Software

# The Active Directory Management and Security You've Always Dreamed of

Written By Don Jones, Co-founder, Concentrated Technology

## Introduction

Windows security is something we all have to live with, although most of us probably prefer to talk about it in less-than-complimentary terms. If you think about it, you will remember that Windows' native security structure has remained essentially unchanged since the release of Windows NT in 1993. At that time, Windows was gaining ground as a departmental or small business file and print server, and its security system worked well for those scenarios. Today, things are drastically different. Businesses and organizations need a lot more from Windows' security.

## Business drivers for security

Let's start by looking at some of the major requirements most organizations currently expect from their network operating system.

Perhaps the most important theme is that organizations have put their expensive, limited IT administration resources to use doing low-level identity administration. IT has to do everything in Active Directory: reset passwords, create user accounts, unlock accounts, manage group memberships, and much more.

Those are the time-consuming, day-to-day tasks. At a more general level, IT administrators have also become responsible for managing permissions in general, provisioning and de-provisioning users, ensuring that employees have access to only those resources that they require, and so on. IT is also held responsible for ensuring that the organization's change management processes and controls are followed, since Windows doesn't natively enforce those processes.

Manual processes do more than consume time: They also lead to errors and inconsistencies, which in turn lead to security breaches.

However, in many cases, IT doesn't even have the information needed to determine whether or not a change should be made! "You want me to add Bob to the Sales group? Does Bob belong in the Sales group?" IT usually ends up having to check with the owner of that group – perhaps the Sales Manager, in this case – to see if the change is approved. Either that, or IT starts to implement its own bureaucracy, complete with forms and approvals, that have to be completed before changes can be requested and implemented. The result of that is often more physical and electronic paperwork, which is rarely a complete solution to the actual problem.

Another key problem is the fact that Windows' security is largely managed by a set of manual processes. Administrators have to open dialog boxes, click through checkboxes, and so forth – there's very little in the way of automation. Even Windows PowerShell, Microsoft's effort at improving administrative automation, hasn't yet been equipped with capabilities to really automate permissions management. These manual processes do more than consume time, they also lead to errors and inconsistencies, which in turn lead to security breaches. In fact, most organizations probably have poorly managed security in at least a few places, due in large part to the fact that it's all done manually.

Today's businesses also need to comply with an increasing array of external security and compliance requirements, imposed both by legislation and by industry rules. You've seen the acronyms: HIPAA, SOX, GLB, PCI DSS, take your pick – and those are just efforts within the United States. Begin looking at all of the legislation and requirements implemented globally, and you have quite a lot of them to consider. However, almost all of the major security requirements for these efforts boil down to a few simple ideas:

- Keep track of who's accessing (and trying to access) sensitive resources
- Keep track of who has access to what
- Exercise control over the management of security principals

Unfortunately, some of these seem to be technically difficult, if not outright impossible. What resources does Bob have access to across the entire organization? It's impossible to say without manually checking every single resource to see if Bob, or a group he belongs to, shows up there.

Organizations also rely more and more on the information stored directly within Active Directory. Other applications authenticate from Active Directory and utilize information from Active Directory

in a variety of business processes. It's never been more important that the information in Active Directory is accurate, up-to-date, and consistent.

Again, most of the burden for that falls on IT administrators, who would rather do anything than spend time updating directory attributes. The manual processes used to keep Active Directory "up to date" actually lead to missing information, as well as inaccurate and inconsistent information – and those problems cascade through the business processes and applications that touch Active Directory. Organizations need to secure Active Directory, but there are many technical challenges to overcome.

## Technical challenges

The main technical challenge is that Windows' security architecture dates to an entirely different time, place, and set of assumptions. Even Active Directory, introduced six years into Windows Server's life, adopts many of those same assumptions and models.

First, administrators need to be less involved in manually securing Active Directory. At first look, this goal seems as if it could be easily implemented. After all, Active Directory enables organizations to delegate permissions within the directory, assigning permissions to employees other than IT administrators. A simple wizard, shown below, accomplishes the task quickly and effectively.

There are two real problems with this approach to delegation. The first problem is that delegation can really only follow the organizational unit (OU) structure of the directory itself. If you weren't careful about your OU design, or if you perhaps designed it solely for some other purpose – such as the application of Group Policy – then delegation may be tricky because you'll have to manually apply delegated permissions in a larger number of places. That means more places to manage granular permissions, more time spent managing them, and more likelihood for errors and inconsistencies, especially over time.

> Administrators need to be less involved in manually securing Active Directory.

