Avoid the Pains from Active Directory

Contributed by David Noel-Davies

Over the past eight years I&rsquo;ve helped plan, implement, and operate various Active Directory (AD) infrastructures. And as much as I value AD&rsquo;s power and strength, I&rsquo;ve also learned quite a few annoying things about AD that sometimes prevent it from operating as smoothly as possible. In this article I discuss some of these annoyances and explain how to best work around them. Special Hardware Problems
In general, all AD domains are rather tolerant to hardware problems that take down a single domain controller (DC). Of course this is only true if you follow the best practice of implementing more than one DC per domain and if you continuously monitor that they&rsquo;re replicating the changes amongst themselves. This way, if one DC fails for some reason, clients wanting to authenticate to the domain will leverage DNS to find another DC in the network to connect and authenticate to. For normal operations, no problems occur even if one of the special Flexible Single-Master Operation role-holder DCs goes down for a few hours or even a few days. AD is designed to operate without all the FSMO DCs being available all the time. Obviously, you shouldn&rsquo;t update your schema or mass-create new objects in your domain when specific FMSO DCs are down. But normal operations, such as users changing their passwords or administrators adding an occasional new object to the domain, will still run. This is one of the key strengths of AD and its multi-master replication model.
But sometimes it isn&rsquo;t the hardware failure of the DC that causes a problem. Sometimes problems don&rsquo;t start until you repair the hardware and reboot the DC&mdash;especially if the DC is your domain&rsquo;s PDC emulator. By default, all DCs in an AD domain synchronize their time with the PDC emulator of the respective domain. Computers and servers joined to the domain then synchronize their time with the DC they use for authentication&mdash;usually a DC within their AD site. For Kerberos authentication to work, all these clients and DCs must be synchronized in time. (In an AD domain, Windows 2000 Server and later clients and servers leverage Kerberos by default.) If the time skew (difference) is too large between a client and the server it wants to access a resource from, such as a file share, authentication to the resource server fails. The default accepted time skew in an AD forest is five minutes. So even if a user or computer properly authenticates to a domain, it might fail to access a server because of a time difference. What does all of this have to do with the hardware failure of a DC in your domain, potentially even the PDC? Quite simple: If your hardware repair involves replacing a server&rsquo;s motherboard, you usually also replace the on-board clock. And it&rsquo;s highly unlikely that the time set on the new motherboard&rsquo;s system clock is in sync with the rest of your AD forest. If you then just reboot the PDC while it&rsquo;s on the network, the other DCs will synchronize their times with the PDC when they see that it&rsquo;s online again. Thus, you might introduce a time skew on various machines in your environment that&rsquo;s unacceptable to Kerberos. Although your PDC might have been properly configured to replicate with an external time source, the wrong time has now made its way into your network and will cause problems such as Microsoft Exchange Server servers not being able to leverage Global Catalogs (GCs) in their site for LDAP lookups, or users not being able to access file shares. Your environment might not normalize for hours or days and might even require manual intervention. The solution to this problem is as simple as the problem itself: If you need to replace a DC&rsquo;s motherboard, particularly for a broken DC that hosts the PDC emulator FSMO role, remove the network cable before you reboot the DC. After the DC reboots successfully (which might take longer than normal because the DC won&rsquo;t be able to find other DCs to replicate with), you need to log on locally and update the time on the DC. Afterwards, you can plug the network cable back in. Alternatively, if your PDC is still responsive, you can temporarily transfer the PDC role to another DC and transfer it back after replacing the motherboard. These methods prevent time-synchronization problems that in turn cause trouble with network authentication. Cross-Forest Authentication
It&rsquo;s difficult enough for most AD administrators to understand how clients leverage DNS to locate DCs of their own forest or domain within their own network. So before we look at how this process might work across different AD forests and networks, let&rsquo;s quickly review the DC location process within an AD domain.In short, a Win2K or later client that has never authenticated to an AD domain will query DNS to ask for any DC that&rsquo;s responsible for its own domain. The client does so by asking the DNS server to return the list of all DCs that have registered the generic DC locator record (which by default includes all DCs in an AD domain). To retrieve these records, the clients first query for the generic LDAP service records in the DNS hierarchy&rsquo;s _msdcs zone. For an AD domain called MyCompany.net, these generic records are located in the following DNS hierarchy: _ldap._tcp.dc._msdcs .MyCompany.net. The client then contacts a few of the DCs in the list returned from the DNS server, notifies them of its intention to be authenticated, and waits for the first DC to respond. However, the DCs are smart enough to understand the situation and therefore check the IP address that the client is using in its request. They see that the client is joined to the domain, and they compare the client&rsquo;s IP address with the site and subnet data stored in the AD configuration partition. With this data, the DCs determine the client&rsquo;s proper site, and they tell the client to connect back to the DNS server and query for the actual DC to authenticate to in its own site. The client then requests the proper Kerberos service record. Let&rsquo;s assume the client is located in a branch office of the MyCompany.net AD domain and the AD site name is BranchSite. The client would query DNS for the sitespecific Kerberos service records registered for this site. These records are located in the following DNS hierarchy: _kerberos._tcp .BranchSite._sites.dc._msdcs.MyCompany .net. The DNS server will then return only DCs that are responsible for the client&rsquo;s site, which the client in turn leverages to authenticate to the domain. Fortunately, the client stores the information for the last AD site it belonged to in the registry and leverages this information directly the next time it needs to locate a DC. To find the AD site name that a client cached for itself, go to the HKEY_LOCAL_MACHINE\SYSTEM\Current ControlSet\Services\Netlogon\Parameters DynamicSiteName registry subkey. Even after a user authenticates to his or her proper domain, cross-domain resource access involves a few more steps in multi-domain forests. Part of the DC locator process is repeated when the user

