paragraphs that follow.

Active Directory Domain Services

What was *AD* in the two previous Windows Server operating systems is now *Active Directory Domain Services,* or *AD DS,* in Windows Server 2008. The majority of this book deals with this component of Active Directory because this is the most commonly deployed component of the AD umbrella. But don't worry; I discuss all the other technologies found beneath the Active Directory umbrella as well.

Active Directory Lightweight Directory Services

Beginning with Windows Server 2003, Microsoft created a directory service application separate from Active Directory called *Active Directory Application Mode* or *ADAM* for short. ADAM was designed to address an organization's needs to deploy a directory service that didn't necessarily need all the features that Active Directory provided. Microsoft includes this application in Windows Server 2008 but renamed it *Active Directory Lightweight Directory Services* or *AD LDS*. I talk about AD LDS in Chapter 8.

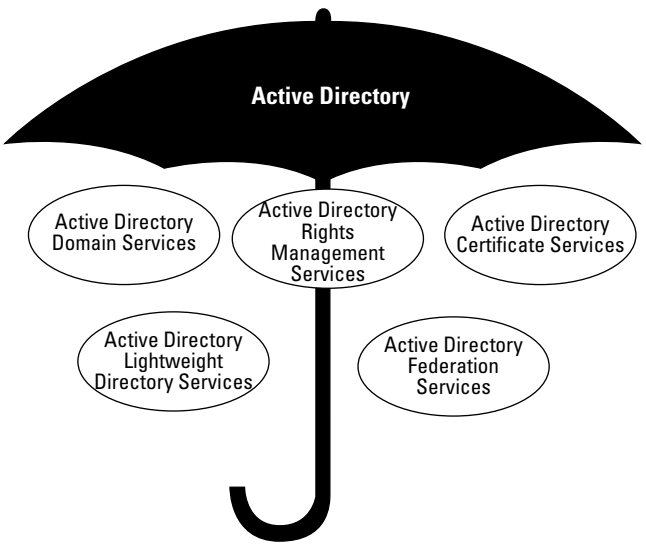

Figure 1-1: The Active Directory umbrella.

Active Directory Federation Services

Beginning in the R2 release of Windows Server 2003, Microsoft included an optional software package called *Federation Services*. As you see later in this book, federations provide a Single Sign-on (SSO) service helping to minimize the number of logon IDs and passwords users must remember as well as simplifying how users can access resources in other IT environments. This software is now a part of the Windows Server 2008 AD umbrella and has been renamed *Active Directory Federation Services* or *AD FS*.

Active Directory Certificate Services

Certificate Services has been around in Windows Server software for a while now. With this software, you can provide certification authorities that can issue public key certificates used for such things as authentication via smart cards or encrypting data before it's transmitted over a network. Certificate Services also provides the necessary management of these certificates so that they can be renewed and revoked. In Windows Server 2008, Certificate Services is a part of Active Directory and is referred to as *Active Directory Certificate Services (AD CS)*.

Active Directory Rights Management Services

Managing what users can do with data has always been an issue for most organizations. Although Active Directory did a good job of controlling whether a user could access a document, it didn't have the ability to control what that user did with the data after he or she got it. Enter *Active Directory Rights Management Services (AD RMS)*. With a properly deployed AD RMS environment, organizations can retain control over sensitive documents, for example, so that they cannot be e-mailed to unauthorized users.

I use the term *Active Directory* interchangeably with *Active Directory Domain Services*. This is because in previous versions of Windows Server software, Active Directory was what is now called *Active Directory Domain Services.* When I refer to the Active Directory umbrella as Active Directory, I make it clear that I'm not just talking about AD DS. Additionally, when I refer to the other elements of AD, such as Active Directory Federation Services, I call it that or use its acronym.

Active Directory is an information store

First and foremost, Active Directory is a store of information. This information is organized into individual objects of data, each object having a certain set of attributes associated with it. A telephone white pages directory, for example, is an information store. Each object in this store represents a home or business that contains attributes for such information as names, addresses, and telephone numbers (see Figure 1-2).