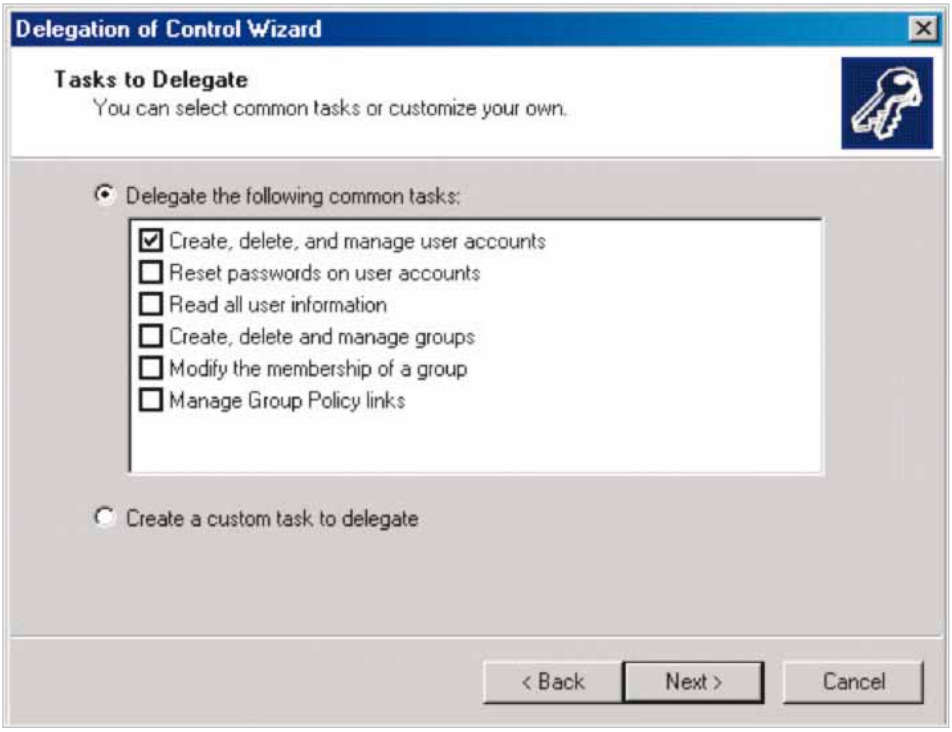


*Figure 1. Example of an Active Directory Wizard dialog box*

In a simple environment, Active Directory's groups can step in as a basic form of role-based management. But in larger environments, those groups simply can't function that way.

The second problem is that there's no wizard for removing or changing delegations. In other words, once you delegate permissions, you're back to manual management of access control lists (ACLs) to maintain that delegation. Again, that creates an increased risk of getting something wrong, forgetting to change something when necessary, and so on. It gets complicated quickly. There's not even an easy way to review the permissions you've delegated – you're stuck looking at dialog after dialog of discrete permission lists, like the one shown below in figure 2.

All of that could be mitigated if the directory offered some kind of role-based security mechanism – but it doesn't. In a simple environment, Active Directory's groups can step in as a basic form of role-based management. But in larger environments, those groups simply can't function that way. They can't work across untrusting forests or domains, for example. In other words, the groups themselves are simply too reliant upon the directory structure. The directory's groups are also overloaded in terms of functionality: They are used as email lists, they are used for low-level security permissions assignment,

