[**Editor's Note:** The following excerpt is from Chapter 2 of the free eBook *The Definitive Guide to Windows 2000 Security* (Realtimepublishers.com) written by Paul Cooke and available at http://www.bindview.com/ebook/.]

Active Directory Best Practices

Win2K has been around for well over a year now, but there isn't a lot of consensus about how to best provide security in AD. This can be partly attributed to the fact that AD is actually quite complicated and not as well understood as it should be. I've even heard that Microsoft has problems trying to fully understand the complexities of AD. Another reason is the lack of AD implementations that are out there.

Sure, Microsoft and a few other large organizations have implemented AD in their environments, but most organizations have taken a wait-and-see attitude toward AD; they've deployed Win2K to their desktops and server infrastructures while leaving their NT 4.0 domain infrastructure fully intact. Nevertheless, there are some basic best practices when it comes to AD that I don't think many people would argue with. This will be the focus of this section.

Native-Mode Domains

A homogeneous Win2K environment provides the optimum security capabilities for your enterprise. Unfortunately, many organizations find this a lot harder to achieve than you'd first anticipate. All too often, applications that are critical to the success of your business are running on a legacy version of NT, and management is afraid to touch them, or the applications just won't run on Win2K for some reason. Even worse, you may run into the stubborn vice president who just can't live without his trusty old Pentium 90 running Windows 95. So while your goal is to have all your computers running Win2K, it may not be feasible to achieve it immediately.

If you can't update your entire environment to Win2K, I suggest that you at least run your domain infrastructure as a homogeneous Win2K environment in native mode. Using the PDC emulator role, your domain infrastructure can still support down-level, non-AD–aware applications while experiencing the benefits of running in native mode.

Domain Forests

When you first deploy AD in your enterprise, you're best off starting with a single forest design. This design is the simplest to create and maintain. It also benefits from the fact that all of your domain trusts will be automatic, two-way, and transitive in nature. Your users will also benefit because they only have to search a single forest to find resources in the environment.

I'm a strong believer in a single-forest implementation, but situations do arise that necessitate creating more than a single forest. Typically, a multiple-forest deployment becomes necessary because of trust issues and the need to isolate a domain in one forest from other domains in other forests. These issues of trust tend to arise when different parts of an organization want control over the ability to add and delete domains from the environment, control over schema modifications and change procedures, and tighter control over who accesses their resources.

While I strongly urge a single production forest for all of your users, you might want to keep a separate forest for integration testing with your environment. This will allow you to design solutions for your enterprise without disturbing your core business.

Domains and Domain Trees

I lean heavily toward a single forest for most organizations, but I don't think it's necessarily the best approach for all organizations with regard to domain trees. If your organization is relatively small and located in a single geographic region, maybe a single domain is best for you. Microsoft, and others, urge adopters of AD to start their designs with a single domain, and they seem to believe that there are only three reasons to use more than a single domain: to preserve any existing Windows NT domains, to create administrative partitioning, and/or to create physical partitioning.

While I understand the rationale behind the first reason, I don't really buy it. Many deployments of NT 4.0 domains suffer because of the limitations of SAM. There is no reason to carry this baggage with you as you build your Win2K infrastructure. It's the second two reasons that I think should drive all but the smallest organizations to consider at least two domains.

You also need to consider multiple domains if a variety of security policies exist for user and password policies. Although I'll talk about this more in the section on Group Policy, I'd like to point out now that there is a small set of security policies that can only be applied on a domain basis. These policies apply to domain user accounts as follows: password policy, account lockout policy, and Kerberos policy.

There is also another reason why you might consider multiple domains, and that is if your organization uses multiple DNS domain names. This often occurs when one company is acquired by another, but there are many other situations in which you may need to support multiple domain trees. It's also hard to predict what future mergers and acquisitions your organization may be involved in, so even if you do end up with a single domain implementation of AD, you need to have a plan for incorporating more in the future.

Organizational Units

The first thing that everyone is tempted to do when they start designing their OU structure is to model it after the structure of their organization. I have one thing to say about this: Don't do it. Creating an OU structure that mirrors your business structure will prove very difficult to maintain. Sure, OUs are much easier to create, delete, move, and manage than, say, a domain, but it isn't wise to follow this path.