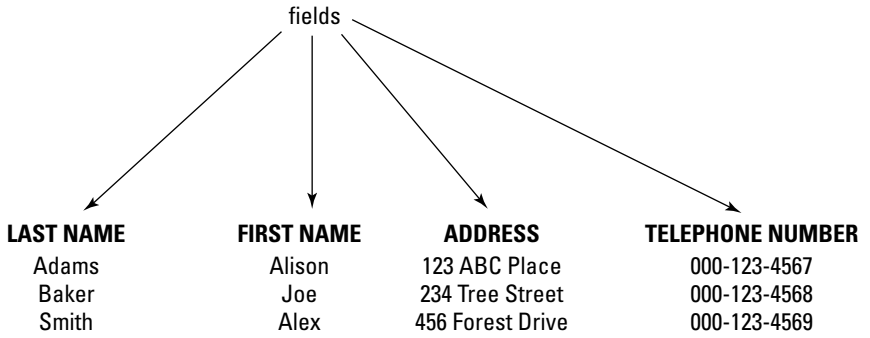

Figure 1-2: A telephone directory is a store containing fields of information.

This store of data as well as the capability of retrieving and modifying the data makes Active Directory a *directory service*. Why then don't I consider Active Directory to be a database? It certainly shares some common functionality including storage, retrieval, and replication of data, but there are some important differences, too. First, directory services are normally optimized for reads because these are the vast majority of the operations executed, and the data is generally non-changing. Also, the data is structured in some sort of hierarchy that allows for it to be organized in the directory store. Repeating my phone book analogy, the Yellow Pages organizes objects by types of business. This makes finding what you're looking for easier. The same can be said of a directory service — you can organize your objects into a hierarchy of containers so that finding the objects is easier. In comparison, a relational database, such as Microsoft SQL Server, is designed to optimize both reads and writes to the store because the data is frequently being read and written to. Also, a database generally doesn't force a hierarchy on the data like a directory service does.

Where did it come from?

Active Directory Domain Services has evolved, but it actually began its life as the directory service for Microsoft Exchange Server V4.0 through V5.5. AD DS actually derives from a directory service standard — *X.500*. The X.500 standard is a set of recommendations for designers of directory services to ensure that the products of various vendors can work together. These are the X.500 protocols:

- Directory Access Protocol (DAP)
- Directory System Protocol (DSP)
- Directory Information Shadowing Protocol (DISP)
- Directory Operational Binding Management Protocol (DOP)

Active Directory, however, actually uses the Lightweight Directory Access Protocol (LDAP) Version 3 (defined in RFC 1777 and RFC 2251), to access the directory database instead of using any of the preceding X.500 protocols. Therefore, Active Directory is X.500 *compatible,* meaning that it can work with other X.500-based directory services, but not X.500 *compliant* — it doesn't strictly adhere to all the X.500 specifications.

In Active Directory, the term *object* can refer to a user, a group, a printer, or any other real component and its accompanying attributes. Active Directory is an information store containing all the objects in your Windows 2008 environment.

Active Directory has a structure (Or hierarchy)

A directory service, such as Active Directory, allows for the objects in it to be stored in a hierarchy or structure. This structure is one of the areas that you design as a part of deploying Active Directory. This structure has two sides:

- **A logical side:** The logical structure provides for the organization of the objects. These AD objects can represent users, computers, groups, and a variety of other items that are in your IT environment. This structure is primarily dependent on how you want to administer your IT infrastructure as well as how your organization is structured.
- **A physical side:** All the services under the Active Directory umbrella are provided by servers running the AD software. These servers represent physical objects that must be placed within your network. After these servers are placed, you must define how these servers speak to each other and how users are directed to them. This physical topology is critical to proper AD functionality.

Staying with the phone book analogy, unless the books are placed in the proper locations (homes, restaurants, pay phones), no one can find the books to utilize the information contained within them.