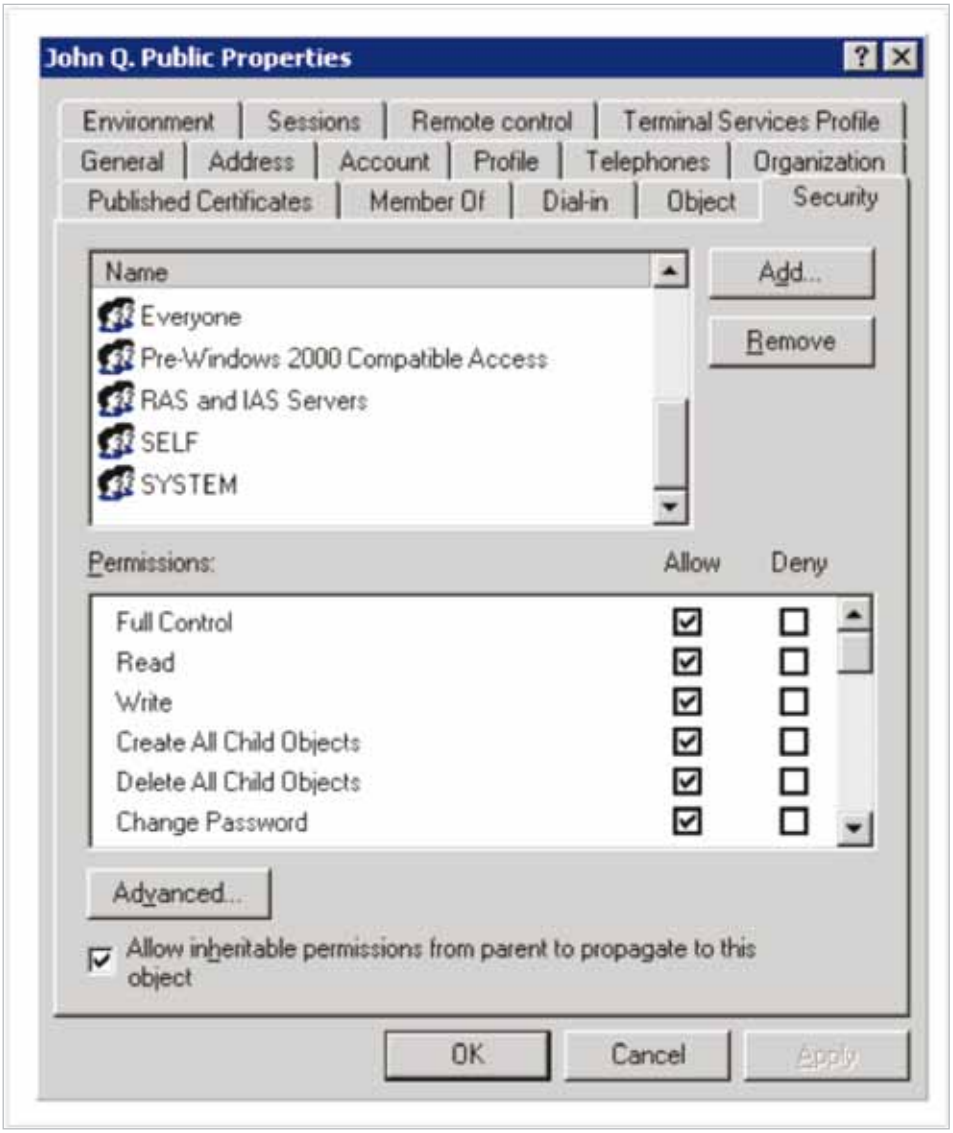


*Figure 2. Example of a discrete permission list dialog box*

and so on. Groups could actually be used to implement role-based access control, provided there is another layer above them to manage the roles and the groups themselves.

Note: If you'd like a brief explanation of how role-based access control typically differs from Windows' native ACL-based security, visit http://en.wikipedia.org/wiki/Role-based_access_control for a concise article and a list of other resources.

Role-based access control can help alleviate another challenge presented by Active Directory's native security: Permissions inventories. Today, finding out "who has access to what" is a manual and time-consuming process that's impractical. We simply don't have the time to scan through tens of thousands of ACLs to see where a user has permissions assigned. Administrators can't simply rely on inheritance of permissions when reviewing access controls, because that inheritance can be shut off at any point and direct permissions can be applied to an object. That means that every single object must be directly examined. Active Directory doesn't store permissions for any resources other than directory objects, and even those objects have their own discrete permissions sets (or at least have the potential to have them), and there's no place in the directory to get a complete view of all the objects to which a given user or group has been assigned permissions.

Another challenge is that Active Directory doesn't have any native means for enforcing consistency of data. Administrators are free to type whatever they like. Much of Active Directory's business logic is implemented in its tools, not in the directory itself; for example, while the directory won't allow you to enable a user account that doesn't have a policy-conforming password, it will allow you to create that user. The inability to create an account with a weak password comes from the tool, not the underlying directory. Once again, this behavior can easily lead to security errors and inconsistencies, as well as general inconsistencies in the information within the directory. Depending upon what native tools you're using to manage the directory, you may find different rules enforced in different ways.

Active Directory's native tools are the final technical challenge facing organizations and security goals. Simply put, Active Directory's native management tools are bare bones. They're one-size-fits-all. If you've delegated permissions to an assistant so that they can reset user passwords within their department, you'll either have to give them the complete Active Directory Users and Computers console, or you'll have to build a custom tool for them. We need tools that are a bit more task-specific, so that delegation can really, properly, be implemented across a variety of tasks.

Active Directory's native management tools are bare bones. Enterprises need tools that are a bit more task-specific, so that delegation can really, properly, be implemented across a variety of tasks.