accesses resources in another domain in the forest. Although all the domains in a forest trust one another, the Kerberos Ticket Granting Ticket (TGT) that the client received from its own DC at logon is valid only for requesting service tickets that in turn grant access to resources in the client&rsquo;s own domain. When a user accesses a resource in another domain in the same forest (e.g., a file server), the client again first queries DNS to locate a DC of the file server&rsquo;s domain to request a TGT that&rsquo;s valid in this domain. The good thing is that the client will immediately find the correct DC to use. Because the client already knows what site it&rsquo;s in, it uses a site-specific DNS query (such as the one I described) to locate a DC of the other domain. One reason this process works so efficiently within a forest without the need to query for the generic DC locator records of the other domain is that all domains in the same forest replicate and use the same AD configuration partition. This partition also contains the site and subnet information, so that DCs from any domains in the forest properly register locator records for the respective AD sites. If no DC exists in an AD site, or if a site doesn&rsquo;t have a DC for every domain in the forest, the AutoSiteCoverage mechanism ensures that the closest DC will register a locator record in DNS. This means that within its forest, a client can always locate a site-specific DC for any domain in the forest. The client doesn&rsquo;t have to leverage the generic DC locator records, which might direct the client to a DC on the other side of the world to authenticate with. However, remember that this process assumes that the site-specific DCs are available&mdash;if not, the client will fail back to leveraging the generic DC locator records and might therefore experience slow authentication and GPO processing. Suppose that you&rsquo;ve just acquired another company that has its own AD forest. To efficiently allow collaboration between the employees of both companies, you decide to establish a trust across both forests. Your plan might be to later consolidate both forests; however, a forest trust is often the first step to allow access to resources in different forests for both parts of a merged company. The annoying thing about cross-forest authentication is that the client frequently fails to locate the correct DC in a trusted forest and is instead authenticated by a random DC&mdash;often not the DC you prefer the client to use. The good news is that you can easily solve this problem. Similar to the cross-domain DC location process that I already described, when a client accesses a resource across a forest trust, the client also queries the trusted forest&rsquo;s DNS servers for the correct DCs for authentication purposes. The key thing to understand here is that the client queries DNS again for the site-specific Kerberos service records. The client does so by combining the site name from its own domain (MyCompany.net) with the domain name of the trusted forest&rsquo;s domain (OtherCompany .net) and would thus query in the following DNS hierarchy: _kerberos._tcp.BranchSite._sites.dc._ msdcs.OtherCompany.net. If no such AD site exists in the trusted forest, which is highly likely for a forest designed by a different AD team, the DNS query fails and the client must request the generic DC locator records. The client might then be authenticated by DCs of the trusted forest that are less than ideal to use, which will effectively slow down the cross-forest access to data and applications. To prevent this problem, simply ensure that you create AD sites with the same names in both forests. Consider these new sites to be &ldquo;shadow&rdquo; sites of the trusted forest. You don&rsquo;t need to add new IP subnets to these sites for this solution to work, nor do the shadow sites need to contain real DCs. Instead, create a dedicated site link between each &ldquo;shadowsite- for-MyCompany&rdquo; and the closest &ldquo;realsite- of-OtherCompany&rdquo; connecting the two sites. Make sure no other sites are joined to this site link and that the shadow sites themselves aren&rsquo;t joined to any other site links. Doing so will guarantee that the proper DC of the OtherCompany.net forest adds the required SRV records to the respective shadow site via AutoSiteCoverage. These shadow sites will now ensure that your clients leverage the correct and closest DCs for authentication when accessing resources across a forest trust. Enhancing the Usability of the Least Privilege Model