Active Directory can be customized

As you can with an electronic phone book, you can search Active Directory for the objects that you want to access. Unlike a phone book, however, you can customize Active Directory to include additional objects and object attributes that you deem important. This feature makes Active Directory *extensible,* which means that you can add to it.

Getting Hip to Active Directory Lingo

Experience shows that new terminology often accompanies new technologies, and Active Directory is no exception. Although most of the terms that you use in describing the system might seem familiar, they take on new meaning in relation to Active Directory. So before beginning to plan and implement Active Directory, you need to master its new language.

The building blocks of Active Directory

Active Directory embodies both a *physical* and a *logical* structure. The *physical structure* encompasses the network configuration, network devices, and network bandwidth. The *logical structure* is conceptual; it aims to match the Active Directory configuration to the business processes of a corporation or organization. In the best logical structures, Active Directory resources are structured for how employees work and how the environment is administrated, not to simplify construction of the network.

If you logically organize the components within the Active Directory, the actual physical structure of the network becomes inconsequential to the end-users. If user JoeB wants to print to a printer named A5, for example, he no longer needs to know which server hosts the printer or in which domain the print server resides. In Active Directory, he simply pulls up an Active Directory list of all available printers and chooses printer A5.

Although you might think that this process sounds too good to be true, this new functionality doesn't quite configure itself! You, the system administrator, must first design the logical structure of your organization's Active Directory, matching its structure to how employees interact within the organization. Chapters 2 through 7 help you to plan and implement, but first, you must be familiar with the individual components that you use for planning the physical and logical structures.

Domain

In Active Directory, Microsoft defines a *domain* as a security boundary or an administrative boundary, which means that all the users within a domain normally function under the same security policy and user-account policy. If you want to assign different policies to some users, those users belong in a separate domain.

JohnB, for example, is a regular user in the Sales department who must change his password every 30 days. SueD, on the other hand, is a user in the Treasury department who has access to sensitive information and, therefore, must change her password every 14 days. The two departments — Sales and Treasury — have different user-account policy settings. Because you assign user-account policies according to domain, users in these two departments belong in separate domains.

In Windows Server 2008, the lines between domain boundaries and password policies has blurred somewhat. Normally, all users in a domain receive the same password policy; however, in 2008, you can do some fine-tuning so that users in the same domain actually receive different policies. I cover this in more detail in Chapter 14.

Here are some other important characteristics of an Active Directory domain:

- A domain has at least one *domain controller*. A domain controller is a server that *authenticates* (validates the password and ID) users seeking access to the domain. You find out more about domain controllers in a moment.
- A domain's directory database *replicates* between all domain controllers in the domain. Replication is the exchange of updated information among domain controllers so that all the domain controllers contain identical information.
- A single domain can form a *tree* (which you find out more about in the following section).

In the design process for the logical structure of an Active Directory database, you typically use a triangle in the design flowchart to represent a domain (see Figure 1-3).

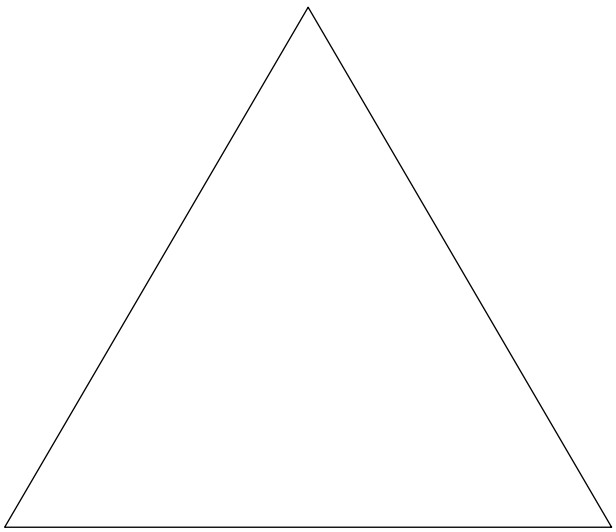

Figure 1-3: A triangle represents a domain when drawing an AD logical design.

