**[SQL University: Authentication and Authorization](http://www.sqlservercentral.com/blogs/brian_kelley/2009/10/25/sql-university-authentication-and-authorization/)**

SQL Server 2008

Welcome back to class, folks, here at [SQL University](http://sqlchicken.com/sql-university/). This week we're going to take a look at the basics of security for SQL Server. Since this is an introductory class, we're going to focus primarily on SQL Server itself, only delving into the operating system when we have to do so. So let's start with first steps. In security there are usually three As we're concerned with:

* **Authentication** - Can you prove you are who you say you are?
* **Authorization** - Okay, so you've proven it. Now, what do you have access to?
* **Auditing** - Can we track what you and others are doing?

While auditing is important, that's a more advanced topic, so we'll stick to the first two As: authentication and authorization. If you remember back to your [first week of class](http://sqlchicken.com/2009/09/sql-university-basic-tools/), we briefly talked about authentication. That's what we're gong to focus on today. Authorization will be for Wednesday's class. And for Friday we'll look at both in a bit more detail. Now, onto today's material. One of the things we discussed is that either SQL Server can handle the authentication or it can pass it off to Windows. Windows authentication can also be called integrated security. Since Microsoft recommends using Windows authentication, let's talk about why.

**Windows Authentication - Ease of Use for the End User:**

Imagine for a second that you're the manager of an apartment complex and let's say you've got 250 different apartments. Now, you've got two choices on how to handle *your* access to these apartments you're responsible for. Either you can have 250 different keys or you can have one master key. Which would you choose? If it were me, I'd want one master key for me. Sure, the system needs to be able to handle issuing 250 different keys so one apartment dweller can't get into the apartment of another, but for me and my maintenance staff, I need the master key. Otherwise I could sit there forever trying to figure out how to get into the one apartment where the toilet is overflowing and flooding the apartment below. Not a good situation if I have to sort through 250 different keys, right?

Well, imagine your internal network is like a huge apartment complex. And let's say you have 250 different systems. If your end users have to enter a username/password combination for every system, your users would quickly grow upset. Even if we're talking 5 to 10 systems, you're going to hear grumbling. If you make it worse and they have to have different passwords for every system, you've really got trouble. Sooner or later they're going to do things like keep a sticky note on their monitor with all of their combinations recorded. But if you can give them one key to access all of the systems, they'll be very happy with you. And that's one reason to use Windows authentication. Your users already have to use a username/password combination to log onto your Windows domain. If we can take those credentials and re-use them, they'll be very happy. It also means they are less likely to do something like that sticky note trick. I did say less likely. Get used to it, because no matter how easy you make it on your users (unless you take passwords away altogether), someone is going to do it. In times like those, take the time to show 'em right. Help them understand why it's dangerous. And if you don't get through the first time or the hundred and first time, keep at it. It's important.

**Windows Authentication - "One Stop Shop" Lockdown:**

But you know, it's not just about ease of use for our end users. Imagine you went the 250 key method. Now you were trying to be smart and you got [keys marked Do Not Duplicate](http://en.wikipedia.org/wiki/Key_(lock)). While this isn't an effective way to ensure they keys aren't duplicated, let's say, for example's sake that it is. And one of your maintenance folks was caught doing something they ought not be doing (like stealing from one of the apartments). You now need to get all the keys back from that maintenance person. And he hands you 250 keys which look right. The question is, "How do you know?" The only way you're going to be able to tell is to go through and try each key in the locks until you're sure you have them all. After all, there's nothing stopping the maintenance person for substituting several keys that look authentic but aren't. And then your maintenance guy still has access to some of the apartments in the building. So unless you take the time to immediately verify all the keys, you have a problem. The same thing is true if you have folks with different username/password combination all over the place. You might be able to turn them off in one place, but unless you go system by system, you've got the same problem if someone's access needs to be turned off.