As companies have become more sensitive to IT security, AD administrators have begun to use at least two accounts with different privileges: one account for logon to clients and to perform regular office tasks (this account doesn&rsquo;t have any administrative rights in AD) and another account with sufficient privileges in AD to perform administrative tasks such as user and group management. As is often the case, increasing security doesn&rsquo;t make the processes more user friendly for the end user. In this case the end user is the AD administrator. Handling multiple accounts can be a hassle even when using the different OS features, such as right-clicking the MMC Active Directory Users and Computers snap-in and selecting Run as to start the tool with the administrator account. Although this method works, I find it particularly annoying that the Run as dialog box used to enter the administrative credentials never remembers my AD account&mdash;I need to enter it for every situation in which I need to elevate my privileges.My favorite solution to this challenge is to set up a centrally hosted terminal server where all relevant administration tools are installed. AD administrators can then connect to this terminal server, authenticate with their administrator account, and perform all required administrative tasks through the terminal server session. There&rsquo;s no requirement to use Run as, and I can even use RDP files on my desktop that are preconfigured with the correct account name. Of course you shouldn&rsquo;t opt to store the administrator&rsquo;s account password when saving the RDP files. Administrators can also use a central terminal server to connect from any client to perform their duties&mdash;the Windows Server Administration Tools Pack (adminpak.msi) tools don&rsquo;t have to be installed locally on the client. Many environments, however, might be too small to allow implementation of a terminal server purely for administrative purposes, or other reasons might exist for running the AD management tools from your local machine. Another option to avoid the tedious and error-prone task of right-clicking the icons and entering your credentials is to create your own shortcuts that execute the Runas utility directly, fully leveraging its command-line options as well as those of the snap-ins themselves. And if your administrator accounts have a clear prefix or postfix added to the normal user account, you can leverage simple things such as the expansion of the USERNAME environment variables to further simplify managing shortcuts for multiple users. As an example, my normal (unprivileged) user account might be called GUIDOG, whereas my administrative account is ADM.GUIDOG. To allow multiple administrators to use the same shortcut on different machines, I need to expand various environment variables in the shortcut I create, which is achieved by enclosing the respective environment variables between two percent signs. The following is a sample command for a shortcut

I&rsquo;d use to start the Active Directory Users and Computers snap-in with the administrative account and request that it binds to a specific DC in my domain: %windir%\system32\runas /env / user:%userdomain%\adm.%username% &ldquo;mmc %windir%\system32\dsa.msc / server=w2k8core01&rdquo; When I create the shortcut for this command, I can further enhance the warning effect and notify myself that I&rsquo;m switching to a higher privileged account by using the color options of the shortcut itself&mdash;as for any command prompt, you can also configure background and font colors for a shortcut. The command prompt&rsquo;s red background color warns me that I&rsquo;m about to elevate my privileges. Because the Run as command in the shortcut appropriately expands %userdomain% ADM.%username%, I no longer have to enter my account name. Instead I&rsquo;m prompted to enter my password for the CORP\ADM.GUIDOG account right away. Because Windows shortcuts are real files (with .lnk extensions), you can copy them to an appropriate share that&rsquo;s available to any other administrator in your AD forest. So if a different user, JOER for example, also has an administrative account using the same naming convention, he can use the same shortcut for launching the Active Directory Users and Computers snap-in and will be asked to authenticate as CORP\ADM.JOER.
64-Bit Windows Challenges
The current trend is a strong move toward 64-bit Windows versions, especially for applications that can benefit from the increased memory space that 64-bit OSs offer. AD is one of those applications&mdash;if your AD database doesn&rsquo;t fit within the memory limitations that it incurs on 32-bit Windows Server versions, upgrading your DCs to 64-bit Windows on hardware with sufficient physical memory can greatly improve AD&rsquo;s performance, as well as the performance of dependent applications (e.g., Exchange).I won&rsquo;t go into the details of when and where you should consider leveraging 64- bit Windows for your DCs. In general, you won&rsquo;t run into any problems combining 32-bit DCs with 64-bit DCs in your AD forest (e.g., leveraging Windows Server 2003 x64). No 64- bit&ndash;specific schema extensions are required, and 64-bit DCs replicate just fine with their 32-bit counterparts. Of course you need to ensure that you have proper versions of your drivers, antivirus software, and monitoring agents that support your 64-bit Windows OS. However, some of the Microsoft ADrelated management tools that used to run on your DCs might no longer function. The most important ones that I know of are the AD Replication Monitor support tool (Replmon), the Group Policy Management Console (GPMC), and the Active Directory Migration Tool (ADMT) Password Export Server (PES) service. Most 32-bit applications run with no problem on a 64-bit Windows OS because they leverage the OS&rsquo;s Windowson- Windows 64-bit (WoW64) compatibility feature. Replmon, GPMC, and ADMT PES are true exceptions to this rule because they have other dependencies that WoW64 can&rsquo;t meet. Replmon and GPMC run without a problem on a 32-bit member server and can thus be further leveraged in a pure 64-bit AD forest and connect remotely to the 64-bit DCs. ADMT PES must be installed on a DC, so you either need to have a 32-bit DC that you can leverage for this service, or you need to wait for the upcoming ADMT release&mdash;which is slated for release with Windows Server 2008 and will include a 64-bit version of the PES service. Forewarned Is Forearmed
AD is a powerful directory and authentication service. Its multi-master replication model ensures the high availability that companies request from such an important service. But sometimes the small things that might not work as expected are the ones that cause the most pain for an administrator. Knowing about some of these annoyances ahead of time will hopefully help you avoid them in your own infrastructure.