Consider defining an additional domain to keep replication traffic local — confined among domain controllers connected by a local area network (LAN). The transmission speed between domain controllers in a LAN is much faster than it is between domain controllers that are connected by a slower, wide area network (WAN). The exchange of updated database information among domain controllers during replication causes additional traffic that can clog the network and result in slower response times. So by keeping your replication local, you can keep replication time to a minimum and ensure that the network lines are available for other traffic. (I talk more about defining domains in Chapter 5.)

Tree

A *tree* is a hierarchical grouping of domains within the same namespace. A *namespace* is a logically structured naming convention in which all objects are connected in an unbroken sequence. (I talk more about namespaces later in this chapter and in Chapter 4.) When you design an Active Directory tree, you begin with the topmost domain, which oddly enough is the *root* (or *parent*) domain. Subdomains (sometimes *child domains*) branch downward from the root, as shown in Figure 1-4. Supposedly, if you turn your logical structure drawing upside down, it resembles a tree. (Go on — turn the book upside down and look for the image of a tree in Figure 1-4!)

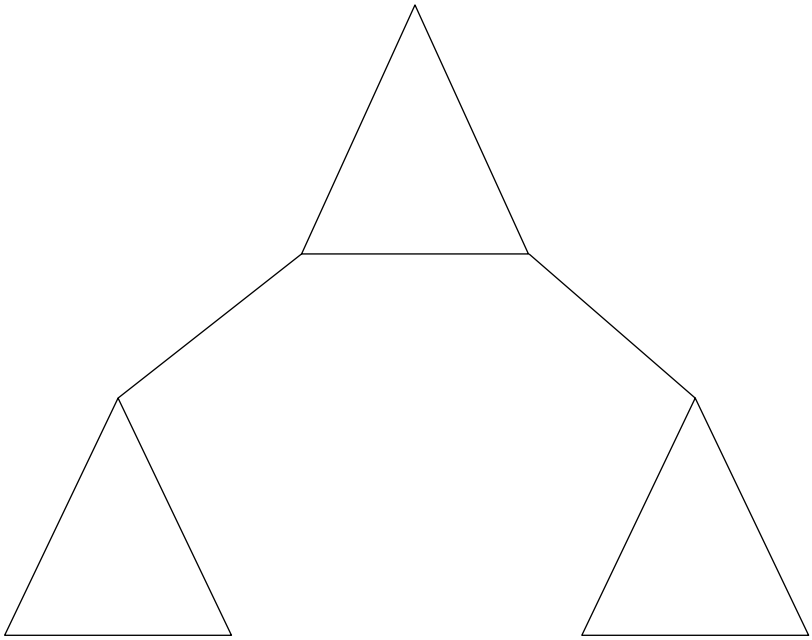

Figure 1-4: A tree diagram in Active Directory.

Regardless of whether you actually see a tree when you turn the book upside down, the term *tree* is one that you use often in discussing directory services. And the arboricultural (it's a real word — honest!) terminology doesn't stop there — as you discover when you find out more about Active Directory.

When you add domains to an Active Directory tree, you automatically create *transitive* trust relationships. Transitive trusts extend the relationship between two trusted domains to any other domains that those two domains trust. These trusts are bidirectional and enable users in one domain to access resources in the other domain. In an Active Directory tree, all domains are connected through transitive trusts, so a user in one domain can access any other domain in the tree.

You can also link trees or forests through *explicit,* or one-way, trusts. By creating an explicit trust between Tree A and Tree B, for example, you can specify that users from Tree A can access resources in Tree B, but users in Tree B cannot access resources in Tree A.

Forest

A *forest* is a logical grouping of trees that you join together in a transitive trust relationship, as shown in Figure 1-5. A forest has the following characteristics:

- Each tree in a forest has a distinct namespace.
- The trees in a forest share the same schema and global catalog. (I discuss schema and global catalog a little later in this chapter.)