Now let's go back to that master key idea. If you only issued them master keys, and you have some way of verifying they weren't able to duplicate the master keys, once you get it back from the maintenance person you're letting go, it's a simple matter to try it in a handful of apartment locks to verify it's real. And then you can kick the guy to the curb. Well, if Windows user accounts are being used, specifically domain user accounts, you have the same idea. You've got an employee who has been doing something they ought not do, and it's time to take away their access as HR processes them out the door. If all access is through the domain, then as soon as I lock down that person's domain user account, I have confiscated the master key. They aren't going to get in anywhere. And that's what we want.

**Windows Authentication - Security Forces at the Gate:**

As a member of the United States Air Force, one of the job types I came into contact with every day were what are known as Security Forces (like the guy to our left). These are people whose job it is to provide base and flightline protection. Because they did this job, I could focus on my job: computers. Now, they weren't just base and flightline defense, either. They also were law enforcement personnel. If you were speeding on base, it would be these guys who would pull you over in a heartbeat. And while I know that some civilian law enforcement personnel have the discretion to give a warning instead of a ticket, I never met one of these guys who did. They took their job seriously, whether it was pulling people over for speeding, preventing looting of a base devastated by a hurricane (Keesler Air Force base after Hurricane Katrina), or trying to ensure no one is trying to get on base that shouldn't be there. Well trained and well motivated, they will always do a better job than someone like me, because they are trained for it, constantly do it, and constantly retrain on it.

And that describes Windows authentication. The Windows operating system already has authentication functionality built-in. It has to do so. And if you're in a domain environment, there are special computers, called Domain Controllers, where authentication actually takes place. Those systems tend to be treated more security than regular servers, even SQL Servers. And therefore, if the username/password combinations are going to be stored on the most secure systems in the domain, that would be on the domain controllers (*as a side note, a hash is what is actually stored there, not the password itself*). If you let Windows do the authentication, then SQL Server doesn't have to store anything related to a username/password combination. And it's up to Windows to check the security. In other words, SQL Server is handing off this sort of authentication to processes which do it constantly. That's good, because it also means that if you're a DBA and your organization is large enough to have someone else managing the domain security, then you can leave the authentication to them, which is something they're doing constantly. And you have the option of focusing on what you do best: SQL Server.

**So Why SQL Server Authentication?**

Because you have to. Yup, I said it. The classic example is a third-party application that only supports SQL Server based authentication. You can pull out your hair and gnash your teeth all you want, but if the vendor isn't going to change the product and you need their product, you're stuck. And you'll have to allow SQL Server to manage the authentication. That's not as bad as some may make it seem. SQL Server does a secure job of things. However, you lose the advantages I've mentioned above.

Another classic example is when you have a system that isn't on the domain that needs to talk to the SQL Server. For instance, you have linux servers that you want to connect back with and in that case, SQL Server authentication is the most straight-forward way to go. Another classic example along these lines is a web server that sits in the DMZ. Not the one between North and South Korea, mind you, but where you have your Internet facing systems located. These systems are partially protected from an Internet attack and your internal network is protected from most attacks that could originate from these DMZ servers, should those servers be compromised. So it's a buffer zone between your internal environment and the Internet. In this case, it can be considered too risky to put said web server in the DMZ on the internal Windows domain. If that's the case, you can't use Windows authentication to connect. So you've got to use SQL Server authentication.

**Some Practical Instruction**

Okay, let's depart from the theory and look at some of this stuff in your SQL Server environment. Here are some videos which explore the topic of authentication a bit further. They're all short, five minutes or less, and should give you a basic foundation on how folks come into your SQL Server:

* [SQL Server Authentication Modes](http://www.sqlshare.com/sql-server-authentication-modes_582.aspx)
* [SQL Server Logins vs. Windows Logins](http://www.sqlshare.com/sql-server-login-vs-windows-authenticated-logins_583.aspx)
* [Adding a SQL Login to SQL Server (GUI only)](http://www.sqlshare.com/adding-a-login-to-sql-server_559.aspx)
* [Adding a Windows-based Login to SQL Server (GUI only)](http://www.sqlshare.com/adding-a-windows-based-login-to-sql-server_560.aspx)
* [Disabling SQL Server Logins in SQL Server 2005 (also applicable to SQL Server 2008)](http://www.sqlshare.com/disabling-sql-server-logins-in-sql-server-2005_571.aspx)