## Dreaming of a better way

**So what do we need to fix the problem?** First, consider true role-based access control. As already stated, this needs to be implemented at a layer above the directory, and use the directory's native security architecture to actually implement the access controls.

Imagine a top-level software application that isn't tied to a specific domain, but can instead look at all of your domains and forests. Users are placed into roles that correspond to their job titles or work duties – and that could be done by Personnel or Human Resources, not by IT administrators. The software then implements that access control, ensuring that the user gets placed into the right groups to gain access to the exact resources they need. This can happen across Active Directory, across files and folders, in Exchange and SQL Server, anywhere. It isn't even necessarily limited to Windows: With the right software, that same role-based management would extend to non-Microsoft business

> Imagine a top-level software application that isn't tied to a specific domain, but can instead look at all of your domains and forests. Users are placed into roles that correspond to their job titles or work duties – and that could be done by Personnel or Human Resources, not by IT administrators.

applications as well. This approach also separates duties: IT administrators are no longer in charge of deciding who gets access to what, and they don't even need to be involved in the assignment of users to roles. When a user moves within the organization, they're simply moved to a different role. The role-based management system reassigns their permissions across the environment.

That automated, higher-level management of Active Directory's groups could also be dynamic and automatic. If a user's department is listed as "Sales" in the directory, then they're automatically added to specified groups – perhaps security groups, perhaps distribution lists, or perhaps both. That's a way of helping to automatically assign certain shared resources to a user, and have them un-assigned or re-assigned as the user's information changes.

But if we're going to add yet another dependency on accurate directory information, then we need consistency. Since there's no way within the directory to enforce data consistency, then we'll need to rely on external tools that can intercept directory changes, apply externally defined data rules, and "scrub" the data going into the directory. In some cases, that software might simply fill in defaults for missing attributes; in others, it might block the creation of objects that don't contain proper data. This capability would be coupled with front-end tools that understand the consistency rules, and could help apply them during data entry.

Finally, we need a solution that can provide a permissions inventory. It's likely that a role-based access control system would, because it would need that inventory in order to actually manage the various native ACLs on resources. Such a system would likely begin by inventorying existing permissions, and then letting administrators manage permissions. Permissions would be inventoried in a database, which could be easily queried and used as the basis for complex, almost-instantaneous reports. The software would then permit you to manage the database rather than managing permissions on resources; changes you made would be pushed out to the actual resources' ACLs. This has the effect of getting all the permissions information into a central repository, where we can work with it more effectively and efficiently.

## A blue-sky wish list

While we're dreaming up a new way of managing permissions, let's throw a few extra wish list items into the mix.

First, a centralized console would be nice. Something that can see all of the permissions in the environment, manage security principal creation, everything. It would also incorporate our data consistency rules, delegation rules, and so forth. Sort of an "Active Directory Users and Computers, Plus." We'd also need task-specific consoles, or at least ways of configuring that central console to make it task-specific based on an individual user's delegated permissions. If a user only has permission to unlock certain user accounts, then that user should only see those user accounts, not a bunch of other stuff that they shouldn't be messing with.

In this day and age, support for automation should come built right in, preferably in the form of support for Windows PowerShell. Being able to script repetitive tasks would make them more consistent, free up administrator time, and enable the creation of customized business processes that a solution's vendor didn't think of.

## Okay – when can we have it?

Solutions exist today that offer these capabilities– and in some cases a great deal more. They differ in key ways, such as in their ability to scale to different-sized environments, and their ability to support specific business scenarios. But role-based access control, automated delegation of permissions, centralized permissions reporting, data consistency

rules, it all exists. It's time for you to start exploring, and to find the solution that meets your organization's needs.

## About the author

Don Jones is a co-founder of Concentrated Technology (Concentrated Tech.com). His consulting practice specializes in making the connection between technology and business, helping businesses realize more value from their IT investment, and helping IT align more closely to business needs and values.

Don is the author of more than 30 books on information technology. He has been an IT journalist for more than eight years and is currently a contributing editor for Microsoft *TechNet Magazine*. Don is a sought-after speaker at industry conferences and symposia.

## About Dell

Dell Inc. (NASDAQ: DELL) listens to customers and delivers worldwide innovative technology, business solutions and services they trust and value. For more information, visit www.dell.com.

If you have any questions regarding your potential use of this material, contact:

**Dell Software**
5 Polaris Way
Aliso Viejo, CA 92656
www.dell.com
Refer to our Web site for regional and international office information.