Chapter 5 helps you determine when to create a tree and when to create a forest.

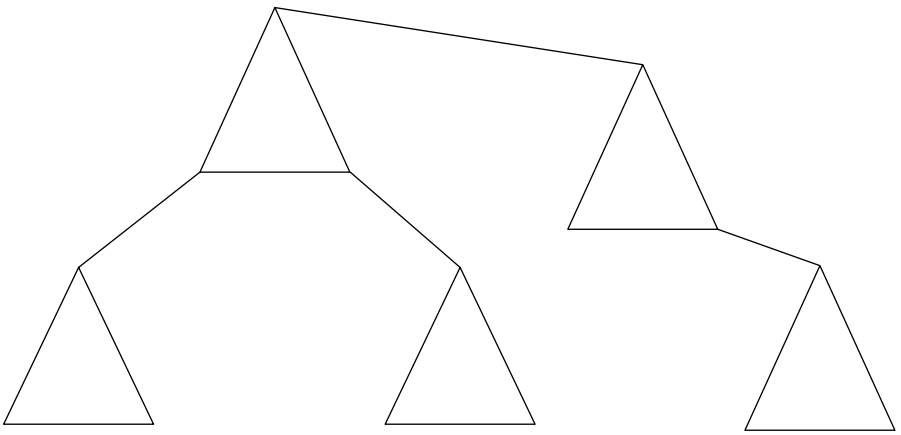

Figure 1-5: A diagram of an Active Directory forest.

Organizational unit (OU)

An *organizational unit* (or OU) is nothing more than a container within a domain. You use it to store similar objects so that they're in a convenient location for administration and access. Here are some of the objects that you store in an OU:

- Printers
- File shares (a folder located anywhere on the network that has been designated as *shared* so that others can access it)
- Users
- Groups (a grouping of users that can be jointly administered)
- Applications

While you plan your Active Directory structure, you also plan the logical structure of the OUs within each domain. Keep the following points in mind as you become familiar with OUs:

- You can *nest* OUs within each other to create a hierarchical structure.
- Each domain can have a hierarchy of OUs, or the OU hierarchy can be identical in each domain. You cannot, however, extend an OU across domains. OUs are always completely contained within a single domain.
- Structure OUs correspond with the business practices of your company. Earlier in the chapter, I talk about matching the logical structure to where employees work. OUs can help you organize network resources so that they're easy to locate and manage.

Many factors can influence your OU structure or model. An OU model might reflect the administrative model of the organization or the company's structure either by organizational chart or by work locations.

A domain that you name West, for example, represents your company's western region of the United States. This domain includes OUs that you name California, Washington, and Oregon, as shown in Figure 1-6. The California OU contains two nested OUs that you name San Francisco and San Diego. The Washington OU contains objects that you organize in OUs that you name Tacoma and Seattle. To ease administration by keeping things similar, the East domain follows the same conventions used in the West domain.

If you want, you can further organize the city OUs so that San Francisco, San Diego, Tacoma, and Seattle each contain nested OUs for user objects and printer objects.

You can create transitive trusts between forests A and B so that all the domains in Forest A trust all the domains in Forest B and vice versa. Having forest-level transitive trust can greatly simplify your life!

Object

An *object* is any component within your Active Directory environment. (I talk briefly about objects in the "Active Directory is an information store" section earlier in this chapter.) A printer, a user, and a group, for example, are all objects. All objects contain descriptive information, or *attributes*.

Sites and Site Links

A *site* is a grouping of IP subnets connected by high-speed or high-bandwidth links. Sites are part of your network's physical topology (or physical shape), and each site can contain domain controllers from one or more domains.

During your planning stages for implementing Active Directory, you define a site topology for your environment. You use sites to optimize a network's bandwidth by controlling replication and logon-authentication traffic. (Chapter 12 tells you how to use sites to control traffic.)