In my opinion, Microsoft made a mistake when they called this container an organizational unit. From my perspective and in my experience, it should have been called an administrative unit. This is where OUs are really at their best. OUs should be used for delegating administration, not creating some pretty hierarchical tree in your Active Directory Users and Computers MMC snap-in. When you're designing the OU structure for each of your domains, I believe that you should only create OUs when you want to delegate administration. I've heard others suggest that you should also create OUs to apply Group Policy or to hide objects. Ultimately, I believe in keeping it simple and only creating an OU to help delegate administrative functions.

Delegated Administration

One of the great promises of AD was its ability to provide *delegated administration*. As I discussed earlier in this chapter, the Delegation of Control wizard is the primary tool provided by Win2K for delegating administrative functionality.

In Chapter 1, I talked briefly about role-based authorizations. When it comes time to delegate administrative tasks, role-based authorizations are the best approach. The first step in creating a role-based authorization is to create a new security group, then use the Delegation of Control wizard to delegate the appropriate AD rights to the group. As people need to perform this role, you can add them to the appropriate security group.

☞ Never assign rights, privileges, or ACLs to an individual computer or user object. Instead, create a security group, assign the appropriate permissions to it, then add computer or user objects to it. For example, say you found out that your Help Desk needed permissions to perform another task. Changing permissions or privileges on a single group is far preferable to modifying a couple of hundred individual user objects.

While it's tempting to create a few dozen administrative roles right from the start, I strongly caution you against it. AD is so complicated that I've seen entire Win2K domains dismantled because delegated administration groups were missing key rights or privileges in AD that kept them from functioning. I recommend that you strive initially for three to four levels of administration delegation, as follows:

- **Help-Desk Administrators**—Primarily to troubleshoot and reset passwords
- **AD Administrators**—To add and delete users, groups, and computers but not reset passwords
- **Domain Administrators**—Built-in group with control over an entire domain
- **Enterprise Administrators**—Built-in group with control over an entire forest.

While this is better than what NT 4.0 provided right out of the box, it's probably a lot less than you expected after hearing all the hype about Win2K's delegation of administration capabilities. If you want more than three or four roles (and who doesn't), I strongly urge you to consider using a third-party tool to manage administrative roles.

One of the rumors on the street is that even Microsoft figured out that administering AD with its own tools wasn't going to work, so it uses a third-party product. But in its defense, it seems to have been very clear during the development of Win2K and AD that Microsoft actively encouraged third-parties to fill in the administration gaps left behind in the OS.

☞ If you're interested in third-party administration tools for Win2K, here are three products that you can evaluate:

bv-Admin for Active Directory from BindView Corporation (www.bindview.com)

ActiveRoles from FastLane Technologies (www.fastlane.com)

Enterprise Delegation Manager from Aelita Software Corporation (www.aelita.com)

Securing DHCP and DNS

If you've ever experienced a rogue DHCP server, you know what a mess it can cause. In a standard DNS environment, you could end up with duplicate IP addresses and communications among computers being interfered with. In a DDNS environment, requests for name resolution can be given incorrect addresses and further confound your users. Win2K does provide some limited protection against rogue DHCP servers, but only if they're also running on Win2K.

By default, a DHCP server running on the first domain controller of a forest is authorized to dispense IP addresses, but any other installed DHCP server must first be authorized in AD. To do this, choose Action>Authorize in the DHCP MMC snap-in. Keep in mind, though, that to authorize new DHCP servers, you must be a member of the Enterprise Administrators group. (This is discussed in Chapter 5.)

Authorizing a DHCP server before it allocates addresses won't keep someone from installing a DHCP service on a computer. But if it's installed on a Win2K domain member server or domain controller, it'll keep the service from running. To ensure that your environment is devoid of non-Win2K, rogue DHCP servers, you'll have to perform regular network audits.

Once your DHCP servers are all running on Win2K and authorized in AD, turn your attention to securing your Win2K DNS servers. You can easily secure DDNS by configuring it for secure updates. This requires that your DNS zones are AD-integrated zones, that you modify zones to allow Only Secure Updates, and that you specify which users and groups can modify zones and resource records using ACLs.

Once you've configured DDNS, only computers with domain accounts can create DNS records, only the computer that created a record can update it, and zone updates are accepted only from the DNS servers you specify in your ACLs. If you still have legacy clients, don't be concerned; if they use the secured DHCP server, it will add and own the DNS record for the client.

[**Editor's Note:** This content was excerpted from the free eBook *The Definitive Guide to Windows 2000 Security* (Realtimepublishers.com) written by Paul Cooke and available at http://www.bindview.com/ebook/.]