By dividing the network into sites, you can limit the amount of replicated Active Directory data that you must send across slow WAN links. Domain controllers within a single site exchange uncompressed data because they're connected by fast links; domain controllers spread across different sites exchange compressed data to minimize traffic.

Of course, you can't just define sites and then expect the sites to start magically communicating with each other. You must define site links that connect your sites. These site links define how the replication and logon-authentication traffic flows between sites.

I devote Chapter 12 to a discussion of controlling replication traffic. But for now, just be aware that replication occurs whenever the domain controllers within a domain exchange directory database information. Updates or additions to the database trigger replication between domain controllers within a site.

You also use sites as authentication boundaries for network clients. Although any domain controller throughout the domain can authenticate a user, designating any but the closest one to do so isn't always the most efficient use of the network. After you specify your site boundaries, the closest available domain controller within the client's site authenticates a client logon. This setup minimizes authentication traffic on the network and speeds response time for the client.

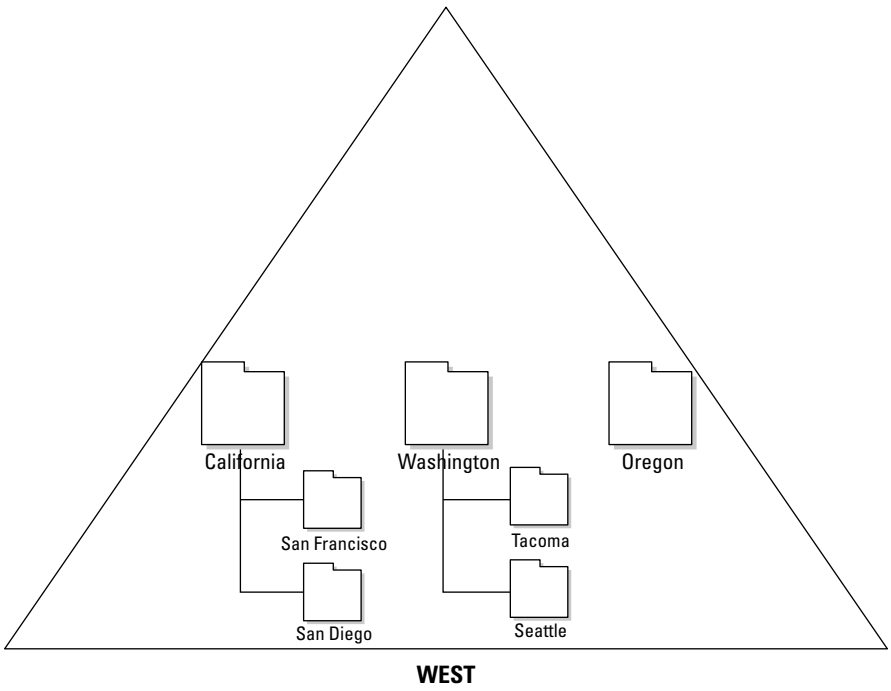

Figure 1-6: Nested organizational units (OUs) in Active Directory.

Object Identifiers (OIDs)

If you decide you want to create your own schema changes, you will need your own *Object Identifier.* Object Identifiers are dotted decimal numbers that the American National Standards Institute (ANSI) assigns to each object class and attribute. ANSI assigns a specific root identifier to a U.S. corporation or organization, and the corporation then assigns variations of its root identifier to the objects and attributes that it creates. For example, Microsoft's OID is 1.3.6.1.4.1.311, which maps to the following path:

```
iso.org.dod.internet.private.
   enterprise.microsoft.
```

The Active Directory schema

Along with the basic Active Directory components that I discuss in the preceding sections, you must also be familiar with the Active Directory *schema*. The schema contains definitions of all object classes (or object categories) and attributes that make up that object. That is, the schema is where the rules are about what kind of objects can be stored in the directory and what attributes are associated with each type of object.

Normally, an AD administrator doesn't make changes to the schema on a regular basis. The majority of the time, you modify the schema only when you're installing an application that uses Active Directory to store and retrieve information. One good example is Microsoft Exchange Server. A number of new attributes and object classes must be created and modified so that Exchange can work. But there can be instances where you might perform a schema modification on your own. For example, let's assume that all the employees of Steveco Corp. have a company-specific attribute (say, an employee number) associated with them and you want to put that information into Active Directory. There isn't any attribute in the default schema called `SteveCoEmpNum` so you must make the necessary changes to the schema to include this attribute.

At the time that you install Active Directory, you also install a base schema by default. This schema contains the object class definitions and attributes of all components available in Windows Server 2008. While your directory tree grows, you can extend or modify the schema by adding or altering classes and attributes as follows:

- You can create a new object class.
- You can create a new attribute.
- You can modify an object class.

- You can modify an attribute.
- You can disable an object class.
- You can disable an attribute.

(In Chapter 13, I show you how to do all the schema modifications shown in the preceding list.)

By definition, an object must have defining attributes; each object has *required attributes* and *optional attributes*. Among the required attributes of any object are the following:

- Name
- Object Identifier (OID) (See the "Object Identifiers (OIDs)" sidebar.)
- List of required attributes
- List of optional attributes

Doesn't it seem odd that a list of *optional* attributes is a *required* attribute for an object? Of course your list of optional attributes could be empty!

Not just anyone can modify the directory schema. Only members of the Schema Administrators group can do so. The Schema Administrators group is a built-in group installed by default when you install Active Directory. The group is preconfigured with the appropriate privileges for performing particular tasks. As system administrator, you can assign particular users to this group by adding their user IDs to the group. (See Chapter 11 for the details on adding users to groups.)

Limit the number of administrators in your organization's Schema Administrators group to protect yourself against unintended results! Every organization should have a precise change-control policy that governs changes to the directory schema. The schema affects an entire forest, so any change is replicated to every domain in the forest. The potential for disaster is huge!

Domain Controllers and the global catalog

Domain controllers (DCs) are the servers that actually provide all the AD DS services as well as the actual storage of the directory data. The AD data on the DC is split into four types of regions or partitions:

- **Domain Naming Partition:** Each domain in the forest has at least one domain controller that is a member of that domain. The Domain Naming Partition is where the copy of all the objects within this domain controller's domain is stored. This information is replicated to all other domains controllers within the same domain. Every DC has a single domain naming partition because the DC can only be in one domain.

- **Configuration Partition:** This partition is used to store information that's needed across all domain controllers in the same AD forest. Within the configuration partition, the information about the physical environment, including site and site link definitions is held. This partition is located on every domain controller in the forest.
- **Schema:** Every domain controller in a forest has an identical local copy of the Active Directory schema stored in a schema partition. That way, every DC understands the rules of what objects and attributes can exist.
- **Application:** Application partitions are optional partitions that can be used to store data that is to be replicated between a set of domain controllers and used by an AD-enabled application. One good example is DNS, as I discuss in Chapter 4.

The replication of these partitions between the domain controllers is handled with a *multimaster model.* What does that mean? Multimaster model means that changes to these partitions can be on any DC and those changes will be replicated to every copy of that partition in the forest. Of course, there are some exceptions to this rule (you knew there would be!). Because of a schema's critical nature, only one DC in the forest has a writeable copy of the schema — the Schema Master. Table 1-1 summarizes these partitions and their replication method and scope.

Table 1-1 Active Directory Partition Replication

Partition Type	*Multimaster*	*Replication Scope*
Domain naming	Yes	Domain-wide
Configuration	Yes	Forest-wide
Schema	No	Forest-wide
Application	Yes	Domain controller–specific within the same forest

Windows Server 2008 AD DS introduces a new type of domain controller — a read-only Domain Controller, or RODC. I cover RODCs in detail in Chapter 6, but for now, understand that there's a special case when you can configure a DC where none of the partitions on a DC are writeable. You will see that RODCs are a great solution for deploying AD DS services in smaller, less secure locations.

Domain controllers provide two primary services to users: network authentication and directory object storage and retrieval. *Network authentication services* are provided by a DC through the Kerberos Key Distribution Center (KDC). In Active Directory security, Kerberos is everything. Every Active Directory user must get a Kerberos key at login. This key identifies the user to the network and controls what resources the user can access. In addition to the KDC, DCs provide the ability to store and retrieve the directory information in the partitions that the DC holds.

One other option on a domain controller that you need to understand is the *global catalog*. A global catalog (GC) is a searchable index that enables users to locate network objects without needing to know their domain locations. A partial replica of the Active Directory, GCs contain a list of objects in the forest but don't necessarily list all the attributes of every object in the forest. GCs aren't separate from domain controllers: They're an option that you can select on the DC's configuration. In other words, all GCs are DCs but all DCs aren't necessarily GCs.

The global catalog enables searches among trees in a forest. You can also use it to speed lengthy searches within a single tree. By default, the global catalog doesn't contain all the attributes of every object. The default global catalog configuration includes only those attributes that you're most likely to use for a search, such as a user's first or last name. Similarly, you can search the global catalog for all color printers instead of browsing through all the printers on the network.

The default schema settings determine which object attributes appear in the global catalog. All objects appear in the global catalog, but only a small subset of the objects' attributes are included. To add additional attributes to the global catalog, you have to modify the schema. (See Chapter 13 for additional information on modifying the schema.)

By default, the first domain controller that you create in a forest becomes a global catalog server. If the environment consists of multiple sites, you can optimize network traffic by creating a global catalog server in each site.

The *global catalog* is a service that runs on domain controllers. You manage the service by using the Active Directory Sites and Services snap-in for the Microsoft Management Console (MMC). The MMC is a Windows 2008 Server system file that you access by choosing Run from the Start menu and then typing **mmc**. From within MMC, open the Console menu, choose Add/Remove Snap-in, and then choose AD Sites and Services from the list that appears.

The DNS namespace

DNS (Domain Name Service) is the predominant name-resolution service on the Internet, so Microsoft chose to use DNS to translate host names to IP addresses in the Active Directory service. The DNS namespace is the single most important requirement for a successful Active Directory implementation, and the two are tightly interwoven. If you don't plan the DNS namespace appropriately, your Active Directory service is difficult to administer and doesn't adequately serve the user community.

A thorough understanding of DNS and of TCP/IP is essential for planning and implementing Windows 2000 and Active Directory. A good source of information is *TCP/IP For Dummies* by Cameron Brandon (published by Wiley).

I discuss in Chapter 4 that you must plan the DNS namespace before you can design the Active Directory. You use the DNS namespace design that you create (or one that already exists for your organization) to design a domain namespace for Active Directory.

If you're not using the Microsoft DNS service, you must use another DNS service that's compliant with RFC 2136 and RFC 2052.

Because It's Good for You: The Benefits of Active Directory

I don't know about you, but whenever Mom told me to eat my vegetables because "they're good for you," I still wasn't particularly motivated. I needed to know more about what that broccoli was actually going to do for me.

So maybe, like me with my vegetables, you need to hear about the real benefits you ultimately can realize if you bite the bullet now and make the management and design changes required by Windows 2008 and Active Directory.

Active Directory offers appealing features for administrators and end-users alike:

- Ease of management because of the centralized nature of the Active Directory database.
- Enhanced scalability (it can get lots bigger!) that enables the Active Directory database to hold millions of objects without altering the administrative model.
- A searchable catalog that enables you to quickly and easily search network resources. The network becomes less intrusive, enabling users to concentrate on their work rather than their tools.
- Active Directory forms an infrastructure backbone that many IT platforms and applications can utilize.

I encourage you to follow through all the planning and testing steps that I present in this book. With the right preparation, Active Directory can offer tremendous advantages for both you and your organization.